



# **IT Asset Management 2024 R2.3**

Inventory Adapters and Connectors  
Reference

# Legal Information

IT Asset Management (Cloud)

**Document Name:** IT Asset Management Inventory Adapters and Connectors Reference version 2024 R2.3 (for cloud implementation)

**Part Number:** FMS-23.3.0-AR01

**Product Release Date:** March 26, 2025

## Copyright Notice

Copyright © 2025 Flexera.

This publication contains proprietary and confidential technology, information and creative works owned by Flexera and its licensors, if any. Any use, copying, publication, distribution, display, modification, or transmission of such publication in whole or in part in any form or by any means without the prior express written permission of Flexera is strictly prohibited. Except where expressly provided by Flexera in writing, possession of this publication shall not be construed to confer any license or rights under any Flexera intellectual property rights, whether by estoppel, implication, or otherwise.

All copies of the technology and related information, if allowed by Flexera, must display this notice of copyright and ownership in full.

IT Asset Management incorporates software developed by others and redistributed according to license agreements. Copyright notices and licenses for this externally-developed software are provided in the link below.

## Intellectual Property

For a list of trademarks and patents that are owned by Flexera, see <http://www.flexera.com/intellectual-property>. All other brand and product names mentioned in Flexera products, product documentation, and marketing materials are the trademarks and registered trademarks of their respective owners.

## Restricted Rights Legend

The Software is commercial computer software. If the user or licensee of the Software is an agency, department, or other entity of the United States Government, the use, duplication, reproduction, release, modification, disclosure, or transfer of the Software, or any related documentation of any kind, including technical data and manuals, is restricted by a license agreement or by the terms of this Agreement in accordance with Federal Acquisition Regulation 12.212 for civilian purposes and Defense Federal Acquisition Regulation Supplement 227.7202 for military purposes. The Software was developed fully at private expense. All other use is prohibited.

# IT Asset Management Inventory Adapters and Connectors Reference (Cloud Edition)

## IT Asset Management (Cloud)

IT Asset Management relies on software and hardware inventory collected from the computers in your computing estate to calculate what licenses are required. While the system includes a full inventory-gathering capacity, it also allows you to import inventory collected by other tools you may already have.

As well, the system uses a considerable amount of business-related data, including organizational structure, and records of purchases, to correctly track your existing license entitlements. This business data can also be imported from other sources in your enterprise. However, business-related data is not the subject of this document: instead, see *Using FlexNet Business Adapters*, which covers both the creation of business adapters and import of data through them.

This document, therefore, focuses on the collection of software and hardware inventory.

The data collected by the 'third party' tools usually needs to be rationalized in different ways, and mapped into the data fields within the IT Asset Management database. The interfaces that allow this mapping are called *adapters*, structured as XML files. On deploying a beacon, several of the adapters are downloaded by default and automatically available without any action from the user (for example SCCM, ILMT, BigFix, AWS, and Azure). The rest of the adapters are available for download from the Flexera Product and License Center as a zip file. You can also build custom adapters for inventory using the Inventory Adapter Studio, supplied with the product.

Some other inventory systems allow a rather simpler interface, where the preparation of XML-based adapters is not necessary. Instead, it is enough to declare a connection to these systems, most often from an inventory beacon. Some such systems are databases for third-party tools, where a direct connection may be possible, given a suitable connection string. The incoming data can then be mapped automatically into the operations databases for IT Asset Management, most often directly into the compliance database. Since the preparation of XML-based adapter files is not required in these cases, they are referred to simply as *connectors*.

This document gives priority to several of the standard adapters available for IT Asset Management, because these are the more complex cases that may require special installation, or custom XML files and the like. Over time, coverage of connectors will also increase.

Before diving into the details of the various connectors and adapters, some useful tables compare the kinds of information available from various data sources (see [Inventory Comparison Matrix](#)).

Following the details about available adapters and connectors, the Inventory Adapter Studio is documented. This is for advanced and experienced developers who wish to create custom adapters (see [Inventory Adapter Studio](#)).

Between these 'bookends', the following adapters and connectors are documented, in this (alphabetical) order:

- ADDM — now rebranded as BMC Discovery (see below)
- AWS EC2 — see [AWS Adapter](#)
- BMC Discovery — see [BMC Helix Discovery Adapter](#)

- Citrix Cloud - see [Citrix Cloud Adapter](#)
- Citrix XenApp server adapter — see [Citrix XenApp Server Adapter](#)
- Data Platform — see [Data Platform Integration](#)
- Flexera One SaaS Management — see [Flexera SaaS Manager Adapter](#)
- HP DDMI — see [HP DDMI Adapter](#)
- HPE Universal Discovery — see [HPE Universal Discovery Adapter](#)
- Microsoft App-V — see [Microsoft App-V Server Adapter](#)
- Microsoft Azure — see [Microsoft Azure Adapter](#)
- Microsoft Intune — see [Microsoft Intune Adapter](#)
- Microsoft Office 365 — see [Microsoft 365 Adapter](#)
- Oracle Cloud Infrastructure — see [Oracle Cloud Infrastructure \(OCI\) Adapter \(beta\)](#)
- Oracle Enterprise Manager — see [Oracle Enterprise Manager Adapter](#)\*
- Salesforce — see [Salesforce Adapter](#)
- ServiceNow — see [ServiceNow Integration with IT Asset Management](#)
- Tanium Connector — see [Tanium Adapter](#)
- VMware Horizon — see [VMware Horizon Adapter](#).

# Contents

<b>1. Inventory Comparison Matrix .....</b>	<b>16</b>
<b>Part I. AWS Adapter .....</b>	<b>21</b>
<b>1. Prerequisites and Setting Up .....</b>	<b>23</b>
<b>2. Thinking about Inventory and Licensing.....</b>	<b>25</b>
Collecting Inventory from Instances .....	25
BYOL Licensing Considerations .....	29
<b>3. Images for Different Kinds of Instances .....</b>	<b>32</b>
Configuring an AMI for Short-Lived Instances .....	32
Configuring an AMI for Longer-Life Instances.....	39
<b>4. Appendices: Technical Data .....</b>	<b>45</b>
Appendix 1: Cmdlets in the AWS adapter .....	45
Appendix 2: Data Imported by Amazon connector .....	46
Appendix 3: Enhanced Inventory Gathered by Agent .....	49
<b>Part II. BMC Helix Discovery Adapter .....</b>	<b>51</b>
<b>1. Choosing a Configuration.....</b>	<b>52</b>
How the Adapter Works .....	52
Database Table Creation.....	53
The Adapter Executable.....	54
How the Connector Works, and Prerequisites .....	58
The FlexNet inventory agent.....	59
<b>2. Configuring BMC Discovery .....</b>	<b>60</b>
Optional Patterns .....	60
Installing Optional Patterns.....	63
Enabling Optional Patterns .....	65
Rediscovering Affected Computers .....	65
Account Configuration for Adapter .....	66
<b>3. Adapter Installation and Configuration .....</b>	<b>69</b>
Choosing a Staging Server .....	69
Download the Adapter.....	70

<b>Creating the Staging Database Tables .....</b>	<b>71</b>
<b>Installing and Configuring the Staging Tool .....</b>	<b>72</b>
<b>Installing the FlexNet inventory agent .....</b>	<b>74</b>
<b>Validation and Operation.....</b>	<b>75</b>
<b>4. Connector Configuration.....</b>	<b>76</b>
<b>5. Known Issues.....</b>	<b>80</b>
<b>6. Appendix A: Details of Patterns.....</b>	<b>81</b>
<b>Overview of Patterns.....</b>	<b>81</b>
<b>FileEvidence .....</b>	<b>82</b>
To configure the FileEvidence pattern .....	84
<b>InstallAnywhereEvidence .....</b>	<b>85</b>
<b>InstallShieldMultiplatformEvidence .....</b>	<b>85</b>
<b>UnixHardwareData .....</b>	<b>86</b>
<b>WindowsLastLoggedOnUser .....</b>	<b>89</b>
<b>7. Appendix B: Data Mappings .....</b>	<b>91</b>
<b>Source to Staging.....</b>	<b>92</b>
ADDMNetworkDevices_ci (Staging Table).....	92
ADDMPrinters_ci (Staging Table) .....	93
ADDMVersion_ci (Staging Table) .....	94
Cluster_ci (Staging Table) .....	94
ClusterHost_ci (Staging Table) .....	95
CPUInformationDetail_ci (Staging Table) .....	95
DiscoveredInstalledPackages_ci (Staging Table) .....	97
DiscoveredPackages_ci (Staging Table).....	98
DiscoveredService_ci (Staging Table) .....	99
DiscoveredVirtualMachine_ci (Staging Table) .....	99
FileEvidenceDetail_ci (Staging Table) .....	100
FileSystem_ci (Staging Table).....	101
HardwareEvidenceDetail_ci (Staging Table) .....	102
Host_ci (Staging Table) .....	104
HostContainerProcessorInfo_ci (Staging Table) .....	107
HostDetail_ci (Staging Table) .....	108
HostInfo_ci (Staging Table) .....	108
InstallerEvidenceDetail_ci (Staging Table) .....	110
LastLoggedOnUserDetail_ci (Staging Table) .....	110

NetworkInterface_ci (Staging Table).....	111
SoftwareInstance_ci (Staging Table).....	112
SoftwareInstanceVirtualMachine_ci (Staging Table).....	112
<b>Staging to IT Asset Management.....</b>	<b>114</b>
Inventory Device (Computer) .....	114
Installer Evidence.....	119
File Evidence .....	121
WMI Evidence .....	123
<b>Part III. Citrix Cloud Adapter.....</b>	<b>125</b>
<b>1. Prerequisites .....</b>	<b>127</b>
<b>2. Architecture and Operation .....</b>	<b>128</b>
<b>3. Creating the Citrix Cloud Connector Connection .....</b>	<b>130</b>
<b>Part IV. Citrix XenApp Server Adapter .....</b>	<b>134</b>
<b>1. Architecture, Operations and Prerequisites .....</b>	<b>136</b>
Architecture and Operation .....	136
Prerequisites.....	144
<b>2. Setting Up the XenApp Server Adapter .....</b>	<b>145</b>
Creating the Staging Database .....	145
Installing the XenApp Server Agent .....	147
Create a Scheduled Task.....	148
Create Connections for Data Upload .....	150
<b>3. Command-Line Options .....</b>	<b>155</b>
XenApp Server Agent Command Line Options.....	155
<b>4. Validation, Troubleshooting, and Limitations .....</b>	<b>159</b>
Validation and Problem Solving .....	159
Limitations.....	160
<b>5. Database Impacts.....</b>	<b>162</b>
Affected Database Tables.....	162
<b>Part V. Data Platform Integration .....</b>	<b>164</b>
<b>1. Integration with Data Platform v5.....</b>	<b>165</b>
<b>2. Configuring the Data Platform Connector.....</b>	<b>169</b>
<b>3. Data Mappings, Gaps, and Impacts.....</b>	<b>172</b>

Comparison of Sources .....	173
Imported Installer Evidence .....	175
Imported Computers (Inventory Devices) .....	177
Imported VMs and Hosts .....	186
Imported Installation Records .....	189
Imported Users .....	190
Imported Software Usage .....	192
<b>Part VI. Flexera SaaS Manager Adapter .....</b>	<b>195</b>
1. The Flexera SaaS Manager Connector .....	196
2. Configuring SaaS Manager Connector .....	198
3. Deprecating Other Connectors .....	200
4. Imported Data Mapping .....	204
<b>Part VII. HP DDMI Adapter .....</b>	<b>210</b>
1. Purpose and Architecture of the HP DDMI Adapter .....	211
2. Installation and Configuration .....	213
Download Adapter Tools Archive .....	213
Creating the Staging Database .....	214
Configuring the FlexNet Agent for HP DDMI .....	217
Configuring Upload and Import Connection .....	219
<b>Part VIII. HPE Universal Discovery Adapter .....</b>	<b>221</b>
1. Selecting a Configuration .....	222
Architecture and Working of the HPE Universal Discovery Adapter .....	222
The Adapter Executable .....	223
2. Installation and Configuration .....	225
Download Adapter Tools Archive .....	225
Selecting a Staging Server .....	226
Creating the Staging Database Tables .....	226
Configuring HPE Universal Discovery System .....	227
Installing and Configuring the Staging Tool .....	228
3. Operation and Validation .....	230
HPE-UD Adapter Operation .....	230
Validating the HPE-UD Adapter .....	230
<b>Part IX. Inventory Adapter Studio .....</b>	<b>232</b>



<b>1. What Is Inventory Adapter Studio?</b>	<b>233</b>
<b>2. Cautions, Prerequisites, and References</b>	<b>234</b>
<b>3. The Inventory Adapter Studio Interface</b>	<b>236</b>
Toolbar	237
Step Explorer	239
Edit panel	241
<b>4. Installing Inventory Adapter Studio</b>	<b>246</b>
<b>5. Understanding Inventory Adapters</b>	<b>247</b>
The Architecture of Compliance Importer	247
Structure of an Inventory Adapter	248
Structure of Templates for Inventory Adapters	252
<b>6. Creating a New Adapter</b>	<b>253</b>
<b>7. Editing an Existing Adapter or Template</b>	<b>255</b>
<b>8. To Create a Source Connection</b>	<b>257</b>
<b>9. Overview: Process for Developing an Inventory Adapter</b>	<b>260</b>
Adding a New Step to an Inventory Adapter	261
Removing a Step from an Inventory Adapter	261
Reordering Steps in an Inventory Adapter	262
<b>10. Disconnected Mode</b>	<b>263</b>
<b>11. Tips for Editing an Adapter</b>	<b>264</b>
<b>12. To Save an Adapter</b>	<b>268</b>
<b>13. Testing an Adapter</b>	<b>269</b>
To Run a Full Import	269
To Diagnose Readers for Your Adapter	270
Diagnosing Writers for Your Adapter	270
<b>14. Publishing Your Adapter</b>	<b>272</b>
<b>15. Inventory Adapter Object Model</b>	<b>274</b>
Inventory Object: AccessingDevice	274
Inventory Object: AccessingUser	275
Inventory Object: ActiveDirectoryComputer	275
Inventory Object: ActiveDirectoryDomain	276
Inventory Object: ActiveDirectoryExternalMember	277

Inventory Object: ActiveDirectoryGroup .....	277
Inventory Object: ActiveDirectoryMember .....	278
Inventory Object: ActiveDirectoryUser .....	278
Inventory Object: ActiveSyncDevice .....	279
Inventory Object: ClientAccessEvidence .....	280
Inventory Object: ClientAccessEvidenceMapping.....	281
Inventory Object: ClientAccessedAccessEvidence .....	281
Inventory Object: ClientAccessedAccessOccurrence .....	283
Inventory Object: Cluster .....	284
Inventory Object: ClusterGroup .....	285
Inventory Object: ClusterGroupMember .....	286
Inventory Object: ClusterHostAffinityRule .....	286
Inventory Object: ClusterNode.....	287
Inventory Object: Computer .....	288
Inventory Object: ComputerCustomProperty .....	293
Inventory Object: ConsolidatedAccessEvidence .....	293
Inventory Object: ConsolidatedCluster .....	297
Inventory Object: ConsolidatedClusterGroup .....	298
Inventory Object: ConsolidatedClusterHostAffinityRule .....	299
Inventory Object: ConsolidatedComputer .....	299
Inventory Object: ConsolidatedFileEvidence .....	309
Inventory Object: ConsolidatedInstallerEvidence .....	312
Inventory Object: ConsolidatedOracleDatabaseUser.....	316
Inventory Object: ConsolidatedRemoteAccessFile .....	318
Inventory Object: ConsolidatedRemoteAccessInstaller .....	321
Inventory Object: ConsolidatedVMPool.....	322
Inventory Object: ConsolidatedWMIEvidence .....	323
Inventory Object: Domain .....	324
Inventory Object: EvidenceAttribute .....	325
Inventory Object: FileEvidence .....	326
Inventory Object: ILMTPVUCounts .....	327
Inventory Object: InstalledFileEvidence.....	329
Inventory Object: InstalledFileEvidenceUsage.....	329
Inventory Object: InstalledInstallerEvidence.....	331
Inventory Object: InstalledInstallerEvidenceAttribute .....	333

Inventory Object: InstalledInstallerEvidenceUsage .....	334
Inventory Object: InstalledWMIEvidence .....	336
Inventory Object: InstallerEvidence .....	338
Inventory Object: InstallerEvidenceRepackageMapping .....	339
Inventory Object: Instance .....	340
Inventory Object: InstanceUser .....	342
Inventory Object: LicenseUser .....	343
Inventory Object: RelatedInstalledInstallerEvidence .....	344
Inventory Object: RemoteUserToApplicationAccess .....	346
Inventory Object: Site .....	348
Inventory Object: SiteSubnet .....	348
Inventory Object: SoftwareLicense .....	349
Inventory Object: SoftwareLicenseAllocation .....	350
Inventory Object: User .....	351
Inventory Object: VDI .....	352
Inventory Object: VDI Template .....	354
Inventory Object: VDIUser .....	356
Inventory Object: VMHostManagedBySoftware .....	357
Inventory Object: VMPool .....	357
Inventory Object: VirtualMachine .....	359
Inventory Object: WMIEvidence .....	361
<b>Part X. Microsoft 365 Adapter .....</b>	<b>363</b>
<b>1. Microsoft 365 License Management Considerations .....</b>	<b>364</b>
<b>2. Managing Microsoft 365 Licenses through IT Asset Management .....</b>	<b>367</b>
<b>3. Connecting to Microsoft 365 .....</b>	<b>370</b>
<b>Prerequisites and Configuration Considerations .....</b>	<b>371</b>
<b>Creating Connections to Microsoft 365 .....</b>	<b>374</b>
Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365 .....	375
Registering an App to Connect to Microsoft 365 Using the Azure Portal .....	377
Configuring Token Lifetimes in Azure Active Directory .....	381
<b>Creating Connections Using the Microsoft Office 365 (Deprecated) Connector .....</b>	<b>384</b>
<b>Troubleshooting Imports from Microsoft 365 .....</b>	<b>386</b>
Troubleshooting Microsoft 365 Connector Imports from Microsoft 365 .....	386
Troubleshooting Microsoft Office 365 (Deprecated) Connector Imports from Office 365 .....	387

4. Migrating to a New Microsoft Connector .....	390
<b>Part XI. Microsoft App-V Server Adapter .....</b>	<b>391</b>
1. Architecture, Components, and Prerequisites .....	392
Architecture and Operation for App-V 4.6 .....	392
Architecture and Operation for App-V 5 and Later .....	394
2. Set Up and Operations .....	399
Obtaining (and Deploying) the Adapter Components .....	399
Command Line for PowerShell Script .....	402
File Format for .raa .....	404
Configuring the Adapter .....	406
Import Evidence and Recognize Applications .....	411
3. Issues and Limitations .....	415
Limitations .....	415
Investigating Issues .....	416
Known Issues .....	419
4. Data mapping .....	421
App-V Release 4.6 Data Transfers .....	421
App-V Release 5.0 (and Later) Data Transfers .....	422
<b>Part XII. Microsoft Azure Adapter .....</b>	<b>425</b>
1. Prerequisites and Setting Up .....	427
2. Certificate-based Authentication Prerequisites and Setting Up .....	430
3. Thinking about Inventory and Licensing .....	433
Collecting Inventory from Instances .....	433
BYOL Licensing Considerations .....	437
4. Configuring an Azure VM Image for Instances .....	440
5. Appendices: Technical Data .....	446
Appendix 1: Cmdlets in Azure PowerShell .....	446
Appendix 2: Data Imported by Microsoft Azure Connector .....	447
Appendix 3: Enhanced Inventory Gathered by Agent .....	448
<b>Part XIII. Microsoft Intune Adapter .....</b>	<b>450</b>
1. What data is retrieved .....	452
2. Register an Application in Azure Active Directory .....	455

3. Creating the Microsoft Intune Connection .....	457
Part XIV. Oracle Cloud Infrastructure (OCI) Adapter (beta) .....	460
1. Prerequisites for the OCI Adapter .....	461
Part XV. Oracle Enterprise Manager Adapter .....	462
1. Understanding the Oracle Enterprise Manager Adapter .....	463
How the Adapter Assists in Inventory Gathering .....	463
Prerequisites for the OEM Adapter .....	465
Components .....	465
Download Adapter Tools Archive .....	466
2. Installing the Adapter, and More .....	467
Installing the OEM adapter .....	467
Grant Permissions to Account .....	471
Other Setup Activities .....	472
Inventory-Gathering Accounts on Oracle Servers .....	473
Save Inventory Account in Password Manager .....	476
Assign Beacon to Subnet .....	476
Configure Collection of Oracle Inventory .....	477
3. Modifying the Adapter .....	482
Reconfiguring the OEM Adapter .....	482
Updating Connection Details .....	483
Configure Data Staging .....	484
Managing Email Alerts .....	485
Configure Logging .....	486
Part XVI. Salesforce Adapter .....	487
1. Salesforce License Considerations .....	488
2. Viewing Salesforce License Information with IT Asset Management .....	489
3. Connecting to the Salesforce Online Service .....	491
Managing Connections to Salesforce.com .....	491
Part XVII. ServiceNow Integration with IT Asset Management .....	496
1. Key Concepts .....	498
Data Types that Can Be Merged .....	498
How Data Is Merged .....	499
Source of Truth .....	499

Where to View Merged Data .....	499
ServiceNow Computer and Application Records .....	500
<b>2. Architecture, Components, and Prerequisites .....</b>	<b>503</b>
Architecture .....	503
Prerequisites .....	504
Download Adapter Tools Archive .....	505
<b>3. Installation and Configuration .....</b>	<b>507</b>
Setting Up the Integration .....	507
Installing the Flexera Integration Application from the ServiceNow Store .....	507
Creating a ServiceNow Integration User .....	508
Setting Up Data Flows from IT Asset Management to ServiceNow .....	509
Configuring ServiceNow for Import .....	509
Configuring IT Asset Management for Export .....	510
Setting Up Data Flows from ServiceNow to IT Asset Management .....	512
Setting Up a MID Server .....	513
Configuring ServiceNow for Export .....	513
Configuring FlexNet Beacon for Import .....	515
<b>4. Operational Details .....</b>	<b>518</b>
Process for Exports from IT Asset Management to ServiceNow .....	518
Import Runs Columns .....	520
Import Transactions Columns .....	521
Transform Maps for ServiceNow Integration .....	522
Process for Exports from ServiceNow to IT Asset Management .....	530
Properties for Export Columns .....	531
Business Adapter Mappings .....	532
<b>5. Appendices .....</b>	<b>535</b>
Integration Properties .....	535
Additional ServiceNow Indexes for Performance .....	537
Removing a Legacy Integration Application .....	538
Configuring the Software Asset Management Foundation Plugin .....	539
Deleting Records from ServiceNow .....	540
<b>Part XVIII. ServiceNow Inventory Adapter .....</b>	<b>542</b>
<b>1. ServiceNow Inventory Connector .....</b>	<b>544</b>
Prerequisites .....	544

Purpose and Architecture .....	545
<b>2. Setting up ServiceNow for the Inventory Connector .....</b>	<b>547</b>
Setting up the Integration Application .....	547
Creating a ServiceNow Integration User .....	548
Setting Up a MID Server .....	549
Scheduling Inventory Exports .....	549
<b>3. Configuring the Inventory Connector on the Beacon .....</b>	<b>552</b>
Setting the Import Schedule .....	552
Setting Connector Details, and Validation .....	553
<b>4. Appendices: Data Flows .....</b>	<b>556</b>
Imported Installer Evidence (with Discovery only) .....	557
Imported Computer Property Mapping .....	558
Imported Installer Evidence (with SAM) .....	559
<b>Part XIX. Tanium Adapter .....</b>	<b>560</b>
<b>1. Prerequisites .....</b>	<b>561</b>
FlexNet Beacon Server Prerequisites .....	561
Tanium Server Prerequisites .....	562
Creating a Flexera SQL Edition view within Tanium Asset .....	562
<b>2. Configure FlexNet Beacon to Connect with Tanium .....</b>	<b>564</b>
<b>3. Validating Imports to the Inventory Beacon .....</b>	<b>567</b>
<b>4. Tanium Adapter Architecture .....</b>	<b>569</b>
<b>5. Investigating Tanium Connection Errors .....</b>	<b>571</b>
<b>6. Appendix: Technical Data .....</b>	<b>573</b>
<b>Part XX. VMware Horizon Adapter .....</b>	<b>576</b>
<b>1. Architecture and Operation .....</b>	<b>578</b>
<b>2. Prerequisites .....</b>	<b>581</b>
<b>3. Creating the VMware Horizon Connection .....</b>	<b>582</b>

# Inventory Comparison Matrix

## IT Asset Management (Cloud)

Not all inventory sources are created equal. Some sources do better at collecting details necessary for licensing calculations than others. The following tables let you quickly assess what data may be missing from various sources.

Strictly speaking, not all the products listed in these tables are inventory sources (particularly in the later tables). However, they are all *data* sources that can provide data for IT Asset Management to use in license compliance calculations.



**Note:** For supported versions of these products, please review the IT Asset Management System Requirements and Compatibility on [docs.flexera.com](https://docs.flexera.com) for the current version of IT Asset Management.

## Inventory Sources

Information Collected per Product									
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB
			Installer	File		Server	Application		
FlexNet Inventory Agent (previously Managesoft)									
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Flexera Normalize (Data Platform) <sup>1</sup>									
Yes	Yes	No	No	No	No	No	No	No	No
BigFix platform (previously Tivoli Endpoint Manager and IBM BigFix) <sup>2</sup>									
Yes	Yes	Yes	Yes	No	No	No	No	No	No
BMC Helix Discovery and older BMC On-Prem products									
Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No
BMC BladeLogic Client Automation (previously Marimba) <sup>3</sup>									
Yes	Yes	Yes	Yes	Yes	Yes	Partial	No	No	No



Information Collected per Product									
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB
			Installer	File		Server	Application		
HP Discovery and Dependency Mapping Inventory (DDMI)									
Yes	Yes	Yes	Yes	Yes	No	No	No	No	No
HPE Universal Discovery (HP-UD)									
Yes	Yes	Yes	Yes	Yes	No	No	No	No	No
IBM License Metric Tool (ILMT) on Microsoft SQL Server									
Yes	No	Yes	Yes	No	No	Yes	No	Yes	No
IBM License Metric Tool (ILMT) on IBM DB2									
Yes	No	Yes	Yes	No	No	Yes	No	Yes	No
Microsoft Exchange ActiveSync <sup>4</sup>									
Yes	Yes	No	No	No	No	No	No	No	No
Microsoft Endpoint Configuration Manager (previously Microsoft SCCM or SMS) <sup>5</sup>									
Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No
Microsoft Intune <sup>6</sup>									
Yes	Yes	No	Yes	No	No	No	No	No	No
Symantec IT Management Suite (formerly Altiris) <sup>7</sup>									
Yes	Yes	Yes	Yes	Yes	No	Partial	No	No	No
Tanium Asset									
Yes	Yes	Yes	Yes	Yes	No	No	No	No	No

1. Doesn't support BDNA running on Oracle DB. Flexera Normalize v5 was released April 9th, 2018 as a separate download.
2. Doesn't collect publisher information, and has potential for false recognition positives. BigFix platform inventory source does not provide the serial number from UNIX-like platforms.
3. Virtualization: VMware, and Solaris Zones. Collects VM properties, but doesn't collect host-VM relationships.
4. Mobile devices only.
5. App-V integration provides application virtualization in Microsoft Endpoint Configuration Manager (previously Microsoft SCCM or SMS).
6. The Microsoft Intune adapter, only captures Windows devices with an ownership type of **Corporate**. Personal, Apple and Android devices are not retrieved. For a full list of what inventory is collected from Microsoft Intune and imported into IT Asset Management, see [What data is retrieved](#).
7. Virtualization: VMware, and Hyper-V. Doesn't collect pool or resource allocations.

## Server Virtualization

Information Collected per Product									
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB
			Installer	File		Server	Application		
Microsoft Hyper-V*									
						Yes		Yes	
Oracle VM Server for x86									
						Yes		Yes	
VMware vCenter*									
						Yes		Yes	
VMware ESXi Server*									
						Yes			
VMware vSphere*									
						Yes			

\* Refer to the **Inventory Sources** table above for sources that provide data for these technologies.

## Application Virtualization

Information Collected per Product									
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB
			Installer	File		Server	Application		
App-V Server <sup>1</sup>									
Partial	Partial	Yes	No	No	Yes	No	Yes	No	No
App-V integrated with SCCM <sup>2</sup>									
Yes	Yes	Yes	No	No	Yes	No	Yes	No	No
EdgeSight for Citrix Virtual Apps (previously XenApp EdgeSight) <sup>3</sup>									
Yes	Yes	Partial	No	Yes	Yes	No	Yes	No	No
Citrix XenApp server									
Yes	Yes	Yes	No	Yes	Yes	No	Yes	No	No
Citrix Virtual Desktops <sup>4</sup>									
							Yes		
Flexera One SaaS Management									
Dummy <sup>5</sup>	Yes	Yes	Partial <sup>6</sup>	No	Yes	No	Yes	No	No

1. Requires mapping of packages to applications within IT Asset Management. Requires Active Directory integration and FlexNet Inventory Agent.
2. Provided by Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) adapter.
3. Installer evidence is gathered when combined with Flexera's XenApp server agent.
4. Provided by a dedicated FlexNet Inventory Agent.
5. Flexera One SaaS Management does not track computers. Since IT Asset Management requires a computer record for various database links, a single dummy record is created (if not already present from previous imports) during import from this connector. Note that this connector imports completed licenses managed in Flexera One SaaS Management, including all entitlements, for inclusion in the nightly license compliance calculations.
6. Imports from the Flexera One SaaS Management connector import installer evidence for Office 365, allowing for application recognition in the nightly license reconciliation. This matches the behavior of earlier connectors for Office 365.

## Other System Integrations

Information Collected per Product									
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB
			Installer	File		Server	Application		
Oracle Database*									
									Yes
Oracle E-Business Suite*									
									Yes
Oracle Enterprise Manager <sup>1</sup>									
Salesforce									
									Yes
Office 365									
									Yes
SAP BASIS*									
									Yes

\* Provided by a dedicated FlexNet Inventory Agent.

1. Provides Oracle database connection details for use with a dedicated FlexNet Inventory Agent.

ITSM System Integrations

Information Collected per Product													
Computer	User	Product code	Evidence		Usage	Virtualization		Clusters	Oracle DB				
			Installer	File		Server	Application						
ServiceNow <sup>1</sup>													
BMC Atrium CMDB and Remedy <sup>2</sup>													

- 1. Export computers and installed applications to ServiceNow. Import contracts and assets from ServiceNow into IT Asset Management.
- 2. Export computers and installed applications to Atrium. Import asset details from Remedy and Atrium into IT Asset Management.



# AWS Adapter

## IT Asset Management (Cloud)

This part introduces the AWS adapter, with a particular focus on managing licenses that you provide for software used in the cloud (often called "bring your own [software] license", or BYOL). The complementary case, where you purchase instances fully provisioned by Amazon Web Services (including the provision of relevant licenses), is of less interest to your license management team, and is perhaps more the focus of your finance team.

The AWS adapter serves two main, and closely related, purposes:

- It tracks *instances* (typically, virtual machines that are tracked as inventory devices) that are hosted for you in the Amazon Elastic Compute Cloud (EC2)
- It performs discovery of Oracle Database installations in Amazon Relational Database Service (RDS). From the data uploaded from the AWS adapter, IT Asset Management automatically creates both discovered device and inventory device records. The discovered device records allow you to target those Oracle Databases for direct inventory collection; and the database inventory is combined with data received from the connector to provide licensable results for your matching inventory device records.

In relation to EC2 instances, the AWS adapter is only a part of a complete BYOL management strategy for this cloud service provider. By itself, the AWS adapter does not gather sufficient data to allow for license management: for example, it does not gather software inventory for the applications that are running on your cloud-hosted devices.

For that reason, this part covers more than the AWS adapter itself. As well as details of prerequisites, set up, and data gathering with the AWS adapter, the part includes some insights into:

- Processes for gathering *complete* hardware and software inventory for your EC2 instances
- Considerations for license management
- Preparing different Amazon Machine Images (AMI) from which to launch instances capable of reporting software and hardware inventory.

In fact, using those guidelines, you *could* configure inventory gathering in AWS EC2 without using the AWS adapter at all. Provided that you are deploying the latest FlexNet Inventory Agent (13.2.x or later), inventory alone even identifies the cloud service provider. What the AWS adapter adds is:

- Automatically setting the **Hosted in** property in the inventory device records created when inventory is returned
- Adding the cloud service provider and instance details when these are missing from inventory collected by legacy

versions of FlexNet Inventory Agent, or from third-party inventory sources

- Populating the **Cloud Service Provider Inventory** page with records of instances currently running in your AWS EC2 environment, including a few properties that are additional to the normal hardware and software inventory
- Importing permanent records of instances that were previously running but that have now been terminated (and for which, as a result, any prior inventory devices records have now been deleted)
- Triggering automatic deletion (completed at the next full import and compliance calculation) of any inventory device record linked to an instance that is now terminated
- Discovering Oracle Database installations in Amazon RDS.

This part focuses on background about the AWS adapter, and additional material you need in order to build license management around the AWS adapter. You will not find details about how to choose the right connection method for your business processes (there are three methods available), or how to undertake the actual connection here. Instead, step-by-step instructions for configuring the connection to AWS EC2 are available in the online help at ***FlexNet Manager Suite Online Help > Inventory Beacons > Inventory Systems Page > Connecting to External Inventory Systems > Managing PowerShell Connections > Managing AWS Connections.***

# 1

## Prerequisites and Setting Up

### IT Asset Management (Cloud)

The AWS adapter is configured on a convenient inventory beacon, for which there are the following prerequisites:

- PowerShell 3.0 or later is running on Windows Server 2008 R2 SP1 or later, or Windows 7 SP1 or later; with the PowerShell execution policy set to RemoteSigned.



**Note:** If you choose to install *AWS.Tools.Common*, which is the modular collection where you can install individual AWS Tools for PowerShell, the minimum version is increased to PowerShell 5.1 or later.

- .NET Framework 4.7.2 or newer is required.
- The FlexNet Beacon software installed on the inventory beacon must be release 13.1.1 (shipped with IT Asset Management 2018 R2) or later.
- The inventory beacon must have Internet access to the central application server for IT Asset Management, so that it can upload inventories and download policies.
- A web browser is installed and enabled on the inventory beacon.
- You must log onto the inventory beacon, and run FlexNet Beacon, using an account with administrator privileges.
- You must have downloaded AWS Tools for PowerShell from <https://aws.amazon.com/powershell/>, and installed them on the inventory beacon. The minimum required version of these tools is 3.3.283.0.



**Tip:** To check the version installed on your inventory beacon:

1. As administrator, run PowerShell.
2. Execute the `Get-AWSPowerShellVersion` cmdlet.

New versions are available for download from <https://aws.amazon.com/powershell/>.



**Note:** The permissible values for **Instance region** are currently hard coded in the AWS Tools for PowerShell. This means that if AWS create additional regions, and you want to have instances in one of the new regions, you will need to update AWS Tools for PowerShell at that time.

From release 2.0.0 of the PowerShellGet cmdlet (which allows you to Install-Module), the Scope option

defaults to `CurrentUser`. Unless you are installing the modules using the account that normally runs your FlexNet Beacon and triggers the AWS connector, you must specify the non-default value of `AllUsers`, such that the inventory beacon's operating account (typically `SYSTEM`) can run the connector. Example command lines for installation are:

- For the modular AWS Tools for PowerShell (requires PowerShell 5.1 or later), run this sample script that encompasses all the required modules needed to run the connector:
- For the bundled, legacy AWS Tools for PowerShell (requires PowerShell 3.0 or later), run this sample script:

After installation, the tools are found in one of the paths listed in the preference `PSModulePath`, such as `C:\Program Files\WindowsPowerShell\Modules` (check the preference value on your inventory beacon).

On the AWS side, you must first create:

- A policy allowing access to your EC2 service
- A policy allowing access to an Identity and Access Management (IAM) entity
- The IAM roles to grant access to AWS resources (not required when using the *Configuring Connections to AWS EC2 using IAM Users* method)
- The IAM user account (still within AWS) with minimum privileges that makes the connection to AWS APIs and imports the available data (only required for the *Configuring Connections to AWS EC2 using IAM Roles* **without** FlexNet Beacon installed on EC2 instance only).

Finally, on the inventory beacon that is to make the connection to AWS, you must specify the connection (which is automatically scheduled for you).

### Using Proxy Connections

The connection to AWS supports optional use of a proxy for connections that *do not* connect using a locally installed FlexNet Beacon.

Step-by-step instructions for creating these, and then configuring the connection to AWS ECS, are available in the online help at ***FlexNet Manager Suite Online Help > Inventory Beacon Help > Inventory Systems Page > Connecting to External Inventory Systems > Managing PowerShell Connections > Managing AWS Connections.***



## 2

# Thinking about Inventory and Licensing

## IT Asset Management (Cloud)

In many ways, collecting inventory and working out license consumption for instances in AWS EC2 is much the same as licensing virtual machines on host servers sitting within the data center in your own offices. However, there are some differences that are worth thinking about. For example, within your own data center, you have full control of, and presumably full inventory for, the host server as well as the VMs running on it; and for some license types, host details are relevant even when licensing software on a virtual machine. Depending on what you purchase, this may not be the case in AWS EC2.

The following topics bring together some points to consider, first about inventory collection, and then about licensing.

## Collecting Inventory from Instances

### IT Asset Management (Cloud)

The AWS adapter gathers useful information, but it cannot gather information valuable for license consumption calculations from each of the instances running in your AWS cloud. To gather useful inventory from all instances, best practice is to include the FlexNet Inventory Agent (release 13.2.*buildNumber* or later) in the Amazon Machine Image (AMI) from which your devices are instantiated. For details, see the AWS documentation, [Create a Standard Amazon Machine Image Using Sysprep](#) for Windows, or for Linux [Creating an Amazon EBS-Backed Linux AMI](#). For additional guidance about the kinds of customizations your AMI may need (depending on the life expectancy of instances to be launched from the image), see either:

- [Configuring an AMI for Short-Lived Instances](#)
- [Configuring an AMI for Longer-Life Instances](#).

### Unique device naming is critical

It is critical to your license management in AWS to ensure that each instance is *uniquely* named on instantiation. If you do not do this, incoming inventory reports are not differentiated, and are interpreted as updated inventory from a single device, named with the default device name provided by the base image.

One easy way that *appears* to give each instance a unique name different from its AMI is through the AWS **Ec2 Launch**

**Settings** dialog, by selecting the **Set Computer Name** check box (or setting the Boolean in the `LaunchConfig.json` file). However, be aware that, since this name is of the form `ip-hexInternalIP`, the name may not be unique *over time*, particularly if you have short-life instances. If an internal IP address is dropped when a first instance is shut down, and reused for *another* instance that is instantiated later, then these different instances are overlaid in the FlexNet inventory database as being updates to the same device (having the same name). However, when the inventory records are then imported over time into the compliance database, they are differentiated by their different instance IDs, producing separate (duplicate) device records with the same device name. Therefore, the following topics provide another way to get a *truly* unique name for each device reported in inventory; and recommended best practice is to always leave the **Set Computer Name** check box clear. At the very least, use this convenient AWS method for naming instances only when you deem the risk of duplicate device records in IT Asset Management to be low enough to ignore (or you are prepared to manually fix any duplicate inventory device records that do occur).

To ensure a device name for each instance that is unique over time, methods vary across platforms, and are detailed in following topics. In summary:

- The easiest method on Windows is to use Sysprep when creating your AMI, leaving the name unset so that a new name is provided on instantiation.
- For Linux, if you already provide for unique `Hostname` values on all your instances, no further action is required; and otherwise, a little extra preparation can ensure that the unique `InstanceID` for each instance is also returned in inventory as the `MachineID`, where it differentiates all returned inventory as coming from devices with distinct names.

## Schedule specialization

Scheduling inventory collection by the locally-installed FlexNet Inventory Agent requires that you take either of two different approaches, based the 'life expectancy' of each instance:

- Short-lived, on-demand instances may not have a long enough life cycle for default FlexNet inventory processes to come to full operation. For these short-lived instances, consider a customized default schedule and configuration file in the AMI that will:
  - Provide an upload location (`ManageSoftRL`) for inventory collection
  - Omit any download location, since the instance life-cycle is projected to be too short to require any policy update or system-wide schedule update
  - Trigger inventory collection on start-up, followed immediately by inventory upload to the defined `ManageSoftRL` (this cycle might typically require 3-5 minutes all up, a figure which may change based on other customizations you may want to include in your AMI)
  - Provide a back-up schedule of daily inventory collection, covering off the possibility that, unexpectedly, this instance now needs to run for a longer period.

Keep in mind, as well, that removing the normal paths for policy update also removes the normal updates to `InventorySettings.xml`, which includes a range of advanced capabilities for FlexNet Inventory Agent, including Oracle inventory collection and advanced inventory techniques for Microsoft and other vendors. Your strategy therefore includes embedding your most recently downloaded `InventorySettings.xml` in your AMI (and later, updating it as required). For more guidance on customizing short-lived instances, see [Configuring an AMI for Short-Lived Instances](#).

- Instances with longer life spans can be managed exactly as you manage third-party installations on other inventory

devices (particularly virtual machines) within your enterprise. These instances do not require a custom schedule, since they can share the same schedule for local inventory collection that applies throughout your enterprise. You do still need to consider customizations built into the AMI for these instances, such as providing a bootstrap inventory beacon with upload and download locations, as well as giving the instance its unique device name. For more details, see [Configuring an AMI for Longer-Life Instances](#).

## Planning your inventory beacons

The final major question is the location of your inventory beacon(s) for uploading inventory data from these instances. Depending on the reliability of network communications between your cloud-based instances and your own enterprise network, you may choose either:

- To run (at least) one instance within your AWS cloud as an inventory beacon, ready 24/7 for any of your instances in the cloud to upload their inventory. This is especially valuable for short-lived instances, where the locally-installed FlexNet Inventory Agent has no opportunity to retry uploads around any network failures; and where fast network speeds may best support your planned short life-cycle for these instances. The cloud-based inventory beacon can then provide a more rugged upload to your enterprise network, since it has longer up-time and built-in mechanisms for retrying any uploads that are interrupted by transient networking issues. As always, this inventory beacon must have Internet access so that it can connect to the central application server for IT Asset Management to upload collected inventories and download changed policies. This approach also enables running the AWS adapter without having to create an IAM User.
- Run an inventory beacon within your enterprise network (or even, potentially, in a demilitarized zone outside your firewall), to which all the cloud-based instances have network access, allowing them to upload their inventory files as and when needed. This approach may be most helpful for direct inventory of Oracle Database in Amazon RDS, since it allows the inventory beacon to be 'steered' to the appropriate AWS region, limiting its inventory collection and uploads in a way similar to managing subnets in your local network. If you have a large number of Oracle Databases spread across various Amazon RDS regions, or a large number of EC2 instances to upload from, consider installing multiple inventory beacons and 'filtering' each one appropriately.

Choosing between these alternative places for your inventory beacon(s) may require balancing questions of security and performance against cost. Whichever choice you make, a good practice is to use a DNS alias to identify the inventory beacon(s), so that you can quickly and easily make changes without disrupting the rest of your infrastructure.



**Tip:** As for all communication between your enterprise network and the AWS network, an Amazon Virtual Private Cloud (VPC) is required for either of the above architectures; and if you require a link from your inventory beacon in the AWS cloud to your central application server in your datacenter, consider a hardware VPN connection as well (see the AWS documentation for details).

Keep in mind that an inventory beacon does not normally initiate ('push') communication to FlexNet Inventory Agents installed on your AWS instances. (The only exceptions are for the 'remote execution' tasks known as adoption and zero footprint inventory collection, neither of which you expect to use in a cloud environment.) Since all other communications are initiated by the FlexNet Inventory Agent ('pull' communications, such as policy updates, collection of self-update packages, and the upload of inventory), this means that your implementation only needs to point the FlexNet Inventory Agent on each instance to its bootstrap inventory beacon (as described in following topics). You do not need to know details like the IP addresses (public or private) for the instances reporting inventory through the FlexNet Inventory Agent.

## Inventory records and exceptions

After a full inventory import and license reconciliation, you can expect to see records created/updated in IT Asset Management as follows:

- For every instance where the locally-installed FlexNet Inventory Agent has uploaded inventory, an inventory device record
- Similarly, for every instance reporting inventory through a third-party tool, such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), an inventory device record
- For every cloud instance appearing in EC2 data from the AWS adapter, regardless of whether it appears in uploaded inventory or not, a cloud instance record in the `CloudServiceInstance` table (and of those, the ones that *also* have an inventory device record show a link to it in the **Cloud Service Provider Inventory** page)
- For every installation of Oracle Database in Amazon RDS, a linked pair of discovered device and inventory device records
- For dedicated hosts identified through the AWS adapter (and *only* dedicated hosts, not any other kind of host), an inventory device record for the host.

You will *not* find inventory device records for any of the following:

- A cloud instance that does not report inventory
- A host device that is anything other than a dedicated host (such as a default host, or a host for dedicated instances)
- Any instance that you have terminated.

Terminated instances are a special case.

1. Before termination, the running instance may have had an inventory device record (provided that it was reporting inventory).
2. When you terminate the instance, AWS keeps the terminated instance visible through the API for about an hour, and if you have implemented your AWS adapter on an inventory beacon with the default 30 minute schedule, the termination is imported through the AWS adapter, and the record in the `CloudServiceInstance` table is updated with the terminated status. (This record, with its terminated status, is visible in the **Cloud Service Provider Inventory** page, if you set the filters on that page appropriately.)
3. At the next full inventory import (by default, overnight, immediately before the license compliance calculations), *any inventory device record linked to a terminated instance is deleted*, because you do not want terminated instances consuming from your licenses.
4. Of course, the terminated instance no longer reports inventory; but its previously-recorded inventory is still in the inventory database (and in normal processes is not cleaned up for a while). Although the next full import (typically the next night) normally imports records from the inventory database, for terminated instances this is prevented by the terminated flag in the `CloudServiceInstance` table. This prevents the deleted inventory device record from reappearing.

Should you ever need to review the software previously installed on a terminated instance, you can:

1. Go to the **Cloud Service Provider Inventory** page (**Inventory > Virtual Devices > Cloud Service Provider Inventory**).
2. Change the default filter (top left) for **Include terminated instances** to Yes.

3. Optionally, filter the **Last known state** column to show only Terminated.
4. Optionally, use other filters, such as **Instances reported** in an appropriate period, or the **Instance type** and **Instance region** columns, to narrow your search for this instance.
5. Check the **Image ID** column to identify the original image from which this instance was launched (you may need to drag this column out of the column chooser).
6. Through your AWS console, identify this AMI, and inspect its software load and resulting license impacts.

## Data integration

Inventory from a hosted EC2 instance may come from multiple sources, and must be integrated correctly. Various rules are used depending on the sources being combined:

- AWS adapter + current FlexNet Inventory Agent are linked by:
  - CloudServiceProviderID (displayed as the **Cloud service provider** name in the web interface)
  - **Instance ID**
- AWS adapter + other inventory sources (such as legacy versions of FlexNet Inventory Agent, or third-party inventory tools, neither of which provides the instance ID) are linked by:
  - **MAC address** (provided that the MAC address is unique in the unmatched inventory and cloud data).

Matching of the incoming data with existing inventory device records (that is, determining whether this is an update to an existing record, or a new record) uses the standard device matching rules, with the **Cloud service provider** and **Instance ID** checked as last priority.

Finally, the merging of data between the AWS adapter and the current FlexNet Inventory Agent uses these priorities:

1. If one source contains a data point that the other does not, the data is used.
2. If both sources contain the same data point but with different values, the data with the most recent inventory date is used.
3. If different values were imported on the same day, the FlexNet Inventory Agent is given priority if it has been set as the primary inventory source.
4. Otherwise, the data is used from the most recently created inventory source, with the winning source visible in the inventory device properties as **Last inventory source**.

# BYOL Licensing Considerations

## IT Asset Management (Cloud)

Full software and hardware inventory may be returned from an AWS EC2 instance when either:

- FlexNet Inventory Agent is locally installed on the instance, usually because it is part of the Amazon Machine Image (AMI) from which the instance was launched
- You have some third-party tool, such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) collecting inventory from your AWS instances, and these results are then being imported into IT Asset Management.

Whatever the method, when inventory is returned from a running AWS EC2 instance, it has an inventory device record

in IT Asset Management. As always for all inventory devices, the installations found in software inventory for that device consume entitlements from the licenses to which each software record is linked.

However, there are a few special adjustments for inventory from AWS EC2 instances.

## Beware of *terminating* short-lived instances

For some special purposes, instances in the AWS EC2 cloud may be launched, run for a short period (from a few minutes to a few hours), and then shut down until needed again.

It is quite possible to configure the AMI from which short-lived instances are launched so that inventory is collected and uploaded through one or more inventory beacons to your IT Asset Management application server (for details, see [Configuring an AMI for Short-Lived Instances](#)).

However, the licensing implications are quite different when you *stop* an EBS-backed instance than when you *terminate* an instance:

- The assumption is that a *stopped* instance may be restarted when required, and resume operation with (quite likely) the same software installed. It is therefore reasonable to calculate license consumption for such an instance, and, if you use the configuration described in [Configuring an AMI for Short-Lived Instances](#), this happens as usual for virtual machines (whether those are VMs within your enterprise network or instances in the cloud).
- A *terminated* AWS instance cannot be restarted, and if you are choosing BYOSL for instances that you terminate, you should check your license terms carefully and make appropriate provision. The working assumption in IT Asset Management is that a terminated instance should no longer consume from your software licenses, just as license consumption stops when you uninstall software from a device, or decommission a hardware asset. As described in [Collecting Inventory from Instances](#), this is achieved by permanently deleting any inventory device record that may have previously been created for the instance that is now terminated. (See the same topic for a way of re-discovering the software load for a terminated instance, and hence its potential license implications.)

The complexity of managing licenses on transient instances is why it is more common to purchase these based on a fully-loaded image provided by (and licensed through) AWS. However, if you need repeated runs of a short-lived instance for special purposes, and such an instance must include software for which you provide the license (BYOL), consider running such an instance as a stop/start instance, rather than terminating it and re-launching it, to simplify your license management.

## Thirty-minute inventory and IBM PVU licenses

The default schedule for the AWS adapter is to have it run every 30 minutes. Coincidentally, this happens to be the same interval that IBM requires for hardware checks of devices linked to IBM PVU licenses (when you have IBM's approval to use IT Asset Management as the source for sub-capacity licensing calculations, for which see the *Sub-Capacity Licensing with IBM PVU* topic in *IT Asset Management System Reference*).

However, each of these coincidental but distinct schedules has its own separate reasons:

- The AWS adapter should run every 30 minutes to allow it to capture transient information that AWS keeps available for only an hour, so that you can track your terminated instances where required. This is the *connector* schedule, and the AWS adapter does not return enough inventory detail to satisfy the IBM requirements.
- The hardware check for IBM PVU licenses must run every 30 minutes as part of your amended contractual agreement with IBM. This is an *inventory* schedule that controls operations of FlexNet Inventory Agent embedded on an operational instance.

If you are providing IBM PVU licenses for use in the AWS EC2 cloud, *and* have the IBM amended agreement to allow sub-capacity calculations within IT Asset Management, these two default schedules conveniently satisfy both requirements within AWS. However, be aware of the following requirements:

- IBM PVU licenses in the BYOL model are not appropriate for very short-life instances that are then terminated (see previous section about the pitfalls of terminating instances with BYOL). Start/stop operations can be used when necessary, if the instance is maintained and not terminated.
- Persistent (unchanging) inventory device names and matching instance IDs are critical to this license model, allowing proper calculations of peak consumption values over time. For details about setting a unique but persistent device name for inventory, see [Configuring an AMI for Longer-Life Instances](#).
- As always, full hardware and software inventory (from the instance) is required for IBM PVU licensing. Compliance with your IBM license requires regular software inventory (daily is recommended best practice) as well as the 30-minute hardware check. Use of the FlexNet Inventory Agent is also mandated by your adjusted license agreement. The techniques in [Configuring an AMI for Longer-Life Instances](#) prepare an AMI from which you can launch instances with the operational FlexNet Inventory Agent embedded, satisfying those requirements.
- As always, the first full software and hardware inventory must be uploaded and processed on the central application server *before* the 30-minute hardware checks take effect. (This allows the calculation of updated rule targets that identify all devices affected by IBM PVU licensing processes.) Typically the relevant compliance calculations take place overnight, after which the revised device policy must be distributed through the inventory beacon hierarchy to affected installations of FlexNet Inventory Agent, bringing with it the special schedule for the hardware check. All of this process means that in the normal course of events, IBM PVU operations typically are running effectively from the day *after* an instance is launched with PVU-licensed software in operation.

## Uncapped calculations except on dedicated hosts

Some license types allow that, where the total license liabilities for many VMs on the same host theoretically exceed factors like the total number of cores available on the host, the summing process is 'capped' or given an upper limit by the actual capacity of the host. For example, if five VMs on a host are each allocated 2 cores, for a total of 10 allocated cores when the host actually only has 8 cores available, then, just as in reality the time-sharing between the VMs limits the cores in use at any moment, so also the license rules cap (or reduce) the calculated figure (10 cores) to the total capacity of the host (8 cores).

Obviously, capping of consumption calculations requires inventory not only of all the VMs on a host, but also of the host itself. However, in the case of AWS EC2 instances, inventory details are not available for any host types other than 'dedicated hosts', over which you have full control. This means that on all other, non-dedicated hosts in AWS EC2, capping of license calculations cannot occur.

All calculations for BOYSL licensing of instances on non-dedicated AWS hosts are based entirely on the inventory taken from the instances themselves, without capping. This does not expose you to any risk of under-licensing (provided that all instances are returning inventory); the only risk is that you may *perhaps* be over-calculating the consumption that *might* apply on a known host with capping applied.

However, this may not be an issue at all. Carefully check your BYOL terms and use rights. You may find that software publishers already disallow capping in cloud environments, because of the impossibility of tracking instances that may pop up now on this host and now on another, depending on resource allocations within AWS.

## 3

# Images for Different Kinds of Instances

## IT Asset Management (Cloud)

The easiest way to collect inventory from all your EC2 instances is to make sure that they are launched from Amazon Machine Images (AMI) that already contain everything needed for the collection of FlexNet inventory.

However, the required customization varies somewhat, based on your plans for the resulting instances:

- Instances with short lifespans (from a few minutes to a few hours) need special provision so that they collect and upload inventory as soon as they are launched. For details of ways to achieve this, see [Configuring an AMI for Short-Lived Instances](#).
- Instances with longer lifespans (two days or more, whether or not they are actually running for all of that period) can use a more conventional customization that brings them into the normal policy-based management processes used everywhere by IT Asset Management. For these, see [Configuring an AMI for Longer-Life Instances](#).

## Configuring an AMI for Short-Lived Instances

### IT Asset Management (Cloud)

The requirements for short-lived on demand instances in your AWS EC2 service are somewhat specialized because of the need to capture and upload inventory within a short space of time. This means you need to prepare a separate Amazon Machine Image (AMI) from which to instantiate these short-lived instances. The requirements include:

- Installing the latest approved FlexNet Inventory Agent into the image
- Adding a customized configuration file that identifies the upload location for inventory
- Installing a customized schedule to manage FlexNet Inventory Agent in the resulting instances, triggering inventory gathering on start-up (and best practice is to include a backup schedule for further inventory collection in case an instance from this AMI runs for an unexpectedly long time)
- Including the current version of `InventorySettings.xml` that provides advanced functionality for FlexNet



Inventory Agent, since the short lifetime does not allow for normal policy-based downloads

- Ensuring a unique name for each instance created from this AMI.

At this summary level, the requirements are the same for Windows and Linux platforms. In the details, of course, there are platform specifics.



#### **To prepare an AMI for short-lived instances:**

1. Go to the **Inventory Settings** page (**Data Collection > IT Assets Inventory Tasks > Inventory Settings**).

The **Inventory Settings** page displays.

2. Expand the **Inventory agent for download** section.

3. Collect the template configuration file:

- For a Windows-based AMI, click **Download bootstrapping template file**, and save `mgssetup.ini` to a convenient working folder (such as `C:\temp`). (Do not change the file name.)
- For a Linux-based AMI, in *Gathering FlexNet Inventory*, copy the text from *Agent third-party deployment: Sample UNIX Bootstrap Configuration File* and save it as `mgsft_rollout_response` in a convenient working folder (such as `C:\temp`).

4. From the **Inventory agent** drop-down list, select the version of FlexNet Inventory Agent you want to install in your AMI, and save it to your working folder.

In general, install the latest available version, subject to your corporate policies. This provides access to the latest functionality. For example, to include advanced inventory for AWS EC2, you must use FlexNet Inventory Agent 13.2.0 or later.

5. In your preferred flat text editor, customize your bootstrapping template file to be used for FlexNet Inventory Agent in your AWS EC2 environment as follows, saving the edited version in a separate subfolder.

For Windows, the only *mandatory* change is to identify the upload location for gathered inventory, as described below. (A download location for policy updates is not needed for short-life instances on Windows.) On Linux, FlexNet Inventory Agent requires both an upload location and a download location. As always, on either platform, experts may also customize other preferences needed for your implementation, as described in the platform-specific topics in the *Gathering FlexNet Inventory* PDF:

- *Agent third-party deployment: Edit the Configuration File for Microsoft Windows*
- *Agent third-party deployment: Configure the Bootstrap File for UNIX.*

The upload (and download) locations require a URL to the host inventory beacon. Best practice is to provide a DNS alias for your chosen inventory beacon, so that when circumstances change, a simple switch of the alias leaves any running and future instances fully operational, without requiring changes to the AMI.

- For **Windows**, in `mgssetup.ini`:
  - a. Locate this section for Common preferences, and uncomment (remove the leading semi-colon) and edit the following settings:

```
;=====
; Registry settings to be created under
```

```
; HKLM\Software\ManageSoft Corp\ManageSoft\Common
[Common]
desc0 = UploadSettings\Bootstrap Server\Protocol
val0 = http
desc1 = UploadSettings\Bootstrap Server\Priority
val1 = 100
desc2 = UploadSettings\Bootstrap Server\AutoPriority
val2 = False
desc3 = UploadSettings\Bootstrap Server\Host
val3 = beacon.fnms.com
desc4 = UploadSettings\Bootstrap Server\Port
val4 = 80
desc5 = UploadSettings\Bootstrap Server\Directory
val5 = /ManageSoftRL/
```

- b. Optionally, modify the Protocol value if you want to use HTTPS (val0 = https).
- c. You *must* customize the Host preference (shown above as val3 = beacon.fnms.com) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an AWS EC2 instance. For an inventory beacon hosted on an AWS instance, you can identify its public and private DNS hostnames on the AWS console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read both its public and private DNS hostnames in the details pane).



**Tip:** If you have an inventory beacon on an AWS instance, your installations of FlexNet Inventory Agent on other instances can be configured to use the private DNS hostname for accessing the inventory beacon (provided that both are within the scope of your VPC). The private hostname has greater stability, particular if the inventory beacon host is stopped and started from time to time.

- d. Optionally, customize the port setting (for example, if you are switching to the HTTPS protocol, the default port is 443).



**Important:** The Directory preference is mandatory, and must be set as shown above.

- e. Create a subfolder (such as ShortLifeAMI) and save your edited mgssetup.ini there. Do not change the file name, which is mandatory.
- For **Linux**, in mgsft\_rollout\_response:
  - a. In the first section of the file, customize the setting for the download location (see also the comments for the upload location, next):

```
# The initial download location(s) for the installation.
                                # For example, http://myhost.mydomain.com/
ManageSoftDL/
                                # Refer to the documentation for further
                                details.

MGSFT_BOOTSTRAP_DOWNLOAD=http://beacon.fnms.com:8080/ManageSoftDL/
```

- b. In the next section of the file, customize the setting for the reporting (upload) location:

```
# The initial reporting location(s) for the installation.
# For example, http://myhost.mydomain.com/
ManageSoftRL/
# Refer to the documentation for further
details.

MGSFT_BOOTSTRAP_UPLOAD=http://beacon.fnms.com:8080/ManageSoftRL/
```

You may customize the protocol to HTTPS if required, and omit or customize the port number as required. You *must* customize the host URL (shown above as `beacon.fnms.com`) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an AWS EC2 instance. For an inventory beacon hosted on an AWS instance, you can identify its DNS hostname on the AWS console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read its DNS hostname in the details pane).



**Important:** In each case, the trailing `/ManageSoftDL/` (download location) or `/ManageSoftRL/` (upload or reporting location) is mandatory, and must be specified as shown above.

- c. In the last line of the file, switch the value to prevent policy running after installation, as short-lived instances do not have time to wait for policy and its outcomes:

```
MGSFT_RUNPOLICY=0
```

- d. Create a subfolder (such as `ShortLifeAMI`) and save your edited `mgsft_rollout_response` there. Do not change the file name, which is mandatory.
- e. Unless you already have a strategy and practice that gives every Linux instance a unique name, also create a new text file for your Linux AMI, called `tmpconfig.ini`, with exactly the following two lines:

```
[ManageSoft\Common]
MachineID=
```

This file is deliberately incomplete at this stage, and is to be completed at instance start-up, as described later.

6. For either platform, copy the following and paste it into a plain ASCII text file, and save it in your `ShortLifeAMI` folder as `DefaultSchedule.nds` (this file name is mandatory). Of course, you may choose to unwrap those lines wrapped for readability here.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Schedule PUBLIC "Flexera Inventory Manager schedule"
"http://www.managesoft.com/dtd/schedule.dtd">

<Schedule NAME="Default Schedule" TIMESTAMPTAMP="20180903T044104Z"
SCHESHEMA="60">
  <Event NETWORK="False"
    NAME="Generate Inventory"
    CATCHUP="Never" ID="{F065A477-B500-4611-8340-3BF313FB785B}">
```

```

        <LogicalCommand PARAM=" -o UserInteractionLevel=Quiet"
            ACTION="Report" COMPONENT="Tracker"/>
        <Trigger TYPE_PARAM="0" TIMESTART="143950"
            DATESTART="20180903" TYPE="Logon" MAXDELAY="0"/>
    </Event>
    <!-- Following two events are optional (see notes) --!>
    <Event NETWORK="False"
        NAME="Generate Inventory" IDLEDURATION="86400"
        CATCHUP="Never" ID="{F065A477-B500-4611-8340-3BF313FB785B}">
        <LogicalCommand PARAM=" -o UserInteractionLevel=Quiet"
            ACTION="Report" COMPONENT="Tracker"/>
        <Trigger TYPE_PARAM="1" TIMESTART="001000"
            DATESTART="20180903" TYPE="Daily" REPEAT="21600"
            TIMEWINDOW="3600" DURATION="86400"/>
    </Event>
    <Event NETWORK="False"
        NAME="Upload Client Files"
        CATCHUP="Never" ID="{9CB87BAE-2829-498C-8643-10A9BD3BFA56}">
        <LogicalCommand PARAM="-a"
            ACTION="Upload" COMPONENT="Uploader"/>
        <Trigger TYPE_PARAM="1" TIMESTART="002000"
            DATESTART="20180903" TYPE="Daily" REPEAT="600"
            DURATION="86400"/>
    </Event>
</Schedule>

```

The first event in this schedule triggers machine inventory collection on startup (TYPE="Logon"), and the tracker component automatically attempts an upload to the bootstrap inventory beacon as soon as inventory gathering is complete. If you are certain that all instances from this image are short-lived, only this event is required. If you are using an EBS-backed instance, and you choose to stop and restart your instance (rather than to terminate and re-launch it), this inventory is checked on each restart.

In addition, two further schedule events provide easy management of cases where an instance proposed to have a short life in fact keeps on running for some longer time.

The second event also generates inventory, and is triggered every 6 hours (REPEAT="21600", with all times expressed in seconds), to start within an hour thereafter (TIMEWINDOW="3600"). However, this inventory collection is different than the first case, because here the FlexNet Inventory Agent does not repeat the inventory collection if it has been successful within the last 24 hours (IDLEDURATION="86400"). This means that, effectively, the FlexNet Inventory Agent checks four times a day whether there has been a successful inventory collection in the last 24 hours, and if not, it runs a new inventory collection.

In a perfect world, the third event is redundant because the tracker component uploads inventory as soon as it is collected. This third, separate upload event is a catch-up to work around transient network issues. It checks every 10 minutes to see whether any inventory files have failed to upload (and are therefore still waiting in the staging folder). In normal cases, nothing is waiting, and the uploader shuts down immediately, causing negligible load on the instance. But where there have been networking issues, upload is attempted again every 10 minutes until all files are successfully uploaded and removed from the staging folder.

7. On a convenient inventory beacon, navigate to %CommonAppData%\Flexera Software\Staging\Common\

ClientConfiguration, take a copy of InventorySettings.xml, and save it to your ShortLifeAMI working folder.

This completes the preparation of materials for your AMI for instances of this expected life. Now continue to use these materials to prepare the AMI. For both platforms, the high-level process is to customize a running instance, and then save it as an AMI.

**For a Windows-based AMI**, the recommended best practice is to run Sysprep, as this can easily ensure that each instantiation from this AMI receives a unique device name. For Sysprep, AWS offers two different sets of scripts to assist, with separate documentation:

- For a Windows 2016 AMI, use EC2Launch, and see <https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ec2launch.html#ec2launch-sysprep>
- For an AMI for an earlier version of Microsoft Windows, use the EC2Config service, and see <https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ami-create-standard.html>

**For a Linux AMI**, the following notes relate to an EBS-backed instance, since this is the simpler process, and the flexibility of the resulting instances is greater (for example, supporting stopping an instance without necessarily terminating it). If you have instead chosen to use an instance store-backed instance, follow the AWS documentation closely, and give it priority where there may be any disparities with this document. Choose from either:

- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html> (preferred, for your EBS-backed instance)
- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-instance-store.html> (for an instance backed by the instance store).

8. Following the AWS documentation, select (or create), launch and connect to your chosen instance from which you are creating your AMI.

9. To customize your instance:

- Ensure that the destination folder exists for your installation of FlexNet Inventory Agent, and if not create it.

The default paths for each platform are:

- On Windows, C:\Program Files (x86)\Flexera Software\Agent
- On Linux, /opt/managesoft/bin.

- From your local ShortLifeAMI working folder, copy these files into the destination folder on your instance:

- DefaultSchedule.nds
- InventorySettings.xml
- Only on Windows, mgssetup.ini
- Only on Linux, tmpconfig.ini.

- For Linux only, also copy your amended mgsft\_rollout\_response file to /var/tmp/ on your instance.

- Collect the downloaded installer for FlexNet Inventory Agent from your working folder, and run the

installer on your instance, installing FlexNet Inventory Agent into the prepared destination folder.

The installer automatically configures the custom settings declared in the configuration file (either `mgsetup.ini` or `mgsoft_rollout_response`, depending on platform). For more information about running the installation, see the *Agent third-party deployment: Details* chapter of the *Gathering FlexNet Inventory* PDF, focusing on the installation topic for your preferred platform.

- e. When installation is complete, in the same destination folder and for both platforms, run the following command to install your default schedule:

```
start ndschedag.exe -t machine .\DefaultSchedule.nds
```



**Tip:** When creating your AMI in the following steps, make the name for this image meaningful, such as "Short-life instances", to remind you of the special conditions embodied in the AMI.

10. **Only for Windows:** complete the remaining steps in the AWS document to Sysprep your instance, finishing by clicking **Shutdown with Sysprep** (or, for pre-2016 versions of Windows, click **OK** and confirm **Yes** to run Sysprep and shut down the instance).



**Important:** If you are using EC2Launch on Windows 2016, in the **Ec2 Launch Settings** dialog, be sure to leave the **Set Computer Name** check box clear. This allows Sysprep to clear the computer name from the instance as it shuts down, so that new instances launched from this image each receive a unique device name.

11. **Only for Linux** (for an EBS-backed image):

- a. Navigate to your instance configuration.

For example, if you are creating a new instance, select the third tab across the top of your console, showing **3. Configure Instance**.

- b. If necessary, scroll down, and expand the **Advanced Details** panel.
- c. In the **User data** field, add the following two lines **As text** to ensure that each Linux instance reports a unique device name in its inventory (lines are wrapped here for publication, but each command should be entered on a single line):

```
sudo sed -i
                                "s/MachineID=/MachineID=$(curl
http://169.254.169.254/latest/meta-data/instance-id)/g"
                                tmpconfig.ini
                                /opt/managesoft/bin/mgsconfig -i /var/tmp/
tmpconfig.ini
```

The first line replaces the unfinished `MachineID` setting in the `tmpconfig.ini` file with a setting completed with the Instance ID, which is recovered in the standard way from the AWS metadata for the current image. The second line runs a small FlexNet utility that merges the new value into the standard `config.ini` for the Linux-based FlexNet Inventory Agent. These two lines of code are run as the instance is being instantiated, before the FlexNet Inventory Agent is run. The result is that, whenever an instance is launched from the image you are creating, FlexNet Inventory Agent takes the `MachineID` setting from the `config.ini` file, and reports that unique device name as part of its uploaded inventory.

- d. If your instance is running, stop it now. (Making an AMI from a stopped instance is preferred for lower risk

and better stability in your image.)

12. Monitor the **Instances** page in the Amazon EC2 console, until the state for your chosen instance displays stopped.
13. Follow your preferred process to create an AMI from that stopped instance.

For example, on Windows, since you have already installed AWS Tools for Windows PowerShell, you could use the cmdlet `New-EC2Image`. For Linux, in the navigation pane, you can choose **Instances**, select your instance, and then choose **Actions > Image > Create Image**, and provide the necessary details. For more information on creating AMIs, see either of:

- [https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/Creating\\_EBSbacked\\_WinAMI.html](https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/Creating_EBSbacked_WinAMI.html)
- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html>.

This AMI is now ready to instantiate short-lived instances. As soon as each instance is spawned, the embedded FlexNet Inventory Agent gathers inventory from the instance, and immediately uploads it to the bootstrap inventory beacon. Depending on the resources on each instance, this process may take around 2-5 minutes, after which the instance can be stopped (but remember, not terminated if you wish to track related software licenses in IT Asset Management).

## Configuring an AMI for Longer-Life Instances

### IT Asset Management (Cloud)

An AWS EC2 instance that runs for a longer period (such as at least a day or two) can take part in the normal processes for inventory collection that you use for any virtual machines, whether on-premises or in the cloud. The requirements include:

- Installing the latest approved FlexNet Inventory Agent into the image
- Adding a customized configuration file that identifies both the download location for requesting the initial policy for FlexNet Inventory Agent on the instance, and the upload location for inventory
- Ensuring a unique name for each instance created from this AMI.

Both the operational schedule for FlexNet Inventory Agent, and the latest `InventorySettings.xml` providing enhanced inventory-gathering functionality, are delivered automatically as part of policy. However, this can take a couple of days in the worst case; so if you wish you may also include a start-up schedule and `InventorySettings.xml` in the image to shorten the time to full functionality of FlexNet Inventory Agent. If you do so, both of these are over-written automatically once policy is fully operational. (You can find some notes about these additional documents in [Configuring an AMI for Short-Lived Instances](#).)

These guidelines cover both Windows and Linux instances, so skip the parts that do not apply to your platform.



#### **To prepare an AMI for longer-life instances:**

1. Go to the **Inventory Settings** page (**Data Collection > IT Assets Inventory Tasks > Inventory Settings**).  
The **Inventory Settings** page displays.
2. Collect the template configuration file:

- For a Windows-based AMI, click **Download bootstrapping template file**, and save `mgssetup.ini` to a convenient working folder (such as `C:\temp`). (Do not change the file name.)
  - For a Linux-based AMI, in *Gathering FlexNet Inventory*, copy the text from *Agent third-party deployment: Sample UNIX Bootstrap Configuration File* and save it as `mgsft_rollout_response` in a convenient working folder (such as `C:\temp`).
3. From the **Inventory agent** drop-down list, select the version of FlexNet Inventory Agent you want to install in your AMI, and save it to your working folder.

In general, install the latest available version, subject to your corporate policies. This provides access to the latest functionality. For example, to include advanced inventory for AWS EC2, you must use FlexNet Inventory Agent 13.2.0 or later.

4. In your preferred flat text editor, customize your bootstrapping template file to be used for FlexNet Inventory Agent in your AWS EC2 environment as follows, saving the edited version in a separate subfolder.

For Windows, the only *mandatory* change is to identify the download location for requesting policy, as described below. (On Windows, all other requirements are delivered through policy; but for safety we'll also provide an upload location.) On Linux, FlexNet Inventory Agent *requires* both an upload location and a download location. As always, on either platform, experts may also customize other preferences needed for your implementation, as described in the platform-specific topics in the *Gathering FlexNet Inventory* PDF:

- *Agent third-party deployment: Edit the Configuration File for Microsoft Windows*
- *Agent third-party deployment: Configure the Bootstrap File for UNIX.*

The download (and upload) locations require a URL to the host inventory beacon. Best practice is to provide a DNS alias for your chosen inventory beacon, so that when circumstances change, a simple switch of the alias leaves any running and future instances fully operational, without requiring changes to the AMI.

- For **Windows**, in `mgssetup.ini`:
  - a. Locate this section for Common preferences, and uncomment (remove the leading semi-colon) and edit the following settings:

```
;=====
; Registry settings to be created under
; HKLM\Software\ManageSoft Corp\ManageSoft\Common
[Common]
desc0 = UploadSettings\Bootstrap Server\Protocol
val0  = http
desc1 = UploadSettings\Bootstrap Server\Priority
val1  = 100
desc2 = UploadSettings\Bootstrap Server\AutoPriority
val2  = False
desc3 = UploadSettings\Bootstrap Server\Host
val3  = beacon.fnms.com
desc4 = UploadSettings\Bootstrap Server\Port
val4  = 80
desc5 = UploadSettings\Bootstrap Server\Directory
val5  = /ManageSoftRL/
```



```

desc6 = DownloadSettings\Bootstrap Server\Protocol
val6  = http
desc7 = DownloadSettings\Bootstrap Server\Priority
val7  = 100
desc8 = DownloadSettings\Bootstrap Server\AutoPriority
val8  = False
desc9 = DownloadSettings\Bootstrap Server\Host
val9  = beacon.fnms.com
desc10 = DownloadSettings\Bootstrap Server\Port
val10  = 80
desc11 = DownloadSettings\Bootstrap Server\Directory
val11  = /ManageSoftDL/

```

- b. Optionally, modify the Protocol values if you want to use HTTPS (val0 = https, and val6 = https).
- c. You *must* customize the Host preference (shown above as val3 = beacon.fnms.com and val9 = beacon.fnms.com) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an AWS EC2 instance. For an inventory beacon hosted on an AWS instance, you can identify its DNS hostname on the AWS console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read its DNS hostname in the details pane).



**Tip:** If you have an inventory beacon on an AWS instance, your installations of FlexNet Inventory Agent on other instances can be configured to use the private DNS hostname for accessing the inventory beacon (provided that both are within the scope of your VPC). The private hostname has greater stability, particular if the inventory beacon host is stopped and started from time to time.

- d. Optionally, customize the port settings at val4 and val10 (for example, if you are switching to the HTTPS protocol, the default port is 443).



**Important:** The Directory preference is mandatory, and in each case must be set as shown above.

- e. Create a subfolder (such as LongLifeAMI) and save your edited mgssetup.ini there. Do not change the file name, which is mandatory.
- For **Linux**, in mgsft\_rollout\_response:
  - a. In the first section of the file, customize the setting for the download location (see also the comments for the upload location, next):

```

# The initial download location(s) for the installation.
# For example, http://myhost.mydomain.com/
ManageSoftDL/
# Refer to the documentation for further
details.

MGSFT_BOOTSTRAP_DOWNLOAD=http://beacon.fnms.com:8080/ManageSoftDL/

```

- b. In the next section of the file, customize the setting for the reporting (upload) location:

```
# The initial reporting location(s) for the installation.
                                # For example, http://myhost.mydomain.com/
ManageSoftRL/
                                # Refer to the documentation for further
                                details.

MGSFT_BOOTSTRAP_UPLOAD=http://beacon.fnms.com:8080/ManageSoftRL/
```

You may customize the protocol to HTTPS if required, and omit or customize the port number as required. You *must* customize the host URL (shown above as `beacon.fnms.com`) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an AWS EC2 instance. For an inventory beacon hosted on an AWS instance, you can identify its DNS hostname on the AWS console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read its DNS hostname in the details pane).



**Important:** In each case, the trailing `/ManageSoftDL/` (download location) or `/ManageSoftRL/` (upload or reporting location) is mandatory, and must be specified as shown above.

- c. Create a subfolder (such as `LongLifeAMI`) and save your edited `mgsft_rollout_response` there. Do not change the file name, which is mandatory.
- d. Unless you already have a strategy and practice that gives every Linux instance a unique name, also create a new text file for your Linux AMI, called `tmpconfig.ini`, with exactly the following two lines:

```
[ManageSoft\Common]
MachineID=
```

This file is deliberately incomplete at this stage, and is to be completed at instance start-up, as described later.

This completes the preparation of materials for your AMI for instances of this expected life. Now continue to use these materials to prepare the AMI. For both platforms, the high-level process is to customize a running instance, and then save it as an AMI.

**For a Windows-based AMI**, the recommended best practice is to run Sysprep, as this can easily ensure that each instantiation from this AMI receives a unique device name. For Sysprep, AWS offers two different sets of scripts to assist, with separate documentation:

- For a Windows 2016 AMI, use EC2Launch, and see <https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ec2launch.html#ec2launch-sysprep>
- For an AMI for an earlier version of Microsoft Windows, use the EC2Config service, and see <https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ami-create-standard.html>

**For a Linux AMI**, the following notes relate to an EBS-backed instance, since this is the simpler process, and the flexibility of the resulting instances is greater (for example, supporting stopping an instance without necessarily terminating it). If you have instead chosen to use an instance store-backed instance, follow the AWS documentation closely, and give it priority where there may be any disparities with this document. Choose from either:

- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html> (preferred, for your EBS-backed instance)

- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-instance-store.html> (for an instance backed by the instance store).
5. Following the AWS documentation, select (or create), launch and connect to your chosen instance from which you are creating your AMI.

6. To customize your instance:

- a. Ensure that the destination folder exists for your installation of FlexNet Inventory Agent, and if not create it.

The default paths for each platform are:

- On Windows, C:\Program Files (x86)\Flexera Software\Agent
- On Linux, /opt/managesoft/bin.

- b. From your local ShortLifeAMI working folder, copy these files as shown onto your instance:

- For Windows, copy mgssetup.ini into the destination folder C:\Program Files (x86)\Flexera Software\Agent
- For Linux, copy mgsft\_rollout\_response into /var/tmp/ on your instance.

- c. Collect the downloaded installer for FlexNet Inventory Agent from your working folder, and run the installer on your instance, installing FlexNet Inventory Agent into the prepared destination folder.

The installer automatically configures the custom settings declared in the configuration file (either mgssetup.ini or mgsft\_rollout\_response, depending on platform). For more information about running the installation, see the *Agent third-party deployment: Details* chapter of the *Gathering FlexNet Inventory* PDF, focusing on the installation topic for your preferred platform.



**Tip:** When creating your AMI in the following steps, make the name for this image meaningful, such as "Long-life instances", to remind you of the special conditions embodied in the AMI.

7. **Only for Windows:** complete the remaining steps in the AWS document to Sysprep your instance, finishing by clicking **Shutdown with Sysprep** (or, for pre-2016 versions of Windows, click **OK** and confirm **Yes** to run Sysprep and shut down the instance).



**Important:** If you are using EC2Launch on Windows 2016, in the **Ec2 Launch Settings** dialog, be sure to leave the **Set Computer Name** check box clear. This allows Sysprep to clear the computer name from the instance as it shuts down, so that new instances launched from this image each receive a unique device name.

8. **Only for Linux** (for an EBS-backed image):

- a. Navigate to your instance configuration.

For example, if you are creating a new instance, select the third tab across the top of your console, showing **3. Configure Instance**.

- b. If necessary, scroll down, and expand the **Advanced Details** panel.

- c. In the **User data** field, add the following two lines **As text** to ensure that each Linux instance reports a unique device name in its inventory (lines are wrapped here for publication, but each command should be entered on a single line):

```

sudo sed -i
            "s/MachineID=/MachineID=$(curl
http://169.254.169.254/latest/meta-data/instance-id)/g"
            tmpconfig.ini
            /opt/managesoft/bin/mgsconfig -i /var/tmp/
tmpconfig.ini

```

The first line replaces the unfinished `MachineID` setting in the `tmpconfig.ini` file with a setting completed with the Instance ID, which is recovered in the standard way from the AWS metadata for the current image. The second line runs a small FlexNet utility that merges the new value into the standard `config.ini` for the Linux-based FlexNet Inventory Agent. These two lines of code are run as the instance is being instantiated, before the FlexNet Inventory Agent is run. The result is that, whenever an instance is launched from the image you are creating, FlexNet Inventory Agent takes the `MachineID` setting from the `config.ini` file, and reports that unique device name as part of its uploaded inventory.

- d. If your instance is running, stop it now. (Making an AMI from a stopped instance is preferred for lower risk and better stability in your image.)
9. Monitor the **Instances** page in the Amazon EC2 console, until the state for your chosen instance displays **stopped**.
10. Follow your preferred process to create an AMI from that stopped instance.

For example, on Windows, since you have already installed AWS Tools for Windows PowerShell, you could use the cmdlet `New-EC2Image`. For Linux, in the navigation pane, you can choose **Instances**, select your instance, and then choose **Actions > Image > Create Image**, and provide the necessary details. For more information on creating AMIs, see either of:

- [https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/Creating\\_EBSbacked\\_WinAMI.html](https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/Creating_EBSbacked_WinAMI.html)
- <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html>.

This AMI is now ready to instantiate longer-life instances. As soon as each instance is spawned, FlexNet Inventory Agent asks its bootstrap inventory beacon for policy. Once downloaded, its device policy triggers delivery of your global schedule for inventory collection by the locally-installed FlexNet Inventory Agent, as well as the latest `InventorySettings.xml` file that brings enhanced functionality.

## 4

# Appendices: Technical Data

## IT Asset Management (Cloud)

The following topics do not include content you need for set-up or operation of the AWS adapter. These are the deep-down technical data that you can ignore until you have a real need.

## Appendix 1: Cmdlets in the AWS adapter

### IT Asset Management (Cloud)

The connection from IT Asset Management to the AWS EC2 service relies on the AWS Tools for Windows PowerShell (as described in [Prerequisites and Setting Up](#)). Within that toolset, the AWS adapter uses the following PowerShell cmdlets:



**Note:**

\* Can only be used with the **Direct** adapter type

\*\* Can only be used with the **AWS Config** adapter type

\*\*\* Can be used with both the **Direct** and **AWS Config** adapter type

- Get-AWSRegion\*\*\* — To identify the available AWS geographic regions where your instances may be running
- Get-CFGConfigurationAggregatorList\*\* — Returns the details of one or more configuration aggregators
- Get-EC2Instance\* — To return the list of the instances (virtual machines) you own, and then collect summary inventory details on each one
- Get-EC2Host\* — To identify any dedicated hosts you have available in each region
- Get-EC2ReservedInstance\* — To return inventory details on any instance/capacity you have reserved in an Availability Zone
- Get-IAMAccountAlias\* — To return the list of the account alias associated with the Amazon Web Services account
- Get-IAMAttachedGroupPolicyList\* — To return the list of all managed policies that are attached to the specified IAM group
- Get-IAMAttachedRolePolicyList\* — To return the list of all managed policies that are attached to the specified

IAM role

- `Get-IAMAttachedUserPolicyList*` — To return the list of all managed policies that are attached to the specified IAM user
- `Get-IAMGroupForUser*` — To return the list of the IAM groups that the specified IAM user belongs to
- `Get-IAMPolicy*` — To retrieve information about the specified managed policy, including the policy's default version and the total number of IAM users, groups, and roles to which the policy is attached
- `Get-IAMPolicyVersion*` — To retrieve information about the specified version of the specified managed policy, including the policy document
- `Get-IAMRole*` — To retrieve information about the specified role and the role's trust policy that grants permission to assume the role
- `Get-IAMUser*` — Used as a test for connection success
- `Get-ORGAccountList**` — Lists all the accounts in the organization
- `Get-RDSDBInstance*` — To returns information about provisioned RDS instances
- `Get-STSCallerIdentity***` — To return details about the IAM user or role whose credentials are used to call the operation
- `Select-CFGAggregateResourceConfig**` — Accepts a structured query language (SQL) SELECT command and an aggregator to query configuration state of Amazon Web Services resources across multiple accounts and regions, performs the corresponding search, and returns resource configurations matching the properties.
- `Set-DefaultAWSRegion***` — To set a default AWS region system name
- `Set-AWSProxy***` — To specify the proxy setting for the following interactions through other cmdlets
- `Use-STSRole*` — To return a set of temporary security credentials that you can use to access AWS resources that you might not normally have access to

## Appendix 2: Data Imported by Amazon connector

IT Asset Management (Cloud)

The AWS adapter returns information about each AWS instance, and its host. All instances are subsequently displayed in the **Cloud Service Provider Inventory** page. Hosts are never displayed in this page, as in general hosting is variable and controlled by AWS.



**Tip:** One exception where you have direct control is the 'dedicated host'. Like all other host types, dedicated hosts do not appear in the **Cloud Service Provider Inventory** listing; but they do automatically appear in inventory device listings, such as the **All Inventory** page, where you can search for them using the **Hosted in** property. Any instances from the **Cloud Service Provider Inventory** listing that are hosted on your dedicated host also display a link to the dedicated host properties in the **Host** column.

## Instance data



**Tip:** All kinds of instances return this set of data, whether they are virtual machines or bare metal instances. The latter return an `Instance` type starting with `i3..`

If the data gathered through the AWS adapter matches inventory collected from another source, the **Cloud Service Provider Inventory** page includes a hyperlink to open the inventory device properties page for the instance. Putting that another way, inventory device records are *not* created when the AWS adapter is the sole source of information; but they *are* created when inventory is returned from another source that covers the cloud instance (one possible example is having FlexNet Inventory Agent running locally on the instance). In this latter case, linked records exist in both the `ComplianceComputer` and `CloudServiceInstance` tables.

Some data is additional to the normal content collected as inventory. For example, the `InstanceTenancyID`, used for identifying whether the current device is a `Dedicated` instance, or is an instance running `On dedicated` host, is imported into the `CloudServiceInstance` table, and, as **Availability type**, is visible in both the **Cloud Service Provider Inventory** page and the **Cloud Hosting** tab of the inventory device properties (where inventory exists for the instance).



**Tip:** AWS "bare metal instances" behave like other instances in terms of data gathered and records created.

When there is an inventory device record for an instance, the inventory-related values imported through the AWS adapter (and shown below in alphabetical order) are displayed in the separate **Cloud Hosting** tab of the inventory device properties:

- Account
- Availability type
- Availability Zone
- Cores
- Creation date (and time)
- Host ID (use to link to host)
- Image ID (for the AMI, or Amazon Machine Image, from which the instance was instantiated)
- Instance ID
- Instance region
- Instance type
- Last known state
- Lifecycle mode (displayed as **Lifecycle** in the **Cloud Service Provider Inventory** page, and as **Purchased option** in the **Cloud Hosting** tab of the inventory device properties)
- MAC address
- Network ID
- Threads per core.

The following Oracle RDS instance data is collected:



**Note:** In order to be collected, the Oracle RDS instance must have one of the following engine values:

- `oracle-ee`
- `oracle-se`
- `oracle-se1`
- `oracle-se2`
- Account
- AccountAvailabilityZone
- DBInstanceClass
- DBInstanceIdentifier
- DBInstanceStatus
- DbiResourceId
- EndpointAddress
- EndpointName
- EndpointPort
- InstanceCreateTime
- LicenseModel
- Region

## Host data

If the host is a "dedicated host" (which means that your enterprise has exclusive use and detailed control), an inventory device record is automatically created for the dedicated host in the `ComplianceComputer` table (with the resulting property sheet). Separate inventory device records are *not* created for any other kind of host; and hosts (other than dedicated hosts) are *not* listed separately on the **Cloud Service Provider Inventory** page.



**Tip:** For dedicated hosts, the following properties are displayed on the **Cloud Hosting** tab of the inventory device properties.

The following properties (listed alphabetically) are collected for hosts of all types through the AWS adapter:

- Availability Zone
- Cores
- Creation time
- Host ID



- Instance region
- Instance type
- SocketsVcpu.

## Appendix 3: Enhanced Inventory Gathered by Agent

### IT Asset Management (Cloud)

As well as the high-level data gathered from the AWS EC2 connection, FlexNet Inventory Agent (version 13.1 or later) gathers additional inventory data for each AWS instance where FlexNet Inventory Agent is locally installed. To do so, FlexNet Inventory Agent accesses two AWS-internal URLs:

- `http://169.254.169.254/latest/meta-data/instance-id` returns the unique identifier for the AWS EC2 instance.
- `http://169.254.169.254/latest/dynamic/instance-identity/document` returns a JSON-formatted value containing data about the instance where FlexNet Inventory Agent is locally installed. For example:

```
{
  "devpayProductCodes" : null,
  "marketplaceProductCodes" : [ "1abc2defghijklm3nopqrs4tu" ],
  "availabilityZone" : "us-west-2b",
  "privateIp" : "10.158.112.84",
  "version" : "2017-09-30",
  "instanceId" : "i-1234567890abcdef0",
  "billingProducts" : null,
  "instanceType" : "t2.micro",
  "accountId" : "123456789012",
  "imageId" : "ami-5fb8c835",
  "pendingTime" : "2018-11-19T16:32:11Z",
  "architecture" : "x86_64",
  "kernelId" : null,
  "ramdiskId" : null,
  "region" : "us-west-2"
}
```

These two data points are then embedded within the normal `.ndi` inventory file as a pseudo-hardware WMI class called `MGS_AWSComputerSystem`. The class looks similar to this example (lines wrapped for presentation):

```
<Hardware class="MGS_AWSComputerSystem" Name="ec2amaz-vrkiu0p" Evidence="metadata">
  <Property Name="instance-id" Value="i-1234567890abcdef0"
    Evidence="/latest/meta-data/instance-id"/>
  <Property Name="document" Value="ewogICJwcm12YXRlSXAiIDogIjE3Mi4zMS4yLjExNiIs..."
    Evidence="/latest/dynamic/instance-identity/document"/>
</Hardware>
```

After a full inventory import, these details are eventually saved in the `CloudServiceInstance` table in the compliance database.



# BMC Helix Discovery Adapter

## IT Asset Management (Cloud)

BMC Discovery is a tool for collecting hardware and software information, and can be a useful inventory source as input to IT Asset Management to help in calculating license consumption as part of assessing your overall license compliance.

IT Asset Management supports two different methods of collecting data from BMC Discovery. Using the distinction made throughout this reference (see [IT Asset Management Inventory Adapters and Connectors Reference \(Cloud Edition\)](#)), these are:

- An XML-based *adapter* that first extracts data into an SQL staging database for mapping into the data fields within the IT Asset Management database
- A direct *connector* to BMC Discovery that allows automatic upload of the data directly to the IT Asset Management database.

The adapter is documented in the following chapters. Please ensure that you keep the distinction clear and that you read the topics that relate to your chosen technology.

## Supported versions

BMC Discovery was previously known as Atrium Discovery and Dependency Mapping (ADDM), and was renamed from version 11.

- Adapter: The BMC Discovery adapter supports inventory import from BMC Helix Discovery and the following older releases of the BMC tool: 9.0, 10.0–10.2 (ADDM), 11.0–24.3
- Connector: The BMC Discovery connector supports inventory import from the following releases of the BMC tool: 20.02 (12.0), 20.08 (12.1), 21.05 (12.2).

## 1

# Choosing a Configuration

## IT Asset Management (Cloud)

With two ways (an adapter, and a connector) to collect data from BMC Discovery, you need to decide which of these to implement in your environment:

- The connector is relatively straight-forward. For details about the connector, skip ahead to [How the Connector Works, and Prerequisites](#).
- The adapter to extract data from BMC Discovery can operate at different levels of detail, and with different overheads, that depend on what level of licensing information you need to collect. The following section gives a brief overview of how the adapter works, and explains the different configurations and how to choose between them. You need to choose the configuration appropriate for your enterprise before implementing the adapter.

## How the Adapter Works

### IT Asset Management (Cloud)

Although it is downloaded as a single zipped archive, the adapter includes several components to improve your license reporting. The overview of the finished system shows:

- An optional set of patterns that can be deployed to each BMC Discovery instance to improve the initial level of inventory detail collected by BMC Discovery.

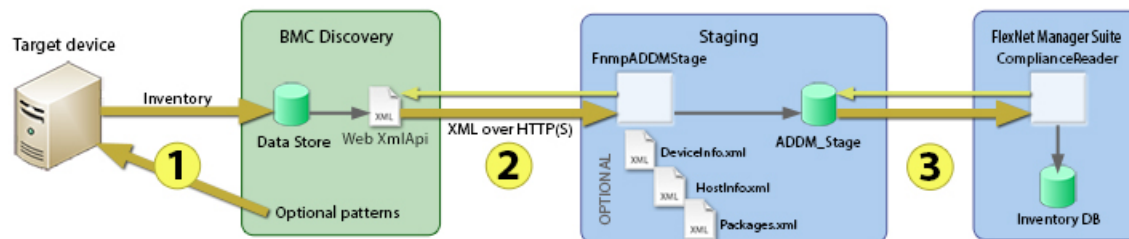


**Tip:** The optional patterns are available for use with BMC Discovery, whether you use the connector or the adapter.

- For the adapter only, a staging server, which includes a simple SQL database where data collected from BMC Discovery can be massaged for upload to IT Asset Management. The executable that speeds the data extraction process also resides here.
- The conversion and upload component, which converts data formats and uploads the result to the central operations databases maintained by IT Asset Management.

Operation is summarized in this high-level diagram:

Figure 1: Process overview



The diagram shows:

1. Optionally making use of additional patterns packaged with the adapter download, inventory details are collected by BMC Discovery (previously ADDM).
2. The executable `FnmppADDMStage.exe` uses the web API to extract the information from BMC Discovery. This method is used because the export process is 100 times faster than using mappings, and there is no requirement to configure custom mappings for each BMC Discovery instance. This executable optionally writes the gathered data into local XML files (available for inspection when required), and (determined by the mode of operation) also writes the data from these XML files into the staging database.
3. A standard component of IT Asset Management called a Compliance Reader, executing from the central application server, collects data from the staging database and imports it into the operations databases. This final stage occurs when you run an inventory import to process the incoming inventory.

## Adapter components explained

The following topics describe each of the adapter components in more detail, with information to help you decide which of these you will implement in your enterprise. Installation and setup follows in a separate section. The adapter components described in later topics are:

- Optional collection patterns to enrich the inventory detail collected by BMC Discovery (see [Optional Patterns](#) in the section about configuring BMC Discovery – and this configuration is common, whether you are using the adapter or the connector)
- The FlexNet inventory agent (see [The FlexNet inventory agent](#), which is required for either the adapter or the connector)
- For the adapter only, the scripts for creating the staging database in Microsoft SQL Server (see [Database Table Creation](#))
- The `FnmppADDMStage.exe` executable to extract inventory information from BMC Discovery, optionally write it to XML files, and insert it into the staging database (see [The Adapter Executable](#)).

## Database Table Creation

IT Asset Management (Cloud)

The staging server requires an operating version of Microsoft SQL Server 2012 or later. Any edition is suitable, including Microsoft SQL Server Express (if its limitations on CPU, RAM, and database size are adequate for your purposes). As described in [Choosing a Staging Server](#), the staging database may be located on a separate staging

server.

In the `SQL\` subdirectory of your unzipped adapter archive, the script `ADDM_staging.sql` is provided for creating the staging database, its tables, and its stored procedure. Obviously, this must be run on the SQL Server instance that is to host the staging database.



**Important:** If you have been using an earlier version of ADDM, and are now migrating to BMC Discovery release 11 (or later), you must run the same `ADDM_staging.sql` script to update your staging database schema. If you are continuing to use ADDM release 10 (or earlier), the update to the staging database schema is not required, and you may omit it for now; but if you choose to apply the upgrade now (so that the staging database is prepared for any future migration to BMC Discovery 11 or later), you must also use the latest version of the `FnpmpADDMSStage.exe` executable from the same `Adapter Tools for IT Asset Management 2024 R2.3.zip` archive. This executable transfers BMC Discovery data (for any BMC Discovery/ADDM version) to the staging database, and is schema-aware for the upgraded staging database.

### Using BMC Discovery to import printers and network devices into Flexera One

Using IT Asset Management in Flexera One provides you with the ability to use BMC Discovery to import printers and network devices into Flexera One. This functionality does not apply to IT Asset Management and only applies to IT Asset Management in Flexera One. If you are using IT Asset Management instead of Flexera One, this printer and network device data is ignored.

To import printers and network devices into Flexera One, you must use the latest version of the `FnpmpADDMSStage.exe` executable from the `Adapter Tools for IT Asset Management 2024 R2.3.zip` archive with new schema, which changes the staging version from 1.11 to 1.12. Other configuration changes remain the same – Flexera One customers using the new adapter from IT Asset Management will not be impacted by this change; however, a schema upgrade is recommended for the adapter to work. IT Asset Management Cloud customers should *not* upgrade to this latest version of the `FnpmpADDMSStage.exe` executable because the new functionality that is included only applies Flexera One.

## The Adapter Executable

IT Asset Management (Cloud)

For use with the adapter, BMC Discovery supports two ways of extracting inventory data it has collected:

- Using export mapping sets
- Using a web API.

While the former is more commonly used, the BMC Discovery adapter for IT Asset Management uses the latter, for the following reasons:

- It avoids the need to deploy and maintain export mapping sets on every BMC Discovery instance in your enterprise.
- Performance of data collection can be 100 times faster using the web API. For example, data extraction that can take over 24 hours using mapping sets can be completed in 20 minutes with the web API. This is because the export feature in BMC Discovery is designed for more complex export capabilities and can therefore be slow to perform simple queries. IT Asset Management requires only simple queries to be executed on BMC Discovery, and these are performed far more efficiently using the web API.

The adapter's tool to query the web API consists of two parts:

- `FnmpADDMStage.exe` — A .NET 4.5 console program capable of querying the XML API of BMC Discovery and writing the results into an SQL Server database, and optionally to XML files on the local file system. This program supports command line arguments, available using `FnmpADDMStage -h`. On running the adapter's executable, `FnmpADDMStage.exe`, a log file called `FnmpADDM.log` is automatically created and saved to the current directory where the `FnmpADDMStage.exe` executable is run from. Logged details include but are not limited to:
  - SQL database and XML errors when they occur. For example, if the `FnmpADDMStage.exe` executable fails to connect to the database.
  - Command line string arguments used to start the executable.
  - Date of when the executable is run.
- `FnmpADDMSettings.xml` — The self-documenting configuration file for `FnmpADDMStage.exe` which contains the queries executed against BMC Discovery (in the BMC Discovery query language), and can include connection settings for BMC Discovery and SQL Server.

In operation, the executable, `FnmpADDMStage.exe`, extracts the inventory data from BMC Discovery and saves it for further processing. There are different ways that it can save the data, based on the following values of its method parameter:

- Stage — Summary: **BMC Discovery to XML**. Inventory gathered from BMC Discovery is saved to a series of XML files on the staging server. It is not imported into the staging database. The XML files allow for review of the gathered data, but the inventory is not imported into IT Asset Management from these files.



**Tip:** The XML file option also allows for disconnected scenarios, where inventory collected from an BMC Discovery server that is out of reach of the staging server can be written to XML, manually copied and transferred to another staging server, and the upload process resumed. See also the *Prestaged* method below.


- Staged — Summary: **BMC Discovery to XML/SQL**. Inventory gathered from BMC Discovery is first written to the XML files on disk (for example for review), and then copied into the staging database where it can be imported into IT Asset Management for use in compliance calculations.
- Prestaged — Summary: **XML to SQL**. In this mode, inventory is not gathered from BMC Discovery. Instead, the XML files present on the disk from a previous inventory collection (and perhaps reviewed and approved by a human agent in this format) are now copied into the staging database where it can be imported into IT Asset Management for use in compliance calculations.
- Stream — Summary: **BMC Discovery to SQL**. Inventory is gathered from BMC Discovery, and loaded into the staging database where it can be imported into IT Asset Management for use in compliance calculations. In this method, inventory is not recorded in XML files on the staging server.

Default values for the method and all other parameters are set in the companion `FnmpADDMSettings.xml` file, and these are the values used when the executable is triggered (or run) without other command-line options. The settings file is self-documented, and the matching command-line options are available using `FnmpADDMStage -h` option.

When the executable writes XML files to (or reads them from) the local disk on the staging server, the files include the following (details are available in `FnmpADDMSettings.xml`):

Filename	Content
ADDMPrinters.xml	List of printers discovered by ADDM.
ADDMNetworkDevices.xml	List of network devices discovered by ADDM.
Cluster.xml	Details for each cluster.
ClusterHost.xml	Computers (host nodes) that are members of a cluster, including a key to the Cluster node to identify that cluster.
CPUInformationDetail.xml	Details of computer processors.
DiscoveredPackages.xml	Raw installer evidence gathered by BMC Discovery from the installer technologies supported by each operating system.
DiscoveredService.xml	A report of particular services, used to help identify capabilities of hosts. This includes the VMMS service used to identify Windows machines with the Hyper-V role enabled.
DiscoveredVirtualMachine.xml	Raw results from BMC Discovery querying the list of virtual machines on a virtualization host.
FileEvidenceDetail.xml	File evidence produced by the Flexera.FNMP.InventoryRawData.FileEvidence pattern, covering software tag files and Windows executables.
FileSystem.xml	Name and size of all local file systems, used to approximate the total disks and storage of host.
HardwareEvidenceDetail.xml	Hardware details produced by the Flexera.FNMP.InventoryRawData.UnixHardwareEvidence pattern, using information gathered by the FlexNet inventory agent.
Host.xml	Details of all hosts known by BMC Discovery including their host name, operating system details, unique identification, processor, memory, and other hardware details.
HostInfo.xml	Raw host details not represented in a Host node, mainly the raw LPAR information from an IBM AIX LPAR environment.
HostContainerProcessorInfo.xml	Processor details for systems that are a container for hosts. For example, a container is a device partitioned into a number of logical hosts.
InstallerEvidenceDetail.xml	Installation evidence gathered using the patterns in the BMC Discovery adaptor, including evidence from installations by Install Anywhere and InstallShield Multi-platform.
LastLoggedOnUserDetail.xml	Details of the last logged-on user for Windows systems. (There is no equivalent data available for UNIX-like systems.)



Filename	Content
NetworkInterface.xml	The IP and MAC addresses of each network interface, used to build a list of these addresses for each host.
	 <b>Note:</b> ADDM 9.0 introduced the <i>IPAddress</i> nodes which cover both IPv4 and IPv6. Prior to that, the IPv4 IP address was in the <i>NetworkInterface</i> node. This query checks both of these sources.
SoftwareInstance.xml	Software installations identified by BMC Discovery's pattern language. BMC Discovery queries various properties such as processes and files of a host to determine which software is installed and its version.
SoftwareInstanceVirtualMachine.xml	These SoftwareInstance nodes are used to represent virtual machines on a host. These records are typically only created when the virtual machine is running.

## Conditions for Use

### IT Asset Management (Cloud)

Use of the adapter's executable, `FnmpADDMStage.exe`, imposes the following conditions:

- Within BMC Discovery, the XML-based API must be enabled (by default, it is enabled). BMC documentation for the XML API is available at <https://docs.bmc.com/docs/display/DISCO111/XML+API>.
- There must be HTTP or HTTPS communication available between the staging server, on which this executable runs, and the server hosting BMC Discovery.
- The staging server requires the .NET 4.5 (minimum) runtime environment.
- The staging tool must be configured with credentials that provide read access to the BMC Discovery instance. These credentials may either be configured in `FnmpADDMSettings.xml` or supplied on the command line for `FnmpADDMStage.exe` (for details see [Account Configuration for Adapter](#)). For Discovery 11.1 and later, you need to grant the API access permission (api-access group), or update group permissions to include the api/access permission for the user that is specified in the `FnmpADDMSettings.xml`, who requires access to the XML Web APIs.
- For linking upstream to the staging database, you can either:
  - Use a trusted connection to the SQL server and run the executable under an account that has read/write access to the staging database; or
  - Use service account credentials for SQL through a connection string, which may either be configured in `FnmpADDMSettings.xml` or supplied on the command line for `FNMPAddmStage.exe` (see [Creating the Staging Database Tables](#) for details).



**Important:** It is critical that the `FnmpADDMStage.exe` tool is not run simultaneously with the `ComplianceReader` tool that uploads from the staging database to the compliance server. The adapter executable commences its writing activity with a truncate of the staging database to clear old results, an action

which (if it occurred in the middle of an upload to the compliance server) could clearly corrupt the imported dataset. While `ComplianceReader` is blocked until `FmpADDMStage.exe` finishes, you must adjust your schedules to prevent starting `FmpADDMStage.exe` while the `ComplianceReader` is running.

## How the Connector Works, and Prerequisites

### IT Asset Management (Cloud)

There are two sides to configuring the connector and its interaction with BMC Discovery:

- An optional set of patterns that can be deployed to each BMC Discovery instance to improve the initial level of inventory detail collected by BMC Discovery.

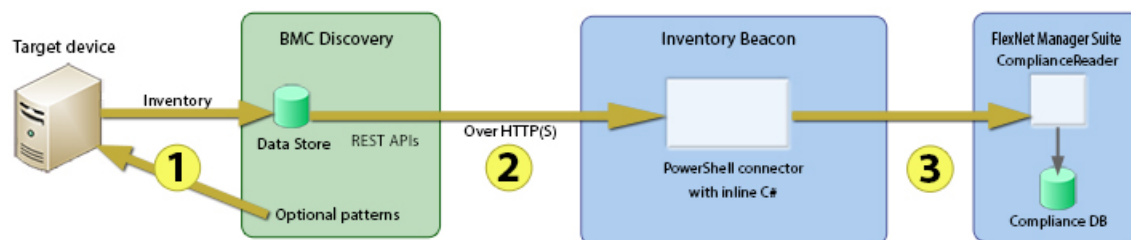


**Tip:** The optional patterns are available for use with BMC Discovery, whether you use the connector or the adapter.

- The connector itself, which is configured on an appropriate inventory beacon. A suitable inventory beacon meets the following requirements:
  - It has network access to your BMC Discovery instance
  - It connects to the application server for IT Asset Management (optionally, through a hierarchy of inventory beacons)
  - FlexNet Beacon is release 17.2.0 (available with IT Asset Management 2021 R1.2) or later
  - PowerShell 5.1 or later is installed
  - A web browser that can access the central application server for IT Asset Management is installed
  - You have the ability to log into the inventory beacon, and run FlexNet Beacon, using an account with administrator privileges.

Operation is straight-forward, and is summarized in this high-level diagram:

**Figure 2:** Process overview



The diagram shows:

1. Optionally making use of additional patterns packaged with the connector, inventory details are collected by BMC Discovery.

2. Configured on an appropriate inventory beacon (one with network access to BMC Discovery), the connector accesses the REST APIs provided by BMC Discovery to extract the required data. You determine the timing of this connection by setting an appropriate schedule as part of the connector configuration.
3. Once the required information is collected, the connector uploads it to the central application server, where it is imported into the inventory database. In the final stage, the regular (default nightly) full inventory import transfers the information to the compliance database, where it is used for license consumption calculations to help determine license compliance.

## The FlexNet inventory agent

### IT Asset Management (Cloud)

By default, the BMC Discovery agent does not support inventory collection from UNIX-like inventory devices; but the optional pattern `UnixHardwareData` extends its capacity to use the FlexNet Inventory Agent on those platforms to collect the appropriate data. Since this is a functional extension of BMC Discovery, this optional pattern, and corresponding use of the FlexNet Inventory Agent on UNIX-like platforms, applies to whichever of the methods you have chosen for collecting information from BMC Discovery (the adapter or the connector). Installation to suit BMC Discovery is described in [Installing the FlexNet inventory agent](#).

The FlexNet Inventory Agent is a standard component of any installation of FlexNet Beacon. For this reason, if you are downloading the *adapter*, you do not find the necessary files included in the BMC Discovery adapter zip archive, since they are already present in any standard implementation of IT Asset Management. Similarly, if you need the FlexNet Inventory Agent for UNIX-like platforms when you are configuring the *connector*, you can find the relevant files already available on the inventory beacon.



**Tip:** If you are gathering FlexNet inventory in parallel with your use of BMC Discovery, it's possible that the FlexNet Inventory Agent may already be installed on target inventory devices. Check before undertaking your own deployment.

For use on Linux or UNIX platforms, the agent has two component files:

- `ndtrack.sh`, the component responsible for collecting inventory details (in this case, for the BMC Discovery inventory system) and writing them in an intermediate format to a data file, ready for upload to an application server. The script has no active elements until it is triggered by BMC Discovery through the enhanced `UnixHardwareData` collection pattern.
- `ndtrack.ini`, a text file that contains configuration variables for `ndtrack.sh`.

The combined disk space requirement of both files is under 13MB.



**Important:** Inventory fails to upload from FlexNet Inventory Agent if the account running `ndtrack.sh` has a hash (#) character in the user name. The hash (#) character is not permitted, and causes failure of the upload of the generated `.ndi` files to the application server.

## 2

# Configuring BMC Discovery

## IT Asset Management (Cloud)

There are three customizations needed for BMC Discovery, the first two of which apply whether you choose to use the adapter or the connector:

- Installing any of the optional patterns needed for BMC Discovery in your environment (see [Optional Patterns](#) to assess your needs, and then [Installing Optional Patterns](#) and [Enabling Optional Patterns](#))
- Ensuring that BMC Discovery applies these patterns to the appropriate target computers (see [Rediscovering Affected Computers](#))
- Only if you are using the adapter, ensuring that the account running the adapter has access to BMC Discovery (see [Account Configuration for Adapter](#)).

## Optional Patterns

### IT Asset Management (Cloud)

The optional patterns described here (and in the two sub-topics) apply in both cases, whether you are using the adapter or the connector to collect information from BMC Discovery.

BMC Discovery allows its data collection to be shaped by a set of optional patterns for choosing what data to collect. IT Asset Management provides five collection patterns in all. This overview may help you choose which patterns you wish to import into BMC Discovery for use in your environment.

1. **FileEvidence** allows collection of file evidence for use within IT Asset Management. This evidence may be of two sub-types:
  - Executable files that form part of the application, which are gathered only for Windows platforms, and may allow application recognition (particularly to the level of editions and versions)
  - Identification files including ISO tag files, which may be gathered across any platform (including Windows and UNIX-based environments).

The next two collection patterns run both on Windows and on non-Windows devices, instructing the BMC Discovery inventory agent to gather additional data:

2. **InstallAnywhereEvidence** returns the list of package titles found in the repository maintained by Flexera's

InstallAnywhere.

- 3. `InstallShieldMultiplatformEvidence` returns the list of packages installed by InstallShield Multiplatform, an earlier installation technology developed by Flexera.  
The next collection pattern is for non-Windows devices only, and uses the FlexNet Inventory Agent to capture additional data elements and integrate these with the BMC Discovery inventory collected in the standard way:
- 4. `UnixHardwareData` gets accurate hardware details to allow license metrics for capacity-based license calculations (such as for Processor, Core, IBM PVU, and other license types).



**Note:** This pattern relies on the installation on UNIX-like devices of two files, `ndtrack.sh` and `ndtrack.ini`, that are available with any installation of the FlexNet Beacon. Configuring this pattern requires that you identify the installation path for these files on the target UNIX-like devices (or accessible file share) – for more information, see [The FlexNet inventory agent](#), and most especially [Installing the FlexNet inventory agent](#).


The last collection pattern contains instructions for the BMC Discovery inventory agent to pull additional data from the Windows Registry and WMI on Windows-based computers:

- 5. `WindowsLastLoggedOnUser` recovers information about end-users that is required for user-based licensing (such as Named User license types).



**Note:** There is no equivalent data available for UNIX-like systems.

To help you assess which of the patterns you wish to use, the following table summarizes the available collection patterns. The default state, whether the pattern is enabled or disabled, is shown in the **Pattern** column. The **Footprint** column details the additional installation impact (other than loading the pattern into BMC Discovery) required for the collection pattern (the downside of using the pattern), and the **Impact** column shows what will *not* work if you omit this pattern (the downside of *not* using it).

Pattern	Footprint	Impact of omitting pattern
<p>FileEvidence</p> <p>Default: Functionality is controlled in two parts:</p> <ul style="list-style-type: none"> <li>Gathering identity (or tag) files (all platforms) is enabled by default</li> <li>Gathering executable evidence (Windows only) is disabled by default.</li> </ul>	<p>No installation required.</p> <p>This pattern is configurable for paths searched (per platform), and for file name extensions used for tag files. Search times on target machines will depend on configuration and the numbers of installed applications found.</p>	<p>An application that</p> <ul style="list-style-type: none"> <li>Is not already correctly recognized by BMC Discovery (perhaps because it is installed but not currently running), and</li> <li>Relies exclusively on file evidence (as distinct from installer evidence) for recognition within IT Asset Management</li> </ul> <p>will not be recognized for inventory collected through BMC Discovery without this pattern. While the Application Recognition Library rarely makes use of file evidence for Windows-based applications, some publishers including IBM and Oracle make use of special identity files. Software identity (SWID) tags are also in increasing use, and these are identified using this pattern.</p> <p>Likely impact of omitting this pattern in total is medium (through the loss of identity files). Likely impact of leaving executable-gathering turned off is low.</p> <hr/> <p> <b>Note:</b> Within IT Asset Management, file evidence is also required for application usage tracking; but no application usage tracking is possible through the BMC Discovery inventory tool.</p>
<p>InstallAnywhere Evidence</p> <p>Default: Enabled</p>	<p>No installation required.</p>	<p>By default, BMC Discovery does not recognize installation evidence from InstallAnywhere. Unless it recognizes the application by other means, the application will be missed without this pattern.</p>
<p>InstallShield</p> <p>MultiplatformEvidence</p> <p>Default: Enabled</p>	<p>No installation required.</p>	<p>By default, BMC Discovery does not recognize installation evidence from InstallShield Multiplatform. Unless it recognizes the application by other means, the application will be missed without this pattern.</p>

Pattern	Footprint	Impact of omitting pattern
UnixHardwareData Default: Enabled	Requires less than 13 MB installation (ndtrack.sh and ndtrack.ini), either on the hard disk of the target machines, or on a network share accessible to them all. Run-time is a second or so when triggered by BMC Discovery.	Without this, BMC Discovery does not capture sufficient hardware attributes from servers to support license consumption calculations for license types based on hardware capacity metrics (such as Processor, Core, IBM PVU, and other license types). Assess impact based on the license types you need to support through BMC Discovery inventory (mandatory for capacity metrics).
WindowsLast LoggedOnUser Default: Enabled	No installation required.	Without this pattern, BMC Discovery does not report any end-user identification that is needed for licenses requiring identification of the individual (such as Named User license types). Note that for general user-based licensing, in the absence of end-user identities, IT Asset Management will calculate every installation of such software as usage by an unknown end-user, so that only license types depending specifically on identity will be affected. If you wish to allocate license entitlements to specific individuals, you also require this identification. Impact: medium.

The patterns are written in the BMC Discovery Pattern Language (TPL), for which documentation is available at .

For more information:

- About each of these patterns, please see [Appendix A: Details of Patterns](#)
- About installing these patterns, see [Installing Optional Patterns](#)
- About enabling or disabling each of the patterns, see [Enabling Optional Patterns](#).

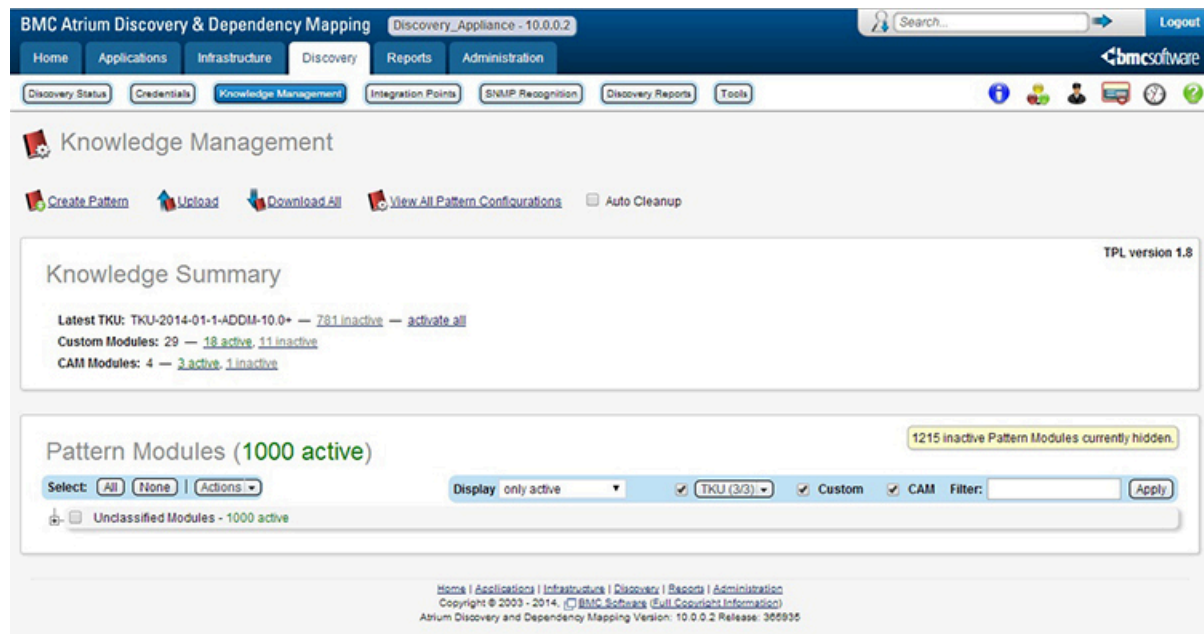
## Installing Optional Patterns

IT Asset Management (Cloud)

For information about choosing which patterns to use, see [Optional Patterns](#). For further details about each of the patterns, and modifying their behavior, see [Appendix A: Details of Patterns](#). All these patterns are contained in a single template (Flexera.FNMP.InventoryRawData.tpl), which must be installed first. Thereafter, individual patterns can be enabled, disabled, and modified.

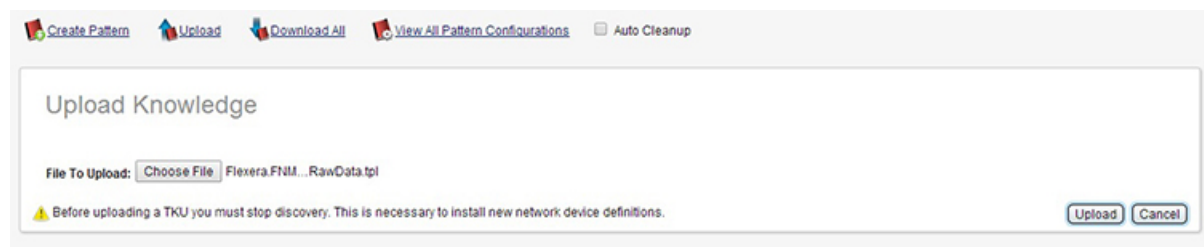
1. Log in to the BMC Discovery interface, and select the **Discovery** tab.
2. Select the **Knowledge Management** button in ADDM version 10, or BMC Discovery version 11 or later (in earlier versions of ADDM, select the **Pattern Management** button).

Figure 3: The workspace in ADDM 10



3. Click on **Upload** (or in earlier versions, **Upload New Package**).

Figure 4: Choose the file from your unzipped archive



4. Click on **Choose File**, and:
  - If you are using the *adapter*, in your unzipped archive of the adapter, select the FlexNet Manager Platform\Installers\BMC Atrium Discovery and Dependency Mapping Tools\patterns\Flexera.FNMP.InventoryRawData.tpl file.
  - If you are using the *connector*, on the inventory beacon navigate to C:\ProgramData\Flexera Software\Compliance\ImportProcedures\Inventory\Reader\BMC Discovery\patterns.zip, unzip that archive, and select the Flexera.FNMP.InventoryRawData.tpl file.
5. If you are using ADDM version 8:
  - a. Optionally modify (or accept) the default name of the package as Flexera.FNMP.InventoryRawData.
  - b. Optionally, set the description to something you will find helpful, such as FNMP export patterns.
6. Click **Upload** (or in earlier versions, **Upload & Activate**).

Ensure that the message The Requested Changes were Successful is displayed. The Flexera.FNMP.InventoryRawData pattern is now in the Active state. For further fine tuning, jump ahead



to step 4 in the following procedure.



**Tip:** For releases of BMC Atrium Discovery and Dependency Mapping up to and including 11.0, if you are using data imported through the BMC Discovery adapter to manage points-based licenses for IBM and Oracle, there is an additional optional pattern that collect further details about processors available on request from Flexera. Notice that for BMC Discovery release 11.1 or later, the additional pattern is deprecated. BMC Discovery 11.1 (and later) has improved processor information collection which makes this additional pattern unnecessary.

## Enabling Optional Patterns

IT Asset Management (Cloud)

By default, after installation some patterns are enabled and others disabled (see [Optional Patterns](#)). The following procedure allows you to turn individual patterns on and off as required, as well as enabling (or disabling) the entire module.



**To enable optional patterns within BMC Discovery:**

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that the `Flexera.FNMP.InventoryRawData` item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)
3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.

Several closed groups are displayed, one for each optional pattern and another for configuration of the FlexNet inventory agent on UNIX platforms.

5. For each group that you want to modify, select **Edit Configuration** to the right of the group. The group opens, and configurable controls are displayed. Select the **True** radio button to turn on a pattern, and the **False** radio button to turn off a pattern.

More details about configuration are included in [Appendix A: Details of Patterns](#).

6. Click **Apply**.

The modified settings are displayed.



**Tip:** If you later use the **Edit Module** button to modify the module, be sure to activate, validate, and then commit your changes.

## Rediscovering Affected Computers

IT Asset Management (Cloud)

This topic applies in both cases, whether you are configuring the adapter or the connector.

Where (as is common) you are adding these patterns to an operational BMC Discovery instance that has already taken

inventory of computers in your estate, you must rediscover any computers that are to be targeted through these new patterns, so that BMC Discovery applies to patterns to the rediscovered computers. To achieve this, you may either:

- Create (or wait for) a scheduled scan
- Initiate a Snapshot discovery scan
- Manually execute each pattern for suitably grouped targets, as summarized in the following procedure.

For more information about these options, see the BMC Discovery documentation available at <http://discovery.bmc.com/confluence/display/90/Documentation>.



#### **To manually execute the new patterns:**

1. In the BMC Discovery interface, select target computers (hosts or other nodes) by adding them to a group, creating separate groups to receive distinct types of patterns (for example, one group for Windows-related patterns, and another group for UNIX-related patterns). Create your groups using either of these approaches:
  - From a view node (including host) page, select **Groups** from the **Actions** list and add the node to a group.
  - From a report or other search result, select the required target computers. Then, select **Groups** from the **Actions** list and add the target machines to a group.
2. From the **Discovery** tab, click **Pattern Management**.
3. Select the Flexera.FNMP.InventoryRawData pattern from the package list.
4. Click the **Pattern Modules** link.
5. Select the Pattern Module containing the pattern that you want to run.
6. Click the **Pattern** link in the heading table.
7. From the **Actions** list, select **Run Pattern**.
8. In the **Run against Group** list, select the group containing your target machines for the current pattern(s).
9. Set the Expand and Execution Logging preferences for the run.
10. Set Additional Discovery to **Get all new discovery data**. This forces a new discovery.

More details about this procedure are available from <http://discovery.bmc.com/confluence/display/90/Manual+pattern+execution>.

## Account Configuration for Adapter

### IT Asset Management (Cloud)

The user name and password that the adapter needs to access the BMC Discovery XML API are defined in FnmpADDMSettings.xml (see [Installing and Configuring the Staging Tool](#)). Here we grant that account adequate rights.



#### **To configure rights for the account calling the XML API:**

1. In the BMC Discovery interface, select the **Administration** tab.

2. In the **Security** section of the **Administration** page, click the **Users** icon.
3. To create a new account in the **Users** page, click **Add** at the bottom of the page.



**Tip:** If the account has already been registered in BMC Discovery, you can select it from the list of users and review its settings.

4. Configure the adapter account, which can be a member of the **readonly** group. Ensure that the user name and password are exactly same values that you configured in `FnmpADDMSettings.xml`. For BMC Discovery 11.1 and later, the permission required to access the XML API has changed. You need to grant the new API access permission (**api-access** group), or update group permissions to include the **api-access** permission for users who require access to the XML Web APIs. Existing integrations will fail if you do not add the permission.

**Figure 5:** Sample settings for a adapter account to access BMC Discovery

Administration > Users > Add User

## Add User

Template	User
Username	exportuser
Full Name	FNMP Export User
Password	..... <input checked="" type="checkbox"/> Accepted
Verify Password	..... <input checked="" type="checkbox"/> Verified
Password Rules	Passwords must contain at least 6 characters Passwords must contain at least one uppercase character Passwords must contain at least one lowercase character Passwords must contain at least one numeric character Passwords must contain at least one non alphanumeric character Passwords may not contain sequences of more than two repeated characters
Options	<input type="checkbox"/> Force Password Change On First Login
Groups	<input type="checkbox"/> admin <input checked="" type="checkbox"/> api-access <input type="checkbox"/> appmodel <input type="checkbox"/> cmdb-export-administrator <input type="checkbox"/> discovery <input type="checkbox"/> event-source <input checked="" type="checkbox"/> never-deactivate <input type="checkbox"/> public <input checked="" type="checkbox"/> readonly <input type="checkbox"/> system <input type="checkbox"/> unlocker

5. Save your settings.

Now that you have:

- Configured the optional patterns for data collection (see [Installing Optional Patterns](#))
- Ensured that all affected inventory devices are recognized in BMC Discovery (see [Rediscovering Affected Computers](#))
- Set up the necessary account permissions (in this topic)

BMC Discovery is now configured for use of the adapter.

# 3

## Adapter Installation and Configuration

### IT Asset Management (Cloud)

The topics within this section apply only to the case where you are choosing the adapter to manage your imports from BMC Discovery.

For **cloud** implementations of IT Asset Management, you need to download the adapter archive from the Flexera Customer Community knowledge base. Although this archive also contains enhancements for on-premises implementations of the application server, there are additional components you require in the download.

You need credentials supplied by Flexera to access this download. Details of the download are included in [Download the Adapter](#).

- The build number for this adapter is 12.0 (or higher). You can identify this number by right-clicking on `FNMPADDMStage.exe`, selecting **Properties** and looking at the **Details** tab.

Full details of setting up the BMC Discovery adapter are included in the following sections.

## Choosing a Staging Server

### IT Asset Management (Cloud)

The FlexNet adapter for BMC Discovery requires a ‘staging server’ that supports installation of the adapter’s executable and configuration file. Optionally, this server may also support the staging database, or at least has fast network access to the database server providing the staging database. Several configurations are possible. The staging server may be installed on:

- A dedicated stand-alone server (or virtual machine)
- Any other suitable machine in your enterprise, such as a print server
- An inventory beacon.

The requirements for a suitable server include:

- A Windows-based operating system

- Access to an installation of Microsoft Microsoft SQL Server 2012 or later, in any edition, where the staging database may be implemented (it may be on the staging server, or on a separate database server)
- The .NET 4.5 runtime environment installed
- Internet access to the cloud implementation of IT Asset Management
- Efficient network access to each BMC Discovery server in your enterprise, using the HTTP or HTTPS protocols.



**Note:** Disconnected scenarios are also possible, using the intermediate XML files saved to disk to allow manual intervention. For more information, see [The Adapter Executable](#).

## Download the Adapter

### IT Asset Management (Cloud)

The download includes the executable for the adapter, its configuration file for you to customize, a script to create the staging database, some optimized patterns for BMC Discovery, and so on. All these resources are included in a combined downloadable archive for all adapters.



#### To download the adapter and resources:

1. Download the latest Adapter Tools for IT Asset Management `version.zip` archive from the Flexera Product and License Center:

- a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
- b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of Adapter Tools for IT Asset Management 2024 R2.3.zip commences.

- c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as `C:\temp` on a central, accessible server).

If your browser saves the file to a default location (such as your Downloads folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the downloaded zip archive, and choose **Extract All...**, saving the files in your working location.

The path of interest in the unzipped archive is `Tier 1 Adapter Tools\BMC ADDM - Atrium Discovery and Dependency Mapping Tools`, which contains three folders:

- `FnmpADDMStage`
- `patterns`
- `SQL`.

You will access the extracted archive in the following procedures, setting up and configuring the adapter and supporting infrastructure.

# Creating the Staging Database Tables

## IT Asset Management (Cloud)

Once you have selected your staging server to be used with your adapter, and that staging server can access an operating implementation of Microsoft SQL Server running a database instance you intend to use for the staging database, you should use the script provided to create the staging database and set up the appropriate database tables within it. This can be done from SQL Server Management Studio, or from the command line as described in the following procedure.



### **To create staging database tables from the command line:**

1. Navigate through the unzipped archive to Adapter Tools for IT Asset Management 2024 R2.3.zip > BMC Atrium Discovery and Dependency Mapping Tools > SQL.
2. If necessary, copy the script ADDM\_staging.sql from the \SQL folder of your unzipped adapter archive to a temporary folder on your staging server.
3. Open a command prompt on the staging server, and navigate to the folder containing the script.
4. In the command prompt window, execute the following command, as amended:

```
sqlcmd -S ServerName\InstanceName -i TemporaryPath\ADDM_staging.sql
```

where:

- *ServerName* is the name of the database server hosting the staging database, or its IP address, or “.” (dot) if you are running the staging script on the same server as the database instance
- *InstanceName* is the name of the database instance to use for the staging tables (this parameter may be omitted if the instance is the default instance)
- *TemporaryPath* is the location where you saved the SQL procedure.

Example:

```
sqlcmd -S 192.100.0.20\Development -i C:\temp\ADDM_staging.sql
```

The database ADDM\_Staging is created with all necessary tables, indices, and so on.

5. Ensure that the account under which the adapter executable will run has read/write/execute permissions on this database instance.



**Tip:** Configuration of the account is done through SQL Server Management Studio.

Authentication may be through Windows NT authentication or SQL Server authentication. Using Windows NT authentication:

- The default account is the username running the FnmpADDMStage.exe adapter
- The SQL connection is specified as a standard connection string, which you may supply in FnmpADDMSettings.xml, or override with the -c option on the command line.

The staging database is now ready for operation.

# Installing and Configuring the Staging Tool

## IT Asset Management (Cloud)

This procedure includes some separate sub-processes to complete the set-up of the adapter executable on your staging server, and arrange for regular upload of staged data.



### To install and configure the staging tool:

1. Install the adapter executable along with its configuration file.
  - a. Create a folder (suggested: ADDMAdapter) to contain the adapter executable and its configuration file.  
 Location is not critical: a suggested path is under C:\Program Files\Flexera Software (this path already exists if you have chosen to use an inventory beacon as your staging server).
  - b. From the FnmpADDMStage\ folder within your unzipped archive, copy both FnmpADDMStage.exe and FnmpADDMSettings.xml to your newly created folder (such as C:\Program Files\Flexera Software\ADDMAdapter). Both files must be in the same folder.
2. Update the configuration file (at the very minimum, correct the IP address of your BMC Discovery server).
  - a. Open your copy of FnmpADDMSettings.xml in a text editor of choice, and review the self-documenting comments within that file.
  - b. BMC has changed the permission required to access the XML API. You need to grant the new API access permission (**api-access** group), or update group permissions to include the **api-access** permission for users who require access to the XML web API.



**Important:** Existing integrations will fail if you do not add the permission.

- c. Update values in the first element describing the downstream connection to the BMC Discovery server, including the IP address, the account name and password for access.

Keep a record of the account name and password for registering with BMC Discovery (see [Account Configuration for Adapter](#)). The default values are:

```
<server protocol="http" address="10.200.20.138" username="exportuser"
password="Pa$$w0rd" timeout="3600"/>
```



**Tip:** If you do not wish to record the password in the plain text configuration file, you can use a script to retrieve the password from an encrypted store, and supply it as a command-line option when starting the FnmpADDMStage.exe tool.

- d. Update the second element for the connection to the staging database.

The default values are:

```
<database
connection-string="Server=.;Database=ADDM_Staging;Trusted_Connection=yes;"/>
```

This default assumes that the database is running on this same staging server (represented with the dot



character). If not, modify to suit your situation, using either of the server name or its IP address.

- e. Update the third element to configure whether, and where, the executable should save XML files of the inventory collected from BMC Discovery.

The default value stores any XML files below the location of the executable (but turns off storage anyway):

```
<staging path="." method="stream"/>
```

You may wish to redirect the path setting for easier access for human inspection. The `method` parameter may have any of the values `stage`, `staged` (these are two distinct values), `prestaged`, or `stream` (for details, look back to [The Adapter Executable](#)).

- f. Save your modified settings file.
3. Assuming that you do not wish to trigger the adapter manually every time it needs to run, start Windows Task Scheduler, and create a basic task to run the adapter.

The command line for the scheduled task (assuming that you have saved your preferred settings, as described above) is simply to invoke the executable. Any parameters not specified on the command line are taken from the settings file in the same folder as the executable.



**Important:** It is critical that you schedule the adapter to run at times which cannot overlap with the inventory beacon uploading the results to the central compliance server.

By default, the inventory import is triggered around midnight server time. Therefore you might consider scheduling this task for some time such as 10pm daily. Remember to allow sufficient time for inventory collection from BMC Discovery, followed by extraction from the staging database, and upload to the central application server.

4. Select an inventory beacon with a good network connection to the server hosting your staging database, and configure data extraction and upload.
  - a. Log into the FlexNet Beacon interface as an administrator, and navigate to the **Scheduling** page.
  - b. Either identify, or create, a schedule appropriate for data uploads from the staging database to the central application server.
 

For details about creating a schedule on the inventory beacon, see *FlexNet Manager Suite Help > Inventory Beacons > Scheduling Page > Creating a Data Gathering Schedule*.
  - c. In the navigation pane on the left, select the **Inventory systems** page.
  - d. Click **New...** (choosing the `SQL Server` option if you opened the drop-down) to open the **Create SQL Source Connection** dialog.
  - e. Add a **Connection Name** that is easily recognized within the first few characters (for when it is later displayed in narrow columns).
  - f. For **Source type**, choose `BMC Atrium Discovery and Dependency Mapping`.
  - g. Enter the **Server** name (or IP address) where the staging database is running. (If it is running on this inventory beacon, use the special value `(localhost)` including the parentheses.)



**Tip:** If the database instance you need is not the default one on the server you identify, add the

*instance name, separated with a backslash character. Example:*

```
(localhost)\myInstance
```

- h. Choose the authentication method, and if required add the **Username** and **Password** for accessing the database.
- i. Enter (or choose) the **Database** name.
- j. Click **Test Connection**. For more information, see the online help for this page.

This completes the configuration of the adapter executable itself, and of the upload from the staging database to the central application server. Now we can turn our attention downstream, to possible installations on target UNIX-based machines. You also need to configure enhancements to BMC Discovery itself (as discussed earlier in [Configuring BMC Discovery](#)).

## Installing the FlexNet inventory agent

### IT Asset Management (Cloud)

If you have chosen to install any of the UNIX-related collection patterns to enhance the inventory collection available through BMC Discovery, the FlexNet Inventory Agent must be copied either:

- On to each UNIX-based machine that is a target for enhanced inventory collection; or
- To a pre-configured NFS share that is accessible from each of the target UNIX-based machines.

In the following procedure, your choice of either of the above two locations is referred to as the ‘target location’.

(For background information about the FlexNet Inventory Agent, see [The FlexNet inventory agent](#). For choosing between the collection patterns, see [Optional Patterns](#). Further information about deploying the FlexNet Inventory Agent is available in the separate PDF file *Gathering FlexNet Inventory*.)



**Tip:** If you are also collecting FlexNet inventory in parallel with your use of information from BMC Discovery, it is possible that the FlexNet Inventory Agent may already be installed on target UNIX-based devices as part of that process. Check before undertaking the manual deployment outlined here.



### To install the FlexNet Inventory Agent for UNIX collection patterns:

1. On your preferred inventory beacon, locate the subdirectory that contains the ndtrack files. The default location is C:\Program Files\Flexera Software\Inventory Beacon\RemoteExecution\Public\Inventory.
2. From this folder, copy the two files ndtrack.sh and ndtrack.ini to the target location you selected above (that is, either to a pre-configured NFS file share, or to each target UNIX-based machine). The default location pre-configured in the .ini file is /opt/flexera/, but you may modify this as required.



**Important:** The file path on all individual UNIX-based machines must be identical.

3. Note the file path used (whether a file share, or the identical path used across all target devices) for entry into BMC Discovery when you are enabling the optional collection patterns (see [Optional Patterns](#) and following topics).

# Validation and Operation

## IT Asset Management (Cloud)

Normal operation of the adapter relies on the following sequence of events:

1. BMC Discovery gathers inventory using the enhanced patterns you have enabled (details: [Installing Optional Patterns](#), and [Rediscovering Affected Computers](#)).
2. At the time you scheduled ([Installing and Configuring the Staging Tool](#)), the adapter reads the current content of the BMC Discovery database and stages the new data in the staging database (previous data is first removed with a single truncate statement). The resulting status is flagged within the database.
3. Following the schedule on the central compliance server, and provided that the staging status is Success, the import reader uploads this content to the inventory database.
4. The next inventory import brings the final data set into the compliance database, where it is automatically taken into account for compliance calculations. As always, the inventory records must be recognized by the Application Recognition Library, and you must have the resulting application records linked to the appropriate license, for compliance calculations to proceed.



### **To validate operation of the adapter:**

1. Wait until BMC Discovery collects new inventory, so that the enhanced collection patterns are exercised. For further details, see the BMC Discovery documentation.
2. Manually trigger the adapter executable.

By specifying a different method parameter in the command line, you have a one-time override of the default you set in the settings XML file. For example:

```
C:\Program Files\Flexera Software\ADDMAAdapter\FnmpADDMStage.exe
-f "C:\temp" -m staged
```

This will write XML files under your C:\temp directory for review. As well, it writes data into the staging database.

3. Inspect the saved XML files to validate the inventory gathered.
4. Use SQL Server Management Studio to validate that the data is written to the staging database. Also review the StagingState property in the ADDMStagingDatabaseConfiguration table in the staging database. Possible values are Running, Failed, or Success. This value must be Success before the BMC Discovery data can be uploaded from the staging database to the central inventory database.
5. Wait (for example, overnight) until the next inventory import and calculations have run.
6. Use IT Asset Management to validate that new evidence has been recovered. Identify which evidence has been recognized by the ARL and which new rules are required. Link the applications to appropriate licenses.

## 4

# Connector Configuration

## IT Asset Management (Cloud)

Accounts used to connect to the REST APIs of BMC Discovery (as this connector does) do not use passwords. Instead, they use a bearer token that allows the account to make API calls. For this reason, before setting up the connector in the FlexNet Beacon interface, you need to correctly configure the account details in BMC Discovery, and obtain the bearer token.



### To configure FlexNet Beacon to connect with BMC Discovery:

1. First, collect the bearer token:
  - a. In the BMC Discovery interface, select the **Administration** tab.
  - b. In the **Security** section of the **Administration** page, click the **Users** icon.
  - c. If you need to create a new account in the **Users** page, click **Add** at the bottom of the page. (Otherwise, locate the appropriate account within the list of accounts.)
  - d. In the **Add User** page, complete the details for the account:
    - For **Template**, the **API Access** is appropriate for the connector account. (This choice disables the password-related fields.)
    - For **Username**, create a convenient account name such as FNMSConnector.



**Tip:** There is no requirement for this username to match the account name used on the inventory beacon to trigger the connector.

- For **Full Name**, enter a useful value – perhaps your own name as the contact person for maintaining this account.
    - When done, click **OK** to save your changes.
  - e. In the list of users, locate this account, and on the right-hand end of the row, expand the **Actions** drop-down and select **Generate API Token**.
  - f. In the resulting dialog displaying the bearer token, click **Copy to Clipboard**.
  - g. For safety, paste the value into a Notepad document or similar.

2. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.

3. Select the **Inventory systems** page and click the down arrow on the right of the **New...** split button, and choose **PowerShell**.



**Tip:** Alternatively, you can edit a connection you have defined previously, by selecting it from the list of connections and clicking **Edit...**

4. In the dialog that appears, complete (or modify) the following required fields:
  - **Connection Name:** The name you give this inventory connection is also used in the web interface of IT Asset Management to name the data import task. The name may contain alphanumeric characters, underscores or spaces, but it must start with either a letter or a number, for example BMC Discovery.
  - **Source Type:** Select BMC Discovery from the drop-down list.
5. Optionally, if your enterprise uses a proxy server to enable communication between FlexNet Beacon and BMC Discovery, select the **Use Proxy** check box, and complete the following additional details:
  - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`. If the protocol is omitted, it defaults to `http:`. If the port number is omitted, it defaults to `:80` for `http`, or `443` for `https`.
  - **Username and Password:** If your enterprise is using an authenticated proxy, specify the credentials to access the proxy server you just identified.
6. In the **BMC Discovery** section, complete the following fields:
  - a. **BMC Discovery Server:** Enter the IP address of your BMC Discovery instance, for example `10.20.300.40`. Do not include the `http://` or `https://` protocol with the IP address.
  - b. **Bearer Token:** Paste in the bearer token that you retrieved from BMC Discovery in step 1.
  - c. Optionally, select the **Import Software Key** check box (for more information, see the notes below this process).
7. Click **Test Connection** to make sure that your specifications are correct (adjusting as necessary for success).  
  
When FlexNet Beacon has successfully accessed the BMC Discovery REST APIs using the details supplied, a `Test connection succeeded` message displays.
8. Click **OK** to close this dialog, and you may also click **Save**.  
  
The new connection is added to the connections list in the **Inventory Systems** tab.
9. If you do not already have a suitable schedule for this connection available on this inventory beacon, select the **Scheduling** page and click **New...** to open the **Edit Schedule** dialog and define one.

For more details see *Scheduling a Connection* in the online help.

10. Back on the **Inventory systems** page, select the new connection in the list, click **Schedule...** to select your appropriate schedule in the **Select schedule** dialog and then click **OK** to confirm your selection.
11. In the main **FlexNet Beacon** interface, click **Save** to store your newly scheduled connection.
12. Select the newly-added connection and click **Execute Now**.

The **Execute Now** confirmation dialog displays and the import of BMC Discovery inventory data commences.

13. Click **OK** to confirm.

The resulting import (which follows immediately) may take some time to complete. You may monitor progress in the web interface for IT Asset Management — navigate to the system menu (⚙️ ▼ in the top right corner) and choose **System Health > System Tasks**, and periodically refresh this page. When the process is complete, you can inspect the inventory imported from BMC Discovery in the **All Inventory** page on IT Asset Management.



**Tip:** In the **All Inventory** page:

1. Add the **Connection name** column to the page.
2. Display the simple filter bar.
3. Filter for the name you gave your BMC Discovery connection (the example given was *BMC Discovery*).

You may also find it useful to troubleshoot using the log file on the inventory beacon. To do so:

1. On the inventory beacon, navigate to ProgramData\Flexera Software\Compliance\Logging\ComplianceReader.
2. In your preferred flat text editor, open the file `importer-digits` that was modified on today's date (or the date of the last import), and scroll to the end of the log file. The following entries indicate success:

```
[INFO] 0 source data warnings
[INFO] 0 errors, 0 warnings
[INFO] Import has been completed successfully
```

If there are warnings or errors in the log file, try to correct the issues. If you are unable to do so, save the log file and attach it to a support ticket for examination by a Flexera support engineer. Some typical early issues include:

- Incorrect details for the account name or bearer token for the connector accessing the BMC Discovery server
- Proxy or firewall settings preventing access to the Internet
- Import fails with hexadecimal character in the data stream
- Import fails with unable to create the intermediate zip file (is there sufficient disk space?).

After a successful data import, the hardware and software inventory data from BMC Discovery are visible in the appropriate pages of IT Asset Management.

### Using the Import Software Key option

The **Import Software Key** option, mentioned in step 6 in the above process, is clear (false) by default. If you select the check box (setting the option to true), two changes occur in data gathering through the connector:

- When the connector calls the API to retrieve discovered packages from BMC Discovery, it also fetches the PackageKey value

- When the connector calls the API to retrieve software instances from BMC Discovery, it also fetches the `SoftwareInstanceKey` value.

These two values are useful if you are using BMC Atrium configuration management database (CMDB), and you are using IT Visibility to enrich the collected software data. In this scenario, the process works like this:

1. BMC Discovery collects hardware and software details from devices on the network.
2. BMC Discovery writes the results into BMC Atrium CMDB. When the check box is selected, the `PackageKey` and `SoftwareInstanceKey` values are also imported into the CMDB.
3. Inventory is extracted from BMC Discovery using the connector, and is processed by IT Visibility, where the software records are both normalized and enriched (for example, data is added about the end of service life of products).
4. Subsequently, inventory is again exported from IT Visibility into BMC Atrium CMDB. As software records are processed, the `PackageKey` and `SoftwareInstanceKey` are used to match the corresponding records in the CMDB created in step 2, updating the records with normalized and enriched data from IT Visibility.

If you are *not* using BMC Atrium CMDB and passing data through IT Visibility, leave the **Import Software Key** check box clear (not selected).

# 5

## Known Issues

IT Asset Management (Cloud)

The following issue has been identified:

- UNIX-based devices that have the same host name and no detected serial number in BMC Discovery are merged into a single computer record in IT Asset Management.



# 6

## Appendix A: Details of Patterns

IT Asset Management (Cloud)

While the BMC Discovery discovery tool is a state-of-the-art tool to support ITAM, the out-of-the-box collection of inventory does not capture all of the data elements required by IT Asset Management to perform an accurate software license calculation. The optional patterns in this appendix extend those data capture capacities.

### Overview of Patterns


IT Asset Management (Cloud)

The BMC Discovery Adapter available for IT Asset Management includes six additional patterns that can be incorporated into BMC Discovery to capture these additional data attributes.

Pattern name	FlexNet Inventory Agent dependency	Default
FileEvidence	No dependency.	Tag files (all platforms): Enabled Executables (Windows only): Disabled
InstallAnywhereEvidence	None	Enabled
InstallShieldMultiplatformEvidence	None	Enabled
UnixHardwareData	Required	Disabled
WindowsLastLoggedOnUser	None	Enabled

The patterns are written in the BMC Discovery Pattern Language (TPL), for which documentation is available at <http://discovery.bmc.com/confluence/display/90/The+Pattern+Language+TPL>.

For a summary of the patterns to help decide which ones to enable or disable for your enterprise, see [Optional Patterns](#). The following sections go into more detail about each of the patterns.

 **Note:** For releases of BMC Atrium Discovery and Dependency Mapping up to and including 11.0, the optional patterns

*listed above do not provide enough detail about processors to allow management of points-based licenses for IBM and Oracle based on data collected through BMC Discovery. If you need this capability through BMC ADDM (up to release 11.0), ask your Flexera consultant to request additional optional patterns for you. Notice that for BMC Atrium Discovery and Dependency Mapping release 11.1 or later, the additional pattern is deprecated. BMC Discovery 11.1 (and later) has improved processor information collection which makes this additional pattern unnecessary.*

## FileEvidence

### IT Asset Management (Cloud)

Some applications cannot be recognized by installer package information alone. It is sometimes necessary to examine files that form part of the software installation, for either of two reasons:

- Some files are intended to provide identification details about an application, sometimes in human-readable formats
- Examining executable files installed with the application may help with identification, even though this is not their primary purpose.

Of the first class, identification files may take various forms. For example, many IBM applications are correctly identified by specific files that IBM installs for this purpose. Oracle and Adobe are among other publishers using specific files to identify some applications. Thus the Application Recognition Library (ARL) requires this file information to correctly identify such applications. ISO/IEC 19770-2 SWID tags are also increasingly available, and these ID tags are another useful form of identification file. The BMC Discovery Inventory Agent by default does not capture the complete set of identity files.

Secondly, in addition to gathering those specific files that identify an application (and have no other function in the application), it is sometimes necessary to identify installed executable files that are part of the application. These may be the only way to identify an application, such as a particular edition or version of a product. A standard implementation of BMC Discovery does not track executable files.

With this pattern, the functionality of BMC Discovery is extended to gather identification (tag) files on all platforms, and executable file evidence on Windows platforms. Note that the pattern does not search network shares or NFS mounts; nor does it follow symlinks (because of the risk of self-referencing loops). It also skips any files or folders that are inaccessible. Within these constraints, it provides details of files matching the specification and found within the defined paths on the local file system.



**Important:** *Enabling and configuring the tracking of executables within this pattern should be handled with skill and care, for two reasons:*

- *Tracking Windows executables using BMC Discovery is slow. It may be unacceptably slow to use for a wide range of directories, and you may require very targeted inventory gathering using this facility.*
- *Tracking executable files can produce a very large data set. Across large Windows server farms, the number of installation records can quickly run even into the millions, which may comprise a stress test for BMC Discovery implementations and concentrators, and (to a slightly lesser extent) for your IT Asset Management implementation.*



**Note:** *In IT Asset Management, you can use file evidence to track the usage of an application (perhaps with a view to reclaiming under-used licenses). While that functionality requires you to identify a watch-list of at least one*

*executable file for each tracked application, file evidence alone is not sufficient for usage tracking: it also requires additional FlexNet agents on the managed device, and an upload path through inventory beacons that preserves the additional usage data. File evidence gathered through the BMC Discovery adaptor is not sufficient for usage tracking.*

## Results

The file evidence gathered on Windows servers is somewhat richer than on UNIX-based servers.

- On both platforms, the pattern first collects the target directories from the pattern configuration file (under **Flexera.FNMP.InventoryRawData.FileEvidenceConfigs**).
- In the specified folders and their subfolders, the pattern retrieves:
  - All files with extensions listed under **File extensions to report as tag files** (assuming that **Report software tag files** has its default value of true). For each matching file found, BMC Discovery creates a `Detail` node linked to the host record. This happens for both Windows and UNIX-based systems.
  - On Windows only, and only when the setting for **Report executable files on Windows platforms** has been changed to true, files with a `.exe` extension from the same paths. For each executable file found, by default a WMI query is used to retrieve the file's name, version, and manufacturer. A `DiscoveredWMI` node in BMC Discovery is created for each WMI query (essentially for each file). However, because `DiscoveredWMI` nodes are ephemeral (may not survive future inventory gathering), the information is duplicated into a `Detail` node under the host server for each file discovered there.

By the appropriate means, then, a `Detail` node is created for each file evidence record, with the following properties dependent on the collection method:


Property	Value	Notes
name	File path and name	
type	FNMP_FileEvidence	
size	File size	
key	A unique key for this file, combining the values of <code>name/type/host.key</code>	
version	The release number of the file	Available only on Windows servers when executables are tracked.
company	The publisher of the application of which this file forms a part	Available only on Windows servers when executables are tracked.

## Configuration

In the `FileEvidence` pattern, you may:

- Separately enable or disable collection of:
  - Identity tag files
  - Executable files (remember to consider the potential volume of data for this option)

- Identify the file name extensions for tag files (but there is no need to identify executable file name extensions)
- Separately for Windows and UNIX-like hosts, specify the starting point(s) in the file systems for inventory scanning to begin (searches recurse through local subdirectories).

 **Important:** The BMC Discovery inventory agent stops scanning at partition boundaries, even those on the local file system. This has important implications on systems, such as IBM AIX, that typically mount key paths like `/opt` on separate partitions. You must specify a search starting point within each target partition on the file system. For example, the default values of `/opt`, `/var`, and `/usr` are well suited for inventory gathering with BMC Discovery.

All configuration items are covered in the following procedure.

## To configure the FileEvidence pattern

IT Asset Management (Cloud)

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that your current `Flexera.FNMP.InventoryRawData` item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)
3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.
5. In the **FileEvidence** group, click **Edit Configuration**.
6. Select the appropriate true/false radio button for **Report software tag files**. If you are turning off the collection of tag files, skip the next step.
7. Adjust the list of **File extensions to report as tag files**, if necessary deleting values or adding new ones that you have identified. Each extension must stand alone on its own line.
8. Select the appropriate true/false radio button for **Report executable files**. Review the discussion before this procedure while considering the data quantities implied by enabling this option.

If you have now turned off both options, so that both options now have **False** selected, you have completed the process, and may skip the next step. If either option has **True** selected, continue to define the paths for the inventory gathering.

The configuration changes are saved.

9. For both types of operating system, customize the **...scan these file paths for evidence** list.
  - Each file path must stand alone on its own line, and must be absolute (starting from root).
  - For Windows, the path must include the drive letter with its colon delimiter.
  - These same paths are scanned for identity (tag) files and for executable files.

10. Click **Apply** (at the bottom of this group).

The configuration changes are saved.

# InstallAnywhereEvidence

IT Asset Management (Cloud)

InstallAnywhere is a package installation solution developed by Flexera. Packages track their installation status in an XML file, which is interrogated by this pattern.

After collection of the inventory data, a `Detail` node is linked to the host computer for each package installed on the computer:

Property	Value
<code>name</code>	The name of the application as identified by InstallAnywhere.
<code>type</code>	<code>FNMP_InstallerEvidence</code>
<code>vendor</code>	The publisher of the application
<code>version</code>	The release number of the application.
<code>install_date</code>	The date that the application was installed on this host computer.
<code>evidence</code>	IA (fixed string literal).
<code>key</code>	A unique key for this installation record, combining (with the different literal text separators shown) the values of <code>name:version/type:evidence/host.key</code>

To configure the InstallAnywhereEvidence pattern:

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that your current Flexera.FNMP.InventoryRawData item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)



**Tip:** The `InstallAnywhereEvidence` pattern is enabled by default when the pattern collection is initially enabled. If it has previously been disabled, you can re-enable it with the remainder of this procedure.

3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.
5. In the **InstallAnywhereEvidence** group, click **Edit Configuration**.
6. Select the **True** radio button for **Report installations by InstallAnywhere**.
7. Click **Apply** (at the bottom of this group).

# InstallShieldMultiplatformEvidence

IT Asset Management (Cloud)

InstallShield Multiplatform is an older packaging solution from Flexera, in use by many software publishers. InstallShield stores software information in “vital product data” (VPD) collections, which were originally stored in flat

files and subsequently in SQL scripts.

This pattern locates either format of VPD storage (which may exist in a range of locations across different platforms), extracts the data, and creates a `Detail` node for the software installation linked to the host computer:

Property	Value
<code>name</code>	The name of the application as identified by InstallShield.
<code>type</code>	<code>FNMP_InstallerEvidence</code>
<code>vendor</code>	The publisher of the application
<code>version</code>	The release number of the application.
<code>install_date</code>	The date that the application was installed on this host computer.
<code>evidence</code>	ISMP (fixed string literal).
<code>product_code</code>	The product identification code recorded (usually) by the publisher for the particular application. This is not standardized and may be used as the publisher desires.
<code>key</code>	A unique key for this installation record, combining (with the different literal text separators shown) the values of <code>name:version/type:evidence/host.key</code>

To configure the `InstallShieldMultiplatformEvidence` pattern:

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that your current `Flexera.FNMP.InventoryRawData` item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)



**Tip:** The `InstallShieldMultiplatformEvidence` pattern is enabled by default when the pattern collection is initially enabled. If it has previously been disabled, you can re-enable it with the remainder of this procedure.

3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.
5. In the `InstallShieldMultiplatformEvidence` group, click **Edit Configuration**.
6. Select the **True** radio button for **Report installations by InstallShield Multiplatform**.
7. Click **Apply** (at the bottom of this group).

## UnixHardwareData

IT Asset Management (Cloud)

Especially for managing the corporate Data Centre, it is critical for IT Asset Management to have accurate hardware inventory for capacity-based license metrics (Processor, Core, IBM PVU, and so on). While BMC Discovery can capture this information for a Windows server, it may not consistently capture it for servers running UNIX or Linux. For example,

BMC Discovery does not currently report:

- CPUs and cores on Linux, nor cores on virtual machines
- LPARs on IBM AIX (correctly)
- Solaris resource pools.

This pattern retrieves hardware data on UNIX-based systems using the FlexNet Inventory Agent.



**Note:** The pattern must be configured with the installed location of the FlexNet Inventory Agent. The FlexNet Inventory Agent may be installed either in the same location on the file system of each target UNIX server, or on a file share accessible to all target devices. This is included in the configuration process described below.

The pattern executes the FlexNet Inventory Agent, which writes the collected data to a file in the `/var/tmp/flexera/addm/` folder on the host server. BMC Discovery then reads this output, and for each installed file, creates a `Detail` node linked to the host record:

Property	Value
Name	FNMP hardware evidence for %host.name%
Type	FNMP_HardwareEvidence
Key	%type%/ %host.key%

Property	Value
(Additional properties)	<p>Each records one of the following hardware properties:</p> <ul style="list-style-type: none"> <li>• Disk size</li> <li>• IP Address</li> <li>• MAC Address</li> <li>• Model</li> <li>• Number of cores</li> <li>• Number of disks</li> <li>• Number of logical processors</li> <li>• Number of processors</li> <li>• OS</li> <li>• Processor speed</li> <li>• Processor type</li> <li>• RAM (total physical memory)</li> <li>• Vendor.</li> </ul> <p>If the machine is found to be a virtual machine, the following additional properties are collected:</p> <ul style="list-style-type: none"> <li>• Node capacity</li> <li>• Node capacity in cores</li> <li>• Node capacity in threads</li> <li>• Physical shared pool capacity</li> <li>• Physical shared pool capacity in cores</li> <li>• Physical shared pool ID</li> <li>• Shared pool capacity</li> <li>• Shared pool capacity in cores</li> <li>• Shared pool ID</li> <li>• VM capacity</li> <li>• VM capacity in cores</li> <li>• VM entitlement</li> <li>• VM ID</li> </ul>



Property	Value
	<ul style="list-style-type: none"> <li>• VM is capped</li> <li>• VM is shared type</li> <li>• VM name</li> <li>• VM type</li> </ul>

To configure the UnixHardware pattern:

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that your current Flexera.FNMP.InventoryRawData item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)
3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.
5. In the **Flexera.FNMP.InventoryRawData.CommonFlexeraInventoryAgentConfigs** group, click **Edit Configuration**.
6. Enter the path (path only, not including the file name) to the FlexNet Inventory Agent executable. (For details about installing the FlexNet inventory agent, see [Installing the FlexNet inventory agent](#).)
7. Click **Apply** (at the bottom of this group).
8. In the **UnixHardware** group, click **Edit Configuration**.
9. Select the **True** radio button for **Report UNIX hardware properties**.
10. Click **Apply** (at the bottom of this group).

# WindowsLastLoggedInUser

IT Asset Management (Cloud)

For a Windows-based computer, standard BMC Discovery collection does not capture any inventory related to the Windows Logon. User identification is important for IT Asset Management to accurately calculate license consumption for user-based licensed, such as a Named User license. (There is no equivalent data for UNIX-like systems available through the BMC Discovery adapter.)

This pattern queries the registry at HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\LogonUI\LastLoggedInUser to find the last logged-on user (for legacy Windows platforms before Vista, it queries WMI for the value of UserName from Win32\_ComputerSystem). If the query is successful, a DiscoveredRegistryValue node is created for the host computer:

Property	Value
Name	The user name.

Property	Value
Type	FNMP_LastLoggedOnUser
Key	%type%/%host.key%

To configure the `WindowsLastLoggedOnUser` pattern:

1. In the user interface for BMC Discovery, select the **Discovery** tab.
2. Click **Pattern Management**, and from the list of patterns at the bottom of the page, ensure that your current `Flexera.FNMP.InventoryRawData` item is enabled, and select it. (The module must be enabled before any of its member patterns can operate.)



**Tip:** The `WindowsLastLoggedOnUser` pattern is enabled by default when the pattern collection is initially enabled. If it has previously been disabled, you can re-enable it with the remainder of this procedure.

3. From the **Pattern Package** properties, select the link for the **Pattern Module**.
4. Scroll down to the **Pattern Configuration** area.
5. In the **WindowsLastLoggedOnUser** group, click **Edit Configuration**.
6. Select the **True** radio button for **Report Windows last logged-on user**.
7. Click **Apply** (at the bottom of this group).

## 7

# Appendix B: Data Mappings

## IT Asset Management (Cloud)

This appendix maps the data transformations from BMC Discovery to IT Asset Management in two stages:

- From the source nodes in BMC Discovery datastore to the staging database (suggested: `ADDM_Stage`). The data constructs within this database are mirrored in the XML files that may be saved on the staging server for your inspection. For this reason, this section may also be used as a guide to understanding the sources of data within BMC Discovery that populate the various XML entities.
- From the staging database (suggested: `ADDM_Stage`) to the final destination within the web interface of IT Asset Management. This is not a simple and direct path: the `ComplianceReader` component first loads the data from the staging database to various compliance database tables with names starting with `Imported`, where the data waits until the next "data import" (usually associated with a full license consumption calculation, by default scheduled overnight). In effect, the data first moves from remote staging tables to local staging tables. Thereafter the data is normalized and loaded into a range of compliance database tables, for the most part arrayed around the `ComplianceComputer` table. Since navigating these tables is not straight-forward, we here present the data in its more visible form, within the web interface. Serious devotees of database structures can find all compliance database tables documented in *IT Asset Management Schema Reference*.

In both parts, each destination has its own topic, so that you can jump to destination names using the table of contents of this PDF (in the first of these sections, the destinations are table names within the staging database; and in the second, the destinations are "object" names presented in the web interface).

Because an individual node in BMC Discovery may link to multiple staging tables, it's not quite so easy to find the original nodes in BMC Discovery. One way is to do a search in the PDF file for "SEARCH *nodeName*", since each staging database topic begins with the Query Language statements used to extract the related data from BMC Discovery.



**Note:** This appendix documents the structure released with IT Asset Management version 2020 R1.2 (if necessary, you can validate this version by checking the operating system file properties of `FmpADDMStage.exe`). The database schema is version 1.10 (or later) for these changes. If needed, you can check the staging database schema version with the following query in Microsoft SQL Studio:

```
SELECT TOP (1000) [Property],[Value]
FROM [ADDM_Stage].[dbo].[ADDMStagingDatabaseConfiguration]
```

(If necessary, substitute your chosen staging database name in place of the proposed `ADDM_Stage`.) It is important to

*be aware that this schema update makes no change at all to the data from BMC Discovery imported into IT Asset Management. Instead, the changes to the adapter and staging database schema enable integration with the Discovery and Inventory capabilities within Flexera One, while maintaining existing imports and performance with IT Asset Management.*

## Source to Staging

IT Asset Management (Cloud)

This section provides:

- The Query Language queries exercised against BMC Discovery to extract data
- The tables in the staging database, and within each table the individual properties (database columns), where the data is staged.

For each table topic, the following details are included:

- The source alias (usually identical to the following)
- The property/column name in the staging table
- The data type in the staging table
- For string types, the maximum number of characters (except as noted, excess string length is truncated).

Each topic also notes the name of the XML file that may be saved on the staging server (depending on the command line given to the staging tool).

### ADDMNetworkDevices\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `ADDMNetworkDevices.xml` on the staging server.

 **Important:** This staging table is only used by Flexera One and is not applicable to IT Asset Management.

#### BMC Discovery query

This query extracts the known network devices from BMC Discovery, saving the details in the table below:

```
search NetworkDevice show name, type, vendor, model,
#InferredElement:Inference:Associate:DiscoveryAccess.endpoint as 'Scanned via',
serial as 'Serial ID', key as 'Key'
```

#### Mapping into ADDMNetworkDevices\_ci

Source alias	Cluster_ci property	Type	Max
Name	Name	nvarchar	255

Source alias	Cluster_ci property	Type	Max
Type	Type	nvarchar	255
Vendor	Vendor	nvarchar	255
Model	Model	nvarchar	255
Scanned via	IPAddress	nvarchar	255
Serial ID	SerialNo	nvarchar	255
Key	Key	nvarchar	255

## ADDMPrinters\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in ADDMPrinters.xml on the staging server.



**Important:** This staging table is only used by Flexera One and is not applicable to IT Asset Management.

### BMC Discovery query

This query extracts the known printers from BMC Discovery, saving the details in the table below:

```
search Printer show name, type, vendor, model,
#InferredElement:Inference:Associate:DiscoveryAccess.endpoint as 'Scanned via',
serial as 'Serial ID', key as 'Key'
```

### Mapping into ADDMPrinters\_ci

Source alias	Cluster_ci property	Type	Max
Name	Name	nvarchar	255
Type	Type	nvarchar	255
Vendor	Vendor	nvarchar	255
Model	Model	nvarchar	255
Scanned via	IPAddress	nvarchar	255
Serial ID	SerialNo	nvarchar	255
Key	Key	nvarchar	255

## ADDMVersion\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `ADDMVersion.xml` on the staging server.

### BMC Discovery query

This query extracts the version information for BMC Discovery, saving the details in the table below:

```
LOOKUP Version
```

### Mapping into ADDMVersion\_ci

Source alias	ADDMVersion_ci property	Type	Max
Version	Version	varchar	32
Release	Release	varchar	32
Release Name	Release Name	varchar	32

## Cluster\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `Cluster.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Cluster` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Cluster
  SHOW
    key AS 'ClusterKey',
    type AS 'ClusterType',
    (type = 'vCenter Cluster' AND
      (single(#ManagedElement:Management:Manager:SoftwareInstance.instance)
        + '|' + id)
    OR NONE) AS 'InternalIdentifier',
    (cluster_name OR name) AS 'InternalName',
    single(#HostContainer:HostContainment:ContainedHost:Host.domain)
      AS 'Domain',
    formatTime(modified(#), "%FT%T") AS 'LastUpdate'
```

## Mapping into Cluster\_ci

Source alias	Cluster_ci property	Type	Max
ClusterKey	ClusterKey	nvarchar	400
ClusterType	ClusterType	nvarchar	255
Domain	Domain	nvarchar	512
InternalIdentifier	InternalIdentifier	nvarchar	512
InternalName	InternalName	nvarchar	512
LastUpdate	LastUpdate	datetime	

## ClusterHost\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `ClusterHost.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Cluster` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Cluster
  WHERE (NODECOUNT
    (TRAVERSE HostContainer:HostContainment:ContainedHost:Host) > 0)
  SHOW
    key AS 'ClusterKey',
    EXplode #HostContainer:HostContainment:ContainedHost:Host.key
      AS 'HostKey'
```

## Mapping into ClusterHost\_ci

Source alias	ClusterHost_ci property	Type	Max
ClusterKey	ClusterKey	nvarchar	400
HostKey	HostKey	varchar	64

## CPUInformationDetail\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `CPUInformationDetail.xml` on the staging server.

## BMC Discovery query

This query extracts data from the Detail node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below.



**Tip:** This data relies on a custom CPU pattern available through your Flexera consultant. Be aware that from BMC Discovery release 11.1, BMC provided improved data gathering with new ProcessorInfo functionality. However, this does not gather the full dataset enabled through the custom pattern, and it remains best practice to combine the data gathered by BMC Discovery with additional data available through the custom pattern.

SEARCH Detail

```
WHERE (NODECOUNT(TRAVERSE Detail:Detail:ElementWithDetail:Host) > 0)
AND (type = 'CPU Information')
SHOW
host_partitioning_type AS 'HostType',
(
  (host_partitioning_type = 'physical' AND total_logical_cpus
  OR allocated_threads) OR none
) AS 'LogicalProcessors',
(
  (host_partitioning_type = 'physical' AND total_cpu_cores
  OR involved_cores) OR none
) AS 'Cores',
(
  (host_partitioning_type = 'physical' AND total_cpu_sockets
  OR involved_sockets) OR none
) AS 'Processors',
(total_logical_cpus OR none) AS 'PhysicalLogicalProcessors',
(total_cpu_cores OR none) AS 'PhysicalCores',
(total_cpu_sockets OR none) AS 'PhysicalProcessors',
zone_pool_name AS 'ZonePoolName',
lpar_id_to_partition AS 'LPARIdToPartition',
lpar_id_to_system AS 'LPARIdToSystem',
(
  (lpar_partition_number <> '-' AND lpar_partition_number) OR none
) AS 'LPARPartitionNumber',
lpar_partition_name AS 'LPARPartitionName',
lpar_type AS 'LPARType',
lpar_mode AS 'LPARMode',
(lpar_active_physical_cpus_in_system OR none)
  AS 'LPARActivePhysicalCpusInSystem',
(lpar_shared_physical_cpus_in_system OR none)
  AS 'LPARSharedPhysicalCpusInSystem',
((lpar_shared_pool_id <> '-' AND lpar_shared_pool_id) OR none)
  AS 'LPARSharedPoolID',
(lpar_active_cpus_in_pool OR none) AS 'LPARActiveCpusInPool',
(lpar_entitled_capacity OR none) AS 'LPAREntitledCapacity',
#Detail:Detail:ElementWithDetail:Host.key AS 'HostKey'
```



## Mapping into CPUInformationDetail\_ci

Source alias	CPUInformationDetail_ci property	Type	Max
Cores	Cores	int	
HostKey	HostKey	varchar	64
HostType	HostType	varchar	32
LogicalProcessors	LogicalProcessors	int	
LPARActiveCpusInPool	LPARActiveCpusInPool	int	
LPARActivePhysicalCpusInSystem	LPARActivePhysicalCpusInSystem	int	
LPAREntitledCapacity	LPAREntitledCapacity	float	
LPARIdToPartition	LPARIdToPartition	nvarchar	200
LPARIdToSystem	LPARIdToSystem	nvarchar	200
LPARMode	LPARMode	nvarchar	255
LPARPartitionName	LPARPartitionName	nvarchar	255
LPARPartitionNumber	LPARPartitionNumber	int	
LPARSharedPhysicalCpusInSystem	LPARSharedPhysicalCpusInSystem	int	
LPARSharedPoolID	LPARSharedPoolID	int	
LPARType	LPARType	nvarchar	255
PhysicalCores	PhysicalCores	int	
PhysicalLogicalProcessors	PhysicalLogicalProcessors	int	
PhysicalProcessors	PhysicalProcessors	int	
Processors	Processors	int	
ZonePoolName	ZonePoolName	nvarchar	200

## DiscoveredInstalledPackages\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `DiscoveredInstalledPackages.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Host` node in the BMC Discovery datastore to find installed packages, and provides the aliases for properties referenced in the table below:

```

SEARCH Host
  STEP IN Host:HostedSoftware:InstalledSoftware:Package
  SHOW
    #:InstalledSoftware:Package.key AS 'PackageKey',
    #:Host:Host.key as 'HostKey'

```

### Mapping into DiscoveredInstalledPackages\_ci

Source alias	DiscoveredPackages_ci property	Type	Max
PackageKey	PackageKey	nvarchar	450
HostKey	HostKey	varchar	64

## DiscoveredPackages\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `DiscoveredPackages.xml` on the staging server.



**Tip:** The `PackageKey` property is ignored for IT Asset Management.

### BMC Discovery query

This query extracts data from the `Package` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```

SEARCH Package
  SHOW
    key AS 'PackageKey',
    name AS 'Package',
    version AS 'Version',
    vendor AS 'Vendor'

```

### Mapping into DiscoveredPackages\_ci

Source alias	DiscoveredPackages_ci property	Type	Max
PackageKey	PackageKey	varchar	450
Package	Package	nvarchar	255
Vendor	Vendor	nvarchar	255
Version	Version	nvarchar	255

## DiscoveredService\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `DiscoveredService.xml` on the staging server.

### BMC Discovery query

This query extracts data from the Host node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Host
  TRAVERSE InferredElement:Inference:Associate:DiscoveryAccess
    WHERE end_state = 'GoodAccess'
  TRAVERSE DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:ServiceList
  TRAVERSE List:List:Member:DiscoveredService
    WHERE name = 'vmms' AND display_name = 'Hyper-V Virtual Machine Management'
SHOW
  name AS 'Name',
  #Member:List:List:ServiceList.#DiscoveryResult:DiscoveryAccessResult:
    DiscoveryAccess:DiscoveryAccess.#Associate:Inference:
    InferredElement:Host.key AS 'HostKey'
```

### Mapping into DiscoveredService\_ci

Source alias	DiscoveredService_ci property	Type	Max
HostKey	HostKey	varchar	64
Name	Name	nvarchar	255

## DiscoveredVirtualMachine\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `DiscoveredVirtualMachine.xml` on the staging server.

### BMC Discovery query

This query extracts data from the Host node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Host
  TRAVERSE InferredElement:Inference:Associate:DiscoveryAccess
    WHERE end_state = 'GoodAccess'
  TRAVERSE DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:VirtualMachineList
    WHERE access_method IS DEFINED
  TRAVERSE List:List:Member:DiscoveredVirtualMachine
SHOW
```

```

name AS 'Name',
uuid AS 'UUID',
config_os_full AS 'ConfigOSFull',
power_state AS 'PowerState',
vsphere_id AS 'vSphereID',
#Member:List:List:VirtualMachineList.access_method AS 'AccessMethod',
formatTime(modified(#), "%FT%T") AS 'LastUpdate',
#Member:List:List:VirtualMachineList.#DiscoveryResult:DiscoveryAccessResult:
  DiscoveryAccess:DiscoveryAccess.#Associate:
    Inference:InferredElement:Host.key AS 'HostKey'

```

### Mapping into DiscoveredVirtualMachine\_ci

Source alias	DiscoveredVirtualMachine_ci property	Type	Max
AccessMethod	AccessMethod	nvarchar	255
ConfigOSFull	ConfigOSFull	nvarchar	512
HostKey	HostKey	varchar	64
LastUpdate	LastUpdate	datetime	
Name	Name	nvarchar	512
PowerState	PowerState	nvarchar	255
UUID	UUID	nvarchar	255
vSphereID	vSphereID	nvarchar	255

## FileEvidenceDetail\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in FileEvidenceDetail.xml on the staging server.

### BMC Discovery query

This query extracts data from the Detail node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```

SEARCH Detail
WHERE (NODECOUNT(TRAVERSE Detail:Detail:ElementWithDetail:Host) > 0)
AND (type = 'FNMP_FileEvidence')
SHOW name AS 'FileName',
size AS 'FileSize',
description AS 'Description',
company AS 'Company',
version AS 'FileVersion',

```

```
#Detail:Detail:ElementWithDetail:Host.key AS 'HostKey'
```

### Mapping into FileEvidenceDetail\_ci

Source alias	FileEvidenceDetail_ci property	Type	Max
Company	Company	nvarchar	255
Description	Description	nvarchar	255
FileName	FileName	nvarchar	800
FileSize	FileSize	bigint	
FileVersion	FileVersion	nvarchar	255
HostKey	HostKey	varchar	64

## FileSystem\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `FileSystem.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `FileSystem` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH FileSystem
WHERE (NODECOUNT
      (TRAVERSE MountedFileSystem:FileSystemMount:Mounter:Host) > 0)
AND fs_kind = 'LOCAL'
AND size IS DEFINED
AND size > 0
AND fs_type <> 'tmpfs'
AND fs_type <> 'devtmpfs'
AND fs_type <> 'DevFS'
SHOW
  fs_name AS 'Name',
  size AS 'Size',
  #MountedFileSystem:FileSystemMount:Mounter:Host.key AS 'HostKey'
```

### Mapping into FileSystem\_ci

Source alias	FileSystem_ci property	Type	Max
HostKey	HostKey	varchar	64
Name	Name	nvarchar	255

Source alias	FileSystem_ci property	Type	Max
Size	Size	bigint	

## HardwareEvidenceDetail\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `HardwareEvidenceDetail.xml` on the staging server.

This data is generated through use of the optional `UnixHardwareData` pattern, which triggers the Flexera inventory component `ndtrack` ("the tracker"). Before staging, the information is visible in the `Detail` node of BMC Discovery, with a type of `FNMP_HardwareEvidence`.

### BMC Discovery query

This query extracts data from the `Detail` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Detail
WHERE (NODECOUNT(TRAVERSE Detail:Detail:ElementWithDetail:Host) > 0)
AND (type = 'FNMP_HardwareEvidence')
SHOW
os AS 'OS',
num_processors AS 'Processors',
num_cores AS 'Cores',
num_logical_processors AS 'LogicalProcessors',
processor_type AS 'ProcessorType',
processor_speed AS 'ProcessorSpeed',
ram AS 'RAM',
vendor AS 'Vendor',
model AS 'Model',
ip_address AS 'IPAddress',
mac_address AS 'MACAddress',
num_disks AS 'NumDisks',
disk_size AS 'DiskSize',
vm_type AS 'VMType',
node_capacity AS 'NodeCapacity',
node_capacity_in_cores AS 'NodeCapacityInCores',
node_capacity_in_threads AS 'NodeCapacityInThreads',
physical_shared_pool_capacity AS 'PhysicalSharedPoolCapacity',
physical_shared_pool_capacity_in_cores
  AS 'PhysicalSharedPoolCapacityInCores',
physical_shared_pool_id AS 'PhysicalSharedPoolID',
shared_pool_capacity AS 'SharedPoolCapacity',
shared_pool_capacity_in_cores AS 'SharedPoolCapacityInCores',
shared_pool_id AS 'SharedPoolID',
vm_capacity AS 'VMCapacity',
vm_capacity_in_cores AS 'VMCapacityInCores',
```

```

vm_entitlement AS 'VMEntitlement',
vm_id AS 'VMID',
vm_is_capped AS 'VMIsCapped',
vm_is_shared_type AS 'VMIsSharedType',
vm_name AS 'VMName',
partition_id AS 'PartitionID',
partition_number AS 'PartitionNumber',
#Detail:Detail:ElementWithDetail:Host.key AS 'HostKey'

```

## Mapping into HardwareEvidenceDetail\_ci

Source alias	HardwareEvidenceDetail_ci property	Type	Max
Cores	Cores	smallint	
DiskSize	DiskSize	bigint	
HostKey	HostKey	varchar	64
IPAddress	IP	nvarchar	512
LogicalProcessors	LogicalProcessors	smallint	
MACAddress	MAC	nvarchar	512
Model	Model	nvarchar	512
NodeCapacity	NodeCapacity	smallint	
NodeCapacityInCores	NodeCapacityInCores	smallint	
NodeCapacityInThreads	NodeCapacityInThreads	smallint	
NumDisks	NumDisks	int	
OS	OS	nvarchar	256
PartitionID	PartitionID	nvarchar	512
PartitionNumber	PartitionNumber	int	
PhysicalSharedPoolCapacity	PhysicalSharedPoolCapacity	smallint	
PhysicalSharedPoolCapacityInCores	PhysicalSharedPoolCapacityInCores	smallint	
PhysicalSharedPoolID	PhysicalSharedPoolID	nvarchar	255
Processors	Processors	smallint	
ProcessorSpeed	ProcessorSpeed	int	
PhysicalLogicalProcessors	PhysicalLogicalProcessors	int	

Source alias	HardwareEvidenceDetail_ci property	Type	Max
ProcessorType	ProcessorType	nvarchar	128
RAM	RAM	bigint	
SharedPoolCapacity	SharedPoolCapacity	smallint	
SharedPoolCapacityInCores	SharedPoolCapacityInCores	smallint	
SharedPoolID	SharedPoolID	nvarchar	255
Vendor	Vendor	nvarchar	255
VMCapacity	VMCapacity	float	
VMCapacityInCores	VMCapacityInCores	float	
VMEntitlement	VMEntitlement	float	
VMID	VMID	nvarchar	255
VMIsCapped	VMIsCapped	bit	
VMIsSharedType	VMIsSharedType	bit	
VMName	VMName	nvarchar	512
VMType	VMType	nvarchar	128

## Host\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in Host.xml on the staging server.

### BMC Discovery query

This query extracts data from the Host node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below.



**Tip:** The following properties are recent additions to BMC Discovery:

- `num_cores` is available from BMC Discovery release 11.0
- `zone_pool_name` is available from BMC Discovery release 11.1.

```
SEARCH Host
SHOW
hostname AS 'Hostname',
(dns_domain OR domain) AS 'Domain',
os AS 'OS',
os_type AS 'OSType',
```



```

os_class AS 'OSClass',
os_edition AS 'OSEdition',
os_version AS 'OSVersion',
os_vendor AS 'OSVendor',
os_build AS 'OSBuild',
service_pack AS 'ServicePack',
uuid AS 'UUID',
(
    num_processors
    OR nodecount(traverse Host:Detail:Hardware:ProcessorInfo)
) AS 'Processors',
(
    num_cores
    OR ((num_processors > 0 AND cores_per_processor > 0)
        AND (num_processors * cores_per_processor))
    OR ((num_logical_processors > 0 AND threads_per_core > 0)
        AND (num_logical_processors/threads_per_core))
    OR none
) AS 'Cores',
num_logical_processors AS 'LogicalProcessors',
processor_type AS 'ProcessorType',
processor_speed AS 'ProcessorSpeed',
ram AS 'RAM',
vendor AS 'Vendor',
model AS 'Model',
serial AS 'Serial',
hostid AS 'HostID',
partition_id AS 'PartitionID',
lpar_name AS 'LPARName',
lpar_partition_number AS 'LPARNumber',
wparid AS 'WPARID',
npar_partition_number AS 'NPARNumber',
vpar_partition_number AS 'VPARNumber',
ldom_name AS 'LDOMName',
ldom_role AS 'LDMRole',
zonename AS 'Zonename',
zone_uuid AS 'ZoneUUID',
(zone_pool_name OR none) AS 'ZonePoolName',
formatTime(modified(#), "%FT%T") as 'ModifiedDate',
key AS 'HostKey'

```

## Mapping into Host\_ci

Source alias	Host_ci property	Type	Max
Cores	Cores	smallint	
Domain	Domain	nvarchar	255

Source alias	Host_ci property	Type	Max
HostID	HostID	nvarchar	255
HostKey	HostKey	varchar	64
Hostname	Hostname	nvarchar	255
LDOMName	LDOMName	nvarchar	255
LDOMRole	LDOMRole	nvarchar	255
LogicalProcessors	LogicalProcessors	smallint	
LPARName	LPARName	nvarchar	255
LPARNumber	LPARNumber	int	
Model	Model	nvarchar	512
ModifiedDate	ModifiedDate	datetime	
NPARNumber	NPARNumber	nvarchar	64
OS	OS	nvarchar	1024
OSClass	OSClass	nvarchar	64
OSBuild	OSBuild	nvarchar	255
OSEdition	OSEdition	nvarchar	255
OSType	OSType	nvarchar	255
OSVendor	OSVendor	nvarchar	255
OSVersion	OSVersion	nvarchar	255
PartitionID	PartitionID	nvarchar	255
Processors	Processors	smallint	
ProcessorSpeed	ProcessorSpeed	int	
ProcessorType	ProcessorType	nvarchar	128
RAM	RAM	bigint	
Serial	Serial	nvarchar	255
ServicePack	ServicePack	nvarchar	128
UUID	UUID	nvarchar	255
Vendor	Vendor	nvarchar	255
VPARNumber	VPARNumber	nvarchar	64
WPARID	WPARID	int	
Zonename	Zonename	nvarchar	255

Source alias	Host_ci property	Type	Max
ZonePoolName	ZonePoolName	nvarchar	255
ZoneUUID	ZoneUUID	nvarchar	255

## HostContainerProcessorInfo\_ci (Staging Table)

IT Asset Management (Cloud)

This data is available only from ADDM 11.0 and later releases.

Data in this staging table may also be saved in HostContainerProcessorInfo.xml on the staging server.

### BMC Discovery query

This query extracts data from the ProcessorInfo node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH ProcessorInfo
WHERE NODECOUNT(TRAVERSE Hardware:Detail:HostContainer:HostContainer) > 0
SHOW
  #Hardware:Detail:HostContainer:HostContainer.num_logical_processors
  AS 'LogicalProcessors',
  #Hardware:Detail:HostContainer:HostContainer.num_cores AS 'Cores',
  #Hardware:Detail:HostContainer:HostContainer.num_processors
  AS 'Processors',
EXPLODE #Hardware:Detail:HostContainer:HostContainer.#HostContainer:
  HostContainment:ContainedHost:Host.#InferredElement:Inference:Associate:
  DiscoveryAccess.#DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:
  HostInfo.systemid AS 'SystemID',
EXPLODE #Hardware:Detail:HostContainer:HostContainer.#HostContainer:
  HostContainment:ContainedHost:Host.hostid AS 'HostID',
  #Hardware:Detail:HostContainer:HostContainer.#HostContainer:HostContainment:
  ContainedHost:Host.vendor AS 'Vendor'
```

### Mapping into HostContainerProcessorInfo\_ci

Source alias	HostContainerProcessorInfo_ci property	Type	Max
Cores	Cores	smallint	
HostID	HostID	nvarchar	255
LogicalProcessors	LogicalProcessors	smallint	
Processors	Processors	smallint	
SystemID	SystemID	nvarchar	255

Source alias	HostContainerProcessorInfo_ci property	Type	Max
Vendor	Vendor	nvarchar	255

## HostDetail\_ci (Staging Table)

IT Asset Management (Cloud)

This data is available only from BMC Discovery 11.1 and later releases. The latest Technology Knowledge Update must be applied to the BMC Discovery 11.1 release.

Data in this staging table may also be saved in `HostDetail.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Detail` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Detail
WHERE type = 'LPAR Resource'
SHOW
  id_to_partition as 'LPARIdToPartition',
  id_to_system as 'LPARIdToSystem',
  #Detail:Detail:ElementWithDetail:Host.key as 'HostKey'
```

### Mapping into HostDetail\_ci

Source alias	HostDetail_ci property	Type	Max
HostKey	HostKey	varchar	64
LPARIdToPartition	LPARIdToPartition	nvarchar	200
LPARIdToSystem	LPARIdToSystem	nvarchar	200

## HostInfo\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `HostInfo.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Host` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Host
TRAVERSE InferredElement:Inference:Associate:DiscoveryAccess
WHERE end_state = 'GoodAccess'
```

```

TRaverse DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:HostInfo
WHERE NODECOUNT(Traverse Primary:Inference:InferredElement:Host) > 0
SHOW
systemid AS 'SystemID',
lpar_partition_number AS 'LPARPartitionNumber',
lpar_partition_name AS 'LPARPartitionName',
lpar_type AS 'LPARType',
lpar_mode AS 'LPARMode',
lpar_active_physical_cpus_in_system AS 'LPARActivePhysicalCpusInSystem',
lpar_shared_physical_cpus_in_system AS 'LPARSharedPhysicalCpusInSystem',
lpar_shared_pool_id AS 'LPARSharedPoolID',
lpar_active_cpus_in_pool AS 'LPARActiveCpusInPool',
lpar_entitled_capacity AS 'LPAREntitledCapacity',
formatTime(#DiscoveryResult:DiscoveryAccessResult:DiscoveryAccess:
           DiscoveryAccess.starttime,"%FT%T") as 'InventoryDate',
#Primary:Inference:InferredElement:Host.key AS 'HostKey'

```

### Mapping into HostInfo\_ci

Source alias	HostInfo_ci property	Type	Max
HostKey	HostKey	varchar	64
InventoryDate	InventoryDate	datetime	
LPARActiveCpusInPool	LPARActiveCpusInPool	int	
LPARActivePhysicalCpusInSystem	LPARActivePhysicalCpusInSystem	int	
LPAREntitledCapacity	LPAREntitledCapacity	float	
LPARMode	LPARMode	nvarchar	255
LPARPartitionName	LPARPartitionName	nvarchar	255
LPARPartitionNumber	LPARPartitionNumber	int	
LPARSharedPhysicalCpusInSystem	LPARSharedPhysicalCpusInSystem	int	
LPARSharedPoolID	LPARSharedPoolID	int	
LPARType	LPARType	nvarchar	255
SystemID	SystemID	nvarchar	255

## InstallerEvidenceDetail\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `InstallerEvidenceDetail.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Detail` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH Detail
WHERE (NODECOUNT(TRAVERSE Detail:Detail:ElementWithDetail:Host) > 0)
AND (type = 'FNMP_InstallerEvidence')
SHOW name AS 'Name',
version AS 'Version',
vendor AS 'Vendor',
install_date AS 'InstallDate',
evidence AS 'Evidence',
product_code AS 'ProductCode',
#Detail:Detail:ElementWithDetail:Host.key AS 'HostKey'
```

### Mapping into InstallerEvidenceDetail\_ci

Source alias	InstallerEvidenceDetail_ci property	Type	Max
Evidence	Evidence	nvarchar	64
HostKey	HostKey	varchar	64
InstallDate	InstallDate	nvarchar	255
Name	Name	nvarchar	512
ProductCode	ProductCode	nvarchar	110
Vendor	Vendor	nvarchar	400
Version	Version	nvarchar	144

## LastLoggedOnUserDetail\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `LastLoggedOnUserDetail.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `Detail` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```

SEARCH Detail
WHERE (NODECOUNT(TRAVERSE Detail:Detail:ElementWithDetail:Host) > 0)
AND (type = 'FNMP_LastLoggedOnUser')
SHOW name AS 'UserName',
#Detail:Detail:ElementWithDetail:Host.key AS 'HostKey'

```

### Mapping into LastLoggedOnUserDetail\_ci

Source alias	LastLoggedOnUserDetail_ci property	Type	Max
HostKey	HostKey	varchar	64
UserName	UserName	nvarchar	255

## NetworkInterface\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `NetworkInterface.xml` on the staging server.

### BMC Discovery query

This query extracts data from the `NetworkInterface` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```

SEARCH NetworkInterface
WHERE (NODECOUNT
      (TRAVERSE InterfaceOfDevice:DeviceInterface:
        DeviceWithInterface:Host) > 0)
SHOW
(ip_addr
OR (NODECOUNT(TRAVERSE InterfaceWithAddress:InterfaceAddress:
  IPv4Address:IPAddress) > 0
AND fmt("%.510s", join(#InterfaceWithAddress:InterfaceAddress:
  IPv4Address:IPAddress.ip_addr, ', ')))
OR ''
) AS 'IP',
mac_addr AS 'MAC',
#InterfaceOfDevice:DeviceInterface:DeviceWithInterface:Host.key
AS 'HostKey'

```

### Mapping into NetworkInterface\_ci

Source alias	NetworkInterface_ci property	Type	Max
HostKey	HostKey	varchar	64
IP	IP	nvarchar	512

Source alias	NetworkInterface_ci property	Type	Max
MAC	MAC	nvarchar	512

## SoftwareInstance\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `SoftwareInstance.xml` on the staging server.



**Tip:** The `SoftwareInstanceKey` property is ignored for IT Asset Management.

### BMC Discovery query

This query extracts data from the `SoftwareInstance` node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below:

```
SEARCH SoftwareInstance
WHERE (NODECOUNT(TRAVERSE RunningSoftware:HostedSoftware:Host:Host) > 0)
AND vm_type IS NOT DEFINED
SHOW
type + (edition AND (' ' + edition) OR '') AS 'Name',
product_version AS 'Version',
(publisher OR single(#Element:Maintainer:Pattern:Pattern.publishers))
AS 'Publisher',
EXPLODE #RunningSoftware:HostedSoftware:Host:Host.key AS 'HostKey',
key AS 'SoftwareInstanceKey'
```

### Mapping into SoftwareInstance\_ci

Source alias	SoftwareInstance_ci property	Type	Max
HostKey	HostKey	varchar	64
Name	Name	nvarchar	255
Version	Version	nvarchar	255
Publisher	Publisher	nvarchar	255
SoftwareInstanceKey	SoftwareInstanceKey	nvarchar	512

## SoftwareInstanceVirtualMachine\_ci (Staging Table)

IT Asset Management (Cloud)

Data in this staging table may also be saved in `SoftwareInstanceVirtualMachine.xml` on the staging server.



## BMC Discovery query

This query extracts data from the SoftwareInstance node in the BMC Discovery datastore, and provides the aliases for properties referenced in the table below (note that the regex is wrapped for printing, and should appear all on one line):

```
SEARCH SoftwareInstance
WHERE (NODECOUNT(TRAVERSE RunningSoftware:HostedSoftware:Host:Host) > 0)
AND vm_type IS DEFINED
SHOW
vm_type AS 'VMType',
(zone_uuid OR vm_bios_serial) AS 'VMSerial',
(zonename OR vm_name) AS 'VMName',
(zoneid OR
  (vm_type = 'VMware Virtual Machine'
    AND extract(vm_uuid, regex "^(..) (..) (..) (..) (..) (..) (..)
                        (..)-(..) (..) (..) (..) (..) (..)
                        (..) (..)$$",
    regex "\1\2\3\4-\5\6-\7\8-\9\10-\11\12\13\14\15\16")
    OR vm_uuid
  )
) AS VMID,
vm_guest_os AS 'GuestFullName',
zonepath AS 'VMPPath',
formatTime(last_update_success, "%FT%T") as 'LastUpdateSuccess',
single(#HostContainer:HostContainment:ContainedHost:Host.key)
  AS 'GuestHostKey',
single(#RunningSoftware:HostedSoftware:Host:Host.key) AS 'HostKey'
```

## Mapping into SoftwareInstanceVirtualMachine\_ci

Source alias	SoftwareInstanceVirtualMachine_ci property	Type	Max
GuestFullName	GuestFullName	nvarchar	512
GuestHostKey	GuestHostKey	varchar	64
HostKey	HostKey	varchar	64
LastUpdateSuccess	LastUpdateSuccess	datetime	
VMID	VMID	nvarchar	255
VMName	VMName	nvarchar	512
VMPPath	VMPPath	nvarchar	512
VMSerial	VMSerial	nvarchar	255
VMType	VMType	nvarchar	255

# Staging to IT Asset Management

## IT Asset Management (IT Asset Management)

This section provides the mapping from the staging database to the final presentation in the web interface for IT Asset Management.

This section is organized alphabetically by object names in IT Asset Management (for example, 'computers' are known as 'inventory devices'). This is useful when you are starting from data displayed in IT Asset Management, tracing it back to the staging database, and then using the preceding section to see the source of that data point in BMC Discovery.

If you are coming the other direction, it is easiest to use your PDF search tool to find the staging database table name, a dot separator, and the property name. For example, to find out where the `Hostname` property from the `Host_ci` table of the staging database ends up, search for `Host_ci.Hostname`. You will find it listed against the **Name** property in the **All Inventory** page (and other similar listings) of the web interface.



**Remember:** You cannot trace the contents of the `ADDMNetworkDevices_ci` or `ADDMPrinters_ci` staging tables forward in this way, since these two tables are ignored for IT Asset Management, and are imported only for Flexera One.

For each object-based topic, the following details are included:

- Whether the import of new data that was not previously in the compliance database creates new records on import.
- Whether the import of changed data for records with matching key values is allowed to update the compliance database.
- Whether the absence of records previously imported from *only* the BMC Discovery source causes the old records to be removed from the compliance database.
- The source table and property (in dot-separated format) from the staging database. Look back to the previous section to check data types and maximum string sizes in the staging database.
- The property name in the destination page of the web interface. Each listing is sorted alphabetically on this property name. Where a property forms part of a compound key identifying records in the underlying compliance database, it is appropriately marked as a partial key ("p/key").
- The data type of the property.
- For string values, the maximum length allowed in the web interface (which is also normally the maximum string length available in compliance database). If this is less than the string length allowed in the staging database, the string is normally truncated to length (except as noted in the comments for an individual property).
- Any general notes.

## Inventory Device (Computer)

### IT Asset Management (Cloud)

Inventory devices are (in the main) computers that have been found in imported inventory; and in the current context, they are found in data imported from BMC Discovery. (Keep in mind that it is possible for a device to be reported in more than one inventory source, which becomes important when you are assessing automatic deletion of records no

longer appearing in the source inventory.)

IT Asset Management correlates data from several points within BMC Discovery to create inventory device records. For example, the main sources from lowest to highest priority are:

1. The `HostInfo` node, by default populated during the discovery phrase of data gathering by BMC Discovery. Current data is staged in the `HostInfo_ci` staging table; but if BMC Discovery does not inventory the target device regularly, its information may be cleaned up and lost from this node.
2. The `Host` node, which by default BMC Discovery populates with data from the `HostInfo` node.
3. Data (of type `CPU Information`) visible in the `Detail` node of BMC Discovery, and staged into the `CPUInformationDetail_ci` staging table.



**Tip:** This data depends on a custom CPU pattern available only by request (see [CPUInformationDetail\\_ci \(Staging Table\)](#) for details).

4. The optional `UnixHardwareData` pattern (delivered as part of the BMC Discovery adapter). Data from this source is also visible in the `Detail` node of BMC Discovery (with the type `FNMP_HardwareEvidence`), and is staged in the `HardwareEvidenceDetail_ci` table.

Generally, after import into IT Asset Management, inventory devices are stored in the `ComplianceComputer` table of the underlying compliance database.

Because inventory devices have so many properties, the following tables group their properties into smaller subsets. Within each subset, the tables are sorted alphabetically by the **Display property** column.

- Are new records created by imports from the staging database table? — **Yes.**
- Can records with matching compound keys be updated by these imports? — **Yes.**
- Are records unmatched by an import from the staging table deleted from the compliance database (provided that the staging database for BMC Discovery data is the last/only source for these records)? — **Yes.**

## Inventory device: Computer (general properties)

All these properties are available in listings such as the **All Inventory** page; and within the properties of an individual inventory device, these all appear on the **General** tab.

Source table.property	Display property	Type	Max	Notes
<code>Host_ci.Domain</code>	<b>Domain name (p/key)</b>	String	100	
<code>Host_ci.Serial</code>	<b>Firmware serial number</b>	String	100	Visible in the <b>General</b> tab of the inventory device properties. Identifies the firmware on the inventory device.
<code>NetworkInterface_ci.IP</code>	<b>IP address</b>	String	256	
<code>NetworkInterface_ci.MAC</code>	<b>MAC address</b>	String	256	

Source table.property	Display property	Type	Max	Notes
Host_ci.Vendor	<b>Manufacturer</b>	String	128	
Host_ci.Model	<b>Model</b>	String	128	
Host_ci.Hostname	<b>Name</b> (p/key)	String	256	
Host.Serial	<b>Serial number</b>	String	100	A duplicate of Host_ci.Serial, this time taken from the Host database view.

### Inventory device: Hardware properties

Except as noted, these properties are available in listings such as the **All Inventory** page; and within the properties of an individual inventory device, these mostly appear on the **Hardware** tab (where several values may be manually overridden if the collected inventory data is inaccurate).

Source table.property	Display property	Type	Max	Notes
Host_ci.ProcessorSpeed	<b>Clock speed (MHz)</b>	Integer		The maximum clock speed of the fastest processor in the computer in megahertz.
Host_ci.Cores (or CPUInformationDetail_ci.Cores for ADDM 11.0 or earlier); or, when the optional UnixHardwareData pattern has been used, HardwareEvidenceDetail_ci.Cores	<b>Cores</b>	Integer		The number of cores in the device (or, for a virtual machine, the number of cores assigned to the VM).
FileSystem_ci.Size (summed)	<b>Disk (GB)</b>	Integer		All FileSystem_ci records with an identical HostKey are selected from the staging database, and their Size values are summed and the result converted to GB for display.
<i>Not applicable</i>	<b>Display adapters</b>	Integer		The number of display adapters is not returned by BMC Discovery.
FileSystem_ci.COUNT(HostKey)	<b>Hard drives</b>	Integer		

Source table.property	Display property	Type	Max	Notes
<i>Not applicable</i>	<b>Inventory chassis type</b>	String	128	After import from BMC Discovery, this value is always Unknown (although some inventory tools can return the chassis type, BMC Discovery is not among them). The result is only available in the property sheet for an inventory device (and its linked asset, if any), and not in device listings. There is also an <b>Assigned chassis type</b> where you can correct inventory data if required.
NetworkInterface_ci. IP (counted)	<b>Network cards</b>	Integer		The number of network cards in the device is not returned by BMC Discovery. Instead, the count of distinct IP-enabled interfaces sharing a common HostKey is used.
Host_ci.LogicalProcessors (or CPUInformationDetail_ci. LogicalProcessors for ADDM 11.0 or earlier); or, when the optional UnixHardwareData pattern has been used, HardwareEvidenceDetail_ci.Cores	<b>Threads</b>	Integer		Saved in the ComplianceComputer table in the compliance database, and generally represented in the web interface as <b>Threads</b> for virtual machines.
Host_ci.OS	<b>Operating system</b>	String	128	
Host_ci.Processors (or CPUInformationDetail_ ci.Processors (for ADDM 11.0 or earlier); or, when the optional UnixHardwareData pattern has been used, HardwareEvidenceDetail_ci.Cores	<b>Processors</b>	Integer		The number of processors reported in the inventory device.
Host_ci.ProcessorType	<b>Processor type</b>	String	256	
Host_ci.RAM	<b>RAM (GB)</b>	String	128	The imported value (in bytes) is converted to GB for display.
Host_ci.ServicePack	<b>Service pack</b>	String	128	

Source table.property	Display property	Type	Max	Notes
n/a	<b>Sockets</b>	Integer		The number of sockets — a value that is not reported by BMC Discovery.

### Inventory device: Last inventory details

These properties are available in listings such as the **All Inventory** page; and within the properties of an individual inventory device, these appear on the **General** tab.

Source table.property	Display property	Type	Max	Notes
HostInfo_ci. InventoryDate	<b>Last inventory date</b>	Date		For virtual hosts, the most recent inventory date for any guest system may update this value.
BMC Atrium Discovery and Dependency Mapping	<b>Last inventory source</b>	String	128	The string literal "BMC Atrium Discovery and Dependency Mapping" is always inserted as the source value when using this adapter.

### Inventory device: User details

These properties are available in listings such as the **All Inventory** page; and within the properties of an individual inventory device, these appear on the **Ownership** tab.

Source table.property	Display property	Type	Max	Notes
n/a	<b>Assigned user</b>	String	512	This value is set manually by an operator in IT Asset Management.
n/a	<b>Calculated user</b>	String	512	Set automatically as the logged-on user seen most frequently in the last 10 inventory imports.
LastLoggedInUser Detail_ci.Username (when the optional pattern WindowsLastLoggedInUser has been used)	<b>Last logged on user</b>	String	512	This information is available only for Windows platforms, and only when WindowsLastLoggedInUser has been used. On data import, if the username cannot be matched in the ComplianceUser table of the compliance database, a new record is created, and appropriately linked to the ComplianceComputer record.

# Installer Evidence

IT Asset Management (Cloud)

Installer evidence is a record of software installation on an inventory device. 'Normalized' installer evidence (meaning installer evidence that includes standard string values as noted below) allows the automatic matching of the evidence to the application that it represents, using the Application Recognition Library (ARL), frequently updated by Flexera.

- Are new records created by imports from the staging database table? — **Yes.**
- Can records with matching compound keys be updated by these imports? — **No.**
- Are records unmatched by an import from the staging table deleted from the compliance database (provided that the staging database for BMC Discovery data is the last/only source for these records)? — **Yes** (records linking applications to inventory devices are deleted under these conditions).

## Installer evidence mapping

The properties for installer evidence are taken as a UNION of values from the `DiscoveredPackages_ci`, `InstallerEvidenceDetail_ci`, and `SoftwareInstance_ci` tables of the staging database. The union is used to create a de-duplicated, distinct set of values as listed below. If there are clashes in the values (for example, if one table lists the Publisher as Adobe and another lists to Vendor as Adobe Software), separate records are created of distinct evidence listing each of those different values. Since most of the properties are combined to form the primary key for the compliance database, this may give the *appearance* of duplicate records at first glance, so examine the values carefully for slight variations that trigger the creation of separate inventory evidence records.

Most of these properties are available in listings such as the **Installer evidence** tab of the **All Evidence** page; and within the properties of an individual installer evidence record, these appear on the **General** tab.

Source table.property	Display property	Type	Max	Notes
DiscoveredInstalledPackages_ci.HostKey, InstallerEvidenceDetail_ci.HostKey, and SoftwareInstance_ci.HostKey	Not displayed. (p/key)	varchar	64	Identifies the inventory device on which the installer evidence was found. This value is not displayed in listings of, nor properties for, installer evidence (because in those contexts, the installer evidence may be relevant to many different devices). However, the link from the installer evidence to the application, plus this link to an individual device where the installer evidence has been found, allows the list of installed applications to appear on the <b>Applications</b> tab of the inventory device properties.
DiscoveredPackages_ci.Package, InstallerEvidenceDetail_ci.Name, and SoftwareInstance_ci.Name	<b>Name</b> (p/key)	String	256	For recognition using the standard ARL to work reliably, this value must be the Display Name from Add/Remove Programs on Windows (or equivalent).
InstallerEvidenceDetail_ci.ProductCode	Not displayed.	String	110	The publisher's product code is not displayed in IT Asset Management.
DiscoveredPackages_ci.Vendor, InstallerEvidenceDetail_ci.Vendor, and SoftwareInstance_ci.Publisher	<b>Publisher</b> (p/key)	String	200	For recognition using the standard ARL to work reliably, this value must be the Publisher from Add/Remove Programs on Windows (or equivalent).



Source table.property	Display property	Type	Max	Notes
ADDM (string literal)	<b>Type</b>	String	32	The string literal "ADDM" is supplied as the installer evidence type for this adapter.
DiscoveredPackages_ci.Version, InstallerEvidenceDetail_ci.Version, and SoftwareInstance_ci.Version	<b>Version</b> (p/key)	String	72	For recognition using the standard ARL to work reliably, this value must be the Display Version from Add/Remove Programs on Windows (or equivalent).

## File Evidence

### IT Asset Management (Cloud)

File evidence consists of files found on the disk of an inventory device. 'Normalized' file evidence (meaning file evidence that includes standard string values as noted below) allows the automatic matching of the file evidence to the application of which it forms part, using the Application Recognition Library (ARL), frequently updated by Flexera.

- Are new records created by imports from the staging database table? — **Yes**.
- Can records with matching compound keys be updated by these imports? — **No**.
- Are records unmatched by an import from the staging table deleted from the compliance database (provided that the staging database for BMC Discovery data is the last/only source for these records)? — **Yes**.

### File evidence mapping

Most of these properties are available in listings such as the **File evidence** tab of the **All Evidence** page; and within the properties of an individual file evidence record, these appear on the **General** tab. (This listing is alphabetical by **Display property** values.)

Source table.property	Display property	Type	Max	Notes
Host_ci.HostKey	Not displayed. (p/key)	n/a		Identifies the inventory device on which the file evidence was found. This value is not displayed in listings of, nor properties for, file evidence (because in those contexts, the file evidence may be relevant to many different devices). However, once the file evidence is linked to an application, this link to an individual device where the file evidence has been found may affect the list of installed applications that appears on the <b>Applications</b> tab of the inventory device properties.
FileEvidenceDetail_ci.Company	<b>Company</b> (p/key)	String	100	For recognition using the standard ARL to work reliably, this value must be the Company WinPE header property from Windows executables.
FileEvidenceDetail_ci.Description	<b>Description</b> (p/key)	String	200	For recognition using the standard ARL to work reliably, this value must be the Description WinPE header property from Windows executables.
FileEvidenceDetail_ci.FileName	<b>Name</b> (in listings) or <b>File name</b> (in file evidence properties) (p/key)	String	256	For recognition using the standard ARL to work reliably, this value must be: <ul style="list-style-type: none"> <li>On Windows, the file name without path</li> <li>On UNIX-like platforms, the full absolute file path.</li> </ul>
FileEvidenceDetail_ci.FileSize	<b>Size</b>	Integer		
FileEvidenceDetail_ci.FileVersion	<b>Version</b> (p/key)	String	100	For recognition using the standard ARL to work reliably, this value must be the File Version WinPE header property from Windows executables.

The following extended properties of file evidence records are not populated in an BMC Discovery import:

- **File path**
- **Language**
- **Product name**
- **Product version.**

# WMI Evidence

## IT Asset Management (Cloud)

Windows Management Instrumentation (WMI) evidence is most often customized to help recognize operating systems, and (when the FlexNet Inventory Agent is in use) to help recognize aspects of special applications like Microsoft SQL Server. 'Normalized' WMI evidence (meaning WMI evidence that includes standard string values as noted below) allows the automatic matching of the WMI evidence to the application to which it relates, using the Application Recognition Library (ARL), frequently updated by Flexera.

- Are new records created by imports from the staging database table? — **Yes**.
- Can records with matching compound keys be updated by these imports? — **No**.
- Are records unmatched by an import from the staging table deleted from the compliance database (provided that the staging database for BMC Discovery data is the last/only source for these records)? — **Yes**.

## WMI evidence mapping

In the web interface for IT Asset Management, there is no separate listing of WMI evidence. It is only possible to see these properties using the search facility in the **WMI** section of the **Evidence** tab of application properties. (This listing is alphabetical by **Display property** values.)

Source table.property	Display property	Type	Max	Notes
Host_ci.HostKey	Not displayed. (p/key)	n/a		Identifies the inventory device on which the WMI evidence was found. This value is not displayed in the application properties <b>Evidence</b> tab. However, once the WMI evidence is linked to an application, this link to an individual device where the WMI evidence has been found may affect the list of installed applications that appears on the <b>Applications</b> tab of the inventory device properties.
Caption (string literal)	<b>Property name</b> (p/key)	String	50	The string literal "Caption" is inserted for all WMI evidence records from BMC Discovery.
Host_ci.OS (left-most 256 characters)	<b>Property value</b> (p/key)	String	256	For OS recognition using the standard ARL to work reliably, this value must be the equivalent of the <code>Win32_OperatingSystem.Name</code> WMI property.

Source table.property	Display property	Type	Max	Notes
Host_ci.OSType converted as per notes.	<b>WMI class</b> (p/key)	String	50	<p>The BMC Discovery data converts to standard class types used by IT Asset Management as follows:</p> <ul style="list-style-type: none"> <li>• Windows becomes Win32_OperatingSystem</li> <li>• Either of VMware ESX or VMware ESXi becomes VMWARE_OperatingSystem</li> <li>• Any other value is replaced by MGS_OperatingSystem.</li> </ul>



# Citrix Cloud Adapter

## IT Asset Management (Cloud)

The **Citrix Cloud** adapter, provided by Flexera, allows you to collect data from Citrix Cloud persistent and non-persistent VDIs and import it into IT Asset Management.

Imported VDI data and application evidence collected by the FlexNet Inventory Agent will give you full IT visibility of inventory running on the Citrix Cloud platform, and provide licensing capabilities for all applications used by end-users on persistent and non-persistent VDIs for Citrix Cloud.



**Note:** Virtual desktop infrastructure (VDI) is a technology that refers to the use of virtual machines to provide and manage virtual desktops. VDI hosts desktop environments on a centralized server and deploys them to end-users on request.

## The difference between non-persistent and persistent VDIs

Once a user logs off a virtual machine (VM), that VM and associated data is destroyed. This is referred to as a non-persistent VDI. Any changes made on the VM will not persist when the VM is destroyed. Because of this, non-persistent VMs are very difficult to license.

Unlike non-persistent VMs, persistent VMs are not destroyed once a user logs off. Persistent VMs are similar to that of a standard VM, and when set up, administrators can give access to groups/users who can continue to use it until the administrator deletes it.

## What will you see in the UI

In IT Asset Management, you can view a VDI template for each image that has been configured Citrix Cloud. This template allows you to see the full list of applications on all VDIs deployed from that template.



**Tip:** The "VDI template" terminology is used in the IT Asset Management UI. For customers not familiar with this terminology, the "VDI template" can also be referred to as the "golden image virtual machine".

For each user who has access to any of the VDIs that were deployed from a VDI template, the application evidence is mapped over to any inventory device for which that user is the primary user.

For any end-user who can access a VDI template who is not a primary user of any inventory device, a remote device will be created for them, and the application evidence will be mapped.

## Supported versions

- Any version of Citrix Remote PowerShell SDK
- The Citrix Cloud adapter is supported by version 2022 R2 of IT Asset Management onwards.

## What's not supported

- Elastic app layering in Citrix Cloud is currently not supported. This technology allows you to dynamically assign applications to a specific VDI instance outside of the VDI template.
- Delivery groups with the **allow all authenticated users access policy** are not supported.
- The **entitlement policy** is not evaluated, as this policy may limit users from launching desktops.

**Access to delivery group** is the only permission that is evaluated. This permission gives users access to any desktops in a specific delivery group.

## 1

# Prerequisites

## IT Asset Management (Cloud)

The Citrix Cloud adapter is available in the FlexNet Beacon UI from IT Asset Management 2022 R2 onwards.

The Citrix Cloud adapter requires the following:

- Any version of Citrix Remote PowerShell SDK installed onto the beacon.
- Create at least one delivery group that will provide VDIs to users.
- Install the FlexNet Inventory Agent. It is recommended that you install the FlexNet Inventory Agent onto the VDI template to ensure inventory from at least one VDI device can be collected per delivery group, unless another inventory source is providing software inventory from the VDI devices.



**Tip:** The "VDI template" terminology is used in the IT Asset Management UI. For customers not familiar with this terminology, the "VDI template" can also be referred to as the "golden image virtual machine".

- An inventory beacon (or multiple if required) that collects Active Directory data from the domain where the users are located who have been given access to a delivery group.
- Configure a connection in the FlexNet Beacon UI to Citrix Cloud. See [Creating the Citrix Cloud Connection](#).
- PowerShell 5.0 or higher on the beacon where the Citrix Cloud adapter is set up.
- **Read-only** Citrix Cloud account rights are the minimum permissions required to log in and generate the API Key for the integration to work.



**Important:** API clients inherit the permissions of the administrator that creates them. For example, if an administrator has limited access, the API client's access is similarly restricted. In this instance, it is recommended that you create a Citrix Admin account with read-only permissions and subsequently log in as a read-only admin to create the API client.

## 2

# Architecture and Operation

## IT Asset Management (Cloud)

The Citrix Cloud adapter has been created to collect supplementary VDI data. This data will show:

- Existing VDI devices and templates
- Existing delivery groups in Citrix where these VDI devices and templates are installed
- What users have access to these delivery groups.

The FlexNet Inventory Agent which is installed on the VDI template, collects application evidence from each of the VDI devices purported by the Citrix Cloud adapter. This application evidence shows all of the software that end-users have access to.

To import the collected supplementary VDI data into IT Asset Management, the Citrix Cloud adapter uses the Citrix Remote PowerShell SDK in order to connect to Citrix Cloud and query the relevant API(s).

Citrix Cloud documentation pertaining to the API used for gathering application evidence on the connection server is available [here](#).

There are 4 main components in the Citrix Cloud adapter:

- **Delivery group (Citrix):** A collection of existing virtual machines. The FlexNet Inventory Agent collects the application evidence from these machines which is then mapped to users who have access to that delivery group. Note: Access to a delivery group is defined in Active Directory.
- **Inventory Beacon:** Connects to a single connection to Citrix Cloud. Inventory is then uploaded to the Batch and Inventory Servers. The inventory beacon also imports data from Active Directory, including groups (and their members), users, and computers, and the security identifiers for each item within Active Directory. (These security identifiers, or SIDs, are the same identifiers that the Citrix Cloud adapter reports for usage of the applications delivered by Citrix Cloud).
- **Inventory Server:** Is where the application evidence (.NDI file from each VDI device) is received, processed and imported to the IM inventory database. .NDI files are produced by running the FlexNet Inventory Agent on the VDI.
- **Batch Server:** Is where data from the IM Inventory Database is processed and imported to the IT Asset Management Compliance database which in turn drives the VDI template UI. Note: The Citrix Cloud adapter has been configured as a new compliance connection. VDI data is sent to the Batch server as intermediate data files which are then processed (matched/merged) with data from other compliance connections to produce a single view of the data and imported



to the IT Asset Management database.

## What data is retrieved

The data listed below is retrieved by means of running functions in the PowerShell reader that is used to connect to the Citrix Cloud API on the configured connection server.

Functions	Retrieved data
Site name	The Name property of the Citrix site as returned by the Get-BrokerSite cmdlet.
Delivery groups	<p>The delivery groups are interrogated with the Get-BrokerMachine cmdlet. Relevant properties are:</p> <ul style="list-style-type: none"> <li>• DesktopGroupName</li> <li>• CatalogName</li> <li>• DesktopGroupUUID</li> </ul>
Machines	<p>The VDIs are queried with the Get-BrokerMachine cmdlet. Relevant properties are:</p> <ul style="list-style-type: none"> <li>• DNSName</li> <li>• DeliveryType</li> <li>• PersistUserChanges</li> </ul>
User access	<p>User access for Citrix Virtual Desktop is collected with the Get-BrokerAccessPolicyRule and Get-BrokerEntitlementPolicyRule(desktop entitlements).</p> <p>User access for Citrix Virtual Application is collected with Get-BrokerApplication cmdlet.</p> <p>Active Directory SID for each user or group in the IncludedUsers property is collected.</p>
Applications	<p>Application data and corresponding delivery group is queried with the Get-BrokerApplication cmdlet. Relevant properties are:</p> <ul style="list-style-type: none"> <li>• AppName</li> <li>• ApplicationUID</li> <li>• AssociatedDesktopGroupUids</li> <li>• AssociatedDesktopGroupUUIDs</li> <li>• AssociatedUserSIDs</li> <li>• BrowserName</li> <li>• Uid.</li> </ul>
Test connection	<p>A test connection button is available in the FlexNet Beacon UI. Selecting test connection will show a successful test if the configured user is able to successfully log into the API, going through any configured proxy.</p> <p>If the connection fails, the relevant error is fed back to the user.</p>

## 3

# Creating the Citrix Cloud Connector Connection

IT Asset Management (Cloud)

Make sure to allow access to the following URLs and endpoints from their environments by adding these to the trusted sites on one of the browsers you are using with Flexera One

Use the following procedure to create a connection to Citrix Cloud on the FlexNet Beacon. A connection is required to be configured for a single connection to your Citrix site. The inventory beacon is responsible for uploading the data to the central operations databases of IT Asset Management.



## To create the Citrix Cloud connection in the FlexNet Beacon UI:

1. Log into your selected inventory beacon.



**Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.

2. Add the permitted URLs that are listed on the [Citrix DaaS SDKs and APIs](#) page to the trusted sites on the inventory beacon. Access to these URLs are required to use the Citrix DaaS Remote PowerShell SDK.
3. Start the FlexNet Beacon interface.
4. In the navigation pane on the left, select the **Inventory systems** page. To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.

The **Create PowerShell Source Connection** dialog appears.



**Tip:** The **New...** button defaults to creating a connection for Microsoft SQL Server. If you use the down arrow on the split button, you can choose between *SQL Server*, *Spreadsheet*, *PowerShell*, and *Other* connections. However, while you are creating a connection to a Microsoft SQL Server database (regardless of the **Source Type** of the connection), use only the *SQL Server* option.

5. Complete the values in the dialog, as follows:

Control	Comments
<b>Connection Name</b>	A descriptive name for this connection, such as Citrix VDI Data. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name.
<b>Source Type</b>	Select <b>Citrix Cloud</b> from the list.
<b>Use Proxy</b>	Optionally, if you use a proxy server to enable Internet access, complete (or modify) the values in the <b>Proxy Settings</b> section of the dialog box in order to configure the proxy server connection.
<b>Proxy Server</b>	Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format <code>https://ProxyServerURL:PortNumber</code> , <code>http://ProxyServerURL:PortNumber</code> , or <code>IPAddress:PortNumber</code> . This field is enabled when the <b>Use Proxy</b> check box is selected.
<b>Username and Password</b>	If your enterprise is using an authenticated proxy, specify the <b>username</b> and <b>password</b> of an account that has credentials to access the proxy server that is specified in the <b>Proxy Server</b> field. These fields are enabled when the <b>Use Proxy</b> check box is selected.
<b>Customer ID</b>	Enter the customer ID of the Citrix Cloud account for which you want to collect the VDI data.
<b>API Key</b>	Enter your Citrix Cloud API Key username. The key can provide programmatic access. You must log into your Citrix Cloud account and create an API client in order to generate the API Key. See <a href="#">here</a> for more information on how to get the API Key.
<b>Secret Key</b>	Enter your Citrix Cloud Secret Key. Similar to the API Key, you must log into your Citrix Cloud account and create an API client in order to generate the Secret Key. See <a href="#">here</a> for more information on how to get the Secret Key.

Control	Comments
<b>Connection is in test mode (do not import results)</b>	<p>Controls the uploading and importing of data from this connection:</p> <ul style="list-style-type: none"> <li>When this check box is clear, the connection is in production mode, and data collected through this adapter is uploaded to the central server and (in due course) imported into the database there.</li> <li>When the check box is set: <ul style="list-style-type: none"> <li>The adapter for this connection is exercised, with data written to the intermediate file in the staging folder on the inventory beacon (%CommonAppData%\Flexera Software\Beacon\IntermediateData)</li> <li>The immediate upload that normally follows data collection is suppressed, so that you can inspect the contents of the file</li> <li>The catch-up process that retries stalled uploads, normally scheduled overnight, runs as usual and uploads the file to the central server</li> <li>At the central server, the file contents are discarded (and not imported into the central database).</li> </ul> </li> </ul>
<b>Overlapping Inventory Filter</b>	This control does not apply to the Citrix Cloud adapter, and you may leave it at the default setting.

## 6. Click **Test Connection**

This will make sure that you can successfully authenticate against the API and the specified **Customer ID**, **API Key** and **Secret Key** are correct. Note: a request to log into the API is part of the test connection.

- If the connection is successful, click **OK** to close the message. Click **Save** to complete the addition. The connection is added to (or updated in) the list.
- If the connection is unsuccessful, the appropriate error message will display. Click **OK** to close the message. Edit the connection details and retest the connection.  
You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.

7. In the FlexNet Beacon PowerShell Source Connection dialog, click **Save** to save the connection.

8. Select your new connection from the displayed list, and click **Schedule....**

9. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.

10. At the bottom of the FlexNet Beacon interface, click **Save**, and if you are done, also click **Exit**.

After a successful data import, existing VDI devices and templates, existing delivery groups and application names where these VDI devices and templates are installed and what user groups have access to these delivery groups and application names are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** page of FlexNet Beacon, see the **Inventory Systems Page** in the online help. For scheduling data imports through this connection, see **Scheduling a**

---

*Connection, also in the online help.*

# IV

## Citrix XenApp Server Adapter

### IT Asset Management (Cloud)

The Citrix XenApp server adapter, provided by Flexera, allows you to collect software inventory from Citrix Virtual Apps (formerly Citrix XenApp) and import it into IT Asset Management. Depending on the type of virtualized applications being served, the evidence appears in either the installer evidence list (for App-V or streaming profile applications delivered through Citrix Virtual Apps), or in the file evidence list (for file-based applications managed through Citrix Virtual Apps).

In either case, the evidence must be linked to an application record. This can happen in either of two ways:

- Where the evidence is matched by an existing inventory rule for an application, it is automatically linked to that application record.
- Where required details (below) are incomplete, or there is no exact match in any existing evidence rules/records (either supplied by the Application Recognition Library or created locally in your enterprise), the evidence is left in the **Discovered Evidence** page (and **All Evidence** page) with its **Assigned** property set to No. The fields that require matching are:
  - For installer evidence, the application name, version and publisher
  - For file evidence, the file name, version, company, and description.

Once the evidence is linked to an application, the application must be linked to a license. The license should then be linked to purchase records to determine your entitlements. Of course, these are manual tasks outside the scope of the adapter's operations.

The term *XenApp server* is used in this documentation as a generic term to cover the differently-named control servers for different versions of Citrix Virtual Apps:

- In version 6.x, the XenApp server was officially named the Zone and Data Collector. One such controlling server was required per *farm*.
- In version 7.5 and later, the XenApp server is called the Delivery Controller. One such controlling server is required per *delivery site*.

### Supported versions

Citrix Virtual Apps server (previously XenApp server) 6.0, 6.5, 7.5-7.9, 7.11-7.17 (XenApp), 7.1808-7.2407 (Citrix Virtual Apps)

If it happens that you have multiple of these versions of Citrix Virtual Apps in operation (for example in different domains), you can use the same structure described in this section to link them all to IT Asset Management.

## 1

# Architecture, Operations and Prerequisites

IT Asset Management (Cloud)

This chapter provides a useful framework for your understanding of the more detailed content to follow.

## Architecture and Operation

IT Asset Management (Cloud)

In order to track licenses for applications delivered remotely to users from a Citrix Virtual Desktops App environment, IT Asset Management needs information about which users and devices have access to which applications. There are several sources of such data available from XenApp servers, depending on the version of Citrix Virtual Desktops (formerly XenApp):

- For XenApp 6.0 and 6.5:
  - Access control lists (ACLs)
  - Streaming profiles
  - Citrix EdgeSight servers.
- For XenApp 7.5 and later:
  - Access control lists (ACLs)
  - User filters collected from Applications and Delivery Groups
  - App-V 5 packages (and the applications they contain).



**Tip:** No usage tracking is possible for XenApp 7.5, as in this release Citrix did not include usage tracking capabilities, in XenApp.

- For XenApp 7.6 and later:
  - The XenDesktop database supplied as part of XenApp that tracks application usage.



- For XenApp 7.9 and later:
  - The collection of Application Group data
  - User filters collected from Application Groups.



**Note:** A one time staging database update is required to use XenApp server agent to collect Application Group data and to collect user filters from Applications, Application Groups, and Delivery Groups.

- For Citrix Virtual Apps 7.1808 and later:
  - The collection of Application Group data
  - User filters collected from Application Groups.



**Note:** A one time staging database update is required to use XenApp server agent to collect Application Group data and to collect user filters from Applications, Application Groups, and Delivery Groups.

These sources are discussed in turn in the following sections.

## Access control lists (ACLs)

These are lists which specify the permissions associated with an object, such as an application's executable file, on a server. The FlexNet Manager Agent for XenApp Server (XenApp server agent) is a tool that extracts information about users and which applications they can access remotely, and transfers that information to an inventory beacon for use in licensing calculations in IT Asset Management. To do this, the XenApp server agent must be installed:

- For Citrix XenApp 7.5 (and later) and Citrix Virtual Apps 7.1808 (and later), on one Delivery Controller for each Delivery Site. If you have multiple Delivery Sites, you may choose either of the following:
  - Install the XenApp server agent on one Delivery Controller in each Delivery Site
  - Use only a single XenApp server agent, and provide that XenApp server agent with the required network access and credentials to access all required Citrix Virtual Apps Delivery Controllers.
- For XenApp 6.0 or 6.5, on one controlling XenApp server in each Citrix farm.



**Tip:** While the XenApp server agent is installed only on one server per farm, for XenApp 6.x you also need software inventory from every XenApp server, in order to identify the editions of applications available to users and computers. This separate inventory of the XenApp servers can be obtained either by installing the FlexNet Inventory Agent on the XenApp server, or using Zero-footprint inventory collected by an inventory beacon. Do not get the two separate agents (FlexNet Inventory Agent, and XenApp server agent) confused. The main focus of this adapter documentation is the XenApp server agent.

The XenApp server agent is supplied as an integral part of the Citrix XenApp server adapter.



**Tip:** In earlier releases, the XenApp server agent extracted Active Directory names and details of users and devices from the XenApp server. Now, the XenApp server agent collects only Active Directory SIDs (a large performance improvement). As a result, best practice is that your inventory beacon completes its import of Active Directory data before importing Citrix Virtual Apps data, so that all Active Directory SIDs can be resolved against the user names, devices, and groups collected directly from Active Directory.

## Streaming profiles for XenApp version 6.0 and 6.5

The XenApp server agent is also able to read the contents of the `.profile` and the key executable files associated with streamed applications published to your XenApp servers.



**Tip:** As the streaming profile is not stored in the XenApp server's database, the XenApp server agent must have at least read access to the streaming profile location to be able to read and extract this information.

As these applications are not physically installed on your XenApp server, combining the XenApp server agent's data from `.profile` files with EdgeSite server information may be the only way for IT Asset Management to recognize usage of such applications.



**Note:** IT Asset Management is only able to recognize usage of files streamed to a XenApp server, not those streamed directly to client devices.

## Citrix EdgeSight servers for XenApp 6.0 and 6.5

Citrix EdgeSight for XenApp monitors and profiles the usage of remote and streamed applications by users, telling you both who is using that application, and on what device. The data from EdgeSight is very valuable for IT Asset Management: you may use it for license optimization (for example, tightening access through ACL permissions to exclude users who evidently do not need to use the applications); or it may be critical for any user-based or usage-based licensing of applications delivered through XenApp 6.0 or 6.5.

EdgeSight agents may be installed on each XenApp server, and report back to a central EdgeSight server, which can keep track of application usage on multiple XenApp machines, belonging to one or more farms. The FlexNet Beacon can connect to each EdgeSight server and collect this usage information for use in compliance calculations.

Unlike data from the XenApp server agent, EdgeSight data does contain details of which devices access a particular application. Thus, EdgeSight data is usually more valuable to an enterprise for calculating license compliance than the data about application availability returned by the XenApp server agent alone. If, however, your enterprise deploys streamed applications, EdgeSight usage data may need to be supplemented by XenApp server agent information to accurately recognize these applications.

The following information is returned from the EdgeSight server:

- A list of applications (product name, version, publisher, and description) and the users who use them
- The devices on which users request and run applications
- The XenApp servers from which users request applications
- The farms to which the XenApp servers belong.



**Tip:** The EdgeSight data does not include application editions. Because different XenApp servers may have different editions installed (and available to users), it is important to take software (and hardware) inventory of the XenApp servers themselves, using the FlexNet Inventory Agent (either installed locally on each server or operating remotely from an inventory beacon). This inventory reveals the software editions available on each of the servers, which can be combined with the information listed above to give complete usage data required for license calculations. For example, if server XenApp01 offers Visio Standard, while server XenApp02 offers Visio Professional, the inventory from XenApp01 and XenApp02, combined with the data listed above, allows the license consumption calculations to link users to the appropriate license.

To use EdgeSight data, you must create a database connection to the EdgeSight SQL server database.

## App-V 5 packages for Citrix XenApp 7.5 (and later) and Citrix Virtual Apps 7.1808 (and later)

The XenApp server agent is able to inspect the contents of App-V 5 packages and recover the name, version, and publisher of the application contained in each package. The XenApp server agent also returns the user's ability to access these App-V packages (as recorded in the ACLs described earlier). However, in the ability to track which users actually use the applications, there are differences across versions:

- Version 7.5 has no technology like the EdgeSight server available in version 6.x, and so cannot report application usage
- From version 7.6, XenApp (Citrix Virtual Apps) again allows tracking application usage through connection to the Citrix Virtual Desktops (formerly XenDesktop) database incorporated in XenApp 7.6 and later.

## VDI images for Citrix XenApp 7.5 (and later) and Citrix Virtual Apps 7.1808 (and later)

The same capabilities apply to VDI images. The XenApp server agent interrogates any VDI device managed by the XenApp server to read the applications listed in all VDI source images available (including spinning up any images that are currently dormant to inspect their applications). As with App-V packages:

- For XenApp version 7.5, there is no ability to track who uses any VDI image, or when
- For Citrix XenApp 7.6 (and later) and Citrix Virtual Apps 7.1808 (and later), the included Citrix Virtual Desktops (formerly XenDesktop) database allows collection of application usage information.

## Changed architecture across versions

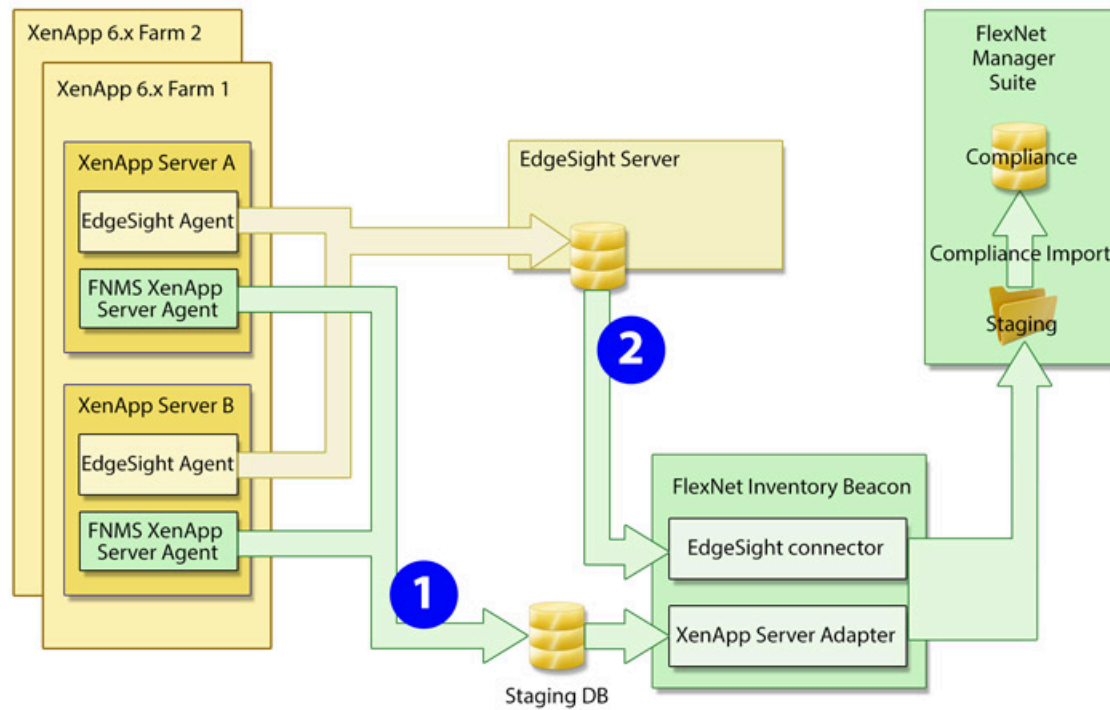
Because XenApp release 7.5 follows an extensive rewrite of the XenApp line by Citrix, the architectures of the two systems (and therefore the ways that the adapter integrates with the architecture) are quite different from version 6.x to 7.x.



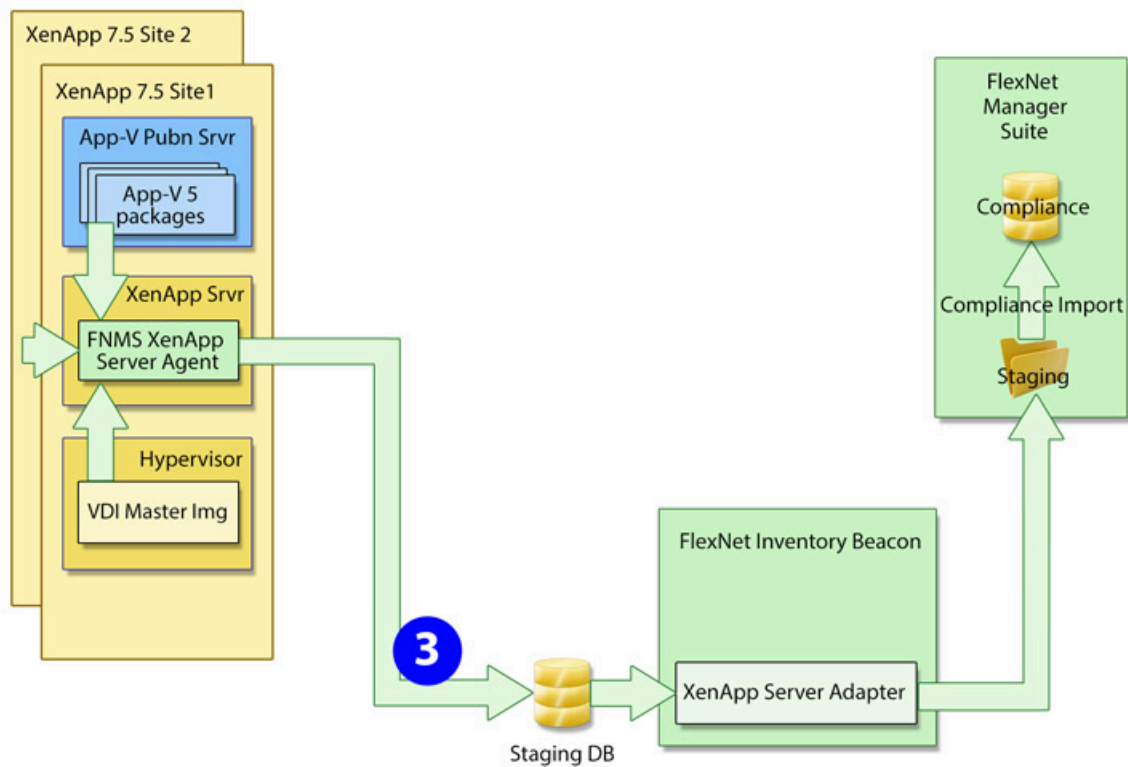
**Tip:** The following three diagrams do not include the import of Active Directory data by the inventory beacon, as this is not part of the adapter itself. However, the prior import of Active Directory data (typically, by the same inventory beacon connecting to the staging database) is a prerequisite for operation of the adapter.

Also keep clearly in mind the distinction between the XenApp server agent, an executable installed on your XenApp server(s), and the Citrix XenApp server adapter(s), an XML file (with embedded SQL) installed on your inventory beacon. The XenApp server agent is responsible for the first part of the process, collecting data and normally writing it into a staging database; and the Citrix XenApp server adapter(s) takes the second part, loading data from the staging database to the central application server and its compliance database.

This diagram represents the architecture and data flows for the Citrix XenApp server adapter(s) connected to XenApp 6.0 or 6.5 (the key numbers are referenced in the table below):

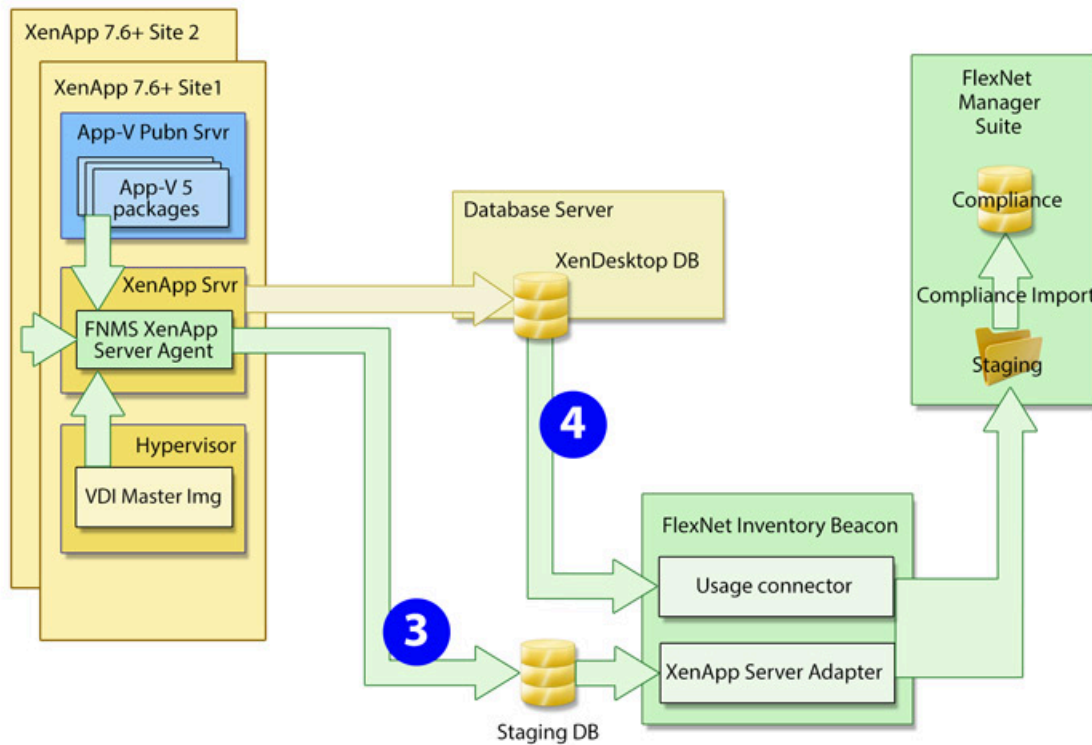


The next diagram shows the architecture and data flows when connected to XenApp 7.5. The diagram is laid out similarly to highlight the changes in architecture, and in particular the absence of usage information:



Finally, the third diagram shows the architecture and data flows when connected to Citrix XenApp 7.6 (or later) and

Citrix Virtual Apps 7.1808 (or later). The main point to note is the return of usage information:



The following table shows the data collected by the adapter through the different channels numbered in the three diagrams.

**Table 1:** Lists imported by Citrix XenApp server adapter

List	Case 1	Case 2	Case 3	Case 4
User SIDs, Active Directory Group SIDs	Y		Y	
File evidence (.exe) details – file name, version, company, description	Y	Y	Y	
Installer evidence (from App-V/streaming profiles) details – name, version, publisher	Y	Y	Y	Y
Application access rights per user SID	Y		Y	
Application usage per user		Y		Y*
Client computer SIDs (with user SIDs)		Y		Y
XenApp servers with applications present for delivery	Y			
App-V packages (and the applications therein) managed by Citrix Virtual Apps (formerly XenApp)			Y	
XenApp servers that served applications		Y		
Applications available in App-V packages (creates App-V 'evidence' records too)			Y	Y

List	Case 1	Case 2	Case 3	Case 4
Applications available as streaming profiles	Y			

\* Application usage by users and devices for Citrix XenApp 7.6 (and later) and Citrix Virtual Apps 7.1808 (and later) releases is limited to applications delivered in App-V packages. (Usage based on imported file evidence is not collected.)

## Operation with Citrix XenApp 7.5 (and later) or Citrix Virtual Apps 7.1808 (or later)

The XenApp server agent is installed on the XenApp server (at your discretion, on only one server that can access all other controllers for Citrix Virtual Apps (formerly XenApp) in your enterprise, or as many as one per Citrix Virtual Apps site).

Triggered by a Windows scheduled task, the agent runs according to your settings. It may collect inventory details only from the XenApp server on which it is installed (default), or it may collect from several controlling XenApp servers in sequence (identified with the `-s` command line option, detailed in [XenApp Server Agent Command Line Options](#)).

### a. VDI images

Based on information found on the XenApp server, the XenApp server agent may connect to any relevant XenApp servers hosting VDI images that contain applications. XenApp allows an administrator to nominate applications within VDI images for delivery

1. As individual applications only
2. Within a VDI (delivered as a whole environment) only
3. Either as individual applications or within a VDI.

The XenApp server agent collects information on all applications within the VDI source images that are identified for individual delivery (options 1 or 3 above). The VDI evidence is returned to IT Asset Management as file evidence. (For XenApp version 7.6 and later, usage tracking is not available for the file evidence.)



**Tip:** To track inventory delivered within an entire VDI environment (option 2 above), use XenDesktop discovery and inventory through the rules-based process in the web interface of IT Asset Management.

### b. App-V packages

Again based on information gathered from the XenApp server, the XenApp server agent connects to any Microsoft App-V publication server to inspect any App-V packages registered for delivery through XenApp. Because XenApp requires App-V version 5 (or later) for integration, the XenApp server agent can interrogate the packages to identify the applications inside. The App-V evidence is returned to IT Asset Management as installer evidence.

For XenApp version 7.6 and later (but not for version 7.5), a connection is also made to the XenDesktop database from the appropriate inventory beacon. This collects details of App-V application packages that users and devices have accessed. This additional information restores the ability (missing for version 7.5) to determine which users and devices have actually used each application, as distinct from merely having access to them. This may allow for more accurate license consumption calculations in later compliance calculations for applications delivered in this way.

### c. Processing

So both kinds of applications are returned to IT Asset Management as evidence:

- From VDI images, file evidence is produced that normally includes file name, version, company, and description

- For App-V packages, installer evidence is returned that normally includes application name, version, and publisher.

When the data is finally imported into IT Asset Management:

- The incoming evidence is tested against existing evidence records (including "rules" generalized with wild cards) already linked to applications (either from the Application Recognition Library or from records produced in your enterprise).
- If the incoming evidence matches any existing rule or record, it is recorded against the linked application, and its presence is recorded in the properties of the appropriate user or device as an "installation" record. For XenApp 7.6 (or later), a usage record is automatically created for each user and each device shown to have accessed the application.
- If the incoming evidence is not matched, it is displayed in evidence listings (for example, go to the **Discovered Evidence** page (**Applications & Evidence > Evidence > Discovered Evidence**), select the **Installer evidence** tab for App-V applications and the **File evidence** tab for VDI applications). You can select the evidence in the appropriate tab, and click **Assign** to choose an application record (or create a new one) to link to the evidence. (You only need do this the first time that new evidence is reported. Once linked to the application, your evidence serves as a 'rule' for matching future imports of the same evidence.)
- Once the incoming evidence is linked to an application (either automatically or manually), license reconciliation attempts to calculate consumption through XenApp on any license linked to the application. For this to take effect, you must correctly configure at least one license attached to the application:
  1. Navigate to the **Use rights & rules** tab of the license properties.
  2. Ensure that the **License consumption rules** heading is expanded (if not, click the heading).
  3. Select **Access granted to users, or usage, consumes license entitlements** to expose additional controls.
  4. Depending on the terms of your license, choose one of **Consume one entitlement for each user** or **Consume one entitlement per device owned by each user**.
  5. For XenApp 7.5, set **Consume entitlements based on** to **Access** (because only access records are available through XenApp 7.5). For other versions tracking App-V applications, check the terms of your license to see whether usage-based licensing is acceptable for this application, and make selections accordingly.

## Collection of Application Group data and user filters from Applications, Application Groups, and Delivery Groups

The collection of Application Group data from XenApp server and XenApp Desktop is supported in IT Asset Management for XenApp and XenDesktop versions 7.9 and later. IT Asset Management also supports the collection of user filters from Applications, Application Groups, and Delivery Groups to more accurately calculate application access. (XenApp and XenDesktop provide the ability to add user filters to Applications, Application Groups, and Delivery Groups in order to restrict access.)



**Note:** To take advantage of support for Application Groups and the collection of user filters, you must use the new XenApp server agent, **XenAppAgent**. In addition, a one time staging database update is required to use **XenAppAgent** to collect Application Group data and user filters from Application Groups. The **XenAppAgent** installer can be found in the within the **XenAppAgent** folder of the **Citrix XenApp Server Agent** subdirectory that is provided in the **Adapter Tools for FlexNet Manager Suite** archive. The **Adapter Tools for FlexNet Manager Suite** archive is available in the Customer Community link <https://community.flexera.com/s/article/adapter-tools-for-flexnet->

---

*manager-suite*.

---



**Tip:** Application Groups are groups that let you manage collections of applications. Being able to categorize applications into collections enables some settings to be administered for all applications in a group. The concept of Application Groups was added in the 7.9 release of XenApp and XenDesktop.

## Prerequisites

### IT Asset Management (Cloud)

The Citrix XenApp server adapter requires the following:

- The executable and supporting files for the XenApp server agent. These are available as described in [Creating the Staging Database](#).
- The XenApp server (on which the XenApp server agent is installed) requires:
  - .NET version 4.5 or greater
  - PowerShell 2.0 or greater.
- A staging database that can run in a convenient Microsoft SQL Server instance. For example, this may be a database running on the inventory beacon, or on the XenApp server hosting the XenApp server agent. For more about the requirements for this database,, see [Creating the Staging Database](#).database
- An inventory beacon (or multiple if required) that collects Active Directory data for the domain(s) where your XenApp server(s) are located.
- An inventory beacon (possibly the same as in the previous point) that can connect to your staging database and upload the inventory to the central IT Asset Management database. This process is performed by the XenApp server *adapter* on the inventory beacon.
- If you are using XenApp 6.x with the recommended EdgeSight server, an inventory beacon (almost invariably the same one as in the previous point) that can connect to the EdgeSight database.



**Tip:** For XenApp 6.x with EdgeSight, also be sure to take inventory from your XenApp servers so that edition information is available to your license consumption calculations. This inventory may also be collected by an appropriate inventory beacon.

- If you are using Citrix XenApp 7.6 (or later) or Citrix Virtual Apps 7.1808 (or later), an inventory beacon (usually the same one) that can connect to your XenDesktop database, and uploaded the imported data to your central operations databases.

### Supported versions

6.0, 6.5, 7.5–7.9, 7.11–7.17 (XenApp), 7.1808–7.2407 (Citrix Virtual Apps)



# 2

## Setting Up the XenApp Server Adapter

IT Asset Management (Cloud)

The Citrix XenApp server adapter is available for different versions of Citrix Virtual Apps (previously XenApp):

- For version 6.0 and 6.5, it augments information available from EdgeSight, particularly about streamed applications
- For version 7.5, it is the primary means of gathering inventory information from XenApp
- For version 7.6 and later, it again augments information about application usage collected from the XenDesktop database included with XenApp.

The adapter is easily downloaded from the Flexera Product and License Center. Installation consists of five main activities, all described in the following topics:

- Setting up the staging database for the inventory that is collected (see [Creating the Staging Database](#))
- Copying the appropriate folder for the adapter to your chosen XenApp server(s) (see [Installing the XenApp Server Agent](#))
- Ensuring an appropriate account is available to run the adapter (also in [Installing the XenApp Server Agent](#))
- Setting up a local scheduled task to run the agent as you require (see [Create a Scheduled Task](#))
- Setting up a connection from an appropriate inventory beacon to the staging database (see [Create Connections for Data Upload](#)).

## Creating the Staging Database

IT Asset Management (Cloud)

The staging database allows the XenApp server agent to drop collected data in a conveniently close location. From here, an inventory beacon collects the data for transfer to the central operations databases for IT Asset Management.

The staging database can be installed in any convenient Microsoft SQL Server 2012 (or later) database:

- If your inventory beacon is located on a SQL server, the staging database can be on the inventory beacon

- Where there is an inventory beacon with network access to your XenApp server (and Citrix Virtual Apps is running its database in SQL Server), the staging database can be installed into the same database as used by Citrix Virtual Apps
- If none of these suit, use any other SQL Server instance that allows network access both from the XenApp server agent on the XenApp server and from the inventory beacon.

It is a small footprint database (half a dozen tables and one stored procedure) that is size-limited by the scale of your Citrix Virtual Apps implementation, with data being replaced at each upload.



**To create the staging database (using supplied script):**

1. Use your browser to access the following Flexera Customer Community link: <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>, and then download the **Adapter Tools for FlexNet Manager Suite** archive and save it to a convenient location (such as C:\temp) on a suitable server.
2. In your unzipped archive, navigate into the \Citrix XenApp Server Agent subdirectory.
3. Further navigate into the appropriate sub-folder for your version of Citrix Virtual Apps (noting that the **XenAppAgent** folder applies to XenApp and XenDesktop versions 7.5 and later, and to Citrix Virtual Apps and Virtual Desktops versions 7.1808 and later):
  - XenAppAgent6
  - XenAppAgent65
  - XenAppAgent



**Note:** *If it happens that you have multiple different versions of Citrix Virtual Apps (for example, in different domains), you may use a single database for all versions of Citrix Virtual Apps. The following script is identical in all these folders.*

4. From your chosen folder, collect a copy of the database creation/update script `SetupXenAppAgentStagingDatabase.sql`.
5. On your selected SQL Server, drop a copy of this file, and execute it in SQL Server Administration Studio against your chosen database instance.

An appropriate SQL Server database instance:

- Is accessible from the XenApp server(s) running the XenApp server agent
- Is accessible from the inventory beacon responsible for uploading collected inventory to IT Asset Management
- Grants read/write access to the account running the scheduled task for the Citrix XenApp server adapter (if you choose to use Windows Authentication — if not, take note of the account name and password for database access that you will include in the database connection string)
- Grants read access to the service account running the inventory beacon engine (if you choose to use Windows Authentication — if not, take note of the account name and password for database access that you will include in the database connection string).

The script generates the SQL schema for the database, including creating the appropriate stored procedure. Take note of the connection string needed to connect to this database for your future reference.

# Installing the XenApp Server Agent

IT Asset Management (Cloud)

This procedure assumes you have already downloaded the XenApp server agent archive and unzipped it to a convenient location (if not, check back in [Creating the Staging Database](#)).



## **To install the XenApp server agent:**

1. In your unzipped archive, navigate into the \Citrix XenApp Server Agent subdirectory.
2. Copy the appropriate sub-folder to match your installed version of Citrix Virtual Apps (formerly Citrix XenApp) (noting that the XenAppAgent folder applies to Citrix XenApp versions 7.5 or later and Citrix Virtual Apps versions 7.1808 or later):

- XenAppAgent6
- XenAppAgent65
- XenAppAgent



**Tip:** If you have different versions of Citrix Virtual Apps (formerly XenApp) deployed in different domains, install the appropriate agents on the correct XenApp servers. Agents for different versions may connect to a single staging database.

3. Using your network, a memory stick, or other available means, paste the entire folder into an appropriate location on your XenApp server(s).

For example, you could paste the folder at the root level (such as C:\XenAppAgent). Keep in mind that for version 6 and 6.5, you must install the agent on a single XenApp server for each Citrix farm. For version 7.5 (or later), you may choose to install an agent on one XenApp server in each site, or to install on only one XenApp server that has network access (and credentials) for all your sites.

4. Ensure that, on your XenApp server (Delivery Controller), there is an account with sufficient privileges to run the agent in production.

Such an account:

- Can run a Windows scheduled task on the XenApp server where the agent is installed
- Has read access to the file system on the XenApp server(s) where it is to collect inventory
- Has read access to any file shares used to house XenApp packages (versions 6 and 6.5 only), App-V packages, or applications hosted in VDI images
- For versions 6 and 6.5, is a Citrix Admin account (a limitation in the PowerShell API for those versions means that only a Citrix Admin can retrieve the list of XenApp servers in a farm)
- For version 7.5 (or later), is a Citrix Read only admin account, with read access to the file systems of any other XenApp servers sharing locally published applications for which inventory is to be collected
- Is recognized by the SQL Server hosting the shared database (if you prefer to use Windows Authentication for access to the staging database; otherwise, you may choose to include an account name and password in the

connection string for the staging database).

## Create a Scheduled Task

### IT Asset Management (Cloud)

The XenApp server agent must be run locally on the XenApp server, where it collects inventory and transfers the data immediately to the staging database, or to an inventory beacon for upload to the central compliance database. This is triggered by a Windows scheduled task on the XenApp server.

Because the XenApp server agent, as its first action for each inventory collection, clears all old data from the staging database, it is important that the XenApp server agent does not run at the same time as the inventory beacon collects data from the staging database (otherwise, corrupt or incomplete data may result). A buffer of 2 hours provides a good safety margin (depending on the scale of your Citrix Virtual Apps implementation).

Another consideration is that you want your Citrix Virtual Apps inventory uploaded to the central IT Asset Management database before the system import and compliance calculations take place. Typically, this process starts around 2am central server time. A two-hour upload buffer should be more than adequate.

These considerations suggest (within a single time zone) a collection schedule around 10pm, an inventory beacon connection around midnight, and everything in place for the nightly compliance calculation.



**Tip:** The central cloud servers are on US West Coast time, and German time, respectively.

The process for setting up Windows scheduled tasks varies across different editions of Windows Server. The following example is for Windows Server 2012. Adjust for your XenApp server's conditions.



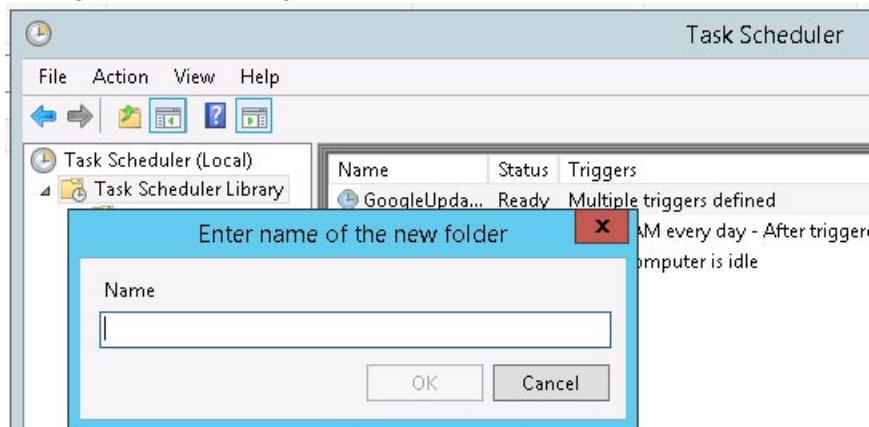
#### To create the scheduled task (Windows Server 2012 example):

1. In Windows Explorer, navigate to **Control Panel > System and Security > Administrative Tools**, and double-click **Task Scheduler**.

The **Task Scheduler** window appears.

2. In the navigation tree on the left, select **Task Scheduler Library**, and then in the **Actions** list on the right, click **New Folder....**

A dialog appears for entering the folder name.



A suggested value is IT Asset Management.

3. Click **OK**, and select the new folder in the navigation tree.

4. Select **Action > Create Task....**

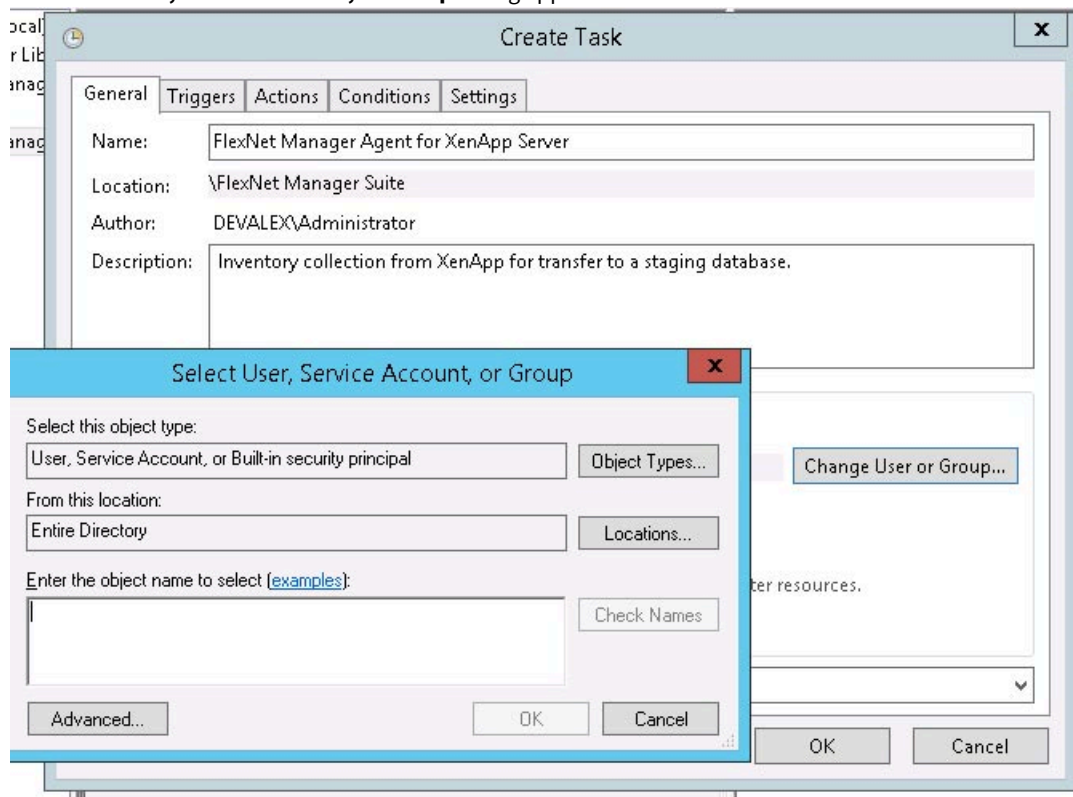
The **Create Task** dialog appears.

5. Enter an appropriate **Name**, such as XenApp server agent, and add any **Description** to help future maintenance of this task.

Your description may be something like Collects application and access information from a Citrix XenApp server and transfers this information to a staging database.

6. Click **Change User or Group....**

The **Select User, Service Account, or Group** dialog appears.



7. Enter the account name that is to run the scheduled task, and click **OK**.

This is the account you identified in [Installing the XenApp Server Agent](#), and you can check the requirements for this account there.

8. Further down in the **Security options** group, select **Run whether user is logged in or not**.
9. Switch to the **Triggers** tab, and click **New....**

The **New Trigger** dialog appears.

10. Ensure that the default setting **Begin the task On a schedule** is selected, set the parameters for the schedule, and from the **Advanced settings** group, be sure that **Enabled** is selected.

The suggested schedule is daily at 10pm local time, but be sure that this suits the upload procedures for your enterprise.

11. Switch to the **Action** tab, and click **New...**

The **New Action** dialog appears.

12. Ensure that the default **Action**, **Start a program**, is selected, and browse to your local copy of `FnmpXenAppServerAgent.exe`.

13. In the **Add arguments (optional)** field, specify all the command-line arguments you need for the agent.

All command line arguments are documented in [XenApp Server Agent Command Line Options](#). For common implementations, you need to define only the connection string to the staging database. In addition, if you are using Citrix XenApp 7.5 (or later), or Citrix Virtual Apps 7.1808 (or later) and want a single agent to collect inventory from multiple servers, you need the option to define those servers.

---

*Example 1: Command line arguments to connect to your staging database.*

```
-d "Data Source=192.168.13.38;Initial Catalog=MyStaging; User
ID=accountName;Password=password"
```

*Example 2: For Citrix XenApp 7.5 (or later) and Citrix Virtual Apps 7.1808 (or later), accessing multiple XenApp servers and recording their details in a common staging database (all on one line):*

```
-d "Data Source=192.168.13.38;Initial Catalog=MyStaging; User
ID=accountName;Password=password"
-s "localhost, xda01.fqdn.com"
```

14. Click **OK**.
15. Optionally, make any preferred adjustments to the **Conditions** or **Settings** tabs (normally the defaults are acceptable).
16. Click **OK** to close the **Create Task** dialog.

The new task appears in the list of scheduled tasks for this server.

17. Right-click the new task, and click **Run** in the context menu.

This checks that the scheduled task completes successfully.

18. Validate operations in the following ways:
  - Review the log file (`FnmpXenAppAgent.log` in the same folder as the agent's executable file) for any errors or warning messages.
  - Use Microsoft SQL Server Management Studio to check the contents of the staging database.

## Create Connections for Data Upload

IT Asset Management (Cloud)

The XenApp server agent gathers inventory information from your XenApp server, and saves it in a staging database. Now an inventory beacon is responsible for uploading the data to the central operations databases of IT Asset

Management. This requires two things:

- Defining a connection to the staging database.
- Defining a schedule to trigger collection of inventory data from the staging database.

In addition, with the exception of XenApp version 7.5, a connection to the appropriate database is strongly recommended, as this allows tracking who is actually using applications:

- If you are using XenApp 6.0 or 6.5, the connection is to your EdgeSight server database
- If you are using XenApp 7.6 (or later) or Citrix Virtual Apps 7.1808 (or later), the connection is to the XenDesktop database included with Citrix Virtual Apps (formerly Citrix XenApp).

Perform this process on the inventory beacon.



#### **To create connections for data upload:**

1. Log in to your selected inventory beacon.



**Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.

2. If you have not already specified a schedule (or two) that can be linked to the connection(s) you are about to create, it's convenient to do so now.

There are two connections for each version of Citrix Virtual Apps (formerly XenApp) except version 7.5, which needs only one. Remember that you already decided on the schedule for data collection on the XenApp server (see [Create a Scheduled Task](#)), and the schedules on the inventory beacon need to tie in with that plan. For example, you might schedule the connection to the staging database at midnight, and the secondary connection for usage data at 12:15am. For details about creating a schedule on the inventory beacon, see *IT Asset Management Help > Inventory Beacons > Scheduling Page > Creating a Data Gathering Schedule*.

3. In the navigation pane on the left, select the **Inventory systems** page, and towards the bottom of the page, click **New...**


The **Create SQL Source Connection** dialog opens.



**Tip:** The **New...** button defaults to creating a connection for Microsoft SQL Server. If you use the down arrow on the split button, you can choose between *SQL Server*, *Spreadsheet*, *PowerShell*, and *Other connections*. However, while you are creating a connection to a Microsoft SQL Server database (regardless of the **Source Type** of the connection), use only the *SQL Server* option.

4. Complete the values in the dialog, as follows:

Control	Comments
<b>Connection Name</b>	A descriptive name for this connection, such as Citrix Virtual Apps <i>ServerName Staging DB</i> .
<b>Source Type</b>	Select <b>Citrix XenApp (Server Agent)</b> . (Don't be confused by <b>Citrix XenApp (EdgeSight)</b> , which you may use shortly.)

Control	Comments
<b>Server</b>	<p>Type the server name or IP address where the database is hosted. Use the special value (localhost) if the database is installed on this same inventory beacon server. If the database instance you need is not the default one on the server you identify, add the instance name, separated with a backslash character. Example:</p> <pre>(localhost)\myInstance</pre>
<b>Authentication</b>	<p>Select one of:</p> <ul style="list-style-type: none"> <li>• <b>Windows Authentication</b> — Select this option to use standard Windows authentication to access the database server. The credentials of the account (on the inventory beacon) running the scheduled task for importing inventory are used to access the SQL Server database. This account must be added to an Active Directory security group that has access to the database.</li> <li>• <b>Windows (specific account)</b> — Use the following two fields (enabled when you make this choice) to specify an account on the inventory beacon that can make a connection to the SQL database.</li> <li>• <b>SQL Authentication</b> — Use the following two fields to specify an account and password registered as a user with database access on SQL Server. This account is used to access the database, regardless of the local account running the scheduled task on the inventory beacon server.</li> </ul> <hr/> <p> <b>Tip:</b> The account used needs read-only privileges.</p>
<b>Username</b>	The account name used for <b>SQL authentication</b> , or <b>Windows (specific account)</b> . (Not required for <b>Windows Authentication</b> .)
<b>Password</b>	The password for the account name required for <b>SQL authentication</b> , or <b>Windows (specific account)</b> . (Not required for <b>Windows Authentication</b> .)
<b>Database</b>	Enter the name of the database, or use the pull-down list to select from database names automatically detected on your specified server. For example, for a connection to Technopedia, select BDNA_Publish from the drop-down list.



Control	Comments
<b>Connection is in test mode (do not import results)</b>	<p>Controls the uploading and importing of data from this connection:</p> <ul style="list-style-type: none"> <li>When this check box is clear, the connection is in production mode, and data collected through this adapter is uploaded to the central server and (in due course) imported into the database there.</li> <li>When the check box is set: <ul style="list-style-type: none"> <li>The adapter for this connection is exercised, with data written to the intermediate file in the staging folder on the inventory beacon (%CommonAppData%\Flexera Software\Beacon\IntermediateData)</li> <li>The immediate upload that normally follows data collection is suppressed, so that you can inspect the contents of the file</li> <li>The catch-up process that retries stalled uploads, normally scheduled overnight, runs as usual and uploads the file to the central server</li> <li>At the central server, the file contents are discarded (and not imported into the central database).</li> </ul> </li> </ul>
<b>Overlapping Inventory Filter</b>	This control does not apply to Citrix Virtual Apps inventory records, and you may leave it at the default setting.

#### 5. Click **Test Connection**.

- If the inventory beacon can successfully connect to the nominated database using the details supplied, a `Database connection succeeded` message displays. Click **OK** to close the message. Click **Save** to complete the addition. The connection is added to (or updated in) the list.
- If the inventory beacon cannot connect, a `Database connection failed` message is displayed, with information about why that connection could not be made. Click **OK** to close the message. Edit the connection details and retest the connection.

You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.

- When the first connection to the database is successful, and if you are using any Citrix Virtual Apps (XenApp) version other than 7.5, repeat the steps above to define a second connection to the usage database (EdgeSight database for version 6.x, and Citrix Virtual Desktops (XenDesktop) database for version 7.5 and later). This time through, for all these versions set the **Source Type** control to **Citrix XenApp (EdgeSight)** (including for Citrix Virtual Apps (XenApp) version 7.5 and later).
- On the **Inventory systems** page, from the list of connections select the one to the staging database you created first in this process, and below the list, click **Schedule...**
- In the resulting dialog, choose the schedule you wish to use for this connection, and then click **OK** to close the dialog, and **Save** to apply the selected schedule to your connection.
- For any Citrix Virtual Apps (XenApp) version other than 7.5, on the **Inventory systems** page, from the list of connections select the EdgeSight connection, and repeat the scheduling process, choosing the second schedule you created.

Don't forget, as you move this Citrix XenApp server adapter into production, to ensure that **Connection is in test mode (do not import results)** is clear (not checked).

## 3

# Command-Line Options

IT Asset Management (Cloud)

For those times when you want to control execution of the command-line agent directly, this chapter covers all options.

## XenApp Server Agent Command Line Options

IT Asset Management (Cloud)

XenApp server agent (`FnmpXenAppAgent.exe`) is a command line tool which runs regularly on a Citrix XenApp server (for Citrix Virtual Apps) to determine which end-users have the right to run applications through that server. The data it collects is sent back to IT Asset Management and processed further once the inventory import process runs. Here are the details for manual operation from the command line, or configuring a scheduled task.

**Syntax:**



`FnmpXenAppAgent.exe [options...]`


**Options**

`-d connection`  
`-f site_name_override`  
`-h`  
`-i true/false`  
  
`-o output_file`  
`-r true/false`  
`-s servers`  
`-t timeout`

`-v 0/1`

where

<code>-d connection</code>	A database connection string to your staging database. Refer to <a href="http://www.connectionstrings.com/sql-server-2008">http://www.connectionstrings.com/sql-server-2008</a> for some examples.
 <b>Note:</b> Do not use the <code>-d</code> option and the <code>-o</code> option at the same time.	
<code>-f site_name_override</code>	Force an override of the default Citrix Virtual Apps farm/site name.
<code>-h</code>	Displays usage for the XenApp server agent.
<code>-i true/false</code>	(Default false.) Ignore errors. Used only for debugging purposes so that the adapter runs end-to-end and logs all issues in the log file (in the same directory as the agent executable).
<code>-o output_file</code>	The full path of a file to store the output of the XenApp server agent as it runs. Use such an output file for debugging purposes, to see the content collected by the agent. The output file is not required for normal operations (the collected data is uploaded directly to the staging database).
 <b>Note:</b> Do not use the <code>-d</code> and <code>-o</code> options at the same time.	
<code>-r true/false</code>	<p>Option only for Citrix XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later). Values may be:</p> <ul style="list-style-type: none"> <li><code>false</code> - (default) Disable remote registry environment variable resolution, and enable <i>local</i> variable resolution on the XenApp server.</li> <li><code>true</code> - Direct the XenApp server agent to remotely read the Windows registry on the Citrix Virtual Apps <i>client</i> machine, using values there to resolve environment variables embedded in file paths.</li> </ul> <p>As background, when a file path in Citrix contains a Windows environment variable (for example, %ProgramFiles%), the variables can have different values on the server and on the client device. This switch lets you choose the value that is resolved locally on the XenApp server, or choose the environment variables remotely read from the registry of the Citrix Virtual Apps client device.</p> <p>Resolving environment variables using this command requires both of the following:</p> <ul style="list-style-type: none"> <li>The <b>Remote Registry</b> service must be started (status of <b>Running</b>) on the client machine in order to resolve remote environment variables</li> <li>The account executing the XenApp server agent must exist on the Citrix Virtual Apps client machine to properly resolve environment variables using this command.</li> </ul>

<code>-s servers</code>	<p>Option only for Citrix XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later). For both 6.5 (which does not support this option) and 7.5 or later (where the option may be omitted), the XenApp server agent assumes that it is running on the XenApp server from which it is to collect inventory. In both systems, therefore, you may handle multiple XenApp servers by installing the XenApp server agent on each one. When the agent is installed locally on the XenApp server from which it gathers inventory, omit this option.</p> <p>In Citrix XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later), you have the option for a XenApp server agent installed on a single XenApp server to collect the inventory for all XenApp servers in a server farm. To do this, create a comma-separated list of fully-qualified domain names or IP addresses for all the servers from which this agent should collect inventory. Inside such a list, use the keyword <code>localhost</code> to include the server on which the XenApp server agent is installed.</p>
<div>  <b>Note:</b> The account executing the XenApp server agent must have permissions to connect to each of the servers named in your list. </div>	
<code>-t timeout</code>	The timeout period (in seconds) when connecting to the staging database (default 600 seconds). If the timeout expires, the upload fails, and the data collected from the XenApp servers on this occasion is lost. An entry is made in the log file (in the same directory as the agent executable) to record the failed connection.
<code>-v 0/1</code>	(Default 1). Sets logging messages to verbose mode. For less information, specify <code>-v 0</code> .

## Examples

The following examples are split across several lines for readability. Run each example on a single command line.

The examples depend on which form of authentication you have chosen to use:

- Windows Authentication, using a Windows (Active Directory) account that is recognized by the SQL Server hosting your staging database
- A separate SQL Server account specified in the database connection string.

(You decided on your approach in [Installing the XenApp Server Agent](#).) These two approaches are mutually exclusive, so each example applies to one or the other, but not both.

Use Windows Authentication (for the account running the scheduled task, or the command line) to connect to the staging database with a ten minute (600 second) timeout:

```
FnmpXenAppAgent.exe
-d "Server=192.168.13.38;Database=MyStaging;Trusted_Connection=yes"
-t=600
```

Using SQL Server authentication, identify a particular user name and password to connect to the staging database:

```
FnmpXenAppAgent.exe
-d "Data Source=192.168.13.38;Initial Catalog=MyStaging; User
ID=accountName;Password=password"
```

Using Windows Authentication (with the relevant account known to both SQL Server hosts) to collect inventory in XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later) from two servers, saving the collected data in an XML file for debugging purposes:

```
FnmpXenAppAgent.exe  
    -s "localhost, xda01.fqdn.com"  
    -o "c:\XenAppTest.xml"
```

## 4

# Validation, Troubleshooting, and Limitations

IT Asset Management (Cloud)

This chapter may assist in diagnosing operations of the adapter, as well as explaining certain inherent limitations in its operation.

## Validation and Problem Solving

IT Asset Management (Cloud)

Because the Citrix XenApp server adapter has a number of moving parts, validation and problem solving may also involve multiple steps.

After initial implementation, the simplest validation is simply to inspect the additional installer evidence (for App-V applications) and file evidence (for VDI applications) that are collected by the adapter, and displayed in the web interface for IT Asset Management. Keep in mind that to complete the end-to-end process, you may need to

- Manually link the evidence to an appropriate application
- Ensure that the application is linked to a suitable license
- Record entitlements on the license, typically by linking purchases to it.

Also remember that you must be importing information from Active Directory prior to importing inventory through the Citrix XenApp server adapter.

When more detailed analysis is required, you may use the following checks.

### XenApp server agent

The XenApp server agent, installed on your XenApp server, records a log file in the same folder where it is installed. The log file is replaced at each inventory collection (that is, each time the scheduled task triggers the agent). Review the log for details of any problems. To increase the level of detail, run the agent with the command-line option `-v 1`.

To see all the information that the XenApp server agent has collected, run the agent without a `-d` option (the path to the staging database) and instead using a `-o` option with a path to a convenient local folder. This saves a plain-text XML

file of the collected inventory that you can inspect in your preferred text editor. This is a valuable check point when some inventory is being returned, but particular expected applications seem to be missing. If these are missing from a file output with the `-o` option, look for reasons preventing agent access to the source information. Examples might include credentials or access rights to folders containing packages.

If you use the `-o` option, don't forget to replace it with the `-d` option for normal operations!

## Staging database

When records missing from the web interface for IT Asset Management are present in the output of the XenApp server agent (see previous section), next use Microsoft SQL Server Administration Studio to inspect the contents of the staging database. Recall that the contents of the staging database are over-written with each inventory collection by the XenApp server agent. This means that there may be legitimate differences between an output file obtained in the previous section, and the database contents examined in this section. Such differences may come about if there is additional access granted to Citrix Virtual Apps (formerly XenApp) applications (the source data) in between the time the agent is run to output the test file, and the time it is run to populate the staging database. In general, however, there should be a high degree of correlation between the two data sets.

## The inventory beacon intermediate file

Next, you can validate the data set that the inventory beacon collects from the staging database. Simply trigger the connection to the staging database in test mode, as described in [Create Connections for Data Upload](#). This allows you to inspect the zip archive, which would otherwise be uploaded to the central server, in `%CommonAppData%\Flexera Software\Beacon\IntermediateData` on the inventory beacon. Provided that you make comparisons before the next run of the XenApp server agent, you should find 1:1 correspondence between the data in the staging database and the intermediate file.

## Uploads and imports

If the required data has made it into the intermediate file on the inventory beacon, you may need to debug uploads from the inventory beacon to the central server (for example, see *IT Asset Management Help > Inventory Beacons > Inventory Beacon Reference > Troubleshooting: Inventory Not Uploading*).

Keep in mind that there is a delay between the upload from the inventory beacon to the central server, and the appearance of the data in the web interface for IT Asset Management. There must be an inventory import and compliance calculation that occurs between these two events. If you are a member of the Administrator role, in the web interface you may manually trigger an import and compliance calculation.

# Limitations

## IT Asset Management (Cloud)

The following limitations apply to the Citrix XenApp server adapter:

- Information about who has access to applications on the one hand, and about who actually uses applications on the other, is collected separately. Usage data relies on a Citrix EdgeSight server (for XenApp 6.x) or the Citrix Virtual Desktops (XenDesktop) database included with XenApp version 7.5 (or later) and Citrix Virtual Apps version 7.1808 (or later). Since XenApp 7.5 does not support it, no usage information is available for this version; and if you are using any other version without the appropriate database connection, again no usage information is available.



- When the XenApp server agent connects to a VDI image to read the applications it contains, and the image is not currently running, the agent attempts to start up a server running the image for interrogation (and will shut it down again afterward). This relies on the remote support of power actions (on and off, and so on) for the source image. Where these are not available, or the Citrix Virtual Desktops (formerly XenDesktop) do not have sufficient resources to spin up another image, the start-up attempt fails. In these cases, the agent imports the name of the executable, but it is missing the version, company, and description details. These cases appear in the list of file evidence with the executable name, but with the name, version, company, and description columns blank.
- Citrix Virtual Apps (formerly Citrix XenApp) supports "application delivery from a remote PC" and "application delivery from the cloud". Neither of these is supported by the Citrix XenApp server adapter, and no inventory is returned for such cases.
- To improve performance compared with earlier versions of the Citrix XenApp server adapter, the XenApp server agent no longer collects user names and computer names through the XenApp server. Instead, it collects only the Active Directory security IDs (SIDs) from Citrix Virtual Apps (formerly XenApp), and relies on the separate import from Active Directory to flesh out the SIDs with more complete identities. This has two immediate implications:
  - If you are upgrading from an earlier version of the Citrix XenApp server adapter, the data from the access control lists (ACLs) on the XenApp server is removed on upgrade. An import from Active Directory is then required to populate the SIDs and other identity details. After the first inventory import in the upgraded system, the access data is repopulated.
  - If Active Directory information is not imported by an inventory beacon for the domain(s) in which the XenApp servers (the ones hosting the XenApp server agent) are located, users and computers newly registered in Active Directory (since the last import of Active Directory data) will not be recognized or displayed in IT Asset Management.
- For XenApp version 7.6 (and later) and Citrix Virtual Apps version 7.1808 (and later), application usage information is available only for App-V packages and applications, and streaming profile applications. Usage information is not available for file-based applications, including VDI applications delivered through Citrix Virtual Apps.

## 5

# Database Impacts

## IT Asset Management (Cloud)

This chapter provides an overview of database tables within IT Asset Management that are affected by the adapter. These brief notes can be augmented by reviewing the IT Asset Management Schema Reference PDF file for the current release.

## Affected Database Tables

### IT Asset Management (Cloud)

The Citrix XenApp server adapter causes data to be imported to the following database tables in the operations databases for IT Asset Management (specifically the compliance database). You may prepare custom reports against these:

- **ComplianceDomain** records the domain of the XenApp server where the XenApp server agent is installed. If the record does not already exist, on import both the FQDN and the flat name are populated.
- **ComplianceUser** records are created for any user names identified in the Citrix Virtual Apps inventory but not previously in the operations databases. In these new records, only the **UserName**, **SAMAccountName**, and domain are available and saved (with **ComplianceUserID** calculated automatically). (The domain is a link to the **ComplianceDomain** table, which once again is updated as required with any new records. Such updates should be rare, since domains should be identified from Active Directory.) Because the **ComplianceUser** records created from Citrix Virtual Apps inventory imports are far from complete, you may want to enhance these with additional information.
- **ComplianceComputer** records may be created if a **ComplianceUser** record is created (or identified) that has no link to an existing **ComplianceComputer** record. For many types of license, the consumption record is linked to an individual **ComplianceComputer**, so when it is impossible to identify a computer for a particular **ComplianceUser**, a new skeleton computer record is created. These have a computer name of the form

*UserName* (Remote)

and have their type shown as **Remote Device**. (This type is identified through a foreign key to the **ComplianceComputerType** table.) They are also linked to the **ComplianceUser** for whom they are created.



**Tip:** *If, in future, inventory from another source identifies the same `ComplianceUser` as either the assigned user or the calculated user for a computer identified by that inventory, then this placeholder record for the remote device is removed, and any license consumption recorded against it is moved to the newly-identified (real) computer that belongs with the user.*

- For App-V applications managed by Citrix XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later), or any Citrix Virtual Apps applications managed by version 6.x:
  - `InstallerEvidence` records are created as the primary inventory data for the use of those applications.
  - `InstalledInstallerEvidence` records are create to link that evidence to the appropriate `ComplianceComputer` records.
  - Where the application name, version, and publisher in the `InstallerEvidence` table match an existing evidence rule for an application, an entry is created in the `SoftwareTitleInstallerEvidence` table to link (by foreign keys) the installer evidence to the application record in the `SoftwareTitle` table.
- For VDI applications managed by Citrix XenApp 7.5 (or later) or Citrix Virtual Apps 7.1808 (or later):
  - `FileEvidence` records are created as the primary inventory data for the use of those applications.
  - `InstalledFileEvidence` records are create to link that evidence to the appropriate `ComplianceComputer` records.
  - Where the file name, version, company, and description in the `FileEvidence` table match an existing evidence rule for an application, an entry is created in the `SoftwareTitleFileEvidence` table to link (by foreign keys) the file evidence to the application record in the `SoftwareTitle` table.
- `InstalledApplications` records are created for any applications identified in the `SoftwareTitle` table, linking the application to the `ComplianceComputer` record.



# Data Platform Integration

## IT Asset Management (Cloud)

This part introduces the integration between IT Asset Management and Data Platform, an umbrella product that includes both Technopedia and Normalize.

The method of integration is a straight-forward connector, available without special installation on your inventory beacons. On the Data Platform side, an enhanced license term and a new data pack are required (see [Configuring the Data Platform Connector](#) for details).

Such simplicity means that the main use for the documentation in this part is to understand the mapping between

- Database tables and columns that underlie Normalize, and
- The destination tables and columns in the compliance database for IT Asset Management.

Also important is the embedded commentary on the current limitations of Data Platform as a *sole* data source driving license compliance calculations. When you have multiple inventory sources, imports from Data Platform can be a valuable addition to your data gathering.

## Supported version

The Data Platform v5 connector supports import only from version 5 of Normalize.

## 1

# Integration with Data Platform v5

## IT Asset Management (Cloud)

Data Platform version 5 (formerly from BDNA, and now from Flexera) combines Technopedia, a reference catalog with 120 million datapoints across more than 1.7m products, and Normalize, which provides a unified view of the data applicable to your environment.

To integrate this data for your use in license and asset management within IT Asset Management, release 2024 R2.3 includes a connector that imports normalized data from Normalize v5 into your compliance database. The imported data includes:

- Records of inventory devices (for both physical servers and virtual machines) found in your environment by the inventory tools (such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) or BMC ADDM) that are feeding into Normalize v5
- Details of applications recognized within your enterprise through those tools
- Installation records that link the found software to the inventory devices where it is installed
- Details of end-users who log onto the inventory devices
- Where available, records of application usage (how often, and for how long, an end-user utilized an application).

Application records are dealt with in the same way as inventory data from any other source. This means that the data representing applications within Normalize v5 is imported into IT Asset Management in the first place as inventory evidence (in this case, as installer evidence). What happens next depends on the details of the software:

- If the application is already known to the Application Recognition Library (ARL), and therefore has a Flexera ID included in the import from Data Platform, the evidence is recognized by the ARL and a matching installed application record is created (if it did not exist already) and displayed in the **Installed Applications** page. For such records, the inventory evidence type is set to `FlexeraID`, and the installer evidence record is linked to the application record.



**Tip:** It may happen that the application is only recognized by a later update of the ARL than is currently deployed in your enterprise. If this occurs, a widget on the **System Health Dashboard** (on the **Data Status** page (**Data Collection > IT Assets Inventory Tasks > Data Status**)) highlights the need to refresh your local, downloaded ARL (or you can wait until the next automated download, which typically happens each weekend).

- If the installer evidence is not [yet] recognized by the ARL, then following normal system processes, it cannot be linked to an application record, and it remains in the **Unrecognized Evidence** page, in the listing under the **Installer**

**evidence** tab. For these cases, the inventory evidence type is displayed as `Data Platform`. (Future updates to Data Platform and the ARL may mean that the same installer evidence re-imported thereafter may be recognized, and an installed application record automatically created at that time. For this reason, it is best not to manually trigger application creation from such records of *public* applications, but to wait for Flexera's continuing alignment work to automatically complete the links for you. If you find such a case that is needed in your enterprise with some urgency, you can ask Flexera to prioritize its delivery through a later release of the ARL.)



**Tip:** A subset of these unrecognized cases may be application records that you created locally within your enterprise (records that in Data Platform are called "private applications", and in IT Asset Management are called "local applications"). As such records may never be matched by the ARL, you may wish to start from the unrecognized evidence record to re-create your local application record within IT Asset Management. In summary, best practice is to manually trigger the creation of application records only for locally-created, 'private' applications; and to await updates from Flexera for 'public' applications.

## Connector, not adapter

The integration with Data Platform v5 is not strictly an "adapter", for the following reasons:

- It does not require any custom XML file to be maintained through either the Inventory Adapter Studio or the Business Adapter Studio
- There is no need (nor purpose) for any staging database
- There is no separate download and installation required (with the possible exception of the initial release, where shipment of the connector is not synchronized with the product release; but in future it is embedded as a standard part of the product).

Instead, this integration is through a *connector* that is available on any up-to-date inventory beacon. You set up the import in exactly the same way as for any other inventory connection: specifying the details in the new inventory connection dialog on the **Inventory systems** page on your preferred inventory beacon; and nominating your preferred schedule for the regular imports. Full details are included below in [Configuring the Data Platform Connector](#).

## Inventory gaps and limitations

Following topics cover the mapping of data from within Data Platform to the import tables within the compliance database in IT Asset Management. (Hints are also given for tracking data from the import tables to the operational tables within the compliance database, with further information being available in the *IT Asset Management Schema Reference* PDF, available through the title page of online help.) Within those topics are specific comments about the impact of particular inventory gaps on tracking your license compliance.

At a high level, inventory gathered *only* from Normalize v5 is insufficient for:

- Oracle Processor license calculations (missing the partial number of processors)
- IBM PVU license calculations (missing the partial number of processors)
- Any licenses that rely on details about the host settings for virtual machines, such as *assigned* core or processor counts (for VMs, this connector can only import the virtual host, and no other details of the VM, including pools, clusters, capping and the like). Notice, however, that inventory is normally gathered separately both from the virtualization host and from available guest machines, so that records are available for each device's view of its own properties.

## Validate your data processes

When there are inventory gaps, the best-practice work-around is to add a second inventory source that fills the gaps, and allow IT Asset Management to merge the sources to produce a unified data set. The risk factor is that record merging may fail (because of different values in key fields), leading to duplicate records in your compliance database that cause over-counting.

Such a case is the possible merging of imports from Data Platform with another source, such as the FlexNet Inventory Agent. One of the key fields for merging inventory device records is the domain name that the device reports, and currently the values for domain are by default saved differently. Data Platform v5 typically saves a flat domain name (such as `flexera`); and while IT Asset Management can handle either format, by default it saves and uses the DNS domain name format (distinguished name) that includes the root domain (such as `flexera.com`). At first glance this difference looks high-risk; but there is a compliance database table that helps to resolve this issue, called the `Domain` table. Since this stores both the fully-qualified distinguished name for the domain, and also its flat name, it allows reliable matching between the two forms, *provided that* it has been populated with the appropriate records before a mapping is requested.

So, to prevent duplicate inventory device records, it is important to ensure that the `Domain` table is populated with all your domain names in both formats before importing from Data Platform (or any other inventory source that reports the flat name for domains). There are three ways to do this, of which the first is best practice:

- Run an import from Active Directory. This populates the `Domain` table appropriately with both domain name formats.
- Import FlexNet inventory, since the FlexNet Inventory Agent reports both name formats.
- Import inventory from Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) 2012 or later, since from that release, Microsoft Endpoint Configuration Manager also reported the domain name in both formats.

The first option is clearly preferable since it gives the most complete coverage of all your domains. Therefore, before importing from Data Platform, ensure that your Active Directory imports are up-to-date.



**Tip:** *If you have domains that are outside the reach of your Active Directory implementation, you can create a spreadsheet and write a business adapter to populate the `Domain` table.*

A match between host name, domain, and serial number is sufficient for merging (de-duplicating) inventory device records. Where these high-priority values are not available, IT Asset Management continues checking lower-priority properties including:

- `HostIdentifyingNumber`
- `HostType`
- `ILMTAgentID`.

Since all these properties are unavailable from Data Platform v5 (see [Imported Computers \(Inventory Devices\)](#)), this makes correct mapping between the naming formats of your domains all the more important in avoiding duplicate device records.



**Attention:** *Even when you follow best practice and use the above techniques to populate the `Domain` table with both the flat name and the distinguished name for each domain, issues with domain reporting may still cause creation of duplicate records for computers or users. This is because some data sources (notably HP-UD and ADDM) report domain properties for users and computers in ambiguous ways, and where overlapping data comes from another source (including coming through Data Platform even from the same original source), duplicate records may result. You may*

*both prevent and repair such duplication simply by setting your **Primary** inventory source to a known-good choice from the overlapping sources, such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), FlexNet inventory, or Data Platform. (To find this setting, go to the **Data Imports** page (**Data Collection > IT Assets Inventory Tasks > Data Imports** and select the **Inventory Data** tab)). To clear duplicate records previously created through this issue, set your good source that's reporting the domain as primary, and wait for (or trigger) a full import and compliance calculation. By default, these full imports and calculations happen overnight, so you can check that your previous duplicates are cleaned up the day after adjusting your primary inventory source.*

If you are considering the use of two data paths for overlapping inventory records (such as having a common source imported directly into IT Asset Management on the one hand, and imported through Data Platform v5 on the other), one caveat to be aware of is the need for additional expertise. Suppose that you are trying to analyze a problem with recognition of a particular application. With two data paths, you must first determine which data path delivered the record, because the recognition logic differs between the paths:

- If the path was direct from the inventory source (such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM)) into IT Asset Management, then the evidence rules in the Application Recognition Library (ARL) must be assessed for effective matching.
- If that path from the inventory source led through Data Platform v5, you first need expertise to analyze the recognition process within that domain, because it is Data Platform (and in particular Normalize v5) that has recognized the application. In addition, you must validate that Data Platform v5 is supplying a Flexera ID for the application it recognizes, and that the same Flexera ID is listed as evidence in your installed ARL — otherwise, despite its initial recognition, the 'application' languishes in the **Unrecognized Evidence** page within IT Asset Management.

In summary, combining both data paths provides a growth opportunity.

## Supported versions

The connector is validated for operations between Data Platform v5 and IT Asset Management 2018 R1. Within that framework, the following recommendations apply:

- **Data Platform:** The preferred release of Data Platform (and in particular, Normalize) is 5.5.4 (or later), as this release fixed a problem where, for virtual machines, imported data contained incorrect values for the count of **Cores** and **Threads** (or logical processors). Both these values are visible in IT Asset Management on the **Hardware** tab of the inventory device properties. In the same location, you may also manually override both these values when a correction is needed, and your overridden values are unaffected by future inventory imports (for more details, see the online help for the **Hardware** tab).
- **IT Asset Management:** While formal validation is limited to release 2018 R1 and later, informal spot testing suggests backward compatibility of the connector should extend back to at least release 2017 R1 of IT Asset Management.



**Important:** If you are testing with an earlier version of IT Asset Management, be sure that, on all relevant inventory beacons, you remove the superseded folder `C:\ProgramData\Flexera Software\Compliance\ImportProcedures\Inventory\Reader\BDNA` before distributing the new `Data Platform v5` folder as a replacement (full details about the new folder are included in [Configuring the Data Platform Connector](#)).



## 2

# Configuring the Data Platform Connector

IT Asset Management (Cloud)

Configuring the Data Platform connector is a one-time task, and quite straight-forward. Thereafter, the imports continue at the intervals you schedule. Have ready your connection details for the Normalize v5 database.



## To configure the Data Platform connector:

1. To minimize the risk of duplicate records when you have imports from multiple inventory sources, ensure you have completed Active Directory imports from all your domains.

For more details, refer back to [Integration with Data Platform v5](#). For guidance about Active Directory imports, see *IT Asset Management Help > Inventory Beacons > Active Directory Page*.

2. If necessary, apply for your updated Data Platform license to enable integration with IT Asset Management, by contacting your Flexera Support consultant.

This updated license is available free of charge to existing customers who have implemented Data Platform. It authorizes additional functionality and a new data pack. If you are unsure whether you already have this updated license, there are two ways you may check:

- You can send a question to Data Platform Support
- You can run the following SQL statement against your Normalize v5 database:

```
Select IS_SUBSCRIBED from BDNA_CP where CP_NAME = 'Technopedia Flexera Mapping'
```

If the result returns Y, your Data Platform instance includes the license term, and you may skip ahead to step 4. If it returns N, you need to apply for the license update.

3. When you receive the updated activation key:
  - a. In BDNA Admin, navigate to the **Preferences** page.
  - b. Ensure that the **Registration** tab is selected (top left).
  - c. Delete the current contents of the **Activation Key** field.

- d. Copy the key in full from the email you received, and paste it into the **Activation Key** field.
  - e. Click **Register New Key**.
4. Download the new data pack:
- a. Navigate to the **Technopedia** page in BDNA Admin.
  - b. Click **Start Catalog Sync**, and wait for the data pack to download.
5. Ensure that the Data Platform connector is in place in your IT Asset Management implementation.

You can check whether the Data Platform connector is already in place by inspecting any inventory beacon. In each case, open Windows Explorer and navigate to C:\ProgramData\Flexera Software\Compliance\ImportProcedures\Inventory\Reader. If you find a folder there called Data Platform v5, your connector is already in place, and you may skip ahead to the next major step. If not, wait until the connector is released through the central application server, after which the connector on your inventory beacons is updated automatically, and you can resume this process.

- a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
  - b. In that article, click **Adapter Tools for FlexNet Manager Suite - Cloud Edition**.  
A new browser tab may appear temporarily, and the download of **Adapter Tools for IT Asset Management 2024 R2.3.zip** commences.
  - c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as C:\temp on a central, accessible server).  
  
If your browser saves the file to a default location (such as your Downloads folder), move or copy it to the appropriate working location when the download is finished.
6. Ensure that you have your preferred schedule for imports from Data Platform set on the appropriate inventory beacon:
- a. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Tip:** Remember that you must run the inventory beacon software with administrator privileges.

- b. From the **Data collection** group in the navigation bar, choose **Scheduling**.
  - c. If there is not already a suitable schedule in the list, click **New...** and complete the details (see the online help for that page for more information). Otherwise, identify the schedule you will use.
7. Define the connection to your Data Platform implementation:
- a. Select the **Inventory systems** page (in the same group), and then click **New...**  
  
If you used the down arrow on the split button, use only the **SQL Server** option.
  - b. Complete the details, using a name that is distinct within the first few characters so that it is recognizable in a narrow column.



**Important:** Take care that:

- The **Source Type** is *Data Platform v5*

- 
- You provide the connection details for your Normalize database (not the Technopedia one).

See the help for this page for more details. When you are done, click **Save**.

8. Select your new connection from the displayed list, and click **Schedule...**
9. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.
10. At the bottom of the **FlexNet Beacon** interface, click **Save**, and if you are done, also click **Exit**.



**Tip:** Consider whether you want to select your connection, and click **Execute Now**, before you exit.

The connection is now configured, and imports data from Data Platform on your chosen schedule. Shortly after each import is completed, the inventory beacon automatically uploads the resulting inventory to the central application server. After the next full import and compliance calculation (typically scheduled overnight), the inventory is visible in the web interface of IT Asset Management.

## 3

# Data Mappings, Gaps, and Impacts

## IT Asset Management (Cloud)

The following topics provide a mapping between the source tables (and their columns) in the Normalize v5 database, and the corresponding destination tables and columns in the compliance database within IT Asset Management. This mapping covers only the first stage of the process, into the tables with names starting *Imported . . .* These are effectively staging tables that hold the data between the upload from Data Platform, and the major transition (or import) into the operational tables in the compliance database, normally followed immediately by the license compliance calculations (the calculations of consumption against entitlements). The descriptions also identify the operational tables where the data eventually resides, and for a more detailed understanding of those tables and their columns, see *IT Asset Management Schema Reference*.

The topics are separated by the target *Imported . . .* tables, each of which is aligned with a particular data object within IT Asset Management. Within each listing, the columns are in the same order that they appear in the *Schema Reference*, for your convenience if you are cross-referencing both documents.

As well as the list of imported data, a second list within each topic identifies the values that *cannot* be imported through Data Platform v5, together with notes on the impact that these gaps have on compliance calculations. Some gaps affect the suitability of this data source when you are managing certain types of licenses.

Each topic lists *every* column within the relevant *Imported . . .* table, so that if you are working across both this and the *Schema Reference*, there is no ambiguity about any column: it is either listed in the imports, or in the list of items missing from the import. Notice that this means some columns included for completeness appear in the list of "missing" items when in fact there is no expectation that they should be imported from any inventory source (or, to put it another way, they are *correctly* omitted from the import). These items are marked as "internally generated" by IT Asset Management.

## Exempted kinds of data

The following kinds of data are *not* imported from Data Platform v5 to IT Asset Management:

- File evidence — this evidence type is not needed from Data Platform, as the import of installer evidence includes all required information
- WMI evidence — this evidence type is not needed from Data Platform, as the import of installer evidence includes all required information
- Client access evidence (used for Microsoft Client Access Licenses, or CALs) — this evidence is not available in Data Platform v5

- App-V evidence — this evidence is not available in Data Platform v5, and App-V application usage cannot be tracked if Data Platform v5 is the single inventory source
- Oracle Database and Oracle option evidence, and Oracle GLAS evidence — this evidence is not imported from Data Platform v5, so that recognition of Oracle Database version and edition, or of Oracle options, is not possible if Data Platform v5 is the single inventory source (but this information, along with detailed content to deliver to Oracle License Management Service, is fully available within IT Asset Management using the FlexNet Inventory Agent).

## Comparison of Sources

### IT Asset Management (Cloud)

This topic has two parts:

- A comparison of the inventory sources that are supported by IT Asset Management and Data Platform v5
- For data sources fully supported by both, a summary of the inventory gaps (some of which may affect compliance calculations) with data imported from each source either directly into IT Asset Management, or firstly into Data Platform and then imported from Data Platform v5 into IT Asset Management.

For additional detail, down to the level of individual data fields, refer to subsequent topics.

### Supported inventory sources

For each of the data sources (listed alphabetically), this table displays:

- "Y" when either product offers out-of-the-box support for imports from that source
- A blank in either product's column when the data source is not supported
- "C" when either product has a *custom* extractor/adaptor available to collect data from the source and import it into the connecting product.



**Tip:** Support in IT Asset Management may be offered through a connector, requiring minimal configuration other than identifying the source; or an adapter, which may require a separate download and installation, and possibly setting up a staging database. In addition, IT Asset Management supports the development of custom adapters, for example, through the *Inventory Adapter Studio* (see [Inventory Adapter Studio](#)). For Data Platform, the equivalent terminology is *extractor*.

Inventory source	IT Asset Management	Data Platform v5
Symantec IT Management Suite (formerly Altiris)	Y	C
App-V Standalone	Y	
BMC BladeLogic Client Automation (previously Marimba)	Y	C
BMC Discovery (previously ADDM)	Y	Y
EdgeSight for Citrix Virtual Apps (previously XenApp EdgeSight)	Y	

Inventory source	IT Asset Management	Data Platform v5
Citrix XenApp server agent (for Citrix Virtual Apps)	Y	
FlexNet Inventory Manager (previously ManageSoft)	Y	
FlexNet Manager for Engineering Applications	Y	
HP Discovery and Dependency Mapping Inventory (DDMI)	Y	C
HPE Universal Discovery (HP-UD)	Y	Y
IBM BigFix Platform (previously Tivoli Endpoint Manager)	Y	Y
IBM License Metric Tool (ILMT) or HCL BigFix Inventory (see note 1 and 2)	Y	
JAMF Pro (previously Casper)	C	Y
Microsoft Exchange ActiveSync	Y	
Microsoft Office 365	Y	
Microsoft Endpoint Configuration Manager (previously Microsoft SCCM or SMS)	Y	Y
Salesforce	Y	
SAP (through FlexNet Manager for SAP Applications)	Y	
ServiceNow	(see note 3)	Y
SolarWinds Orion	C	Y
Symantec IT Management Suite (formerly Altiris)	Y	C
Tanium Asset	C	Y



**Note:** In the table above:

1. For the BigFix Inventory source type, the IBM License Metric Tool (ILMT) adapter must be selected. No BigFix Inventory source type is available.
2. For imports from IBM License Metric Tool (ILMT), only applications with IBM as the Publisher are recognized by the Application Recognition Library. For imports from HCL BigFix Inventory, applications from all publishers may be recognized.
3. IT Asset Management does provide for integration with ServiceNow, but in this case it is not importing raw inventory. ServiceNow is regarded as a source of business data rather than inventory data for IT Asset Management.

## Inventory gaps by import path

The following four inventory sources (column 1) are fully supported by both IT Asset Management and Data Platform v5. Based on the import path, there may be gaps in the imported inventory, and these gaps may be significant enough to prevent compliance calculations for related license types:

- The middle column identifies inventory gaps when IT Asset Management imports inventory data *directly* from each source
- The final column identifies inventory gaps if the inventory is first imported into Normalize v5 (within Data Platform), and subsequently imported from there into IT Asset Management.

Data source	Direct import to IT Asset Management	Import through Data Platform
BMC Discovery (previously ADDM)	No inventory gaps.	All virtualization data is missing.
HPE Universal Discovery (HP-UD)	No inventory gaps.	Advanced virtualization data missing ( <i>see note below</i> ). DNS domain name missing (domain flat name is available).
IBM BigFix (previously Tivoli Endpoint Manager)	All virtualization data is missing.	All virtualization data is missing.
Microsoft Endpoint Configuration Manager (previously Microsoft SCCM or SMS)	All virtualization data is missing.	Advanced virtualization data missing. DNS domain name missing (domain flat name is available).



**Note:** This table distinguishes between:

- Simple virtualization calculations for technologies like VMware ESX and Hyper-V — these require only the host/guest relationships to be tracked
- Advanced virtualization calculations for hardware partitioning technologies like HP-UX vPar, IBM LPAR, and Solaris zones — these require tracking of greater detail, such as partial processors and processor pool information. Information about clusters and host affinity (and the like) is also included in the advanced class.

And obviously, "all virtualization data" includes both of the above.

## Imported Installer Evidence

### IT Asset Management (Cloud)

The `ImportedInstallerEvidence` table is a staging table used for input of records about installation evidence. From here, de-duplicated and normalized records are merged into the `InstallerEvidence` table.

`InstallerEvidence` typically lists installer evidence left behind after a particular item of software has been installed on a computer. In the case of imports from Data Platform v5, applications identified there are first imported as installer evidence. As always, this installer evidence is tested against rules saved in the Application Recognition Library (ARL), and when matched, the appropriate application record is added to the `InstalledApplications` table (if it does not

already exist there), and is linked to the installer evidence. This application record is then visible in the **Installed Applications** page of the web interface for IT Asset Management. A record of installer evidence that cannot be matched by the ARL appears in the web interface on the **Unrecognized Evidence** page.


This process means there are three cases to consider, any of which may produce inventory gaps that can impact compliance calculations:

1. For applications recognized within Data Platform, and for which the normalized installer evidence has been mapped to an entry in the Application Recognition Library (ARL): there is no inventory gap as such, since at a minimum all these items are recognized within IT Asset Management; but in a few cases, some application variants (such as interface language, hotfix number, or platform) may be recognized by either tool but not the other.
2. For applications recognized within Data Platform, for which there is as yet no mapping to the ARL: the unrecognized evidence in IT Asset Management may mean that license consumption is under-counted because the applications have not been recognized within IT Asset Management, and therefore the licenses cannot measure consumption against the 'missing' applications.
3. For applications not recognized within Data Platform, but which could have been recognized by the ARL from other inventory sources: where Data Platform is your *only* data source, the unrecognized application may result in under-counting of license consumption.

There are two listings below. The first maps what is imported from Data Platform v5. The second lists values that IT Asset Management expects to load into the ImportedInstallerEvidence table, that are not available from Data Platform v5.

## Data mapping for imported installer evidence

The following listing matches the source data from Data Platform v5 with the equivalent column in the ImportedInstallerEvidence table. Other columns from the ImportedInstallerEvidence table cannot be populated from Data Platform v5, and so are omitted here (and shown below). For more details on the database tables and columns within IT Asset Management, please see *IT Asset Management Schema Reference*.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_SW_PROD) / CAT_SW_UUID	(ImportedInstallerEvidence) / ExternalInstallerID	The identifier used in Data Platform for the installer evidence.
(MATCH_HOST_SW_PROD) / CAT_PRODUCT_NAME or the Flexera ID for the application if available in ARL_ID	(ImportedInstallerEvidence) / DisplayName	The display name of the software recorded in Data Platform, which is imported only when there is no Flexera ID available. When the Flexera ID is returned instead, this allows matching in the ARL, which then supplies its standard display name for the application instead.
 <b>Tip:</b> A prefix of <i>arL :</i> is prepended to the <i>ARL_ID</i> on import.		



Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_SW_PROD) / CAT_VERSION or an empty string when the Flexera ID for the application is available	(InstallerEvidence) / Version	When the Flexera ID is known, the value is left empty for later supply from the ARL. For applications unknown in the ARL, the version value is imported from Data Platform.
(MATCH_HOST_SW_PROD) / CAT_MANUFACTURER or an empty string when the Flexera ID for the application is available	(ImportedInstallerEvidence) / Publisher	When the Flexera ID is known, the value is left empty for later supply from the ARL. For applications unknown in the ARL, the publisher's name is imported from Data Platform.
Hard-coded values: <ul style="list-style-type: none"> <li>FlexeraID when it is available in (CAT_FLEXERA_ARL) / ARL_ID</li> <li>BDNA when the Flexera ID for the application is not available.</li> </ul>	(ImportedInstallerEvidence) / Evidence	When the Flexera ID is known, the evidence type FlexeraID reflects this. Otherwise, when Data Platform is unaware of any matching entry in the ARL, it is identified as the evidence type.
(CAT_FLEXERA_ARL) / ARL_ID when the Flexera ID is available, or an empty string	(ImportedInstallerEvidence) / ProductCode	The Flexera ID (without the ar1:// prefix) is used when available. Otherwise an empty string.



**Tip:** The internal database  
value BDNA is replaced with  
Data Platform for presentation  
in the web interface.

## Data not imported from Data Platform v5

These columns in the ImportedInstallerEvidence table cannot be populated by inventory imports from Data Platform v5. These missing data points do not impact license compliance calculations. For further details about these columns, please see the schema reference.

- ComplianceConnectionID — *internally generated*
- AccessModeID (relevant only to file evidence, whereas this import is of installer evidence).

# Imported Computers (Inventory Devices)

IT Asset Management (Cloud)

The ImportedComputer table is a staging table used for input of inventory device records. From here, de-duplicated and normalized records are merged into the ComplianceComputer table.


There are two listings below. The first maps what is imported from Data Platform v5. The second lists values that IT Asset Management expects to load into the `ImportedComputer` table, that are not available from Data Platform v5, and the impact of these gaps on license reconciliation calculations where these rely on data from only this source.

## Data mapping for imported inventory devices

The following listing matches the source data from Data Platform v5 with the equivalent column in the `ImportedComputer` table. Other columns from the `ImportedComputer` table cannot be populated from Data Platform v5, and so are omitted here (and shown below). For more details on the database tables and columns within IT Asset Management, please see *IT Asset Management Schema Reference*.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
<p><i>Fabricated from:</i></p> <p>(MATCH_TASK_DET) / PROCESS_ID</p> <p>(MATCH_TASK_DET) / TASK_NAME <i>and the checksum for the same</i></p> <p>(MATCH_HOST_OS) / HOST_ID</p>	<p>(ImportedComputer) / ExternalID</p> <p>The format is of this form (shown with nonsense values):</p> <div>PID123   TheTask   654321 :Host789</div>	<p>An external ID is fabricated from the various details shown (truncated as required and joined with various separator characters). This is then mapped through the <code>ImportedStringMapping</code> table to convert string IDs to integer IDs.</p>
(MATCH_HOST_OS) / HOSTNAME	(ImportedComputer) / ComputerName	<p>The name of the computer.</p> <ul style="list-style-type: none"> <li>In Windows, this is the NetBIOS name of the local computer, as returned by <code>GetComputerName()</code></li> <li>For UNIX-like platforms, it is the host name of the machine, as returned by <code>gethostname(2)</code>.</li> </ul>
(MATCH_HOST_OS) / DOMAIN	(ImportedComputer) / Domain	<p>The domain reported by the computer.</p> <div>  <b>Note:</b> Data Platform exports a domain flat name, while the <code>ImportedComputer</code> table handles either a flat or DNS domain name, but the DNS domain name is preferred. Watch out for possible problems merging device records obtained from multiple inventory sources. </div>


Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_OS) / DISCOVERED_OS_NAME	(ImportedComputer) / OperatingSystem	The operating system of the computer.
(MATCH_HOST_OS) / DISCOVERED_OS_PATCH_LEVEL	(ImportedComputer) / ServicePack	The service pack installed for the operating system.
(MATCH_HOST_OS) / CALC_NUM_OF_PHYSICAL_CPUS	(ImportedComputer) / NumberOfProcessors	The number of processors in the computer.
(MATCH_HOST_OS) / DISC_CPU_MODEL	(ImportedComputer) / ProcessorType	The type of processor in the computer.
(MATCH_HOST_OS) / DISC_SPEED_MHZ	(ImportedComputer) / MaxClockSpeed	The maximum clock speed of the fastest processor in the computer.
(MATCH_HOST_OS) / CALC_NUM_OF_CORES	(ImportedComputer) / NumberOfCores	The number of cores in the computer.
<div>  <b>Note:</b> Currently Data Platform gets an incorrect result from virtual machines (defect repair estimate 1Q2018). This affects the consumption calculations for all license types that use cores as a metric (most of these assume 1 core per processor when the number of cores is not available): <ul style="list-style-type: none"> <li>• Device (Core-Limited)</li> <li>• Core Points</li> <li>• IBM PVU</li> <li>• Microsoft Server Core</li> <li>• Microsoft Server/ Management Core</li> <li>• Oracle Processor.</li> </ul> </div>		
(DISC_ALL_OS) / TOTALMEMORY	(ImportedComputer) / TotalMemory	The total RAM in the computer, in bytes.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes														
(MATCH_HOST_HW_PROD_MODEL) / CAT_TAXONOMY_CATEGORY2	(ImportedComputer) / ChassisType	<p>The type of case of the computer. Some license types use this information to optimize the licensing position, particularly with desktop and laptop computers. Types used in Data Platform are mapped to types in IT Asset Management as follows:</p> <table><tr><th>Data Platform</th><th>FNMS</th></tr><tr><td>Empty</td><td>Unknown</td></tr><tr><td>Handhelds</td><td>Tablet</td></tr><tr><td>Notebooks</td><td>Laptop</td></tr><tr><td>Mainframes</td><td>Main System Chassis</td></tr><tr><td>Servers</td><td>Rack Mount Chassis</td></tr><tr><td>All others</td><td>Desktop</td></tr></table>	Data Platform	FNMS	Empty	Unknown	Handhelds	Tablet	Notebooks	Laptop	Mainframes	Main System Chassis	Servers	Rack Mount Chassis	All others	Desktop
Data Platform	FNMS															
Empty	Unknown															
Handhelds	Tablet															
Notebooks	Laptop															
Mainframes	Main System Chassis															
Servers	Rack Mount Chassis															
All others	Desktop															
(DISC_HDDS) / The count of entries for this device	(ImportedComputer) / NumberOfHardDrives	<p>The number of hard drives in the computer.</p> <div> <b>Note:</b> Collected into Data Platform only by a legacy extractor or custom extractor. IT Asset Management imports it where it is available.</div>														


Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(DISC_ALL_OS) / LOCALFILESYSTEMSPACETOTAL	(ImportedComputer) / TotalDiskSpace	<p>The total size of all hard drives in the computer. The results differ for inventory collected from Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) directly by IT Asset Management, or indirectly through Data Platform v5:</p> <ul style="list-style-type: none"> <li>The FlexNet connection imports the <i>physical</i> disk space (DISK_DATA)</li> <li>Data Platform imports the <i>logical</i> hard drive space (v_GS_Logical_DISK).</li> </ul>
(DISC_NICs) / The count of entries for this device	(ImportedComputer) / NumberOfNetworkCards	The number of network cards in the computer.
(DISC_MONITORS) / The count of entries for this device	(ImportedComputer) / NumberOfDisplayAdapters	<p>The number of graphics cards in the computer.</p> <p> <b>Note:</b> Collected into Data Platform only by a legacy extractor or custom extractor. IT Asset Management imports it where it is available.</p>
(DISC_NICs) / IP_ADDRESS	(ImportedComputer) / IPAddress	The IP address of the computer.
(DISC_NICs) / MACADDRESS	(ImportedComputer) / MACAddress	The MAC address of the computer.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_HW_PROD_MODEL) / CAT_MANUFACTURER <i>Failover to</i> (DISC_COMPUTERSYSTEM) / MANUFACTURER	(ImportedComputer) / Manufacturer	<p>The manufacturer of the computer hardware. Some examples include:</p> <ul style="list-style-type: none"> <li>On Windows, the SMBios manufacturer (the WMI Manufacturer property of the 'Win32_ComputerSystem' class).</li> <li>On Linux, 'Manufacturer' in the 'System Information' section resulting from the 'dmidecode' command. Sample command: 'dmidecode -s system-manufacturer'</li> <li>On Solaris x86, as for Linux, with failovers first to 'sysinfo SI_HW_PROVIDER' and then to 'ModelNo'.</li> <li>On Solaris SPARC, the 'sysinfo SI_HW_PROVIDER'. Typically this value is 'Sun_Microsystems' or, more recently, 'Oracle Corporation'. Failover to the 'ModelNo'.</li> <li>On HP-UX, the string literal 'HP'.</li> <li>On AIX, the 'modelName' system attribute preceding the comma character. For example, if the 'modelName' system attribute is 'IBM,8202-E4B', then use 'IBM'. This value is typically 'IBM'.</li> </ul>

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_HW_PROD_MODEL) / CAT_HARDWARE_PRODUCT and CAT_HW_MODEL, joined with a space Failover to (DISC_COMPUTERSYSTEM) / MODEL	(ImportedComputer) / ModelNo	The model of the computer hardware or the virtual machine. This value is defined for the context of the current execution environment, rather than the physical server that may be hosting a virtual machine or partition. For examples across operating systems, see the schema documentation.
(MATCH_HOST_OS) / SERIALNUMBER Failover to (DISC_ALL_OS) / SERIALNUMBER	(ImportedComputer) / SerialNo	The hardware serial number of the computer. The goal of this value is to be tied to the physical hardware, partition or virtual machine and to be as unique as possible across all computers in the organization. This is due to its use in tracking computers, particularly after an operating system rebuild. This value is also used to socialize computer inventory from different inventory sources, and is used to map virtual machine guest operating system inventory to the VM host on which the virtual machine is running.
(DISC_HOSTS_USERLOGON) / DOMAIN_USERNAME	(ImportedComputer) / LastLoggedOnUser	The DOMAIN\SAMAccountName of the user last logged onto the computer. Data Platform uses the flat domain name.
(DISC_ALL_OS) / ScanDate	(ImportedComputer) / InventoryDate	The date the computer last had inventory reported.
Hard-coded value Data Platform	(ImportedComputer) / InventoryAgent	The name of the person or tool that performed the last inventory.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_CPU) / DISC_NUM_OF_SOCKETS	(ImportedComputer) / NumberOfSockets	<p>The number of sockets in the computer.</p> <hr/> <p> <b>Tip:</b> Data Platform supplies this value from Technopedia. Other inventory sources supported by IT Asset Management cannot gather this value from inventory; and in general, for compliance calculations where the number of sockets is not available in inventory, the value is approximated by using the number of processors (which is satisfactory when there are no empty sockets in the inventory device). In devices that have any empty sockets, non-blank information from Technopedia is superior. Notice that certain Oracle license types (such as Oracle Database Standard Edition) have a restriction on the number of sockets on the host server.</p>



Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_CPU) / CALC_NUM_OF_LOGICAL_CPUS	(ImportedComputer) / NumberOfLogicalProcessors	The number of logical processors in the computer.
<div>  <b>Note:</b> Currently Data Platform gets an incorrect result from virtual machines (defect repair estimate 1Q2018). This affects the consumption calculations for all license types that use processors as a metric: <ul style="list-style-type: none"> <li>• Device (Processor-Limited)</li> <li>• Microsoft Server Processor</li> <li>• Processor</li> <li>• Processor Points.</li> </ul> </div>		

## Data not imported from Data Platform v5

These columns in the ImportedComputer table cannot be populated by inventory imports from Data Platform v5. The majority of these missing data points do not impact license compliance calculations (these are listed here without comment). Those that do have an impact are noted below. For further details about these columns, please see the schema reference.

- HardwareInventoryDate
- ServicesInventoryDate
- ComplianceConnectionID — *internally generated*
- ComplianceComputerID — *internally generated*
- ComplianceDomainID — *internally generated*
- IncompleteRecord — This is always set to zero for imports from Data Platform (that is, the imported record is regarded as 'sufficient' for its normal use in license compliance calculations, subject to the known limitations described in these pages).
- PartialNumberOfProcessors — Absence of this value prevents subcapacity license consumption calculations for IBM PVU and Oracle Processor licenses.
- UntrustedSerialNo
- FullDetailsFromExternalID
- FullDetailsFromComplianceConnectionID
- ComplianceComputerTypeID — This is the foreign key to the ComplianceComputerType table, identifying

whether the device is a VM, a VM host, and so on. *Compliance impact:* Prevents (unrecognized) VDI templates consuming from a license. Prevents (unrecognized) VMs from observing license rules that require (or disallow) consumption by VMs.

- `ILMTAgentID` — For imports from ILMT, identifies the device as a candidate for subcapacity calculations on any related IBM PVU license(s). *Compliance impact:* Device consumption may be at full capacity, and calculated only when a full inventory import and compliance calculation is run (typically, once daily).
- `FNMPComputerUID`
- `HostIdentifyingNumber` — Useful for matching records from different sources, but no direct effect on compliance calculations.
- `HostType` — Useful for matching records from different sources, but no direct effect on compliance calculations.
- `IsRemoteACLDevice`
- `IsDuplicate`
- `LegacySerialNo`
- `UUID`
- `IMEI`
- `PhoneNumber` (for mobile devices)
- `EmailAddress` (for mobile devices)
- `CalculatedUser` — Typically, calculated by IT Asset Management as the most frequent user in the last ten log-on records. Data Platform v5 only reports this last logged on user name (`LastLoggedOnUser`); but this is sufficient.
- `LastSuccessfulInventoryDate`
- `MDScheduleGeneratedDate`
- `MDScheduleContainsPVUScan`
- `FirmwareSerialNumber`
- `MachineID`
- `IgnoredDueToLicense`.

## Imported VMs and Hosts

### IT Asset Management (Cloud)

The `ImportedVirtualMachine` table is a staging table used for input of virtual machine records, including their host server. From here, de-duplicated and normalized records of the VM properties are merged into the `VirtualMachine` table; and if not already present, records for both the VM and the host are added to the `ComplianceComputer` table.



**Tip:** This data is in addition to the inventory imported for the virtual machine (and, separately, its host) and imported

as part of the inventory device records. For details, see [Imported Computers \(Inventory Devices\)](#).

There are two listings below. The first maps what is imported from Data Platform v5. The second lists values that IT Asset Management expects to load into the ImportedVirtualMachine table, that are not available from Data Platform v5, and the impact of these gaps on license reconciliation calculations where these rely on data from only this source.

## Data mapping for imported virtualization records

The following listing matches the source data from Data Platform v5 with the equivalent column in the ImportedVirtualMachine table. Other columns from the ImportedVirtualMachine table cannot be populated from Data Platform v5, and so are omitted here (and shown below). For more details on the database tables and columns within IT Asset Management, please see [IT Asset Management Schema Reference](#).

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
Fabricated from: (MATCH_TASK_DET) / PROCESS_ID (MATCH_TASK_DET) / TASK_NAME and the checksum for the same (DISC_VIRTUAL_HOSTGUEST) / HOST_ID	(ImportedVirtualMachine) / HostComputerID The format is of this form (shown with nonsense values): <div>PID123 TheTask 654321:Host789</div>	A computer ID for the host is fabricated from the various details shown (truncated as required and joined with various separator characters). This is then mapped through the ImportedStringMapping table to convert string IDs to integer IDs.
(MATCH_HOST_HW_PROD_MODEL) / CAT_MANUFACTURER	(ImportedVirtualMachine) / VirtualMachineType	The type of virtual machine. Values from Data Platform are transformed as follows: <ul style="list-style-type: none"> <li>• Microsoft becomes VMType.HyperV</li> <li>• VMware becomes VMType.VMware</li> <li>• Any other value becomes VMType.Unknown.</li> </ul>
(DISC_ALL_OS) / OSCOMPUTERNAME	(ImportedVirtualMachine) / ComputerName	As Data Platform has only one name for the VM, the previous value is replicated here (again).
(MATCH_HOST_HW_PROD_MODEL) / CAT_HW_MODEL Failover to (MATCH_HOST_HW_PROD_MODEL) / DISCOVERED_MODEL	(ImportedVirtualMachine) / ModelNo	The model number reported for the VM.
(MATCH_HOST_HW_PROD_MODEL) / CAT_MANUFACTURER	(ImportedVirtualMachine) / Manufacturer	The manufacturer visible to the guest OS running on the VM.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
Fixed value Data Platform	(ImportedVirtualMachine) / InventoryAgent	The tool that provided this inventory record.
(DISC_ALL_OS) / OPERATINGSYSTEM_TYPE_LABEL	(ImportedVirtualMachine) / GuestFullName	The operating system configured for the guest.
Fabricated from: (MATCH_TASK_DET) / PROCESS_ID (MATCH_TASK_DET) / TASK_NAME and the checksum for the same (DISC_VIRTUAL_HOSTGUEST) / Guest_ID	(ImportedVirtualMachine) / VMComputerID The format is of this form (shown with nonsense values): <div>PID123   TheTask   654321:Host789</div>	A computer ID for the guest is fabricated from the various details shown (truncated as required and joined with various separator characters). This is then mapped through the ImportedStringMapping table to convert string IDs to integer IDs.

## Data not imported from Data Platform v5

The following columns in the ImportedVirtualMachine table cannot be populated by inventory imports from Data Platform v5. Notice that several of them (such as NumberOfProcessors, NumberOfHardDrives, and IPAddress) have matching data points imported as part of the inventory device import for the same guest machine. While several of these missing data points do not impact license compliance calculations, the following gaps are significant:

- When the original input to Data Platform was from BMC Discovery (ADDM), information about virtual machines (and hosts) is *not* available. In contrast, when the data source for Data Platform was Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), the identification of hosts and guests is available for supported platforms such as Hyper-V.



**Tip:** This is the inverse of the result obtained when IT Asset Management takes inventory from these sources directly (rather than through Data Platform). In the this unmediated case, inventory from Microsoft Endpoint Configuration Manager does not include the host/guest, and the inventory from ADDM does include the host/guest relationship. Of course, if you have multiple inventory sources collecting from a common target device, the results are merged within IT Asset Management to provide the most complete record possible.

- When data is available (from inventory sources such as Microsoft Endpoint Configuration Manager into Data Platform), the import from Data Platform to IT Asset Management is missing information about resource pools, vMotion, and partitions. This means that, when Data Platform v5 is the only inventory source for these machines, compliance calculations are impossible for consumption from some licenses such as IBM PVU, Microsoft, or Oracle server licenses.

For further details about these columns that are missing data, please see the schema reference.

- VMName
- VCObjectID
- FriendlyName
- UUID

- TotalMemory
- PoolName
- CPUUsage
- MemoryUsage
- MaxNumberOfLogicalProcessors
- VMEnabledStateID
- NumberOfProcessors
- ProcessorType
- NumberOfNetworkCards
- ComplianceConnectionID
- VMLocation
- PoolType
- ZoneResourceManagementMethodType
- AffinityEnabled
- CPUAffinity
- CoreAffinity
- PartitionID
- PartitionNumber
- FullComputerName
- IPAddress.

## Imported Installation Records

### IT Asset Management (Cloud)

The `ImportedInstalledInstallerEvidence` table is a staging table used to collect records of which installation evidence has been found on what inventory devices. From here, de-duplicated and normalized records are merged into the `InstalledInstallerEvidence` table.

`InstalledInstallerEvidence` is a simple table that mainly links installer evidence records with inventory device records. In the case of imports from Data Platform v5, only the two external IDs are needed, with other values supplied by IT Asset Management as and when required. The first listing below maps the two columns from Data Platform source to IT Asset Management destination. Below is a list of the remaining columns in the `ImportedInstalledInstallerEvidence` table that are not populated by this import (and do not need to be).

## Data mapping for imported installed installer evidence

The following listing matches the source data from Data Platform v5 with the equivalent column in the ImportedInstalledInstallerEvidence table. For more details on the database tables and columns within IT Asset Management, please see *IT Asset Management Schema Reference*.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_HOST_SW_PROD) / CAT_SW_UUID	(ImportedInstalledInstallerEvidence) / ExternalInstallerEvidenceID	The same value as the (ImportedInstallerEvidence) / ExternalInstallerID.
Fabricated from: (MATCH_TASK_DET) / PROCESS_ID (MATCH_TASK_DET) / TASK_NAME and the checksum for the same (MATCH_HOST_SW_PROD) / HOST_ID	(ImportedInstalledInstallerEvidence) / ExternalID The format is of this form (shown with nonsense values): PID123   TheTask   654321 : Host789	An external ID for the inventory device is fabricated from the various details shown (truncated as required and joined with various separator characters). This is then mapped through the ImportedStringMapping table to convert string IDs to integer IDs. The same value as the (ImportedComputer) / ExternalID.
(MATCH_HOST_SW_PROD) / DISCOVERED_INSTALLDATE	(ImportedInstalledInstallerEvidence) / InstallDate	The installation date recorded for the installer evidence.

## Data not imported from Data Platform v5

These columns in the ImportedInstalledInstallerEvidence table cannot be populated by inventory imports from Data Platform v5. These missing data points do not impact license compliance calculations. For further details about these columns, please see the schema reference.

- ComplianceConnectionID — *internally generated*
- ExternalInstanceID
- DiscoveryDate (will be set to the date of first import from this connection).

# Imported Users

IT Asset Management (Cloud)

The ImportedUser table is a staging table used for input of records about users. From here, de-duplicated and normalized records are merged into the ComplianceUser table.

ComplianceUser stores information about end-users in the enterprise, including contact details, login details, and inventory source details (if applicable). End-users are the people using computers within your enterprise (as distinct from operators, who are people permitted to use IT Asset Management).

There are two listings below. The first maps what is imported from Data Platform v5. The second lists values that IT Asset Management expects to load into the ImportedUser table, that are not available from Data Platform v5.

## Data mapping for imported users

The following listing matches the source data from Data Platform v5 with the equivalent column in the ImportedUser table. Other columns from the ImportedUser table cannot be populated from Data Platform v5, and so are omitted here (and shown below). For more details on the database tables and columns within IT Asset Management, please see *IT Asset Management Schema Reference*.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(DISC_HOSTS_USERLOGON) / USERDOMAIN and USERLOGON combined	(ImportedUser) / ExternalID	Mapped through the ImportedStringMapping table to convert string IDs to integer IDs.
(DISC_HOSTS_USERLOGON) / USERDOMAIN	(ImportedUser) / Domain	The domain of the end-user. Data Platform exports the flat domain name.
(DISC_HOSTS_USERLOGON) / USERLOGON	(ImportedUser) / SAMAccountName	The login name (SAM account name) of the end-user.
<i>Hard-coded value</i> Data Platform	(ImportedUser) / InventoryAgent	The name of the person or tool that performed the last inventory.

## Data not imported from Data Platform v5

These columns in the ImportedUser table cannot be populated by inventory imports from Data Platform v5. These missing data points do not impact license compliance calculations. For further details about these columns, please see the schema reference.

- ComplianceConnectionID — *internally generated*
- UserName
- ComplianceDomainID
- LocationID
- FirstName
- LastName
- Email
- EmployeeNumber
- CostCenter (as reported in SAP)
- ComplianceUserID — *internally generated*
- ComplianceDomainID — *internally generated*

- IsBlacklisted — *internally generated*
- MapUsingEmailAddress.

## Imported Software Usage

### IT Asset Management (Cloud)

The ImportedInstalledInstallerEvidenceUsage table is a staging table used to collect records of actual use of applications related to installation evidence found on inventory devices. From here, de-duplicated and normalized records are merged into the InstalledSoftwareUsageData table.

There are two listings below. The first maps what is imported from Data Platform v5. The second lists values that IT Asset Management expects to load into the ImportedInstalledInstallerEvidenceUsage table, that are not available from Data Platform v5, and the impact of these gaps on license reconciliation calculations where these rely on data from only this source.

### Data mapping for imported usage related to installer evidence

The following listing matches the source data from Data Platform v5 with the equivalent column in the ImportedInstalledInstallerEvidenceUsage table. Other columns from the ImportedInstalledInstallerEvidenceUsage table cannot be populated from Data Platform v5, and so are omitted here (and shown below). For more details on the database tables and columns within IT Asset Management, please see *IT Asset Management Schema Reference*.

Data Platform (Table)/Column	FNMS (Table)/Column	Notes
(MATCH_SOFTWARE_METERING) / USAGE_YEAR and USAGE_MONTH_N Failover to (MATCH_SW_RECENTLY_USED_APPS) / LASTUSEDDATE	(ImportedInstalledInstaller EvidenceUsage) / StartDate	The start date is set to a string formatted similarly to ISO 8601 (except for the slash separator: 2019/04/01) by conjoining the year and month-number from Data Platform, and adding an assumed day for the first of the month. Only if there is no value available by this path, we use the date available from the alternate table shown.



Data Platform (Table)/Column	FNMS (Table)/Column	Notes
<p><i>Fabricated from:</i></p> <p>(MATCH_TASK_DET) / PROCESS_ID</p> <p>(MATCH_TASK_DET) / TASK_NAME</p> <p><i>and the checksum for the same</i></p> <p>(MATCH_SOFTWARE_METERING) / HOST_ID</p>	<p>(ImportedInstalledInstallerEvidenceUsage) / ExternalID</p> <p>The format is of this form (shown with nonsense values):</p> <p>PID123 TheTask 654321:Host789</p>	<p>An external ID is fabricated from the various details shown (truncated as required and joined with various separator characters). This is then mapped through the ImportedStringMapping table to convert string IDs to integer IDs.</p> <hr/> <p> <b>Tip:</b> If the <i>MATCH_SOFTWARE_METERING</i> table cannot provide a host ID, it is taken from <i>MATCH_SW_RECENTLY_USED_APPS</i>.</p>
(MATCH_SOFTWARE_METERING) / CAT_SW_UUID	(ImportedInstalledInstallerEvidenceUsage) / ExternalInstallerID	<p>The identifier used in Data Platform for the installer evidence.</p> <hr/> <p> <b>Tip:</b> If the <i>MATCH_SOFTWARE_METERING</i> table cannot provide an external ID, it is taken from <i>MATCH_SW_RECENTLY_USED_APPS</i>.</p>
(MATCH_SOFTWARE_METERING) / USAGECOUNT <i>plus</i> TSUSAGECOUNT	(ImportedInstalledInstallerEvidenceUsage) / NumberOfSessions	If the number of sessions in which the evidence was in use cannot be calculated by this method, the default value 1 is inserted.
<p>(MATCH_SOFTWARE_METERING) / LASTUSEDDATE</p> <p>Failover to</p> <p>(MATCH_SW_RECENTLY_USED_APPS) / LASTUSEDDATE</p>	(ImportedInstalledInstallerEvidenceUsage) / LastUsedDate	The last known date when usage of the application is recorded (in one table or the other).
(MATCH_SW_RECENTLY_USED_APPS) / LASTUSERDOMAIN, <i>then a backslash</i> \, <i>then</i> LASTUSERLOGON	(ImportedInstalledInstallerEvidenceUsage) / ExternalUserID	The end-user identified in the source as using the software. The identifier is fabricated by conjoining the flat domain name and logon name (separated by a backslash).

## Data not imported from Data Platform v5

These columns in the ImportedInstalledInstallerEvidenceUsage table cannot be populated by inventory imports from Data Platform v5. These missing data points do not impact license compliance calculations. For further details about these columns, please see the schema reference.

- ComplianceConnectionID — *internally generated*
- ExternalInstanceID (used only for Oracle).

# VI

## Flexera SaaS Manager Adapter

IT Asset Management (Cloud)

This part introduces the integration between IT Asset Management and Flexera One SaaS Management, a cloud-based product designed to help you manage your licensing costs for SaaS-based software.

The method of integration is a simple connector; and since both products are cloud implementations, configuration of the connector is as easy as selecting a check box.

This connector supersedes older connectors separately available for Salesforce and Microsoft Office 365. This part includes detailed guidelines for retiring the old, and doing some data clean-up to normalize your records.

## 1

# The Flexera SaaS Manager Connector

## IT Asset Management (Cloud)

Assuming that you have licensed both products, IT Asset Management allows you to make a connection to Flexera One SaaS Management so that you can import data, and unify all your licensing information in the one place (sometimes called having "a single pane of glass" to see all your licensing data).

The data flow with this connector is strictly one way (from Flexera One SaaS Management into IT Asset Management), and any changes you make locally after the import are not reflected back into the source data in Flexera One SaaS Management. So data maintenance remains separate, and happens in the product where you created the records. For example, if you renegotiate your SaaS license for a product to increase your entitlements, record the new number in Flexera One SaaS Management, because that is the source of truth for this license. The information for display in IT Asset Management is automatically updated in the nightly full inventory import and license reconciliation calculations.

To make it easier to switch contexts, the properties for every license imported from Flexera One SaaS Management display a link to that product from within IT Asset Management (direct to the matching license record, if you are already logged in). There is also a general link you can display in the top bar of IT Asset Management to make switching back easy, even without opening any license properties.

For more information about setting up the Flexera One SaaS Management connector, and enabling the display of the top bar link, see the summary in [Configuring SaaS Manager Connector](#).

Even though the data flow is strictly one way, there are some changes you may choose to make within your IT Asset Management record that are preserved during any update through the nightly imports:

- You may still link purchases or contracts to your imported licenses. However, it is important to understand that these are now for information only, and have no impact on the available license entitlements or the nightly license reconciliation. Your entitlements are part of the one-directional import, and consumption is counted exclusively against the entitlements recorded in Flexera One SaaS Management (and imported from there).
- You may still choose to record details like ownership responsibility or group assignments against the imported licenses, and these additions are preserved.



**Note:** Any changes to SaaS license names need to be done in SaaS Management.

If you have previously been using other SaaS-related connectors, once you have switched over you should consider retiring those connectors, since their data overlaps with the imports from Flexera One SaaS Management. To avoid possible confusion between licenses created by various connectors, you might also consider some data clean-up to

normalize your environment. For more information, see [Deprecating Other Connectors](#).

This chapter concludes with some technical details about the data imported from Flexera One SaaS Management, where it is staged in the IT Asset Management database, and where you can look in the web interface to review the imported data (see [Imported Data Mapping](#)).

## 2

# Configuring SaaS Manager Connector

IT Asset Management (Cloud)

## Prerequisites

With the cloud implementation of IT Asset Management, integration with Flexera One SaaS Management is simple. The general prerequisite is that you have licensed Flexera One SaaS Management.

For a license to show in IT Asset Management, you also need to make sure that the following fields of the license have values populated in the Flexera One SaaS Management. For more details about these fields, see [Licenses Tab](#) in the Flexera One Help.

- **# of Items Allowed**
- **Amount**
- **Payment Frequency**
- **Effective Date**
- **Ending Date**

## Set up

Go to the IT Asset Management Settings **General** page (**Administration > IT Asset Management Settings > General**), select the **Integrations** tab, expand the **SaaS Manager** section, and select the **Enable SaaS Manager integration** check box. There are more details in the online help for that page.



**Tip:** In the FlexNet Beacon software (from version 14.3.0), there is a separate connector to Flexera One SaaS Management in the group of inventory connections configured with PowerShell. You should ignore that connector. That is available for other customers using on-premises implementations of IT Asset Management; but since you are using the Flexera cloud implementation, integration for you is simpler and quicker, as described above.

## Scheduling

No scheduling is required. The import from Flexera One SaaS Management happens automatically as part of the overnight full inventory import and license consumption calculations. This means that from the moment you enable the

connector, data from Flexera One SaaS Management appears in your system the next day.

## Validation

The day after configuring your connector, validate operations by:

- Identifying a licensed application listed in Flexera One SaaS Management.
- Finding the same application listed in IT Asset Management, and validate that the appropriate data is recorded. For example, there is a SaaS User license attached (except for Microsoft Office 365, which should have a Named User license attached).

For more information about validation, data rationalization, and connector clean-up, see [Deprecating Other Connectors](#).

## 3

# Deprecating Other Connectors

## IT Asset Management (Cloud)

When you configure your connection to Flexera One SaaS Management, you may need to delete one or more other connectors that have overlapping functionality:

Connector name	Set up instructions (to help identify the connector)
Microsoft Office 365 (deprecated)	<i>FlexNet Manager Suite Help &gt; Inventory Beacons &gt; Inventory Systems Page &gt; Connecting to External Inventory Systems &gt; Managing PowerShell Connections &gt; Creating Connections to Microsoft Office 365 &gt; Creating Connections using the Microsoft Office 365 (Deprecated) Connector</i>
Microsoft 365	<i>FlexNet Manager Suite Help &gt; Inventory Beacons &gt; Inventory Systems Page &gt; Connecting to External Inventory Systems &gt; Managing PowerShell Connections &gt; Creating Connections to Microsoft Office 365 &gt; Using FlexNet Manager Suite's Multi-Tenant App to Connect to Microsoft 365</i>
Salesforce	<i>FlexNet Manager Suite Help &gt; Inventory Beacons &gt; Inventory Systems Page &gt; Connecting to External Inventory Systems &gt; Managing PowerShell Connections &gt; Managing Connections to Salesforce.com</i>

If you have any of the above connectors in use, the licenses created from those imports are automatically marked as **Retired** after overlapping licenses are imported from Flexera One SaaS Management, and the first full inventory import and license reconciliation calculations are completed (typically overnight). The retired licenses no longer contribute to the license consumption calculations, and are replaced in that role by the replacement licenses imported from Flexera One SaaS Management. Therefore, using the 'old' connectors to import updates to the **Retired** licenses is now redundant, and you might consider removing the old connectors and cleaning up their associated retired licenses.

In some cases, there may also be 'surplus' application records. This is particularly true if you have been using the **Salesforce** connector, which returns Salesforce 'components', each of which has been created as a separate application record in IT Asset Management. In contrast, Flexera One SaaS Management returns just the one "Salesforce" application, so that you may also wish to rationalize your application records, as described below.



**Caution:** Do not delete one of the above connections too early – specifically, do not delete an existing connection until data flow is established from your new Flexera One SaaS Management connection. If you leave your system without



any connection for certain inventory items, and the nightly inventory import and license consumption calculations happen when the relevant connection is still missing, **all inventory imported from the missing connection is removed** from IT Asset Management. This is because it is assumed that there has been a clean-up in your computing estate, and records are deleted to match the incoming data.

Use the following process to avoid both data loss (from deleting a connection too early) and data redundancy (from leaving overlapping connectors and updating retired licenses).



#### **To manage switching between connectors:**

1. Identify one or more licenses created from imports through the connector(s) listed above.

These are licenses for the SaaS products you are tracking, and are perhaps best searched for by name in the **All Licenses** page.

2. Set up your connector to Flexera One SaaS Management.

This is as simple as turning on the appropriate check box, as described in [Configuring SaaS Manager Connector](#).

3. Wait until tomorrow.

Once the connection is enabled, information is imported into the compliance database of IT Asset Management during the next full inventory import and license consumption calculations. By default, these are scheduled nightly.

4. Validate that data has been imported from Flexera One SaaS Management.

- a. It's convenient to filter the list of **All Licenses** by setting the **License type** column equal to SaaS User (or Named user if instead you are investigating data for Office 365).
- b. Look for pairs of licenses that may be similarly named.



**Tip:** Licenses imported from Flexera One SaaS Management initially reflect exactly the license names you created in Flexera One SaaS Management. Licenses imported with older connectors may have different naming conventions. Two techniques may be helpful:

- Sorting by the **Name** column
  - Adding the **Status** column from the column chooser (note that this is a separate column different than the **Compliance status** column), where the licenses imported from old connectors now display *Retired*.
- c. Right-click each of the license names, and choose your browser's options to have the two sets of license properties open for comparison, such as in two browser windows. In both cases, select the **Identification** tab of the license properties.

Here are the key differences between the two licenses:

Old license from deprecated connector	New license from Flexera One SaaS Management
<p><b>Status</b> field displays <b>Retired</b>.</p> <p>This value is set while processing the import from your new Flexera One SaaS Management connector, and it validates that the process has run.</p>	<p><b>Status</b> field displays <b>Active</b>, and certainly <i>not</i> retired.</p>
<p>Nothing extra in the title area of the license properties.</p>	<p>The title area of the license properties includes a hyperlink (just above the tabbed area) to navigate to Flexera One SaaS Management and review the same license there.</p> <p>The presence of this link validates that this license was created by the import from Flexera One SaaS Management.</p>

These two checks validate that the overnight processing of the import from has completed as expected.

- You may choose to edit the name of your new license, for example by adding "from FSM" to the license name, to eliminate ambiguity and reduce possible confusion.

The license name is not used for matching in future imports, and you may adjust the name as you require.

Check for any additional data management that you may want to do, as follows.

- In the properties for the old, retired license, check for attachments or other data that you may wish to transfer to the new license.

For example:

- Has anyone manually added **Notes** on the **Identification** tab? You may want to copy any important content to the new license.
- Are any purchases shown on the **Purchases** tab? You may want to **Remove** these from the old license, and link them instead to the new license.



**Remember:** Purchases linked to a license created from the Flexera One SaaS Management connector are purely informational, and do not affect the available entitlements or the nightly license compliance calculations. Your actual entitlements are recorded in and imported from Flexera One SaaS Management, and displayed in the **Compliance** tab, either as a distinct quantity in the **Extra entitlements** field, or as an **Entitlements are: Unlimited** setting.

- Similarly for the **Contracts** tab, you may want to swap any linked contracts to the new license. (As with purchases, linked contracts are informational only on your new license.)
  - Check other tabs for information to copy across (any regular charges on the **Financial** tab, any managerial or other assignments on the **Ownership** tab, or less commonly for SaaS licenses, any **Group assignment** or **Restrictions**).
- In both sets of license properties, switch to the **Applications** tab, and review the applications linked to each license. Decide on your corporate approach to the old application records – do you wish to remove old records that are effectively replaced by new application records created through the Flexera One SaaS Management connector?

If you plan to remove the retired licenses from deprecated connector(s), you might also delete the application records that are attached *only* to those licenses (check the **Licenses** tab in the application properties to make sure that the application is linked to just the one license). Such records will be redundant clutter when their related licenses are removed.

8. Depending on your corporate data retention policies, you may wish to do either of the following:
  - *Clean up*: Close the browser tab displaying the properties of the old, retired license; and then delete the license in the **All Licenses** page. (Using this option to delete the old licenses, it is especially important to follow through with the later step below to remove the old connector – because if this old connector runs again, it will recreate records for any licenses it finds.)
  - *Archive*: Edit the name of the old, retired license to make it easier to identify in lists, reducing future confusion of old with new (for example, insert "Retired - " at the start of the license name).
9. Repeat your chosen process for any/all other licenses that have been Retired following the import from Flexera One SaaS Management.



**Tip:** You can conveniently identify these in the **All Licenses** page by adding the **Status** column from the column chooser (note that this is a separate column different than the **Compliance status** column), and then filtering that column to show only the Retired value. Once you find a retired license, clear the filter and look for its active replacement, as described in the earlier steps of this process.

10. On the inventory beacon where the old connector(s) were configured, log in as administrator, run FlexNet Beacon, navigate to the **Inventory systems** page, and delete *only* the old connector(s).

You have now cleaned up licenses created through the old connectors, validated that your replacement licenses have been created by the Flexera One SaaS Management import, updated these with any required additional data, cleaned up application records as required, and removed the old connectors to remove overlapping data imports.

## 4

# Imported Data Mapping

## IT Asset Management (Cloud)

Data imported through the Flexera One SaaS Management connector is matched to records previously imported through the same connector into IT Asset Management (where available), or new records are automatically created.



**Important:** Imports through the Flexera One SaaS Management connector do not update records created from other connectors such as the Salesforce connector, the Microsoft 365 connector, or the Microsoft Office 365 (deprecated) connector. For information about switching connectors and data clean-up, see [Deprecating Other Connectors](#).

The data flow through this connector is one way – some imported data points (such as the available entitlements) are not editable within IT Asset Management, and must be edited within Flexera One SaaS Management and reimported in the next nightly full inventory import. A link is provided on the property sheet of relevant licenses to help you switch contexts, and you can also activate a button in the navigation bar for switching to Flexera One SaaS Management (see the **Integrations** tab of the IT Asset Management Settings **General** page (**Administration > IT Asset Management Settings > General**)). Similarly, deleting a matched data record (such as a license) within IT Asset Management does not automatically delete the matching data in Flexera One SaaS Management, where the record must also be deleted manually (otherwise the next inventory import simply recreates the matching record to align with the source data).



**Tip:** As well, deleting a license within Flexera One SaaS Management does not automatically delete the matching license in IT Asset Management, even after the next import and license compliance calculation. Similarly, deleting an imported application record does not delete the equivalent managed application within Flexera One SaaS Management. To delete a license or managed application originating in Flexera One SaaS Management and mirrored within IT Asset Management, you must access each of the records separately in the two products, and manually delete each one.

Almost all SaaS products are handled identically in IT Asset Management, and represented with SaaS User licenses. The one exception is Microsoft Office 365: because a license entitlement here may authorize use either as a SaaS application or as a locally-installed application on a device, these are handled separately and represented as Named User licenses. As a result, the data flows/storage is quite distinct for this case.



**Tip:** Flexera One SaaS Management allows you to create one Office 365 license that is linked to multiple plans, such as Office 365 (E1) and Office 365 (E4). Because each plan brings with it changed license entitlements, within IT Asset Management each plan for Office 365 is automatically given a separate license. This may mean that you see two (or

more) licenses in IT Asset Management that link back to the one source license in Flexera One SaaS Management.

The following tables show properties found in the user interface of Flexera One SaaS Management, and track their initial staging within the compliance database of IT Asset Management (with the database table name shown in the header of each list), and their common property name in the web interface. Fields in the target table are omitted if no related data is imported.



**Note:** The following *ImportedUser* table is populated only for Microsoft Office 365 users. These records are linked to existing user properties (*ComplianceUser* records) when these already exist; or else new user records are created.

**Table 2:** User data mapping for Office 365 (*ImportedUser* table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
<b>First name</b>	FirstName (up to 128 characters)	User properties > <b>General</b> tab > <b>First name</b>
<b>Last name</b>	LastName (up to 128 characters)	User properties > <b>General</b> tab > <b>Last name</b>
<b>Email</b>	Email (up to 200 characters)	User properties > <b>Details</b> tab > <b>Email</b>
Hard-coded to 1	MapUsingEmailAddress	<i>Not visible. Attempts to match existing compliance user record by email address.</i>
<b>Unique ID</b> in user properties	ExternalID	<i>Not visible.</i>



**Note:** The following *ImportedAccessingUser* table is used for all SaaS products and licenses with the exception of Office 365. (A user of both kinds of licenses appears separately in each of these imports.) Be aware that, for these SaaS products, there may be accessing users who come from outside your enterprise (for example, if you allow consultants or customers to access your Salesforce pages).

**Table 3:** User data mapping for other SaaS applications (*ImportedAccessingUser* table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
<b>Email</b>	UserName (up to 200 characters)	License properties > <b>Consumption</b> tab > <b>User</b>  External users show their email address. Users from within your enterprise (who have been matched on their email addresses) display their full name as a hyperlink to their user properties.
<b>Email</b>	SAMAccountName (up to 64 characters)	<i>Not visible in user properties.</i>

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Derived from the <b>Email</b> value	DomainName (up to 100 characters)	License properties > <b>Consumption</b> tab > <b>Domain name</b>
Internal ID	ExternalAccessingUserID	<i>Not visible. Used for record matching.</i>

**Table 4:** Access data mapping (ImportedClientAccessEvidence table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Internal ID app.id ( <i>not visible</i> )	ExternalAccessEvidenceID	<i>Not visible. Used for record matching.</i>
Managed Applications > <b>APPLICATION</b>	ProductName	Application properties > <b>General</b> tab > <b>Product</b>
Managed Applications > <b>VENDOR</b>	Publisher	Application properties > <b>General</b> tab > <b>Publisher</b>

**Table 5:** Access counting (ImportedClientAccessedAccessOccurrence table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Internal ID AppUsr.id ( <i>not visible</i> )	ImportedClientAccessedAccessEvidenceID	<i>Not visible. A separate record is created for each pairing of a user and application. For example, if UserA accessed two SaaS applications, there are two records.</i>
Hard-coded to 1 for each date	AccessCount	License properties > <b>Consumption</b> tab > <b>Last used count</b>
None. Uses the date when data is imported from Flexera One SaaS Management to IT Asset Management.	InventoryDate	Inventory device properties > <b>General</b> tab > <b>Last inventory date</b>
Does <i>not</i> use the <b>Effective Date</b> from the <b>License Details</b> tab of the license properties.	LicenseDate	<i>Not separately visible. Uses the inventory date in a different format.</i>
<b>Last Activity</b> listed for this user in the <b>Users</b> tab of the application properties.	AccessDate	Application properties > <b>Devices</b> tab > <b>Last used date</b>

**Table 6:** Evidence to application links (ImportedSoftwareTitleAccessEvidence table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Internal ID app.id ( <i>not visible</i> )	ExternalSoftwareTitleID	<i>Not visible. Foreign key to the ImportedSoftwareTitle table, identifies an imported application</i>

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Internal ID app.id (not visible) (same value)	ExternalClientAccess EvidenceID	Not visible. Foreign key to the ImportedClientAccessEvidence table, identifies an imported record of user access



**Note:** The following mapping into the ImportedComputer table creates (if it does not already exist) a dummy device record (needed within IT Asset Management, but not available in Flexera One SaaS Management). Only a single dummy computer record (for each connector) is required.

**Table 7:** Computer data mapping (ImportedComputer table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
None. Uses ComplianceConnectionID.	ExternalID	Not visible –
Hard-coded to "Flexera SaaS Manager"	ComputerName	Inventory device properties > <b>General</b> tab > <b>Name</b>
Hard-coded to "flexera.com"	Domain	Inventory device properties > <b>General</b> tab > <b>Domain name</b>
Hard-coded to "Flexera SaaS Manager " with the numeric value of the ComplianceConnectionID appended	SerialNo	Inventory device properties > <b>General</b> tab > <b>Serial number</b>
Hard-coded to "Flexera SaaS Manager"	InventoryAgent	Inventory device properties > <b>General</b> tab > <b>Last inventory source</b>
Hard-coded "False"	UntrustedSerialNo	Not visible
Hard-coded "False"	IsRemoteACLDevice	Not visible
Hard-coded "False"	IsDuplicate	Not visible
Hard-coded "False"	IgnoredDueToLicense	Not visible



**Note:** The following mapping into the ImportedSoftwareLicense table updates or creates either of two kinds of license, matching those in Flexera One SaaS Management:

- Named User licenses for Microsoft Office 365
- SaaS User licenses for any other imported SaaS applications.

**Table 8:** License data mapping (ImportedSoftwareLicense table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
<i>Not visible</i> termId	ExternalLicenseID	<i>Not visible</i>
License properties > <b>License Name</b> combined with the <b>Name</b> field (adjacent to the <b>License Type</b> control). For example: <ul style="list-style-type: none"> <li>License name = G Suite</li> <li>Name = Monthly/User</li> </ul>	LicenseName Example value imported is: <div>G Suite - Monthly/User</div>	License properties > <b>Identification</b> tab> <b>Name</b>
<b>Managed Applications</b> > <b>Vendor</b> for most applications (when creating SaaS User licenses); or for Named User licenses, value take from SKU is Microsoft Corporation.	PublisherName	License properties > <b>Identification</b> tab> <b>Publisher</b>
<b># of Items Allowed</b> as a number	EntitlementCount	License properties > <b>Compliance</b> tab > <b>Extra entitlements</b> (see also next item)
<b># of Items Allowed</b> shows Unlimited	UnlimitedConsumption	License properties > <b>Compliance</b> tab > <b>Entitlements are: Unlimited</b> (see also previous item)
Set to 13 (the ID for the Named User license type)	SoftwareLicenseTypeID	License properties > <b>Identification</b> tab> <b>License type</b> (displays as Named User)
License properties > <b>Ending Date</b> (or, in the list of <b>Application Licenses</b> , the <b>END DATE</b> )	ExpiryDate	License properties > <b>Identification</b> tab> <b>Expiry date</b>
The prefix FLX-0365- followed by the internal termSKU value ( <i>not visible</i> )	PartNo	License properties > <b>Purchases</b> tab > <b>Part no./SKU</b>
Hard-coded to true	IsSubscription	License properties > <b>Identification</b> tab> <b>Purchased as</b> displays Subscription
<i>Not visible</i> Id	SaaSManagerLicenseID	<i>Not visible</i> – used for creating the hyperlink from the license in IT Asset Management back to the matching license in Flexera One SaaS Management.





**Note:** The following mapping into the `ImportedSoftwareTitle` table updates or creates software titles (application records) for SaaS applications other than Microsoft Office 365.

**Table 9:** Application data mapping (`ImportedSoftwareTitle` table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
Not visible the application id	ExternalSoftwareTitleID	Not visible.
<b>APPLICATION</b> , and <b>Display Name</b> in application properties	FullName (up to 512 characters)	Application properties > <b>General</b> tab > <b>Name</b>
As above, <b>APPLICATION</b> , and <b>Display Name</b> in application properties	ProductName (up to 200 characters)	Application properties > <b>General</b> tab > <b>Product</b>
<b>Managed Applications</b> list > <b>VENDOR</b>	PublisherName (up to 200 characters)	Application properties > <b>General</b> tab > <b>Publisher</b>

**Table 10:** Application to License links (`ImportedSoftwareTitleLicense` table)

FSM object/label	FNMS staging fields	Example FNMS UI / Notes
	ExternalSoftwareTitleID	Not visible. Foreign key to the <code>ImportedSoftwareTitle</code> table, identifies an imported application
	ExternalSoftwareLicenseID	Not visible. Foreign key to the <code>ImportedSoftwareLicense</code> table, identifies a license created to match the imported data

# VII

## HP DDMI Adapter

IT Asset Management (Cloud)

This part introduces the FlexNet adapter for importing data from HP DDMI (for those enterprises that have not yet migrated to HPE Universal Discovery, for which see [HPE Universal Discovery Adapter](#)).

## 1

# Purpose and Architecture of the HP DDMI Adapter

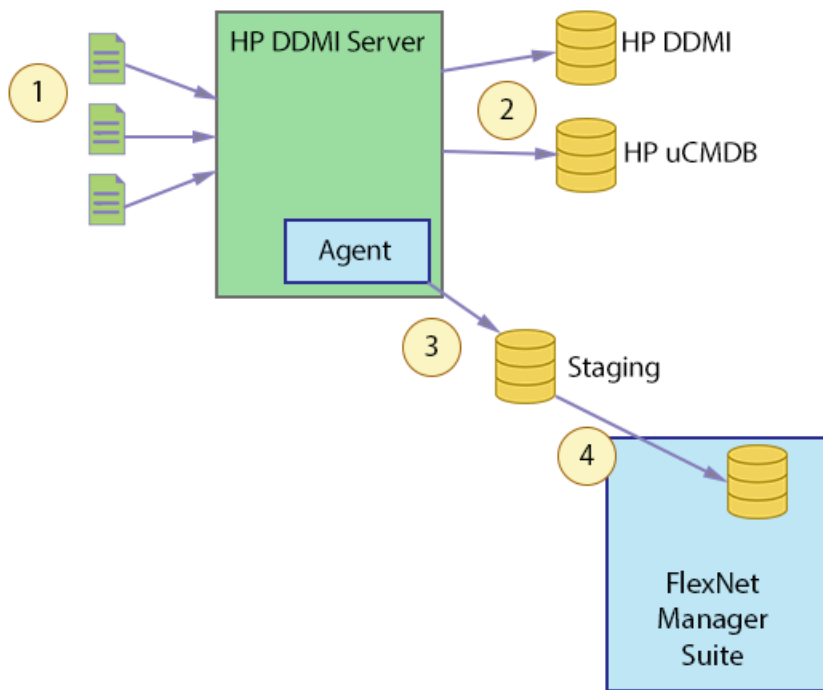
IT Asset Management (Cloud)

The FlexNet adapter for HP Discovery and Dependency Mapping Inventory (HP DDMI) collects data from your HP DDMI implementation, and stores it in an intermediate staging database. From there, the IT Asset Management importers bring the data into the compliance database so that you have a single, unified view of all inventory sources, and of the impact that the collected inventory has on your license position.

The work is done in two separate stages, and by two separate components:

- The FlexNet DDMI *agent* reads inventory directly from the HP DDMI incoming inventory (.xsf) files, and writes the data into the staging database.
- The FlexNet DDMI *adapter connection* connects to the staging database, massages the data into the format required for integration with IT Asset Management, and imports it to the compliance database.

This architecture is summarized in the following diagram, with the numbers described below:



1. XSF files (compressed XML files) are produced by HP agents and uploaded to a central HP DDMI server.
2. XSF files are imported into the databases for HP DDMI and HP uCMDB. These two stages are the normal operation of HP DDMI, and are unaffected by the presence of the FlexNet adapter.
3. The first component of the FlexNet adapter for HP DDMI (an agent which may be installed on the HP DDMI server, where the XSF files are uploaded, or on any other device with read access to the same collection of XSF files) reads the uploaded files and writes the data into the staging database required for this adapter. Copying the data from the original XSF files allows IT Asset Management to capture evidence that, if unrecognized by HP DDMI, is otherwise discarded.
4. The data import connection (configured from an inventory beacon, which is not shown in this simplified diagram) links to the staging database, extracts the latest data, and uploads it to the compliance database for IT Asset Management.

## 2

# Installation and Configuration

## IT Asset Management (Cloud)

Everything needed to set up the FlexNet adapter for HP DDMI is included in the Adapter Tools archive available from the Flexera Customer Community. You need credentials supplied by Flexera to access this download. The download includes:

- Two SQL scripts to define the schema for, and populate stored procedures for, the staging database.
- The agent executable `FnmPDDMIAgent.exe`, and an accompanying `CommandLine.dll` to enable command-line use.

This chapter includes all the details needed for installation and configuration. There are validation notes included at each stage of the set-up.

This chapter assumes that you have a functional implementation of HP DDMI, and an account with sufficient privileges on your HP DDMI server to install the agent executable there (or similar in your chosen location, if you are placing the agent elsewhere).

## Download Adapter Tools Archive

### IT Asset Management (Cloud)

The Adapter Tools archive includes content for many adapters, and is updated on the Flexera website from time to time. Start this procedure using a web browser on a computer that has good network accessibility from all the machines needing installations for your adapter.



---

**To download the adapter tools archive:**

1. Download the latest Adapter Tools for IT Asset Management *version*.zip archive from the Flexera Product and License Center:
  - a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
  - b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of Adapter Tools for IT Asset Management 2024 R2.3.zip commences.
  - c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a

convenient working location (such as C:\temp on a central, accessible server).

If your browser saves the file to a default location (such as your Downloads folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the zip archive, and choose **Extract All...**

The folders are now available for the range of adapters in the Adapter Tools archive.

## Creating the Staging Database

### IT Asset Management (Cloud)

The FlexNet adapter for HP DDMI requires a staging database, which is used as follows:

- The agent installed on your HP DDMI server reads the DDMI inventory files directly, performs minimal data rationalization, and saves the results to the staging database
- The connection from IT Asset Management reads the data from the staging database, shapes it appropriately, and uploads it to the compliance database.

The staging database can be created on any convenient instance of Microsoft SQL Server that meets the following criteria:

- Provides high-speed network access from the HP DDMI server (or other location of the installed agent), and from an appropriate inventory beacon that can upload to the application server for IT Asset Management
- Allows read/write access to the account running the FlexNet agent on the HP DDMI server, and read access by the service account running the BeaconEngine service on the inventory beacon (or you may choose to implement either a special Windows account and an SQL account to collect the DDMI staged data)
- Is running an instance of Microsoft SQL Server 2012 or later.

The staging database size will be roughly equivalent to the sum of the inventory (.xsf) files uploaded to your HP DDMI server. The agent overwrites any previous record from the same device with the latest inventory upload, to that the net database size does not grow appreciably larger than the sum of the source inventory files.

Your download of the Tier 1 adapters archive (see [Download Adapter Tools Archive](#)) included two SQL scripts that can assist with creating the staging database. You can achieve this through SQL Server Management Studio, or from the command line as described in the following procedure. Use whichever approach best suits your enterprise practices.



#### **To create the staging database from the command line:**

1. Navigate through the unzipped tools archive to Tier 1 Adapter Tools > HP Discovery and Dependency Mapping Inventory Tools > SQL.
2. If necessary, copy the script DDMISTagingSchema.sql from the this folder of your unzipped adapter archive to a temporary folder on your staging database server.
3. Logged in on the database server as an account with read/write access to the database, open a command prompt.
4. In the command prompt window, execute the following command, as amended:

```
sqlcmd -S ServerName\InstanceName -i TemporaryPath\DDMISTagingSchema.sql
```

where:

- *ServerName* is the name of the server hosting Microsoft SQL Server 2012 or later. Use a “.” (dot) if you are running the staging script on the same server as the database instance; or if you are adapting these instructions to a different environment, insert the server name or its IP address.
- *InstanceName* is the name of the database instance to use for the database staging tables. You can omit this parameter (and the backslash separator) if the default instance name is being used.
- *TemporaryPath* is the location where you saved the SQL procedure.

---

Example:

```
sqlcmd -S .\Development -i C:\temp\DDMIStagingSchema.sql
```

The database DDMIStaging is created with all necessary tables, indices, and so on.

5. Similarly, execute the second script (with the same parameter substitution) to create the stored procedures necessary in the staging database.

```
sqlcmd -S ServerName\InstanceName -i TemporaryPath\DDMIStagingProcedures.sql
```

---

Example:

```
sqlcmd -S .\Development -i C:\temp\DDMIStagingProcedures.sql
```

The staging database is now ready for operation.



**Tip:** If you wish to do so, you may now rename this database. Keep the name handy for the configuration processes that follow.

The schema for the staging database (below) is, in the main, self-explanatory, reflecting the content of the .xsf files found on your HP DDMI server. The one unique table is InventoryImportDate. This table lists the names of inventory files that have already been loaded into the staging database, and saves the last modified date of each file as its most recent inventory date. When the FlexNet agent scans through the inventories for possible import into the staging database, it uses this table to manage differential imports, only importing an inventory file if it is newer than the last recorded inventory file of that name that was imported.

**StagedComputer**

Column Name	Data Type	Allow Nulls
ExternalComputerID	int	<input type="checkbox"/>
AssetTag	nvarchar(256)	<input type="checkbox"/>
ComputerName	nvarchar(256)	<input checked="" type="checkbox"/>
DomainQualifiedName	nvarchar(100)	<input checked="" type="checkbox"/>
DomainFlatName	nvarchar(32)	<input checked="" type="checkbox"/>
OperatingSystem	nvarchar(128)	<input checked="" type="checkbox"/>
ServicePack	nvarchar(128)	<input checked="" type="checkbox"/>
NumberOfProcessors	int	<input checked="" type="checkbox"/>
ProcessorType	nvarchar(256)	<input checked="" type="checkbox"/>
MaxClockSpeed	int	<input checked="" type="checkbox"/>
NumberOfCores	int	<input checked="" type="checkbox"/>
NumberOfSockets	int	<input checked="" type="checkbox"/>
TotalMemory	bigint	<input checked="" type="checkbox"/>
ChassisType	nvarchar(128)	<input checked="" type="checkbox"/>
ChassisSerialNumber	nvarchar(100)	<input checked="" type="checkbox"/>
NumberOfHardDrives	int	<input checked="" type="checkbox"/>
TotalDiskSpace	bigint	<input checked="" type="checkbox"/>
NumberOfNetworkCards	int	<input checked="" type="checkbox"/>
NumberOfDisplayAda...	int	<input checked="" type="checkbox"/>
IPAddress	nvarchar(256)	<input checked="" type="checkbox"/>
MACAddress	nvarchar(256)	<input checked="" type="checkbox"/>
Manufacturer	nvarchar(128)	<input checked="" type="checkbox"/>
ModelNo	nvarchar(128)	<input checked="" type="checkbox"/>
SerialNo	nvarchar(100)	<input checked="" type="checkbox"/>
LastLoggedOnUser	nvarchar(64)	<input checked="" type="checkbox"/>
InventoryDate	datetime	<input checked="" type="checkbox"/>
InventoryAgent	nvarchar(128)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

**StagedDomain**

Column Name	Data Type	Allow Nulls
QualifiedName	nvarchar(100)	<input type="checkbox"/>
Flatname	nvarchar(32)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

**StagedInstallerEvidence**

Column Name	Data Type	Allow Nulls
ExternalComputerID	int	<input type="checkbox"/>
ExternalInstallerID	int	<input type="checkbox"/>
DisplayName	nvarchar(256)	<input type="checkbox"/>
Version	nvarchar(72)	<input checked="" type="checkbox"/>
Publisher	nvarchar(200)	<input checked="" type="checkbox"/>
Evidence	nvarchar(32)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

**StagedFileEvidence**

Column Name	Data Type	Allow Nulls
StagedFileEvidenceID	int	<input type="checkbox"/>
FileName	nvarchar(256)	<input type="checkbox"/>
FileVersion	nvarchar(100)	<input checked="" type="checkbox"/>
ProductVersion	nvarchar(100)	<input checked="" type="checkbox"/>
ProductName	nvarchar(100)	<input checked="" type="checkbox"/>
Company	nvarchar(100)	<input checked="" type="checkbox"/>
Description	nvarchar(100)	<input checked="" type="checkbox"/>
FileSize	int	<input checked="" type="checkbox"/>
Language	nvarchar(200)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

**StagedComputerFileEvidence**

Column Name	Data Type	Allow Nulls
ExternalFileID	int	<input type="checkbox"/>
ExternalComputerID	int	<input type="checkbox"/>
StagedFileEvidenceID	int	<input type="checkbox"/>
FilePath	nvarchar(400)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

**StagedUser**

Column Name	Data Type	Allow Nulls
ExternalUserID	int	<input type="checkbox"/>
UserName	nvarchar(64)	<input checked="" type="checkbox"/>
Domain	nvarchar(100)	<input type="checkbox"/>
SAMAccountName	nvarchar(64)	<input type="checkbox"/>
		<input type="checkbox"/>

**StagedWMIEvidence**

Column Name	Data Type	Allow Nulls
ExternalWMIEvidenceID	int	<input type="checkbox"/>
ExternalComputerID	int	<input type="checkbox"/>
ClassName	nvarchar(50)	<input type="checkbox"/>
PropertyName	nvarchar(50)	<input type="checkbox"/>
PropertyValue	nvarchar(256)	<input type="checkbox"/>
		<input type="checkbox"/>

**InventoryImportDate \***

Column Name	Data Type	Allow Nulls
InventoryFileName	nvarchar(256)	<input type="checkbox"/>
InventoryDate	datetime	<input checked="" type="checkbox"/>
		<input type="checkbox"/>



# Configuring the FlexNet Agent for HP DDMI

## IT Asset Management (Cloud)

The FlexNet agent for HP DDMI is the code entity that accesses the inventory (.xsf) files uploaded to your HP DDMI server, and saves the resulting data to the staging database that you have just established. Operationally, it consists of two parts:

- A small C# application (an .exe and a .dll) that acts only as a transport, decompressing the .xsf files (which are compressed XML) and feeding them to the second part
- A stored procedure, already created in the staging database (see [Creating the Staging Database](#)), that unpacks the XML elements and saves the resulting data in the staging database.

Clearly, it is only the first of these that is the focus of this topic.

The agent may be installed anywhere, as long as it has access to the folder on your HP DDMI server where the .xsf inventory files are uploaded, and to the staging database. Optional locations include:

- On the HP DDMI server (optimal from a network performance viewpoint)
- On the server hosting the staging database
- On an inventory beacon that has fast network access to both of the above
- On any other computer meets that same requirement of fast network access to the data source and destination.

Although much of the inventory processing load is delegated to the SQL stored procedure, you can further control the impact of the executable on its host device by specifying the number of CPU threads it devotes to throughput of the .xsf files. In any case, the agent is normally scheduled to run only once daily in off-peak hours.



### To configure the agent executable:

1. Navigate through the unzipped tools archive to Tier 1 Adapter Tools > HP Discovery and Dependency Mapping Inventory Tools > DDMI Agent.
2. Copy both the FnmpDDMI Agent.exe and CommandLine.dll files to their new operational location.



**Important:** Both files must be installed in the same folder.

3. Identify or configure the account that is to run the agent.

A suitable account:

- Has execute rights on the device where the agent is installed
- Has at least read-only privileges on the folder where the .xsf inventory files are saved on your HP DDMI server
- Has read/write privileges on the staging database.

You may prefer to make this a service account to prevent interactive use.

4. Configure a Windows schedule task (or use your preferred scheduling tool) to trigger execution of the agent running as your selected account on your preferred schedule.

Choose a schedule to suit your environment. This should allow enough time to:

- Process all new .xsf files into the staging database
- Upload the results to the compliance database for IT Asset Management
- Align with the nightly full import and license compliance calculations, by default scheduled for 2am nightly.

A suggested schedule is daily at 10pm.

The task must invoke the FnmpDDMIAgent.exe executable with a command line that includes some of the following parameters (the agent also supports -h to list command-line options):

```
FnmpDDMIAgent.exe
-s PathToInventories
-d StagingDBConnectionString
-l LoggingLevel
-t TimeOutSeconds
-n ThreadCount
```

where the parameters are:

Switch	M/O	Value
-s	Mandatory	The location of the HP DDMI inventory files (no default value). The path may be enclosed in double quotation marks, which are mandatory when there is any white space in the file path. Example: "C:\HPDDMI\Inventories"
-d	Mandatory	The connection string for the staging database (no default value). Example: "server=172.16.38.83;Database=DDMIStaging;Trusted_Connection=yes"
-l	Optional	An integer in the range 0-3 for the level of logging: <ul style="list-style-type: none"><li>• 0: no logging (default value)</li><li>• 1: Errors only</li><li>• 2: Errors and warnings</li><li>• 3. Verbose logging.</li></ul>
-t	Optional	The number of seconds the executable should wait for a response to any command given to the staging database. Default value is 3600 (an hour).
-n	Optional	An integer in the range 1-20 (default 5), being the number of CPU threads the agent should use for processing the .xsf inventory files. Each thread invokes the database stored procedures independently.



**Tip:** Depending on the hosting device architecture, optimum throughput is achieved somewhere within the range of 1-20 threads, after which increasing the number of threads overloads the system and decreases performance. Given the variety of factors involved, there is no real substitute for experiment to determine the optimum value for your environment.

Example usage (all on one line):

```
FnmpDDMIAgent.exe
-s "C:\HPDDMI\Inventories"
-d "server=172.16.38.83;Database=DDMIStaging;Trusted_Connection=yes"
-l 2
```

When thus invoked, the agent reads through all updated XSF files in the source directory, and uploads their data into the staging database. Files unchanged in the nominated path since the last upload are ignored.

5. On the assumption that you already have .xsf files in the nominated folder on your HP DDMI server, you may wish to run the agent once now, and use Microsoft SQL Server Studio to validate the content is stored in the staging database. (Loading the staging database also provides material for testing the next stage of the process.) If you enabled logging, you may also check the log file, which is written to the same folder where the executable is running.

## Configuring Upload and Import Connection

IT Asset Management (Cloud)

With data arriving in the staging database, it's time to configure its upload to the central application server for IT Asset Management, followed by its import to the compliance database. This process is configured on, and managed by, an inventory beacon.



**To configure the connection for upload from the staging database:**

1. Run the inventory beacon interface (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Tip:** Remember that you must run the inventory beacon software with administrator privileges.

2. Open the **Inventory systems** page.
3. Expand the pull-down beside the **New...** button below the list, and choose **SQL Server**.

The **Create SQL Source Connection** dialog appears.

4. Complete the details to define the connection to the staging database:
  - a. Create a distinctive **Connection Name** that is easily recognized in the list of connections (even when the columns are narrow).  
For example: HPDDMI Staging.
  - b. For **Source Type**, you must choose **HP Discovery and Dependency Mapping Inventory (DDMI)**.
  - c. Provide the host name or IP address (in IPv4 format) for the **Server** hosting the staging database.
  - d. Choose the **Authentication** method that the BeaconEngine process should use to connect to the database:

- If you select **Windows Authentication**, ensure that the service account that runs the BeaconEngine process on this inventory beacon has (as a minimum) read access to the staging database.
  - With either of the remaining two options, specify the **Username** and **Password** in the next two fields. Of course, validate that the nominated account has (as a minimum) read access to the staging database.
- e. Use the **Test connection** button to validate that you have specified the details correctly, and adjust if necessary.
  - f. For the **Overlapping Inventory Filter**, accept the default **Import the inventory from this source for possible merging** selection (unless you are in the process of migrating inventory collection away from HP DDMI, in which case see the online help for the **Inventory systems** page).
  - g. Click **Save** to close this dialog and add your new connection to the list in the **Inventory systems** page.
5. If you do not already have a suitable schedule for this connection available on this inventory beacon, select the **Scheduling** page, and click **New...** to open the **Edit Schedule** dialog and define one.
  6. Back on the **Inventory systems** page, select the new connection in the list, and click **Schedule...** to select your appropriate schedule in the **Select schedule** dialog. Click **OK** to confirm your selection.
  7. In the main **FlexNet Beacon** interface, click **Save** to store your newly scheduled connection.
  8. If you have already run the agent to populate your staging database, you may test the new connection by keeping it selected in the list, and clicking **Execute Now**.

The upload, the resulting import (which follows immediately), and the subsequent license compliance calculation may take some time to complete. You may monitor progress in the web interface for IT Asset Management — go to the **System Tasks** page (**Data Collection > IT Assets Inventory Status > System Tasks**), and see the online help for that page. When the process is complete, you can inspect the inventory imported from HP DDMI in the **All Inventory** page.



**Tip:** In the **All Inventory** page:

- a. Add the **Connection name** to the listing from the column chooser
- b. Display the simple filter bar
- c. Filter for the name you have your HP DDMI connection (the suggestion was *HPDDMI Staging*).

# VIII

## HPE Universal Discovery Adapter

### IT Asset Management (Cloud)

You can use the HPE Universal Discovery (HPE-UD) adapter tool to collect and import inventory data from HPE Universal Discovery System to IT Asset Management. The HPE-UD adapter fetches all hardware, software, and virtualization information from the HPE-UD system and stores it in the compliance database maintained by IT Asset Management. The HPE-UD adapter support is available only with IT Asset Management version 2015 and later.



**Note:** The HPE-UD adapter tool works only with version 10.10, 10.11, 10.33–11.5, 2020.05, 2020.08, 2020.11, 2021.05, 2021.11 of the HPE Universal Discovery System. These versions do not support Solaris 11 zones, for which reason the HPE-UD adapter cannot import these zones. Since host serial number and zone name are used for rationalizing duplicate inventory records across different inventory sources, this may mean that a device imported through the HPE-UD adapter cannot be merged with another record of the same device imported through another inventory source that supports zones information.

# Selecting a Configuration

IT Asset Management (Cloud)

This chapter gives an overview of the architecture, working, and configuration of the HPE-UD adapter. Choose the appropriate configuration for your enterprise before you implement the adapter.

## Architecture and Working of the HPE Universal Discovery Adapter

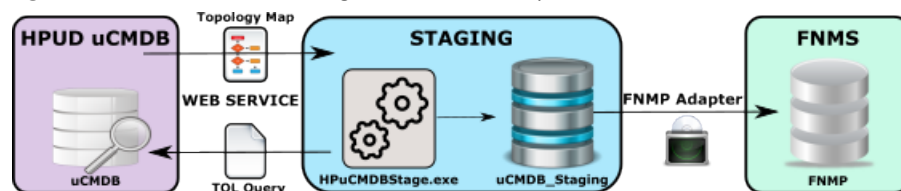
IT Asset Management (Cloud)

The HPE-UD adapter architecture has the following main components:

- **HPE-UD Server:** This server has the HPUD uCMDB database and HPE discovery tools installed on it. The HPUD uCMDB database stores the HPE-UD inventory details.
- **Staging Server:** This server contains the Flexera HPuCMDBStage.exe tool and a staging database. The HPuCMDBStage.exe tool retrieves topology maps by executing TQL queries on HPUD uCMDB. The retrieved topology maps are stored in a staging database (uCMDB\_Staging) present on the staging server. The query execution is performed using the web service interface of HPUD uCMDB.
- **IT Asset Management:** The Compliance Reader component residing on this server fetches data from the staging database and loads it into the IT Asset Management database when you run an inventory import.

The following diagram shows the architecture and working of the HPE-UD adapter:

**Figure 6:** Architecture and working of the HPE-UD adapter



1. The HPuCMDBStage.exe tool extracts the inventory information through the web API of HPUD uCMDB. This tool executes Topology Query Language (TQL) queries on the HPUD uCMDB database and retrieves responses in the

form of topology maps. With the default configuration, this tool stores the retrieved topology maps directly into the staging database without storing them in XML files. However, you can configure this tool to write the retrieved topology maps to XML files and then write data from those XML files to the staging database.



**Note:** One topology map is generated for each TQL query executed on HPUD uCMDB.

2. The Compliance Reader component of IT Asset Management collects data from the staging database. When you run an inventory import on the IT Asset Management server, the data extracted from the staging database is written to the compliance database on the application server

## The Adapter Executable

### IT Asset Management (Cloud)

HPE Universal Discovery (HPE-UD) adapter uses the `ucmdb-api` to obtain the version of the uCMDB instance. It uses the `UcmdbService` web service API to retrieve the topology maps by executing TQL queries on the uCMDB instance. You can retrieve the WSDL definition of the `UcmdbService` web service API from `http://<uCMDB Server>/axis2/services/UcmdbService?wsdl`. The configuration of the staging tool includes the TQL queries that will be executed on the HPUD uCMDB database. You don't have to import the TQL queries into the HPUD uCMDB database and there is no need to deploy and maintain the TQL queries on every HPE-UD instance of your enterprise.



**Note:** The URL to retrieve the WSDL definition of the `UcmdbService` web service API may use 'https' instead of 'http'. It may also include a port number. Please check with your HPE-UD system administrator for more details on the web service URL.

The tool to query the web API has the following two parts:

- `HPuCMDBStage.exe` — A console program capable of querying the `UcmdbService` API of HPE-UD and writing the results into an SQL Server database, and optionally to XML files on the local file system. This program supports command line arguments, available using `HPuCMDBStage.exe -h`. Here is the list of available command line options.

<code>-h</code>	This help
<code>-x &lt;settings file&gt;</code>	Settings file
<code>-s &lt;address or URL&gt;</code>	uCMDB server address or URL
<code>-u &lt;user name&gt;</code>	uCMDB export user name
<code>-p &lt;password&gt;</code>	uCMDB export user password
<code>-c &lt;connection string&gt;</code>	SQL Server staging database connection string
<code>-m &lt;staging method&gt;</code>	Set staging method (stage/staged/prestaged/stream)
<code>-f &lt;staging file path&gt;</code>	Set the staging file path

- `FNMPuCMDBSettings.xml` — The self-documenting configuration file for `HPuCMDBStage.exe` which contains TQL queries executed against HPE-UD, and can include connection settings for HPE-UD and SQL Server.

In operation, the executable, `HPuCMDBStage.exe`, extracts the inventory data from HPE-UD and saves it for further processing. The value of the `method` parameter determines how the data is saved. The `method` parameter can have the following values:

- `Stage` — This option enables you to save inventory data from HPE-UD to a series of XML files on the staging server

**(HPE-UD to XML).** The XML files are not imported into the staging database, but you can review the extracted inventory data. This option also enables you to collect inventory data from the HPE-UD servers that are not directly connected to the staging server. You can manually copy and upload the inventory XML files to the staging server. Also see the Prestaged method below.

- **Staged** — This options enables you to write the data extracted from the HPE-UD servers to XML files and then copy the information to the staging database **(HPE-UD to XML/SQL)** where it can be imported into IT Asset Management for use in compliance calculations.
- **Prestaged** — This option takes information stored in XML files (from the Stage and Staged method) and loads it into a staging SQL database **(XML to SQL)**. Inventory is not gathered from HPE-UD in this mode.
- **Stream** — This option enables you to extract the inventory information from HPE-UD and load it into the staging database directly, without storing it in XML files on the staging server **(HPE-UD to SQL)**. You can import these files to into IT Asset Management for use in compliance calculations.

You can set the default values for the method parameter and all other parameters in the `FNMPuCMDBSettings.xml` file. The adapter tool uses the default parameter values when you use it without other command-line options. The settings file is self-documented and the matching command-line options are available using `HPuCMDBStage.exe -h`.

## Files Created by the Staging Tool

The following files are created when the staging tool performs a read/write operation on the local disk of the staging server. You can view the contents of the following files to review HPE-UD configuration item details that have been extracted.

Filename	Content
<code>Computer.xml</code>	Details of computers and their properties.
<code>Virtualization.xml</code>	Virtualization and partitioning details.
<code>InstalledSoftware.xml</code>	Installed software evidence linked to each computer.



## 2

# Installation and Configuration

## IT Asset Management (Cloud)

For **cloud** implementations of IT Asset Management, you need to download the staging tool, the configuration file, and the staging schema for this tool from the Flexera Customer Community. You do not, of course, need to make any changes to your central application server, but those additional components are required in the download.

You need credentials supplied by Flexera to access this download. Details of the download are included in [Download Adapter Tools Archive](#).

- The HPE-UD adapter suits IT Asset Management releases 2015 and later for on-premises delivery.
- The build number for this adapter is 10.3.0.12006 (or higher). You can identify this number by right-clicking the HPuCMDBStage.exe file in Windows Explorer, selecting **Properties**, and looking at the **Details** tab.

Save the zipped archive to a suitable temporary location, and unzip it.

Full details of setting up the HPE-UD adapter are included in this chapter. The following chapter (see [HPE-UD Adapter Operation](#)) covers testing the completed installation and validating the results.

## Download Adapter Tools Archive

### IT Asset Management (Cloud)

The Adapter Tools archive includes content for many adapters, and is updated on the Flexera website from time to time.

Start this procedure using a web browser on a computer that has good network accessibility from all the machines needing installations for your adapter.



---

#### **To download the adapter tools archive:**

1. Download the latest Adapter Tools for IT Asset Management *version*.zip archive from the Flexera Product and License Center:
  - a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
  - b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of `Adapter Tools for IT Asset Management 2024 R2.3.zip` commences.

- c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as `C:\temp` on a central, accessible server).

If your browser saves the file to a default location (such as your `Downloads` folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the zip archive, and choose **Extract All...**

The folders are now available for the range of adapters in the Adapter Tools archive.

## Selecting a Staging Server

IT Asset Management (Cloud)

Multiple configuration options are there to configure the staging server. You can install the staging server on any of the following types of computers:

- A dedicated stand-alone server or a virtual machine.
- Any other suitable machine in your enterprise, such as a print server.
- An inventory beacon.

The following are the requirements for a staging server:

- Microsoft Windows 2008 or later.
- Access to an instance of Microsoft SQL Server 2012 or later to host the staging database. You can implement the staging database on the staging server or on a separate database server.
- Microsoft .NET 4.0 run time environment installed.
- Efficient network access to each HPE-UD server in your enterprise, using the HTTP or HTTPS protocols.

## Creating the Staging Database Tables

IT Asset Management (Cloud)

Once your selected staging server is able to access the Microsoft SQL Server instance, you can use the provided script to create the staging database and set up the appropriate database tables. You can achieve this through SQL Server Management Studio, or from the command line as described in the following procedure.



### ***To create the staging database from the command line:***

1. Download and install the Adapter Tools archive. For more information, see [Download Adapter Tools Archive](#).
2. Navigate through the unzipped archive to `Adapter Tools > HP Universal Discovery Tools > SQL`.
3. If necessary, copy the script `uCMDB_staging.sql` from the `SQL\` folder of your unzipped adapter archive to a temporary folder on your staging server.

4. Open a command prompt on the staging server.
5. In the command prompt window, execute the following command, as amended:

```
sqlcmd -S ServerName\InstanceName -i TemporaryPath\uCMDB_staging.sql
```

where:

- The database `uCMDB_staging` is created with all necessary tables, indices, and so on.
- `ServerName` is the name of the server hosting Microsoft SQL Server 2012 or later. You can install the database on the staging server or on any remote server. You can use the name of the staging server or its IP address, or a “.” (dot) if you are running the staging script on the same server as the database instance.
- `InstanceName` is the name of the database instance to use for the database staging tables. You can omit this parameter if the default instance name is being used.
- `TemporaryPath` is the location where you saved the SQL procedure.

Example:

```
sqlcmd -S 192.100.0.20\Development -i C:\temp\uCMDB_staging.sql
```

The staging database is now ready for operation.

## Configuring HPE Universal Discovery System

IT Asset Management (Cloud)

You must configure the HPE-UD adapter settings to include a user which can perform the Run Legacy API and the Run Query by Definition actions on the HPE-UD system.



**To create and configure a user in HPE-UD:**

1. In the HPE-UD interface, click **Security, Roles Manager**.
2. Create a new role for FlexNet Manager import.
3. Select the role that you just created and click **General Actions** and assign the Run Query By Definition and the Run Legacy API actions to this role.
4. In the HPE-UD interface, click **Security, Users and Groups**.
5. Create a new user for FlexNet Manager import.
6. Click the **Roles** tab and assign the role that you created in Step 2 to this user.
7. Click the **Permissions Overview** tab and review the assigned permissions for this role.
8. Click **Data Flow Management, Universal Discovery, Zone-Based Discovery** and expand the **Mapping Options** section in **Preferences**.

9. Check the **Raw OS Installed Software** and set the **Include** option to `name=.*`. This enables HPE-UD adapter to capture all raw installation evidence.

You have created and configured an HPE-UD user for use in the HPE-UD staging tool. You must add this user to the `FNMPuCMDBSettings.xml` file.

## Installing and Configuring the Staging Tool

IT Asset Management (Cloud)

This procedure includes many separate sub-processes to complete the setup of the HPE-UD adapter.



### *To install and configure the HPE-UD adapter:*

1. Ensure that the account under which the adapter executable will run has read/write/execute permissions on this `uCMDB_staging` database. Authentication may be through Windows authentication or SQL Server authentication. Using Windows authentication, the default account is the username running the `HPuCMDBStage.exe` adapter. The username and password for SQL Server authentication are specified in the database connection string, which you may supply in `FNMPuCMDBSettings.xml`, or override with the `-c` option on the command line.
2. To extract information from the HPE-UD system, the `HPuCMDBStage.exe` tool must have the required permissions to query the HPE-UD system. You must create a user in the HPE-UD system and assign it with permission to perform the Run Legacy API and the Run Query by Definition actions. You must specify this user in the `FNMPuCMDBSettings.xml` file. For more information on configuring the HPE-UD system, see [Configuring HPE Universal Discovery System](#).
3. Navigate to the fixed location on the inventory beacon where the inventory reader must find configuration files to control its uploads. Verify the existence of the following path: `C:\ProgramData\Flexera Software\Compliance\ImportProcedures\Inventory\Reader\HP Universal Discovery`.
4. Create a folder to contain the adapter executable and its configuration file. Location is not critical; a suggested path is under `C:\Program Files\Flexera Software` (for version 2015 and later, or for a stand-alone server that is not an inventory beacon or application server). In your chosen location, create a folder such as `HPUDAdapter`.
5. From the `HPuCMDBStage\` folder within your unzipped archive, copy both `HPuCMDBStage.exe` and `FNMPuCMDBSettings.xml` to your newly created folder (such as `C:\Program Files\Flexera Software\HPUDAdapter`).
6. Open your copy of `FNMPuCMDBSettings.xml` in a text editor of choice, and review the self-documenting comments within that file. Modify the following values as required. Ensure that you specify the IP address of your HPE-UD server. You can use the command line options to configure `HPuCMDBStage.exe`. Use the `'-h'` option to get the list of available options.
  - Update values in the first element describing the downstream connection to the HPE-UD server, including the IP address, port, the account name and password for access. Keep a record of the account name and password for registering with HPE-UD. The default values are:

```
<server protocol="http" address="10.200.20.138"
```

```
port="8080" username="exportuser" password="Pa$$w0rd" timeout="3600"/>
```



**Tip:** If you do not wish to record the password in the plain text configuration file, you can use a script to retrieve the password from an encrypted store, and supply it as a command-line option when starting the `HPuCMDBStage.exe` tool. Here is an example:

```
HPuCMDBStage.exe -p <password>
```

- Update the second element for the connection to the staging database. The default values are:

```
<database
connection-string="Server=.;Database=uCMDB_Staging;Trusted_Connection=yes;"/>
```

- Update the third element to configure whether, and where, the executable should save XML files of the inventory collected from HPE-UD. The default value stores any XML files below the location of the executable: `<staging path="StagingData" method="stream"/>` You may wish to redirect the path setting for easier access for human inspection.
  - Make sure that the user details in the `<Server>` section match with the user configured in the HPE-UD system.
  - Save the settings file.
7. Validate the HPE-UD adapter operation. For more information about validating the adapter, see [Validating the HPE-UD Adapter](#).
  8. Assuming that you do not wish to trigger the adapter manually every time it needs to run, start Windows Task Scheduler, and create a basic task to run the adapter.



**Important:** It is suggested that you schedule the adapter to run at times which cannot overlap with the inventory reader uploading the results to the application server. Check the schedule for the compliance reader on the central application server, and avoid this time slot.

By default, the inventory import (starting with the reader) is triggered around 2 am. Therefore you might consider scheduling this task for some time such as 10 pm daily. The command line for the scheduled task (assuming that you have saved your preferred settings) is simply to invoke the executable. Any parameters not specified on the command line are taken from the settings file in the same folder as the executable. Here is an example:

```
HPuCMDBStage.exe -x <settings file>
```

This completes the configuration of the adapter executable.

## 3

# Operation and Validation

IT Asset Management (Cloud)

This section describes the normal operation of HPE-UD adapter and also lists the steps to validate its operation. This section has the following tasks:

- HPE-UD Operation: See [HPE-UD Adapter Operation](#)
- Validating the HPE-UD Adapter: See [Validating the HPE-UD Adapter](#)

## HPE-UD Adapter Operation

IT Asset Management (Cloud)

Normal operation of the HPE-UD adapter relies on the following sequence of events:

1. According to the scheduled instructions, the adapter reads the current content of the HPE-UD database and stages the new data in the staging database after removing the previously stored data. A flag stored in the staging database indicates the status of the data extraction. The resulting status is flagged within the database. For more information on installing and configuring the staging tool, see [Installing and Configuring the Staging Tool](#)
2. Following the schedule on the central application server, and provided that the staging status is Success, the import reader uploads this content to the inventory database.
3. The next compliance import brings the final data set into the compliance database, where it is automatically taken into account for compliance calculations. As always, for compliance calculations to proceed, the inventory records must be recognized by the Application Recognition Library, and you must have the resulting application records linked to the appropriate license.

## Validating the HPE-UD Adapter

IT Asset Management (Cloud)



**To validate the adapter operation:**

1. Manually trigger the adapter executable.

You can specify a value for the `method` parameter if you need to override the default settings set in the settings XML file. For example:

```
'C:\Program Files\Flexera Software\HPUD\HPuCMDBStage.exe' -f 'C:\temp' -m staged
```

This will write XML files under your `C:\temp` directory for review. It will also write data into the staging database.

2. Inspect the saved XML files to validate the inventory gathered.
3. Use SQL Server Management Studio to validate that the data is written to the staging database. Also review the `StagingState` property in the `uCMDBStagingDatabaseConfiguration` table in the staging database. Possible values are `Running`, `Failed`, or `Success`. This value must be `Success` before the HPE-UD data can be uploaded from the staging database to the central inventory database.
4. Wait until the next inventory import and calculations have run. You might have to wait overnight for this.
5. Use IT Asset Management to validate that new evidence has been recovered. Identify which evidence has been recognized by the ARL and which new rules are required. Link the applications to appropriate licenses.



# Inventory Adapter Studio

IT Asset Management (Cloud)

Inventory Adapter Studio is a tool to develop adapters for custom inventory gathering into IT Asset Management.

This section covers the use of Inventory Adapter Studio and the management of the adapters you develop.



## 1

# What Is Inventory Adapter Studio?

## IT Asset Management (Cloud)

As well as gathering inventory directly from devices in your computing estate, IT Asset Management can also import inventory gathered by other software and hardware inventory tools. IT Asset Management ships with several factory-supplied adapters that read data from inventory tools such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) or IBM's ILMT. The Inventory Adapter Studio enables modification of these existing adapters; but more importantly, it allows creation of new adapters to connect with systems not supported out of the box.



**Note:** *The Inventory Adapter Studio is tailored specifically to building adapters for inventory tools. IT Asset Management is also able to import additional business-related data that influences license compliance calculations, but these connectors are built with the separate Business Adapter Studio.*

The Inventory Adapter Studio provides the following benefits:

- A graphical user interface that simplifies creation and editing of the underlying XML files that define the adapters.
- Template adapters, which include approved code for data transformation and import into the IT Asset Management operations database. This allows you to focus exclusively on the data gathering aspects of the adapter.
- Syntax highlighting, and highlighting of steps that require further editing.
- Data isolation by hiding test connections from your production implementation of IT Asset Management.
- Reduced testing effort, with a built-in filter to limit the number of records processed in a testing cycle.
- Context sensitive error reporting, with progress monitoring of each statement in the user interface.
- Detailed error reporting and tracing.


At the end of the section on the Inventory Adapter Studio, the object model for inventory adapters running on your inventory beacon is fully documented.

## 2

# Cautions, Prerequisites, and References

IT Asset Management (Cloud)

---

 **Caution:** Be aware that the Inventory Adapter Studio is an advanced tool, and incorrect use can result in data changes in your IT Asset Management environment that will affect your license position. If you are uncomfortable with performing these changes yourself, please contact the Flexera services team.

The Inventory Adapter Studio has the following requirements:

- **Location.** The Inventory Adapter Studio must be an inventory beacon that will run the downstream functions of the adapter, connecting to the third-party inventory source within your enterprise. The intermediate files collected on this inventory beacon are then uploaded automatically to the central server in the cloud.
- **File access.** On the beacon where it is installed, Inventory Adapter Studio requires permission to create and edit files in the C:\ProgramData\Flexera Software\Compliance\ImportProcedures directory.
- **Downstream database access.** Inventory Adapter Studio requires permission to access the source databases. Read-only access may be sufficient, depending on how you write the adapter: many adapters write temporary tables on the source database to allow joins with existing data (for example, to create differential inventory of changes since the last import). Clearly, testing adapters such as these means that Inventory Adapter Studio must have full access to the source database.
- **Upstream database access.** Inventory data gathered by the adapters created on the inventory beacon is uploaded automatically to the hosted service, and imported into the operations database there.

## Operator Requirements

To use the Inventory Adapter Studio successfully, you will need:

- A working knowledge of SQL, so that you can modify queries in the templates provided.
- Some data analysis experience.
- To edit the SQL queries to collect data from the source database(s), you need to know where to get the data in those source databases is located. This probably requires read access to the database and also access to the inventory tool's

user interface for data validation.

- Direct access to the inventory beacon for adapter development and testing. This may be on the beacon console directly, or using terminal services.
- A detailed working knowledge of the IT Asset Management product. This is required so that data imports can be verified, along with the expected application installations.

## Restrictions

Several inventory adapters are supplied as standard with IT Asset Management (these are sometimes referred to as "Tier 1" adapters). These live on the central server, and are automatically downloaded to inventory beacons as required. This means:

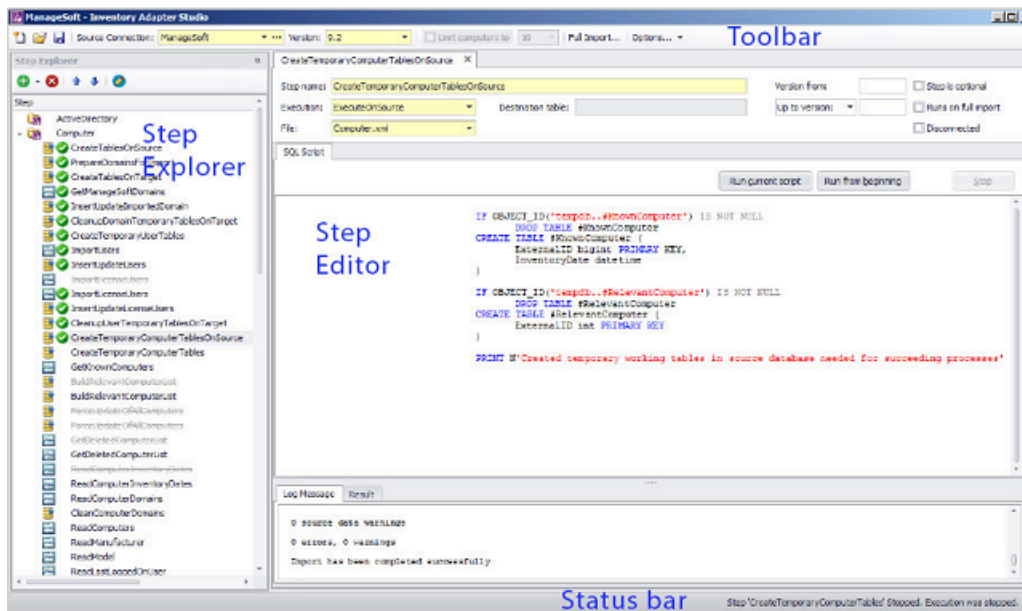
- You cannot modify Tier 1 inventory adapters.
- You cannot store anything else in the same folder on the inventory beacon used to store Tier 1 adapters distributed by the central server. Changes are always removed automatically within a short time, keeping the distributed adapters safe and in line with the latest versions stored on the central server.
- You can create custom inventory adapters on inventory beacons, using the Inventory Adapter Studio and the object adapter model described in the following topics. These are stored separately, and are not over-written by downloads from the central server. This also means that a custom inventory adapter is by nature restricted to the inventory beacon on which it is created. However, if you need the identical adapter operating from multiple inventory beacons, you can manually copy the adapter between beacons, always using the same file path (*%CommonAppData%\Flexera Software\Compliance\ImportProcedures\ObjectAdapters*).

## 3

# The Inventory Adapter Studio Interface

IT Asset Management (Cloud)

The Inventory Adapter Studio has the following key areas in its interface:



Element	Purpose
Toolbar	<ul style="list-style-type: none"> <li>Creates new adapters or open existing ones.</li> <li>Manages database connections.</li> <li>Saves changes.</li> <li>Specifies the database connection to work on.</li> <li>Specifies the data size limits to apply when testing.</li> </ul>

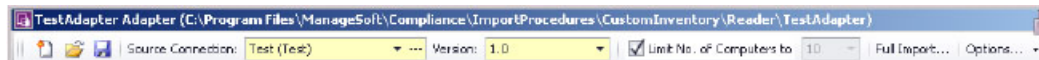
Element	Purpose
Step Explorer	Shows the steps in the currently open adapter. These open the edit panels on the right. <ul style="list-style-type: none"> <li>Import execution status will show as icons in this element.</li> <li>Bold steps in the templates show where user customization is required; other steps have been completed by Flexera.</li> <li>Steps may be added, deleted or have their execution order changed using the toolbar in the step explorer.</li> </ul>
Step editor	Shows: <ul style="list-style-type: none"> <li>The name of the step and its settings.</li> <li>The script in the step, with a Run button for testing.</li> <li>The logs, showing import results.</li> <li>The Result panel, which shows datasets from your SQL queries.</li> </ul>
Status bar	Shows import progress.

Each section is discussed in more detail in the following topics.

## Toolbar

IT Asset Management (Cloud)

The toolbar contains the following controls:



### New button

A button that launches the **Create New Adapter** dialog.

### Open button

A button that launches the **Open Existing Adapter** dialog. This allows browsing of all custom and factory-supplied adapters.

### Save button

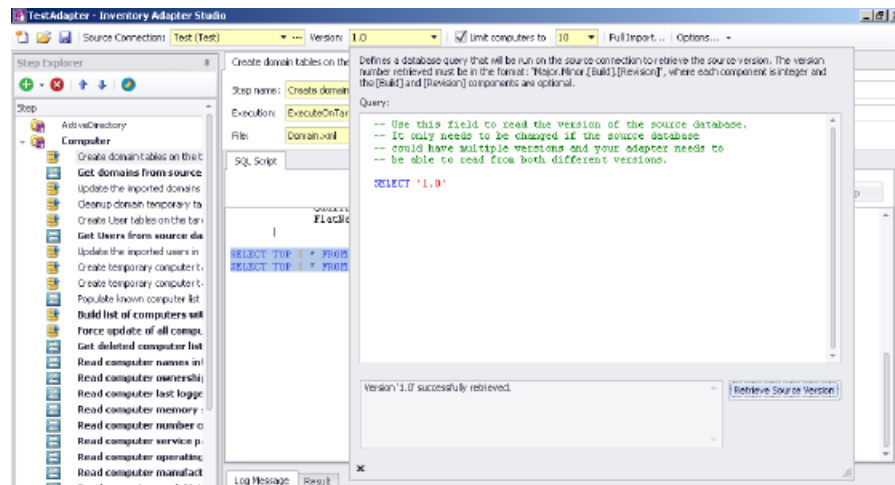
The Save button saves all files in the adapter that is being edited. This includes changes due to steps being moved in the Step Explorer, or versions being changed in the Version field on the toolbar.

## Source Connection control

The drop-down portion of this control allows selection of an existing database connection. New connections can be created and existing connections may be edited using the ‘...’ button, which launches the **Select Source Connection** dialog.

## Version control

This control shows the version of the currently selected source connection. This version is evaluated by executing a query against the source database. Clicking the drop-down button displays a dialog that allows you to change this query for an adapter. Clicking the **Retrieve Source Version** button will execute the query and show the results in the dialog.



Use of this control is particularly important if your adapter supports importing inventory data from multiple versions of the same system. This usually occurs as enterprises upgrade systems over time.

## Limit Number of Computers control

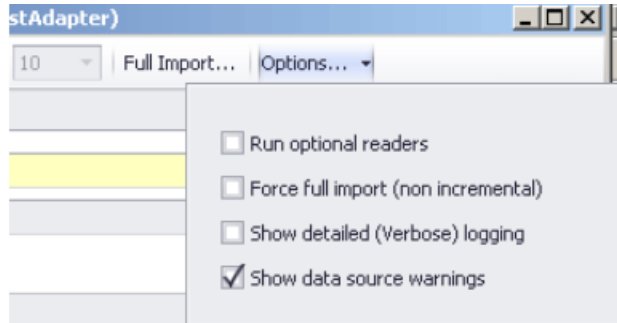
This control is checked and enabled by default for test connections. When set to a value, it limits the number of computers read by an adapter.

## Full Import button

This button launches the **Full Import** dialog, which allows you to execute your adapter end to end and check your results in IT Asset Management.

## Options

There are several options that apply to the Run buttons on the Edit panel. These control the way the Compliance Importer executes your adapter and correspond to command line arguments.



Options include:

- **Run optional readers** — the next run command will exercise steps marked with the **Step is optional** attribute.
- **Force full import** — the next run command includes steps marked with the **Runs on full import** attribute.



**Tip:** When the attribute values prevent the execution of the step in the next run, it is grayed out with a strike-through in the step explorer.

## Step Explorer

IT Asset Management (Cloud)

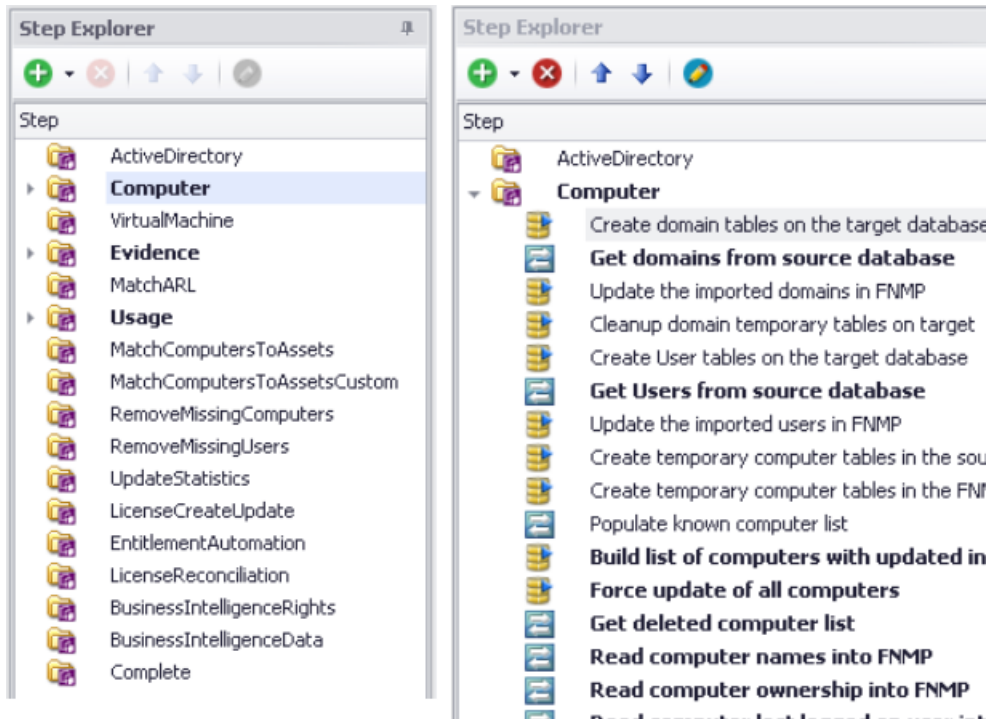
The step explorer shows the procedures in the Compliance Importer, and the steps within those procedures that will be executed for the current adapter.

The step explorer is a docking control and may be moved within the Adapter Studio interface. It also has a column that shows the file a step is being saved to.

Expanding one of the top level procedures shows all the steps within it.

Bold steps and procedures are parts of the template requiring your customization. There are queries in that part of the template that need to be replaced with code that applies to your data source.

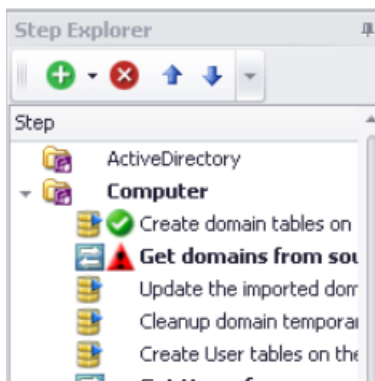
**Figure 7:** Collapsed (left) and expanded, with bold steps requiring customization. Custom SQL and data transfer steps visible.



The toolbar allows steps to be added, removed, edited and to change their execution order. There are two types of steps that may be added:

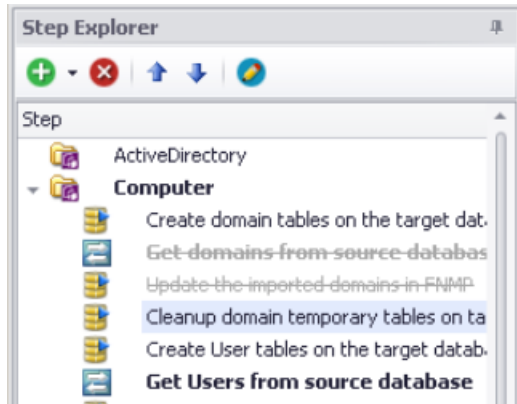
- Custom SQL steps have a yellow database icon and run commands on the source or target database.
- Data transfer steps have a blue icon with white arrows, and copy data from one database to another using a bulk transfer.

When testing an adapter, the step explorer also shows the status of the current run with green and red icons.



When the attribute values of a step will prevent it being executed for the version of the current connection, it will be grayed out with a strike-through in the step explorer.



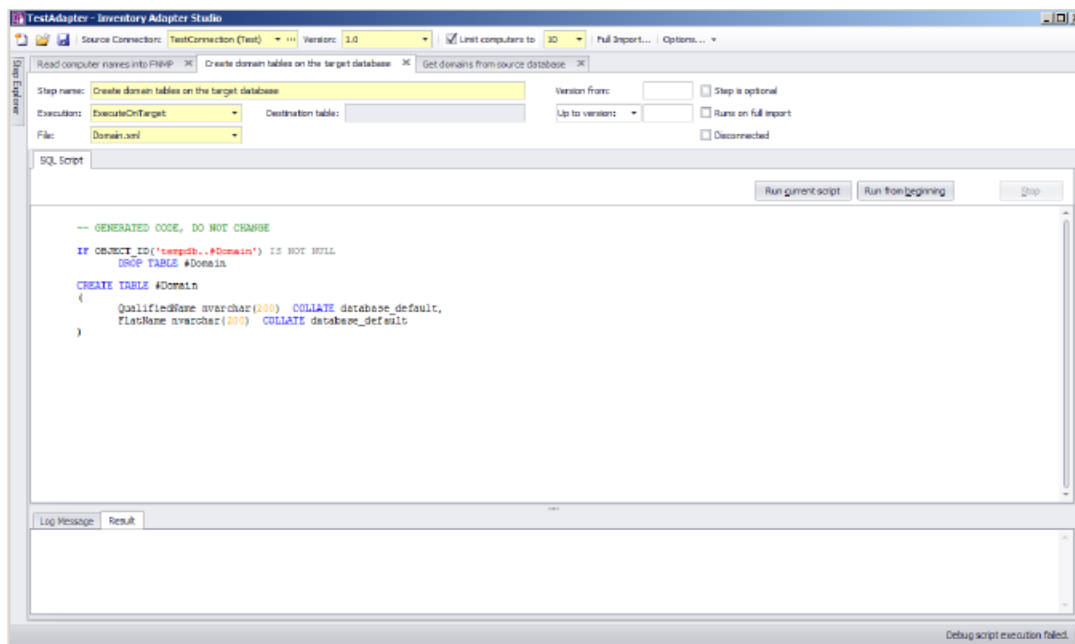


## Edit panel

IT Asset Management (Cloud)

The edit panel consists of three main areas:

- A properties section
- An SQL script
- A log message and result panel.



## Properties section

### Step name

This is the name of the step, and is shown on the step explorer as well as the top of the edit panel tab. It is best to choose a descriptive name to simplify future maintenance.

## Execution

In *disconnected mode* (when the inventory adapter is running on your inventory beacon), some values are available for your custom inventory adapters, while others may appear only in factory-supplied adapters, as shown in the table below. DO NOT use these values in your custom adapters, as these steps are blocked (for security reasons) in disconnected mode, and using them will cause your inventory import to fail.

Value	Step type	Notes
ExecuteOnSource	Custom SQL	The SQL script will run on the source database.
ExecuteOnTarget	Custom SQL	Factory use only. DO NOT modify. DO NOT use in custom inventory adapters.
GetVariableFromSource	Custom SQL	Allows you to declare, and get a value for, a custom SQL variable populated from the source database. This variable and its value are automatically in scope and available for all subsequent steps in this inventory adapter, alongside the standard variable <code>ComplianceConnectionID</code> .
SourceToObject	Data Transfer	The SQL script reads from the source database, and writes approved data directly to an intermediate package saved on the inventory beacon. A separate process uploads this intermediate package to the application server, where another scheduled process imports the data into the operations database. <code>SourceToObject</code> steps may only write to the predefined objects displayed in the Inventory Adapter Studio.
SourceToTarget	Data Transfer	Factory use only. DO NOT modify. DO NOT use in custom inventory adapters.
TargetToSource	Data Transfer	Factory use only. DO NOT modify. DO NOT use in custom inventory adapters.

### Destination table

Do not use for inventory adapters running on your inventory beacon (that is, in disconnected mode). Factory-supplied standard adapters may display values in this field. Do not alter such an adapter.

### File

This is the file name where the step is saved. In the templates it is specified for you, and has little impact on the execution of the adapter.

### Step is optional

An optional step will not be executed by default when the Compliance Importer is run. The only example of this in the factory-supplied adapters is when file information that does not match the Application Recognition Library is returned. Use this when the data returned by the step is not needed for critical tasks.

### Runs on full import

By default, adapters perform differential imports and only update computer records that have changed since the last import. The `#RelevantComputers` temporary table in the templates implements this feature. This flag is set for steps that are designed to override this functionality. The provided templates have one step with this option set, and causes all computers to be updated instead of the differential import.

### Version from

This is the first version field, and it causes the step to only execute when the **Version** field in the toolbar equals the specified value or higher. The version format must be in the 1.2.3.4 form.

### Up to version/Before version

This is the second version field, and it causes the step to only execute when the **Version** field in the toolbar is less than the specified value (less than or equal in the case of **Up to Version**). The version format must be in the 1.2.3.4 form.

## SQL Script section

### Run current script

This button executes the script for the current step. The SQL is executed and any result sets is displayed in the **Result** tab. You may execute multiple queries and return multiple result sets. You may execute parts of the script by selecting them before using the button.

Different step types execute on the following databases by default: for details, see the **Execution** field in the *Properties section* above.

There is a right-click menu available in the script edit panel that allows you to specify execution of the script on a different database.

### Run from beginning

This button executes the adapter from the beginning up to the current step. Once the current step is reached, execution is terminated, but the database connections are left open so you can run queries to inspect database values as desired. The results will be in the **Log Message** tab.

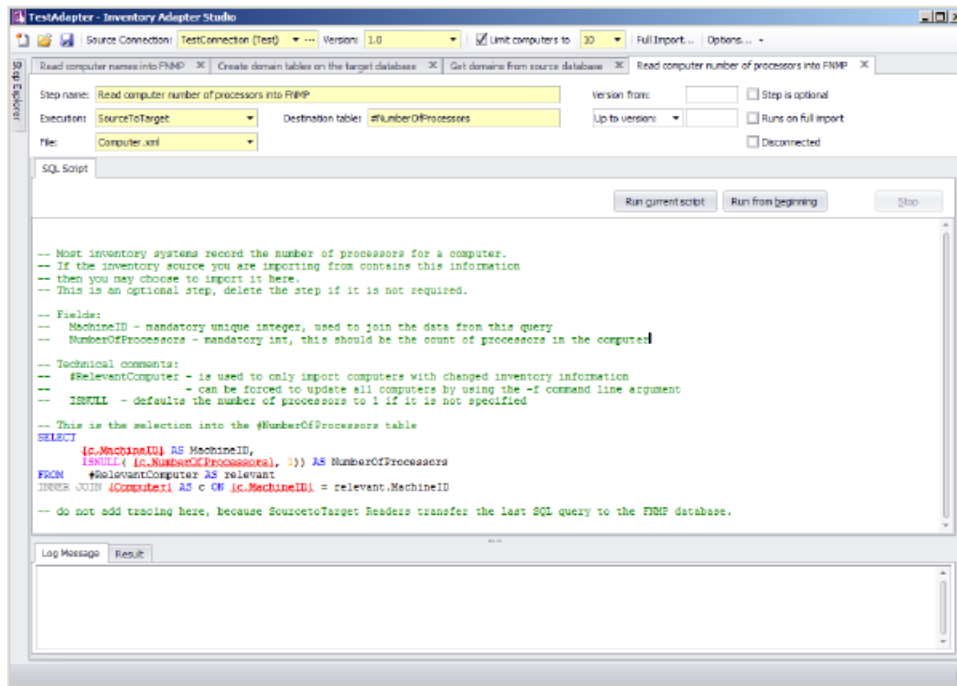
### Stop

This button stops an adapter that is in the process of running.

### SQL Script

This area contains the SQL scripts that make up the adapter. They are used for gathering data and transferring it to the IT Asset Management database. The template adapter provided performs all the differential updates required to move the data into the final IT Asset Management tables. This script tab provides SQL syntax highlighting, and a special red underlined highlight that shows where you need to modify queries with your own data values.

**Figure 8:** Red underlined text should be replaced with your own database column and table names.



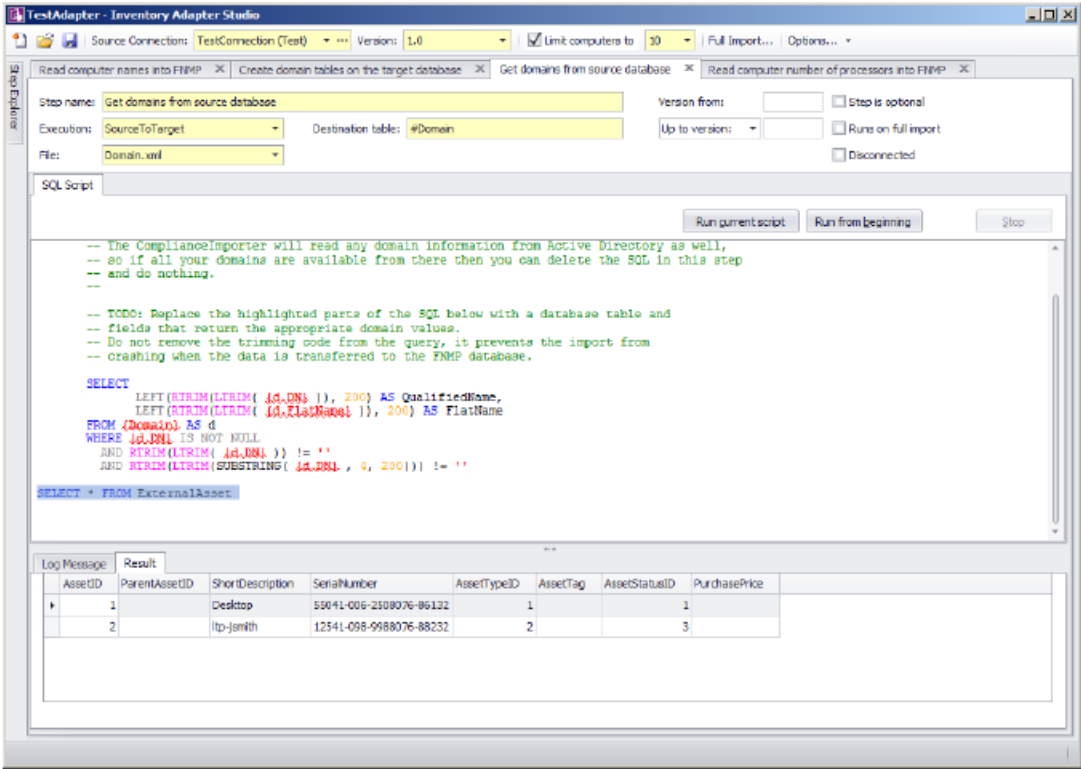
## Log Message and Result panel

### Log Message

The log message tab shows the results of executing the adapter. This is the same as the command line logging from the Compliance Importer. Look here for error messages. You can get more detail by setting the verbose tracing option on the toolbar.

### Result

This shows the results of highlighting a query in the SQL Script and pressing the **Run Current Script** button. Multiple results sets can be displayed.



## 4

# Installing Inventory Adapter Studio

IT Asset Management (Cloud)

On your inventory beacon, no separate installation is required. Inventory Adapter Studio is installed as part of the installation process for the inventory beacon.

The Inventory Adapter Studio executable: InventoryAdapterStudio.exe

Default location (beacon): C:\Program Files (x86)\Flexera Software\Inventory Beacon\DotNet\bin

Template file storage: C:\ProgramData\Flexera Software\Compliance\ImportProcedures\, followed by:

- AdapterStudioTemplates for templates downloaded from the central application server
- CustomInventory for use in an on-premises installation (not used for IT Asset Management as a cloud service)
- Inventory for standard adapters supplied with the product. These implement standard integration with other products. You must not edit these in any way, or they will cease to operate and cause all attempts to import using them to fail completely.
- ObjectAdapters, with a subfolder Reader for adapters customized on an inventory beacon to run in disconnected mode.

## Starting Inventory Adapter Studio

After installation, you can find a shortcut to Inventory Adapter Studio in the Windows Start menu (**All Programs > Flexera > Inventory Adapter Studio**).

## 5

# Understanding Inventory Adapters

IT Asset Management (Cloud)

Inventory adapters exist to extract data from one database (the inventory source), transform it as required, and write it into the destination (or target) database.

To understand the work in creating or modifying an adapter, it is helpful to know a little about:

- The Compliance Importer, considered as the framework which runs adapters
- The resulting structural requirements for an inventory adapter
- What is provided in templates to help you quickly build inventory adapters
- The object model (legal database objects and their properties) for saving content into the destination database when your inventory adapter is running on your inventory beacon in disconnected mode (see [Inventory Adapter Object Model](#)).

## The Architecture of Compliance Importer

IT Asset Management (Cloud)

Compliance Importer is the framework within which inventory adapters function, and therefore dictates the requirements for each adapter.

The Compliance Importer is the software that executes inventory adapters to import data into IT Asset Management. It is a generic data import framework, but specific procedures are provided to import inventory data from source databases.

Once data is imported, it is matched to existing information in IT Asset Management, the Application Recognition Library is applied, and license compliance is calculated.

The overall model of the Compliance Importer is as follows:

- A set of procedures is defined for the import process. Procedures are grouped by their purposes as Readers, Writers and Export procedures.
- Tables are provided for intermediate storage and workspace for data used by each procedure.
- The main purpose of readers is to read data from a source database, and use it to populate the staging tables in the

operations database. When operating in disconnected mode on an inventory beacon (as is always the case with IT Asset Management as a cloud service), the readers work in two stages: writing the gathered data to an intermediate package on the inventory beacon for later upload to the central application server; and subsequently loading the intermediate package data into the staging tables.

- Readers may also perform operations on data in the source database, usually to prepare data before returning results.
- Writers update the operations database, using the data in the staging tables to determine the changes to make.

Understanding the function of the Reader procedures is especially important to preparing inventory adapters.

## Structure of an Inventory Adapter

### IT Asset Management (Cloud)

Each inventory adapter is a set of reader instructions for the compliance importer. The permitted structure depends on the origin and operational mode for this adapter.

Each adapter runs in one of (up to) three modes:

- "Connected mode", where the adapter has simultaneous access to both the source and target databases (for example, when the source database is accessible from the server running the operations database for IT Asset Management).



**Note:** For security reasons, connected mode is not available when you are using IT Asset Management as a cloud service.

- "Disconnected mode", where you need to install an inventory beacon, either because the source database and target database are on separate networks, or because you are using IT Asset Management as a cloud service. For more information see [Disconnected Mode](#).
- "Disconnected mode (Tier 1)", where the same operational conditions apply with the adapter running on an inventory beacon, but because the adapter is factory-supplied, security provisions take a different form, discussed below.



The operations that are available when creating a custom adapter depend on the mode in which it runs.



**Note:** Adapters engineered by Flexera and provided as standard functionality (sometimes called Tier 1 adapters) may include operations of all types. However, when working on an inventory beacon, you must not edit any Tier 1 adapters. For security reasons, a modified Tier 1 adapter in disconnected mode is automatically failed, and cannot import any inventory.



**Table 11:** Availability of all adapter operation types

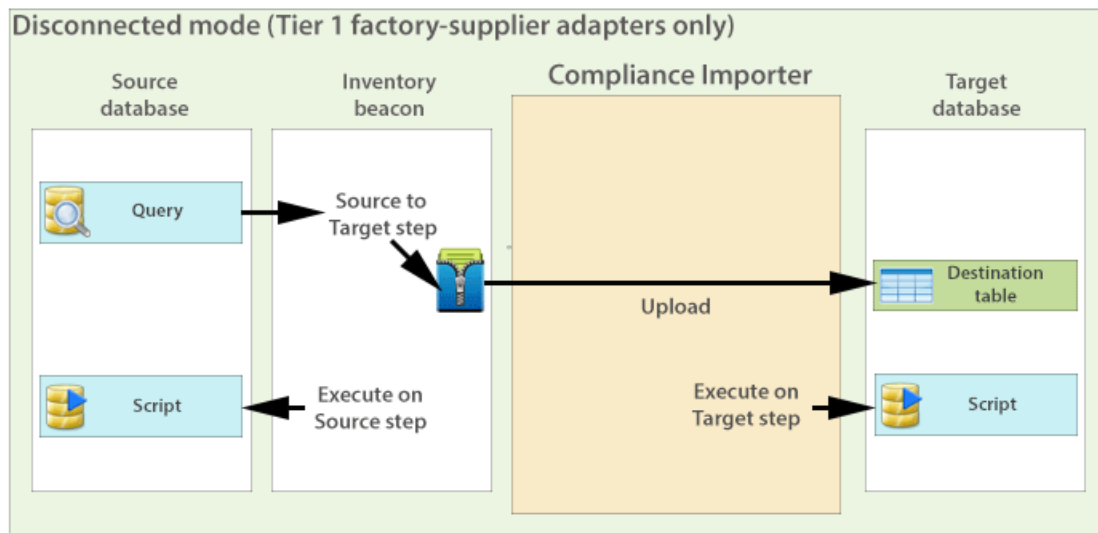
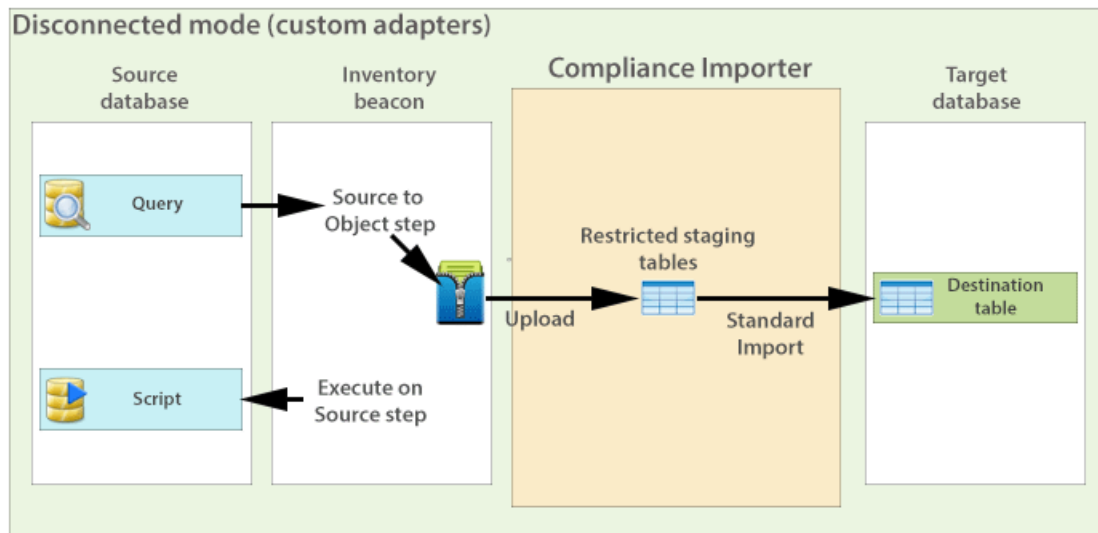
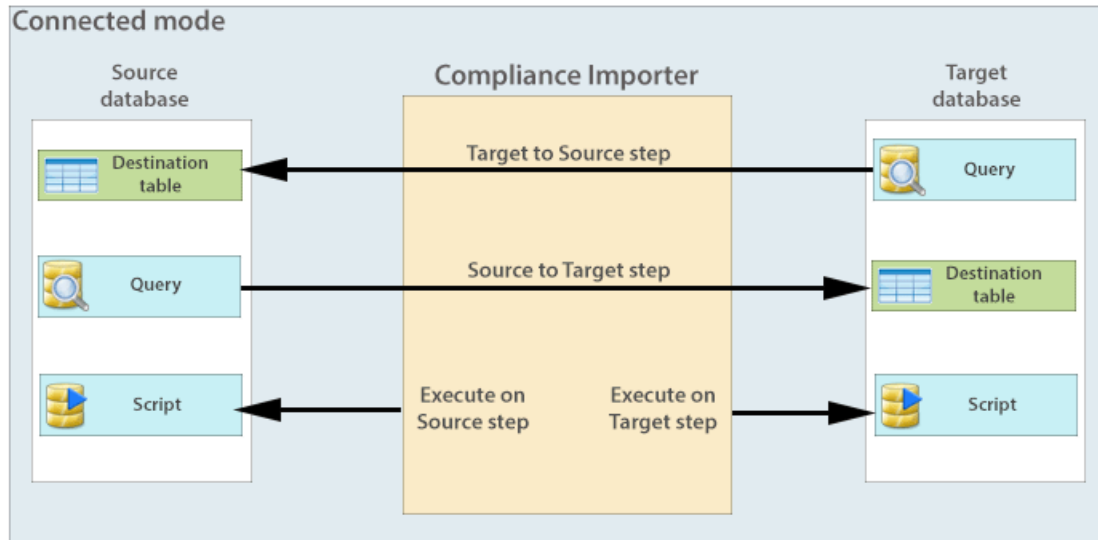
Type of operation	Description	Available in modes
Target to source	Executes a database query on the IT Asset Management database and copies the result to a table in the source database. This data is to provide context for a subsequent query on the source database.	Connected
Source to target	<p>Executes a database query on the source database and copies the result to a table in the IT Asset Management database.</p> <p> <b>Restriction:</b> For fast transfers, the Compliance Importer uses SQL bulk copy operations to move data from one database to another. This means that the query on the source and the table on the target must match exactly in column order and data type.</p> <p>For disconnected (tier 1) operations, a source to target step is modified so that data is saved to intermediate packages before upload.</p>	Connected Disconnected (Tier 1)
Source to object	<p>Replaces 'source to target' for use in disconnected mode with custom adapters. Instead of writing source data directly to the target database, it is written into an intermediate package format and saved on the inventory beacon. The intermediate packages are uploaded asynchronously to the cloud server, and processed (by default overnight) for import into the operations database.</p> <p> <b>Restriction:</b> Since you cannot modify the internal processing, the data in the intermediate packages must map correctly to a standard set of objects and their attributes (see <a href="#">Inventory Adapter Object Model</a>).</p>	Disconnected
Execute on source	Executes an SQL script on the source database.	Connected Disconnected (switchable for disconnected <i>only</i> ) Disconnected (Tier 1)
Execute on target	Executes an SQL script on the IT Asset Management database.	Connected Disconnected (Tier 1)

The three architectures are shown in the diagrams below. From a security perspective, the distinction between the modes is:

- Connected mode applies only for on-premises installations where the security of both databases is entirely in your control.
- Disconnected mode for custom adapters protects the central operations database in the cloud service by disallowing any custom SQL on the target side. This also limits the permissible imports to the standard set of objects and

attributes available in the Inventory Adapter Studio running on an inventory beacon. You can also find an XML file defining those objects and attributes on your inventory beacon at `C:\ProgramData\Flexera Software\Compliance\ImportProcedures\ObjectAdapters\InventoryObjectModel.xml`.

- In disconnected mode, Tier 1 (factory-supplied) adapters are able to use factory-approved custom SQL on the target side. Security is provided by disallowing the slightest revision of any kind to these adapters. If you change anything on Tier 1 inventory adapters for disconnected mode, they will not run, and importing your inventory will completely fail.



# Structure of Templates for Inventory Adapters

IT Asset Management (Cloud)

Templates are provided for inventory adapters, to speed your development effort.

The adapter templates shipped with the Inventory Adapter Studio use the adapter structure as follows:

- Temporary database tables are set up on the source and IT Asset Management databases.
- Sample queries are written in Source to Target steps. These write to the temporary tables already created.
- To make each query as easy to write as possible, the minimum number of columns is sent in each query. This also documents the minimum requirements for importing the inventory source.
- Areas of the query that require change are enclosed with curly brackets, colored red, and underlined: {Replace this text}.
- As many of the other steps as possible are already completed. They rely on the fact that data has been transferred in the Source to Target steps.

Each step in the templates has comments describing the updates that need to occur. Optional fields are identified, and the adapter will still work if the optional fields are not provided.

At the end of each procedure there is a lot of provided code that performs a differential update into the Imported database tables in the IT Asset Management database. It is recommended that you do not change this code for your adapter. There may be special cases where this is required, but an error will prevent any data from being imported into IT Asset Management.

## 6

# Creating a New Adapter

IT Asset Management (Cloud)

Once completed and published to IT Asset Management, each adapter may be used to import from multiple databases that have the same structure. In other words, a number of similar connections (to the databases) may reuse the same adapter.



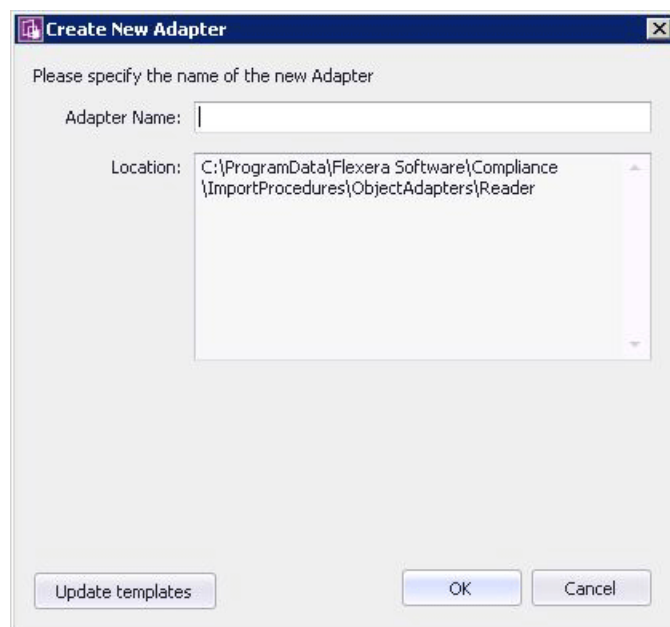
**Note:** There is a separate process for editing an existing adapter.



**To create a new adapter:**

1. Click the New icon in the toolbar.

The **Create New Adapter** dialog opens.



**Tip:** If the Inventory Adapter Studio cannot locate downloaded templates, it displays a warning message in this

---

*dialog. You can download the latest templates using the **Update templates** button (if necessary, first re-establishing your link to the application server in the inventory beacon interface).*

2. Specify the name for your adapter. It is best practice to choose a name similar to the data source you plan to import from.

You may not change the directory the new adapter is saved in. This is because IT Asset Management uses specific directories to separate out-of-the-box and custom adapters. For example, if you are creating this adapter on an inventory beacon, the default path is C:\ProgramData\Flexera Software\Compliance\ImportProcedures\ObjectAdapters.

Your adapter appears, pre-populated with samples for each available object. You may remove the examples you do not need, and complete the ones required for your adapter.



**Tip:** *The templates in the new adapter depend on the context in which you are working. For example, if you created this adapter on an inventory beacon, only **Source to Object** steps and **Execute on Source** steps are available.*

After you create a new adapter, you must create a new database connection that matches the type of the adapter.

## 7

# Editing an Existing Adapter or Template

IT Asset Management (Cloud)

You may edit an existing, custom adapter that was created in your enterprise. In disconnected mode (that is, using the cloud service solution), do not attempt to edit any factory-supplied (Tier 1) adapters. If you edit any part of a Tier 1 adapter, it ceases to operate.



---

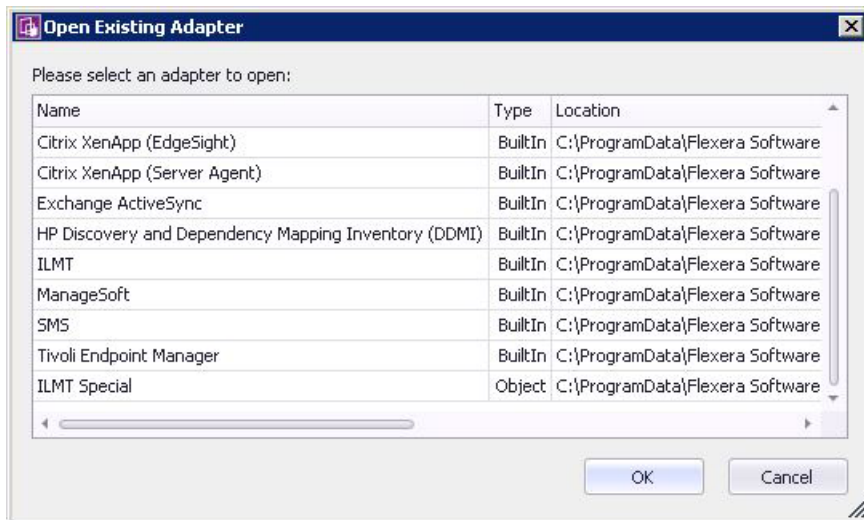
***To edit an existing adapter or template:***

1. Click the Open icon in the toolbar.

The **Open Existing Adapter** dialog appears. The meaning of the **Type** column is as follows:

- Adapters of type **BuiltIn** are factory-supplied adapters that implement standard connectivity. You may read but not edit these adapters on an inventory beacon.
- Adapters of type **Object** are those which you have previously edited.

On an inventory beacon, you can only open **Object** adapters, stored (by default) in C:\ProgramData\Flexera Software\Compliance\ImportProcedures\ObjectAdapters.



2. Select the desired custom adapter from the list, and click **OK**.

Details of the adapter appear in the **Step Explorer** and edit panel.



## 8

# To Create a Source Connection

## IT Asset Management (Cloud)

You must know either the server name or its IP address (together with database instance name, if any) to type in during the following process. (There is no browse facility to find the server.)

This process establishes the link between the adapter and the source inventory database. This connection may be used for both reading inventory, and also writing data (if additional context is required for good information gathering).

This process assumes that you already have the appropriate adapter open in the Step Explorer and edit panel. This process creates a test connection, because new adapters are not ready for production. Test connections are not imported by the Compliance Importer in its normal operations. Data from this connection is imported only after you publish the completed and tested connector.

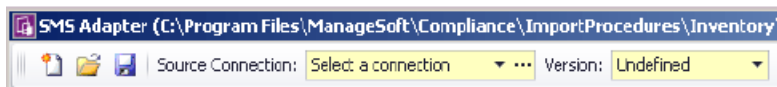


**Note:** Creating a new test connection adds this connection to the list available in the inventory beacon interface.

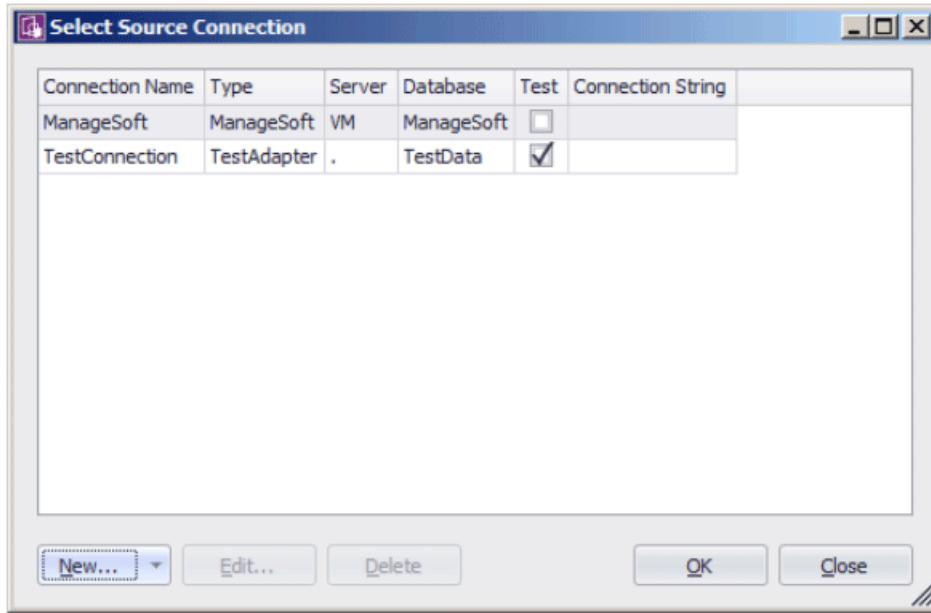


### To create a source connection:

1. In the toolbar, on right-hand end of the **Source Connection** control, click the [...] ellipsis button.



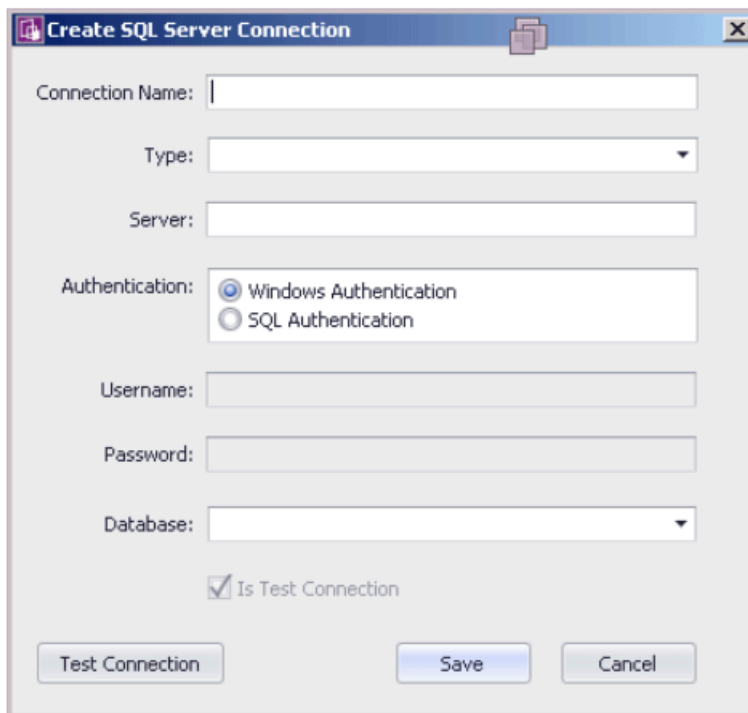
The **Select Source Connection** dialog opens.



**Tip:** You must create a new test connection for this adapter. You may not reuse an existing connection for this adapter. (The existing connections are those previous declared on your inventory beacon, and editing or deleting from this dialog affects the connections for your inventory adapter.) Only test connections, those displaying a check mark in the **Test** column, may be created from the Inventory Adapter Studio.

2. Click **New....**

The **Create SQL Server Connection** dialog opens. (If not, see note below.)



3. Complete the details:

- a. Provide a descriptive name in the **Connection Name** field, perhaps referencing the name of the adapter using this connection.
- b. From the **Type** pull-down, select the *name of the adapter* you are editing (or just created). Scroll down the list to find your adapter's name. Every connection must be tightly coupled to an adapter through this **Type** setting.
- c. In the **Server** field, type the server name or IP address. If the server hosts multiple SQL instances, you may append the appropriate instance name after a backslash. For example, with an instance called `Inst1232`, you could enter `10.200.3.102\Inst1232`.

- d. In the **Authentication** field, select the authentication method:

**Windows Authentication** — Uses standard Windows authentication to access the server. The credentials of the operator currently logged on will be used to access the SQL Server database. Any operators that require access to the database must be added to the security groups that already have access to the database.

**SQL Authentication** — If you select this option, you must then specify an account and password already known to SQL Server. This account's credentials will be used to access the source database, regardless of the operator or account running the adapter.

- e. If you selected **SQL Authentication**, complete the **Username** and **Password** fields with the account name and password to be used during SQL authentication.
- f. In the **Database** field, type the name of the database, or use the pull-down list to select from database names automatically detected on your specified server.
- g. Click **Test Connection**. If the compliance server can successfully connect to the nominated database using the server and authentication details supplied, a Database connection succeeded message displays. Click **OK** to close the message. Click **Save** to complete the addition of these connection details.

You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.



**Note:** There is a second type of connection that may be created using the **New...** button on the **Select Source Connection** dialog. This is used for database connections to non-Microsoft databases. The difference is that a full database connection string must be entered manually for this connection.

## 9

# Overview: Process for Developing an Inventory Adapter

IT Asset Management (Cloud)

Here is your mental roadmap through the development process for inventory adapters, with links to the details.



## **The development process for inventory adapters:**

1. Create a new adapter, normally pre-populated with an appropriate set of steps to gather and process data ([Creating a New Adapter](#)).
2. Specify a new connection to a data source. Initially this is a test connection during your development phase. (See [To Create a Source Connection](#))
3. Use the **Step Explorer** to:
  - Add a new step to one of the grouping folders (see [Adding a New Step to an Inventory Adapter](#))
  - Remove any steps provided automatically that are not required in your inventory adapter (see [Removing a Step from an Inventory Adapter](#))
  - Change the execution order of steps within your inventory adapter (see [Reordering Steps in an Inventory Adapter](#))
  - Update the details included within an individual step.



**Note:** You cannot add, delete, or reorder folders in the **Step Explorer**. These are optimized for the order of insertion into the operations database. You may only modify or reorder the steps within a given folder.

4. Test each step in your adapter as you develop it (see [Tips for Editing an Adapter](#) and [Testing an Adapter](#)). Cycle through until you have created and tested all the steps needed to complete your inventory adapter.
5. Run the entire adapter, and validate that the collected data is as you expect (again, see [Tips for Editing an Adapter](#)). On an inventory beacon, validation means unzipping the intermediate package and examining the XML file it contains. Look for your created intermediate package in C:\Program Data\Flexera Software\Beacon\IntermediateData.

- Put the completed and test inventory adapter into production (see [Publishing Your Adapter](#)).

## Adding a New Step to an Inventory Adapter

IT Asset Management (Cloud)

You may add customized steps to your inventory adapter, choosing its position in the appropriate folder. (Remember that you cannot edit a factory-supplied inventory adapter.)



### To add a new step to an inventory adapter:

- In the **Step Explorer**, select one of the following:

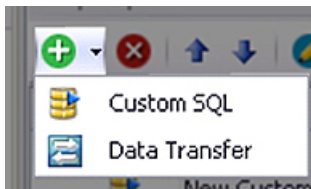
- An empty folder (this has no expander icon to the left of the folder name)
- In an expanded folder of steps, the current step *after* which you want to insert a new step.



**Tip:** To insert a new step as the first in a folder containing existing steps, insert it as the second step and then move it up the list with the up-arrow icon above the list of steps.

- Use the down arrow next to the add icon to expand the choices, and click either:

- Custom SQL** to write any SQL scripting that runs on the source database. Use these steps to massage data within the database, such as scrubbing data into a temporary table within the same database.
- Data Transfer** to move data from one database to another. These steps can include custom SQL statements to select the data for transfer.



The new step appears in the **Step Explorer**, and its details appear in the editing pane on the right, potentially in a new tab (if you are already editing other steps).

- In the editing pane, change the default value in the **Step name** field to something meaningful that will assist with future maintenance of this adapter.
- Complete the details of your new step in the editing pane. For more about the fields in the editing pane and the permitted values, see [Edit panel](#).

## Removing a Step from an Inventory Adapter

IT Asset Management (Cloud)

Templates for custom inventory adapters may include sample steps that you don't require. When planning removal of any steps from your custom inventory adapter, remember to consider the potential impact on subsequent steps. There

is no undo available for deleting a step.

**To remove a step from an inventory adapter:**

1. In the Step Explorer, select the step you want to remove. If this step has previously been operational, review its contents to check for possible flow-on effects from its removal.



**Tip:** Do not select a folder, as you cannot delete the folders. You may remove all the steps contained within a folder if need be; but the folders must remain. The delete icon is disabled when you select a folder.

2. Click the delete icon (✖) at the top of the **Step Explorer**.

A confirmation dialog appears. Remember that there is no undo available for this delete action.

3. Click **Yes** to proceed with the removal of the step (or **No** to reconsider).

## Reordering Steps in an Inventory Adapter

IT Asset Management (Cloud)

An inventory adapter executes in the order shown in the Step Explorer, from top to bottom. Therefore to change the execution order of the step in your adapter, simply change the display order.

**To reorder the steps in an inventory adapter:**

1. In the **Step Explorer**, select the step you want to move.
2. Use the up and down icons at the top of the **Step Explorer** to move the step within its folder.



**Tip:** You cannot move a step outside the boundaries of its folder; and you cannot reorder the folders themselves.

Remember to save your changes with the save icon in the toolbar of the Inventory Adapter Studio.

# 10

## Disconnected Mode

### IT Asset Management (Cloud)

Whenever there is a network discontinuity between the source and target databases, your inventory adapter must function in disconnected mode.

Using IT Asset Management as a cloud service (software as a service, or SaaS), you have access only to your inventory beacon(s), with no direct access to the central operations database. This means your custom inventory adapters always run in *disconnected mode*.

However, factory-supplied inventory adapters may have some steps that are designed for on-premises implementations that may access a central operations database. Such steps, designed for connected mode, cannot run in disconnected mode (and are automatically skipped when the inventory adapter runs on an inventory beacon). For that reason, factory-supplied adapters may have alternate steps that run *only* when the inventory adapter runs on your inventory beacon, in disconnected mode. Such steps are differentiated by having the **Disconnected** check box set.

# 11

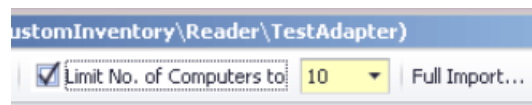
## Tips for Editing an Adapter

IT Asset Management (Cloud)

The following principles are helpful when you are developing and testing your inventory adapter. If you need information about the fields in the editing pane and the permitted values, see [Edit panel](#).

### Minimize processing times

During development, use the setting to limit the number of computers for test imports. This will potentially save hours of processing while testing your adapter. The control applies a filter to the number of computers that the adapter reads from the source database, allowing validation of your work without requiring that all the data is read and processed.

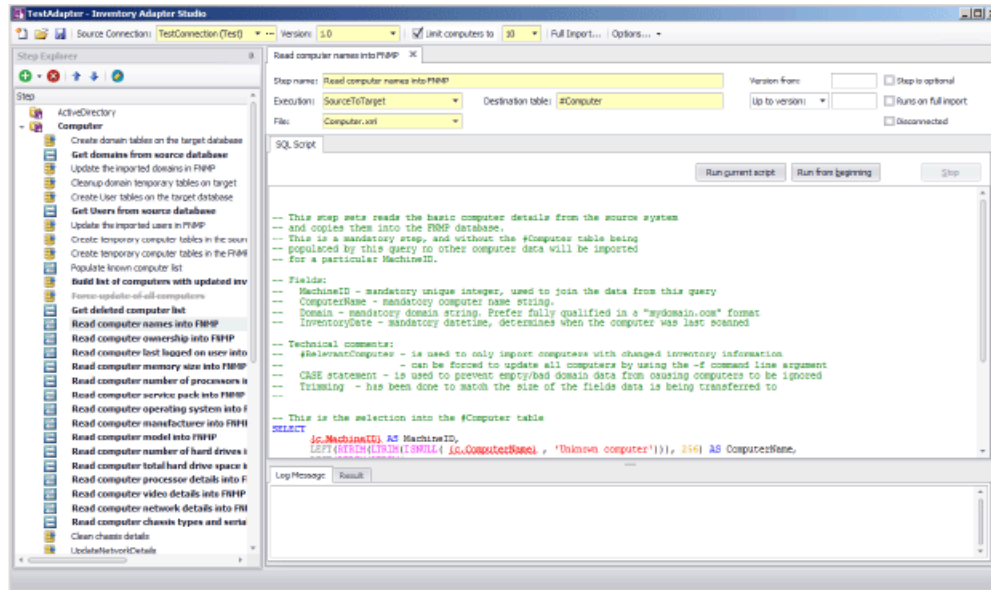


### Start with a template

Start with one of the supplied templates and customize it to suit your data source. Focus on the areas needing change:

- In the Step Explorer, each step that requires editing is shown in bold text. The bold is automatically removed when all areas requiring change have been modified.
- When you have selected a step so that its details are shown in the edit panel, the individual edits required are shown within curly braces, underlined, and in red.
- Every step that needs editing has extensive comments on the data structures and requirements provided in the step details. Following these guidelines will provide the quickest path to a working adapter.





## Test each SQL step

As you modify each step, you can test the SQL and inspect the data set it produces, by highlighting the section of your SQL to test, and clicking the **Run current script** button. The result is shown in the **Result** tab below.

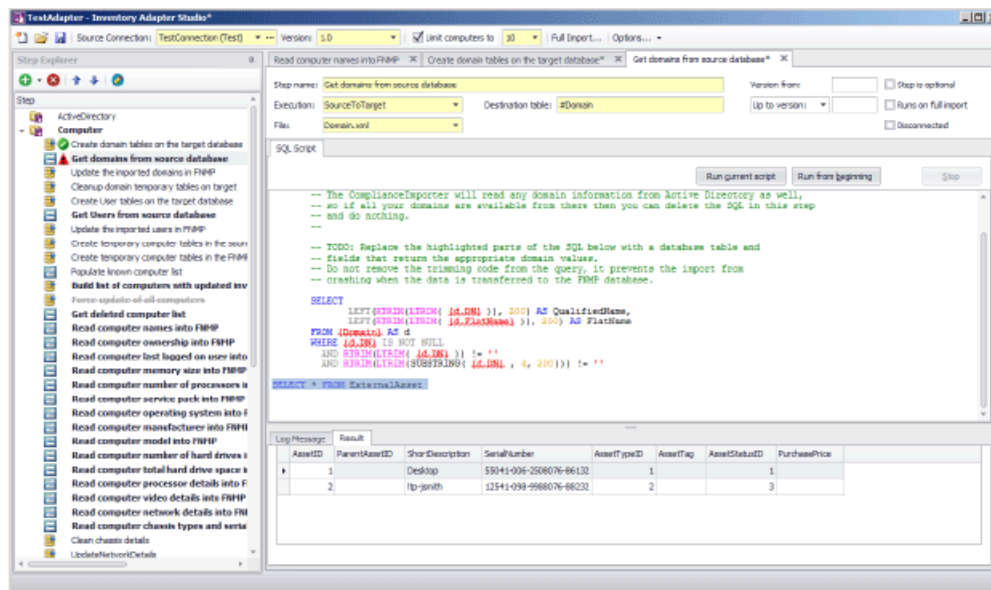


**Tip:** The **Run current script** button will run the entire step if there is no selection in the SQL script.



**Note:** If the selection (or, when there is no selection, the step) includes any red, underlined text that still requires customization, this produces a syntax error when run.

**Figure 9:** This step still requires customization, but the customizable text is not included in the selection, which can safely be run to inspect the results of the individual statement.



## Test progressively

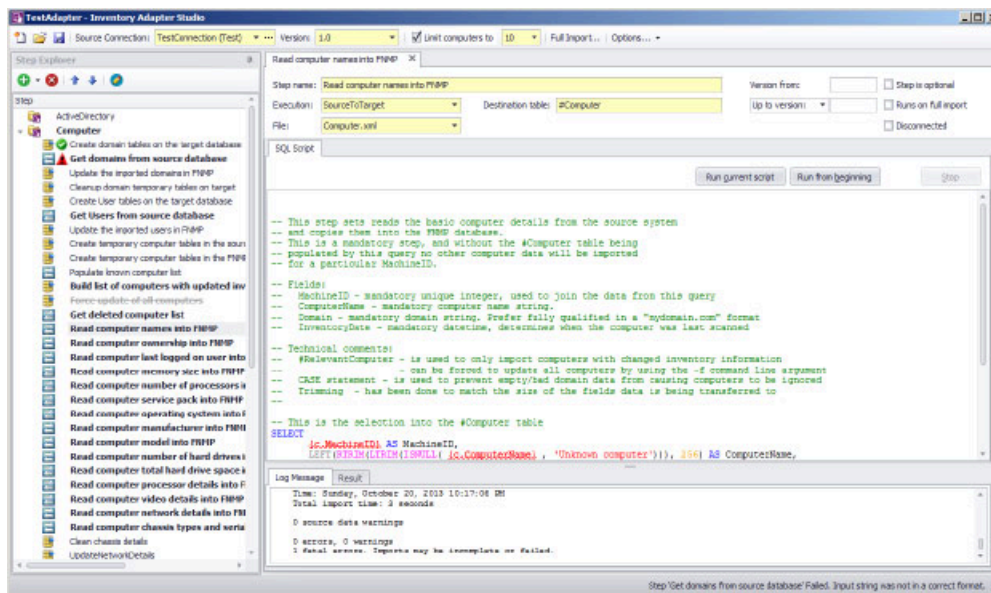
As you complete each step, click the **Run from beginning** button. This executes all the steps in the adapter from the start up to and including the one currently being edited. Errors are shown in the **Log Message** tab. In the Step Explorer, steps which succeed are marked with a check mark (tick) and those that fail show a red warning symbol.



**Tip:** The adapter is executed in the appropriate mode. For example, when you are developing the adapter on an inventory beacon, it must run in *disconnected mode*, meaning that all *Target* to *Source* steps are skipped, and all steps with the **Disconnected** check box set are exercised.



**Note:** If any steps between the start and your present position are still bold (require editing to customize them), they will produce a syntax error on execution.

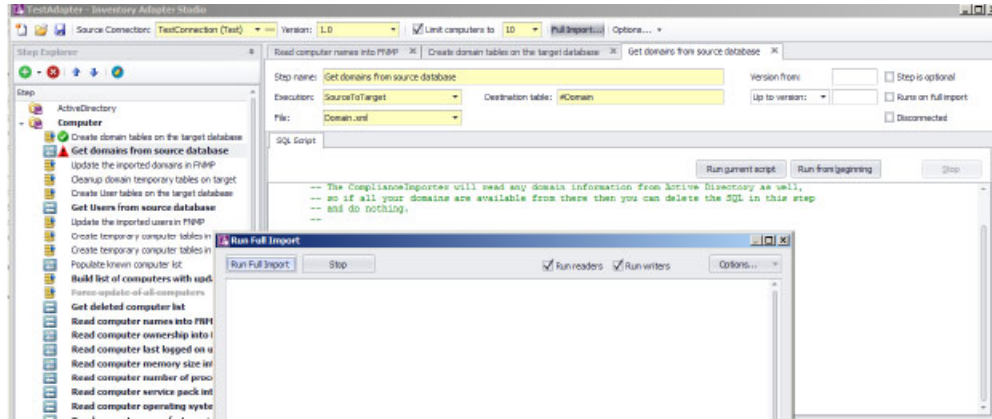


## Final test

Finally, when you are ready to run an end-to-end test of your adapter, use the **Full Import** toolbar button.

Whereas your earlier testing displayed results in the Inventory Adapter Studio, a full import also writes the gathered inventory into a zipped archive ready for upload to the operations database. Look for your created intermediate package in C:\Program Data\Flexera Software\Beacon\IntermediateData. You can unzip the archive package, and examine the XML file it contains to ensure it contains all the required data.

When you are working on an inventory beacon (*disconnected mode*), in the **Run Full Import** dialog, the **Run readers** check box is set, and the **Run writers** check box is cleared, and both are disabled. This indicates that your full import gathers the data from your source databases to the intermediate package; but that the stage of writing data to the operations database is completely asynchronous, and cannot be controlled from the beacon. As long as the connection for this adapter is in test mode, the inventory data it gathers is never visible in the IT Asset Management compliance console.



# 12

## To Save an Adapter

IT Asset Management (Cloud)

Save early, save often.



### ***To save an adapter:***

1. Your adapter is saved every time you do any one of the following:

- a. Click the Save icon in the toolbar.
- b. Use the Ctrl-S keyboard shortcut.
- c. Click the **Run from beginning** button to test the adapter.

All the files are saved first, because running the adapter uses the Compliance Importer to read the files from disk. Anything not saved is not tested by the **Run from beginning** button.

# 13

## Testing an Adapter

IT Asset Management (Cloud)

Completely validating the operation of an inventory adapter requires checking two stages: the reader and the writers.

Once you have finished updating all the steps that require customization in your new adapter (so that nothing is left shown bold in the Step Explorer), it is time to test that it functions correctly. This can proceed in two stages:

- Executing the adapter from the **Run from beginning** button performs the first half of the import, reading data and displaying results in the **Results** tab of the Inventory Adapter Studio.
- To validate the second stage, migrating the data from the staging tables into the operations database so that the data is visible in the compliance console, uploading the collected data to the IT Asset Management cloud database so that it becomes visible in the compliance browser, you need to perform a full import.

These two stages are described in the following tasks.

### To Run a Full Import

IT Asset Management (Cloud)

Only a full import causes the Inventory Adapter Studio to work through the entire process for inventory import. You should attempt a full import only after every step in your adapter has been individually tested. Keep in mind that importing large scale data incorrectly can create a significant workload to back out all the incorrect data! Consider using the **Limit computers to** filter for the early testing, even of the full import process.



#### **To run a full import:**

1. Inspect the Step Explorer to validate that no step names are displayed in bold text (still awaiting customization).

If there are bold steps in the Explorer, you cannot run a full import. Instead, circle back and complete the required customization.

2. In the toolbar, click **Full Import....**

The **Run Full Import** dialog appears.

3. Click **Run Full Import** within the dialog.

The logging from the Compliance Importer is echoed in the dialog.

When the full import is finished, inventory gathered by your adapter has been written in a zip archive that packages version information, a package manifest, and the inventory data in a zip file awaiting upload to the operations database.

## To Diagnose Readers for Your Adapter

IT Asset Management (Cloud)

Because the inventory adapter is running on your inventory beacon, you can only inspect the uploadable package to validate reader operations.

Follow this procedure using a plain text or XML editor of your choice on the inventory beacon.



### **To diagnose readers for your adapter:**

1. Locate the uploadable package that the adapter has saved on the inventory beacon.

By default, this is located in `C:\ProgramData\Flexera Software\Beacon\IntermediateData` (if this location is not in use, check the registry key `HKLM\SOFTWARE\ManageSoft Corp\ManageSoft\Beacon\CurrentVersion\BaseDirectory`, and append `IntermediateData` to the value found there). In test mode, your adapter creates a folder here named for your connection name. (Once this adapter is in production, this folder is replaced by a zipped archive named with the connection name plus 14 digits representing the date/time when the file was zipped.)

2. Examine the files in the folder. Each contains:

- A version identifier - you cannot alter this by modifying your adapter (and should never edit this file in production).
- A package manifest `package.xml` — likewise, not configurable through your adapter, and do not edit in production.
- An XML file containing your inventory information.

3. Review the contents of this XML inventory file to validate that the dataset is as expected.

Within this file the `meta` element defines the destination database object, and the attributes represented in each logical column of data. Thereafter, each row contains the data in the matching columns.

## Diagnosing Writers for Your Adapter

IT Asset Management (Cloud)

With the software as a service (SaaS) solution, you can inspect results in the compliance browser after the next upload and compliance calculation.

The simplest way to validate that your data is being imported all the way into the operations database is to inspect the results in the web interface.

**To diagnose writers for your adapter:**

1. To ensure the archived package is uploaded to the operations server, publish your adapter (by turning off the **Test** setting), and run a **Full Import**. Wait until the resulting upload is completed.

Each full import triggers an upload, and there is a 'catch-up' scheduled upload to retry any failures scheduled overnight.

2. To ensure that the uploaded data is processed from the staging tables into the operations database, do either of the following:
  - Wait until the next scheduled inventory import. By default, the inventory import and recalculation is triggered overnight.
  - Trigger an inventory import/recalculation now. Be aware that this processes all current data, and is not restricted to your new inventory import. As a result, it may take some time (hours, for a large computer estate). Use the following steps:
    - a. In your compliance browser, go to the **Reconcile** page (**Data Collection > Process Data > Reconcile**).



**Restriction:** You must be in a role with administrator rights to be able to include your new inventory import in the reconciliation you run manually.

- b. Select the **Update inventory for reconciliation** check box.

This setting ensures that uploaded content is incorporated into the operations database and used for the compliance recalculation. Wait for the import and reconciliation to succeed (monitor the **Last successful reconcile** display on the right of the title bar, refreshing your browser page as necessary).

3. When the reconciliation is complete, examine your imported information, for example in the following locations:
  - The **All Inventory** page (**Inventory > Inventory > All Inventory**) shows you all the computers that are being imported, as well as the hardware properties that you have set (remember to check the column chooser). Consider filtering by **Created** date to isolate your new imports.
  - The **All IT Asset Users** page (**Organization > All IT Asset Users**) shows all the users you have imported and the attributes that have been set.
  - The **All Evidence** page (**Applications & Evidence > Evidence > All Evidence**) has separate tabs to show the installation evidence, file evidence, and access evidence (for application virtualization) that was imported.
  - The **All Applications** page (**Applications & Evidence > Applications > All Applications**) shows all the application installations identified as a result of the evidence import. If your inventory revealed an application for the first time, check for a status of Unmanaged.



**Note:** If imported evidence did not match any existing application rules, the application does not show in any application list. It will appear only when the Application Recognition Library is updated with new rules incorporating your new evidence.

# 14

## Publishing Your Adapter

### IT Asset Management (Cloud)

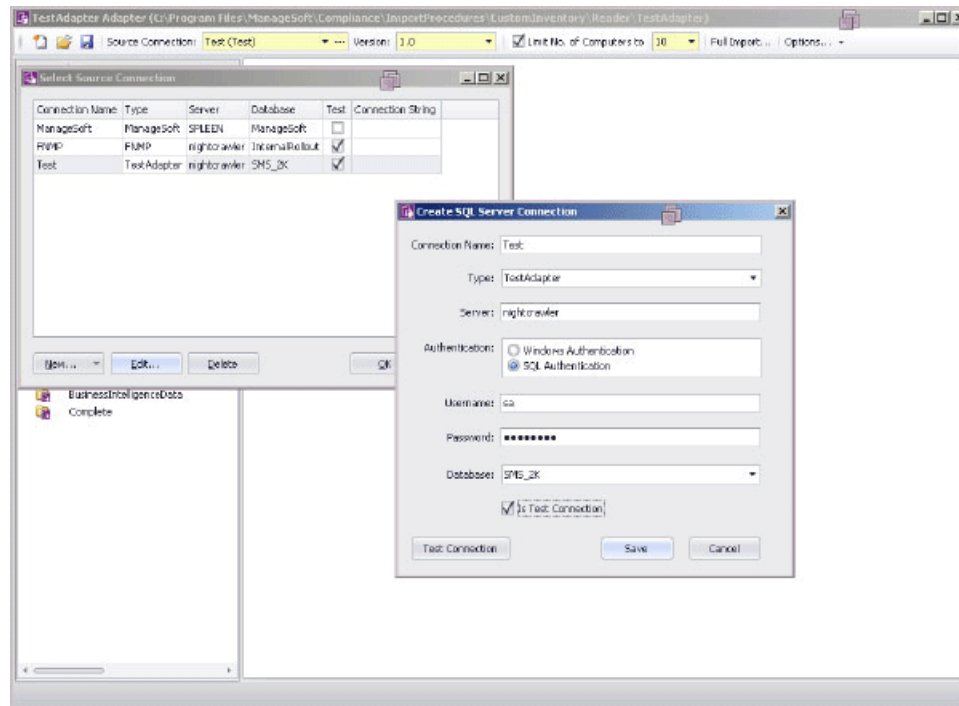
Publishing an inventory adapter means taking it out of test mode and putting it into production. The data uploaded by a published adapter goes into your production database. Be sure you have adequately validated the information gathered by your adapter before taking this step. On an inventory beacon, validation includes unzipping the archive package, saved at C:\ProgramData\Flexera Software\Beacon\IntermediateData, and examining the data contained in the XML file. When you are satisfied with the gathered data, you can publish your adapter into production.



#### ***To publish your adapter:***

1. In the toolbar, on right-hand end of the **Source Connection** control, click the [...] ellipsis button.
2. In the **Select Source Connection** dialog, ensure that the correct connection is selected, and click **Edit...**
3. In the **Create SQL Server Connection** dialog, clear the **Is Test Connection** check box.





4. Click **Save**.

5. Click **OK**.

This connection is now visible in the connections dialog on the inventory beacon. The Compliance Importer can now use this connection with your adapter for future inventory imports. You declare and choose a schedule for execution of this connection in the inventory beacon user interface.

## 15

# Inventory Adapter Object Model

IT Asset Management (Cloud)

A reference for all database objects, and their properties, that can be imported through your inventory beacon.

Here is a complete list of the database objects (and their permissible attributes) that you may import through a custom inventory adapter that runs on your inventory beacon.

## Inventory Object: AccessingDevice

IT Asset Management (Cloud)

AccessingDevice objects are uploaded to the ImportedAccessingDevice table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedAccessingDevice table holds a record client access device information.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessingDeviceID	A numeric reference into a static table. May be null.	Matching accessing device ID. Foreign key to the AccessingDevice table.
ComputerName	Alpha-numeric text (maximum 256 characters). May be null.	Computer name of the client accessing device.
Domain	Alpha-numeric text (maximum 100 characters). May be null.	Domain name of the client accessing device.
ExternalAccessingDeviceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used to identify the device in source connection
IPAddress	An ASCII string of alphanumeric characters and punctuation (length 256 characters). May be null.	IP Address of the client accessing device.

Property	Attributes	Notes
SerialNo	Alpha-numeric text (maximum 100 characters). May be null.	Serial no of the client accessing device.

## Inventory Object: AccessingUser

IT Asset Management (Cloud)

AccessingUser objects are uploaded to the ImportedAccessingUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedAccessingUser table holds a record of the user access information.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessingUserID	A numeric reference into a static table. May be null.	The matching AccessingUser ID. Foreign key to the AccessingUser table.
DomainName	Alpha-numeric text (maximum 100 characters). May be null.	Domain name of the accessing user.
ExternalAccessing UserID	Unsigned integer (bigint). Mandatory. Database key.	The accessing user id. This is part of the key.
SAMAccountName	Alpha-numeric text (maximum 64 characters). May be null.	SAM account name of the accessing user.
UserName	Alpha-numeric text (maximum 256 characters).	User name of the accessing user.

## Inventory Object: ActiveDirectoryComputer

IT Asset Management (Cloud)

ActiveDirectoryComputer objects are uploaded to the ImportedActiveDirectoryComputer table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveDirectoryComputer table stores the incoming active directory data for computers.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ComputerName	Alpha-numeric text (maximum 64 characters).	The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by <code>GetComputerName()</code> . For UNIX, it is the host name of the machine, as returned by <code>gethostname(2)</code> .
DomainName	Alpha-numeric text (maximum 100 characters).	The domain name for the computer.
GUID	A universally unique identifier. Mandatory. Database key.	The GUID of the computer.
SID	Alpha-numeric text (maximum 256 characters). May be null.	The SID of the computer.

## Inventory Object: ActiveDirectoryDomain

IT Asset Management (Cloud)

ActiveDirectoryDomain objects are uploaded to the `ImportedActiveDirectoryDomain` table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The `ImportedActiveDirectoryDomain` table stores the incoming active directory domains for a connection source.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
DomainFQDN	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	The fully qualified name domain name of the AD domain
FlatName	Alpha-numeric text (maximum 32 characters).	The AD domain flat name
LastADImportTime	Date/time field.	The last time the AD data was imported

# Inventory Object: ActiveDirectoryExternalMember

IT Asset Management (Cloud)

ActiveDirectoryExternalMember objects are uploaded to the ImportedActiveDirectoryExternalMember table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveDirectoryExternalMember table stores the incoming active directory data for external AD member objects.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ParentGroupGUID	A universally unique identifier. Mandatory. Database key.	The parent AD group GUID.
SID	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The SID of the member object.

# Inventory Object: ActiveDirectoryGroup

IT Asset Management (Cloud)

ActiveDirectoryGroup objects are uploaded to the ImportedActiveDirectoryGroup table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveDirectoryGroup table stores the incoming active directory data for a connection source.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
DomainName	Alpha-numeric text (maximum 100 characters).	The domain name for the user.
GUID	A universally unique identifier. Mandatory. Database key.	The GUID of the AD group.
Name	Alpha-numeric text (maximum 128 characters). May be null.	The AD group name
SID	Alpha-numeric text (maximum 256 characters). May be null.	The SID of the AD group.

# Inventory Object: ActiveDirectoryMember

IT Asset Management (Cloud)

ActiveDirectoryMember objects are uploaded to the ImportedActiveDirectoryMember table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveDirectoryMember table stores the incoming active directory data for AD member objects.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
GUID	A universally unique identifier. Mandatory. Database key.	The GUID of the member object.
ParentGroupGUID	A universally unique identifier. Mandatory. Database key.	The parent AD group GUID.

# Inventory Object: ActiveDirectoryUser

IT Asset Management (Cloud)

ActiveDirectoryUser objects are uploaded to the ImportedActiveDirectoryUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveDirectoryUser table stores the incoming active directory data for users.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
DomainName	Alpha-numeric text (maximum 100 characters).	The domain name for the user.
GUID	A universally unique identifier. Mandatory. Database key.	The GUID of the user.
SAMAccountName	Alpha-numeric text (maximum 20 characters).	The user name.
Sid	Alpha-numeric text (maximum 256 characters). May be null.	The Sid for the user.


# Inventory Object: ActiveSyncDevice

IT Asset Management (Cloud)

ActiveSyncDevice objects are uploaded to the ImportedActiveSyncDevice table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedActiveSyncDevice table stores details of ActiveSync partnerships. A partnership is a user/device pair, so there may be multiple rows for one device.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ActiveSyncID	Alpha-numeric text (maximum 512 characters). Mandatory. Database key.	<p>The EASIdentity presented by the source, a combination of the AD user and the unique device ID.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
DeviceID	Alpha-numeric text (maximum 100 characters). May be null.	The unique device identifier.
DeviceModel	Alpha-numeric text (maximum 100 characters). May be null.	The device model.
DeviceOS	Alpha-numeric text (maximum 100 characters). May be null.	The device operating system.
DeviceType	Alpha-numeric text (maximum 50 characters). May be null.	The device type.
DeviceUserAgent	Alpha-numeric text (maximum 100 characters). May be null.	The device user agent; an ActiveSync client-specific value that may identify the device type.
Domain	Alpha-numeric text (maximum 100 characters). May be null.	The domain of the device. This may be a flat name or FQDN.
EmailAddress	Alpha-numeric text (maximum 256 characters). May be null.	The user's primary email address.
ExchangeServer	Alpha-numeric text (maximum 256 characters). May be null.	The source exchange server for this information.

Property	Attributes	Notes
IMEI	Alpha-numeric text (maximum 256 characters). May be null.	IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
LastSuccessSync	Date/time field. May be null.	The last successful sync time for this partnership, in UTC.
LastSyncAttemptTime	Date/time field. May be null.	The last attempted sync time for this partnership, in UTC.
PhoneNumber	Alpha-numeric text (maximum 128 characters). May be null.	The phone number of the device. Used for mobile devices.
UserDisplayName	Alpha-numeric text (maximum 256 characters). May be null.	The AD user display name.
WhenCreatedUTC	Date/time field. May be null.	The date/time this partnership was created, in UTC.

## Inventory Object: ClientAccessEvidence

IT Asset Management (Cloud)

ClientAccessEvidence objects are uploaded to the ImportedClientAccessEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClientAccessEvidence table holds all of the client access evidence which has been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
Edition	Alpha-numeric text (maximum 50 characters). May be null.	The edition of the installed product.
ExternalAccessEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier of the client access evidence.
ProductName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the product being accessed by user or computer. This may include version and edition too.
UALRoleGUID	A universally unique identifier. May be null.	The UAL role GUID of the product being accessed by user or computer. This is used when retrieve data using UAL.



Property	Attributes	Notes
UALRoleName	Alpha-numeric text (maximum 256 characters). May be null.	The UAL role name of the product being accessed by user or computer. This is used when retrieve data using UAL.
Version	Alpha-numeric text (maximum 72 characters). May be null.	The version of the installed product.

## Inventory Object: ClientAccessEvidenceMapping

IT Asset Management (Cloud)

ClientAccessEvidenceMapping objects are uploaded to the ImportedClientAccessEvidenceMapping table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClientAccessEvidenceMapping is the mapping table for imported access evidence and access evidence

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessEvidenceID	A numeric reference into a static table. Mandatory. Database key.	Access evidend id. Foreign key to AccessEvidence table.
ExternalAccessEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	External Access evidend id. Foreign key to ImportedClientAccessedAccessEvidence table.



## Inventory Object: ClientAccessedAccessEvidence


IT Asset Management (Cloud)

ClientAccessedAccessEvidence objects are uploaded to the ImportedClientAccessedAccessEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClientAccessedAccessEvidence table holds a record of the installer evidence that has been installed on a computer from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClientAccessSource	Alpha-numeric text (maximum 100 characters). Default: 'Unknown'. Mandatory. Database key.	Referencing to the client access source type.
ExternalAccessEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	Access evidence id .Foreign key to the ImportedClientAccessEvidence table.
ExternalAccessingDeviceID	Unsigned integer (bigint). Mandatory. Database key.	Accessing computer id .Foreign key to the ImportedAccessingDevice table
		 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
ExternalAccessing UserID	Unsigned integer (bigint). Mandatory. Database key.	Accessing userid. Foreign key to the ImportedAccessingUser table
		 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property	Attributes	Notes
ExternalServer ComputerID	Unsigned integer (bigint). Mandatory. Database key.	Server computer id .Foreign key to the ImportedComputer table.
 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.		
ImportedClient AccessedAccess EvidenceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the installer evidence.

## Inventory Object: ClientAccessedAccessOccurrence

IT Asset Management (Cloud)

ClientAccessedAccessOccurrence objects are uploaded to the ImportedClientAccessedAccessOccurrence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClientAccessedAccessOccurrence table holds the access information of device or user

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessCount	Unsigned integer (int). Default: 1.	Number of access frequency for given date
AccessDate	Date/time field. May be null.	The access date.
ImportedClient AccessedAccess EvidenceID	Unsigned integer (bigint). Mandatory. Database key.	Access evidence id. Foreign key to the ImportedClientAccessedAccessEvidence table.
InventoryDate	Date/time field.	Date on which inventory occurrence was recorded.
LicenseDate	Date/time field. Mandatory. Database key.	Date which will be used for licensing purpose.


# Inventory Object: Cluster

## IT Asset Management (Cloud)

Cluster objects are uploaded to the `ImportedCluster` table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The `ImportedCluster` table holds all of the clusters which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterTypeID	A numeric reference into a static table. Default: 1.	The type of cluster.
DPM	Boolean (0 or 1). May be null.	Whether Distributed Power Management (DPM) is enabled
DRS	Boolean (0 or 1). May be null.	Whether Distributed Resource Scheduler (DRS) is enabled
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	<p>The unique identifier for this imported cluster.</p> <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
ExternalName	Alpha-numeric text (maximum 256 characters). May be null.	The identifier of the cluster in the external cluster management system.
InventoryAgent	Alpha-numeric text (maximum 64 characters). Default: ". May be null.	The name of the person or tool that performed the last inventory.
InventoryDate	Date/time field. May be null.	The date the cluster last had inventory reported.
Name	Alpha-numeric text (maximum 256 characters).	The user-visible name of the cluster.
Namespace	Alpha-numeric text (maximum 256 characters). May be null.	The name of the domain/datacenter containing the cluster.


# Inventory Object: ClusterGroup

IT Asset Management (Cloud)

ClusterGroup objects are uploaded to the ImportedClusterGroup table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClusterGroup table holds all of the group objects defined on clusters which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster.
ClusterID	A numeric reference into a static table. May be null.	The assigned identifier for this cluster group.
ClusterTypeID	A numeric reference into a static table. Default: 3.	Foreign key to the ClusterType table.
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for this imported cluster group.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		
Name	Alpha-numeric text (maximum 256 characters).	The name of the cluster group.

# Inventory Object: ClusterGroupMember

IT Asset Management (Cloud)

ClusterGroupMember objects are uploaded to the ImportedClusterGroupMember table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClusterGroupMember table holds all of the group memberships defined on clusters which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterGroupExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster group.
ComputerExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the external computer which is a member of the group.



**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

# Inventory Object: ClusterHostAffinityRule

IT Asset Management (Cloud)

ClusterHostAffinityRule objects are uploaded to the ImportedClusterHostAffinityRule table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClusterHostAffinityRule table holds all of the host affinity rules for a cluster which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster.

Property	Attributes	Notes
ClusterHostAffinityRuleTypeID	A numeric reference into a static table. Default: 1.	A unique identifier indicating a type of Cluster Host Affinity Rule.
ClusterHostGroupExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster host group.
ClusterVMGroupExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster VM group.
Name	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The name of the cluster group.

## Inventory Object: ClusterNode

IT Asset Management (Cloud)

ClusterNode objects are uploaded to the ImportedClusterNode table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedClusterNode table holds all of the cluster nodes which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterExternalID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster.
ClusterNodeTypeID	A numeric reference into a static table. Default: 1.	Foreign key to the ClusterNodeType table.
ComputerExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the external computer which is a member of the cluster.



**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

# Inventory Object: Computer

## IT Asset Management (Cloud)


Computer objects are uploaded to the `ImportedComputer` table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The `ImportedComputer` table holds all of the computers which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
CalculatedUser	Alpha-numeric text (maximum 128 characters). May be null.	The domain/SAMAccountName of the calculated user. Some inventory systems calculate the user who owns a computer. For example, it might be the user who, over the last ten logins, logged in most often.
ChassisType	Alpha-numeric text (maximum 128 characters). May be null.	The type of case of the computer. The value must be a (case insensitive) exact match for one of the values shown. Note that some license types use this information to optimize the licensing position, particularly with desktop and laptop computers.
ComplianceComputerTypeID	Unsigned integer (int). May be null.	If you know that the computer is a virtual machine or VM host, record that data here. If you are unsure, leave this cell empty (NULL): this allows the system to infer the computer type (for example, a computer with VMs linked to it is inferred to be a VM host). If data comes from multiple inventory sources, leaving this value as null also allows the value to be inserted from another source. So, unless there is a very good reason, do not just specify 'Computer', but allow the inference rules to help.
ComputerName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by <code>GetComputerName()</code> . For UNIX, it is the host name of the machine, as returned by <code>gethostname(2)</code> .
Domain	Alpha-numeric text (maximum 100 characters). May be null.	The domain of the computer.
EmailAddress	Alpha-numeric text (maximum 256 characters). May be null.	The email address associated with the device. Typically used for mobile devices.



Property	Attributes	Notes
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
FirmwareSerialNumber	Alpha-numeric text (maximum 100 characters). May be null.	Serial number in the system firmware such as BIOS, EEPROM etc.
HardwareInventoryDate	Date/time field. May be null.	<p>The date (and optionally time) when the hardware was last inventoried. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale. Notice that this value is not available in the web interface.</p>
HostID	Alpha-numeric text (maximum 100 characters). May be null.	The HostID hardware property for the server hosting this machine partition (when inventorying a machine partition such as Solaris Zone, AIX IPar, HP-UX nPar/vPar).
HostIdentifyingNumber	Alpha-numeric text (maximum 128 characters). May be null.	Virtual hosts may have an identifier that is unique only across that hardware model. It is less unique than the true hardware serial number, for example.
HostType	Alpha-numeric text (maximum 128 characters). May be null.	The type of the physical host computer.

Property	Attributes	Notes
ILMTAgentID	Unsigned integer (bigint). May be null.	The unique ID used by the IBM License Metric Tool (ILMT) inventory agent on this device, if the inventory source is aware of this value. This can be used to track a computer over time and can be used to socialize different inventory sources. Currently the ILMT and ManageSoft inventory adapters report this value.
IMEI	Alpha-numeric text (maximum 256 characters). May be null.	IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
IPAddress	Alpha-numeric text (maximum 256 characters). May be null.	The IP address of the computer.
IgnoredDueToLicense	Boolean (0 or 1). Default: 0.	True if this machine is not imported into compliance computer table due to license limitation
IncompleteRecord	Boolean (0 or 1). May be null.	Used to identify records which do not have all information specified. Primarily used for ManageSoft source connections where the domain name was not reliably reported.
InventoryAgent	Alpha-numeric text (maximum 128 characters).	The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required.
InventoryDate	Date/time field. May be null.	The date the computer last had inventory reported.
IsDuplicate	Boolean (0 or 1). Default: 0.	Used to identify that imported computer is a duplicate of another, whereby a new computer will not be created.
IsRemoteACLDevice	Boolean (0 or 1). Default: 0.	Used to determine if the current record is a remote ACL based device.
LastLoggedOnUser	Alpha-numeric text (maximum 128 characters). May be null.	The DOMAIN/SAMAccountName of the user last logged onto the computer.

Property	Attributes	Notes
LastSuccessfulInventoryDate	Date/time field. May be null.	For incremental imports, this represents the inventory date of the computer in the source at the time this record was last successfully imported. If the import procedure has failed, this may be different to the inventory date. At the end of a successful incremental import, this value is updated to match the inventory date. If no value is present in this field, either there has not been a successful import of this computer or the reader for this record is not using an incremental update model.
LegacySerialNo	Alpha-numeric text (maximum 100 characters). May be null.	A previous serial number of this computer that can also be used for matching.
MACAddress	Alpha-numeric text (maximum 256 characters). May be null.	The MAC address of the computer.
MDScheduleContainsPVUScan	Boolean (0 or 1). Default: 0. May be null.	Does this managed device include an event in its current schedule for running extra IBM PVU hardware scans.
MDScheduleGenerated Date	Date/time field. May be null.	The last time the managed device schedule was regenerated.
MachineID	Alpha-numeric text (maximum 100 characters). May be null.	For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms.
Manufacturer	Alpha-numeric text (maximum 128 characters). May be null.	The manufacturer of the computer hardware.
MaxClockSpeed	Unsigned integer (int). May be null.	The maximum clock speed of the fastest processor in the computer.
ModelNo	Alpha-numeric text (maximum 128 characters). May be null.	The model number of the computer.
NumberOfCores	Unsigned integer (int). May be null.	The number of cores in the computer.
NumberOfDisplay Adapters	Unsigned integer (int). May be null.	The number of graphics cards in the computer.
NumberOfHardDrives	Unsigned integer (int). May be null.	The number of hard drives in the computer.
NumberOfLogical Processors	Unsigned integer (int). May be null.	The number of logical processors in the computer.
NumberOfNetworkCards	Unsigned integer (int). May be null.	The number of network cards in the computer.

Property	Attributes	Notes
NumberOfProcessors	Unsigned integer (int). May be null.	The number of processors in the computer.
NumberOfSockets	Unsigned integer (int). May be null.	The number of sockets in the computer.
OperatingSystem	Alpha-numeric text (maximum 128 characters). May be null.	The operating system of the computer.
PartialNumberOfProcessors	Fractional number (float). May be null.	The fractional processor count available to this computer.
PhoneNumber	Alpha-numeric text (maximum 128 characters). May be null.	The phone number of the device. Used for mobile devices.
ProcessorType	Alpha-numeric text (maximum 256 characters). May be null.	The type of processor in the computer.
SerialNo	Alpha-numeric text (maximum 100 characters). May be null.	The serial number of the computer.
ServicePack	Alpha-numeric text (maximum 128 characters). May be null.	The service pack installed for the operating system.
ServicesInventoryDate	Date/time field. May be null.	The date when services (for example, Oracle) were last scanned on this computer. For automated/scheduled data uploads through an inventory beacon, make sure that inventory dates are kept current, as they are used to report out-of-date inventory sources. For a one-time upload to the central application server, leave inventory dates empty (null). At each import from the saved file, the import date is used as the inventory date, which prevents the inventory becoming stale.
TotalDiskSpace	Unsigned integer (bigint). May be null.	The total size of all hard drives in the computer.
TotalMemory	Unsigned integer (bigint). May be null.	The total RAM in the computer, in bytes.
UUID	A universally unique identifier. May be null.	The BIOS UUID of the computer.
UntrustedSerialNo	Boolean (0 or 1). Default: 0.	Is this computer known to have a serial number from a data source that should not be trusted.

# Inventory Object: ComputerCustomProperty

IT Asset Management (Cloud)

ComputerCustomProperty objects are uploaded to the ImportedComputerCustomProperty table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedComputerCustomProperty table is used by the importer to import custom properties for computers.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier, in the source connection, of the computer that this property belongs to.
PropertyNameID	Unsigned integer (int). Mandatory. Database key.	The identifier for custom property in the ImportedCustomPropertyName table.
PropertyValue	Alpha-numeric text (maximum 256 characters).	The value of the custom property.

# Inventory Object: ConsolidatedAccessEvidence




IT Asset Management (Cloud)

ConsolidatedAccessEvidence objects are uploaded to the ConsolidatedAccessEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.



ConsolidatedAccessEvidence provides a simpler interface to specify client access happening on application installed on server computers. It combines the server computer, and its access evidence details into a single row.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessCount	Unsigned integer (int). Default: 1. May be null.	Number of times the product was accessed on the given access date.

Property	Attributes	Notes
AccessDate	Date/time field. Mandatory. Database key.	<p>The access date of the access evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
AccessingDevice ComputerName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>IP Address of the device accessing the product.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
AccessingDeviceDomain	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>Domain name of the device accessing the product.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
AccessingDeviceIP Address	An ASCII string of alphanumeric characters and punctuation (length 256 characters). Mandatory. Database key.	IP Address of the accessing device.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
AccessingDevice SerialNo	Alpha-numeric text (maximum 100 characters). May be null.	Serial number of the device accessing the product.
AccessingUser	Alpha-numeric text (maximum 128 characters). Mandatory. Database key.	The DOMAIN/SAMAccountName of the user accessing the product.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
ClientAccessSource	Alpha-numeric text (maximum 100 characters). Default: Manual. May be null.	The source type of the access evidence.
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.

Property	Attributes	Notes
Edition	Alpha-numeric text (maximum 50 characters). Default: . Mandatory. Database key.	<p>The edition of the software as reported by the access evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InventoryDate	Date/time field. Default: getdate(). May be null.	The date (and optionally time) the access evidence record was inventoried.
ProductName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The product name of the software as reported by the access evidence.
Version	Alpha-numeric text (maximum 72 characters). Default: . Mandatory. Database key.	<p>The version of the software as reported by the access evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>



# Inventory Object: ConsolidatedCluster

IT Asset Management (Cloud)

ConsolidatedCluster objects are uploaded to the ConsolidatedCluster table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The Cluster spreadsheet provides a simple interface for defining server clustering. It is useful when combined with the ClusterGroup and ClusterHostAffinityRule spreadsheets.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for this imported cluster. This may be a string or an integer.
ClusterName	Alpha-numeric text (maximum 128 characters).	The name of the cluster in the external cluster management system.
ClusterType	Alpha-numeric text (maximum 128 characters).	The kind of cluster. The value must be an exact case-insensitive match to one of the permitted values.
DPM	Boolean (0 or 1). May be null.	Whether Distributed Power Management (DPM) is enabled on the cluster.
DRS	Boolean (0 or 1). May be null.	Whether Distributed Resource Scheduler (DRS) is enabled on the cluster.
InventoryAgent	Alpha-numeric text (maximum 64 characters). May be null.	The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required.
InventoryDate	Date/time field. May be null.	The date (with optional time) that the cluster last had inventory reported.
Namespace	Alpha-numeric text (maximum 256 characters). May be null.	Where the cluster is contained.

# Inventory Object: ConsolidatedClusterGroup

IT Asset Management (Cloud)

ConsolidatedClusterGroup objects are uploaded to the ConsolidatedClusterGroup table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ClusterGroup spreadsheet uses data from the Cluster spreadsheet and defines groups of servers as well as computers that are members of these groups.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterGroupID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for this cluster group. This may be a string or an integer.
ClusterGroupName	Alpha-numeric text (maximum 128 characters). May be null.	The name of the cluster group. Depending on the value of the ClusterGroupType this will be a group of hosts or virtual machines.
ClusterGroupType	Alpha-numeric text (maximum 128 characters).	The kind of cluster included in the group. The value must be an exact case-insensitive match to one of the permitted values.
ClusterID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster. This may be a string or an integer and must match a value for the ClusterID in the cluster spreadsheet.
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the 'Computer' spreadsheet for a computer which is a member of the group. To identify all the members of the group, repeat as many lines as required in your spreadsheet where the other values in the row are identical, and only the 'ComputerID' value changes. Values in this column must match a ComputerID in the computer spreadsheet or the row will be skipped.

# Inventory Object: ConsolidatedClusterHostAffinityRule

IT Asset Management (Cloud)

ConsolidatedClusterHostAffinityRule objects are uploaded to the ConsolidatedClusterHostAffinityRule table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ClusterHostAffinity spreadsheet defines the groups of virtual machines which may run on groups of host servers.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClusterHostAffinityRuleType	Alpha-numeric text (maximum 128 characters).	The type of affinity rule. The value must be an exact case-insensitive match to one of the permitted values.
ClusterHostGroupName	Unsigned integer (bigint). Mandatory. Database key.	The name of the group of hosts that the ClusterVMGroupName virtual machines may run on.
ClusterID	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for the imported cluster, to which this affinity rule applies. This may be a string or an integer and must match a ClusterID from the cluster spreadsheet.
ClusterVMGroupName	Unsigned integer (bigint). Mandatory. Database key.	The name of the virtual machine group that may run on the ClusterHostGroupName hosts.
Name	Alpha-numeric text (maximum 128 characters). May be null.	The name of the cluster host affinity rule.

# Inventory Object: ConsolidatedComputer

IT Asset Management (Cloud)

ConsolidatedComputer objects are uploaded to the ConsolidatedComputer table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

'ConsolidatedComputer' consolidates data for the Computer, VirtualMachine, Domain, User, Cluster and CloudInstance objects, providing a simpler way to populate this information. Any spreadsheet row that includes a 'HostComputerID' is making that row a virtual machine, and the import process expects that virtualization data will be provided.




**Important:** A validation error will occur when an optional property has been populated for an object type without providing the other mandatory values. For example, if an optional property of a virtual machine is populated, then the VM type must be provided too. However, not all computers are VMs, meaning it is feasible to leave all properties concerning a VM empty.


Attributes are listed here in alphabetical order.


Property   (associated object type)	Attributes	Notes
Account (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The Account that is used to create the instance.
AffinityEnabled (VirtualMachine)	Boolean (0 or 1). Default: 0.	Set this to true (or 1) if this VM has affinity for its current host (so that it is unable to move to different host computers).
AvailabilityZone (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	Location of the instance or host.
BIOSUUID (Computer)	A universally unique identifier. May be null.	The BIOS UUID of the computer (physical or virtual), as provided by the operating system.
CalculatedUser (Computer)	Alpha-numeric text (maximum 128 characters). May be null.	The domain/SAMAccountName of the calculated user. Some inventory systems calculate the user who owns a computer. For example, it might be the user who, over the last ten logins, logged in most often.
ChassisType (Computer)	Alpha-numeric text (maximum 128 characters). May be null.	The chassis type of the device.
CloudServiceProvider (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	A unique identifier for a cloud service provider record. Note: Two different cloud instances cannot have the same InstanceCloudID and CloudServiceProvider.





**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property   (associated object type)	Attributes	Notes
ClusterID (ClusterNode)	Unsigned integer (bigint). Mandatory. Database key.	<p>The unique identifier for the cluster containing this computer. This must match the ClusterID used in the Cluster spreadsheet. If both the ClusterID and the ClusterNodeType do not match the data provided in the Cluster spreadsheet then the computer will not be associated with a cluster.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ClusterNodeType (Cluster)	Alpha-numeric text (maximum 128 characters). Default: 1. May be null.	The Cluster node type of the computer. Must be a (case insensitive) exact match for one of the values shown. If both the ClusterID and the ClusterNodeType do not match the data provided in the Cluster spreadsheet then the computer will not be associated with a cluster.
ComplianceComputerType (Computer)	Alpha-numeric text (maximum 128 characters). May be null.	If you know that the computer is a virtual machine or VM host, record that data here. If you are unsure, leave this cell empty (NULL): this allows the system to infer the computer type (for example, a computer with VMs linked to it is inferred to be a VM host). If data comes from multiple inventory sources, leaving this value as null also allows the value to be inserted from another source. So, unless there is a very good reason, do not just specify 'Computer', but allow the inference rules to help.
ComputerID (Computer, ClusterNode, VirtualMachine, CloudServiceInstance)	Unsigned integer (bigint). Mandatory. Database key.	The unique identifier for a computer (either physical or virtual). This identifier can either be an integer or a string. Keep this consistent across multiple imports: it is used to track the computer over time.

Property   (associated object type)	Attributes	Notes
ComputerName (Computer)	Alpha-numeric text (maximum 256 characters).	The name of the computer. In Windows, this is the NetBIOS name of the local computer, as returned by <code>GetComputerName()</code> . For UNIX, it is the host name of the machine, as returned by <code>gethostname(2)</code> .
CoreAffinity (VirtualMachine)	Alpha-numeric text (maximum 256 characters). May be null.	Contains a comma-separated list of core numbers (or ranges) for which this virtual machine has affinity. Cores are numbered sequentially up the sequence of processors. Example: 1, 5-8, 10
CPUAffinity (VirtualMachine)	Alpha-numeric text (maximum 256 characters). May be null.	Contains a comma-separated list of processor numbers (Host Logical Processors) or ranges for which this virtual machine has affinity. Example: 1, 3-5, 8
CPUUsage (VirtualMachine)	Unsigned integer (int). May be null.	The maximum CPU usage of the virtual machine (MHz).
DomainFlatName (Domain)	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>The flatname of the domain of the computer. Example: 'mycompany'.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
DomainQualifiedName (Computer, Domain)	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>The fully qualified domain name for the computer. Example: 'prod.mycompany.eu'.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property   (associated object type)	Attributes	Notes
EmailAddress (Computer)	Alpha-numeric text (maximum 256 characters). May be null.	The email address associated with the device. Typically used for mobile devices.
FirmwareSerialNumber (Computer)	Alpha-numeric text (maximum 100 characters). May be null.	The Serial number in the system firmware such as BIOS, EEPROM etc.
HostComputerID (VirtualMachine, CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The ComputerID of the server this virtual machine is hosted on. This may be a string or an integer and must match the ComputerID for another computer in this spreadsheet.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
HostID (Computer, CloudServiceInstance)	Alpha-numeric text (maximum 100 characters). May be null.	The HostID hardware property for the server hosting this machine partition (when inventorying a machine partition such as Solaris Zone, AIX IPar, HP-UX nPar/vPar).
HostIdentifyingNumber (VirtualMachine)	Alpha-numeric text (maximum 128 characters). May be null.	Virtual hosts may have an identifier that is unique only across that hardware model. It is less unique than the true hardware serial number, for example.
HostType (VirtualMachine)	Alpha-numeric text (maximum 128 characters). May be null.	The type (similar to model number) of the host, used for matching.
ImageID (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The ID of the image used to launch the instance.
IMEI (Computer)	Alpha-numeric text (maximum 256 characters). May be null.	IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone sets. Leave blank (null) for other device types.
InstanceAffinity (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The affinity setting for the instance on the Dedicated Host.

Property   (associated object type)	Attributes	Notes
InstanceCloudID (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The identifier used in the source connection for the cloud instance. Note: Two different cloud instances cannot have the same InstanceCloudID and CloudServiceProvider.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InstanceTenancy (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	Identifies whether the current device is a dedicated instance, or if the instance is running on a dedicated host.
InstanceType (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	Cloud provider instance type.
IPAddress (Computer)	Alpha-numeric text (maximum 256 characters). May be null.	The IP address of the computer in IPv4 or IPv6 format.
InventoryDate (Computer, CloudServiceInstance)	Date/time field. Default: getdate(). May be null.	The date (and optionally time) the computer last had inventory reported. This field is generally used for differential updates (that is, if the date/time has not changed since the previous import, the data record is not imported/updated).
LastLoggedOnUser (Computer, User)	Alpha-numeric text (maximum 128 characters). Mandatory. Database key.	<p>The DOMAIN/SAMAccountName of the user last logged onto the computer.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>



Property   (associated object type)	Attributes	Notes
LastLogonDate (Computer, User)	Date/time field. May be null.	The date and time when the user last logged on to the computer.
LaunchTime (CloudServiceInstance)	Date/time field. May be null.	The time the cloud instance was launched or when the reserved instance started.
LifecycleMode (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The time the instance was launched.
MACAddress (Computer, CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The MAC address of the computer. This may be a comma-separated list if there is more than one active network adapter in the system. Do not include inactive network adapters and network adapters with invalid MAC addresses.
MachineID (Computer)	Alpha-numeric text (maximum 100 characters). May be null.	For AIX, it is the System ID. For HP-UX, it is the Machine/Software ID. It is unset for other platforms.
Manufacturer (Computer, VirtualMachine)	Alpha-numeric text (maximum 128 characters). May be null.	The manufacturer of the computer.
MaxClockSpeed (Computer)	Unsigned integer (int). May be null.	The maximum clock speed of the fastest processor in the computer in kHz. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
MemoryUsage (VirtualMachine)	Unsigned integer (bigint). May be null.	The maximum memory usage of the virtual machine (bytes).
ModelNo (Computer, VirtualMachine)	Alpha-numeric text (maximum 128 characters). May be null.	The model number of the computer.
NetworkID (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The ID of the Virtual Private Cloud.
NumberOfCores (Computer, CloudServiceInstance)	Unsigned integer (int). May be null.	The total number of cores in the computer. If there is more than one physical processor in the computer, then this would be the sum of the core counts for all the processors. For example, in a computer with two quad-core processors, this value would be 8. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.

Property   (associated object type)	Attributes	Notes
NumberOfDisplayAdapters (Computer)	Unsigned integer (int). May be null.	The number of graphics cards in the computer.
NumberOfHardDrives (Computer, VirtualMachine)	Unsigned integer (int). May be null.	The number of physical hard drives in the computer. While the intent is physical drives, often this can end up being the number of disk partitions.
NumberOfLogicalProcessors (Computer)	Unsigned integer (int). May be null.	The number of logical processors in the computer. This is the number of 'execution contexts' the operating system has access to. It will commonly be equivalent to the number of processors in a single core, non-multi-threaded processor architecture, to the number of cores in a multi-core single threaded processor architecture, and to the number of threads in a multi-threaded processor architecture. For example, in a two processor, quad-core and hyper-threaded computer, this value would be 16. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
NumberOfNetworkCards (Computer, VirtualMachine)	Unsigned integer (int). May be null.	The number of network cards in the computer.
NumberOfProcessors (Computer, VirtualMachine)	Unsigned integer (int). May be null.	The total number of physical processors (CPU) in the computer. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
NumberOfSockets (Computer)	Unsigned integer (int). May be null.	The number of physical sockets into which a processor may be placed in the computer. It is rare that an inventory source can know this value. If unset, it is typically approximated by the number of processors.
OperatingSystem (Computer, VirtualMachine)	Alpha-numeric text (maximum 128 characters). May be null.	The operating system of the computer. For virtual machines, it is the configured operating system of the guest. Note that this operating system identification is not used for licensing.

Property   (associated object type)	Attributes	Notes
PartialNumberOfProcessors (Computer)	Fractional number (float). May be null.	Used in processor-based licensing, this is the non-integer number of cores allocated to this partition or virtual machine. When this property is null, the 'NumberOfCores' is used. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
PhoneNumber (Computer)	Alpha-numeric text (maximum 128 characters). May be null.	The phone number of the device. Used for mobile devices.
PoolName (VirtualMachine)	Alpha-numeric text (maximum 100 characters). May be null.	The name of the pool that the virtual machine belongs to.
PoolType (VirtualMachine)	Alpha-numeric text (maximum 100 characters). May be null.	The type of the pool that the virtual machine belongs to.
ProcessorType (Computer, VirtualMachine)	Alpha-numeric text (maximum 256 characters). May be null.	The descriptive string of the processor(s) in the computer. This may be a comma-separated list in the case where there is more than one physical processor in the system. Note that a number of server-based licenses depend on complete details of the processor types, counts and speeds to calculate a correct license position.
Region (CloudServiceInstance)	Alpha-numeric text (maximum 256 characters). May be null.	The cloud instance or host region. Note: This is not related to the Location ownership property on the computer.
SerialNo (Computer)	Alpha-numeric text (maximum 100 characters). May be null.	The serial number of the computer.
ServicePack (Computer)	Alpha-numeric text (maximum 128 characters). May be null.	The service pack installed for the operating system.
ThreadsPerCore (CloudServiceInstance)	Unsigned integer (int). May be null.	The number of thread per core of the instance.

Property   (associated object type)	Attributes	Notes
TotalDiskSpace (Computer)	Unsigned integer (bigint). May be null.	The total size of all hard drives in the computer in bytes. Note that this can be a very large number on modern systems. The maximum value for a bigint is 9,223,372,036,854,775,807, which can represent about 9.2 exabyte. While in practice it is unlikely that this size of storage capacity is reached for a single system, some systems can end up with large values through virtualized drives. Therefore, it is worth considering capping values when calculating total disk space, particularly when converting values from kilobytes or megabytes to bytes.
TotalMemory (Computer)	Unsigned integer (bigint). May be null.	The total RAM in the computer, in bytes.
VMEnabledState (VirtualMachine, CloudServiceInstance)	Alpha-numeric text (maximum 128 characters). Default: 4. May be null.	The operational state of the virtual machine. If present, the value must be a (case insensitive) exact match to one of the values shown.
VMLocation (VirtualMachine)	Alpha-numeric text (maximum 256 characters). May be null.	Location of the virtual machine on the file system.
VirtualMachineType (VirtualMachine)	Alpha-numeric text (maximum 100 characters). May be null.	The type of the virtual machine. If present, the value must be a (case insensitive) exact match to one of the values shown.
VirtualMachineUUID (VirtualMachine)	Alpha-numeric text (maximum 256 characters). May be null.	The unique identifier of the virtual machine provided by the virtualization infrastructure. (This may have the same value as the 'BIOSUUID', or have byte order reversed, or be altogether different.)



# Inventory Object: ConsolidatedFileEvidence



IT Asset Management (Cloud)

ConsolidatedFileEvidence objects are uploaded to the ConsolidatedFileEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.


ConsolidatedFileEvidence provides a simpler interface to specify files and their usage on computers. It combines the computer, file evidence and usage details into a single row.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessMode	Alpha-numeric text (maximum 128 characters). Default: 1. Mandatory. Database key.	<p>The access mode of the file evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
Company	Alpha-numeric text (maximum 100 characters). Default: . Mandatory. Database key.	<p>The company in the file header.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
Description	Alpha-numeric text (maximum 200 characters). Default: . Mandatory. Database key.	<p>The description in the file header.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
FileName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The name of the file used as evidence of software installation. For unix operating systems include the full path in the file name, including the opening '/'. For Windows operating systems the file path is specified in the FilePath column and this column must only contain the file name.
FilePath	Alpha-numeric text (maximum 400 characters). May be null.	The path of the file used as evidence of software installation.
FileSize	Unsigned integer (int). Default: 0. Mandatory. Database key.	<p>The size of the file in bytes.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property	Attributes	Notes
FileVersion	Alpha-numeric text (maximum 100 characters). Default: . Mandatory. Database key.	<p>The version number of the file used as evidence of software installation.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
Language	Alpha-numeric text (maximum 200 characters). May be null.	The language in the file header.
LastUsedDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The last used date of the usage.
NumberOfSessions	Unsigned integer (bigint). May be null.	The number of sessions that the file evidence was in use by the user specified in the UserID column during the usage tracking period. If multiple users used the same application on the computer, create one row for each user with usage.
ProductName	Alpha-numeric text (maximum 200 characters). May be null.	The product name in the file header.
ProductVersion	Alpha-numeric text (maximum 200 characters). May be null.	The product version number in the file header.
StartDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The start date of the usage.

Property	Attributes	Notes
UserID	Unsigned integer (bigint). Mandatory. Database key.	The DOMAIN/SAMAccountName for the user that the file evidence was used by. If this software was used by multiple users, create one row for each user of the software on the computer.
		 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

## Inventory Object: ConsolidatedInstallerEvidence


IT Asset Management (Cloud)

ConsolidatedInstallerEvidence objects are uploaded to the ConsolidatedInstallerEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.



ConsolidatedInstallerEvidence provides a simpler interface to specify installed applications and their usage on computers. It combines the computer, installer evidence and usage details into a single row.


Attributes are listed here in alphabetical order.



Property	Attributes	Notes
AccessMode	Alpha-numeric text (maximum 128 characters). Default: 1. Mandatory. Database key.	<p>The access mode of the installer evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
DatabaseName	Unsigned integer (bigint). Mandatory. Database key.	<p>If this installer evidence is an Oracle Database, then this field specifies the name of the database.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
DiscoveryDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The date that the installer evidence was first seen.
DisplayName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The display name of the software as reported by the installer evidence.

Property	Attributes	Notes
Evidence	Alpha-numeric text (maximum 32 characters). Default: . Mandatory. Database key.	<p>Identifier for the type of installer evidence.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
InstallDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The install date of the installer evidence.
InstanceName	Unsigned integer (bigint). Mandatory. Database key.	<p>If this installer evidence is an Oracle Database, then this field specifies the name of the database instance. If there are multiple instances, create a row for each instance in this spreadsheet.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
LastUsedDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The last used date of the usage.
NumberOfSessions	Unsigned integer (bigint). May be null.	The number of sessions that the installer evidence was in use by the user specified in the UserID column during the usage tracking period. If multiple users used the same application on the computer, create one row for each user with usage.
ProductCode	Alpha-numeric text (maximum 55 characters). May be null.	The product code of the evidence. This is usually the MSI product code.

Property	Attributes	Notes
Publisher	Alpha-numeric text (maximum 200 characters). Default: . Mandatory. Database key.	<p>The publisher of the software as reported by the installer evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
StartDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	<p>The start date of the usage.</p>
UserID	Unsigned integer (bigint). Mandatory. Database key.	<p>The DOMAIN/SAMAccountName for the user that the installer evidence was used by. If this software was used by multiple users, create one row for each user of the software on the computer.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
Version	Alpha-numeric text (maximum 72 characters). Default: . Mandatory. Database key.	<p>The version of the software as reported by the installer evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>


## Inventory Object: ConsolidatedOracleDatabaseUser


IT Asset Management (Cloud)

ConsolidatedOracleDatabaseUser objects are uploaded to the ConsolidatedOracleDatabaseUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

ConsolidatedOracleDatabaseUser provides a list of the users for each Oracle database instance.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessMode	Alpha-numeric text (maximum 128 characters). Default: 1. Mandatory. Database key.	<p>The access mode of the installer evidence. Leave this blank unless this row is a virtualized application. In that case choose one of the values below that matches your application or desktop virtualization infrastructure.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
AccountStatus	Alpha-numeric text (maximum 256 characters). May be null.	The current status of the end-user account.
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.
CreationDate	Date/time field. May be null.	The date and time when the end-user was created.
DatabaseName	Unsigned integer (bigint). Mandatory. Database key.	This field specifies the name of the database. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID or this row will be skipped.
DefaultTablespace	Alpha-numeric text (maximum 256 characters). May be null.	The default tablespace for an Oracle end-user.
DisplayName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The display name of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, Version, Publisher, DatabaseName and InstanceName or this row will be skipped.
Evidence	Alpha-numeric text (maximum 32 characters). Default: . Mandatory. Database key.	<p>Identifier for the type of installer evidence.</p> <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
InstanceName	Unsigned integer (bigint). Mandatory. Database key.	This field specifies the name of the database instance. If there are multiple instances, create a row for each instance in this spreadsheet. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID and DatabaseName or this row will be skipped.
LastLogonDate	Date/time field. May be null.	The date and time when the end-user last logged on to the computer.

Property	Attributes	Notes
Name	Alpha-numeric text (maximum 256 characters).	The name of the user.
Publisher	Alpha-numeric text (maximum 200 characters). Default: . Mandatory. Database key.	The publisher of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, DisplayName, Version, DatabaseName and InstanceName or this row will be skipped.
TempTablespace	Alpha-numeric text (maximum 256 characters). May be null.	The temporary tablespace for an Oracle end-user.
UserID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the instance end-user. This may be an integer or a string.
Version	Alpha-numeric text (maximum 72 characters). Default: . Mandatory. Database key.	The version of the software as reported by the installer evidence. It must match a row in the InstallerEvidence spreadsheet for the same ComputerID, DisplayName, Publisher, DatabaseName and InstanceName or this row will be skipped.




## Inventory Object: ConsolidatedRemoteAccessFile



IT Asset Management (Cloud)

ConsolidatedRemoteAccessFile objects are uploaded to the ConsolidatedRemoteAccessFile table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The spreadsheet consolidates ImportedRemoteUserToApplicationAccess, ImportedFileEvidence and ImportedUser tables.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessMode	Alpha-numeric text (maximum 128 characters). Default: 1. Mandatory. Database key.	<p>The AccessMode states how an application has been accessed.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
Company	Alpha-numeric text (maximum 100 characters). Default: . Mandatory. Database key.	<p>The company in the file header.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
Description	Alpha-numeric text (maximum 200 characters). Default: . Mandatory. Database key.	<p>The description in the file header.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
FileName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The name of the file used as evidence of software installation. For unix operating systems include the full path in the file name, including the opening '/'. For Windows operating systems the file path is specified in the FilePath column and this column must only contain the file name.</p>

Property	Attributes	Notes
FilePath	Alpha-numeric text (maximum 400 characters). May be null.	The path of the file used as evidence of software installation.
FileSize	Unsigned integer (int). Default: 0. Mandatory. Database key.	<p>The size of the file in bytes.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
FileVersion	Alpha-numeric text (maximum 100 characters). Default: . Mandatory. Database key.	<p>The version number of the file used as evidence of software installation.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
Language	Alpha-numeric text (maximum 200 characters). May be null.	The language in the file header.
ProductName	Alpha-numeric text (maximum 200 characters). May be null.	The product name in the file header.
ProductVersion	Alpha-numeric text (maximum 200 characters). May be null.	The product version number in the file header.
ServerID	Unsigned integer (bigint). Mandatory. Database key.	This is the ComputerID of the server that publishes this virtual application. The ComputerID must match a computer from the Computer spreadsheet, and that computer must have an installation of the application this file is part of. If the server does not have an installation of an appropriate application then the user will not be shown as having access to that application. This is a mandatory field.



Property	Attributes	Notes
UserID	Unsigned integer (bigint). Mandatory. Database key.	The fully qualified name of the user.


# Inventory Object: ConsolidatedRemoteAccessInstaller

IT Asset Management (Cloud)

ConsolidatedRemoteAccessInstaller objects are uploaded to the ConsolidatedRemoteAccessInstaller table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The spreadsheet consolidates ImportedRemoteUserToApplicationAccess, ImportedInstallerEvidence and ImportedUser tables.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessMode	Alpha-numeric text (maximum 128 characters). Default: 1. Mandatory. Database key.	The AccessMode states how an application has been accessed.  <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
DisplayName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The display name of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.
Evidence	Alpha-numeric text (maximum 32 characters). Default: . Mandatory. Database key.	The evidence type of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.
ProductCode	Alpha-numeric text (maximum 55 characters). May be null.	The product code of the evidence. This is usually the MSI product code and is not part of the unique identifier.
Publisher	Alpha-numeric text (maximum 200 characters). Default: . Mandatory. Database key.	Publishers of software applications (for example, "Microsoft").

Property	Attributes	Notes
UserID	Unsigned integer (bigint). Mandatory. Database key.	The DOMAIN\SAMAccountName of the user.
Version	Alpha-numeric text (maximum 72 characters). Default: . Mandatory. Database key.	The version of the software as reported by the installer evidence and is part of the unique identifier for installer evidence.


## Inventory Object: ConsolidatedVMPool

IT Asset Management (Cloud)

ConsolidatedVMPool objects are uploaded to the ConsolidatedVMPool table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The VMPool spreadsheet provides a simple method to associate virtual machines with groups (pools) on their host.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
HostComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer which is hosting the pool. The HostComputerID should match the ComputerID in the Computer spreadsheet. Otherwise the record will be ignored.
NumberOfCores	Fractional number (float). May be null.	The number of cores in this pool.
NumberOfProcessors	Fractional number (float). May be null.	The number of processors in this pool.
ObjectType	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The type of pool.  <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
ParentName	Alpha-numeric text (maximum 100 characters). May be null.	The name of the parent pool.
ParentObjectType	Alpha-numeric text (maximum 256 characters). May be null.	The type of pool of the parent.

Property	Attributes	Notes
PoolFriendlyName	Alpha-numeric text (maximum 256 characters).	The friendly name of the pool.
PoolName	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	The name of the pool.

## Inventory Object: ConsolidatedWMIEvidence


IT Asset Management (Cloud)

ConsolidatedWMIEvidence objects are uploaded to the ConsolidatedWMIEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

ConsolidatedWMIEvidence provides a simpler interface to specify Windows Management Instrumentation (WMI) properties on computers. Other Web-Based Enterprise Management (WBEM) properties are supported from Unix computers as well. The most important data to provide in this spreadsheet is operating system installs. The 'Win32\_OperatingSystem' class and the 'Name' property contains this data.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClassName	Alpha-numeric text (maximum 50 characters). Mandatory. Database key.	The WMI class name of the evidence. An example is 'Win32_OperatingSystem'.
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer. It must match the ComputerID from the Computer spreadsheet or the row will be ignored.

Property	Attributes	Notes
InstanceName	Alpha-numeric text (maximum 256 characters). Default: . Mandatory. Database key.	<p>The name of the WMI class instance. This is important when there a multiple instances of a WMI class on a computer. An example is the 'Win32_VideoController' class that may have many instances with the same properties. In this case you need to specify the name of the instance here, 'Intel(R) HD Graphics Family' or 'NVIDIA Quadro K2100M' for example.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
PropertyName	Alpha-numeric text (maximum 50 characters). Mandatory. Database key.	The WMI property name of the WMI evidence. An example is 'Name'.
PropertyValue	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The value of the property of the WMI evidence. An example is 'Microsoft Windows 7 Enterprise'

## Inventory Object: Domain



### IT Asset Management (Cloud)

Domain objects are uploaded to the ImportedDomain table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedDomain table holds all of the domains which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ComplianceDomainID	Unsigned integer (int). May be null.	Identifier of the domain in the ComplianceDomain table that this imported domain links to. This is populated as part of the import process and does not need to be provided by the source connections.

Property	Attributes	Notes
FlatName	Alpha-numeric text (maximum 200 characters). Mandatory. Database key.	<p>The flat name of the domain.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
QualifiedName	Alpha-numeric text (maximum 200 characters). Mandatory. Database key.	<p>The fully qualified name of the domain.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.



## Inventory Object: EvidenceAttribute

IT Asset Management (Cloud)

EvidenceAttribute objects are uploaded to the ImportedEvidenceAttribute table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedEvidenceAttribute table holds all of the instance attributes from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AttributeID	Unsigned integer (int). Mandatory. Database key.	The identifier used in the source connection for the instance attribute.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
AttributeName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The name of the instance attribute.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

## Inventory Object: FileEvidence


IT Asset Management (Cloud)

FileEvidence objects are uploaded to the ImportedFileEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedFileEvidence table holds all of the file evidence which has been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessModeID	Unsigned integer (int). Default: 1. May be null.	The access mode ID of the file evidence.
Company	Alpha-numeric text (maximum 100 characters). May be null.	The company in the file header.
Description	Alpha-numeric text (maximum 200 characters). Default: "".	The description in the file header.

Property	Attributes	Notes
ExternalFileID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the file evidence.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
FileName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the file used as evidence of software installation.
FilePath	Alpha-numeric text (maximum 400 characters). May be null.	The path of the file used as evidence of software installation.
FileSize	Unsigned integer (int). May be null.	The size of the file.
FileVersion	Alpha-numeric text (maximum 100 characters). May be null.	The version number of the file used as evidence of software installation.
Language	Alpha-numeric text (maximum 200 characters). May be null.	The language in the file header.
ProductName	Alpha-numeric text (maximum 200 characters). May be null.	The product name in the file header.
ProductVersion	Alpha-numeric text (maximum 200 characters). May be null.	The product version number in the file header.


## Inventory Object: ILMTPVUCounts

### IT Asset Management (Cloud)

ILMTPVUCounts objects are uploaded to the ImportedILMTPVUCounts table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

This table allows the summarized PVU sub capacity numbers to be imported from ILMT. These numbers are calculated by ILMT for a particular date range as PVU "reports".

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalNodeID	Unsigned integer (bigint). Mandatory. Database key.	The external ID of the server to which these points apply.
ExternalVMID	Unsigned integer (bigint). Mandatory. Database key.	The external ID of the virtual machine associated with the node (server).
		 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
FullCapacityCores	Unsigned integer (int). Default: 0.	The number of full-capacity licensable cores for the license on the computer.
FullCapacityPVU	Unsigned integer (int). Default: 0.	The number of full-capacity PVU counts consumed for the license on the computer.
PeakFullCapacityPVU	Unsigned integer (int). Default: 0.	The peak number of full-capacity PVU counts consumed for the license on the computer.
PeakSubCapacityPVU	Unsigned integer (int).	The peak number of sub-capacity PVU counts consumed for the license on the computer.
Publisher	An ASCII string of alphanumeric characters and punctuation (length 254 characters). Mandatory. Database key.	The name of the publisher of the title these points apply to.
SubCapacityCores	Unsigned integer (int). Default: 0.	The number of sub-capacity licensable cores for the license on the computer.
SubCapacityPVU	Unsigned integer (int). Default: 0.	The number of sub-capacity PVU counts consumed for the license on the computer.
TitleName	Alpha-numeric text (maximum 512 characters). Mandatory. Database key.	The name of the title these points apply to.





# Inventory Object: InstalledFileEvidence

IT Asset Management (Cloud)

InstalledFileEvidence objects are uploaded to the ImportedInstalledFileEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstalledFileEvidence table holds a record of the file evidence that has been installed on a computer from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalFileID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the file evidence.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
ExternalFilePathID	Unsigned integer (bigint). May be null.	The identifier used in the source connection for the path of the file evidence.
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer that the file evidence is installed on.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

## Inventory Object:



# InstalledFileEvidenceUsage


IT Asset Management (Cloud)

InstalledFileEvidenceUsage objects are uploaded to the ImportedInstalledFileEvidenceUsage table in the operations (inventory) database.

The ImportedInstalledFileEvidenceUsage table holds a record of end-users that are using file evidence from the source connection.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ActiveTimeInSeconds	Unsigned integer (bigint). May be null.	The number of seconds that the file evidence was in use during the usage tracking period.
ExternalFileID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the file evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer that the file evidence is installed on.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
ExternalUserID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the end-user that has used the file evidence.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>		
LastUsedDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The last used date of the file evidence.
NumberOfSessions	Unsigned integer (bigint). May be null.	The number of sessions that the file evidence was in use during the usage tracking period.
StartDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The start date of the file evidence usage tracking period.

## Inventory Object: InstalledInstallerEvidence




IT Asset Management (Cloud)

InstalledInstallerEvidence objects are uploaded to the ImportedInstalledInstallerEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstalledInstallerEvidence table holds a record of the installer evidence that has been installed on a computer from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
DiscoveryDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The date that the installer evidence was first seen.

Property	Attributes	Notes
ExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer that the installer evidence is installed on.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalInstallerEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the installer evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalInstanceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the instance that the installer evidence is associated with.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InstallDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The install date of the installer evidence.

# Inventory Object:



## InstalledInstallerEvidenceAttribute


IT Asset Management (Cloud)

InstalledInstallerEvidenceAttribute objects are uploaded to the ImportedInstalledInstallerEvidenceAttribute table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstalledInstallerEvidenceAttribute table holds a record of the values of the instance attributes for each installer evidence which is reported to be installed on a computer.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AttributeID	Unsigned integer (int). Mandatory. Database key.	The identifier used in the source connection for the instance attribute.
ExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer that the installer evidence is installed on.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		
ExternalInstallerEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the installer evidence.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		

Property	Attributes	Notes
ExternalInstanceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the instance that the installer evidence is associated with.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>		
Value	Alpha-numeric text (maximum 2Gb storage, or about 1 billion double-byte characters).	The value of the instance attribute for the installed installer evidence.




## Inventory Object: InstalledInstallerEvidenceUsage


IT Asset Management (Cloud)

InstalledInstallerEvidenceUsage objects are uploaded to the ImportedInstalledInstallerEvidenceUsage table in the operations (inventory) database.

The ImportedInstalledInstallerEvidenceUsage table holds a record of installed evidence being used from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer that the installer evidence is installed on.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalInstallerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the installer evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalInstanceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the instance that the installer evidence is associated with.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

Property	Attributes	Notes
ExternalUserID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the user that the installer evidence was used on.   <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
LastUsedDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The last used date of the installed installer evidence.
NumberOfSessions	Unsigned integer (bigint). May be null.	The number of sessions that the installer evidence was in use during the usage tracking period.
StartDate	An ASCII string of alphanumeric characters and punctuation (length 10 characters). May be null.	The start date of the installer evidence usage tracking period.

## Inventory Object: InstalledWMIEvidence




IT Asset Management (Cloud)

InstalledWMIEvidence objects are uploaded to the ImportedInstalledWMIEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstalledWMIEvidence table holds a record of the WMI evidence that has been installed on a computer from the source connections.

Attributes are listed here in alphabetical order.



Property	Attributes	Notes
ExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer that the WMI evidence is installed on.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the WMI evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InstanceName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The name of the WMI class instance used in the source connection for the WMI evidence</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>


# Inventory Object: InstallerEvidence

IT Asset Management (Cloud)

InstallerEvidence objects are uploaded to the ImportedInstallerEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstallerEvidence table holds all of the installer evidence which has been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessModeID	Unsigned integer (int). Default: 1. May be null.	The access mode ID of the file evidence.
DisplayName	Alpha-numeric text (maximum 256 characters). May be null.	The display name of the software as reported by the installer evidence.
Evidence	Alpha-numeric text (maximum 32 characters). May be null.	Identifier for the type of installer evidence.
ExternalInstallerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the installer evidence.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		
ProductCode	Alpha-numeric text (maximum 55 characters). May be null.	The product code of the evidence. This is usually the MSI product code.
Publisher	Alpha-numeric text (maximum 200 characters). May be null.	The publisher of the software as reported by the installer evidence.
Version	Alpha-numeric text (maximum 72 characters). May be null.	The version of the software as reported by the installer evidence.

# Inventory Object: InstallerEvidenceRepackageMapping

IT Asset Management (Cloud)

InstallerEvidenceRepackageMapping objects are uploaded to the ImportedInstallerEvidenceRepackageMapping table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.



The ImportedInstallerEvidenceRepackageMapping table is used by the importer to map the original and current installer evidence of repackaged softwares as reported by the ISO tag evidence.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
CurrentDisplayName	Alpha-numeric text (maximum 256 characters). May be null.	The current display name of the repackaged software as reported by the ISO tag evidence.
CurrentPublisher	Alpha-numeric text (maximum 200 characters). May be null.	The current publisher of the repackaged software as reported by the ISO tag evidence.
CurrentVersion	Alpha-numeric text (maximum 72 characters). May be null.	The current version of the repackaged software as reported by the ISO tag evidence.
OrigDisplayName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The original display name of the repackaged software as reported by the ISO tag evidence.



**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property	Attributes	Notes
OrigPublisher	Alpha-numeric text (maximum 200 characters). Mandatory. Database key.	<p>The original publisher of the repackaged software as reported by the ISO tag evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
OrigVersion	Alpha-numeric text (maximum 72 characters). Mandatory. Database key.	<p>The original version of the repackaged software as reported by the ISO tag evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

## Inventory Object: Instance




IT Asset Management (Cloud)

Instance objects are uploaded to the ImportedInstance table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstance table holds all of the instances which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AuditEvidence	Binary data from a file (maximum 2Gb). May be null.	Oracle GLAS CVS files in zip archive.
AuditEvidenceDate	Date/time field. May be null.	Oracle GLAS CSV files collection date.

Property	Attributes	Notes
ExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InstanceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the instance.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InstanceName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the instance.
ParentInstanceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the parent instance.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>


# Inventory Object: InstanceUser

IT Asset Management (Cloud)

InstanceUser objects are uploaded to the ImportedInstanceUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedInstanceUser table holds all of the end-users of an instance which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccountStatus	Alpha-numeric text (maximum 256 characters). May be null.	The current status of the end-user account.
ApplicationID	Alpha-numeric text (maximum 400 characters). Mandatory. Database key.	<p>The Oracle EBS application ID the user has access to.</p> <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
ComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer.
CreationDate	Date/time field. May be null.	The date and time when the end-user was created.
DefaultTablespace	Alpha-numeric text (maximum 256 characters). May be null.	The default tablespace for an Oracle end-user.
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the instance end-user.
InstanceID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the instance.
LastLogonDate	Date/time field. May be null.	The date and time when the end-user last logged on to the computer.
TempTablespace	Alpha-numeric text (maximum 256 characters). May be null.	The temporary tablespace for an Oracle end-user.


# Inventory Object: LicenseUser

IT Asset Management (Cloud)

LicenseUser objects are uploaded to the ImportedLicenseUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedLicenseUser table holds all of the external end-users (such as those used by Oracle or SAP licenses) retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
CostCenter	Alpha-numeric text (maximum 128 characters). May be null.	The cost center of the external end-user, as reported in SAP. Does not necessarily map to a cost centre in the GroupEx table.
Description	Alpha-numeric text (maximum 400 characters). May be null.	The description of the external end-user.
Email	Alpha-numeric text (maximum 400 characters). May be null.	An e-mail address for the external end-user.
EmployeeNumber	Alpha-numeric text (maximum 256 characters). May be null.	The employee number of the external end-user.
ExternalID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the external end-user.
 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.		
FirstName	Alpha-numeric text (maximum 256 characters). May be null.	The first name of the external end-user.
LastName	Alpha-numeric text (maximum 256 characters). May be null.	The last name or surname of the external end-user.
LicenseUserID	Unsigned integer (int). May be null.	The identifier of the external end-user in the LicenseUser table that this imported end-user links to. This is populated by the import process and does not need to be provided by the source connections.

Property	Attributes	Notes
UserName	Alpha-numeric text (maximum 400 characters). May be null.	The name of the external end-user.

# Inventory Object: RelatedInstalledInstallerEvidence

IT Asset Management (Cloud)

RelatedInstalledInstallerEvidence objects are uploaded to the ImportedRelatedInstalledInstallerEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedRelatedInstalledInstallerEvidence table holds parent-child relationship between installer evidence.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ChildExternal	ComputerID	Unsigned integer (bigint). Mandatory. Database key.
		The identifier used in the source connection for the computer that the installer evidence is installed on.






**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

ChildExternal	InstallerEvidenceID	Unsigned integer (bigint). Mandatory. Database key.
		The identifier used in the source connection for the installer evidence.



**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.



Property	Attributes	Notes
ConfidenceLevel	Unsigned integer (int). May be null.	Confidence level for each bundled installer evidence (as a percentage).
IsCharged	Boolean (0 or 1). Mandatory. Database key.	<p>The identifier used in the source connection to determine the pricing relation between parent and child installer evidence (specifies if it is charged = 1 or free = 0).</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ParentExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer that the installer evidence is installed on.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ParentExternalInstallerEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the installer evidence.</p> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>



# Inventory Object: RemoteUserToApplicationAccess




IT Asset Management (Cloud)

RemoteUserToApplicationAccess objects are uploaded to the ImportedRemoteUserToApplicationAccess table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedRemoteUserToApplicationAccess table stores the applications that remote users have access to

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AccessModeID	Unsigned integer (int). Mandatory. Database key.	The access mode ID for the remote application access.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		
ExternalFileID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the file evidence.
<div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.         </div>		

Property	Attributes	Notes
ExternalInstallerEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the installer evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalServerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The External Server ID for the remote server.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalUserID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the end-user that has used the file evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
LastUsedDate	Date/time field. May be null.	The last time the remote application was used by the user.
VDIGroupUUID	A universally unique identifier. May be null.	The desktop group UUID from which the application is published

# Inventory Object: Site

IT Asset Management (Cloud)

Site objects are uploaded to the ImportedSite table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedSubnet contains sites imported from Microsoft Active Directory

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AutoPopulated	Boolean (0 or 1). Default: 0.	Is the site auto populated at source?
Enabled	Boolean (0 or 1). Default: 1.	Is the site enabled?
Name	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The site's name.

# Inventory Object: SiteSubnet

IT Asset Management (Cloud)

SiteSubnet objects are uploaded to the ImportedSiteSubnet table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedSiteSubnet contains sites and subnets imported from Microsoft Active Directory

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AutoPopulated	Boolean (0 or 1). Default: 0.	Is the subnet auto populated at source?
Enabled	Boolean (0 or 1). Default: 1.	Is the subnet enabled?
IPSubnet	Alpha-numeric text (maximum 64 characters). Mandatory. Database key.	The IP subnet.
IPSubnetBits	UNRECOGNIZED TYPE Mandatory. Database key.	The IP subnet mask in CIDR notation.
SiteName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	The site's name.


# Inventory Object: SoftwareLicense

IT Asset Management (Cloud)

SoftwareLicense objects are uploaded to the ImportedSoftwareLicense table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedSoftwareLicense table holds all of the licenses which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
EntitlementCount	Unsigned integer (int). May be null.	The number of entitlements for the license.
ExpiryDate	Date/time field. May be null.	The expiry date of a subscription license.
ExternalLicenseID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the license.
		 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
IsSubscription	Boolean (0 or 1). Default: 0.	Indicates whether or not the license is a subscription license.
LicenseName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the license.
PartNo	Alpha-numeric text (maximum 100 characters). May be null.	The publisher's part number for this license.
SoftwareLicenseID	Unsigned integer (int). May be null.	Identifier of the license in the SoftwareLicense table that this imported license links to. This is populated by the import process and does not need to be provided by the source connections.
SoftwareLicenseTypeID	Unsigned integer (int). May be null.	The license type ID of the license.



# Inventory Object: SoftwareLicenseAllocation

IT Asset Management (Cloud)

SoftwareLicenseAllocation objects are uploaded to the ImportedSoftwareLicenseAllocation table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedSoftwareLicenseAllocation table holds the links between licenses and end-users which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalLicenseID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the license.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ExternalUserID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the license.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>

# Inventory Object: User

IT Asset Management (Cloud)

User objects are uploaded to the `ImportedUser` table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The `ImportedUser` table holds all of the end-users which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
<code>ComplianceDomainID</code>	Unsigned integer (int). May be null.	Identifier of the domain in the <code>ComplianceDomain</code> table that this end-user belongs to. This is populated by the import process and does not need to be provided by the source connections.
<code>ComplianceUserID</code>	Unsigned integer (int). May be null.	Identifier of the end-user in the <code>ComplianceUser</code> table that this imported user links to. This is populated by the import process and does not need to be provided by the source connections.
<code>CostCenter</code>	Alpha-numeric text (maximum 128 characters). May be null.	The cost center of the end-user, as reported in SAP. Does not necessarily map to a cost centre in the <code>GroupEx</code> table.
<code>Domain</code>	Alpha-numeric text (maximum 100 characters). May be null.	The domain of the end-user.
<code>Email</code>	Alpha-numeric text (maximum 200 characters). May be null.	The email address of the end-user.
<code>EmployeeNumber</code>	Alpha-numeric text (maximum 128 characters). May be null.	The employee number of the end-user.
<code>ExternalID</code>	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the end-user.



**Note:** Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property	Attributes	Notes
FirstName	Alpha-numeric text (maximum 128 characters). May be null.	The first name of the end-user.
InventoryAgent	Alpha-numeric text (maximum 64 characters). May be null.	The name of the person or tool that performed the last inventory. For imported spreadsheets, you may wish to include the name of the person preparing the data, in case there is subsequent follow-up required.
IsBlacklisted	Boolean (0 or 1). Default: 0.	This is populated by the import process and does not need to be provided by the source connections. The field is set to True if the end-user matches a record from the <code>UserNameBlacklist</code> table, meaning the account should not be included in compliance calculations.
LastName	Alpha-numeric text (maximum 128 characters). May be null.	The last name or surname of the end-user.
MapUsingEmailAddress	Boolean (0 or 1). Default: 0.	Indicates whether or not the user's email address should be used to try and map it to an existing <code>ComplianceUser</code> record.
SAMAccountName	Alpha-numeric text (maximum 64 characters). May be null.	The SAM account name of the end-user.
UserName	Alpha-numeric text (maximum 64 characters). May be null.	The account name of the end-user.

## Inventory Object: VDI

### IT Asset Management (Cloud)



VDI objects are uploaded to the `ImportedVDI` table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.



The `ImportedVDIUser` table stores the list of VDI devices, their source VM template and the VDI group the VDI device resides under.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ApplicationDelivery Only	Boolean (0 or 1). May be null.	Determines whether the VDI device is used only to server applications.



Property	Attributes	Notes
BrokerType	Alpha-numeric text (maximum 64 characters). Mandatory. Database key.	<p>The broker type of the VDI device.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.
ComputerName	Alpha-numeric text (maximum 64 characters). May be null.	The computer name of the VDI.
Domain	Alpha-numeric text (maximum 100 characters). May be null.	The domain name of the VDI device.
ExternalDeviceID	Unsigned integer (bigint). May be null.	The identifier used in the source connection for the VDI device.
IsPersistent	Boolean (0 or 1). May be null.	Determine whether the VDI device is a persistent VDI device.
SiteName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The site name of the VDI.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

Property	Attributes	Notes
TemplateName	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>The VDI template the VDI is cloned from.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
VDIGroupName	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>The VDI group the VDI device belongs to.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
VDIGroupUUID	A universally unique identifier. May be null.	<p>The group UUID the VDI device belongs to.</p>




## Inventory Object: VDI Template

IT Asset Management (Cloud)

VDITemplate objects are uploaded to the ImportedVDITemplate table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedVDITemplate table stores the list of VDI templates.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
BrokerType	Alpha-numeric text (maximum 64 characters). Mandatory. Database key.	<p>The broker type of the VDI template.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
SiteName	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The site name of the VDI.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
TemplateName	Alpha-numeric text (maximum 64 characters). Mandatory. Database key.	<p>The template name of the VDI template.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
VDITemplateExternalID	Unsigned integer (bigint). May be null.	<p>The ExternalID of the VDI template in the ImportedComputer table.</p>


# Inventory Object: VDIUser

IT Asset Management (Cloud)

VDIUser objects are uploaded to the ImportedVDIUser table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedVDIUser table stores the list of users that have been granted access to VDI groups.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
BrokerType	Alpha-numeric text (maximum 64 characters). May be null.	The broker type of the VDI for the end user.
ExternalUserID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the end-user that has access to the VDI.</p> <div>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory. </div>
SiteName	Alpha-numeric text (maximum 256 characters). May be null.	The site name of the VDI.
VDIGroupName	Alpha-numeric text (maximum 100 characters). May be null.	The VDI group the end-user has access to.

# Inventory Object: VMHostManagedBySoftware

IT Asset Management (Cloud)

VMHostManagedBySoftware objects are uploaded to the ImportedVMHostManagedBySoftware table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedVMHostManagedBySoftware table contains relationships between installer evidence of management software and VM hosts it manages.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ExternalComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the computer that the management software installer evidence is installed on.
ExternalInstallerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for an installer evidence of management software.
ExternalVMHostID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the VM host computer that is managed by a management software.
RelationType	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	Identifier for the type of relation, to be matched against ImporterString column of RelationType table.



# Inventory Object: VMPool


IT Asset Management (Cloud)

VMPool objects are uploaded to the ImportedVMPool table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedVMPool table holds all of the virtual machine pools which have been retrieved from the source connections and the number of processors and cores that are assigned to each pool.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
HostComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the computer which is hosting the pool.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
NumberOfCores	Fractional number (float). May be null.	The number of cores available to this pool.
NumberOfProcessors	Fractional number (float). May be null.	The number of processors available to this pool.
ObjectType	Alpha-numeric text (maximum 256 characters). Mandatory. Database key.	<p>The type of pool.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
ParentName	Alpha-numeric text (maximum 100 characters). May be null.	The name of the parent pool. This is the PoolName property for the parent pool.
ParentObjectType	Alpha-numeric text (maximum 256 characters). May be null.	The type of pool of the parent.
PoolFriendlyName	Alpha-numeric text (maximum 256 characters). May be null.	The friendly name of the pool.

Property	Attributes	Notes
PoolName	Alpha-numeric text (maximum 100 characters). Mandatory. Database key.	<p>The name of the pool.</p> <hr/>  <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.

## Inventory Object: VirtualMachine

IT Asset Management (Cloud)

VirtualMachine objects are uploaded to the ImportedVirtualMachine table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.


The ImportedVirtualMachine table holds all of the virtual machines which have been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
AffinityEnabled	Boolean (0 or 1). Default: 0.	Set this to True if this VM is unable to move to different host computers.
CPUAffinity	Alpha-numeric text (maximum 256 characters). May be null.	Contains the CPU Affinity value for virtual machine(Host Logical Processors)
CPUUsage	Unsigned integer (int). May be null.	The maximum CPU usage of the virtual machine (MHz).
ComputerName	Alpha-numeric text (maximum 256 characters). May be null.	The computer name of the virtual machine.
CoreAffinity	Alpha-numeric text (maximum 256 characters). May be null.	Contains the Core Affinity value for virtual machine
FriendlyName	Alpha-numeric text (maximum 256 characters). May be null.	The friendly name of the virtual machine.
GuestFullName	Alpha-numeric text (maximum 256 characters). May be null.	Configured operating system for the guest.

Property	Attributes	Notes
HostComputerID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the virtual machine's host computer.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
InventoryAgent	Alpha-numeric text (maximum 64 characters). May be null.	The name of the person or tool that performed the last inventory.
Manufacturer	Alpha-numeric text (maximum 128 characters). May be null.	The manufacturer of the virtual machine.
MemoryUsage	Unsigned integer (bigint). May be null.	The maximum memory usage of the virtual machine (bytes).
ModelNo	Alpha-numeric text (maximum 128 characters). May be null.	The model number of the virtual machine.
NumberOfHardDrives	Unsigned integer (int). May be null.	The number of hard drives in the virtual machine.
NumberOfNetworkCards	Unsigned integer (int). May be null.	The number of network cards in the virtual machine.
NumberOfProcessors	Unsigned integer (int). May be null.	The number of processors in the virtual machine.
PartitionID	Alpha-numeric text (maximum 100 characters). May be null.	Partition ID generated and used by the managing virtualization platform
PartitionNumber	Unsigned integer (int). May be null.	Number of this partition
PoolName	Alpha-numeric text (maximum 100 characters). May be null.	The name of the pool that the virtual machine belongs to.
PoolType	An ASCII string of alphanumeric characters and punctuation (length 100 characters). May be null.	The type of the pool that the virtual machine belongs to.
ProcessorType	Alpha-numeric text (maximum 256 characters). May be null.	The type of processor in the virtual machine.



Property	Attributes	Notes
TotalMemory	Unsigned integer (bigint). May be null.	The total RAM in the computer, in bytes.
UUID	Alpha-numeric text (maximum 256 characters). May be null.	The UUID of the virtual machine.
VMComputerID	Unsigned integer (bigint). Mandatory. Database key.	The identifier used in the source connection for the virtual machine's computer.
 <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.		
VMEnabledStateID	Unsigned integer (int). May be null.	The state of the machine (powered on, off, etc).
VMLocation	Alpha-numeric text (maximum 256 characters). May be null.	Location of the virtual machine on the file system.
VMName	Alpha-numeric text (maximum 256 characters). May be null.	The name of the virtual machine.
VirtualMachineType	An ASCII string of alphanumeric characters and punctuation (length 100 characters). May be null.	The type of virtual machine.

## Inventory Object: WMIEvidence


### IT Asset Management (Cloud)

WMIEvidence objects are uploaded to the ImportedWMIEvidence table in the operations (inventory) database. Multiple imports will merge updated data with existing records, and add new records as applicable.

The ImportedWMIEvidence table holds all of the WMI evidence which has been retrieved from the source connections.

Attributes are listed here in alphabetical order.

Property	Attributes	Notes
ClassName	Alpha-numeric text (maximum 50 characters). May be null.	The WMI class name of the WMI evidence.

Property	Attributes	Notes
ExternalEvidenceID	Unsigned integer (bigint). Mandatory. Database key.	<p>The identifier used in the source connection for the WMI evidence.</p> <hr/> <p> <b>Note:</b> Strictly, this attribute may be null, because it forms part of a compound database key. However, null values may cause import errors (where this object does not get imported), and multiple records from the same connection having nulls may cause data loss through over-writing. It is therefore best practice to treat this field as mandatory.</p>
PropertyName	Alpha-numeric text (maximum 50 characters). May be null.	The WMI property name of the WMI evidence.
PropertyValue	Alpha-numeric text (maximum 256 characters). May be null.	The value of the property of the WMI evidence.



# Microsoft 365 Adapter

## IT Asset Management (Cloud)

Microsoft 365 is a set of productivity tools available online (through a cloud-based subscription license) and offline (installed locally). Microsoft 365 subscriptions include a range of productivity tools including Microsoft Office, Microsoft Exchange, Microsoft SharePoint, Microsoft Yammer, and Microsoft Skype for Business product lines. Multiple subscription plans (such as E1, E3, and E5) are available. Some of these plans allow you to install Office applications locally.

With detailed information about Microsoft 365, the chapters in this part may help you in managing Microsoft 365 licenses through IT Asset Management.



---

**Note:** For managing a your subscription licenses for Microsoft 365, you may prefer to use the Flexera One SaaS Management connector, which not only integrates information for Microsoft 365, but simultaneously imports information for all your SaaS-based products you choose to manage there. If you choose the Flexera One SaaS Management connector, you should not use either of the Microsoft 365 connectors described in this part.

## 1

# Microsoft 365 License Management Considerations

## IT Asset Management (Cloud)

Microsoft 365 licenses that allow local Office installations also allow you (subject to the chosen plan) to install Microsoft 365 on up to five PCs or Macs, and up to five tablets or smart phones, per Named User license. The following section provides an overview of the tools provided in each enterprise plan.

With the FlexNet Manager for Clients product, you can use IT Asset Management to manage your Microsoft 365 subscriptions. You can:

- Validate that the desired users have the right plan
- Determine how many Microsoft 365 licenses you need to purchase, while negotiating license agreements
- Ensure that business units are not under- or over-subscribed.

Based on the selected plan, a subscription of Microsoft 365 may allow you to access multiple Office applications online as well as offline (local installations) on up to five computers, and five tablets or smart phones. Because of this mixture, the compliance calculations for Microsoft 365 licenses are different than other applications.

For example, the E3 plan allows you to install and use the following applications:

- Exchange Online Plan 2 for Microsoft 365
- Skype for Business Online for Microsoft 365
- Office Professional Plus for Microsoft 365
- Azure Rights Management for Microsoft 365
- SharePoint Online for Microsoft 365
- Yammer for Microsoft 365



**Tip:** For a detailed description of the available Microsoft 365 subscription plans, and the applications supported in each plan, see the Microsoft 365 website.

## Considerations for managing Microsoft 365 licenses

To differentiate user records at various stages of processing, those already recorded within the compliance database of IT Asset Management are here called "compliance users".

- **Users:** Each user imported from Microsoft 365 is mapped to the appropriate compliance user record within IT Asset Management (by matching the **Email** or **Alternate email** field values). If an imported user cannot be mapped to any compliance user, a new compliance user record is created with a dummy inventory device assigned to it. These users consume against the Microsoft 365 multi-product license created for the Microsoft 365 tenant's subscription.



**Tip:** There are frequently differences between a user's corporate email address (such as *Sam.Doe@tmn.is.com*) and the same user's email address registered in Microsoft Office 365 (such as *Sam.Doe@tmnisoffice365portal.com*). Since only the email addresses are available for matching (given that your Active Directory accounts are not reflected in the Microsoft 365 online service), this can lead to duplicated user records. To avoid duplicate records, you can, prior to importing from Microsoft 365 Online Service, populate your existing user records with the Microsoft 365 email address in the **Alternate email** field. You may want to consider using a Business Adapter to make widespread updates. If you have not done this prior to importing from Microsoft 365 (and as a result some user records are now duplicated), you can update the **Alternate email** field in your AD-based records at any time, and the next full import and compliance calculation automatically cleans up those duplicates.

- **Devices:** Although the installed device information is available to Microsoft 365 administrators through the Microsoft 365 Admin Center, IT Asset Management imports no device information from Microsoft 365. Instead, imported users are mapped to compliance users within IT Asset Management, and the inventory devices linked to those compliance users are considered to be the accessing devices. The method of linking the inventory devices and users may be any of assigned user, calculated user, or last logged on user (in descending order of priority, and all visible in the **Ownership** tab of the inventory device properties). The system also handles the following exceptions:
  - *Missing device:* If an imported user maps to a compliance user who is *not* linked to any inventory device, a dummy remote device is created by using the naming convention *User Name (Remote)*. Such a dummy device is not treated as a managed device by IT Asset Management (and specifically, does not count as a licensed device for the license for IT Asset Management itself).
  - *Installations on extra devices:* If an installation of Microsoft 365 is detected on a device, and the device is *not* linked to any compliance user matched to a user imported from Microsoft 365, the installation consumes from an existing perpetual license.
- **Mobile devices:** Because IT Asset Management does not collect inventory from all mobile devices (such as iPads), you may need to monitor these devices separately. You can get the details of installed devices from the Microsoft 365 Admin Center. If this reveals too many devices where a single named user is authoring or editing Microsoft 365 documents, you may need to *deactivate* Microsoft 365 from one (or more) of the mobile devices.



**Tip:** You can read or present documents on a deactivated copy of Microsoft 365.

- **Local installations:** A Microsoft 365 subscription is represented as a Named User license, with each user permitted to have installations on multiple devices. Only Microsoft manages the access to Microsoft 365. Separately, if imported inventory identifies a local installation related to Microsoft Office on an inventory device within your enterprise, the compliance user linked to the inventory device is checked. If this compliance user is matched in the subscription for Office 365, IT Asset Management assumes that the software installation is licensed. If the device user is not matched in

the subscription for Office 365, IT Asset Management looks for an existing perpetual license to cover this installation.

- **Office versions:** A perpetual license of Microsoft 365 enables you to downgrade to an earlier version, but the subscription license does not. If you have a perpetual license of the older Office version, and you buy a subscription, the license validity of the older version gets extended until the subscription expiry date.
- **License entitlements:** The entitlements data for Microsoft 365 is directly imported from Microsoft 365. When an import matches an existing license record for Microsoft 365, its available entitlements are adjusted with each import. On the other hand, if no license exists for an imported subscription to Microsoft 365, a new Named User license is created for each subscription owned by each Microsoft 365 tenant, and the available entitlements are recorded there. In the license properties for this automatically-created license:
  - The **Duration** value (on the **Identification** tab) is set to *Subscription*.
  - Initially, no purchases are linked (on the **Purchase** tab), since your entitlements are imported from Microsoft 365. Should you wish to link purchases to this automatically-created license (for example, as a way of independently keeping track of your subscription purchases), notice that the value of the **Assigned** column is set to 0 (that is, the effective quantity on any linked purchases does *not* affect your total entitlements in the usual way). As well, the **Entitlement status** column displays *Not contributing*, to again remind you that for such a license, the compliance calculations are based entirely on the entitlement figures imported from Microsoft, rather than on purchase records.

## 2

# Managing Microsoft 365 Licenses through IT Asset Management

IT Asset Management (Cloud)

Here is a high-level procedure for managing Microsoft 365 licenses using IT Asset Management.



## To manage Microsoft 365 licenses:

1. To avoid duplicate user records, ensure that the user email addresses recorded in Microsoft 365 are saved in the user records visible in the **All IT Asset Users** page (**Organization > All IT Asset Users**).

These may be called "compliance users" after the compliance database table where they are saved. The Microsoft 365 email addresses may be saved in either the **Email** field or the **Alternate email** field (both on the **Details** tab of the user properties). Common practice is to preserve the corporate email address in the first of these, and save the one from Microsoft 365 in the **Alternate email** field.



**Tip:** If you have many records to update, consider preparing a spreadsheet that lists your enterprise (Active Directory) email addresses against the Office 365 email addresses for each user, and creating a business adapter to update your records from the spreadsheet by inserting the Microsoft 365 email addresses into the `ComplianceUser.AlternateEmail` column in your compliance database. Of course, if it is not practical to do this preparation before your first import from Microsoft 365, you can add the Microsoft 365 email addresses to your AD-based user records at any time. Thereafter, the next full import and compliance calculation removes any duplicate user records, based on the now-matching email addresses.

2. Ensure that you have created and scheduled a connector from IT Asset Management for each tenant of Microsoft 365.

Typically, each enterprise is represented as a single tenant, although mergers and acquisitions may mean that your enterprise has multiple Microsoft tenants. For details on how to create and schedule a connector to Microsoft 365, see [Creating Connections to Microsoft 365](#).



**Note:** The **Microsoft 365** connector is the recommended connector to use to create a connection to Microsoft 365. However, if you would like instructions using the legacy **Microsoft Office 365 (deprecated)** connector, see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).

3. During the next discovery and inventory collection (following a scheduled connection and the upload of the resulting data), IT Asset Management creates or updates the following objects:
  - **Users:** Each imported user from Microsoft 365 is mapped to the appropriate compliance user record (by matching the **Email** or **Alternate email** field values). If an imported user cannot be mapped to any compliance user, a new compliance user record is created with a dummy inventory device assigned to this user.
  - **Licenses:** A Named User multi-product license (*MS-Tenant-name Office 365 (Plan XX)*) is created for each subscription plan linked to each Microsoft tenant:
    - This license is derived from a template provided in the downloadable libraries, and selected based on the subscription plan identifier.
    - The name of the Microsoft tenant is included as a prefix in the name of each of these licenses. This helps to distinguish licenses when you have multiple Microsoft Online Service tenants.
    - Multiple applications are linked to this license, as defined in the license template.



**Note:** If a particular plan is not currently supported by IT Asset Management, a license with no linked application is created. When the support for this missing plan is added during subsequent automatic updates to the SKU library and Application Recognition Library, a recommended license change appears, suggesting that you link the newly-identified applications to the license.

- On the **Use rights & rules** tab of the license properties, in the **License consumption rules** section, the **Allocations consume license entitlements** check box is set. This is because Microsoft 365 is a Named User license, and allocations to the users must consume entitlements.
  - On the **Consumption** tab of the license properties, the template sets the **Allocation type** to Permanent.
4. Review the **All Users** and **All licenses** pages to ensure that the required users and licenses have been created.
  5. After license reconciliation has been completed, the **Product Summary** page shows the compliance status for Microsoft 365.
  6. Go to the **All Licenses** page (**Licenses > License Management > All Licenses**). Filter the records to view Microsoft 365 licenses. You can check the **Consumed** and **Used** columns to know how many subscription entitlements are being used.
  7. You can also view the following reports for insights on Microsoft 365 license consumption:
    - User license details
    - License Overlap.



**Note:** A subscription to a Microsoft 365 enterprise plan (except E1) allows you to install Microsoft 365 on up to five PCs or Macs, and up to five tablets. IT Asset Management can only track the installation on devices that are present within your enterprise network. If you wish to include any installation outside the enterprise network, you can create a dummy inventory device and allocate a Microsoft 365 license to track the usage.

### Changes in subsequent imports

Each subsequent import from Microsoft 365 updates the following for each license for Microsoft 365:

- Entitlement count



- Expiry date
- Allocations correctly configured (as described above).

3

# Connecting to Microsoft 365


IT Asset Management (Cloud)



This chapter provides information about how IT Asset Management connects to Microsoft 365.

## Connector overview

There are currently two connectors available. It is recommended that you use the use the **Microsoft 365** connector, a replacement for the existing **Microsoft Office 365 (deprecated)** connector. To help during the transition to the **Microsoft 365** connector, the legacy connector will remain available for a period of time. The following table provides an overview comparison of the connectors. This table assumes you have installed the FlexNet Beacon released with IT Asset Management 2019 R1 or later.

Connector Name	Installation folder	Notes
<b>Microsoft 365</b>	ProgramData\Flexera Software\Compliance\ ImportProcedures\ Inventory\Reader\ Microsoft 365	Note the following: <ul style="list-style-type: none"><li>• This Tier 1 inventory connector is now available to all customers.</li><li>• The <b>Source Type</b> (on the inventory beacon) is <b>Microsoft 365</b> .</li></ul>

 **Note:** This is the latest connector.

Connector Name	Installation folder	Notes
<b>Microsoft Office 365 (deprecated)</b>   <b>Note:</b> This is the legacy connector that is now deprecated.	ProgramData\Flexera Software\Compliance\ ImportProcedures\ Inventory\Reader\ Microsoft Office 365   <b>Note:</b> This installation folder name and path remains unchanged.	Note the following: <ul style="list-style-type: none"> <li>This Tier 1 inventory connector remains available although deprecated and will longer be supported on a future release.</li> <li>This connector's <b>Source Type</b> (specified on the inventory beacon PowerShell dialog) is <b>Microsoft Office 365 (deprecated)</b>. However, If you have not installed the FlexNet Beacon released with IT Asset Management 2019 R1 or later, then <b>Microsoft Office 365 (deprecated)</b> will not appear as a <b>Source Type</b> connection on the <b>Create PowerShell Source Connection</b> dialog, and instead will remain <b>Microsoft Office 365</b>.</li> </ul>

## Generating the Microsoft 365 license compliance position

To generate the compliance position for Microsoft 365 licenses, IT Asset Management needs information about subscriptions and usage of Microsoft 365 licenses (both online and offline):

- The local inventory collection process (within your enterprise) returns the local installations of Microsoft Office that are tied to your Microsoft 365 subscription(s).
- To get subscription and usage information from Microsoft 365 Online Service, the FlexNet Beacon must be configured with an inventory connection of source type **Microsoft 365** or **Microsoft Office 365 (deprecated)** for each tenant of Microsoft 365. (Typically, each enterprise is a single Microsoft tenant; but your corporate history, including mergers and acquisitions, may mean that your enterprise includes multiple Microsoft tenants. Each tenant may then have one or more subscriptions to different licensing plans, such as Midsize Business or Enterprise.)

When you have created and configured the Microsoft 365 connection, the relevant inventory beacon imports the licenses, users, and usage information from the Microsoft 365 Online Service account and uploads it to IT Asset Management according to your chosen schedule. For details on how to create and schedule a connector to Microsoft 365, see [Creating Connections to Microsoft 365](#) or see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).

# Prerequisites and Configuration Considerations

IT Asset Management (Cloud)

This topic describes prerequisites and configuration considerations for:

- Each inventory beacon that needs to download data from Microsoft 365
- The **Microsoft 365** or **Microsoft Office 365 (deprecated)** connector.

## The inventory beacon

The inventory beacon that will collect inventory for Microsoft 365 in the cloud requires a 64-bit operating system: Windows Server 2008 R2 SP1 or later, or Windows 7 SP1 or later.

Make sure that the following PowerShell connector prerequisite is met on each inventory beacon that needs to download data from Microsoft Office. This requirement should have been met when the inventory beacon was installed:

- PowerShell 5.1 or later, with the PowerShell execution policy set to RemoteSigned



**Tip:** Run PowerShell with administrator rights to execute the following commands:  
To check the currently-installed version of PowerShell:

```
$PSVersionTable.PSVersion
```

To set PowerShell execution policy:

```
Set-ExecutionPolicy RemoteSigned
```

Only if you are using the **Microsoft Office 365 (deprecated)** connector, the following additional prerequisites must be also met (and are *not* required for the **Microsoft 365** connector):

- 64-bit version of the Microsoft Online Services Sign-in Assistant
- Microsoft Azure Active Directory Module for Windows PowerShell (Microsoft Office 365 uses Azure Active Directory to manage user identities behind the scenes). Install the Microsoft Azure Active Directory Module for Windows PowerShell with these steps:
  1. Open an administrator-level PowerShell command prompt.
  2. Run the `Install-Module MSOnline` command.
  3. If prompted to install the NuGet provider, type Y and press ENTER.
  4. If prompted that the installer is not signed, type Y and press ENTER
  5. If prompted to install the module from PSGallery, type Y and press ENTER.
- Skype for Business Online, Windows PowerShell Module 64-bit version (see <https://docs.microsoft.com/en-us/skypeforbusiness/set-up-your-computer-for-windows-powershell/download-and-install-the-skype-for-business-online-connector>).

The inventory beacon also needs an appropriate version of the FlexNet Beacon software. The FlexNet Beacon released with IT Asset Management 2018 R2 (13.1.1) or later is required to use the **Microsoft 365** connector. However, installing the FlexNet Beacon included in the 2019 R1 (13.2.0) or later release provides maximum ease-of-use with the **Microsoft 365** connector by including auto-populated values on the **Create PowerShell Source Connection** dialog (that otherwise need to be entered manually). In addition, if you are using a FlexNet Beacon released prior to IT Asset Management 2019 R1, then **Microsoft Office 365 (deprecated)** will not appear as a **Source Type** connection on the **Create PowerShell Source Connection** dialog, and instead will remain **Microsoft Office 365**.

## Microsoft 365 connector configuration

There are three different possibilities for configuring your Microsoft 365 connector, each of which is described in detail

in a following topic:

1. You can configure the **Microsoft 365** connector using a multi-tenant app supplied by Flexera (and thus avoid configuring your own app). This requires the **Cloud Application Administrator** and **Reports Reader** roles, so that FlexNet Beacon can retrieve a token allowing read-only access to Microsoft Graph APIs. Microsoft Graph APIs allow only an administrator to read Users, Subscribed SKUs, and Reports from Microsoft 365. For more background, see <https://docs.microsoft.com/en-us/azure/active-directory/users-groups-roles/directory-assign-admin-roles>. For a step-by-step procedure, see [Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365](#).
2. You can configure the **Microsoft 365** connector using your own single-tenant app. This offers two kinds of authentication:
  - You can use similar credentials to the first case of the Flexera-supplied app
  - Importantly, you can instead configure a client secret that, for the lifetime that you declare, can be used to collect the appropriate data (without using additional credentials).

A detailed procedure covering either kind of authentication is available from [Registering an App to Connect to Microsoft 365 Using the Azure Portal](#).

3. If necessary, you can use the **Microsoft Office 365 (deprecated)** connector. If the maximum privileges of **Global administrator** cannot be used, then in order to collect usage data, the integration user must at a minimum have **Exchange administrator** and **Skype for Business administrator** roles in Microsoft 365 (available as check boxes under the **Custom administrator** role). For more information, see the Microsoft topic [About admin roles in the Microsoft 365 admin center](#). For a step-by-step procedure, see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).

## Microsoft 365 admin center configuration

For each tenant, the **Display concealed user, group, and site names in all reports** option in the **Microsoft admin center** is selected by default. This default configuration will prevent the **Microsoft 365** connector from reading the usage data. Therefore, you need to change the default configuration by performing the following steps:

1. In the **Microsoft admin center**, go to **Settings > Org Settings > Services**.
2. Select **Reports**.
3. Clear the selection for the statement **Display concealed user, group, and site names in all reports**, and then save your changes.

For more information, see the **Show user details in the reports** section in the Microsoft topic [Microsoft 365 Reports in the admin center](#).

## Network

For information relating to the network requirements for your firewall or proxy server, see the Microsoft topic [Office 365 URLs and IP address ranges](#). You may also like to refer to the Flexera knowledge base article [How to configure the Microsoft and Office 365 connectors to connect to the Internet through a proxy or firewall](#).

# Creating Connections to Microsoft 365

IT Asset Management 2024 R2.3 (Cloud)



**Important:** The FlexNet Beacon released with IT Asset Management 2018 R2 or later is required to use the **Microsoft 365** connector. However, installing the FlexNet Beacon included in the 2019 R1 or later release provides maximum ease-of-use with the **Microsoft 365** connector by including auto-populated values on the **Create PowerShell Source Connection** dialog (that otherwise need to be entered manually). In addition, **Microsoft Office 365 (deprecated)** will not appear as a **Source Type** connection on the **Create PowerShell Source Connection** dialog, and instead will remain **Microsoft Office 365**.



**Tip:** The connector to Flexera One SaaS Management supersedes and overrides both the **Microsoft 365** connector and the **Microsoft Office 365 (deprecated)** connector. This means that if you implement the Flexera One SaaS Management connector (see [IT Asset Management Settings: Integrations Tab](#)), any licenses that are created from imports through either of the Microsoft 365 connectors are automatically given a status of **Retired**, so that they have no impact on license reconciliation calculations. Instead, new licenses are created from the Flexera One SaaS Management imports, and these are included in license reconciliations as usual. For further details, see the online help for the **Integrations** tab of the **System Settings** page.

A separate connection is required for each tenant of Microsoft 365. (Typically, there is one tenant per enterprise; but your corporate history, particularly of mergers and acquisitions, may mean that your enterprise has multiple Microsoft tenants.) The inventory beacon requires this connection to import entitlements, users, and usage information from the Microsoft 365 online account. Each per-tenant import covers all subscriptions for that tenant.

When you use the **Microsoft 365** connector, there are two ways of creating connections to the Microsoft 365 online service. Refer to the following sections for associated instructions:

- [Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365](#) — The **Microsoft 365** connector uses a pre-registered multi-tenant app that has been configured by Flexera for the purpose of collecting Microsoft Office 365 data. This app is configured to allow our customers to collect data without the hassle of registering an app in their own instance.
- [Registering an App Using the Azure Portal to Connect to Microsoft 365](#) — This second method registers an app within your own Microsoft Office 365 instance.



**Tip:** This second method allows for the use of a client secret to collect the token for interaction with Microsoft 365, so that you do not need a user account and password with special privileges for regular operation of the connector.

- [Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365](#) — The **Microsoft 365** connector uses a pre-registered multi-tenant app that has been configured by Flexera for the purpose of collecting Microsoft Office 365 data. This app is configured to allow our customers to collect data without the hassle of registering an app in their own instance.
- [Registering an App to Connect to Microsoft 365 Using the Azure Portal](#) — This second method registers an app within your own Microsoft Office 365 instance.



**Tip:** This second method allows for the use of a client secret to collect the token for interaction with Microsoft 365,

---

so that you do not need a user account and password with special privileges for regular operation of the connector.


If you use the **Microsoft Office 365 (deprecated)** connector, refer to the instructions in [Creating Connections using the Microsoft Office 365 \(Deprecated\) Connector](#).

If you use the **Microsoft Office 365 (deprecated)** connector, refer to the instructions in [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).


## Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365

IT Asset Management (Cloud)


---

 **Important:** The instructions in this section are for use with the **Microsoft 365** connector. This is the recommended connector to use to create a connection to Microsoft Office 365 on an inventory beacon. If you would like instructions for using the legacy **Microsoft Office 365 (deprecated)** connector, see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).

---

 **Important:** The instructions in this section are for use with the **Microsoft 365** connector. This is the recommended connector to use to create a connection to Microsoft Office 365 on an inventory beacon. If you would like instructions for using the legacy **Microsoft Office 365 (deprecated)** connector, see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#).

---

 **Important:** The FlexNet Beacon released with IT Asset Management 2018 R2 or later is required to use the **Microsoft 365** connector. However, installing the FlexNet Beacon included in the 2019 R1 or later release provides maximum ease-of-use with the **Microsoft 365** connector by including auto-populated values on the **Create PowerShell Source Connection** dialog (that otherwise need to be entered manually).


The **Microsoft 365** connector uses a pre-registered multi-tenant app that has been configured by Flexera for the purpose of collecting Microsoft Office 365 data. This app is configured to allow our customers to collect data without the hassle of registering an app in their own instance. Use the following procedure to create a connection to Microsoft 365 on an inventory beacon using IT Asset Management's multi-tenant app.

---

 **To to create a connection to Microsoft 365 on an inventory beacon using IT Asset Management's multi-tenant app:**

1. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).

---

 **Tip:** Remember that you must run the inventory beacon software with administrator privileges.

2. From the **Data collection** group in the navigation bar, choose **Inventory Systems**.
3. Choose either of the following:
  - To change the settings for a previously-defined connection, select that connection from the list, and click

**Edit....**

- To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.
4. Complete (or modify) the values for the following required fields:
    - **Connection Name:** The name of the inventory connection. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name.
    - **Source Type:** Select **Microsoft 365** from this list.
  5. Optionally, if your enterprise uses a proxy server to enable Internet access, complete (or modify) the values in the **Proxy Settings** section of the dialog box in order to configure the proxy server connection.
    - **Use Proxy:** Select this check box if your enterprise uses a proxy server to enable Internet access. Complete the additional fields in the **Proxy Settings** section, as needed. If the **Use Proxy** check box is not selected, the remaining fields in the **Proxy Settings** section are disabled.
    - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`). This field is enabled when the **Use Proxy** check box is selected.
    - **Username** and **Password:** If your enterprise is using an authenticated proxy, specify the username and password of an account that has credentials to access the proxy server that is specified in the **Proxy Server** field. These fields are enabled when the **Use Proxy** check box is selected.
  6. In the **Microsoft 365** section, do the following:
    - a. Note that the following fields are auto-populated if you have installed the FlexNet Beacon released with IT Asset Management 2019 R1 or later. If you have not installed this FlexNet Beacon, then manually enter the following values into these fields:
      - **Token Endpoint:** `https://login.microsoftonline.com/common/oauth2/v2.0/token`
      - **Application (client) ID:** `5bb1a5a2-0d97-4335-9448-119f7b27aff9`
      - **Redirect URI:** `https://login.microsoftonline.com/common/oauth2/nativeclient`
      - **Authentication Flow:** from the drop-down, select **Authorization Code**. Additional fields are exposed.
      - **Authorization Endpoint:** `https://login.microsoftonline.com/common/oauth2/v2.0/authorize`
    - b. Next to the **Refresh Token** field, click the **Generate...** button to generate a refresh token that will be used to integrate with Microsoft 365.  
  
When you click the **Generate...** button to the right of the **Refresh Token** field, a **Microsoft** popup appears asking you to **Pick an account** to use to log into Microsoft Office 365.
    - c. Choose an Active Directory account with **Cloud application administrator** role privileges and enter the password.





**Tip:** The **Cloud application administrator** role is required in order for the FlexNet Beacon to retrieve a token that allows read only access to Microsoft Graph. For more information, see <https://docs.microsoft.com/en-us/azure/active-directory/users-groups-roles/directory-assign-admin-roles>. The generated refresh token can only be used to access data that user sees and consents to during the token generation process, which is offline read-only access to Active Directory and Reports (directory.**read.all**, reports.**read.all**, and offline\_access). Offline means the FlexNet Beacon can connect and get data from Office 365 at schedule run without user actually signing in.

A **Permissions requested** dialog appears.

- d. Click **Consent on behalf of your organization** to accept the read only permissions that will be granted to the refresh token.
- e. Click **Accept**.

The **Refresh Token** field is now populated.

- 7. At the bottom of the **FlexNet Beacon** interface, click **Save**.



**Tip:** Optionally, you may wish to select your connection, and click **Execute Now**, before you exit. You may also want to schedule data imports through this connection, for which see [Scheduling a Connection in the online help](#). You may also want to schedule data imports through this connection, for which see [Scheduling a Connection](#).

- 8. When you are done, click **Exit**.

After a successful data import, the users, applications, licenses, and usage data are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see [Inventory System Tab in the online help](#). For scheduling data imports through this connection, see [Scheduling a Connection, also in help](#).



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see [Inventory Systems Tab](#). For more about scheduling data imports, see [Scheduling a Connection](#).

IT Asset Management (Cloud)

Current

## Registering an App to Connect to Microsoft 365 Using the Azure Portal

IT Asset Management (Cloud)



**Important:** The instructions in this section are for use with the **Microsoft 365** connector. This is the recommended connector to use to create a connection to Microsoft Office 365 on an inventory beacon. If you would like instructions for using the legacy **Microsoft Office 365 (deprecated)** connector, see [Creating Connections using the Microsoft](#)

*Office 365 (Deprecated) Connector.* If you would like instructions using the legacy **Microsoft Office 365 (deprecated)** connector, see *Creating Connections Using the Microsoft Office 365 (Deprecated) Connector*.

An alternative method to the instructions provided in *Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365* is to register a single-tenant app in the Azure portal to connect to Microsoft 365 as shown in the steps in this section. An alternative method to the instructions provided in *Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365* is to register a single-tenant app in the Azure portal to connect to Microsoft 365 as shown in the steps in this section. This requires Global Administrator privileges for the initial registration. Using this process, it is possible to specify a client secret (under OAuth) to be used for regular operation of the connector, so that a separate user name and password are not required for operational use.



**To register an app in the Azure portal to connect to Microsoft 365:**

1. With an Azure account with Global Administrator privileges, login to the [Microsoft Azure portal](#).
2. In the left navigation bar, click **Azure Active Directory**.
3. Click **App registrations**.

4. Click the **New registration** button.

The **Register an application** page appears.

5. In the **Register an application** page, do the following:

- a. In the **Name** field, enter **FlexNet Beacon**.
- b. In the **Supported account types** section, choose **Accounts in this organizational directory only**.
- c. Click **Register**.

The Azure AD application is created, and its **FlexNet Beacon** overview page appears.

6. On the **FlexNet Beacon** page, click **Authentication** in the middle group of the navigation bar and do the following:

- a. In the **Redirect URIs** section, either:
  - Choose the redirect you would like to use, such as `https://login.microsoftonline.com/common/oauth2/nativeclient`; or
  - Click **Add URI** to expose a field where you may enter your preferred value, and then click the check mark at the end of that field.
- b. Click **Save** at the top of the page.

After a period of time, the registration completes, and the **FlexNet Beacon** overview page is displayed.

7. If you wish to use a client secret to authenticate normal operations of the Microsoft 365 connector, add a client secret for the FlexNet Beacon add-in:

- a. In the navigation bar, select **Certificates & secrets**.
- b. In the **Client secrets** section of the page, click **New client secret**.

A new **Add a client secret** panel appears.

- c. Add a **Description**, which acts as a friendly name for your client secret.

- d. Choose how long your client secret should remain valid (between 6 months and two years).

Choose a period that best suits your business processes. Shortly before the client secret expires, you need to generate a new client secret and update your add-in.

- e. Click **Add** at the bottom of the panel.

Your new secret appears temporarily in the list of **Client secrets**.



**Important:** You **must** immediately use the copy icon on the right of your new client secret value to capture the client secret and save it to a secure location where you can refer to it later, as it will not be shown in the Azure Partner Center again. Also record the **Expires** date, so you know the period of validity of your client secret (best practice is to set a reminder in your preferred enterprise technology calendar system to update the client secret shortly before it expires, as described in [https://docs.microsoft.com/en-us/office/dev/store/create-or-update-client-ids-and-secrets#bk\\_update](https://docs.microsoft.com/en-us/office/dev/store/create-or-update-client-ids-and-secrets#bk_update)).

- f. Only *after* you have safely copied your new client secret, select **API permissions** in the navigation bar, and then click **Add a permission > Microsoft graph > Application permissions**.

These permissions are required for the connector to authenticate as itself, without user interaction.

- g. Under **Select permissions**:

- a. Expand **Directory**, and select **Directory.Read.All**.
- b. Expand **Reports**, and select **Reports.Read.All**.



**Tip:** These permissions require admin consent, described next. The required button is only enabled when you are an administrator and have selected the permissions as just described.

- h. Above the set of permissions, click **Grant admin consent for *client-name***. In the confirmation dialog that appears, confirm the consent action.

The **Status** column in the list of permissions is updated with green check icons and **Granted** for *client-name* against each permission. This completes the settings in the Azure portal, and you can prepare to copy values into your inventory beacon interface.

- i. In the navigation bar select **Overview** again, and from the tabs across the top, select **Endpoints**.

Keep this panel open for copying values to your inventory beacon interface in step 12.

8. On the inventory beacon that will exercise the connection to Microsoft 365, log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Remember:** You must run the inventory beacon software with administrator privileges.

9. To create a new connection, click the down arrow on the right of the **New** split button, and choose **PowerShell**.

10. Complete the following required fields:

- **Connection Name:** Enter the name of the inventory connection. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name. Example:

Microsoft 365 import

- **Source Type:** Select **Microsoft 365** from this list.
- 11.** Optionally, if your enterprise uses a proxy server to enable Internet access, complete the values in the **Proxy Settings** section of the dialog box in order to configure the proxy server connection:
- **Use Proxy:** Select this check box if your enterprise uses a proxy server to enable Internet access. Complete the additional fields in the **Proxy Settings** section, as needed. If the **Use Proxy** check box is not selected, the remaining fields in the **Proxy Settings** section are disabled.
  - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`). This field is enabled when the **Use Proxy** check box is selected.
  - **Username and Password:** If your enterprise is using an authenticated proxy, specify the username and password of an account that has credentials to access the proxy server that is specified in the **Proxy Server** field. These fields are enabled when the **Use Proxy** check box is selected.

**12.** Complete the fields in the **Microsoft 365** section:

The source contents for most fields in this section are waiting in your Azure session, opened to the **FlexNet Beacon** product overview page.

- a. In the Azure product overview, ensure that you have clicked the **Endpoints** tab (near the top), and have open the panel listing multiple endpoints.
  - a. Click the **Click to copy** icon to the right of the **OAuth 2.0 token endpoint (v2)** field.
  - b. Paste this value into the **Token Endpoint** field in the inventory beacon interface.
- b. In Azure, in the **Overview** page:
  - a. Click the **Click to copy** icon to the right of the **Application (client) ID** field.
  - b. Paste this value into the **Application (client) ID** field in the inventory beacon interface.
- c. In Azure, in the **Overview** page, click the hyperlink in the **Redirect URIs** section. This opens the **Authentication** page.
  - a. Click the **Click to copy** icon to the right of the **Redirect URIs** setting you selected; or if you specified a custom URI, copy that URI.
  - b. Paste this value into the **Redirect URI** field in the inventory beacon interface.
- d. In the inventory beacon interface, from the **Authentication Flow** drop-down, choose either:
  - **Client Credentials** if you are using a client secret to authenticate connection to Microsoft 365. A **Client Secret** field appears: copy your client secret from the secure location where you previously saved it, and paste it into this field.
  - **Authorization Code** if you are *not* using a client secret, and instead are using a refresh token to authenticate connection to Microsoft 365. An **Authorization Endpoint** field appears.
    - a. In Azure, in the **Overview** page, click **Endpoints** again.
    - b. Click the **Click to copy** icon to the right of the **OAuth 2.0 authorization endpoint (v2)** field.

- c. Paste this value into the **Authorization Endpoint** field in the inventory beacon interface.
  - d. To generate a **Refresh Token** to authenticate the connection to Microsoft 365, click the **Generate...** button.
  - e. In the pop-up, log into Azure with your account name and password.
  - f. In Azure, in the panel for **Permissions requested**, select **Consent on behalf of your organization**, and click **Accept**.
  - g. Optionally, validate your settings in Azure by navigating to **API permissions**, where your **Configured permissions** list should display green "Granted" check marks for all four rights under Microsoft Graph.
13. At the bottom of the **FlexNet Beacon** interface, click **Test connection** for a success message (or use the error message to commence your trouble-shooting).
14. When the connection is successful, click **Save**.



**Tip:** Optionally, you may wish to select your connection, and click **Execute Now**, before you exit. You may also want to schedule data imports through this connection, for which see [Scheduling a Connection in the online help](#). You may also want to schedule data imports through this connection, for which see [Scheduling a Connection](#).

15. When you are done, click **Exit**.

After a successful data import, the users, applications, licenses, and usage data are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see [Inventory Systems Tab](#) in the online help. For scheduling data imports through this connection, see [Scheduling a Connection](#), also in help.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see [Inventory Systems Tab](#). For more about scheduling data imports, see [Scheduling a Connection](#).

IT Asset Management (Cloud)

Current

## Configuring Token Lifetimes in Azure Active Directory

IT Asset Management (Cloud)

This section is for Microsoft Azure AD administrators who may want to configure the lifetimes of refresh tokens and access tokens issued by Azure Active Directory. If your organization already have these set, these steps are not necessary.

FlexNet Beacon uses an Microsoft Azure Active Directory (AAD) native app for authentication when using Microsoft 365 inventory connections. You have a choice of whether to use the Flexera-created multi-tenant app for this

authentication, or to create your own single-tenant app.



**Tip:** The following discussion does **not** apply to a single-tenant app using a client secret.

When the authentication is complete and a user consents to access to the resource (Microsoft Graph) with read only permissions, the Azure AD generates and sends two tokens: a refresh token and an access token. These tokens are specific to the user, resource, and permissions. The refresh token is used to authenticate further in the future without a need to login while the access token is a session token. Typically, a refresh token is saved and is used first in every session to generate a new access token, once the access token is generated, it is then used in following calls within that session.

Since these tokens can be used anytime without a need for a user to manually login, Azure AD allows you to configure the lifetime for such tokens. After a refresh token expires, a user must login and consent to access to resources and permissions to get a new refresh token generated. After an access token expires, an app can use a valid refresh token to get a new access token.

The configuration of these tokens' lifetime is an Azure AD functionality and is applied to all applications in that tenant. To configure these tokens, an Azure AD administrator must have the Azure AD PowerShell module installed. For more information about these tokens, their default values and configuration, see <https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-configurable-token-lifetimes>.



#### **To configure lifetimes for the refresh token and an access token:**

1. Run PowerShell as an administrator.
2. Install the Azure AD PowerShell module:

```
Install-Module AzureADPreview
```

3. Connect to Azure AD:

```
Connect-AzureAD
```

4. Check if you already have a token lifetime policy:

```
$defaultTokenPolicy = Get-AzureADPolicy | Where-Object {$_.Type -eq  
"TokenLifetimePolicy" -and  
$_.IsOrganizationDefault -eq $true}
```

5. Check whether a token policy exists:

```
$defaultTokenPolicy
```

6. If a token policy exists, the previous command returns an object; otherwise, a blank. If a value is returned, you may want to examine the current token policies by entering the following:.

```
$defaultTokenPolicy.Definition
```

7. If a policy exists, it is returned. The following shows an example policy returned:

```
PS C:\WINDOWS\system32> $defaultTokenPolicy.Definition
```

```
{
  "TokenLifetimePolicy":
  {
    "Version":1,
    "AccessTokenLifetime":"0.00:10:00",
    "MaxInactiveTime":"90.00:00:00",
    "MaxAgeSingleFactor":"until-revoked",
    "MaxAgeMultiFactor":"until-revoked",
    "MaxAgeSessionSingleFactor":"until-revoked",
    "MaxAgeSessionMultiFactor":"until-revoked"
  }
}
```

The `AccessTokenLifetime` in the above example is set to 10 minutes which means that once an access token is generated, it remains active for ten minutes, after which the app must retrieve another generated access token. The `MaxInactiveTime` is set to 90 days which means that a refresh token expires after 90 days of inactivity. The `MaxAgeSingleFactor` and `MaxAgeMultiFactor` are also related to refresh token and define the maximum lifetime of a refresh token, based on the single or multi-factor authentication setting of your organization.

8. If you want to add or update your Azure AD token lifetime settings, you need to decide on the new settings and execute following (updating the lifetimes as you wish):

```
$newTokenPolicy = @('{
  "TokenLifetimePolicy":
  {
    "Version":1,
    "AccessTokenLifetime":"0.01:00:00",
    "MaxInactiveTime":"90.00:00:00",
    "MaxAgeSingleFactor":"until-revoked",
    "MaxAgeMultiFactor":"until-revoked",
    "MaxAgeSessionSingleFactor":"until-revoked",
    "MaxAgeSessionMultiFactor":"until-revoked"
  }
}')
```



**Note:** As of January 30, 2021 you can no longer configure refresh and session token lifetimes, as Microsoft Entra no longer honors refresh and session token configuration in existing policies. New tokens issued after existing tokens have expired are now set to the default configuration. You can still configure access, SAML, and ID token lifetimes after the refresh and session token configuration retirement, but the existing token's lifetime will not be changed. After they expire, a new token is issued based on the default value.

9. And if you had a token policy, execute the following command to update it.

```
Set-AzureADPolicy -Id $defaultTokenPolicy.Id -DisplayName
"OrganizationDefaultPolicyUpdatedScenario"
-Definition $newTokenPolicy
```

10. To validate that the policy has been applied correctly, execute steps 3 through 5.

# Creating Connections Using the Microsoft Office 365 (Deprecated) Connector




**Important:** The instructions in this section are for creating a connection to Microsoft Office 365 using the legacy **Microsoft Office 365 (deprecated)** connector. However, the **Microsoft 365** connector is recommended connector to use to create a connection to Microsoft 365 on an inventory beacon. For those instructions, refer to [Using IT Asset Management's Multi-Tenant App to Connect to Microsoft 365](#) or see [Registering an App Using the Azure Portal to Connect to Microsoft 365](#). For those instructions, refer to [Using Flexera's Multi-Tenant App to Connect to Microsoft 365](#), or see [Registering an App to Connect to Microsoft 365 Using the Azure Portal](#).

Use the following procedure to create a connection to Microsoft 365 on an inventory beacon using the legacy **Microsoft Office 365 (deprecated)** connector. A separate connection is required for each tenant of Microsoft 365. (Typically, there is one tenant per enterprise; but your corporate history, particularly of mergers and acquisitions, may mean that your enterprise has multiple Microsoft tenants.) The inventory beacon requires this connection to import entitlements, users, and usage information from the Microsoft 365 online account. Each per-tenant import covers all subscriptions for that tenant.



## To create a connection to Microsoft 365:

1. Ensure that you have your preferred schedule for imports from Microsoft 365 set on the appropriate inventory beacon:
    - a. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).
- 
-  **Tip:** Remember that you must run the inventory beacon software with administrator privileges.
- b. From the **Data collection** group in the navigation bar, choose **Scheduling**.
    - c. If there is not already a suitable schedule in the list, click **New...** and complete the details (see [Creating a Data Gathering Schedule](#) for more information)(see online help for more information). Otherwise, identify the schedule you will use.
  2. Select the **Inventory Systems** tab (in the same navigation group).
  3. Choose either of the following:
    - To change the settings for a previously-defined connection, select that connection from the list, and click **Edit....**
    - To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.
  4. Complete (or modify) the values for the following required fields:
    - **Connection Name:** The name of the inventory connection. When the data import through this connection is executed, the data import task name is same as the connection name.
    - **Source Type:** Select **Microsoft Office 365 (deprecated)** from this list.





**Note:** If you have not installed the FlexNet Beacon released with IT Asset Management 2019 R1 or later, then **Microsoft Office 365 (deprecated)** will not appear as a **Source Type** connection. Instead, **Microsoft Office 365** will appear as the legacy connector source type.

5. Optionally, if your enterprise uses a proxy server to enable Internet access, complete (or modify) the values in the **Proxy Settings** section of the dialog box in order to configure the proxy server connection.
  - **Use Proxy:** Select this check box if your enterprise uses a proxy server to enable Internet access. Complete the additional fields in the **Proxy Settings** section, as needed. If the **Use Proxy** check box is not selected, the remaining fields in the **Proxy Settings** section are disabled.
  - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`. This field is enabled when the **Use Proxy** check box is selected.
  - **Username and Password:** If your enterprise is using an authenticated proxy, specify the username and password of an account that has credentials to access the proxy server that is specified in the **Proxy Server** field. These fields are enabled when the **Use Proxy** check box is selected.
6. Complete (or modify) the values in the **Microsoft 365** section of the dialog. All of the following values are required:



**Note:** If you have multiple tenants within Microsoft 365 (for example, separate subscriptions for different corporate units or locations), you need to create a separate connector for each tenant using its own credentials. Within each tenant, a single connection and a single import recovers data for all your subscriptions (if you have multiple subscriptions).

- **Username:** Microsoft 365 tenant user name.
  - **Password:** Microsoft 365 tenant password.
7. **Save** the connection.
  8. Select your new connection from the displayed list, and click **Schedule....**
  9. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.
  10. At the bottom of the **FlexNet Beacon** interface, click **Save**, and if you are done, also click **Exit**.



**Tip:** Consider whether you want to select your connection, and click **Execute Now**, before you exit.

After a successful data import, the users, applications, licenses, and usage data are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see *Inventory System Tab* in the online help. For scheduling data imports through this connection, see **Scheduling a Connection**, also in help.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see *Inventory Systems Tab*. For more about scheduling data imports, see *Scheduling a Connection*.

IT Asset Management (Cloud)  
Current

# Troubleshooting Imports from Microsoft 365


IT Asset Management (Cloud)

The following sections provide help with troubleshooting imports from Microsoft 365:

- [Troubleshooting Microsoft 365 Connector Imports from Microsoft 365](#)
- [Troubleshooting Microsoft Office 365 \(Deprecated\) Connector Imports from Office 365](#)

## Troubleshooting Microsoft 365 Connector Imports from Microsoft 365

IT Asset Management (Cloud)

 **Important:** The troubleshooting in this section applies to connections to Microsoft 365 that use the **Microsoft 365** connector. For help with troubleshooting connections using the **Microsoft Office 365 (deprecated)** connector, see [Troubleshooting Microsoft Office 365 \(Deprecated\) Connector Imports from Office 365](#).

Symptom	Recommendation
The <b>Authorization Endpoint</b> , <b>Token Endpoint</b> , <b>Application (client) ID</b> or <b>Redirect URI</b> values are blank in the PowerShell connection dialog.	<p>Check whether you are using a FlexNet Beacon older than the one included with IT Asset Management 2019 R1.. If so, you will need to manually enter these values:</p> <ul style="list-style-type: none"><li>• <b>Authorization Endpoint:</b> <code>https://login.microsoftonline.com/common/oauth2/v2.0/authorize</code></li><li>• <b>Token Endpoint:</b> <code>https://login.microsoftonline.com/common/oauth2/v2.0/token</code></li><li>• <b>Application (client) ID:</b> <code>5bb1a5a2-0d97-4335-9448-119f7b27aff9</code></li><li>• <b>Redirect URI:</b> <code>https://login.microsoftonline.com/common/oauth2/nativeclient</code></li></ul>

Symptom	Recommendation
A blank pop-up screen appears when you click the <b>Generate...</b> button (next to the <b>Refresh Token</b> field) when attempting to generate a refresh token that will be used to integrate with Microsoft 365.	<p>You most likely need to set the PowerShell execution policy. Run PowerShell with administrator rights to execute the following command:</p> <pre>Set-ExecutionPolicy RemoteSigned</pre>
You get a <b>Need admin approval</b> dialog saying that the FlexNet Beacon needs permission to access resources in your organization that only an admin can grant.	<p>Ensure that the account used to connect to the Microsoft 365 tenant(s) has the <b>Cloud application administrator</b> role. This role is required in order for the FlexNet Beacon to retrieve a token that allows read only access to Microsoft Graph. For more information, see <a href="https://docs.microsoft.com/en-us/azure/active-directory/users-groups-roles/directory-assign-admin-roles">https://docs.microsoft.com/en-us/azure/active-directory/users-groups-roles/directory-assign-admin-roles</a>.</p>
You receive an authorization error that informs that you are not authorized to access this site.	<p>This may be because the machines do not have permissions to access the Microsoft Authentication site. Ensure that you provide proxy information if the machine uses proxy settings. Also, open a Web browser, navigate to <a href="https://login.microsoftonline.com">https://login.microsoftonline.com</a> and log on when asked, to validate that you can successfully authenticate on this machine.</p> <p>If you still see issues, contact Flexera with information from this and other steps. Note the steps that caused your issue and save any necessary screenshots to report to Flexera. Also include what kind of authentication service is used in your organization and what region and country the machine and Microsoft 365 tenant reside. Note that some regions (e.g., Germany) have separate login URLs as explained in <a href="https://docs.microsoft.com/en-us/azure/active-directory/develop/authentication-national-cloud">https://docs.microsoft.com/en-us/azure/active-directory/develop/authentication-national-cloud</a>.</p>
You receive an <b>Inventory gathering</b> or <b>Usage</b> error.	<p>Errors like <b>Inventory gathering failed. Error: The remote server returned an error: (403) Forbidden</b> may occur when the <b>Reports Reader</b> and <b>Cloud Application Administrator</b> roles privilege are not present for the account used to generate your token.</p>

## Troubleshooting Microsoft Office 365 (Deprecated) Connector Imports from Office 365

IT Asset Management (Cloud)



**Important:** The troubleshooting in this section applies to connections to Microsoft Office 365 using the **Microsoft Office 365 (deprecated)** connector. For help with troubleshooting connections using the **Microsoft Office 365** connector, see [Troubleshooting Microsoft 365 Connector Imports from Microsoft 365](#).

This process traces the path of data imported from Microsoft 365 Online Service through an inventory beacon to the central application server of IT Asset Management. You may pick up at any stage of the process as needed for your particular symptoms.

**To troubleshoot imports from Microsoft 365 Online Service:**

1. Validate that you have licensed the FlexNet Manager for Datacenters product:
  - a. Go to the **IT Assets License** page (**Administration > IT Asset Management Settings > IT Assets License**).
  - b. In the **Licensed products** section, scroll down to see the card for FlexNet Manager for Datacenters.

Inventory from a connection to Microsoft 365 Online Service cannot be imported or uploaded by an inventory beacon without this license term being registered on the central application server. If it is missing, please contact your Flexera representative for assistance.

2. Validate imports to the inventory beacon:
  - a. On the inventory beacon that connects to Microsoft 365, log into FlexNet Beacon using an account with administrator privileges.
  - b. On the **Inventory Systems** page, select your connection to Microsoft 365, and click **Execute Now**  
A confirmation shows that the extract has started. A typical import may take 2-3 minutes.
  - c. In the very top-left corner of the FlexNet Beacon interface, right-click the menu icon, and select **Open log file folder**.  
File Explorer opens.
  - d. Ensure that you are viewing the ProgramData\Flexera Software\Compliance\Logging\ComplianceReader folder.
  - e. In Notepad, open the text document importer-[nnnn] that was modified on today's date, and scroll to the bottom.  
If the reading of data is still in progress, exit from Notepad, and try again in a few minutes.
  - f. A successful process shows the following at the end of the log file (with each line prefixed with the date and time):

```
[INFO] 0 source data warnings
[INFO] 0 errors, 0 warnings
[INFO] Import has been completed successfully
```

If, instead, there are warnings or errors in the log file, try to correct the issues. If all else fails, save the log file to include with a Support ticket for Flexera. Typical issues at this stage may include:

- Proxy or firewall settings preventing access to the Internet
- Intermittent issues with the Internet or access to the Microsoft 365 portal
- Incorrect details for the account name or password accessing the portal (see [Creating Connections Using the Microsoft Office 365 \(Deprecated\) Connector](#) for details of configuring this account)
- The connection has not been regularly scheduled (see the same topic), and so succeeds only when executed manually.

Immediately after a successful import is completed, the inventory beacon attempts to upload the archive of imported data to its parent (the parent may be another inventory beacon in the hierarchy, or the central application server):

- When the upload succeeds, the inventory beacon also saves a copy in C:\ProgramData\Flexera Software\Beacon\IntermediateData\Uploaded, and preserves this copy for 14 days
- If the upload fails, the imported archive stays in C:\ProgramData\Flexera Software\Beacon\IntermediateData.



**Tip:** If you are working on a disconnected inventory beacon (one that cannot access the central application server), this *IntermediateData* folder contains the data files that you must transfer to another location where uploads are possible.

- g. To review the data imported from Microsoft 365, navigate to C:\ProgramData\Flexera Software\Beacon\IntermediateData\Uploaded, and inspect the most recent relevant .zip file there.



**Tip:** If the zip archive remains in the *IntermediateData* folder (on a connected inventory beacon), there is an upload problem, as discussed below.

3. To debug uploads from this inventory beacon to its parent, review C:\ProgramData\Flexera Software\Compliance\Logging\ComplianceUpload\upload.log for details of any networking issues, and note the URL of the next server in the upload chain. If this is another inventory beacon, repeat these reviews there, and so on until you exhaust the hierarchy of inventory beacons.

The upload URL is in a log line that starts:

```
dateTime [Upload.UploadProcess ] [INFO] Uploading to http...
```


4. If you are unable to resolve problems, collect your logs together in a zip archive. Ask your registered support contact (a designated person within your enterprise who has access rights and login details) to open a new support case at <https://community.flexera.com/t5/forums/postpage/board-id/@support>, including a clear description of the issue. Once the case has been saved, your support contact can use the **Upload** button (in the **Attachments** section at the bottom) to attach your prepared zip archive of logs.

## 4

# Migrating to a New Microsoft Connector

IT Asset Management (Cloud)

---


 **Important:** The instructions in this section are only applicable if you have previously used the **Microsoft Office 365** (or **Microsoft Office 365 (deprecated)**) connector, have validated the data coming from the new **Microsoft 365** connector, and are ready to switch over completely to the new connector. There are a few basic steps to follow to migrate licenses and relink contracts and purchase orders to the new license.

---

 **To migrate from an older to newer Microsoft connector, migrate licenses, and relink contracts and purchase orders to the new license:**

1. Check your version of the FlexNet Beacon.

---

 **Note:** The FlexNet Beacon released with IT Asset Management 2018 R2 (13.1.1) or later is required to use the **Microsoft 365** connector. However, installing the FlexNet Beacon included in the 2019 R1 (13.2.0) or later release provides maximum ease-of-use with the **Microsoft 365** connector by including auto-populated values on the **Create PowerShell Source Connection** dialog (that otherwise need to be entered manually). In addition, if you are using a FlexNet Beacon released prior to IT Asset Management 2019 R1, then **Microsoft Office 365 (deprecated)** will not appear as a **Source Type** connection on the **Create PowerShell Source Connection** dialog, and instead will remain **Microsoft Office 365**.

2. Delete the connection to the old **Microsoft Office 365** or **Microsoft Office 365 (deprecated)** connector.

---

 **Note:** Although the legacy connector name is now **Microsoft Office 365 (deprecated)**, the previous name of the this connector's **Source Type** was **Microsoft Office 365**.

3. Relink contracts and purchase orders to the new license.

# XI

## Microsoft App-V Server Adapter

### IT Asset Management (Cloud)

Microsoft App-V (full name Microsoft Application Virtualization) is an application virtualization and application streaming solution. It allows access to applications in three different ways:

- Users may stream applications directly from a central App-V Management Server to their client computers, executing the code locally in light virtual machines that provide a protective 'bubble' around the executing software.
- The applications may be deployed using Microsoft Endpoint Configuration Manager (previously Microsoft SCCM).
- Applications may be deployed in 'stand alone' mode, such as manual delivery through file shares or on a USB stick, without the use of any server infrastructure.

Applications delivered in any of these ways require licensing, and you can manage the appropriate licenses using IT Asset Management:

- Applications streamed from the App-V Publishing Server(s) are monitored using the App-V server adapter supplied as a standard part of IT Asset Management, and (for App-V release 5.0 and later) the `AppVMgmtSvr.ps1` PowerShell script installed on the App-V Management Server. Both the App-V server adapter and the `AppVMgmtSvr.ps1` PowerShell script are documented in this chapter.
- Applications deployed using Microsoft Endpoint Configuration Manager (previously Microsoft SCCM) are recorded automatically as part of the standard inventory import from Microsoft Endpoint Configuration Manager.
- Applications deployed manually can be recorded manually in IT Asset Management. Manual deployment is not a recommended best practice because of the inherent difficulties in management and demonstrating compliance, and there is no automation possible through FlexNet direct inventory gathering to cover manual deployment.

### Supported versions

The App-V server adapter supports the current version of IT Asset Management, and releases 4.6, 5.0, and 5.1 of Microsoft Application Virtualization.

Because of significant architectural change between release 4.6 and release 5.0 of App-V, there are matching significant differences in the App-V server adapter for the different versions. It is important to read this document carefully, noting the differences that apply to "release 4.6" and "release 5.0 and later".

# 1

## Architecture, Components, and Prerequisites

### IT Asset Management (Cloud)

Following the changes that Microsoft made in the architecture of App-V between release 4.6 and release 5.0, the App-V server adapter is also significantly different when interfacing to the different App-V releases. There are separate chapters for each different architecture. Identify the release of Microsoft App-V in use in your enterprise, and focus on the architecture that is appropriate to that release.

## Architecture and Operation for App-V 4.6

### IT Asset Management (Cloud)

This discussion applies to use of Microsoft App-V server infrastructure, streaming applications to App-V clients on end-point devices. (Where applications are instead installed by Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), use the inventory import from Microsoft Endpoint Configuration Manager instead of this adapter.)

In its streaming implementation, Microsoft App-V release 4.6 has three main kinds of components:

- A database (referred to here as the App-V Management Server database), which may be on a separate server
- One or more Management Servers that access the App-V Management Server database and provide a user interface for system control
- One or more streaming servers that may directly deliver application packages.

Of these, only the App-V Management Server database is relevant to the App-V server adapter for IT Asset Management.

### Prerequisites

Operation requires that you have:

- A supported version of Microsoft App-V (see [Microsoft App-V Server Adapter](#)).
- An operational App-V Management Server database.
- A FlexNet inventory beacon that has network access to your App-V Management Server database, and is also able to



upload gathered inventory to the central IT Asset Management server (either directly or through a hierarchy of inventory beacons).

- An inventory beacon importing Active Directory data from the same domain where the App-V server resides. (This may be the same inventory beacon that runs the App-V server adapter, but this is not a requirement.)



**Tip:** If you have App-V applications secured by security groups from multiple Active Directory domains, ensure that the Active Directory import runs against all applicable domains in your environment. The simplest approach may well be to ensure that you import from all your Active Directory domains, since if you use foreign security principals from multiple trusted domains, it can be difficult to keep track of access to App-V packages. FlexNet Manager Suite imports only from each individually specified Active Directory domain; so you need to ensure that all applicable domains are specified. As an example of multiple domains being affected:

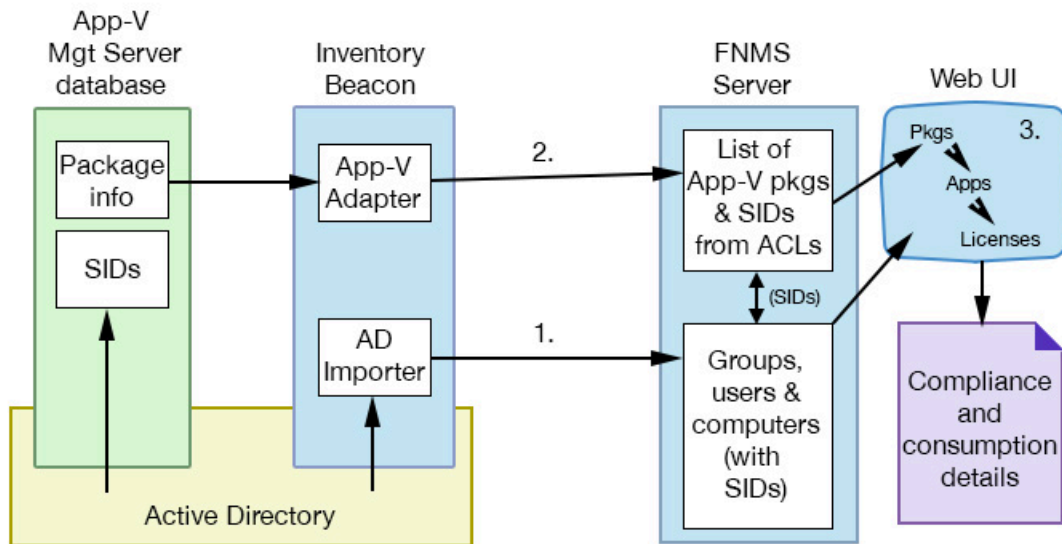
- Suppose you have Group-A in Domain-A that contains a child Group-B, where Group-B actually comes from Domain-B.
  - In this case, granting access to an App-V package to Group-A also grants access to Group-B (because of the parent-child relationship between the groups).
  - This inheritance continues to work even when there is only one-way trust from Domain-B to Domain-A.
  - In such a case, it is imperative that you run an Active Directory import against both Domain-A and Domain-B. When you have many domains, the simplest path is just to run an Active Directory import from every domain.
- Operators who can identify the applications represented by the App-V packages, and link those applications to the appropriate licenses.



**Tip:** You may have multiple App-V Management Servers, and multiple streaming servers, that link to a single App-V Management Server database. This requires only one connection from the IT Asset Management App-V server adapter, because this connects only to the database. However, if you have multiple App-V Management Server databases in your estate, configure a separate connection to each of them on appropriate inventory beacons. Where helpful, you may configure multiple such connections (each separately scheduled as you choose) on one inventory beacon.

## In operation

The following diagram shows the operational architecture for the App-V server adapter for App-V release 4.6.



The numbers here refer to the numbers shown in the diagram above:

1. The inventory beacon imports data from Active Directory, including groups (and their members), users, and computers, and the security identifiers for each item within Active Directory. (These security identifiers, or SIDs, are the same identifiers that App-V reports for usage of the applications delivered through App-V packages.)
  - These are immediately uploaded to the central application server for IT Asset Management.
  - As soon as the upload is completed, the data is imported into the compliance database.
2. On the schedule you specify on the inventory beacon, the App-V adapter:
  - Connects to the App-V Management Server database
  - Imports a list of the App-V packages from the database, and the access control lists (ACLs) that determine which Active Directory groups and users have access to the applications inside the packages. The latter are identified by their security identifiers (SIDs).
  - Immediately uploads the data to the central application server for IT Asset Management. (If the upload fails for some reason, there is a catch-up upload task that by default is scheduled overnight.)
  - The data waits in the staging area on the central application server for the next scheduled inventory import and compliance calculation (by default, scheduled overnight).
3. When information about a new App-V package is first imported, an operator must identify the package and link it (like installer evidence) to an application record. This work must be done manually because (in release 4.6) App-V packages are opaque about the applications they contain. As well, for any meaningful calculations of consumption, the application must be linked to a suitable license. This linking effort is required only for the first import of each new package.

Once the links are established, each subsequent compliance calculation assigns consumption by the correct users and computers to the appropriate (linked) license. This consumption information is then available both in the management views and in reports.

## Architecture and Operation for App-V 5 and

# Later

## IT Asset Management (Cloud)

This discussion applies to use of Microsoft App-V server infrastructure, streaming applications to App-V clients on end-point devices. (Where applications are instead installed by Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), use the inventory import from Microsoft Endpoint Configuration Manager instead of this adapter.)

In its streaming implementation, Microsoft App-V release 5.0 (and later) has the following components, apart from the App-V clients:

- A database (referred to here as the App-V Management Server database), which may be on a separate server
- A separate reporting database (referred to here as the App-V reporting database), which may also be on a separate server (importantly, this database stores application usage information)
- One or more Management Servers that access the App-V Management Server database and provide a user interface for system control
- One or more Reporting Servers that access the App-V reporting database and provide operational reports to help manage the App-V infrastructure
- One or more streaming servers (called App-V Publishing Servers) that may directly deliver application packages.

Of these, for App-V 5.0 and later, only the App-V reporting database and an App-V Management Server are relevant to the App-V server adapter for IT Asset Management. (If you are familiar with the adapter for release 4.6 of App-V, notice that we have switched databases, and added the Management Server — the architecture is completely different.)

## Prerequisites

Operation requires that you have:

- A supported version of Microsoft App-V (see [Microsoft App-V Server Adapter](#)).
- An operational App-V reporting database.
- An operational AppV Management Server.
- The AppVMgmtSvr.ps1 PowerShell script installed, configured and scheduled on your AppV Management Server (see [Obtaining \(and Deploying\) the Adapter Components](#) for details). This is one of the significant changes since the previous adapter.
- A FlexNet inventory beacon that has network access to your App-V reporting database, and is also able to upload gathered inventory to the central IT Asset Management server (either directly or through a hierarchy of inventory beacons).
- An inventory beacon importing Active Directory data from the same domain where the App-V server resides. (This may be the *same* inventory beacon that runs the App-V server adapter, but this is not a requirement.)



**Tip:** If you have App-V applications secured by security groups from multiple Active Directory domains, ensure that the Active Directory import runs against all applicable domains in your environment. The simplest approach may well be to ensure that you import from all your Active Directory domains, since if you use foreign security principals from multiple trusted domains, it can be difficult to keep track of access to App-V packages. FlexNet Manager Suite

---

*imports only from each individually specified Active Directory domain; so you need to ensure that all applicable domains are specified. As an example of multiple domains being affected:*

- *Suppose you have Group-A in Domain-A that contains a child Group-B, where Group-B actually comes from Domain-B.*
  - *In this case, granting access to an App-V package to Group-A also grants access to Group-B (because of the parent-child relationship between the groups).*
  - *This inheritance continues to work even when there is only one-way trust from Domain-B to Domain-A.*
  - *In such a case, it is imperative that you run an Active Directory import against both Domain-A and Domain-B. When you have many domains, the simplest path is just to run an Active Directory import from every domain.*
- Operators who can link the applications identified in the App-V packages to the appropriate licenses.
- 



**Tip:** You need only one connection from the IT Asset Management App-V server adapter (on an inventory beacon) to the App-V reporting database. This single App-V reporting database may support multiple App-V Management Servers, and multiple Publishing Servers; but only a single connection to the database is required.

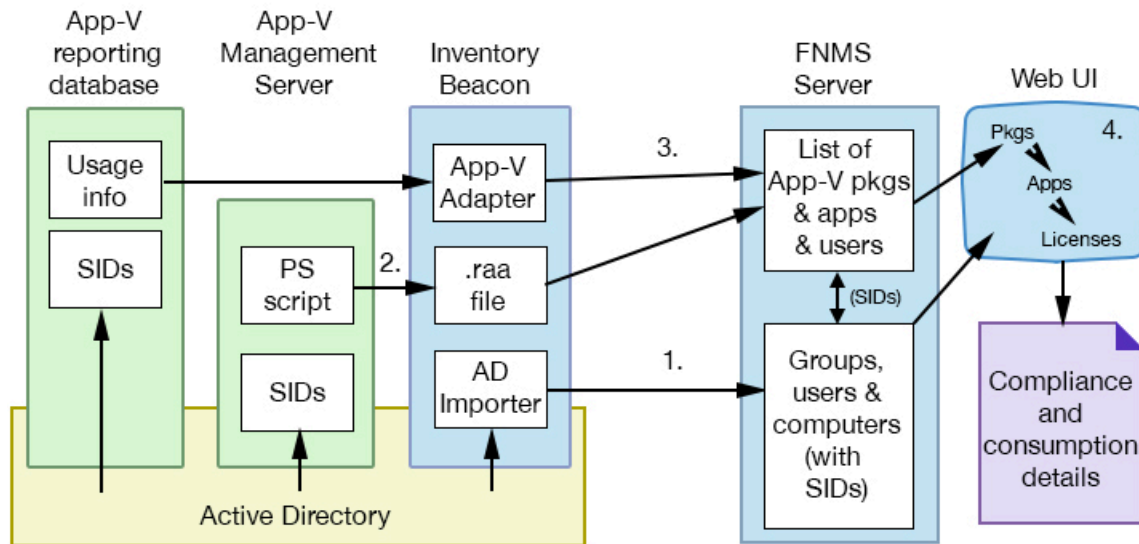
## Limitation

For App-V release 5.0 and later, the system supports installation of the AppVMgmtSvr.ps1 PowerShell script on only one App-V Management Server. This single Management Server may support multiple Publishing Servers (if necessary spread worldwide for faster distribution of App-V packages to App-V clients); and the App-V clients may report to multiple Reporting Servers (independent of the source from which the App-V packages were downloaded). Different App-V Management Servers do not self-identify in the .raa inventory file, and the App-V reporting database does not identify which application usage information is associated with which App-V Management Server. For these reasons, only a single App-V Management Server (for release 5.0 and later) is supported.

If your App-V (release 5.0 or later) environment has multiple Management Servers, choose one as the data source for App-V packages and the applications they contain. For example, if you have Production, Dev, and Test servers, place the AppVMgmtSvr.ps1 PowerShell script on the Production App-V Management Server. Also ensure that the App-V server adapter (on an inventory beacon) connects to the matching Production App-V reporting database.

## In operation

The following diagram shows the operational architecture for the App-V server adapter for release 5.0 and later.



The numbers here refer to the numbers shown in the diagram above:

1. The inventory beacon imports data from Active Directory, including groups (and their members), users, and computers, and the security identifiers for each item within Active Directory. (These security identifiers, or SIDs, are the same identifiers that App-V reports for usage of the applications delivered through App-V packages.)
  - These are immediately uploaded to the central application server for IT Asset Management.
  - As soon as the upload is completed, the data is imported into the compliance database.
2. On the schedule you specify on the App-V Management Server, the AppVMgmtSvr .ps1 PowerShell script:
  - Uses the API to gather a list of the available App-V packages
  - Imports from the database, and the access control lists (ACLs) that determine which Active Directory groups and users have access to the applications inside the packages. The latter are identified by their security identifiers (SIDs)
  - Uploads the collected data in a remote application access (.raa) file to its configured inventory beacon, which in turn uploads the file to the central application server for IT Asset Management.
  - The data waits in the staging area on the central application server for the next scheduled inventory import and compliance calculation (by default, scheduled overnight).
3. On the schedule you specify on the inventory beacon, the App-V adapter:
  - Connects to the App-V reporting database
  - Imports App-V package usage by users and computers. These are all identified by their security identifiers (SIDs).
  - Immediately uploads the data to the central application server for IT Asset Management. (If the upload fails for some reason, there is a catch-up upload task that by default is scheduled overnight.)
  - The .raa file collected by the PowerShell script is uploaded and immediately resolved into staging tables in the database.



**Tip:** If you manually copy an `.raa` file to your application server, you can import it with the following command:

```
> mgsimport -t remoteApplication
```

- The data waits in the staging area on the central application server for the next scheduled inventory import and compliance calculation (by default, scheduled overnight).
4. When the compliance calculation is run, IT Asset Management uses the uploaded SIDs to correlate the various data elements:
- App-V packages are shown as installer evidence (based on the MSI information uploaded by the `AppVMgmtSvr.ps1` PowerShell script).
  - If an appropriate application record exists (either in the Application Recognition Library or as a locally-created record) with a suitable installer evidence rule, the installed evidence (package) is automatically matched with the application.
  - All users with access to an App-V package are shown as having an installation of the related application on every computer for which the user is either the assigned or calculated user.
  - All computers with access to an App-V package are shown as having an installation of the related application.
  - If the application is linked to a license, consumption is shown for the correct users and computers on that license (or, if it is linked to multiple licenses, on the highest priority license still having unconsumed entitlements). This consumption information is then available both in the management views and in reports. (If this is the first import to reveal an application in an App-V package, an operator needs to link the application record to an appropriate license.)

## 2

# Set Up and Operations

IT Asset Management (Cloud)

This chapter covers the configuration of the adapter, and the work needed to enable application recognition from the imported inventory.

Keep clearly in mind the distinctions required for App-V release 4.6, and release 5.0 and later. For example, [Obtaining \(and Deploying\) the Adapter Components](#) documents processes needed only for release 5.0 and later; and [Configuring the Adapter](#) requires a distinct database connection in each of the cases.

## Obtaining (and Deploying) the Adapter Components

IT Asset Management (Cloud)



---

**To download the adapters archive:**

1. Download the latest Adapter Tools for IT Asset Management *version*.zip archive from the Flexera Product and License Center:

- a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
- b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of Adapter Tools for IT Asset Management 2024 R2.3.zip commences.

- c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as C:\temp on a central, accessible server).

If your browser saves the file to a default location (such as your Downloads folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the zip archive, and choose **Extract All...**
3. In the extracted archive, navigate to Adapter Tools\App-V Management Server Agent\

AppVManagementServer5.

4. Use your preferred method to deploy the AppVMgmtSvr.ps1 PowerShell script to your App-V Management Server.

You may install the script in your preferred folder.

5. Use your preferred task scheduling technology to schedule data collection by the PowerShell script.

Typically you want the .raa file uploaded to the central IT Asset Management operations databases before the system import and license calculations take place. By default, this occurs daily at 2am central server time. As a two-hour upload buffer should be more than adequate, this suggests (within a single time zone) a data collection trigger at around midnight.

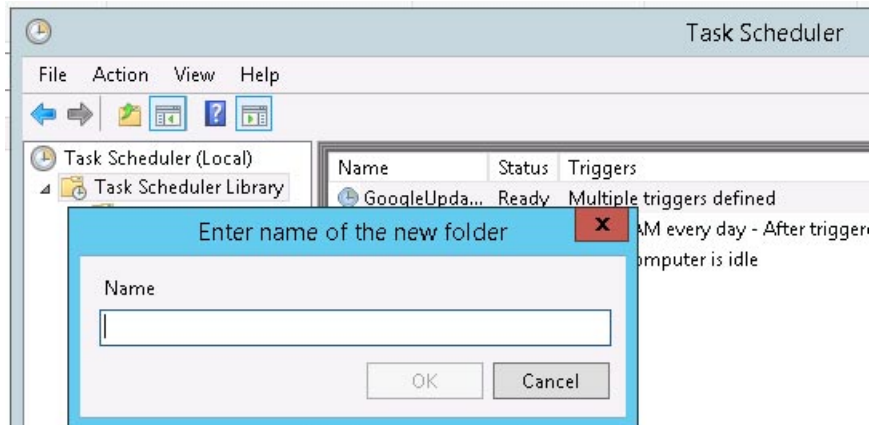
These example steps are for Windows Server 2012. Adjust as necessary for your server operating system, or your chosen scheduling tool.

- a. In Windows Explorer, navigate to **Control Panel > System and Security > Administrative Tools**, and double-click **Task Scheduler**.

The **Task Scheduler** window appears.

- b. In the navigation tree on the left, select **Task Scheduler Library**, and then in the **Actions** list on the right, click **New Folder...**

A dialog appears for entering the folder name.



A suggested value is IT Asset Management.

- c. Click **OK**, and select the new folder in the navigation tree.
- d. Select **Action > Create Task....**

The **Create Task** dialog appears.

- e. Enter an appropriate **Name**, such as FlexNet Manager Agent for App-V 5.x, and add any **Description** to help future maintenance of this task.

Your description may be something like Collects App-V data from the Management Server and uploads to an inventory beacon.

- f. Click **Change User or Group....**

The **Select User, Service Account, or Group** dialog appears.



- g.** Enter the account name that is to run the scheduled task, and click **OK**.

An appropriate account:

- Can run a Windows scheduled task on the App-V Management Server
- Can execute the PowerShell script
- Is an App-V Management Server administrator
- May conveniently be a domain account that can upload the results (using HTTP PUT) to the inventory beacon, although a separate account and password can be configured in the command line (required only where that inventory beacon is using Basic Authentication — if the inventory beacon uses anonymous authentication, ignore this requirement).

- h.** Further down in the **Security options** group, select **Run whether user is logged in or not**.

- i.** Switch to the **Triggers** tab, and click **New...**

The **New Trigger** dialog appears.

- j.** Ensure that the default setting **Begin the task On a schedule** is selected, set the parameters for the schedule, and from the **Advanced settings** group, be sure that **Enabled** is selected.

The suggested schedule is daily at or before midnight local time, but be sure that this suits the upload procedures for your enterprise.

- k.** Switch to the **Action** tab, and click **New...**

The **New Action** dialog appears.

- l.** Ensure that the default **Action**, **Start a program**, is selected, and browse to your local copy of `AppVMgmtSvr.ps1`.

- m.** In the **Add arguments (optional)** field, specify all the command-line arguments you need for the agent.

All command line arguments are documented in [Command Line for PowerShell Script](#). For common implementations, you need to define only the URL to the inventory beacon.

---

*This example uploads the .raa file to the fLexnetbeacon inventory beacon, using the credentials of the account running the scheduled task.*

```
.\AppVMgmtSvr.ps1 -beaconUrl http://fLexnetbeacon
```



**Tip:** You need to specify only the basis server in the URL. Internally, the FlexNet Beacon service briefly saves the uploaded file to %CommonAppData%\Flexera Software\Incoming\RemoteApplications before uploading the file to its parent. (If security within your enterprise prevents the PowerShell script uploading the .raa file to a inventory beacon, you can arrange an alternative method to save the .raa file to this folder on a convenient inventory beacon, and it is automatically uploaded and processed from this point.)

- n.** Click **OK**.

- o.** Optionally, make any preferred adjustments to the **Conditions** or **Settings** tabs (normally the defaults are acceptable).

- p. Click **OK** to close the **Create Task** dialog.

The new task appears in the list of scheduled tasks for this server.

- q. Right-click the new task, and click **Run** in the context menu.

This checks that the scheduled task completes successfully.

- r. Validate operations in the following ways:

- Review the log file (by default, AppVMgmtSvr.log in the same folder as the PowerShell script) for any errors or warning messages.
- Review the content of the output file (by default, FNMS\_AppV.raa in the same folder as the PowerShell script). This file contains the results of the most recent execution of the PowerShell script, and is replaced at each invocation of the script. For more details of the file format, see [File Format for .raa](#).
- If uploaded to the inventory beacon, check for the presence of the FNMS\_AppV.raa output file in %CommonAppData%\Flexera Software\Incoming\RemoteApplications on the inventory beacon. Remember that you have only a brief time window to check this before it is uploaded to the central application server and removed (by default, around 10 minutes).

The AppVMgmtSvr.ps1 PowerShell script is now configured on your App-V release 5.0 or later Management Server. You can now configure the adapter itself that runs on the inventory beacon (see [Configuring the Adapter](#)).

## Command Line for PowerShell Script

IT Asset Management (Cloud)

The AppVMgmtSvr.ps1 PowerShell script is required only when importing inventory from Microsoft App-V release 5.0 or later. (It is not required if you are using App-V release 4.6.)

For applicable releases, AppVMgmtSvr.ps1 is installed on the App-V Management Server, where, on a schedule that you determine, it collects details of the available App-V packages, and the users and computers that have access to the packages. (Separately, usage information is collected by the App-V server adapter, with its separate configuration described in [Configuring the Adapter](#).)

The following options are supported, both for running AppVMgmtSvr.ps1 manually and for executing it from a scheduled task or other scheduling tool. None of the options is mandatory, although some are required for normal operation.

### Syntax

#### Syntax:

AppVMgmtSvr.ps1 [*options...*]

### Options

```
-beaconUrl validURL
    -logFilePath log_file
```

```
-outputFilePath output_file
-password password
-upload $true | $false
-username account
```

where

`-beaconUrl validURL` The URL to the appropriate inventory beacon to which the script should upload the generated `.raa` file. Include the protocol (HTTP or HTTPS), and if your inventory beacon uses a non-default port, include the port number in the standard way.



**Note:** Include only the basic URL of the server. No internal paths are needed, as these details are added automatically by `AppVMgmtSvr.ps1`.

There is no default value for `beaconUrl`, so that in production use (when `-upload $true`), a value must be supplied. It can be omitted for local testing on the server when uploading is not required. If `-upload $true` and `beaconUrl` is not set, obviously the upload must fail.

Example values:

```
http://flexnetbeacon.example.com
https://flexnetbeacon.example.com:499
```

<code>-logFilePath</code> <code><i>log_file</i></code>	A file path (either absolute, or relative to the folder in which the script is executing) and file name for the log file generated by <code>AppVMgmtSvr.ps1</code> . Enclose the path in double quotation marks. A file with consistent name and path over time is replaced at each execution, containing only the results of the last execution, thus preventing unmanaged storage requirements. When this option is not specified, the default value is <code>AppVMgmtSvr.log</code> , saved in the folder where the script is executing.
<code>-outputFilePath</code> <code><i>output_file</i></code>	A file path (either absolute, or relative to the folder in which the script is executing) and file name for the output remote application access ( <code>.raa</code> ) file generated by <code>AppVMgmtSvr.ps1</code> . Enclose the path in double quotation marks. If the same name and path is used, the file is overwritten at each run of the script. When the option is not specified, the default value is <code>FNMS_AppV.raa</code> , saved in the folder where the script is executing.
<code>-password "<i>password</i>"</code>	The password (in plain text) for the account specified in <code>username</code> . Omitted when that option is not required. If specified, enclose the value in double quotation marks. When required and not specified, the script uses details of the account running the process.
<code>-upload \$true   \$false</code>	A Boolean that determines whether to attempt uploading the output file to an inventory beacon identified in <code>beaconUrl</code> . When not specified, the default is <code>true</code> , requiring that <code>beaconUrl</code> is specified so that the upload can succeed.

---

<code>-username "account"</code>	The account name used to upload the generated .raa file to the inventory beacon identified in <code>beaconUrl</code> . This is not required for inventory beacons using anonymous authentication. It may be specified for inventory beacons using Windows Basic Authentication. If specified, enclose the value in double quotation marks. When it is specified, the matching password must be provided in the <code>-password</code> option. When required and not specified, the script uses details of the account running the process.
----------------------------------	--

---

## Examples

(Examples here may be line-wrapped for convenient presentation; but should be entered on a single command line.)

Collect the inventory from the App-V Management Server, saving the file locally for inspection:

```
.\AppVMgmtSvr.ps1 -upload $false
```

Similarly, collect the inventory, saving the output and logs to temporary locations to avoid overwriting the normal output:

```
.\AppVMgmtSvr.ps1 -outputFilePath "C:\temp\FNMS_AppV.raa"
                  -logFilePath "C:\temp\AppVMgmtSvr.log"
                  -upload $false
```

Collect the inventory, and upload it to the specified inventory beacon, using the username and password for the account currently running the script:

```
.\AppVMgmtSvr.ps1 -beaconUrl http://flexnetbeacon.example.com
```

Upload the collected inventory to the specified inventory beacon, using the `testdomain\administrator` account:

```
.\AppVMgmtSvr.ps1 -beaconUrl http://flexnetbeacon.example.com
                  -username "testdomain\administrator"
                  -password "somepassword"
```

## File Format for .raa

IT Asset Management (Cloud)

The `AppVMgmtSvr.ps1` PowerShell script interrogates the App-V Management Server, and saves the resulting data in a remote application access file (filename extension .raa), by default called `FNMS_AppV.raa`. This is an XML file that encapsulates data about each application's App-V package, and the MSI installer information known to App-V.

Here is an excerpt from an `FNMS_AppV.raa` file (here line-wrapped for presentation):

```
<remoteApplications accessModeID="2">
  <app farmName=""
    appID="9b09bc0d-9634-4838-b0ba-8c256ef4710d"
    appName="Blender"
    appFileName=""
    appFileVersion=""
```

```

    appFilePublisher=""
    appFileDesc=""
    userSid=""
    serverName=""
    serverDomainName=""
    isStreamingProfile="1" />
<msiData farmName=""
    appID="9b09bc0d-9634-4838-b0ba-8c256ef4710d"
    msiDisplayName="Blender"
    msiPublisher="Blender"
    msiVersion="1.0"
    msiProductCode="{DFFFE0C6-3E9A-44E9-9EC1-B5C92DCEE4AF}" />
<app farmName=""
    appID="5d80bef6-de4a-44f6-b4f8-9aa6a657880e"
    appName="FileZilla_3.2.4.1_win32-setup"
    appFileName=""
    appFileVersion=""
    appFilePublisher=""
    appFileDesc=""
    userSid="S-1-5-21-1336908958-3350896562-3141117955-1690"
    serverName=""
    serverDomainName=""
    isStreamingProfile="1" />
<msiData farmName=""
    appID="5d80bef6-de4a-44f6-b4f8-9aa6a657880e"
    msiDisplayName="FileZilla Client 3.2.4.1"
    msiPublisher=""
    msiVersion="3.2.4.1"
    msiProductCode="filezilla client" />
<app farmName=""
    appID="02787044-d434-4e7d-8770-cda46c988de8"
    appName="AdobeReader9"
    appFileName=""
    appFileVersion=""
    appFilePublisher=""
    appFileDesc=""
    userSid="S-1-5-21-1336908958-3350896562-3141117955-1690"
    serverName=""
    serverDomainName=""
    isStreamingProfile="1" />
<msiData farmName=""
    appID="02787044-d434-4e7d-8770-cda46c988de8"
    msiDisplayName="Adobe Reader 9.1"
    msiPublisher="Adobe Systems Incorporated"
    msiVersion="9.1.0"
    msiProductCode="{ac76ba86-7ad7-1033-7b44-a91000000001}" />
...
</remoteApplications>

```

Some points of interest to note:

- The complete set of MSI attributes needed for IT Asset Management makes use of Asset Intelligence on the App-V release 5.0+ system, which saves a series of `AssetIntelligenceProperties` in the manifest file. From these properties, `AppVMgmtSvr.ps1` extracts the four `msi...` properties shown for the applications above.
- Not all applications deployed through App-V use MSI, as some applications use third-party installers. App-V packages using third-party installers cannot include the Asset Intelligence properties available through MSI. For such cases, the `AppVMgmtSvr.ps1` PowerShell script interrogates the App-V application registry to get complete installer evidence, populating the same attributes in the `.raa` file. (Thus the presence of `AssetIntelligenceProperties` in the manifest file does not necessarily mean that Asset Intelligence was available; but does mean that equivalent data has been obtained.)
- The `msiDisplayName` becomes the **Name** property of the installer evidence record, the `msiPublisher` maps to **Publisher**, `msiVersion` maps to **Version**. As with all installer evidence, the resulting record is matched against installer evidence "rules" (previously-recorded installer evidence records, most often generalized with judicious use of wild card % characters), and when matched, adds an installation count to the application linked to the rule. This means that in listings of installer evidence, the visible entry remains the generalized rule that matched our individual piece of installer evidence. However, the individual installer evidence record is visible by drilling down into the properties of the inventory device on which the inventory evidence was found.
- For a worked example of how the installer evidence for the FileZilla application in the extract above is matched against an installer evidence rule, see [Investigating Issues](#).

## Configuring the Adapter

IT Asset Management (Cloud)

The (core) App-V server adapter is set up on an inventory beacon. Only two tasks are required for configuring this built-in adapter:

- Specifying the connection to the appropriate database. There are distinct databases in the two versions:
  - For App-V release 4.6, the App-V server adapter connects to the Microsoft App-V Management Server database
  - For App-V release 5.0 and later, the App-V server adapter connects to the Microsoft App-V reporting database.
- Scheduling the imports.

Both tasks are summarized here. Further details are available in the inventory beacon help.



### **To configure the App-V server adapter (summary):**

1. On the appropriate inventory beacon, start the FlexNet Beacon interface.

An appropriate inventory beacon has network access to the appropriate database as described above; and it can upload to the central IT Asset Management server, either directly or through a hierarchy of inventory beacons.





**Tip:** Remember that logging into an inventory beacon requires an account with administrator privileges.

2. Select the **Inventory systems** tab in the FlexNet Beacon interface.

3. At the bottom of the page, click **New....**

The **Create SQL Source Connection** dialog opens.

4. Complete the values required in this dialog:

<b>Connection name</b>	A descriptive name for this connection that you will recognize later in lists.
<b>Source Type</b>	<p>Select <b>App-V Standalone</b>. (Use this same value whether you are connecting to App-V release 4.6, or release 5.0 or later.)</p> <hr/> <p> <b>Tip:</b> 'Standalone' means that you are using the adapter to connect directly to the appropriate App-V database, rather than collecting inventory through another source such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM).</p>
<b>Server</b>	Type the server name or IP address where the database is hosted. If the database instance you need is not the default one on the server you identify, add the instance name, separated with a backslash character.
<b>Authentication</b>	<p>Select one of:</p> <ul style="list-style-type: none"> <li>• <b>Windows Authentication</b> — Select this option to use standard Windows authentication to access the database server. The credentials of the account (on the inventory beacon) running the scheduled task for importing inventory are used to access the SQL Server database. This account must be added to an Active Directory security group that has access to the database.</li> <li>• <b>Windows (specific account)</b> — Use the following two fields (enabled when you make this choice) to specify an account on the inventory beacon that can make a connection to the SQL database.</li> <li>• <b>SQL Authentication</b> — Use the following two fields to specify an account and password registered as a user with database access on SQL Server . This account is used to access the database, regardless of the local account running the scheduled task on the inventory beacon server.</li> </ul> <hr/> <p> <b>Tip:</b> The account used needs read-only privileges.</p>
<b>Username</b>	The account name used for <b>SQL authentication</b> , or <b>Windows (specific account)</b> . (Not required for <b>Windows Authentication</b> .)
<b>Password</b>	The password for the account name required for <b>SQL authentication</b> , or <b>Windows (specific account)</b> . (Not required for <b>Windows Authentication</b> .)
<b>Database</b>	Enter the name of the database, or use the pull-down list to select from database names automatically detected on your specified server. For example, for a connection to Technopedia, select BDNA_Publish from the drop-down list.

**Connection is in test mode (do not import results)**

Ensure that this check box is clear for production use. (For more details, see *Managing Microsoft SQL Server Database Connections* in the online help for IT Asset Management, in the section covering the inventory beacon.)



**Tip:** When using App-V release 4.6, you cannot complete configuration for operation of this adapter (specifically, you cannot map the App-V packages to real applications) until you have run an import in production mode, with this check box clear.



## Overlapping Inventory Filter

If you use more than one inventory source, it is possible to get overlapping inventory (records about the same endpoint device from multiple inventory tools). Because of differences between inventory tools, the overlapping inventory records may contain slightly different data. In the web interface (in IT Asset Management), you may nominate one inventory source as **Primary**, which gives its collected data priority for hardware properties imported from target inventory devices. The choices here give another, separate axis of control, based on whether or not the data from this particular source is to be considered "stale". This is different from the **Primary** setting in the following ways:

- It is assessed *first* (before the primary setting is taken into account), and so may even modify the effect of your **Primary** setting. It is best practice to make sure that your chosen primary inventory source is *not* marked as stale with the following settings.
- It affects not only imported *hardware* properties, but also *software* inventory (installer evidence, file evidence, and so on), as follows:
  - With two inventory sources (when neither one is considered stale), the total software inventory is a union (merging) of the results from both sources. This is useful when two different inventory tools have different specializations for software detection: the union means you are not blind-sided by a software inventory tool that missed something you should have licensed. This is considered the 'normal' operational case in a stable environment using multiple inventory sources.
  - When one of the two available sources is declared "stale" (using either of the first two choices below), all of its overlapping software inventory is excluded from the possible union of data sources (hardware inventory too, but here we're considering software). This behavior is valuable, for instance, when a target inventory device has migrated from one inventory source to another (perhaps by moving offices), but has not yet been obsoleted from this first source. Imagine that, as part of the office move, the MyApp application had also been uninstalled from the inventory device. You do not want the old and stale record from this source insisting there's an installation of MyApp when in fact it's no longer on the device. Of course, best practice is to obsolete the device from this stale source so that it is no longer reported in this inventory source; but these first two settings allow a quick control of *all* the overlapping data from this source, rather than having to obsolete devices one by one in the source tool. Another common scenario is when you are migrating over time from an old version of your chosen inventory tool to a newer version (on a new connection), and during transition both systems are still running. Declaring the old version's connection as "stale" means that as soon as a record appears in the new version's inventory for a particular device, the old device import is automatically superseded, and updates rely entirely on the new version of your inventory tool.

Each of the following settings only takes affect when IT Asset Management is comparing the inventory dates of a device from this current inventory source and an

overlapping record from another inventory source:

- **Ignore the device's inventory from this data source** — When you have *more recent* inventory from another source for the same target inventory device, the record from this source is completely ignored. (Technically, the device record is deleted from the staging table in the database, and so can never be imported.)
- **Ignore this device's inventory if older than *nn* days** — If you select this option, overlapping inventory collected by this source more than the set number of days before the import is ignored. Fresher overlapping data is still imported and considered for data merging.



**Tip:** *In the interests of keeping inventory current, this control has a maximum value of 60 days.*

- **Import the inventory from this source for possible merging** — Choose this option (the default) to declare that overlapping inventory collected from this connection is never considered stale. This is the normal operating setting when you are not trying to manage transitions from one inventory source to another. With this setting, overlapping records are merged in this way:
  - a. If a data point exists in the **Primary** inventory source, it is used
  - b. If two equal-priority sources have different inventory dates, the data point is taken from the most recent inventory
  - c. As a tie-breaker, the connection ID for this source recorded in the database is used (normally meaning that the earliest-created inventory source has priority).

##### 5. Click **Test Connection**.

- If the inventory beacon can successfully connect to the nominated database using the details supplied, a `Database connection succeeded` message displays. Click **OK** to close the message. Click **Save** to complete the addition. The connection is added to (or updated in) the list.
- If the inventory beacon cannot connect, a `Database connection failed` message is displayed, with information about why that connection could not be made. Click **OK** to close the message. Edit the connection details and retest the connection.

You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.

6. If you do not already have a schedule specified that can be used to run the adapter for this connection, create one now (see *Creating a Data Gathering Schedule* in the online help).
7. With the connection for this adapter selected in the **Inventory systems** page, click **Schedule....**

The **Select Schedule** dialog opens.

8. From the drop-down list, select the schedule you wish to apply.



**Tip:** As you select each schedule from the list, the area below displays a summary of the schedule settings and the expected **Next run time** for this schedule.

9. Click **OK** to apply the selected schedule.

10. Click **Save** to store these details.

The list of connections is updated, and the **Next run** column for your selected connection shows the projected run time from the schedule you just attached.

11. With the connection for this adapter still selected in the **Inventory systems** page, click **Execute now**.

The adapter collects information from the appropriate database (App-V Management Server database for App-V release 4.6, and App-V reporting database for App-V release 5.0 and later), packages it, and uploads it to the central IT Asset Management (or, if it uploads to a parent inventory beacon, the file is briefly saved in %CommonAppData%\Flexera Software\Incoming\RemoteApplications on the parent inventory beacon). Allow time for this process to complete before continuing with the next setup procedure.

## Import Evidence and Recognize Applications

### IT Asset Management (Cloud)

Because your App-V package naming may be unique to your enterprise, you may need to identify the applications they contain.

If you have many App-V packages, setting up application recognition may be a significant effort on the first import from your App-V server adapter. Once the initial work is done, it is a simpler task to maintain recognition as new App-V packages are put into production.

This procedure continues from the work just completed at the inventory beacon where the adapter runs. Move now to the web interface for IT Asset Management.



#### **To import evidence for application recognition:**

1. Ensure that the upload of data from the App-V server adapter is complete:
  - a. Go to the **Data Imports** page (**Data Collection > IT Assets Inventory Tasks > Data Imports**).
  - b. Select the **Inventory Data** tab, and select the **Show details** check box near to top.
  - c. Find the App-V server adapter listed for the appropriate inventory beacon, and check its **Last import** date, **Duration**, **Validation issues**, and **Status**.

When these are appropriate (in particular, **Status** is **Successful** on the appropriate **Last import** date), continue this procedure. Until then, you need either to remediate any upload problems, or simply wait until the upload is completed.



**Tip:** This page does not dynamically update results, but shows the status when the page was opened. Therefore, if you are waiting for an upload to finish, refresh the page (F5) from time to time to see updated information.

For App-V 5.0 and later, the upload of the .raa file saves the contents into staging tables in the compliance database, awaiting the arrival of usage information. For both App-V 4.6 and 5.0 (and later), the upload of the datafile from the App-V server adapter (on an inventory beacon) queues a job with the batch scheduler. When this job can next be processed, the adapter data (which, for App-V 5.0 and later, is combined in this process with the installer evidence from the .raa file) is imported into the compliance database. Thereafter, either of two results applies:

- Recognized installer evidence is linked to the application, and shows an installation count for each device on which the evidence was found (and can be further examined on the properties for each of those devices).



**Tip:** The next step, showing consumption against an appropriate license, requires both that the application is linked to the license, and that a reconcile has been run, either automatically on schedule or manually.

- The installer evidence that was successfully imported but *not* matched to an application record needs your attention to map it to the correct application. (This is more common with the adapter for release 4.6.) To do that, continue with this process.

2. Identify newly-imported evidence (the App-V packages) that requires a link to application records:

- Go to the **Discovered Evidence** page (**Applications & Evidence > Evidence > Discovered Evidence**).

The **Discovered Evidence** page displays. Ensure that the default **Installer evidence** tab is selected.

- Near the top of the tab, click **Add filter**, and from the drop-down list select **Type**.

A second drop-down list appears, listing the possible values of the evidence type.

- Select App-V.

- Click **Add filter** again, and from the pull-down drop-down list select **Assigned**, then choose the value **No**. With both filters defined, click the blue check mark (tick) to apply this filter.

The list is redrawn to show only evidence of type App-V (the list of App-V packages discovered through the adapter) that have not been matched (manually or automatically) to an application.



**Tip:** You may have no results when these filters are applied. This is a healthy state, meaning that all your App-V evidence is successfully matched to applications. When you have no records here, and want further validation of success, you can drill down into the properties of devices that have a record of installation for the appropriate application.

3. Use your special knowledge of the App-V packages to link each piece of unassigned evidence to an application record. You may do this either by:

- Clicking the **Name** of the evidence, which opens the property sheet for this evidence where you can work on the **Applications** tab

- Following these guidelines to edit locally, still on the **Discovered Evidence** page:

- Click anywhere else in a row (other than on the **Name**) to select that App-V package from the list.

The action buttons above the list become active.

- Click **Assign**.

A blue editor **Assign evidence to an application** appears above the tabs.

- c. Click in the search field, optionally enter a few characters from the application name, and click **Search**.

The editing area expands to include a list of application results matching your search. These applications include any previously created in your enterprise, together with all matching applications from the Application Recognition Library regularly updated by Flexera.

- d. Select your chosen application from the list, and (above the list) click **Add**.

The search field is updated to show the name of the application you selected.



**Tip:** If you cannot find the correct application in the search results, you may create a new application record by clicking **Create an application**, and completing the details in the application properties (the App-V package is automatically listed in the **Evidence** tab of the application properties).

- e. Click **Assign**.

- The blue editing area closes.
- The App-V package is linked to the application you chose (the App-V package now functions like installer evidence to show consumption against any license linked to the application).
- In the list of evidence, the **Assigned** column is updated to Yes, showing that this package has been linked (or assigned) to an application. (If you currently have a filter on the **Assigned** column to show only rows with a No value, the evidence you just assigned must disappear from the list.)



**Tip:** This links the evidence to the application as an exact match across publisher, name, and version records. To protect against future upgrades of the App-V package, you may wish to generalize the version number with a % wild-card. For example, if the original version was 1.0, manually editing the **Version** property of the installer evidence to 1.% means the link to the application remains valid through all the minor upgrades of this package.

4. If the selected application is not yet linked to a license, you can:

- a. Double-click the App-V package in the evidence list (or, while it is selected, click **Open**).

The evidence property sheet opens.

- b. Select the **Applications** tab.

- c. In the list of applications, click the name of the one you have just assigned to the App-V package (or double-click elsewhere in the row; or select the row and click **Open**).

The application property sheet opens.

- d. Select the **Licenses** tab in the application properties.

- e. Optionally enter a few characters of a license name (where you know it); click **Search**.

The search area expands to show the list of available licenses (matching any characters you entered).

- f. Select the appropriate license from those offered, and click **Add license**.

Where a suitable license does not already exist, you may instead create one (for example, go to the **All Licenses** page (**Licenses > License Management > All Licenses**) and click the **Create a license** button).



**Tip:** *Don't forget to link some purchase records to the license to provide your entitlement count.*

## 3

# Issues and Limitations

IT Asset Management (Cloud)

Diagnosis is covered in this chapter, along with limitations and known issues.

## Limitations

IT Asset Management (Cloud)

The following limitations apply to the current releases of the App-V server adapter:

- In the unlikely event that App-V packages have been shared to non-persistent Citrix Virtual Desktops (formerly XenDesktop) VDI devices (instead of users), and IT Asset Management is linked to a Citrix Virtual Desktops broker, no license consumption occurs for those non-persistent devices (because non-persistent VDI devices are not modeled within IT Asset Management when Citrix Virtual Desktops broker information is available).
- License consumption calculations depend on the integration of data both from Active Directory, and from the appropriate App-V database (App-V Management Server database for release 4.6, or reporting database for release 5.0 and later) and the AppVMgmtSvr.ps1 PowerShell script. As linking of this data occurs when the App-V server adapter data is imported, current users, computers, and groups must be imported from Active Directory *first*, before the latest App-V server adapter import occurs. From IT Asset Management 2014 R2, this ordering is automatic, since Active Directory data is imported to the central database as soon as it is uploaded from an inventory beacon.



**Tip:** You may have multiple App-V Management Servers, and multiple streaming servers, that link to a single App-V Management Server database. This requires only one connection from the IT Asset Management App-V server adapter, because this connects only to the database. However, if you have multiple App-V Management Server databases in your estate, configure a separate connection to each of them on appropriate inventory beacons. Where helpful, you may configure multiple such connections (each separately scheduled as you choose) on one inventory beacon.

- The quality of installer evidence recovered from App-V release 4.6 is not high. You should expect to do remedial work both to generalize the evidence found (for example, using the % wild card to generalize version numbering), and to link the evidence to appropriate applications.
- For App-V release 5.0 and later, the system supports installation of the AppVMgmtSvr.ps1 PowerShell script on only one App-V Management Server. This single Management Server may support multiple Publishing Servers (if

necessary spread worldwide for faster distribution of App-V packages to App-V clients); and the App-V clients may report to multiple Reporting Servers (independent of the source from which the App-V packages were downloaded). Different App-V Management Servers do not self-identify in the .raa inventory file, and the App-V reporting database does not identify which application usage information is associated with which App-V Management Server. For these reasons, only a single App-V Management Server (for release 5.0 and later) is supported.

## Investigating Issues

### IT Asset Management (Cloud)

Proof that the App-V server adapter is operational, and the data upload is also successful, can be seen in either or both of two ways:

- Installation records for the appropriate applications against expected inventory devices (possibly including Remote devices for which no hardware inventory is available)
- The presence of any newly-discovered inventory of type App-V in the **Discovered Inventory** list, typically with an **Assigned** value of No.

If neither of these is the case, the issues could be with:

- Imports from the App-V server adapter on an inventory beacon (for both App-V release 4.6, and App-V release 5.0 or later)
- Imports of the .raa file from your App-V Management Server (only for App-V release 5.0 or later)
- Missing links between the installer evidence (representing App-V packages) and the applications in the packages
- Missing consumption on an appropriate license.

Each of these is covered in turn below.

### No data imports from adapter on inventory beacon

Check the following to identify the problem with imports from the App-V server adapter:

1. Are you sure that there should be new records? Have new packages been brought into production since the last time inventory was collected and fully processed?
2. Check the **Status** of the latest upload (Go to the **Data Imports** page (**Data Collection > IT Assets Inventory Tasks > Data Imports**), choose the **Inventory Data** tab and select **Show details**). Also validate that the **Last import** date is as expected, so that the upload occurred *after* new App-V packages were brought into production.
3. If uploads are not happening, move to the inventory beacon where the adapter runs, and check the status of the App-V connection. Use the **Test connection** button to ensure it can connect to the appropriate database (App-V Management Server database for App-V release 4.6, and the App-V reporting database for release 5.0 and later). Details about setting the connection are in [Configuring the Adapter](#).
4. On the same inventory beacon, test its connection to the central application server for IT Asset Management, using the **Parent connection** page and the **Test connection** button there. (If this inventory beacon is part of a hierarchy, check the connections all the way up the hierarchy to prove that uploads can reach the central server.)
5. Check for stalled uploads by looking for an App-V inventory file in the %CommonAppData%\Flexera Software\Beacon\IntermediateData directory on the inventory beacon (or, in a hierarchy, in the chain of inventory



beacons). (Notice that the folder for files from the App-V server adapter is separate from the folder for .raa files from the PowerShell scripts used with release 5.0 and later.) App-V data files are named in part for the connection you established (see [Configuring the Adapter](#)). Once an inventory file of this type is successfully uploaded, it is removed from this intermediate data location on the inventory beacon; so any file in this folder on any server in the hierarchy has not yet been uploaded to the parent server. Upload failures may occur for temporary reasons, such as a network timeout; but there is a catch-up task run overnight to re-attempt uploads of any stalled files.

6. Has a reconciliation calculation occurred since the upload? Until this occurs (normally overnight), new App-V inventory cannot be displayed in the web interface for IT Asset Management. Check the date and time on the **Reconcile** page (**Data Collection > Process Data > Reconcile**). The last import and reconciliation must be after the latest upload from the inventory beacon.
7. If you are using App-V release 5.0 or later, a failure to upload and save .raa files may also prevent presentation of results, even when the application usage information is successfully imported from the App-V server adapter. Issues with .raa files are covered in the next section.

## No data imports from the PowerShell script for App-V release 5.0 and later

If uploads of .raa files do not appear to be working:

1. Ensure that a new upload is required: that is, that the AppVMgmtSvr.ps1 script has executed successfully since the last inventory import and license consumption calculation (the most recent file is always saved on the App-V Management Server for checking, and is only replaced the next time that the script executes):
  - Check your scheduling for the script's execution, in particular that the command line options are correct (see [Obtaining \(and Deploying\) the Adapter Components](#)).
  - Check the log file for the last run (you may have renamed or relocated the log file, as described in [Command Line for PowerShell Script](#)). In particular, ensure that there were no problems with the file upload to the inventory beacon.
2. Move to that inventory beacon, and check for stalled uploads by looking for an .raa file in `%CommonAppData%\Flexera Software\Incoming\RemoteApplications` (this file path is different from the one used for files uploaded by the App-V server adapter). If you have a hierarchy of inventory beacons, check each in turn.
3. If file uploads are happening successfully, has there been a reconcile since the last .raa file was uploaded? Check the date and time on the **Reconcile** page (**Data Collection > Process Data > Reconcile**). The last import and reconciliation must be after the latest upload from the inventory beacon. If not, you may (as an administrator) manually trigger a reconcile (navigate to **License Compliance > Reconcile**).

## Missing application recognition

For App-V release 4.6, data imported from the App-V server adapter includes only the App-V package name and the Active Directory groups (or individuals) that may access the package, according to the Access Control Lists (ACL). Specifically, this imported information cannot recognize what application is hidden within the App-V package. Application recognition requires a separate step. Even for App-V release 5.0 and later, where much better installer evidence allows automatic matching of the evidence rules for the appropriate application, there may be cases where the installer evidence needs manual attention. This will be the case when:

- There is no matching application available within IT Asset Management, either in application records that you have created locally, or in the Application Recognition Library

- There is an appropriate application, but the values returned from App-V do not match with the existing inventory rules for the application record.

In cases where installer evidence from App-V is unmatched, you can check as follows:

1. First be sure that data uploads and imports are happening, as validated in the previous sections.
2. Go to the **Discovered Evidence** page (**Applications & Evidence > Evidence > Discovered Evidence**) and select the **Installer evidence** tab. Filter for **Type=App-V**, and check the **Assigned** column. If it displays No, you need to link this package to an application, as described in [Import Evidence and Recognize Applications](#).) For App-V release 4.6, newly imported evidence is always unassigned, and requires you to manually associate the evidence (or App-V package) with an application.
3. For App-V release 5.0 (and later), when installer evidence appears in the above listing, it may be worth checking why the data from the App-V installer evidence did not match existing evidence rules for the appropriate application (assuming the application is already present locally or in the Application Recognition Library). To do this, navigate to the **Evidence** tab of the application's properties, where all existing evidence "rules" are listed (making sure that **Installer** is the selected subtab). Compare the data displayed there with content in your .raa file (see sample .raa file in [File Format for .raa](#)). Here is a worked example of successful matching using the FileZilla 3 application:

App-V element's attribute	Application's installer evidence property
msiDisplayName="FileZilla Client 3.2.4.1"	<b>Name</b> FileZilla Client 3.2.%
msiPublisher=""	<b>Publisher</b> (blank)
msiVersion="3.2.4.1"	<b>Version</b> 3.2.%
accessModeID="2" (produces the evidence type App-V)	<b>Type</b> Any

Thus the .raa entry produces installer evidence that is immediately matched by the existing installer evidence rule for the application, and produces an installation count against that application. But looking ahead (in imagination) to the day when the .raa entry covers a version of 4.2.3.1, the App-V package data in the .raa file would no longer match this evidence rule. At that time, if a new rule was yet to be published in the Application Recognition Library, the installer evidence created from the .raa file would appear in the **Discovered Evidence** listing, and you could link it to the application, preferably generalizing it (similarly to the example above) to create a rule that would match several minor releases.

## Missing consumption

Showing consumption of license entitlements for App-V packages (and the applications they contain) requires:

- The adapter gathers data from the App-V server and uploads and imports it into IT Asset Management (see first section above for more details).
- For App-V release 5.0 or later, the .raa file is uploaded from your App-V Management Server
- The application is recognized, either automatically, or because you have linked the App-V package to an application record (see previous section)
- The application is linked to a license (and in turn the license should be linked to purchase records to show your legal

entitlements) — here, this is left as an exercise for the reader.

- Active Directory imports are current, allowing mapping of the groups and users from the ACLs in the App-V Management Server database to user records in IT Asset Management.

These notes address the last stage, enabling Active Directory to map from the ACL lists to user records. This process is automatic, provided that all the necessary data is available.

1. On the appropriate inventory beacon, open the **Active Directory** page (from the **Connections** group), and validate that a connection is both established and scheduled. (For details, see *Importing from Active Directory* in the online help.) Review the **Last run** time to see when data was last collected, uploaded, and imported. You may also choose **Execute Now** if imports have been disrupted.
2. Once sufficient time has passed for Active Directory data collection, upload, and import (normally, 30 minutes should be more than adequate), go to the **All IT Asset Users** page (**Organization > All IT Asset Users**) to review the user list to establish that the expected user names are all available.



**Tip:** If you have App-V applications secured by security groups from multiple Active Directory domains, ensure that the Active Directory import runs against all applicable domains in your environment. The simplest approach may well be to ensure that you import from all your Active Directory domains, since if you use foreign security principals from multiple trusted domains, it can be difficult to keep track of access to App-V packages. FlexNet Manager Suite imports only from each individually specified Active Directory domain; so you need to ensure that all applicable domains are specified. As an example of multiple domains being affected:

- Suppose you have Group-A in Domain-A that contains a child Group-B, where Group-B actually comes from Domain-B.
- In this case, granting access to an App-V package to Group-A also grants access to Group-B (because of the parent-child relationship between the groups).
- This inheritance continues to work even when there is only one-way trust from Domain-B to Domain-A.
- In such a case, it is imperative that you run an Active Directory import against both Domain-A and Domain-B. When you have many domains, the simplest path is just to run an Active Directory import from every domain.



**Note:** You cannot review Active Directory group memberships within IT Asset Management. Only the resulting list of users is available (along with computers, sites, and subnets).

## Known Issues

### IT Asset Management (Cloud)

The following issues relate primarily to the stability of data available for collection from the App-V server.

#### Renaming packages

App-V allows you to rename packages (without necessarily changing their contents). When a package is renamed, the App-V application record no longer links to usage records within the appropriate App-V database. This means that the App-V server adapter (and, for App-V release 5.0 and later, the AppVMgmtSvr.ps1 PowerShell script) cannot import meaningful data for license consumption calculations. As well, the installer evidence generated from the App-V import now has a different name, which may not match evidence rules for the appropriate application. If that is the case, the

consumption calculations for any license linked to an application that was linked to the previous installer evidence from App-V drops to zero (from this import connection).

For this reason, it is *strongly recommended* that you do not change the name of existing App-V packages in the App-V Management Server interface. Where renaming is unavoidable, remember that you may need to link the new package (represented as installer evidence in IT Asset Management) to the application to restore consumption calculations (especially for App-V release 4.6).

## Updating package versions

Typical "rules" for applying installer evidence to identify applications use the publisher, the application, and the version recorded in the installer evidence. If you use a version number for your App-V packages (such as 1.0), and link exactly this App-V 'installer' evidence to the application record in IT Asset Management, then updating the version number of the package on your App-V Server (say, to 1.1) may break the link to the application, and the linked license loses its consumption as a result.

You can avoid this problem (at least for minor version changes) by using wild-cards in the linking of evidence to applications. To do this, after you have linked the App-V evidence to the application:

1. Navigate to the **Installer Evidence Properties** page.
2. Select the **General** tab.
3. Edit the **Version** property, using a wild-card percentage sign (%) to generalize the match across multiple versions.  
For example:
  - The original (say 9 . 2) is an exact match for only one version number.
  - A value of 9 . % matches all the minor releases of version 9.
  - A value of % matches all versions of the App-V package with the same name and publisher.

## 4

## Data mapping

IT Asset Management (Cloud)


This chapter covers the relationships between the data fields in the source App-V data and the final locations of the data inside the IT Asset Management database.

### App-V Release 4.6 Data Transfers

IT Asset Management (Cloud)

This table lists the data extracted from App-V release 4.6, and where it is stored the compliance database in IT Asset Management. Imported data is stored in temporary staging tables for data manipulation, and then moved into their final destination tables as noted below. These columns in the compliance database are available for use in custom reports, and the like.

App-V (Table)/Column	FNMS (Table)/Column	Notes
(REPORTING_CLIENT_INFORMATION) host_name	(ComplianceComputer) ComputerName	If the computer name already exists in the ComplianceComputer table, this device is identified as the consuming device. If it does not already exist, the device is added to the ComplianceComputer table with a ComplianceComputerTypeID of 4, which (through the ComplianceComputerType table) indicates a remote device from which inventory cannot be collected. All such created records display their connection name (as the Inventory Agent), are marked as incomplete records, and are given a fictitious serial number of 1.

App-V (Table)/Column	FNMS (Table)/Column	Notes
(REPORTING_CLIENT_INFORMATION) host_name	(ComplianceComputer) ComplianceDomainID	The domain name is extracted from the host record on the App-V server. If the domain does not already exist in the compliance database, it is added to the ComplianceDomain table, and as required the ComplianceDomainID may be updated in the ComplianceComputer table.
(APPLICATION_USAGE) username	(ComplianceComputer) ComplianceUserID	The last logged on user for this computer.
(VW_APPLICATIONS) app_name	(InstallerEvidence) DisplayName	The name of the App-V package (which may bear no resemblance to the application inside) is used to identify the evidence record.
 <b>Note:</b> It is possible for an App-V package to contain more than one application. This makes it impossible to link the App-V evidence to a single application record. Best practice is to make each package contain exactly one licensable application.		
(VW_APPLICATIONS) version	(InstallerEvidence) Version	The version of the App-V package. Again, this bears no necessary relationship to the version of the application inside the package, and is at the whim of the person preparing the package (for example, it may represent the number of times the application package was modified).
String literal App-V	(InstallerEvidence) Publisher	All App-V evidence is given the publisher name of App-V.

## App-V Release 5.0 (and Later) Data Transfers

### IT Asset Management (Cloud)

This data is imported from Microsoft App-V release 5.0 and later. It is imported by the combination of the PowerShell script installed on your App-v Management Server, and the App-V server adapter on an inventory beacon that can access the App-V reporting database.

Several staging tables in the compliance database store information uploaded from the App-V reporting database and the .raa files. These include:

- ImportedComputer
- ImportedInstalledInstallerEvidenceUsage

- ImportedInstallerEvidence
- ImportedRemoteApplication
- ImportedRemoteApplicationAccess
- ImportedRemoteApplicationServer
- ImportedRemoteApplicationInstallerData.

Data is held in these tables until the import from the App-V server adapter is complete, and then the resolvers are triggered to combine the data from:

- The most recent Active Directory import(s) across all relevant domains
- The most recent imports of the .raa file from all App-V Management Server
- The App-V reporting database.

For this reason, it is important that the import from the App-V reporting database happens last.

The following table lists the elements and their attributes from the .raa file, and their final table and column in the compliance database within IT Asset Management.

XML (Element) / Attribute	FNMS (Table) / Column	Notes
(app) appID	n.a.	Used as a key as required across the temporary tables listed above.
(app) userSid	n.a.	The group SID (from Active Directory) identifying the users and computers with access to the package. This drives the linking of inventory device and user records to the application.
(msiData) msiDisplayName	(InstallerEvidence) DisplayName	Visible as the <b>Name</b> in Installer Evidence properties.
(msiData) msiPublisher	(InstallerEvidence) Publisher	Visible as the <b>Publisher</b> in Installer Evidence properties.
(msiData) msiVersion	(InstallerEvidence) Version	Visible as the <b>Version</b> in Installer Evidence properties.
(msiData) msiProductCode	(InstallerEvidence) ProductCode	Not visible in the web interface.

The following are the views used for source data from the App-V reporting database. It is not possible to give a simple mapping of this source data to columns in particular database tables within IT Asset Management, because the data is normalized and otherwise processed at each stage. For example, the username column collected from view\_ApplicationUsage in the App-V reporting database is already subject to considerable validation and processing before it is stored in the lastloggedonuser column in the staging tables (staging tables are listed above). Thereafter, the user name is correlated with other inventory sources, resulting in further processing before it is used to determine a link between the application and the user. For that reason, this table shows only the source content in the App-V reporting database.

App-V View	Columns
dbo.view_ApplicationUsage	<ul style="list-style-type: none"><li>• app_name</li><li>• app_version</li><li>• end_time</li><li>• host_id</li><li>• start_time</li><li>• username</li><li>• version_guid</li></ul>
dbo.view_ClientInformation	<ul style="list-style-type: none"><li>• host_id</li><li>• host_name</li></ul>
dbo.view_PackageInformation	<ul style="list-style-type: none"><li>• host_id</li><li>• package_id</li><li>• version_guid</li></ul>



# XII

## Microsoft Azure Adapter

### IT Asset Management (Cloud)

This part of the reference introduces the FlexNet connector to Microsoft Azure, enabling you to

- Collect Azure virtual machine data
- Track the license model for Windows Server and SQL Server in the cloud, particularly for installations that are taking advantage of the Microsoft Hybrid Benefit (AHB) that allows you to "bring your own license" (BYOL), re-purposing licenses originally purchased to authorize product installations in your on-premises environment in one of two ways, either:
  - Transferring the license entitlements to cover installations running on instances in Azure instead; or
  - In strictly limited cases, using the same license entitlements to simultaneously authorize installations on-premises *and* in Azure.

Tracking cloud license models for each instance using AHB requires that the connector uses the Microsoft Az module of PowerShell cmdlets for managing resources in Azure.



**Tip:** If you have previously implemented the Azure connector (prior to version 2020 R2.3, internally identified as 16.3.0), the earlier connector used an earlier and less capable module called AzureRM, which is now outdated, no longer receiving bug fixes, and not capable of supporting future functionality. For this reason, best practice is to uninstall the old AzureRM module, and replace it with the current Az module. While strongly recommended, this is not mandatory; but if you persist in using the old connector with the AzureRM module, you cannot track any cloud license model utilizing the Azure Hybrid Benefit.

Because the Azure connector gathers information that is separate from, and does not include, full hardware and software inventory, this reference covers more than the connector itself. As well as details of prerequisites, set up, and data gathering with the connector, the part includes some insights into:

- Processes for gathering *complete* hardware and software inventory for your cloud-based instances
- Considerations for license management.

In fact, using those guidelines, you *could* configure inventory gathering in Azure without using the connector at all. Provided that you are deploying the latest FlexNet Inventory Agent (13.2.x or later), inventory alone even returns the cloud service provider. What the connector adds is:

- Automatically setting the **Hosted in** property in the inventory device records created when inventory is returned

- Adding the cloud service provider and instance details when these are missing from inventory collected by legacy versions of FlexNet Inventory Agent, or from third-party inventory sources
- Populating the **Cloud Service Provider Inventory** page with records of instances currently running in your Azure environment, including a few properties that are additional to the normal hardware and software inventory
- Importing permanent records of instances that were previously running but that have now been terminated (and for which, as a result, any prior inventory devices records have now been deleted)
- Triggering automatic deletion (completed at the next full import and compliance calculation) of any inventory device record linked to an instance that is now terminated
- Importantly (when the Microsoft Az module is in use), tracking use of the Azure Hybrid Benefit to license installations of Microsoft Windows Server and Microsoft SQL Server in the cloud.

This reference focuses on background about the connector, and additional material you need in order to build license management around the connector. You will not find details about making the actual connection here.



**Note:** *The Microsoft Azure inventory connector does not collect virtual machine information for virtual machines managed by Microsoft's classic deployment model. For these devices we recommend you use an alternative such as the inventory spreadsheet upload, the Business Importer or simply through editing manually.*



**Note:** *Containers are not currently supported.*

Step-by-step instructions for configuring the connection to Azure are available in the online help at ***FlexNet Manager Suite Online Help > Inventory Beacons > Inventory Systems Page > Connecting to External Inventory Systems > Managing PowerShell Connections > Managing Azure Connections.***

# 1

## Prerequisites and Setting Up

### IT Asset Management (Cloud)

The connector to Microsoft Azure is configured on a convenient inventory beacon, for which there are the following prerequisites:

- The operating system is either:
  - Windows Server 2008 R2 SP1 or later
  - Windows 7 SP1 or later
- FlexNet Beacon software has been updated to release 16.3.0 (available with IT Asset Management 2020 R2.3) or later
- You must log onto the inventory beacon, and run FlexNet Beacon, using an account with administrator privileges
- .NET Framework 4.7.2 or later is installed (installation instructions and downloads start from <https://docs.microsoft.com/en-us/dotnet/framework/install/>)
- PowerShell 5.1 or later is running (PowerShell 7.x and later is recommended)



**Tip:** To check the version of PowerShell installed on your inventory beacon:

1. As administrator, run Windows PowerShell.
2. Execute the cmdlet:

```
$PSVersionTable.PSVersion
```

- The PowerShell execution policy is set to RemoteSigned
- FlexNet Inventory Agent version 15.0.0 (released with IT Asset Management 2020 R1) or later is available on the inventory beacon
- A web browser is installed and enabled on the inventory beacon



**Important:** If you have configured your existing connection to Microsoft Azure using the original AzureRM module (downloaded from Microsoft), this must be uninstalled before you install the newer Az module. To uninstall the original module:

1. On your inventory beacon, logged in as Administrator, run PowerShell.

2. Uninstall the AzureRM module with the following command:

```
uninstall-module AzureRM
```

- Microsoft Az for Windows PowerShell version 6.1.0 or later is required by the Azure Resource Graph module.



**Tip:** Although the connector is compatible with the Microsoft Az module version 5.2.0 or later, version 6.1.0 or later is required for compatibility with the Azure Resource Graph module which provides the latest performance improvements.

If you have Az modules already installed in your environment, update to the latest Az modules:

1. Ensure you are in a Windows PowerShell session run as administrator.
2. Execute the command:

```
Update-Module -Name Az -Force
```



**Note:** If you run into any issues while updating Az modules, please try uninstalling and reinstalling the Az modules, always while in a Windows PowerShell session run as administrator:

- To uninstall:

```
Get-InstalledModule -Name Az* | Uninstall-module
```

- To install:

```
Install-Module -Name Az -AllowClobber
```



**Tip:** To check the version of the Az module installed on your inventory beacon:

1. As administrator, run Windows PowerShell.
2. Execute the cmdlet:

```
Get-Installedmodule az
```

Additional versions are available for download from <https://www.powershellgallery.com/packages/Az/>.

- Microsoft Azure Resource Graph module version 0.11 or later is installed (remembering that this requires Microsoft Az for Windows PowerShell version 6.1.0 or later, as mentioned above). Flexera recommends that you install this as it optimizes performance, especially for large-scale implementations in Azure.



**Tip:** To install this module on your inventory beacon:

1. Ensure you are in a Windows PowerShell session run as administrator.
2. Execute the command:

```
Install-Module -Name Az.ResourceGraph
```

3. Validate your installation with the cmdlet:

```
Get-InstalledModule -Name 'Az.ResourceGraph'
```

On the Azure side, you must:

- First decide in which Microsoft Azure "environment" your connector will operate. All the connection details and credentials are specific to the environment, and cannot be interchanged. The default environment is AzureCloud (the value used if no other environment is specified for the connection). Other currently-available environments can be determined using the Azure cmdlet:

```
(Get-AzureRmEnvironment).Name
```

Examples include:

- AzureChinaCloud
- AzureGermanCloud
- AzureUSGovernment.

Your choice of environment is specified when configuring the connection.

- Choose or create a service principal identity. Choose the built-in role `Virtual machine contributor`, or create a custom object with **read** access to **Microsoft.Compute/virtualMachines** and **Microsoft.SqlVirtualMachine/sqlVirtualMachines**.

Step-by-step instructions for configuring the connection to your chosen Azure environment are available in the online help at *FlexNet Manager Suite Online Help > Inventory Beacons > Inventory Systems Page > Connecting to External Inventory Systems > Managing PowerShell Connections > Managing Azure Connections*.

## 2

# Certificate-based Authentication Prerequisites and Setting Up

## IT Asset Management (Cloud)

Certificate-based authentication enables you to use managed identities for Microsoft Azure resources as a secure way for authentication. Managed identities for Azure resources provide Microsoft Azure services with an automatically managed identity in Azure Active Directory. This identity can be used to authenticate to any service that supports Microsoft Azure Active Directory authentication, without having credentials in the code.

For more information on certificate-based authentication, see [Overview of Microsoft Entra certificate-based authentication](#) in the *Microsoft Entra Online Help*.

Prior to configuring the Microsoft Azure adapter to use certificate-based authentication, you first need to create a self-signed certificate, import the certificate, and then upload the certificate to the Microsoft Azure Portal. Follow the below steps to complete this process.

For steps on how to configure the Azure adapter to use certificate-based authentication, see [Managing Azure Connections](#) in the Online Help.

### Creating a self-signed certificate

To create a new certificate, run the below script. Each time the script is executed, a new certificate will be created in your local computer certificate store. If you need to rerun the script, it is recommended to delete the previous certificates from the local computer certificate store (see [Deleting the certificate from the certificate store](#)).

The below script creates a self-signed certificate which is stored under the path `cert:\LocalMachine\My`. It also exports `.cer` and `.pfx` files under a given file path.

- The `.pfx` file is used to install the certificate on other machines, this is a password-protected file and the password is **clientId** which is used while creating the certificate.
- The `.cer` file is used to upload the certificate to the Azure portal for your application.



**Note:** The below script creates a self-signed certificate and applies it to the current machine from where the script is being executed. There is no requirement to import the certificate using a `.pfx` file on the machine where the certificate was generated.

**Script to create a self-signed certificate:**

```
function Create-AzureSelfSignedCertificate
{
    param (
        [Parameter(Mandatory = $true)]
        [string]$tenantId,
        [Parameter(Mandatory = $true)]
        [string]$clientId,
        [Parameter(Mandatory = $true)]
        [string]$filePath
    )
    $StoreLocation = 'LocalMachine'
    $expirationYears = 1
    $SubjectName = $tenantId + '.' + $clientId
    $cert_password = $clientId
    $pfxFileName = $SubjectName + '.pfx'
    $cerFileName = $SubjectName + '.cer'
    $PfxFilePath = $filePath + $pfxFileName
    $CerFilePath = $filePath + $cerFileName
    $CertBeginDate = Get-Date
    $CertExpiryDate = $CertBeginDate.AddYears($expirationYears)
    $SecStringPw = ConvertTo-SecureString -String $cert_password -Force -AsPlainText
    $Cert = New-SelfSignedCertificate -DnsName $SubjectName -CertStoreLocation
"cert:\$StoreLocation\My" -NotBefore $CertBeginDate -NotAfter $CertExpiryDate -KeySpec
Signature
    Export-PfxCertificate -cert $Cert -FilePath $PFXFilePath -Password $SecStringPw
    Export-Certificate -cert $Cert -FilePath $CerFilePath
}

#Example
Create-AzureSelfSignedCertificate -tenantId "91034d23-0b63-4943-b138-367d4dfac252"
-clientId "a49b1e97-a1ad-4d8c-896a-dadd4ac8f1da" -filePath "c:\temp\cert\"
```

**Importing the certificate using .pfx file**

1. Open the local computer certificate store using the **certlm.msc** command (run as administrator).
2. In the left pane, click **Personal** and under the **Action** menu click **All Tasks** followed by **Import**.
3. On the Welcome page of the Certificate Import Wizard, click **Next**. On the next page, browse for and select the relevant .pfx file and click **Next**.
4. On the Private key protection page, enter the password for the private key. Select the **Include all extended properties** checkbox and click **Next**.
5. On the Certificate Store page, select the **Place all certificates in the following store** radio button, then browse for and select **Personal** for the certificate store. Click **Next**.
6. On completion of the Certificate Import Wizard, review the following specified settings and click **Finish**.

## Uploading a certificate to the Azure portal

1. Log into your Azure Directory portal ([portal.azure.com](https://portal.azure.com)). From the left menu, click **App Services** and then click the name of your application.
2. From your application's navigation menu, click **Certificates > Public key certificates (.cer) > Add certificate**.
3. In the Add public key certificate dialog, select your .cer file and then enter a certificate name for your application. Once the certificate is uploaded, you can view the certificate thumbprint from the **Certificates** tab.

## Deleting the certificate from the certificate store

Run the following commands to delete a certificate from the cert store:

- `Get-ChildItem cmdlet` - lists all certificates in the certificate store.

Example:

```
Get-ChildItem -Path Cert:\LocalMachine\My
```

- `Remove-Item cmdlet` - removes a certificate when specified with the thumbprint and certificate store path.

Example:

```
$thumbprint = "b6b7dd0c0dd60505c70e95627b68775fa98005fc"  
Remove-Item -Path "Cert:\LocalMachine\My\$thumbprint"
```



## 3

# Thinking about Inventory and Licensing

## IT Asset Management (Cloud)

In many ways, collecting inventory and working out license consumption for instances in Microsoft Azure is much the same as licensing virtual machines on host servers sitting within the datacenter in your own offices. However, there are some differences that are worth thinking about. For example, within your own datacenter, you have full control of, and presumably full inventory for, the host server as well as the VMs running on it; and for some license types, host details are relevant even when licensing software on a virtual machine. Depending on what you purchase, this may not be the case in Microsoft Azure.

The following topics bring together some points to consider, first about inventory collection, and then about licensing.

## Collecting Inventory from Instances

### IT Asset Management (Cloud)

The connector to Microsoft Azure gathers useful information, but it cannot gather information valuable for license consumption calculations from each of the instances running in your Azure cloud. To gather useful inventory from all instances, best practice is to include the FlexNet Inventory Agent (release 13.2.*buildNumber* or later) in the Microsoft Azure managed VM image from which your devices are instantiated. For details, see the Azure documentation [Create a managed image of a generalized VM in Azure](#) for Windows, or for Linux see [How to create an image of a virtual machine or VHD](#).

### Unique device naming is critical

It is critical to your license management in Microsoft Azure to ensure that each instance is *uniquely* named on instantiation. If you do not do this, incoming inventory reports are not differentiated, and are interpreted as updated inventory from a single device, named with the default device name provided by the base image.

By default, the internal name of a virtual machine is the same as the external name given to it when it is created. While you can change the internal name of a virtual machine, you cannot change the external name of the virtual machine as displayed in the Azure portal.

Resource names for virtual machines are immutable. So, if you need to change names, you need to redeploy your

virtual machine. You can do this by deleting the current virtual machine, while maintaining the current virtual machine disks, and then creating a new virtual machine using the correct name and point it to the previously maintained disks.

To ensure a device name for each instance that is unique over time, methods vary across platforms. In summary:

- The easiest method on Windows is to use Sysprep when creating your managed VM image, leaving the name unset so that a new name is provided on instantiation.
- For Linux, if you already provide for unique `Hostname` values on all your instances, no further action is required; and otherwise, a little extra preparation can ensure that the unique `InstanceID` for each instance is also returned in inventory as the `MachineID`, where it differentiates all returned inventory as coming from devices with distinct names.

## Schedule specialization

Scheduling inventory collection by the locally-installed FlexNet Inventory Agent requires:

- Provide an upload location (`ManageSoftRL`) for inventory collection
- Omit any download location, since the instance life-cycle is projected to be too short to require any policy update or system-wide schedule update
- Trigger inventory collection on start-up, followed immediately by inventory upload to the defined `ManageSoftRL` (this cycle might typically require 3-5 minutes all up, a figure which may change based on other customizations you may want to include in your managed VM image)
- Provide a back-up schedule of daily inventory collection.
- Keep in mind, as well, that removing the normal paths for policy update also removes the normal updates to `InventorySettings.xml`, which includes a range of advanced capabilities for FlexNet Inventory Agent, including Oracle inventory collection and advanced inventory techniques for Microsoft and other vendors. Your strategy therefore includes embedding your most recently downloaded `InventorySettings.xml` in your managed VM image (and later, updating it as required). For more guidance on customizing instances, see [Configuring an Azure VM Image for Instances](#).

## Planning your inventory beacons

The final major question is the location of your inventory beacon(s) for uploading inventory data from these instances. Depending on the reliability of network communications between your cloud-based instances and your own enterprise network, you may choose either:

- To run (at least) one instance within your Azure cloud as an inventory beacon, ready 24/7 for any of your instances in the cloud to upload their inventory. This is especially valuable for instances:
  - Where the locally-installed FlexNet Inventory Agent has no opportunity to retry uploads around any network failures; and
  - where fast network speeds may best support your planned life-cycle for these instances.

The cloud-based inventory beacon can then provide a more rugged upload to your enterprise network, since it has longer up-time and built-in mechanisms for retrying any uploads that are interrupted by transient networking issues.

- Run an inventory beacon within your enterprise network (or even, potentially, in a demilitarized zone outside your firewall), to which all the cloud-based instances have network access, allowing them to upload their inventory files as and when needed.

Choosing between these alternative places for your inventory beacon may require balancing questions of security and performance against cost. Whichever choice you make, a good practice is to use a DNS alias to identify the inventory beacon, so that you can quickly and easily make changes without disrupting the rest of your infrastructure.



**Tip:** As for all communication between your enterprise network and the Microsoft Azure network, an Amazon Virtual Private Cloud (VPC) is required for either of the above architectures; and if you require a link from your inventory beacon in the Azure cloud to your central application server in your datacenter, consider a hardware VPN connection as well (see the Microsoft Azure documentation for details).

Keep in mind that an inventory beacon does not normally initiate ('push') communication to FlexNet Inventory Agents installed on your Microsoft Azure instances. (The only exceptions are for the 'remote execution' tasks known as adoption and zero footprint inventory collection, neither of which you expect to use in a cloud environment.) Since all other communications are initiated by the FlexNet Inventory Agent ('pull' communications, such as policy updates, collection of self-update packages, and the upload of inventory), this means that your implementation only needs to point the FlexNet Inventory Agent on each instance to its bootstrap inventory beacon. You do not need to know details like the IP addresses (public or private) for the instances reporting inventory through the FlexNet Inventory Agent.

## Inventory records and exceptions

After a full inventory import and license reconciliation, you can expect to see records created/updated in IT Asset Management as follows:

- For every instance where the locally-installed FlexNet Inventory Agent has uploaded inventory, an inventory device record
- Similarly, for every instance reporting inventory through a third-party tool, such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), an inventory device record
- For every cloud instance appearing in data from the Azure PowerShell connector, regardless of whether it appears in uploaded inventory or not, a cloud instance record in the `CloudServiceInstance` table (and of those, the ones that *also* have an inventory device record show a link to it in the **Cloud Service Provider Inventory** page)
- For dedicated hosts identified through the Azure PowerShell connector (and *only* dedicated hosts, not any other kind of host), an inventory device record for the host.

You will *not* find inventory device records for any of the following:

- A cloud instance that does not report inventory
- A host device that is anything other than a dedicated host (such as a default host, or a host for dedicated instances)
- Any instance that you have terminated.

Terminated instances are a special case.

1. Before termination, the running instance may have had an inventory device record (provided that it was reporting inventory).
2. When you terminate the instance, the `CloudServiceInstance` table is updated with the terminated status. This record, with its terminated status, is visible in the **Cloud Service Provider Inventory** page, if you set the filters on

that page appropriately.

3. At the next full inventory import (by default, overnight, immediately before the license compliance calculations), *any inventory device record linked to a terminated instance is deleted*, because you do not want terminated instances consuming from your licenses.
4. Of course, the terminated instance no longer reports inventory; but its previously-recorded inventory is still in the inventory database (and in normal processes is not cleaned up for a while). Although the next full import (typically the next night) normally imports records from the inventory database, for terminated instances this is prevented by the terminated flag in the `CloudServiceInstance` table. This prevents the deleted inventory device record from reappearing.

Should you ever need to review the software previously installed on a terminated instance, you can:

1. Go to the **Cloud Service Provider Inventory** page (**Inventory > Virtual Devices > Cloud Service Provider Inventory**).
2. Change the default filter (top left) for **Include terminated instances** to Yes.
3. Optionally, filter the **Last known state** column to show only **Terminated**.
4. Optionally, use other filters, such as **Instances reported** in an appropriate period, or the **Instance type** and **Instance region** columns, to narrow your search for this instance.
5. Check the **Image ID** column to identify the original image from which this instance was launched (you may need to drag this column out of the column chooser).
6. Through your Azure console, identify this managed VM image, and inspect its software load and resulting license impacts.

## Data integration

Inventory from a hosted instance may come from multiple sources, and must be integrated correctly. Various rules are used depending on the sources being combined:

- PowerShell connector + current FlexNet Inventory Agent are linked by:
  - `CloudServiceProviderID` (displayed as the **Cloud service provider** name in the web interface)

Matching of the incoming data with existing inventory device records (that is, determining whether this is an update to an existing record, or a new record) uses the standard device matching rules, with the **Cloud service provider** and **Instance ID** checked as last priority.

Finally, the merging of data between the PowerShell connector and the current FlexNet Inventory Agent uses these priorities:

1. If one source contains a data point that the other does not, the data is used.
2. If both sources contain the same data point but with different values, the data with the most recent inventory date is used.
3. If different values were imported on the same day, the FlexNet Inventory Agent is given priority if it has been set as the primary inventory source.
4. Otherwise, the data is used from the most recently created inventory source, with the winning source visible in the inventory device properties as **Last inventory source**.

# BYOL Licensing Considerations

## IT Asset Management (Cloud)

The Azure connector does not return full inventory from instances running in Azure, and so does not create inventory device records. There are two special cases to distinguish:

1. If the import from the Azure connector is matched with *another* record of full software and hardware inventory, the matching process creates an inventory device record. Full software and hardware inventory may be returned from an Microsoft Azure instance when either:
  - FlexNet Inventory Agent is locally installed on the instance, usually because it is part of the Azure VM Image from which the instance was launched
  - You configure zero footprint inventory collection from an inventory beacon that has network access to your target instances, and credentials saved in its Password Manager store
  - You have some third-party tool, such as Microsoft Endpoint Configuration Manager (previously Microsoft SCCM), collecting inventory from your Azure instances, and these results are then being imported into IT Asset Management.

As always for all inventory devices, the installations found in software inventory for that device consume entitlements from the licenses to which each software record is linked.

2. A record may be created and displayed in the **Cloud Service Provider Inventory** page to show the cloud license model (BYOL or PAYG) discovered for instances that may benefit from the Azure Hybrid Benefit (AHB). AHB allows you to benefit from licenses originally purchased to cover on-premises installations of either Microsoft Windows Server, or Microsoft SQL Server, in either of two ways (subject to active Software Assurance): re-purposing them to cover cloud instances instead; or in the special case of Windows Server Datacenter edition, using them to simultaneously authorize both on-premises and cloud instances in Azure. However, in order to determine what software should be licensed (with or without AHB), and measure license consumption, you still need to arrange for inventory collection covering these instances, for example through:
  - The FlexNet Inventory Agent locally installed on the instance
  - Remote inventory collection by an inventory beacon with network access and appropriate credentials for the instance (called zero footprint inventory collection in the *Gathering FlexNet Inventory* reference)
  - A third-party inventory tool.

In those cases where inventory is received that matches your Microsoft Azure instance records, there are a few special consideration.

## Beware of *terminating* instances

For some special purposes, instances in the Microsoft Azure cloud may be launched, run for a short period (from a few minutes to a few hours), and then shut down until needed again.

It is quite possible to configure the Azure VM Image from which instances are launched so that inventory is collected and uploaded through one or more inventory beacons to your IT Asset Management application server (for details, see [Configuring an Azure VM Image for Instances](#)).

However, the licensing implications are quite different when you *stop* an instance than when you *terminate* an instance:

- The assumption is that a *stopped* instance may be restarted when required, and resume operation with (quite likely)

the same software installed. It is therefore reasonable to calculate license consumption for such an instance, and, if you use the configuration described in [Configuring an Azure VM Image for Instances](#), this happens as usual for virtual machines (whether those are VMs within your enterprise network or instances in the cloud).

- A *terminated* instance cannot be restarted, and if you are choosing BYOL for instances that you terminate, you should check your license terms carefully and make appropriate provision. The working assumption in IT Asset Management is that a terminated instance should no longer consume from your software licenses, just as license consumption stops when you uninstall software from a device, or decommission a hardware asset. As described in [Collecting Inventory from Instances](#), this is achieved by permanently deleting any inventory device record that may have previously been created for the instance that is now terminated. All resources attached to the terminated instance will remain and you can then manually delete these resources as desired. (See the same topic for a way of re-discovering the software load for a terminated instance, and hence its potential license implications.)

The complexity of managing licenses on transient instances is why it is more common to purchase these based on a fully-loaded image provided by (and licensed through) Microsoft Azure. However, if you need repeated runs of an instance for special purposes, and such an instance must include software for which you provide the license (BYOL), consider running such an instance as a stop/start instance, rather than terminating it and re-launching it, to simplify your license management.

### Thirty-minute inventory and IBM PVU licenses

The default schedule for the Azure connector is to have it run every 30 minutes. Coincidentally, this happens to be the same interval that IBM requires for hardware checks of devices linked to IBM PVU licenses (when you have IBM's approval to use IT Asset Management as the source for sub-capacity licensing calculations, for which see the *Sub-Capacity Licensing with IBM PVU* topic in *IT Asset Management System Reference*).

However, each of these coincidental but distinct schedules has its own separate reasons:

- The Azure connector should run every 30 minute to allow it to capture transient information that Azure keeps available for only an hour, so that you can track your terminated instances where required. This is the *connector* schedule, and the connector does not return sufficient inventory detail to satisfy the IBM requirements.
- The hardware check for IBM PVU licenses must run every 30 minutes as part of your amended contractual agreement with IBM. This is an *inventory* schedule that controls operations of FlexNet Inventory Agent embedded on an operational instance.

If you are providing IBM PVU licenses for use in the Microsoft Azure cloud, *and* have the IBM amended agreement to allow sub-capacity calculations within IT Asset Management, these two default schedules conveniently satisfy both requirements within Azure. However, be aware of the following requirements:

- IBM PVU licenses in the BYOL model are not appropriate for short-lived instances that are then terminated (see previous section about the pitfalls of terminating instances with BYOL). Start/stop operations can be used when necessary, as long as the instance is maintained and not terminated.
- Persistent (unchanging) inventory device names and matching instance IDs are critical to this license model, allowing proper calculations of peak consumption values over time. For details about setting a unique but persistent device name for inventory, see [Collecting Inventory from Instances](#).
- As always, full hardware and software inventory (from the instance) is required for IBM PVU licensing. Compliance with your IBM license requires regular software inventory (daily is recommended best practice) as well as the 30-minute hardware check. Use of the FlexNet Inventory Agent is also mandated by your adjusted license agreement. The techniques in [Collecting Inventory from Instances](#) prepare an Azure VM Image from which you can

launch instances with the operational FlexNet Inventory Agent embedded, satisfying those requirements.

- As always, the first full software and hardware inventory must be uploaded and processed on the central application server *before* the 30-minute hardware checks take effect. (This allows the calculation of updated rule targets that identify all devices affected by IBM PVU licensing processes.) Typically the relevant compliance calculations take place overnight, after which the revised device policy must be distributed through the inventory beacon hierarchy to affected installations of FlexNet Inventory Agent, bringing with it the special schedule for the hardware check. All of this process means that in the normal course of events, IBM PVU operations typically are running effectively from the day *after* an instance is launched with PVU-licensed software in operation.

### Uncapped calculations except on dedicated hosts

Some license types allow that, where the total license liabilities for many VMs on the same host theoretically exceed factors like the total number of cores available on the host, the summing process is 'capped' or given an upper limit by the actual capacity of the host. For example, if five VMs on a host are each allocated 2 cores, for a total of 10 allocated cores when the host actually only has 8 cores available, then, just as in reality the time-sharing between the VMs limits the cores in use at any moment, so also the license rules cap (or reduce) the calculated figure (10 cores) to the total capacity of the host (8 cores).

Obviously, capping of consumption calculations requires inventory not only of all the VMs on a host, but also of the host itself. However, in the case of Microsoft Azure instances, inventory details are not available for any host types other than 'dedicated hosts', over which you have full control. This means that on all other, non-dedicated hosts in Azure, capping of license calculations cannot occur.

All calculations for BYOL licensing of instances on non-dedicated Azure hosts are based entirely on the inventory taken from the instances themselves, without capping. This does not expose you to any risk of under-licensing (provided that all instances are returning inventory); the only risk is that you may *perhaps* be over-calculating the consumption that *might* apply on a known host with capping applied.

However, this may not be an issue at all. Carefully check your BYOL terms and use rights. You may find that software publishers already disallow capping in cloud environments, because of the impossibility of tracking instances that may pop up now on this host and now on another, depending on resource allocations within Azure.

## 4

# Configuring an Azure VM Image for Instances

## IT Asset Management (Cloud)

The easiest way to collect inventory from all your cloud-based instances is to ensure that they are launched from an Azure image that already contains everything needed for the collection of FlexNet inventory. The requirements include:

- Installing the latest approved FlexNet Inventory Agent into the image
- Adding a customized configuration file that identifies the upload location for inventory
- Installing a customized schedule to manage FlexNet Inventory Agent in the resulting instances, triggering inventory gathering on start-up (and best practice is to include a backup schedule for further inventory collection in case an instance from this AMI runs for an unexpectedly long time)
- Including the current version of `InventorySettings.xml` that provides advanced functionality for FlexNet Inventory Agent, since the short lifetime does not allow for normal policy-based downloads
- Ensuring a unique name for each instance created from this Azure VM Image.

At this summary level, the requirements are the same for Windows and Linux platforms. In the details, of course, there are platform specifics.



### To prepare an Azure Resource Image:

1. Go to the **Inventory Settings** page (**Data Collection > IT Assets Inventory Tasks > Inventory Settings**).

The **Inventory Settings** page displays.

2. Expand the **Inventory agent for download** section.

3. Collect the template configuration file:

- For a Windows-based AMI, click **Download bootstrapping template file**, and save `mgssetup.ini` to a convenient working folder (such as `C:\temp`). (Do not change the file name.)
- For a Linux-based AMI, in *Gathering FlexNet Inventory*, copy the text from *Agent third-party deployment: Sample UNIX Bootstrap Configuration File* and save it as `mgsft_rollout_response` in a convenient working folder



(such as C:\temp).

4. From the **Inventory agent** drop-down list, select the version of FlexNet Inventory Agent you want to install in your AMI, and save it to your working folder.

In general, install the latest available version, subject to your corporate policies. This provides access to the latest functionality. For example, to include advanced inventory for Microsoft Azure, you must use FlexNet Inventory Agent 13.2.0 or later (see [Appendix 3: Enhanced Inventory Gathered by Agent](#)).

5. In your preferred flat text editor, customize your bootstrapping template file to be used for FlexNet Inventory Agent in your Microsoft Azure environment as follows, saving the edited version in a separate subfolder.

For Windows, the only *mandatory* change is to identify the upload location for gathered inventory, as described below. On Linux, FlexNet Inventory Agent requires both an upload location and a download location. As always, on either platform, experts may also customize other preferences needed for your implementation, as described in the platform-specific topics in the *Gathering FlexNet Inventory* PDF:

- *Agent third-party deployment: Edit the Configuration File for Microsoft Windows*
- *Agent third-party deployment: Configure the Bootstrap File for UNIX.*

The upload (and download) locations require a URL to the host inventory beacon. Best practice is to provide a DNS alias for your chosen inventory beacon, so that when circumstances change, a simple switch of the alias leaves any running and future instances fully operational, without requiring changes to the virtual machine image.

- For **Windows**, in `mgsetup.ini`:
  - a. Locate this section for Common preferences, and uncomment (remove the leading semi-colon) and edit the following settings:

```

;=====
; Registry settings to be created under
; HKLM\Software\ManageSoft Corp\ManageSoft\Common
[Common]
desc0 = UploadSettings\Bootstrap Server\Protocol
val0  = http
desc1 = UploadSettings\Bootstrap Server\Priority
val1  = 100
desc2 = UploadSettings\Bootstrap Server\AutoPriority
val2  = False
desc3 = UploadSettings\Bootstrap Server\Host
val3  = beacon.fnms.com
desc4 = UploadSettings\Bootstrap Server\Port
val4  = 80
desc5 = UploadSettings\Bootstrap Server\Directory
val5  = /ManageSoftRL/

```

- b. Optionally, modify the `Protocol` value if you want to use HTTPS (`val0 = https`).
- c. You *must* customize the `Host` preference (shown above as `val3 = beacon.fnms.com`) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an Azure instance. For an inventory beacon hosted on an Azure instance, you can identify its public and private DNS hostnames on the Azure console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read both its public and private

DNS hostnames in the details pane).



**Tip:** If you have an inventory beacon on an Azure instance, your installations of FlexNet Inventory Agent on other instances can be configured to use the private DNS hostname for accessing the inventory beacon (provided that both are within the scope of your VPC). The private hostname has greater stability, particular if the inventory beacon host is stopped and started from time to time.

- d. Optionally, customize the port setting (for example, if you are switching to the HTTPS protocol, the default port is 443).



**Important:** The *Directory* preference is mandatory, and must be set as shown above.

- e. Create a subfolder (such as SysPrepAgent) and save your edited `mgssetup.ini` there. Do not change the file name, which is mandatory.
- For **Linux**, in `mgsft_rollout_response`:
  - a. In the first section of the file, customize the setting for the download location (see also the comments for the upload location, next):

```
# The initial download location(s) for the installation.
# For example, http://myhost.mydomain.com/ManageSoftDL/
# Refer to the documentation for further details.
MGSFT_BOOTSTRAP_DOWNLOAD=http://beacon.fnms.com:8080/ManageSoftDL/
```

- b. In the next section of the file, customize the setting for the reporting (upload) location:

```
# The initial reporting location(s) for the installation.
# For example, http://myhost.mydomain.com/ManageSoftRL/
# Refer to the documentation for further details.
MGSFT_BOOTSTRAP_UPLOAD=http://beacon.fnms.com:8080/ManageSoftRL/
```

You may customize the protocol to HTTPS if required, and omit or customize the port number as required. You *must* customize the host URL (shown above as `beacon.fnms.com`) to your chosen DNS alias (or preferred host setting) and your own enterprise name if the inventory beacon is located in your enterprise network rather than on an Azure instance. For an inventory beacon hosted on an Azure instance, you can identify its DNS hostname on the Azure console (select **Instances** in the navigation pane, choose the instance hosting the inventory beacon, and read its DNS hostname in the details pane).



**Important:** In each case, the trailing `/ManageSoftDL/` (download location) or `/ManageSoftRL/` (upload or reporting location) is mandatory, and must be specified as shown above.

- c. In the last line of the file, switch the value to prevent policy running after installation, as short-lived instances do not have time to wait for policy and its outcomes:

```
MGSFT_RUNPOLICY=0
```

- d. Create a subfolder (such as SysPrepAgent) and save your edited `mgsft_rollout_response` there. Do not change the file name, which is mandatory.
  - e. Unless you already have a strategy and practice that gives every Linux instance a unique name, also create a

new text file for your Linux virtual machine image, called `tmpconfig.ini`, with exactly the following two lines:

```
[ManageSoft\Common]
MachineID=
```

This file is deliberately incomplete at this stage, and is to be completed at instance start-up, as described later.

- For either platform, copy the following and paste it into a plain ASCII text file, and save it in your `SysPrepAgent` folder as `DefaultSchedule.nds` (this file name is mandatory). Of course, you may choose to unwrap those lines wrapped for readability here.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Schedule PUBLIC "Flexera Inventory Manager schedule"
"http://www.managesoft.com/dtd/schedule.dtd">

<Schedule NAME="Default Schedule" TIMEDATESTAMP="20180903T044104Z"
SCHEDSCHEMA="60">
  <Event NETWORK="False"
    NAME="Generate Inventory"
    CATCHUP="Never" ID="{F065A477-B500-4611-8340-3BF313FB785B}">
    <LogicalCommand PARAM=" -o UserInteractionLevel=Quiet"
      ACTION="Report" COMPONENT="Tracker"/>
    <Trigger TYPE_PARAM="0" TIMESTART="143950"
      DATESTART="20180903" TYPE="Logon" MAXDELAY="0"/>
  </Event>
  <!-- Following two events are optional (see notes) --!>
  <Event NETWORK="False"
    NAME="Generate Inventory" IDLEDURATION="86400"
    CATCHUP="Never" ID="{F065A477-B500-4611-8340-3BF313FB785B}">
    <LogicalCommand PARAM=" -o UserInteractionLevel=Quiet"
      ACTION="Report" COMPONENT="Tracker"/>
    <Trigger TYPE_PARAM="1" TIMESTART="001000"
      DATESTART="20180903" TYPE="Daily" REPEAT="21600"
      TIMEWINDOW="3600" DURATION="86400"/>
  </Event>
  <Event NETWORK="False"
    NAME="Upload Client Files"
    CATCHUP="Never" ID="{9CB87BAE-2829-498C-8643-10A9BD3BFA56}">
    <LogicalCommand PARAM="-a"
      ACTION="Upload" COMPONENT="Uploader"/>
    <Trigger TYPE_PARAM="1" TIMESTART="002000"
      DATESTART="20180903" TYPE="Daily" REPEAT="600"
      DURATION="86400"/>
  </Event>
</Schedule>
```

The first event in this schedule triggers machine inventory collection on startup (TYPE="Logon"), and the tracker component automatically attempts an upload to the bootstrap inventory beacon as soon as inventory

gathering is complete. If you are certain that all instances from this image are short-lived, only this event is required. If you are using an instance, and you choose to stop and restart your instance (rather than to terminate and re-launch it), this inventory is checked on each restart.

In addition, two further schedule events provide easy management of cases where an instance proposed to have a short life in fact keeps on running for some longer time.

The second event also generates inventory, and is triggered every 6 hours (REPEAT="21600", with all times expressed in seconds), to start within an hour thereafter (TIMEWINDOW="3600"). However, this inventory collection is different than the first case, because here the FlexNet Inventory Agent does not repeat the inventory collection if it has been successful within the last 24 hours (IDLEDURATION="86400"). This means that, effectively, the FlexNet Inventory Agent checks four times a day whether there has been a successful inventory collection in the last 24 hours, and if not, it runs a new inventory collection.

In a perfect world, the third event is redundant because the tracker component uploads inventory as soon as it is collected. This third, separate upload event is a catch-up to work around transient network issues. It checks every 10 minutes to see whether any inventory files have failed to upload (and are therefore still waiting in the staging folder). In normal cases, nothing is waiting, and the uploader shuts down immediately, causing negligible load on the instance. But where there have been networking issues, upload is attempted again every 10 minutes until all files are successfully uploaded and removed from the staging folder.

7. On a convenient inventory beacon, navigate to %CommonAppData%\Flexera Software\Staging\Common\ClientConfiguration, take a copy of InventorySettings.xml, and save it to your SysPrepAgent working folder.

This completes the preparation of materials for your virtual machine image for instances of this expected life. Now continue to use these materials to prepare the virtual machine image. For both platforms, the high-level process is to customize a running instance, and then save it as a virtual machine image.

**For a Windows-based VM Image** please refer to the following link:

- <https://docs.microsoft.com/en-us/azure/virtual-machines/windows/tutorial-custom-images>

**For a Linux-based VM Image** please refer to the following link:

- <https://docs.microsoft.com/en-us/azure/virtual-machines/linux/tutorial-custom-images>

8. Following the Azure documentation, select (or create), launch and connect to your chosen instance from which you are creating your Azure VM Image.

9. To customize your instance:

- a. Ensure that the destination folder exists for your installation of FlexNet Inventory Agent, and if not create it.

The default paths for each platform are:

- On Windows, C:\Program Files (x86)\ManageSoft
- On Linux, /opt/managesoft/bin.

- b. From your local SysPrepAgent working folder, copy these files into the destination folder on your instance:

- DefaultSchedule.nds
- InventorySettings.xml

- Only on Windows, `mgssetup.ini`
  - Only on Linux, `tmpconfig.ini`.
- c.** For Linux only, also copy your amended `mgsft_rollout_response` file to `/var/tmp/` on your instance.
- d.** Collect the downloaded installer for FlexNet Inventory Agent from your working folder, and run the installer on your instance, installing FlexNet Inventory Agent into the prepared destination folder.

The installer automatically configures the custom settings declared in the configuration file (either `mgssetup.ini` or `mgsft_rollout_response`, depending on platform). For more information about running the installation, see the *Agent third-party deployment: Details* chapter of the *Gathering FlexNet Inventory* PDF, focusing on the installation topic for your preferred platform.

- e.** When installation is complete, in the same destination folder and for both platforms, run the following command to install your default schedule:

```
start ndschedag.exe -t machine .\DefaultSchedule.nds
```

This virtual machine image is now ready. As soon as each instance is spawned, the embedded FlexNet Inventory Agent gathers inventory from the instance, and immediately uploads it to the bootstrap inventory beacon. Depending on the resources on each instance, this process may take around 2-5 minutes, after which the instance can be stopped (but remember, not terminated if you wish to track related software licenses in IT Asset Management).

## 5

# Appendices: Technical Data

## IT Asset Management (Cloud)

The following topics do not include content you need for set-up or operation of the Microsoft Azure connector. These are the deep-down technical data that you can ignore until you have a real need.

## Appendix 1: Cmdlets in Azure PowerShell

### IT Asset Management (Cloud)

The connection from IT Asset Management to the Microsoft Azure service relies on the Az Tools for Windows PowerShell (as described in [Prerequisites and Setting Up](#)). Within that toolset, the connector uses the following PowerShell cmdlets:

- `Connect-AzAccount` — To log into Microsoft Azure
- `Disconnect-AzAccount` — To log out of Microsoft Azure
- `Get-AzEnvironment` — To identify the environments currently available in Microsoft Azure (such as `AzureChinaCloud`, `AzureCloud`, `AzureGermanCloud`, `AzureUSGovernment`)
- `Get-AzLocation` — To identify the available Azure geographic regions of the instance type (size in Microsoft term) where your instances may be running
- `Get-AzSqlVM` — To return the list of the SQL virtual machines you own, and then collect the cloud license model
- `Get-AzVMSize` — To return the list of the instances type based on location
- `Get-AzVM` — To return the list of the instances (virtual machines) you own, and then collect summary inventory details on each one
- `Search-AzGraph` — To return the list of instances for a batch of subscriptions.

# Appendix 2: Data Imported by Microsoft Azure Connector

## IT Asset Management (Cloud)

The PowerShell connector for Microsoft Azure returns information about each Azure instance, and its host. All instances are subsequently displayed in the **Cloud Service Provider Inventory** page; and those tracking installations of Microsoft Windows Server or SQL Server also display the cloud license model (BYOL for bring your own license, or PAYG for pay as you go) in use for each instance. Hosts are never displayed in this page, as in general hosting is variable and controlled by Microsoft Azure.



**Tip:** One exception where you have direct control is the 'dedicated host'. Like all other host types, dedicated hosts do not appear in the **Cloud Service Provider Inventory** listing; but they do automatically appear in inventory device listings, such as the **All Inventory** page, where you can search for them using the **Hosted in** property. Any instances from the **Cloud Service Provider Inventory** listing that are hosted on your dedicated host also display a link to the dedicated host properties in the **Host** column.

## Instance data



**Tip:** All kinds of instances return this set of data, whether they are virtual machines or bare metal instances. The latter return an `Instance` type starting with `i3..`

There is one condition where the **Cloud Service Provider Inventory** page includes a hyperlink to open the inventory device properties page for the instance: if the data gathered through the connector matches inventory collected from another source (one possible example is having FlexNet Inventory Agent running locally on the instance). In this case, linked records exist in both the `ComplianceComputer` and `CloudServiceInstance` tables.

Some data is additional to the normal content collected as inventory. For example:

- The `InstanceTenancyID`, used for identifying whether the current device is a `Dedicated` instance, or is an instance running `On dedicated host`, is imported into the `CloudServiceInstance` table, and, as **Availability type**, is visible in both the **Cloud Service Provider Inventory** page and the **Cloud Hosting** tab of the inventory device properties (where inventory exists for the instance)
- The cloud hosting model is displayed in the **Cloud Service Provider Inventory** page as **SQL server mobility** and/or **Windows server AHB** – both of these columns may display PAYG or BYOL, where the license model is known (and remain blank when it is not). Note that the connector returns only the license model that is applicable *if and when* the relevant product is installed on the instance – it is *not* an inventory report showing whether either product is installed. If you subsequently add full software and hardware inventory for this instance, and the inventory is matched to this record in the next license compliance calculations, a link is added in the **Device name** column so that you can open the inventory device properties and see what software is actually installed.



**Tip:** Azure "bare metal instances" behave like other instances in terms of data gathered and records created.

When there is an inventory device record for an instance, the inventory-related values imported through the Microsoft Azure connector (and shown below in alphabetical order) are displayed in the separate **Cloud Hosting** tab of the inventory device properties:

- **Availability zone**
- **Cores**
- **Image ID**
- **Instance ID** (Azure VmId, which is also UUID of the VM)
- **Instance type**
- **Instance region**
- **Last known state** (state of the instance, will be mapped to Started, Stopped, Suspended, Terminated, or Unknown).



**Tip:** The inventory device record does not store any information about cloud license model for any cloud instance. Cloud license model for an instance running SQL Server or Windows Server applications is available in the **Cloud Service Provider Inventory** page under the columns **SQL server mobility** and **Windows server AHB** respectively.

## Appendix 3: Enhanced Inventory Gathered by Agent

### IT Asset Management (Cloud)

As well as the high-level data gathered from the Microsoft Azure connection, FlexNet Inventory Agent (version 13.1 or later) gathers addition inventory data for each Azure instance where FlexNet Inventory Agent is locally installed. To do so, FlexNet Inventory Agent accesses the following Azure-internal URL:

- `curl -H Metadata:true "http://169.254.169.254/metadata/instance?api-version=YYYY-MM-DD"` — you must replace YYYY-MM-DD with the api-version date to return the unique identifier for the Azure instance. For example:

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance?api-version=2018-10-01"
```

- To view the api-version dates enter `curl -H Metadata:true "http://169.254.169.254/metadata/instance"` which will return an error message which identifies the api-versions. For example:

```
{
  "error": "Bad request. api-version was not specified in the request. For more information refer to aka.ms/azureimds", "newest-versions": [
    "2018-10-01",
    "2018-04-02",
    "2018-02-01"
  ]
}
```

- `http://169.254.169.254/metadata/instance` returns a JSON-formatted value containing data about the instance where FlexNet Inventory Agent is locally installed. For example:



```
{
  "devpayProductCodes" : null,
  "marketplaceProductCodes" : [ "1abc2defghijklm3nopqrs4tu" ],
  "availabilityZone" : "us-west-2b",
  "privateIp" : "10.158.112.84",
  "version" : "2017-09-30",
  "instanceId" : "i-1234567890abcdef0",
  "billingProducts" : null,
  "instanceType" : "t2.micro",
  "accountId" : "123456789012",
  "imageId" : "ami-5fb8c835",
  "pendingTime" : "2018-11-19T16:32:11Z",
  "architecture" : "x86_64",
  "kernelId" : null,
  "ramdiskId" : null,
  "region" : "us-west-2"
}
```

These two data points are then embedded within the normal .ndi inventory file as a pseudo-hardware WMI class called MGS\_CloudMetadata. The class looks similar to this example (lines wrapped for presentation):

```
<Hardware class="MGS_CloudMetadata" Name="ec2amaz-vrkiu0p" Evidence="metadata">
  <Property Name="instance-id" Value="i-1234567890abcdef0"
    Evidence="/latest/meta-data/instance-id"/>
  <Property Name="document" Value="ewogICJwcm12YXRlSXAiIDogIjE3Mi4zMS4yLjExNiIs...."
    Evidence="/latest/dynamic/instance-identity/document"/>
</Hardware>
```

After a full inventory import, these details are eventually saved in the CloudServiceInstance table in the compliance database.

# XIII

## Microsoft Intune Adapter

### IT Asset Management (Cloud)

The **Microsoft Intune Adapter**, provided by Flexera, allows you to collect inventory from Microsoft Intune and import it into IT Asset Management. Microsoft Intune is a cloud-based endpoint management solution. It manages user access and simplifies app and device management across all of your devices, including mobile devices, desktop computers, and virtual endpoints.

Data is collected from Microsoft Intune and imported into IT Asset Management, giving you visibility of managed devices, user data, and managed and unmanaged applications installed on corporate owned devices in your Microsoft Intune estate. To see a full list of what data is imported and visible in IT Asset Management, see [What data is retrieved](#).

IT Asset Management uses the imported data collected from Microsoft Intune to calculate license consumption. **Note:** For application recognition, application evidence must be recognized by the Application Recognition Library (ARL), and linked to the appropriate license in order for compliance calculations to proceed. For more information on application recognition, see the *Application evidence* section in [What data is retrieved](#).

### Prerequisites

- The Microsoft Intune Adapter is configured on the FlexNet Beacon, for which version 22.4 onwards of the FlexNet Beacon is required. For steps on how to upgrade the FlexNet Beacon, see [Upgrading Inventory Beacons](#) in the online help. For FlexNet Beacon prerequisites, see [Prerequisite Software](#) in the *System Requirements and Compatibility* online help.



**Important:** Prior to the release of IT Asset Management 2024 R1.4, the Microsoft Intune adapter used Active Directory Authentication Library (ADAL) for authentication. ADAL has since been superseded by Microsoft Authentication Library (MSAL) and is no longer supported by Microsoft. From IT Asset Management 2024 R1.4 onwards, the `Microsoft.IdentityModel.Clients.ActiveDirectory.dll` has been replaced with `Microsoft.Identity.Client.dll`.

- The Microsoft Intune Adapter is supported by version 2022 R2.4 of IT Asset Management onwards.
- The Microsoft Intune Adapter supports recognition of MAC OS device information from version 2023 R2.1 of IT Asset Management onwards.
- Register an application in Azure Active Directory to connect to Microsoft Graph. See [Register an Application in Azure Active Directory](#) for detailed steps. **Important:** Ensure to register an application before creating the Microsoft Intune

connection on the FlexNet Beacon.

- A connection to Microsoft Intune on the FlexNet Beacon. See [Creating the Microsoft Intune Connection](#) for detailed steps.

## 1

# What data is retrieved

## IT Asset Management (Cloud)

View each of the tables below to see what device, user and application data is retrieved from Microsoft Intune. This data is retrieved by running a PowerShell adapter which calls the Microsoft Graph API and Microsoft Report Graph API. See [Register an Application in Azure Active Directory](#) for more information.



**Important:** In order to import data into IT Asset Management, Flexera's Microsoft Intune adapter relies on the following Microsoft APIs: Microsoft Graph API, Microsoft Report Graph API. The Microsoft Graph API retrieves user data. The Microsoft Report Graph API retrieves device and installed applications data. Both APIs apply restrictions in terms of what data is available for collection or can be exported. At present, only the data specified in the following subsections is retrievable: Device information, User information and Application installer evidence.

## Device information



**Note:** The Microsoft Intune Adapter only supports the collection of device information from devices running Microsoft Windows and MAC OS operating systems.

Retrieved data	Description
<b>Computer name</b>	The system name of the device.
<b>Inventory date</b>	The date when inventory information was last collected for the device.
<b>Last logged on user</b>	The last logged on user on a given device.
<b>Manufacturer</b>	The company making and selling the device.
<b>Model no</b>	The manufacturer's model name or number for the device.
<b>Operating system</b>	The operating system running on the device.
<b>OS version</b> (as build number)	The operating system version build number running on the device. For example, 10.0.19045.2728.
<b>OS edition</b> (not retrieved for MAC OS)	The operating system edition running on the device. For example, Pro, Enterprise, Home or Education.

Retrieved data	Description
<b>Serial no</b>	The serial number of the device, attempting to uniquely identify either the hardware (for a stand-alone device) or the virtualization container (for a virtual machine), as reported in inventory.



**Note:** Device information retrieved from Microsoft Intune is limited, and the following limitations apply:

- Hardware information such as MAC address, IP address, cores, threads, sockets and so on is not retrieved.
- Host information for virtual machines is not retrieved.
- The domain for devices is not retrieved.
- Only WMI evidence is used for OS recognition.
  - The specific Windows OS version (Windows 10, Windows 11) is not retrieved and will not be available to view in the **All Inventory** screen. However, the operating system version build number is retrieved. For example, 10.0.19045.2728. You can view the complete version string (build number) by navigating to **Inventory Device Properties** for an individual device and selecting the **Applications** tab. **Note:** Only build numbers pertaining to Windows 10 or 11 operating systems are supported.



**Important:** For Windows and Apple devices, only devices with an ownership type of Corporate are captured. Personal and Android devices are not retrieved.

## User information



**Note:** The Microsoft Intune Adapter only supports the collection of user information from devices running Microsoft Windows and MAC OS operating systems.

Retrieved data	Description
<b>Domain</b>	The name of the domain to which the user belongs.
<b>Email</b>	The email address of the user.
<b>First name</b>	The first name of the user.
<b>Last name</b>	The last name of the user.
<b>Sam account name</b>	The logon name of the user account.
<b>Username</b>	The name given to the user to uniquely identify them.



**Note:** The domain information for the user is not retrieved directly from Microsoft Intune. Domain information is determined from the user email address.

## Application evidence



**Note:** The Microsoft Intune Adapter only supports the collection of application evidence from devices running Microsoft Windows and MAC OS operating systems.

Retrieved data	Description
<b>Raw name</b>	The display name of the installer evidence as retrieved from the inventory source.
<b>Raw version</b>	The version for the installer evidence as retrieved from the inventory source.



**Note:** Application usage data is not retrieved.



**Note:** Application publisher information is not retrieved. For application recognition, only installer evidences for applications installed on devices are identified.

- If the incoming installer evidence from Microsoft Intune (Raw name, Raw version, without publisher information) matches with existing installer evidence in the ARL, then the application will be recognized.
- For retrieved evidence without publisher information, if the first word of the installer evidence's name (separated by either space or .) is a publisher name that already exists in the ARL, then it is matched with the ARL's installer evidence and the application will be recognized.

## 2

# Register an Application in Azure Active Directory

IT Asset Management (Cloud)

In order to establish a connection with Microsoft Intune on the FlexNet Beacon, you need to specify a **Tenant ID**, **Client ID** and **Client Secret** when setting up the PowerShell source connection. To get this information, you will have to register an application in your Azure Active Directory portal to connect with the **Microsoft Graph API** ([portal.azure.com](https://portal.azure.com)). The **Microsoft Graph API** is called to get the information from **Intune discovered apps**. Intune discovered apps is a report that lists detected apps on Microsoft Intune enrolled devices in your tenant, and acts as a software inventory for your tenant.



**Note:** Intune **discovered apps** report refreshes every 7 days from the time of enrollment (not a weekly refresh for the entire tenant). The only exception to this refresh cycle for the **Discovered apps** report is application information collected through the Intune Management Extension for **Win32 Apps**, which is collected every 24 hours.



**Use the following procedure to register an application in your Azure Active Directory for connecting to the Microsoft Graph API.**

1. Log into your Azure Directory portal ([portal.azure.com](https://portal.azure.com)), navigate to **App registrations** and create a **New registration**.
2. For your new registration, specify the following prior to selecting **Register**:
  - a. A **name** for the new application.
  - b. Who can use the application. **Important:** select "Accounts in this organizational directory only ([ORG\_NAME] only - Single tenant)".
  - c. Provide a redirect URI (optional).
3. After registering, take a note of the **Application (Client) ID**. This is required when creating the PowerShell source connection.
4. In the left pane, under **Manage**, select **Certificates & secrets** and create a new client secret. You will have two options: **New client secret** or **Upload certificate**. Select **New client secret**.



**Important:** Ensure to take a note of the Client secret **Value** before changing screens, as this is displayed one-time only. Later, the value will be masked and you will not be able to retrieve it. Your only option, is to delete the application and create a new one. Similar to the Application (Client) ID, this is required when creating the PowerShell source connection.

5. You now need to request API permissions. In the left pane, under **Manage**, this time select **API permissions** and select **Add a permission**.
6. In the "Request API permissions" screen, do the following:
  - a. Select **Microsoft APIs** followed by **Microsoft Graph**.
  - b. Select **Application Permissions**.
  - c. Expand the **DeviceManagementApps** dropdown and check **DeviceManagementApps.Read.All**.
  - d. Expand the **DeviceManagementManagedDevices** dropdown and check **DeviceManagementManagedDevices.Read.All**.
  - e. Expand the **Directory** dropdown and check **User.Read.All**.
7. Once you have added all the above permissions, select **Grant admin consent for [ORG\_NAME]**.



3


# Creating the Microsoft Intune Connection

IT Asset Management (Cloud)

Use the following procedure to create a connection to Microsoft Intune on the FlexNet Beacon. A connection is required to be configured for a single source for inventory collection. The inventory beacon is responsible for uploading the data to the central operations databases of IT Asset Management.


 **To create the Microsoft Intune connection in the FlexNet Beacon UI:**

1. Log into your selected inventory beacon.

 **Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.

2. In the navigation pane on the left, select the **Inventory systems** page. To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.

The **Create PowerShell Source Connection** dialog appears.

 **Tip:** The **New...** button defaults to creating a connection for Microsoft SQL Server. If you use the down arrow on the split button, you can choose between *SQL Server*, *Spreadsheet*, *PowerShell*, and *Other* connections. However, while you are creating a connection to a Microsoft SQL Server database (regardless of the **Source Type** of the connection), use only the *SQL Server* option.

3. Complete the values in the dialog, as follows:

Control	Comments
Connection Name	A descriptive name for this connection, such as Intune. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name.
Source Type	Select <b>Microsoft Intune</b> from the list.

Control	Comments
<b>Use Proxy</b>	Optionally, if you use a proxy server to enable Internet access, complete (or modify) the values in the <b>Proxy Settings</b> section of the dialog box in order to configure the proxy server connection.
<b>Proxy Server</b>	Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format <code>https://ProxyServerURL:PortNumber</code> , <code>http://ProxyServerURL:PortNumber</code> , or <code>IPAddress:PortNumber</code> . This field is enabled when the <b>Use Proxy</b> check box is selected.
<b>Username and Password</b>	If your enterprise is using an authenticated proxy, specify the <b>username</b> and <b>password</b> of an account that has credentials to access the proxy server that is specified in the <b>Proxy Server</b> field. These fields are enabled when the <b>Use Proxy</b> check box is selected.
<b>Microsoft Intune</b>	In order to establish a connection with Microsoft Intune, you need to specify a <b>Tenant ID</b> , <b>Client ID</b> and <b>Client Secret</b> . See <a href="#">Register an Application in Azure Active Directory</a> for detailed steps on how to retrieve the required information.
<b>Connection is in test mode (do not import results)</b>	Controls the uploading and importing of data from this connection: <ul style="list-style-type: none"> <li>When this check box is clear, the connection is in production mode, and data collected through this adapter is uploaded to the central server and (in due course) imported into the database there.</li> <li>When the check box is set: <ul style="list-style-type: none"> <li>The adapter for this connection is exercised, with data written to the intermediate file in the staging folder on the inventory beacon (<code>%CommonAppData%\Flexera Software\Beacon\IntermediateData</code>)</li> <li>The immediate upload that normally follows data collection is suppressed, so that you can inspect the contents of the file</li> <li>The catch-up process that retries stalled uploads, normally scheduled overnight, runs as usual and uploads the file to the central server</li> <li>At the central server, the file contents are discarded (and not imported into the central database).</li> </ul> </li> </ul>
<b>Overlapping Inventory Filter</b>	This control does not apply to Microsoft Intune adapter, and you may leave it at the default setting.

#### 4. Click **Test Connection**

This will make sure that you can successfully connect to both the endpoint and the specified **username** and **password** are correct. Note: a request to log into the API is part of the test connection.

- If the connection is successful, click **OK** to close the message. Click **Save** to complete the addition. The connection is added to (or updated in) the list.
- If the connection is unsuccessful, the appropriate error message will display. Click **OK** to close the message.

Edit the connection details and retest the connection.

You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.

5. In the FlexNet Beacon PowerShell Source Connection dialog, click **Save** to save the connection.
6. Select your new connection from the displayed list, and click **Schedule...**
7. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.
8. At the bottom of the FlexNet Beacon interface, click **Save**, and if you are done, also click **Exit**.

After a successful data import - managed devices, user data, and managed and unmanaged applications installed on corporate owned devices in your Microsoft Intune estate will be visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** page of FlexNet Beacon, see the *Inventory Systems Page* in the online help. For scheduling data imports through this connection, see *Scheduling a Connection*, also in the online help.

# XIV

## Oracle Cloud Infrastructure (OCI) Adapter (beta)

IT Asset Management (Cloud)

This chapter introduces the OCI adapter, with a particular focus on managing licenses that you assign to Oracle Autonomous Databases (often called "bring your own [software] license", or BYOL).



**Note:** *The subject of this document is not related to the purchase of Oracle Autonomous Databases fully provisioned by Oracle Cloud Infrastructure (licenses included). Purchasing Oracle Autonomous Databases fully provisioned by Oracle Cloud Infrastructure (licenses included) is of less interest to your license management team, and perhaps more important to your finance team.*

The OCI adapter serves one main purpose—It performs discovery of Oracle Autonomous Database installations in Oracle Cloud Infrastructure and Exadata Cloud@Customer. From the data uploaded by the OCI adapter, IT Asset Management automatically creates discovered devices, inventory devices, database instances and database option records.

This topic focuses on background information, and the prerequisite software you need in order to build license management around the OCI adapter. For step-by-step instructions for configuring the connection to Oracle Cloud Infrastructure and choosing the correct connection method for your business processes (there are three methods available), see [Managing Oracle Cloud Infrastructure \(OCI\) Connection](#) in the *Online Help*.



**Note:** *The beta version of the OCI adapter allows for only one connection to Oracle Cloud Infrastructure (OCI) to be configured.*

### BYOL Licensing Considerations

The OCI adapter gathers information for every Oracle Autonomous Database accessible to the querying user, though it will only import those databases configured to use the **Bring your own license** (BYOL) model. This means all databases using the **License included** model will not be shown in IT Asset Management.

The OCI adapter gathers a snapshot of your Oracle Autonomous Database inventory at the moment of the scan, including stopped instances. Autonomous database instances terminated before the scan will not be returned.

# 1

## Prerequisites for the OCI Adapter

IT Asset Management (Cloud)

The following FlexNet Beacon server prerequisites are required to use OCI adapter:

- An inventory beacon running the appropriate release of FlexNet Beacon software
- PowerShell 5.1 or later
- A web browser that can access IT Asset Management
- The ability to log into the inventory beacon, and run FlexNet Beacon using an account with administrator privileges.

# XV

## Oracle Enterprise Manager Adapter

### IT Asset Management (Cloud)

Oracle Enterprise Manager monitors and manages other Oracle software installed on customer sites. Therefore it is a useful aid in gathering inventory from Oracle systems.

The Oracle Enterprise Manager (OEM) adapter, from Flexera, connects to Oracle Enterprise Manager, and extracts a file of connection information for the Oracle systems it monitors. This file is in a standard format (`TNSNames . ora`) used by Oracle. (In fact, if you already have files of the same name generated by Oracle, you can simply copy these to the appropriate folder on an inventory beacon. This may be a viable alternative to using this adapter.)

When the file is saved to a special location on an inventory beacon (and the inventory rules for this inventory beacon allow processing of `TNSNames . ora` files), the connection information in it can be used by FlexNet Manager for Datacenters, a separately-licensed product within IT Asset Management, to collect software inventory information. This document covers the set up and configuration to use the adapter as part of an Oracle inventory solution.



**Note:** The sole purpose of the OEM adapter is to prepare a `TNSNames . ora` file as a method of discovery of Oracle Database servers. There is a completely unrelated process where the normal FlexNet inventory of the database instance used by Oracle Enterprise Manager (the **OEM repository**) also reveals the names of the Oracle database instances it manages, as well as any Oracle options that have been enabled through the related console for Oracle Enterprise Manager. This latter process requires neither an adapter nor any special configuration; it is a natural by-product of gathering inventory from the **OEM repository**, whether by a locally installed FlexNet Inventory Agent, by direct inventory collection through an inventory beacon, or by any other means of gathering FlexNet inventory from the **OEM repository** database instance.

### Terminology

Throughout, the Flexera adapter is referred to as the **OEM adapter**. The database for Oracle Enterprise Manager (to which the OEM adapter connects) is referred to as the **OEM repository**.

The OEM adapter is tested for use with Oracle Enterprise Manager releases 12.1–13.5.

## 1

# Understanding the Oracle Enterprise Manager Adapter

IT Asset Management (Cloud)

As well as providing a functional overview of the straight-forward OEM adapter itself, this chapter provides additional background about the other aspects of your system that collaborate to provide Oracle introspection and inventory reporting. As well, this chapter covers the prerequisites, content, and delivery of the OEM adapter.

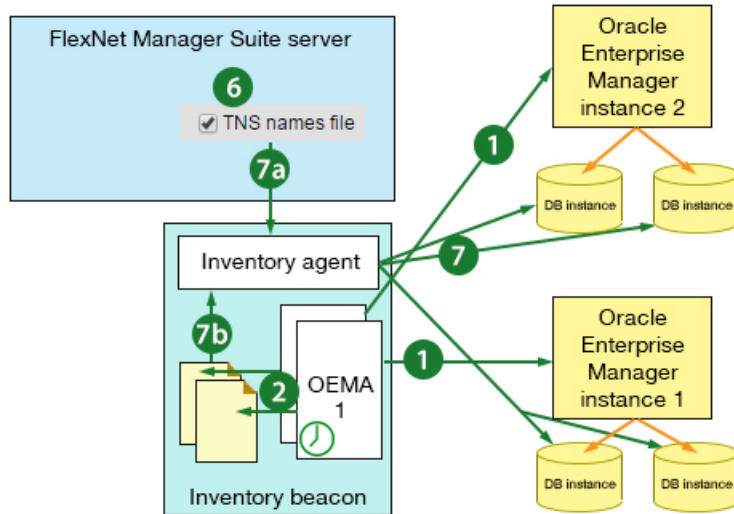
## How the Adapter Assists in Inventory Gathering

IT Asset Management (Cloud)

The OEM adapter provides an alternative or additional method of discovering Oracle Database servers in your computing estate. Discovery (by one means or another) is a prerequisite for collecting software inventory from the Oracle servers.

The OEM adapter is installed on a convenient computer that has network access to Oracle Enterprise Manager and its OEM repository. Typically, this computer may be a FlexNet Beacon. Multiple instances of the OEM adapter may be installed (on one or more computers), each of which can connect with one instance of Oracle Enterprise Manager.

Numbers in the diagram (where "OEMA" identifies the OEM adapter) correspond to the process description below:



**The process runs as follows:**

1. The Windows scheduled task triggers the OEM adapter, which contacts its instance of Oracle Enterprise Manager, and collects the connection data from the OEM repository.
2. The OEM adapter formats the data into a `TNSNames . ora` file, and saves it to a special location on the inventory beacon.

You may configure the name of the saved file, and, if multiple instances of the OEM adapter save files to the same inventory beacon, you *must* modify the file names so that the files do not overwrite each other.



**Note:** This completes the function of the OEM adapter. The remainder of the process is standard operating procedure for IT Asset Management with the FlexNet Manager for Datacenters product installed. If you already have an operational system, several of these steps may already be completed.

3. You set up a special account with read-only permissions on your Oracle Database for all the tables and views needed for Oracle introspection. One helpful practice is to use the same purpose-driven set of credentials on all servers. A utility is available to help created the account(s) required with the correct permissions.
4. The same credentials must be recorded in the Password Manager on each of the relevant inventory beacons.
5. In the web interface for IT Asset Management, you define a subnet (or several subnets) that contain the Oracle Database servers of interest; and you assign to these subnet(s) the inventory beacon(s) where the `TNSNames . ora` files are being saved. (Assignments are distributed automatically to inventory beacon(s), along with rules.)
6. Continuing in the web interface, you create a rule with an action including Oracle inventory collection, with the option to use TNS name file selected. The rule also has a target which matches the subnet(s) you are interested in. This rule is automatically distributed to inventory beacon(s).
7. On the inventory beacon(s) of interest:
  - a. The rule and assignment are received.
  - b. The inventory beacon assesses these, concludes that it is authorized to act, and looks for a any `* . ora` file(s) in the special path on the inventory beacon.



- c. For any Oracle server which is both within the authorized subnet and listed in the `.ora` file, the inventory beacon checks for credentials in its Password Manager, and tests them (from the most closely matching to the most general) until either one gets a response (success), or there are no further applicable credentials (failure).
  - d. When successfully logged in, the FlexNet Inventory Agent running on the inventory beacon, uses the credentials to read the data necessary for Oracle introspection. It writes the data as an Oracle inventory file into its staging folder.
  - e. Within a minute of completion, the regular upload process starts moving this inventory file to the central application server (or, in a multi-server installation, the inventory server).
8. After the next inventory import and resulting consumption calculations, the Oracle inventory is available in the web interface for IT Asset Management; and the Oracle servers originally identified in the `TNSNames .ora` file are visible in the **All Discovered Devices** listing, displaying Yes in the **Oracle** column.

## Prerequisites for the OEM Adapter

### IT Asset Management (Cloud)

The OEM adapter must be installed on a computer with network access to Oracle Enterprise Manager, and requires read access to certain tables and data views there (the required permissions are listed in [Grant Permissions to Account](#)).

The OEM adapter requires that, on the computer where it will execute, the Oracle client version 12.1 is installed. Only the 32-bit version is supported: even for a Windows 64-bit computer, use 32-bit ODAC 12c Release 1 (12.1.0.1.0).

To take advantage of the information gathered by the OEM adapter, there are also the following requirements on the remainder of the system. Once the OEM adapter saves a `TNSNames .ora` file, the subsequent gathering of Oracle inventory requires that:

- The FlexNet Inventory Agent can access each Oracle system. This can be achieved either by installing the FlexNet Inventory Agent on the target Oracle server(s), or by remote execution (Zero-footprint inventory gathering, a term further explained in *Gathering FlexNet Inventory*).
- You have FlexNet Manager for Datacenters, a separately licensed product within IT Asset Management.



**Tip:** You can check whether your implementation has this product licensed in the web interface for IT Asset Management:

1. Go to the **IT Assets License** page (**Administration > IT Asset Management Settings > IT Assets License**).
2. Check the list of **Subscribed and purchased products** on the right, looking for a card for FlexNet Manager for Datacenters. If the card is present, you have this product licensed.

## Components

### IT Asset Management (Cloud)

The OEM adapter is supplied as an installer, called `OEMAdapter .exe`, which installs the following:

- OEM adapter executable and dependencies

- OEM adapter configuration file
- OEM adapter scheduled task, which can be created during installation if desired.

## Download Adapter Tools Archive

IT Asset Management (Cloud)

The Adapter Tools archive includes content for many adapters, and is updated on the Flexera website from time to time.

Start this procedure using a web browser on a computer that has good network accessibility from all the machines needing installations for your adapter.



### **To download the adapter tools archive:**

1. Download the latest Adapter Tools for IT Asset Management *version*.zip archive from the Flexera Product and License Center:

- a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.

- b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of Adapter Tools for IT Asset Management 2024 R2.3.zip commences.

- c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as C:\temp on a central, accessible server).

If your browser saves the file to a default location (such as your Downloads folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the zip archive, and choose **Extract All...**

The folders are now available for the range of adapters in the Adapter Tools archive.

## 2

# Installing the Adapter, and More

IT Asset Management (Cloud)

This chapter covers the fairly simple process of installing the OEM adapter. However, once installed, the OEM adapter is only a small part of gathering Oracle database inventory. Therefore the remainder of this chapter assumes a fairly new implementation of IT Asset Management, and introduces the other kinds of set up necessary for a working system of Oracle introspection. Some of the latter may already be in place within your enterprise.

## Installing the OEM adapter

IT Asset Management (Cloud)

The OEM adapter is normally installed on an inventory beacon that has high-speed network access to the Oracle server to which the adapter must connect. It is possible to install multiple instances of the OEM adapter on the same computer, each configured to access a different OEM repository. Each instance of the OEM adapter can access exactly one OEM repository.



---

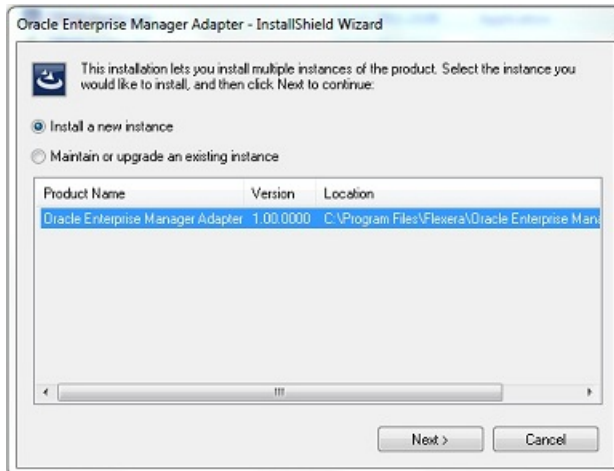
### **To install the OEM adapter:**

1. From the unzipped Adapter Tools archive, navigate into the Oracle Enterprise Manager Adapter folder.

For details about downloading the archive, see [Download Adapter Tools Archive](#).

2. In Windows Explorer, execute the OEMAdapter.exe installer from that folder.

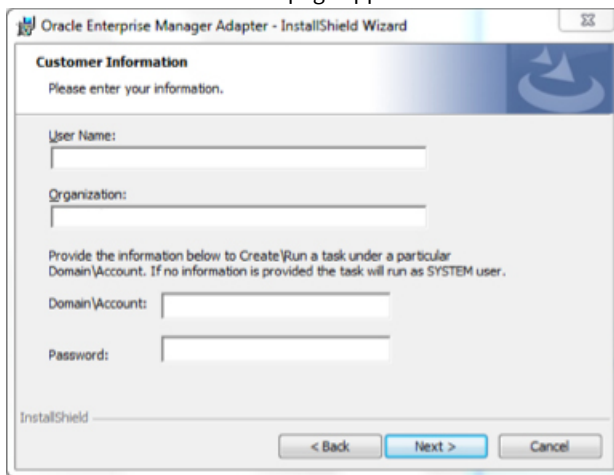
If this is the first installation of the OEM adapter on this computer, the welcome page appears, and you can click **Next**. If another instance of the OEM adapter is already installed on this computer, the following screen appears.



3. If this page appears, choose either of the following:

- a. For an additional installation, click **Install a new instance**, and click **Next**. Follow the remaining instructions below.
- b. To change one of the instances previously installed, select the instance from the list on this page, click **Maintain or upgrade an existing instance**, and click **Next**.

The **Customer Information** page appears.



4. Identify the license owner, and the enterprise name, in the **User Name** and **Organization** fields respectively.
5. Provide credentials for the Windows scheduled task that triggers operation of the OEM adapter.



**Tip:** This account need not be the same as the one to access the OEM repository (identified in a later page).

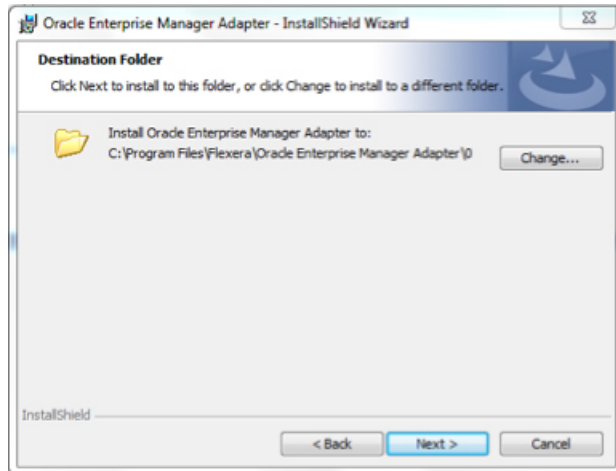
Leave the fields blank to run the scheduled task as the local SYSTEM account on this computer.



**Note:** To change timing or frequency of the scheduled task, use the Microsoft Windows facilities as usual. They are not configurable through the installer.

6. Click **Next >**.

The **Destination Folder** page appears.



7. Optionally, click **Change...** to modify the supplied installation location.

---

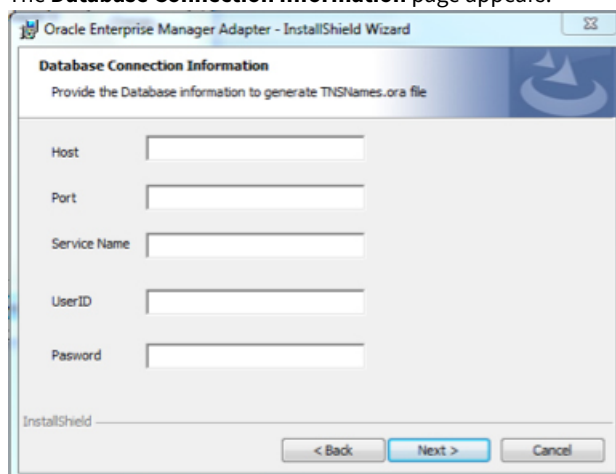
**Note:** If you are installing multiple instances of the OEM adapter on this computer, you must modify the path so that each is installed in a separate folder. The InstallShield wizard offers the default folder 0 for the first instance. Increment this value to give a unique folder for each instance.

---

**Important:** By default the OEM adapter saves the .ora file in the directory where it is executing (that is, the one you specify here). This is useful as a holding bay where the files may be manually inspected for initial testing/verification; but no automated processing of the .ora files occurs from this location. For automated operation after initial testing, you must reconfigure the file path and name as described in [Configure Data Staging](#).

8. When satisfied with the location, click **Next >**.

The **Database Connection Information** page appears.



9. If necessary, confer with your Oracle DBA to complete details about the Oracle Database from which this instance of the OEM adapter collects inventory:
  - a. In the **Host** field, identify the server where Oracle Enterprise Manager is installed.
  - b. The **Port** is used by the OEM adapter to access Oracle Enterprise Manager.

- c. **Service Name** identifies the Oracle service (for this database instance) that was defined when Oracle was installed.
- d. **UserID** is the account name (with its related **Password**) accepted by Oracle Enterprise Manager for login by the OEM adapter.

Suggestion: FNMS-OEMadapter.

When satisfied, click **Next >**, and the **Email Configuration** page appears.

10. To receive email alerts when the OEM adapter encounters any errors:

- a. Enter your email address as the **To Mail** value.
- b. In **From Mail**, enter the email address from which the error alerts should come.
- c. In the **SMTP** field, enter the fully qualified domain name (or IP address) of the email server.



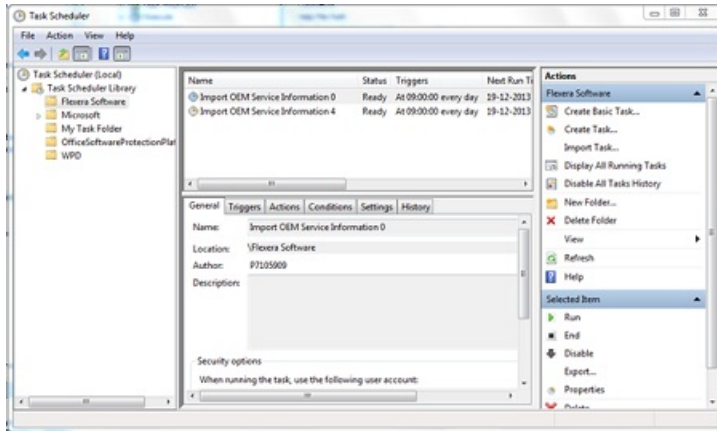
**Tip:** Ensure that the **To Mail** and **From Mail** addresses are both recognized by this email server.

- d. Click **Next >**.

The **Ready to Install the Program** page appears.

11. When satisfied, click **Install**.

The installer writes the OEM adapter executable (OEMAdapter.exe) into your chosen folder, and records your other settings in a configuration file (OEMAdapter.exe.config) saved in the same folder. It also creates a scheduled task to run the OEM adapter. To allow for multiple instances of the OEM adapter on the same computer, the scheduled tasks are named based on the numbering of the installed instances. The default scheduled task for the initial installation on a computer looks like this:



You may update the scheduled task(s) as usual through the Microsoft interface. Before using the OEM adapter in production, you must also do all of the following:

- Confirm that the account used to access Oracle Enterprise Manager (suggested name FNMS-OEMadapter) has adequate permissions to read from the OEM repository (see [Grant Permissions to Account](#))
- Modify the location where the .ora file is saved for production use (see [Configure Data Staging](#))
- Ensure that the appropriate subnets containing target Oracle servers are identified and assigned to the inventory beacon where the TSNNames .ora file is saved (see [Assign Beacon to Subnet](#))
- Configure the collection of Oracle inventory in the web interface for IT Asset Management (see [Configure Collection of Oracle Inventory](#))
- Set up accounts on each Oracle server with adequate permissions to gather inventory, with target machines being based on the contents of the .ora file created by the OEM adapter (see [Inventory-Gathering Accounts on Oracle Servers](#))
- Register the same account(s) in the Password Manager for each relevant inventory beacon (see [Save Inventory Account in Password Manager](#)).

## Grant Permissions to Account

IT Asset Management (Cloud)

The account used to access Oracle (suggested name FNMS-OEMadapter) must have adequate permissions to tables in the OEM repository. If you are setting up multiple instances of the OEM adapter, repeat the process for each adapter (you may be using a common account name for all of them to access each distinct OEM repository, or providing each with a unique account name). This task is normally completed by an Oracle DBA.



**To provide appropriate permissions and validate operation:**

1. To grant adequate read permission, do *either* of the following:
  - a. Make the account (suggested name FNMS-OEMadapter) a member of the EM\_ALLVIEWER role.
  - b. Give the account read permissions on the following:
    - DBA\_TABLES

- MGMT\_TARGETS
- MGMT\_TARGETS\_LOAD\_TIMES
- MGMT\_TARGET\_TYPES
- MGMT\_TARGET\_PROPERTIES

2. When permissions have been set up, test as follows:

- Log in using the account name (suggestion: FNMS-OEMadapter) and password registered during the installation process.
- Execute the following query against the OEM repository:

```
SELECT * from mgmt$Target
```

Note the number of rows returned.

- Log out, and log in using a DBroot account (or other account known to have full permissions); and repeat the same query.

The same number of rows should be returned by the root account with full permissions, and by the account for the OEM adapter with more limited and specific permissions.

3. Repeat the process and testing for any further instances of the OEM adapter accessing other distinct instances of OEM repository.

## Other Setup Activities

### IT Asset Management (Cloud)

The OEM adapter is relatively straight-forward, and, as triggered by the Windows scheduled task set up during installation, regularly collects data from Oracle Enterprise Manager and saves the data as a `TNSNames . ora` file in a special location on the inventory beacon.

However, by itself, when the goal is to gather inventory from Oracle Database servers, the `TNSNames . ora` file is just the first step. Many other 'cogs in the machine' need to do their part for the `. ora` file to be beneficial:

- You need a defined set of sites and subnets, most easily obtained by Active Directory import through an inventory beacon.
- The subnet(s) containing your Oracle servers of interest must be assigned to the inventory beacon that collects the `TNSNames . ora` file, and which subsequently collects the Oracle inventory (see [Assign Beacon to Subnet](#)).
- An inventory rule must be defined that associates the target subnet with an action including inventory collection using the `TNSNames . ora` file, and schedules its execution (see [Configure Collection of Oracle Inventory](#)).
- Quite separately from the account that queries the OEM repository to create the `TNSNames . ora` file, one or more separate accounts must be defined that connect to the Oracle Database servers and collect the inventory data. There are two parts to configuring these accounts:
  - The accounts must be created and given adequate permissions on each Oracle Database server (there is a utility available to assist with this, as described in [Inventory-Gathering Accounts on Oracle Servers](#))



- The same accounts must be registered in the Password Store on the inventory beacon (see [Save Inventory Account in Password Manager](#)).

Strictly speaking, none of these are part of the OEM adapter; but all are integral to the process of Oracle introspection, and therefore are at least summarized in the following sections.

## Inventory-Gathering Accounts on Oracle Servers

IT Asset Management (Cloud)

The inventory beacon collects inventory from Oracle servers using accounts that must be registered at both ends.



### **To set up inventory-gathering accounts on an Oracle server:**

1. Open the knowledge base article:
2. Scroll to the bottom of the article, and click the link `CreateOracleAuditUserQ200934.sql`.

This downloads a SQL script that an Oracle DBA can review. (Ignore the text content which is for earlier products.)

3. In a flat text editor, modify lines 8 and 9 of the script, replacing the default user name and password with credentials of your choosing; and save the modified script.

To minimize configuration and maintenance effort, the same credentials can be implemented on each Oracle Database server. Keep a note of these credentials, as you must also record them in the Password Manager on the inventory beacon.

4. An Oracle DBA can run the script on each target Oracle Database server.

This (re)creates the user name on each run, and grants read access to a numbers of tables and views (shown below) that are required for Oracle Database introspection.

The downloaded SQL script gives the account read-only access to the following tables and views:

- `applsyst.fnd_app_servers`
- `applsyst.fnd_nodes`
- `applsyst.fnd_product_installations`
- `applsyst.fnd_application_tl`
- `applsyst.fnd_user`
- `applsyst.fnd_responsibility`
- `apps.fnd_user_resp_groups`
- `SYS.DBA_USERS`
- `SYS.V_$PARAMETER`
- `SYS.V_$INSTANCE`
- `SYS.V_$DATABASE`

- SYS.V\_\$OPTION
- SYS.DBA\_FEATURE\_USAGE\_STATISTICS
- SYS.DBA\_ENCRYPTED\_COLUMNS
- SYS.DBA\_TABLESPACES
- ODM.ODM\_MINING\_MODEL
- ODM.ODM\_RECORD
- DMSYS.DM\$OBJECT
- DMSYS.DM\$MODEL
- DMSYS.DM\$P\_MODEL
- DVSYS.DBA\_DV\_REALM
- LBACSYS.LBAC\$POLT
- OLAPSYS.DBA\$OLAP\_CUBES
- SYS.DBA\_AWS
- SYS.DBA\_SEGMENTS
- SYS.DBA\_CUBES
- SYS.GV\_\$INSTANCE
- SYS.GV\_\$PARAMETER
- MDSYS.ALL\_SDO\_GEOM\_METADATA
- SYS.V\_\$SESSION
- SYSMAN.MGMT\_LICENSE\_DEFINITIONS
- SYSMAN.MGMT\_ADMIN\_LICENSES
- SYSMAN.MGMT\_LICENSES
- SYS.DUAL
- SYSMAN.MGMT\_LICENSE\_CONFIRMATION
- SYSMAN.MGMT\_TARGETS
- SYSMAN.MGMT\_\$TARGET
- SYSMAN.MGMT\_VERSIONS
- SYSMAN.MGMT\_INV\_COMPONENT
- SYSMAN.MGMT\_FU\_REGISTRATIONS

- SYSMAN.MGMT\_FU\_STATISTICS
- SYSMAN.MGMT\_FU\_LICENSE\_MAP
- SYS.DBA\_REGISTRY
- SYS.V\_\$LICENSE
- SYS.DBA\_TABLES
- CONTENT.ODM\_DOCUMENT
- SYS.V\_\$VERSION
- SYS.USER\_ROLE\_PRIVS
- SYS.USER\_SYS\_PRIVS
- SYS.ROLE\_SYS\_PRIVS
- MDSYS.SDO\_GEOM\_METADATA\_TABLE
- SYS.DBA\_INDEXES
- SYS.DBA\_LOBS
- SYS.DBA\_OBJECTS
- SYS.DBA\_RECYCLEBIN
- SYS.DBA\_MINING\_MODELS
- SYS.REGISTRY\$HISTORY
- SYS.DBA\_TAB\_PARTITIONS
- SYS.DBA\_TAB\_SUBPARTITIONS
- SYS.DBA\_LOB\_PARTITIONS
- SYS.DBA\_LOB\_SUBPARTITIONS
- SYS.V\_\$ARCHIVE\_DEST\_STATUS
- SYS.DBA\_SQL\_PROFILES
- SYS.DBA\_ADVISOR\_TASKS
- SYS.DBA\_SQLSET
- SYS.DBA\_SQLSET\_REFERENCES
- SYS.DBA\_FLASHBACK\_ARCHIVE
- SYS.DBA\_FLASHBACK\_ARCHIVE\_TS
- SYS.DBA\_FLASHBACK\_ARCHIVE\_TABLES

- SYS.V\_\$BLOCK\_CHANGE\_TRACKING
- SYS.V\_\$CONTAINERS

## Save Inventory Account in Password Manager

IT Asset Management (Cloud)

The accounts that collect Oracle inventory must exist in the Password Manager.

This process must be completed on the inventory beacon.



**To register an account in the Password Manager on an inventory beacon:**

1. Log into the inventory beacon, using an account that has administrator privileges.
2. Start the FlexNet Beacon software from the Windows Start menu.
3. In the navigation bar on the left, select the **Password management** tab, and click **Launch Password Manager**.

The separate interface for the Password Manager opens.

4. In the **Current credentials** group, click **New**.

The controls in the **Editor** group are activated. When you are updating an existing entry, the controls are populated with the currently saved values.

5. Provide a **Logical name** to identify this set of credentials.

Logical naming allows you to have one account name, but with different passwords on different servers. For more, see the online help.

6. For **Account type**, choose **Account on Oracle database**.

7. Complete the **User** (account name) and **Password**.

8. Click **View/Edit...**

The **Password Store: Password Filter** dialog displays.

9. Use the **Oracle service names** filter to create a comma-separated list of the Oracle services for which this account/password pair should be used.

The services names are visible in your `.ora` file. Filtering the account/password pair to apply only to specified Oracle services provides maximum efficiency for login and introspection.

10. Click **Apply** to close the dialog, and continue to save the entry in the Password Manager.

Repeat the process as required until all your Oracle machines are covered by one or more entries in the Password Manager. Thereafter you may exit the FlexNet Beacon interface.

## Assign Beacon to Subnet

IT Asset Management (Cloud)

Before rules can take effect, inventory beacons must know their subnets.

It is a requirement for an operational system that your subnets are assigned to appropriate inventory beacons. This summary covers only making adjustment for the collection of Oracle inventory.



#### To assign subsets to an inventory beacon:

1. Go to the **All Subnets** page (**Inventory > Network Discovery > All Subnets**).

This page is populated with the sites and subnets in your enterprise after you have completed an import from Active Directory (for details, consult *IT Asset Management Help > Inventory Beacons > Active Directory Page > Importing from Active Directory*).

2. Expand the appropriate site(s), using the + expander icon, until you can see the subnet that includes your Oracle Database servers.



**Tip:** If the subnet does not yet appear in the listing, you can add its details manually. In the row for the appropriate site, click the + sign on the right-hand end, and enter the subnet IP address in the new row that appears.

3. If the **Beacon name** column shows *Unspecified*, click the editing (pencil) icon at the right-hand end of this row.

The row becomes editable. Beware of accidentally over-writing the IP address, which initially has focus.

4. In the **Beacon name** column, use the drop-down list to choose the appropriate inventory beacon from the list of those registered so far; and click the blue disk icon on the right to save your change.

Your chosen inventory beacon is now authorized to work in the subnet that contains your Oracle Database servers. Now continue and create a rule that dictates what should happen within that subnet.

## Configure Collection of Oracle Inventory

IT Asset Management (Cloud)

Rules must be established that allow Oracle inventory collection, including the use of .ora files.

Inventory rules are created on the Discovery and Inventory Rules page of the web interface for IT Asset Management. Each rule has three parts:

- A *target* that identifies the machines on which the rule is to be exercised
- *Actions* that are to be performed on those target machines
- The *schedule* on which the rule is to be applied.

When an action includes permission to use a .ora file, the relevant inventory beacon uses the locally-available .ora file to 'filter' the related target and identify the Oracle Database servers from which inventory should be collected. For example, if you target an IP range that covers your server room, and include the action setting to apply the .ora file to this range, then Oracle inventory is collected only from the matching machines listed in the .ora file.

Once a discovery and inventory collection has been completed, the individual Oracle Database servers are identified in the list of discovered devices. Should you wish to change to tighter targeting rules, you can use this information to redefine the target until (if you wish) it identifies exactly the Oracle Database servers and no others. Keep in mind that if you use this approach, you will need to adjust the target each time you vary the list of your Oracle Database servers.

To reduce this manual maintenance, keep a target that specifies an appropriate IP range (or ranges), and applies the .ora file to this to identify the individual Oracle Database servers from which to collect inventory.



**To configure Oracle inventory collection:**

1. Go to the **Discovery and Inventory Rules** page (**Data Collection > IT Assets Inventory Tasks > Discovery and Inventory Rules**).
2. If you do not already have a target to reach the Oracle servers from which you want to gather inventory:
  - a. On the left side, select the **Targets** tab.

---

**Fastpath:** In the hints area across the top of the page, click **Create targets**. These hints can guide you through the process.

- b. On the right side, click **Create a target**.
- c. The page appearance changes, allowing you to define a target.
- d. Complete the details requested, with particular attention on **Define machines to target**.

Notice that:

- After you complete each definition for this control, a + icon is displayed that allows you to add more to your definition of target machines. Use these lines to define a target sufficiently broad to capture your Oracle Database servers.
  - If you do not want the FlexNet Inventory Agent to be installed on these Oracle servers, be sure to select **Do not allow these targets to be adopted**.
  - You should likely also select **Do not allow application usage tracking on these targets**, since you may not want to collect a large quantity of file evidence from these servers.
- e. Click **Save**.
3. If you already have an action for Oracle discovery, you can check its settings; or create a new one:
    - a. Click the **Actions** tab (or in the hints section, click **Create actions**).
    - b. If you wish to collect hardware inventory for these servers (perhaps to assist with licensing calculations for your Oracle Database license), expand the General accordion and select **Gather hardware and software inventory**.




---

**Tip:** You may want to clear the check box for **Discover devices**, if you are limiting this action specifically to inventory collection for known Oracle Database servers identified in your .ora file.

- c. Expand the **Oracle database** accordion.
- d. If you are confident that every Oracle Database is identified in your .ora file, you may clear the check box for **Discover Oracle databases**. Alternatively, if you may have rogue servers, leave this check box selected, and the network within your declared target will be probed for other Oracle servers.
- e. Select the check box for **Also gather Oracle database inventory**.

Additional controls, if not already visible, are exposed.

- f. Ensure that the **Port scan** check box is selected, and if necessary use the + icon to add additional ports to the list until every port listed in your .ora file is included.
- g. Ensure that the **SNMP scan** check box is selected.
- h. It is critically important that you select the **TNS names file** check box.

This setting authorizes the relevant inventory beacons to apply any .ora files found in the 'magic path' to the target used for this action (in the rule soon to be completed). This is the mechanism that most efficiently restricts probing to the relevant servers.

- i. Adjust other settings in other parts of the accordion to suit your environment, and click **Save**.

The action is stored, ready for inclusion in a rule.

**4. Select the **Rules** tab (or in the hints area, click the third **Create rules** step).**

- a. Do either of the following:

- If you have an existing rule to review or modify, click the edit (pencil) icon on its right-hand end.
- Click **Create a rule** (upper right) to define a new rule (as described in the following steps).

A rule builder work area appears above the list of existing rules.

- b. Return to the **Actions** tab (for example, using the link in the rule builder), and on the row for your edited Oracle action, click **Add to rule builder**.

The name of your action appears in the rule builder.

- c. Return to the **Targets** tab (for example, using the link in the rule builder), and on the row for your edited target, click **Add to rule builder**.

**5. On the right side of the rule builder, click **Schedule**.**

The rule builder changes to display controls for scheduling:


- a. Choose a value from the **Frequency** drop-down list.

This control sets the style of scheduling. For example, the **Daily** option lets you make further choices about a pattern of days, and does not enforce inventory collection every day.

Option	Notes
Once	A single-shot trigger for inventory collection the next time the declared time window occurs (you cannot nominate a future date). For example, if it is 4pm when you set a Once schedule for 8am, commencing within 4 hours, inventory collection normally occurs next morning. Compare with As soon as possible. Keep in mind the propagation delays for your changed instructions, as described there.

Option	Notes
Daily	<p>An additional drop-down list, <b>Every</b>, appears so that you can choose the pattern of days you want, extending the time intervals between inventory collection in multiples of 24 hours. For example, you may wish to trigger inventory collection every second day. The starting time is within the <b>Commence within</b> time window following the next occurrence of your <b>Start time</b> setting after the changed schedule is distributed (and received by each installed FlexNet Inventory Agent). For example:</p> <ul style="list-style-type: none"> <li>• <b>Start time</b> is set to 8:00 am.</li> <li>• <b>Commence within</b> is set to 2 hours.</li> <li>• The schedule is saved at noon today, and by default the inventory beacon policy is distributed every 15 minutes.</li> <li>• By default, each FlexNet Inventory Agent chooses a random time within a one-hour window from 5am local time to request any policy (and schedule) changes. Most devices therefore start using the changed schedule tomorrow morning.</li> </ul> <p>In short, the setting applies individually to each installed FlexNet Inventory Agent, and does not synchronize the activities of all agents across your enterprise to identical days.</p>
Weekly	<p>New check boxes appear that allow you to choose specific days within the weekly cycle when inventory should be collected. For example, you may want collection on Sunday and Wednesday every week. Select (check) the boxes for the days you prefer.</p>
Monthly	<p>New controls appear that allow you to specify a pattern within the month:</p> <ul style="list-style-type: none"> <li>• Choose the option for <b>On day</b> to nominate a particular day within the month (such as the third Saturday). Make your choices from the two drop-down lists adjacent. Notice that the option <b>Last</b> chooses the fifth occurrence in months long enough to have one, and otherwise takes the fourth occurrence.</li> <li>• Choose the option for <b>On date</b> to nominate the calendar date within the month.</li> </ul>



Option	Notes
As soon as possible	This is a single shot trigger which causes each FlexNet Inventory Agent to randomize an inventory collection within the time window starting when it receives this setting and lasting for the interval you specify in the <b>Commence within</b> controls.
 <b>Tip:</b> Any change you save to these settings is first collected by the inventory beacons on the schedule specified by the <b>Beacon settings</b> (further down this web page), by default checked every 15 minutes. Each inventory beacon then prepares new instructions for any installed FlexNet Inventory Agents that request an update to device policy (the operational FlexNet Inventory Agent checks for updated policy once per day, randomized across a one-hour window in the early morning; but newly installed agents seeking their first policy make a request once every 12 hours on UNIX-like platforms, or once every system restart on Windows). Because of these propagation periods, <i>As soon as possible</i> typically means starting early tomorrow morning.	

6. In the rule builder, click **Save as**, give the rule a name you will recognize later in lists, and click **Save**.

If you accepted the default **Enabled** setting, the rule is ready to run on the schedule you have established. (Remember to allow around 15 minutes for the new rule to be distributed to your inventory beacons.) When the rule is executed on an inventory beacon, if a `.ora` file exists in the 'magic path', the systems that lie both within the inventory beacon's assigned subnet(s) and within the `.ora` file are targeted for Oracle inventory collection.

## 3

# Modifying the Adapter

IT Asset Management (Cloud)

This chapter covers changes you can make to the operations of the OEM adapter by modifying its configuration file. Read the first topic for general guidance on the editing process, and choose the detailed subtopics based on what changes you need.

## Reconfiguring the OEM Adapter

IT Asset Management (Cloud)

During installation, you recorded your preferred settings for the OEM adapter, and no further work is required in the installation process. You can however choose to modify the configuration of an instance of the OEM adapter at a later stage.



---

**To reconfigure the OEM adapter:**

1. In Windows Explorer, navigate to the correct folder for the appropriate instance of the OEM adapter.

Keep in mind that there may be several instances installed on a computer, accessing distinct instances of Oracle Enterprise Manager.

2. In your preferred plain text (or XML) editor, open the OEM adapter's configuration file, called `OEMAdapter.exe.config`.

It is strongly recommended that you make a backup copy of the original configuration file, so that you can easily revert your changes if there are problems.

3. Use the following subtopics to guide your changes to the configuration file.
4. When you have finished, save the updated file.
5. Either wait until the next scheduled run of the OEM adapter, or use the Windows scheduled task interface to trigger an immediate test run to confirm that your changes work as expected.

# Updating Connection Details

IT Asset Management (Cloud)

You can change all details and credentials for connection to the Oracle Enterprise Manager instance.

When the OEM adapter is first run, it encrypts the connection details used to access the Oracle Enterprise Manager implementation. Therefore, the process to edit the connection details is different, based on whether the OEM adapter has been run since the connection details were last recorded. You can tell whether this is so by looking in the configuration file.

The details are all stored in the <connectionStrings> element of the configuration file.

1. If the OEM adapter has been run since the details were recorded in the configuration file, the encrypted <connectionStrings> element appears similar to this:

```
<connectionStrings
configProtectionProvider="DataProtectionConfigurationProvider">
  <EncryptedData>
    <CipherData>
      <CipherValue>AQAAANCmnd8BFdERjHoAwE/C1+sBAAAAeNhcgMAVK0uVGNTqOg/
WbQQAAAAACAA

AAAAADZgAAwAAAAABAAACT09kpn6BptpLvSxExg1UBAAAAAASAAACgAAAAEAAAAELCyiwz5A

XZw9xZXfEPiAAAAgAAPJQYe+G9AfScFMJTYgA0NDbAgZdRA9nB91DN42A1xjeCskUs9+KNjV
U1PSFRV4ujta40evf3IOZy5odyHsIrJRCK00GdhDb1wh4ISEkpJk/
QDna6LeCbbtXXsQK2Lo
  AHQc/plz77UkQZxnXkL5E1PIGH16AoJEXT2F5NGjElJX6GXbUXQDkNnDfi2o6XI/
CdbX8gCu
  MonY1cTLyGe6+AQPPdGcY3rA02ZF0s7/Zb0cK0w7IoZdB6H8OIvrClSsqNkVBd3YfLhP/
KOr
  kQFP8orqj54BJW74E1v3VUZnte1ESgLA5MYb/
F9Ah3M5xi2Q6ITX0QmVRGESrividqr6nyGz
  5APx2yVBuEcoVhpOMYURbEbBSW+6/
aydg8nY1DrcMzkPlXiZ0CQs4yZYHSwt3+bFEN30xh6X

KFH7Tpc8e5y9Tq1BhWk+Kw6AFfZhcdeWApJ4ZkGAR4ixE6gNqWXnomk2puKNFImhJfqlLIuQ

x9T00Uu2bjJ9Y+dU1rcuov12hQX0Ch9nqOdmeIQHPXgw8//eHY0zwy2TgMcq2M2LxXRbQqCT

eWt1P1ucJVyct9gnJlk2L3FSBngMu1C9mncR7MsGDBWoQMxNwC5+Y6P1GT45sPOedBQvIQIT

j7MCuzfgDD1R9c7w1gejTGmr13Kmf33tGPMRYPV7CPNET3DUV9AGQUAAAAaIEkjqfExrbei
  YJvG2usqY03Nk=</CipherValue>
    </CipherData>
  </EncryptedData>
</connectionStrings>
```

In this case, delete the entire element and replace it with a plain text version as shown below.

2. If the OEM adapter has not been run since the configuration file was changed, it has a plain text form similar to this (line wrapping has been added here for legibility). Replace the italicized placeholders with your own values, as described below. Keep the `connectionString` attribute all on one line.

```
<connectionStrings>
  <add name="OEMConn"
    connectionString="Data Source=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)
      (HOST=HostIPAddress)
      (PORT=PortNumber))
      (CONNECT_DATA=(SERVICE_NAME=ServiceName)));
    User Id=AccountName;Password=UnencryptedPassword;"></add>
</connectionStrings>
```

where the following details should be confirmed with your Oracle administrator:

- *HostIPAddress* is the IP address of the Oracle server hosting Oracle Enterprise Manager
  - *PortNumber* is the port that the OEM adapter should use to query Oracle Enterprise Manager
  - *ServiceName* is the service name established when Oracle was installed on the Oracle server
  - *AccountName* is the account (user name) that the Oracle administrator has established for accessing Oracle Enterprise Manager
  - *UnencryptedPassword* is a plain text rendition of the password for the same account (remembering that all the connection details are encrypted together as soon as the OEM adapter runs next).
3. Save the amended configuration file.
- The **Password Store: Password Filter** dialog displays.
4. Use the Windows schedule task interface to run a test of the OEM adapter to confirm that the connections details were correctly entered.

## Configure Data Staging

IT Asset Management (Cloud)

Modify the name and location of the file of Oracle connection information.

The OEM adapter collects information about connections to Oracle systems from which you need to gather inventory information. This file is in a standard Oracle format, used for their `TNSNames . ora` file. It is saved by default where the OEM adapter is executing; but the default location for the OEM adapter does not support automated processing of the `TNSNames . ora` file when the inventory beacon applies rules for gathering Oracle inventory. For the OEM adapter to function seamlessly, you must customize the location as you edit the `OEMAdapter . exe . config` file.

Another reason to customize the file name is if you have multiple instances of the OEM adapter running from the same computer (or if anyone manually adds `TNSNames . ora` files to the processing directory). Each instance must write to a unique file name, so that one output does not over-write the other.

These settings live in an add element with the key attribute of `ConnectionInfoFile`.

**To configure data staging:**

1. In the OEMAdapter.exe.config file, locate the appropriate element. Its default values are similar to the following:

```
<add key="ConnectionInfoFile"
      value="C:\Program Files\Flexera\Oracle Enterprise Manager Adapter\
      TNSNames.ora"></add>
```

This reflects the default location of the OEM adapter, and must be modified.

2. Replace the value string with the new file path and name.



**Important:** The file name **must** use the extension .ora.

You may use a mapped drive on the local computer to specify a network path. This example shows the file in the recommended 'magic path' on the inventory beacon where the .ora file is automatically processed to 'filter' the target supplied from the central application server. Also notice the customized file name to avoid naïve overwriting of the TNSNames.ora file name by other copies saved here:

```
<add key="ConnectionInfoFile"
      value="C:\ProgramData\Flexera Software\Repository\TNSNames\
      TNSNames-01.ora"></add>
```

3. Save the amended configuration file.

The **Password Store: Password Filter** dialog displays.

4. Use the Windows schedule task interface to run a test of the OEM adapter to confirm that the staging file is saved according to your revised specifications. Check the location (recommended: C:\ProgramData\Flexera Software\Repository\TNSNames\) for the presence of a saved file immediately after the test run (if you wait too long, the resulting file may be automatically uploaded and removed from this staging location).

## Managing Email Alerts

IT Asset Management (Cloud)

You can turn the alerts off, changes their addresses, or switch email servers.

The OEM adapter can send email alerts any time that it encounters an error. Manage the emails with the following XML elements in the OEMAdapter.exe.config file.

1. To stop email alerts entirely comment out the *reference* to the SmtpAppender like this:

```
<appender-ref ref="RollingFileAppender"></appender-ref>
<!-- <appender-ref ref="SmtpAppender"></appender-ref> -->
```

(The character sequence <!-- starts an XML comment, and the sequence --> closes the comment.)



**Tip:** Using this technique is preferable to deleting this line entirely, as this can be easily reversed if you decide to reinstate email alerts in future. Notice that, depending on the configuration you saved, there may be other

---

*elements between the <appender-ref> tags.*

2. To reconfigure the email alerts, locate the <appender> element, and modify three of its child elements by changing the example values shown in *italics* here:

```
<appender name="SmtpAppender" type="log4net.Appender.SmtpAppender">
  <to value="toaddress@somedomain.com"></to>
  <from value="fromaddress@somedomain.com"></from>
  <subject value="OEM Adapter Error"></subject>
  <smtpHost value="smtp.somedomain.com"></smtpHost>
</appender>
```

3. Save the amended configuration file.

The **Password Store: Password Filter** dialog displays.

## Configure Logging

IT Asset Management (Cloud)

Change the path and file name for logging by the OEM adapter.

The log file name and location cannot be set during installation of the OEM adapter, but after installation, they can be modified as follows:



### **To configure logging:**

1. In the OEMAdapter.exe.config file, locate the <appender> element called RollingFileAppender.
2. Edit the child <file> element by replacing the drive, path and file name shown as placeholders here:

```
<appender name="RollingFileAppender"
  type="log4net.Appender.RollingFileAppender">
  <file value="drive:\path\path\logfilename.txt"></file>
</appender?>
```

3. Save the amended configuration file.

The **Password Store: Password Filter** dialog displays.

After the next run of the OEM adapter, inspect the log in your new location.

# XVI

## Salesforce Adapter

### IT Asset Management (Cloud)

The benefits of Salesforce subscriptions come at a significant cost for large organizations and licensing is complex. IT Asset Management provides you with ways to better stay in charge of your Salesforce licensing and consumption. IT Asset Management provides an inventory adapter so that you can import Salesforce license allocations and usage data and then leverage IT Asset Management's tools to help you manage Salesforce licensing and consumption costs. With IT Asset Management's integration with Salesforce, you can easily see unused Salesforce subscriptions, thereby helping you strategize how to reduce costs at subscription renewal. IT Asset Management:

- Imports Salesforce license information and creates one SaaS User license for each instance of Salesforce that is connected. In IT Asset Management, under the SaaS User license, a series of applications represents the Salesforce users in your organization.
- Shows you consumption information in the **Consumption** tab of the license properties for each user in your organization that is accessing Salesforce.
- Shows unused Salesforce subscriptions. To determine unused Salesforce subscriptions, you can specify a number of days in a **SaaS Usage Summary** report and drill through to users to view which users have and have not accessed Salesforce over that time period.
- Supports multiple Salesforce instances.
- Provides a list of accessing users for each Salesforce instance.



---

**Note:** You must be connected to Salesforce in order for IT Asset Management to provide Salesforce information.



---

**Note:** For managing a your subscription licenses for Salesforce, you may prefer to use the Flexera One SaaS Management connector, which not only integrates information for Salesforce, but simultaneously imports information for all your SaaS-based products you choose to manage there. If you choose the Flexera One SaaS Management connector, you should not use the Salesforce connector described in this part.

## 1

# Salesforce License Considerations

## IT Asset Management (Cloud)

You can use IT Asset Management to review your Salesforce subscriptions. IT Asset Management provides a way for you to:

- Review Salesforce license consumptions to help determine how many Salesforce licenses you need to purchase, which in turn helps you negotiate license agreements.
- Ensure that business units are not under-subscribed.



**Note:** The first time you import your organization's license information from Salesforce, IT Asset Management creates one SaaS User license for each instance of Salesforce that is connected. If license information for a new user is imported after the initial import, that information is imported to the original SaaS User license in IT Asset Management. Each user imported from Salesforce is mapped to the appropriate accessing user record. These users consume against the Salesforce multi-product license created for the Salesforce tenant.



## 2

# Viewing Salesforce License Information with IT Asset Management

IT Asset Management (Cloud)

The following provides high-level steps for viewing Salesforce license information using IT Asset Management.



**To view Salesforce license information:**

1. Ensure that you have created and configured a connector for each tenant of the Salesforce Online Service. For more information, see [Managing Connections to Salesforce.com](#).
2. During the next discovery and inventory collection, IT Asset Management imports the following objects:
  - **Users:** Each imported user from Salesforce.com is mapped to the appropriate accessing user record. If an imported user cannot be mapped to any accessing user record, an accessing user record is created with a dummy inventory device assigned to this user.
  - **Licenses:** A SaaS User license is created for each Salesforce tenant (typically one tenant per enterprise, although mergers and acquisitions may mean that your enterprise has multiple Salesforce tenants). Note the following:
    - This license is derived from a template provided in the downloadable libraries, and selected based on the plan identifier.
    - Multiple applications are linked to this license that represent feature and user licenses within Salesforce.
    - The name of the Salesforce tenant is included in the name of each of these license and applications. The name of the connector is included as well. This information helps to distinguish licenses when you have multiple Salesforce tenants.
3. Review the **All IT Asset Users** and **All Licenses** pages to ensure that the required users and licenses have been created.
4. After license reconciliation has been completed, the **Product Summary** page should show the compliance status for Salesforce.

5. Go to the **All Licenses** page (**Licenses > License Management > All Licenses**). Filter the records to view Salesforce licenses. You can check the **Consumed** and **Used** columns to know how many subscriptions are being used.
6. You can also view the SaaS Usage report for insights on Salesforce license consumption.

## 3

# Connecting to the Salesforce Online Service

IT Asset Management (Cloud)

To generate the compliance position for Salesforce licenses, IT Asset Management needs information about subscription and usage of Salesforce licenses. To get the subscription and usage information from Salesforce.com, the FlexNet Beacon should be configured with an inventory PowerShell connection of the **Source Type** of Salesforce for each tenant of Salesforce. (Typically, each enterprise is a single Salesforce tenant; but your corporate history, including mergers and acquisitions, may mean that your enterprise includes multiple Salesforce tenants.)

When created and configured with the Salesforce connection, the inventory beacon imports the licenses, users, and usage information from the Salesforce.com account and uploads it to IT Asset Management according to the defined schedule. For details on how to create a connector to Salesforce, see [Managing Connections to Salesforce.com](#).

## Managing Connections to Salesforce.com

IT Asset Management (Cloud)



**Tip:** The connector to Flexera One SaaS Management supersedes and overrides the connector to Salesforce.com. This means that if you implement the Flexera One SaaS Management connector, any licenses that are created for imports through the Salesforce.com connector are automatically given a status of *Retired*, so that they have no impact on license reconciliation calculations. Instead, new licenses are created from the Flexera One SaaS Management imports, and these are included in license reconciliations as usual.

Use the following procedure to create a connection to Salesforce.com on the FlexNet Beacon (when you are *not* integrating with Flexera One SaaS Management). IT Asset Management provides an inventory adapter to import Salesforce.com license allocations and usage data. Operators can easily see unused Salesforce.com subscriptions which, in turn, allows you to manage costs at subscription renewal. A connection is required for each organization of Salesforce.com. The inventory beacon requires this connection to import all Salesforce.com licenses available to the enterprise, show consumption for each allocated license, show unused subscriptions, and support multiple Salesforce.com instances.

You must make sure that the Salesforce tenant user has the required System Administrator privilege for this operation.

Also make sure that the following prerequisites are met on each inventory beacon that needs to download data from Salesforce (noting that the order of installation of prerequisite software may be significant). These requirements should have been met when the inventory beacon was installed:

- PowerShell 5.1 or later is running on Windows Server 2008 R2 SP1 or later, or Windows 7 SP1 or later; with the PowerShell execution policy set to RemoteSigned



**Tip:** The PowerShell execution policy can be set by running PowerShell with administrator rights and executing the following command:

```
Set-ExecutionPolicy RemoteSigned
```

- A browser is installed with JavaScript enabled on the machine the inventory beacon software is installed on.
- A Javascript-enabled browser installed on the machine the inventory beacon software is installed on
- The inventory beacon server supports Transport Layer Security (TLS) 1.2 in order for the connection to Salesforce to work.



#### To create a connection to Salesforce.com:

1. Login to Salesforce and create a connected app. (A connected app is required for a connection to Salesforce.com on the FlexNet Beacon to work.) For information about how to create a connected app in Salesforce, see [https://help.salesforce.com/articleView?id=connected\\_app\\_create.htm](https://help.salesforce.com/articleView?id=connected_app_create.htm). Note the following:

When creating the connected app, ensure that you select **Enable OAuth Settings**, select the **Enable for Device Flow** check box and select all **Available OAuth Scopes**. The connected app in Salesforce uses standard SAML and OAuth protocols to authenticate, provide single sign-on, and provide tokens for use with Salesforce APIs. In addition to standard OAuth capabilities, connected apps allow Salesforce admins to set various security policies and have explicit control over who can use the corresponding apps. After creating the connected app in Salesforce, you can view the app details in order to copy and paste the **Consumer Key** and **Consumer Secret** values to the corresponding fields in the FlexNet Beacon PowerShell Source Connection dialog in **Step 8**.

2. Login to Salesforce and access the connected app to view details.
3. In the FlexNet Beacon interface, ensure that you have your preferred schedule for imports from Salesforce set on the appropriate inventory beacon:
  - a. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Tip:** Remember that you must run the inventory beacon software with administrator privileges.

- b. From the **Data collection** group in the navigation bar, choose **Scheduling**.
  - c. If there is not already a suitable schedule in the list, click **New...** and complete the details (see the online help for that page for more information). Otherwise, identify the schedule you will use.
4. Select the **Inventory Systems** tab (in the same navigation group).
  5. Choose either of the following:
    - To change the settings for a previously-defined connection, select that connection from the list, and click

**Edit....**

- To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.
6. Complete (or modify) the values for the following required fields:
    - **Connection Name:** Enter a name for the inventory connection. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name.
    - **Source Type:** Select **Salesforce** from this list.
  7. Optionally, if your enterprise uses a proxy server to enable Internet access, complete (or modify) the values in the **Proxy Settings** section of the dialog box in order to configure the proxy server connection.
    - **Use Proxy:** Select this check box if your enterprise uses a proxy server to enable Internet access. Complete the additional fields in the **Proxy Settings** section, as needed. If the **Use Proxy** check box is not selected, the remaining fields in the **Proxy Settings** section are disabled.
    - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`. This field is enabled when the **Use Proxy** check box is selected.
    - **Username** and **Password:** If your enterprise is using an authenticated proxy, specify the user name and password of an account that has credentials to access the proxy server that is specified in the **Proxy Server** field. These fields are enabled when the **Use Proxy** check box is selected.
  8. Complete (or modify) the values in the **Salesforce** section of the dialog box. All of the following fields require a value:



**Note:** If you have multiple organizations to Salesforce (for example, separate organizations for different corporate units or locations), you need to create a separate connector for each organization using its own credentials.

- **Salesforce URL:** Enter the address of the Salesforce URL to be used for generating a new token.
  - **Consumer Key:** Copy this value from the Consumer Key field in the Salesforce connected app.
  - **Consumer Secret:** Copy this value from the Consumer Secret field in the Salesforce connected app. In the Salesforce connected app, click **Click to reveal** to view the value.
  - **Refresh Token:** Click the **Generate** button to generate a refresh token that will be used to authenticate the connection to Salesforce.
9. When you click **Generate** to the right of the **Refresh Token** field, a Web browser is launched in Salesforce.com with an 8-digit code automatically populated in the **Code** field. In the Salesforce.com screen, do the following:
    - a. Click **Connect**. A Salesforce login page displays. If you are already logged into Salesforce, skip to step 8e.
    - b. Enter your Salesforce username. If there are multiple Salesforce accounts, select your username from the **Saved Username** list or click **Log In with a Different Username**. A **Password** field appears.
    - c. In the **Password** field, enter your Salesforce password.

- d. Click **Login**. An **Allow Access** page appears.
- e. Click **Allow** to allow Salesforce to access the FlexNet Beacon to have the refresh token sent back to PowerShell Source Connection dialog. A message appears to notify you that the connection is successful. Click **Continue** or close the browser.



**Note:** The following table provides help with potential issues that you may encounter when attempting to connect to Salesforce.

**Table 12:** Troubleshooting Salesforce Connection Errors

Error	Description
Mandatory parameter(s) missing	One or more of the fields in the FlexNet Beacon PowerShell Source Connection dialog is missing a value. Ensure that all mandatory fields contain values before clicking <b>Generate</b> to the right of the <b>Token</b> field.
The remote name could not be resolved: 'login.salesforce.com'	The most common cause is that there is no Internet connection available on the machine the inventory beacon software is installed on. There may be other reasons such as Salesforce.com being blocked or is currently down.
invalid_client_id	The value that you entered for the <b>Consumer Key</b> field does not match the <b>Consumer Key</b> value from the Salesforce connected app. Copy the correct <b>Consumer Key</b> value from the Salesforce connected app and paste it into <b>Consumer Key</b> field in the FlexNet Beacon PowerShell Source Connection dialog.
Timeout	When the Web browser (that is invoked after clicking <b>Generate</b> to the right of the <b>Token</b> field) is not responding in a timely fashion, a Timeout error occurs. Some of the most common scenarios that will trigger this error are: <ul style="list-style-type: none"> <li>• The Internet browser that was invoked when you clicked <b>Generate</b> was subsequently closed inadvertently or prematurely.</li> <li>• A login error was encountered that may be the result of incorrect login credentials.</li> <li>• You may have waited too long before attempting to login to Salesforce.</li> </ul>
invalid_request	You have clicked <b>Deny</b> instead of clicking <b>Allow</b> in Step 5e to access the FlexNet Beacon to have the refresh token sent back to PowerShell Source Connection dialog.

**10.** In the FlexNet Beacon PowerShell Source Connection dialog, click **Save** to save the connection.

**11.** Select your new connection from the displayed list, and click **Schedule....**

12. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.
13. At the bottom of the **FlexNet Beacon** interface, click **Save**, and if you are done, also click **Exit**.



**Tip:** Consider whether you want to select your connection, and click **Execute Now**, before you exit.

After a successful data import, the users, applications, licenses, and usage data are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** tab of FlexNet Beacon, see *Inventory Systems Tab in the online help*. For scheduling data imports through this connection, see *Scheduling a Connection, also in help*.

IT Asset Management (Cloud)

Current

# XVII

## ServiceNow Integration with IT Asset Management

IT Asset Management (Cloud)



**Note:** The documentation provided here includes updates for Flexera Integration application v5.1.3. If you are on older versions of the Flexera Integration app, visit the **Flexera Integration** page of the [ServiceNow Store](#) where you can not only upgrade to the latest version but also obtain the most current documentation (found within the **Supporting Links and Docs** section of the page).

Using the integration application for ServiceNow, you can exchange a limited set of data between IT Asset Management and ServiceNow to provide a consistent view of your hardware and software estate, and related contracts. The **Flexera Integration** app is a scoped application and is now shared between IT Asset Management, Flexera Data Platform, and IT Visibility, and is available in the ServiceNow store.

The Flexera Integration app:

- Requires the Software Asset Management (SAM) Foundation plugin. This plugin needs to be requested from ServiceNow but does not have a licensing cost associated with it. For information, refer to: [Setting up Software Asset Management Foundation plugin](#).



**Important:** If you have software installation records that were previously discovered before the installation of the SAM Foundation plugin, you must run a migration script after installing the SAM Foundation plugin; otherwise these records will be missing in ServiceNow. This is because installing the SAM Foundation plugin disables the `[cmdb_software_instance]` and `[cmdb_ci_spkg]` tables. The migration script can copy the data from these legacy tables to the new SAM tables. For details about the migration script, refer to [Migrate Software Asset Management Foundation plugin software installations](#).

- Currently has only been confirmed and tested to support a ServiceNow cloud instance.
- Does *not* support ServiceNow on-premises instances on an Oracle database.
- May or may not work with a ServiceNow on-premises instance on a MySQL database. This position may change in the future when Flexera has tested it further.





**Note:** For any single instance of ServiceNow, only one source of Flexera data should be exporting data for consumption by ServiceNow. The Flexera Integration app does not support resolving overlapping data. With a view to the long-term, where you have a choice of all of the above, IT Visibility would be the common recommendation. However, notice that this restriction applies to Flexera tools that are exporting data. The (quite separate) ServiceNow inventory connector (see next section of this manual) does not export data to ServiceNow, being strictly an importer of inventory discovered by ServiceNow. Therefore it is perfectly acceptable to use the same Flexera Integration app on the one instance of ServiceNow to combine integration with (say) IT Visibility (or either of the other data exporters) with the ServiceNow inventory connector.

ServiceNow provides cloud-based IT Service Management, while IT Asset Management is focused on Software Asset Management. To help provide a unified view of your management data, there are two parts to the integration of these systems, each of which is available independently:

- Data on hardware assets, application installations, and contracts can be exported from IT Asset Management and imported into ServiceNow
- Data on assets and contracts can be exported from ServiceNow and imported into IT Asset Management.

**ServiceNow Version Compatibility:** For the latest ServiceNow version compatibility information, refer to the **Flexera Integration** app listing within the [ServiceNow Store](#). In the ServiceNow Store, search for the **Flexera Integration** app listing and select it. Here you will see the versions of ServiceNow that are supported for the current release of the Flexera Integration app. To see earlier versions of the Flexera Integration app, click **Other App Versions** in the **Version** section. A matrix appears that includes a list of the ServiceNow versions supported for each version of the **Flexera Integration** app.

This ServiceNow documentation is divided into the following chapters:

- [Key Concepts](#) — key concepts to help you understand how and what type of data is merged with the integration application, where to view merged data, insight into how source of truth settings structure data into an optimized view with no duplicate or missing records
- [Architecture, Components, and Prerequisites](#) — background information including requirements you need to know to get started with the integration application, as well as contextual information to help you understand the configuration and operation of the integration
- [Installation and Configuration](#) — common tasks to be completed for all implementations, procedures for setting up data flows from IT Asset Management to ServiceNow or from ServiceNow to IT Asset Management
- [Operational Details](#) — steps showing how the export of data from IT Asset Management works (to export hardware inventory, installed applications, and contract data), and steps showing how the export of data from ServiceNow works (to export asset and contract data)
- [Appendices](#) — integration property descriptions, a performance improvement tip to create additional indexes on your ServiceNow instance, procedure to remove a previously-installed legacy integration application, and more.

## 1

# Key Concepts

## IT Asset Management (Cloud)

Before using integration application for ServiceNow, it is helpful to understand some key concepts of the integration that are described in the following sections:

- [Data Types that Can Be Merged](#) — data types that can be exported from IT Asset Management and from ServiceNow
- [How Data Is Merged](#) — introduces the role that transform maps play in merging data between IT Asset Management and ServiceNow
- [Source of Truth](#) — how source of truth settings structure data into an optimized view with no duplicate or missing records
- [Where to View Merged Data](#) — where to go in IT Asset Management and ServiceNow in order to view data that has been merged by the integration application
- [ServiceNow Computer and Application Records](#) — table storage options in ServiceNow (Software Asset Management (SAM) Foundation plugin tables) and a description of how the integration application creates or updates computer records in ServiceNow and sets the CI classes.

## Data Types that Can Be Merged

### IT Asset Management (Cloud)

The following table shows the data types that can be exported from IT Asset Management and data types that can be exported from ServiceNow.

When exporting from:	The following data types can be exported:
IT Asset Management	Hardware inventory, installed applications, and contracts.
ServiceNow	Assets and contracts



**Note:** When exporting from IT Asset Management, the three data types each correspond to a check box in the **ServiceNow export settings** section in the **Integrations** tab on the IT Asset Management Settings **General** page. You

can export any or all of these data types to ServiceNow. When exporting from ServiceNow, assets and contracts can be included in the export, as configured in scheduled jobs that can be accessed from either of the following ServiceNow menus: see **Flexera Integration > Export Configuration > Scheduled Jobs**.

# How Data Is Merged

## IT Asset Management (Cloud)

When data is merged from IT Asset Management to ServiceNow, transform maps are used to map the data. For more information, refer to [Transform Maps for ServiceNow Integration](#).


When data is merged from ServiceNow to IT Asset Management, business adapters are used to map the data. For more information, refer to [Business Adapter Mappings](#).

For both scenarios, an integration user needs to be created in ServiceNow so that both systems can communicate. For more information, refer to [Creating a ServiceNow Integration User](#).


# Source of Truth

## IT Asset Management (Cloud)

When exporting from IT Asset Management for import into ServiceNow, a source of truth needs to be defined between systems in order to address any differences encountered during the merge. A set of integration properties are provided in ServiceNow integration application, see **Flexera Integration > Integration Properties**. The integration properties let you specify whether to accept IT Asset Management as the source of truth for various data types. Choosing **No** for any option toggles that setting to ServiceNow as the source of truth. For example, when merging inventory, if a record is not found with given criterion, a new record will be created if IT Asset Management is source of truth for **Adding Inventories**. For more information about source of truth integration properties, refer to [Integration Properties](#).

 **Note:** For records of hardware inventory and installed applications, IT Asset Management is considered the authoritative source of truth.

When exporting assets and contracts from ServiceNow for import into IT Asset Management, scheduled jobs need to be configured. For more information about scheduled jobs, refer to [Setting Up Data Flows from ServiceNow to IT Asset Management](#). Then, the business adapters are used to import data into IT Asset Management. Assets and contracts are both set as the source of truth by default; however, you can edit the business adapters to change these settings. For more information, refer to [Configuring FlexNet Beacon for Import](#).

 **Note:** For assets and contracts, ServiceNow is considered the authoritative source of truth.

# Where to View Merged Data

## IT Asset Management (Cloud)

The following table shows where to go in IT Asset Management and ServiceNow to view data that is merged by the integration application. This not only helps you understand the integration but also helps you understand where to go in both systems in order to view and validate results.

Data type	IT Asset Management menu location	ServiceNow menu location
Hardware inventory	<b>Inventory &gt; Inventory &gt; All Inventory</b>	<b>Flexera Integration &gt; Imported Records &gt; Computers</b>
Installed applications	<b>Applications</b> tab of inventory device properties.	<b>Software Installations</b> tab (SAM) of a computer record
Contracts	<b>Procurement &gt; Contracts &amp; Payments &gt; All Contracts</b>	<b>Flexera Integration &gt; Imported Records &gt; Contracts</b>

You may also want to compare inventory device properties.

Property	IT Asset Management	ServiceNow
General properties of inventory device such as: device type (Computer, Virtual machine), Manufacturer, Serial number, etc.	<b>Inventory device properties</b> (accessed by selecting a record from <b>All Inventory</b> )	<b>Computer</b> record (accessed by selecting a record from <b>Flexera Integration &gt; Imported Records &gt; Computers</b> )

# ServiceNow Computer and Application Records

IT Asset Management (Cloud)

## Installed Applications

When exporting installed application data from IT Asset Management for import into ServiceNow, you no longer have the option to select which tables store the data. Installed application records are stored into Software Asset Management (SAM) Foundation plugin. Therefore, the integration property **Use CMDB and/or SAM tables for installation** is now disabled and SAM tables are automatically used. For more information, refer to [Integration Properties](#).

## Hardware Inventory

When hardware inventory records are exported from IT Asset Management to ServiceNow, the integration creates or updates computer records in ServiceNow. The integration also sets the correct CI class on these records based on the **Inventory Device Type** and **Operating System**. This enables viewing these records in the respective views within ServiceNow. The following table shows the logic of how the CI class of computer records are classified in ServiceNow and also shows where to view the resulting records in ServiceNow.

**Table 13:** Hardware inventory exported from IT Asset Management and imported to ServiceNow

Inventory device type of Inventory record in IT Asset Management	Operating system of Inventory record in IT Asset Management	CI class of Computer record in ServiceNow	View in ServiceNow
VM Host	contains "Microsoft" or "Windows"	Hyper-V server	<b>Configuration &gt; Hyper-V &gt; Hyper-V Servers</b>
	contains "VMware"	ESX server	<b>Configuration &gt; VMware &gt; ESX Servers</b>
	<i>anything else</i>	Visualization server	no direct link, but you can enter the following into the ServiceNow Filter navigator to access view: <code>cmdb_ci_virtualization_server.list</code>
Computer Virtual Machine Mobile Device	contains "AIX"	AIX Server	<b>Configuration &gt; Servers &gt; AIX</b>
	contains "Solaris" or "Sun OS"	Solaris Server	<b>Configuration &gt; Servers &gt; Solaris</b>
	contains "OS X" and contains "Server"	OS X Server	<b>Configuration &gt; Servers &gt; OS X</b>
	contains "HP-UX"	HPUX Server	<b>Configuration &gt; Servers &gt; HPUX</b>
	contains "Unix"	Unix Server	<b>Configuration &gt; Servers &gt; Unix</b>
	contains "Linux"	Linux Server	<b>Configuration &gt; Servers &gt; Linux</b>
	contains "Microsoft" or "Windows" and contains "Server"	Windows Server	<b>Configuration &gt; Servers &gt; Windows Servers</b>
	<i>anything else</i>	Computer	<b>Configuration &gt; Base Items &gt; Computer</b>

For all the virtual machine records in an export, the import creates or updates Virtual Machine Instance records in ServiceNow and sets the correct CI class on these records based on the virtual machine type. The following table shows the logic of how the CI class of virtual machine records are classified in ServiceNow and also shows where to view the resulting records in ServiceNow.

**Table 14:** Virtual machines exported from IT Asset Management and imported to ServiceNow

VM Type of Inventory record in IT Asset Management	CI class of Virtual Machine Instance record in ServiceNow	View in ServiceNow
AWS EC2	EC2 Virtual Machine Instance	no direct link, but you can enter the following into the ServiceNow Filter navigator to access view: <code>cmdb_ci_ec2_instance.list</code>
Hyper-V	Hyper-V Virtual Machine Instance	<b>Configuration &gt; Hyper-V &gt; Virtual Machine Instances</b>

VM Type of Inventory record in IT Asset Management	CI class of Virtual Machine Instance record in ServiceNow	View in ServiceNow
VMware	VMware Virtual Machine Instance	<b>Configuration &gt; VMware &gt; Virtual Machine Instances</b>
Zone	Solaris Virtual Machine Instance	no direct link, but you can enter the following into the ServiceNow Filter navigator to access view: <code>cldb_ci_solaris_instance.list</code>
LPAR	Virtual Machine Instance	<b>Flexera Integration &gt; Imported Records &gt; Virtual Machines</b>
nPar		
Oracle VM		
SRP		
vPar		
WPAR		
Unknown		



**Tip:** Integration properties allow you to choose whether to use IT Asset Management or ServiceNow as the source of truth to create and update records. For information about other properties that allow you to choose whether IT Asset Management or ServiceNow should be used as the source of truth to update CI classes, see [Adding Inventories](#), [Updating inventories](#), and [Updating inventory class name \(sys\\_class\\_name\)](#) in [Integration Properties](#).

# 2

## Architecture, Components, and Prerequisites

IT Asset Management (Cloud)

This chapter provides information you need to get started with the integration application.

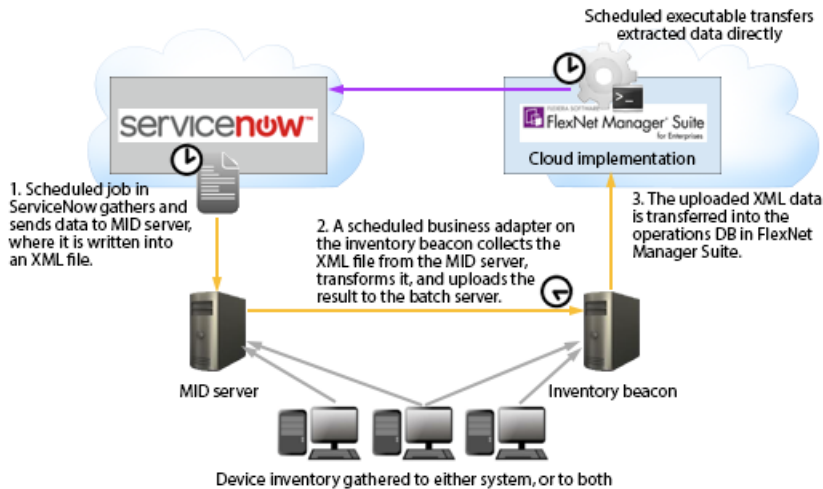
The information is divided into the following sections:

- [Architecture](#) — architectural diagrams to help you understand both configuration and operation of the integration between ServiceNow and IT Asset Management
- [Prerequisites](#) — requirements for integrating IT Asset Management and ServiceNow
- [Download Adapter Tools Archive](#) — how to download the Adapter Tools archive that includes content needed for many adapters

### Architecture

IT Asset Management (Cloud)

This big-picture overview provides context for understanding both configuration and operation of the integration between ServiceNow and IT Asset Management.



The following two processes (illustrated in the diagram) operate completely independently:

- The export of computer, application, and contract data from IT Asset Management into ServiceNow (purple arrow). For details, see [Process for Exports from IT Asset Management to ServiceNow](#).
- The export of computer and contract data from ServiceNow for import into IT Asset Management (gold arrows). For details, see [Process for Exports from ServiceNow to IT Asset Management](#).

## Prerequisites

## IT Asset Management (Cloud)

The following are requirements for integrating IT Asset Management release 2024 R2.3 and ServiceNow. At a high level, you need:

- A functional implementation of IT Asset Management
- An operational ServiceNow instance
- Installation of the ServiceNow integration application
- A ServiceNow integration account user with `x_fls_flexera_fnms.admin` role assigned

The following provides further details of these requirements:

- You need a functional implementation of IT Asset Management.



**Important:** To take full advantage of the features included in the v4.0 and later integration applications, it is recommended that you run IT Asset Management 2019 R1 or later. For more information about limitations with earlier releases of IT Asset Management, see the knowledge article [Feature Support for ServiceNow Integration](#), in the Flexera Customer Community.

- You must have a license issued by Flexera that permits use of the integration application. This license authorizes all communications in both directions through the integration application. To check, go to the **IT Assets License** page (**Administration > IT Asset Management Settings > IT Assets License**). Check under the **License details** for ServiceNow integration enabled: Yes. If this is not the case, request the license from your Flexera



representative.

- You must have an operational inventory beacon that communicates successfully with the cloud-based application server for IT Asset Management.
- You must have an operational ServiceNow instance.
  - For data flows from ServiceNow to IT Asset Management, you need a MID server configured for your ServiceNow implementation.



**Tip:** If you prefer, your FlexNet Beacon and ServiceNow MID server can be implemented on the same physical computer (provided that there are common communications requirements — for example, you don't have a case where one requires a proxy server setting that the other cannot use).

- For data flows from IT Asset Management to ServiceNow: Your ServiceNow instance has the Software Asset Management (SAM) Foundation plugin installed. Installed application data goes to SAM tables.
- For information on supported versions, go to the ServiceNow Store and search for 'Flexera'.



**Note:** ServiceNow has [deprecated older versions of Transport Layer Security \(TLS\)](#). Requires version 1.2.

- If this is your first ever install of the IT Asset Management integration application for ServiceNow, you must install it from the ServiceNow Store (see [Installing the Flexera Integration Application from the ServiceNow Store](#)), and then any updates to the app are also done through the ServiceNow store. Alternatively, if you have previously installed an earlier version of the integration using an update set, refer to the knowledge article [Upgrading ServiceNow Integration Application Using an Update Set](#) in the Flexera Customer Community.
- For first time users, and if you will be exporting data from ServiceNow to IT Asset Management, two business adapters (`ServiceNowAssets.xml` and `ServiceNowContracts.xml`) must be used for assets and contracts. These can be found in the unzipped adapter tools archive (see [Download Adapter Tools Archive](#)). For more information, refer to [Configuring FlexNet Beacon for Import](#).

## Download Adapter Tools Archive

IT Asset Management (Cloud)

The Adapter Tools archive includes content for many adapters, and is updated on the Flexera website from time to time.

Start this procedure using a web browser on a computer that has good network accessibility from all the machines needing installations for your adapter.



**To download the adapter tools archive:**

1. Download the latest Adapter Tools for IT Asset Management `version.zip` archive from the Flexera Product and License Center:
  - a. Access <https://community.flexera.com/s/article/adapter-tools-for-flexnet-manager-suite>.
  - b. In that article, click Adapter Tools for FlexNet Manager Suite - Cloud Edition.

A new browser tab may appear temporarily, and the download of `Adapter Tools for IT Asset Management 2024 R2.3.zip` commences.

- c. In your browser dialog, choose to save the file, and if the browser allows it, direct the saved file to a convenient working location (such as `C:\temp` on a central, accessible server).

If your browser saves the file to a default location (such as your `Downloads` folder), move or copy it to the appropriate working location when the download is finished.

2. Right-click the zip archive, and choose **Extract All...**

The folders are now available for the range of adapters in the Adapter Tools archive.

# 3

## Installation and Configuration

### IT Asset Management (Cloud)

Because the system involves a number of different servers, there are several installation and configuration steps. These are described in full in the following topics.

The information is divided into the following sections:

- [Setting Up the Integration](#) — common tasks that must be completed for all implementations
- [Setting Up Data Flows from IT Asset Management to ServiceNow](#) — required if you want to export the IT Asset Management data on hardware assets, installed applications and contracts for use within ServiceNow
- [Setting Up Data Flows from ServiceNow to IT Asset Management](#) — required if you want to import assets and/or contracts data into IT Asset Management.

## Setting Up the Integration

### IT Asset Management (Cloud)

Most of the groundwork on the IT Asset Management side has already been completed as part of product installation. Therefore the balance of the work that is common to both directions of data flow involves setting up ServiceNow.

- In ServiceNow terminology, you need to install an integration 'application' in ServiceNow (see [Installing the Flexera Integration Application from the ServiceNow Store](#)).
- There must also be a user account correctly configured in ServiceNow (see [Creating a ServiceNow Integration User](#)).

## Installing the Flexera Integration Application from the ServiceNow Store

### IT Asset Management (Cloud)

The steps in this section provide steps for a *first time install* of the integration application for ServiceNow. If you have previously installed an earlier version of the integration using an update set, instead refer to the knowledge article [Upgrading ServiceNow Integration Application Using an Update Set](#) in the Flexera Community (login required).

**To install the integration application for ServiceNow from the ServiceNow store:**

1. Log into ServiceNow Store with ServiceNow HI credentials.
2. In the **Search** box, enter Flexera Integration and press Enter.  
Application information appears in the search results.
3. From the search results, click **Flexera Integration**.  
Information about the application displays.
4. In the right pane of the application window, view the **Compatibility** section to ensure that the integration application is compatible with the version of ServiceNow that you require. If the version of ServiceNow that you require is not listed, contact IT Asset Management Technical Support.
5. In the right pane of the application window, click the **Get** button.  
A notice appears.
6. At the bottom of the notice, click **Continue**.
7. On the **Purchase of Flexera Integration** page, choose the **Make available on specific instances** radio button, and click the **Select** button.  
A **Select Instances** dialog appears.
8. In the **Available Instances** pane, double-click the instance that you want the app to be available on and click **OK**.
9. Select the **I accept** check box (to accept terms of use and subscription terms and conditions) and click **Get**.
10. Log into the ServiceNow instance where you made the integration application available. (You selected this instance in step 8).
11. In the Filter navigator, enter System Applications and then under **System Applications**, click **Applications**.  
The **Applications** list appears.
12. Click the **Downloads** tab.  
The **Downloads** list appears.
13. Scroll to the **Flexera Integration** application, and click the **Install** button.

The application now appears in the ServiceNow menu, displayed in the left-hand navigation panel as **Flexera Integration**.

## Creating a ServiceNow Integration User

IT Asset Management (Cloud)

This simple process sets up the user account used for integration between Service Now and IT Asset Management, and assigns the user to the appropriate role. Complete this task while you are still logged into ServiceNow as an administrator.

**To create the user account in ServiceNow:**

1. In ServiceNow, go to **User Administration > Users**.
2. Click **New**, complete the properties for your new user, and click **Submit**.



**Tip:** Take note of the user ID and password, which you will need again later in the set-up processes.

3. Once the creation process finishes, click the hyperlinked **User ID** for this user, and scroll down to **Roles**.
4. Click **Edit**, and select `x_fls_flexera_fnms.admin` from the collection, and add to **Roles List**.
5. Click **Save**.
6. Close the user properties page.

## Setting Up Data Flows from IT Asset Management to ServiceNow

### IT Asset Management (Cloud)

For records of installed applications and hardware inventory, IT Asset Management is considered the authoritative source of truth. Data on installed applications and hardware inventory, as well as contracts, can be exported from IT Asset Management to ServiceNow. The summary of the export process is:

1. On a weekly schedule (3am on Sunday mornings, application server time), an export utility extracts the appropriate data from IT Asset Management and transfers it to ServiceNow using API calls. This communication relies on *all* of the following details (configured as described in the following topics):
  - The URL of your ServiceNow instance
  - The user account registered for accessing ServiceNow
  - The communications token copied from ServiceNow and saved in IT Asset Management.
2. In ServiceNow, the IT Asset Management integration application processes the data received through the API calls, and schedules imports of that data using the ServiceNow Import Set.



**Tip:** The Transform Maps used during import to switch data from one product's formats to the other are also defined in the integration application.

If you wish to make use of this data flow, you need to complete the following processes:

- [Configuring ServiceNow for Import](#)
- [Configuring IT Asset Management for Export](#)

## Configuring ServiceNow for Import

### IT Asset Management (Cloud)

Continue the following while still (or again) logged into ServiceNow with administrator privileges.

**To configure imports into ServiceNow:**

1. Go to **Flexera Integration > Generate Token**.
2. From the account details you created earlier (see [Creating a ServiceNow Integration User](#)), insert the user ID in the **Username** field and supply the **Password**, and click **Generate OAuth Token**.

A long Refresh Token appears at the top of the page.

3. Copy this Refresh Token value, and transfer to the web interface of IT Asset Management.
4. When doing Installed Application data export, data is automatically imported to SAM tables.
5. Change the account used for processing the import in ServiceNow. To do this:

- a. Go to **Flexera Integration > Advanced > Scheduled Data Imports**.

The Scheduled Data Imports list appears.

- b. Click **FNMS Import Schedule**.



**Note: Flexera Schedule Import** is currently only used by Flexera Data Platform. IT Asset Management users should select **FNMS Import Schedule**.

- c. Change the **Run as** user to the integration user. (This is the user ID that you created earlier (see [Creating a ServiceNow Integration User](#)).
- d. Click **Update**.

## Configuring IT Asset Management for Export

IT Asset Management (Cloud)

You need to complete this configuration once to commence operations. Repeat if your ServiceNow details change at any time in the future. You may also use this same page to trigger an immediate export from IT Asset Management to ServiceNow.

**To configure export from IT Asset Management:**

1. Ensure that you are (still) logged into IT Asset Management with an account that is a member of the Administrator role, and that this account has the **Configure** access right.
2. In the web interface, go to the IT Asset Management Settings **General** page (**Administration > IT Asset Management Settings > General**), select the **Integrations** tab, and expand the **ServiceNow** accordion.



**Tip:** The **Integrations** tab is only visible when your account has the access rights described above.

3. In **Instance URL**, enter the protocol and path to your ServiceNow website (for example, `https://YOUR-INSTANCE.service-now.com`).
4. In **Username**, enter the account you use to access ServiceNow. This must exactly match the **User ID** value shown in the ServiceNow interface (see [Creating a ServiceNow Integration User](#)).

5. Paste the **Token** value from ServiceNow into the **Token** field on this page (see [Configuring ServiceNow for Import](#)).



**Tip:** The **Username** is mandatory for IT Asset Management to commence the export process, but it is not transferred back to ServiceNow, as the **Token** value alone is now sufficient for authentication.



**Note:** In future, if you wish to temporarily disable these data exports from IT Asset Management, you can remove any one or more values from **Instance URL**, **Username**, or **Token**.

6. If necessary, change the default selection of data types to export to ServiceNow. It is best practice to export all available data types.



**Warning:** Hardware inventory details are critical to the ServiceNow data set. Contracts and installed applications both have dependencies on assets in that system. If you clear the **Hardware inventory** check box, you may produce gaps in the contract and software records imported into ServiceNow.

7. Click **Save**.

When details are complete, the export from IT Asset Management to ServiceNow is triggered weekly at 3am on Sunday mornings. If you need to trigger an additional export (for example, when commencing integrated operations), continue with the rest of this process.



**Tip:** While the first export of any given data type is always a full export of all active records, subsequent data transfers from IT Asset Management to ServiceNow are differential (that is, only data that is new or changed since the last successful export of the same type is included). If you wish to transfer all available active data (rather than the differential data set), select the **Perform full export** check box before continuing.

8. Click **Export** to trigger an immediate export using the settings saved in the fields displayed above.

The **Export** button triggers a process that:

- Extracts the required data of the first data type from your operations database (the normal order when all are selected is hardware inventory, contracts, installed applications)
- Segments the data for easier transmission across the Internet, minimizing time-out risks with ServiceNow
- Reassembles the data into staging tables in ServiceNow
- Repeats the process for the next selected data type.

Separate transforms on your ServiceNow instance then map each data type from the staging tables to your operational ServiceNow tables.

To verify progress of a test run:

1. In the ServiceNow navigation bar, go to **Flexera Integration > Import History > Import Transactions**. Each of these transactions represents a data chunk\* or segment transferred to ServiceNow. The segmentation is necessary because of the large number of records that may need to be transferred. When the data is received by ServiceNow, the integration application separately schedules each transaction for import into the operational database tables in ServiceNow. As each transaction is completely imported, its **State** column value is set to Succeeded.
2. Similarly, access **IT Asset Management > Integration History > Import Runs**. Each data type exported from IT

Asset Management creates a separate entry here for import into ServiceNow. For each data type, when all its individual transactions have succeeded, the **State** value on the overall Import Run is also set to Succeeded.

3. Finally, when all the transferred data types have been successfully imported, a status report is sent to IT Asset Management. On the **System Settings** page in the **Integrations** tab, the status is displayed under **Last completed export**.

\* The configuration for data transfers from IT Asset Management to ServiceNow uses these values:

- The number of database records included in each transferred segment: 2000
- The number of retries if ServiceNow returns a connection failure: 2
- The length of time to wait for ServiceNow to respond before timing out: 2 seconds
- The maximum number of records of each data type to be included in the transfer, by type:
  - Hardware inventory: 500,000
  - Contracts: 500,000
  - Installed applications: 500,000.

## Setting Up Data Flows from ServiceNow to IT Asset Management

### IT Asset Management (Cloud)

Details of hardware assets and/or contracts, for which ServiceNow is considered the authoritative source of truth, can be imported into IT Asset Management from ServiceNow. In summary, the process is:

1. Each data collection is triggered by a scheduled job in ServiceNow that invokes a script.
2. The script uses ServiceNow database views to gather the required data and transfers it, in chunks, to the appropriate MID server.
3. The MID server writes the chunks into an XML file.
4. A separate schedule configured on the inventory beacon for IT Asset Management triggers an import through a business adapter, collecting the XML file, transforming it, and uploading it to the central application server.
5. The normal import processes on the application server imports the data (transformed as appropriate) into the operations databases for IT Asset Management.

If you wish to make use of this data flow, you need to complete the following processes:

- [Setting Up a MID Server](#)
- [Configuring ServiceNow for Export](#)
- [Configuring FlexNet Beacon for Import.](#)

No configuration of the central application server is needed, since this part of the integration leverages all standard operating processes.



# Setting Up a MID Server

IT Asset Management (Cloud)

You may already have an operational MID server (accessible from your chosen inventory beacon) that you wish to use for data transfers from ServiceNow to IT Asset Management. If so, and the MID server is validated in ServiceNow, you may skip this topic.

On the other hand, you may prefer to set up a specific MID server especially for this integration. If so, follow these steps while you are still logged into ServiceNow with administrator privileges.



## **To set up a MID server that stages data from ServiceNow:**

1. In the navigation bar, expand **MID Server** and click **Downloads**.
2. Click the link appropriate for your platform.
3. In the navigation bar, in the **MID Server** group, click **Installation Instructions**, and click the link appropriate to your version of ServiceNow.

The documentation opens in a new tab.

4. Step through those instructions to complete installation of your MID server.

Take particular note of the exact name you give this MID server, as you will need it again soon. (It is also available through **MID Server > Servers**.)

5. Be sure to validate your MID server (see the **Validate the MID Server** link on the MID Server installation page).

# Configuring ServiceNow for Export

IT Asset Management (Cloud)

Now that you have a MID server installed, two further configuration points for ServiceNow remain:

- Identifying the MID server as the one to use for the exports to IT Asset Management
- Scheduling the exports from ServiceNow.

Continue the following while still logged into ServiceNow with administrator privileges.



## **To configure ServiceNow for export:**

1. In ServiceNow, go to **Flexera Integration > Integration Properties**.
2. In the list of **Integration Properties**, scroll down to the **MID server name** field (in the **ServiceNow → FNMS** section), and enter the full names of the MID servers. If you have multiple MID servers, enter their names here in a comma-separated list (no spaces).
3. Optionally, in the **Export file path** field, for each MID server, enter an alternative file path location to store the XML file (instead of the default MID server location). When using multiple MID servers, enter the paths for the MID servers as a comma-separated list (no spaces) in the same order they appear in the **MID server name** field.

Because the FlexNet Beacon reads this file using the business adapter, FlexNet Beacon needs access to this file location.



**Note:** The MID servers must have the following attributes: **Status of Up, Validated** is **True**, and **Unresolved Issues** is **0**.

4. Scroll down (if necessary), and click **Save**.
5. In ServiceNow, go to **Flexera Integration > Export Configuration > Scheduled Jobs**.

The list of standard jobs to export data from ServiceNow is displayed, including:

- Export Assets from ServiceNow
- Export Contracts from ServiceNow.

Each of these jobs needs to be configured for your environment.

6. Add a **Run as** column to this page and enter the integration user ID in the column for both jobs. To do so:
  - a. Click the Personalized List (gear) icon in the upper left corner at the top of list of scheduled jobs.
  - b. On the **Personalize List Columns** dialog, select **Run as** from the **Available** columns and move it to the **Selected** columns.
  - c. On the Scheduled Script Executions page, change the **Run as** user (that is blank by default) to the integration user ID (see [Creating a ServiceNow Integration User](#)).
7. Click the first export task to schedule (such as Export Assets from ServiceNow).

The **Scheduled Script Execution** properties are displayed.



**Note:** The MID server collects the exported XML data in chunks, such that the XML file is not well formed until the last chunk has been received. Be sure to schedule sufficient time for the exports to complete before the scheduled import from the MID server into IT Asset Management.

8. Change the value in the **Run** field to your desired schedule.

These data exports may be large, based on the size of your ServiceNow repository. A common practice would be to choose **Weekly** exports.



**Tip:** ServiceNow by default limits its exports from the database to 10,000 records (for a general discussion, see <https://community.servicenow.com/community/blogs/blog/2014/08/01/increasing-the-export-limit>, but note that this page discusses limits for exports from the ServiceNow user interface grids, which are not relevant to the database exports to IT Asset Management). You can modify the maximum number of records exported from ServiceNow by having your ServiceNow administrator add the property `glide.db.max_view_records` and setting a new limit. For instructions on adding the property, see [http://wiki.servicenow.com/index.php?title=Adding\\_a\\_Property#gsc.tab=0](http://wiki.servicenow.com/index.php?title=Adding_a_Property#gsc.tab=0).

9. Use the additional fields that appear to complete configuring your scheduled job.

For example, for **Weekly** exports, set the **Day**, and **Time** of day, when the export should occur. An off-peak time is preferable.

10. Click **Update** to save your changes.

Repeat the scheduling for the **Export Contracts from ServiceNow** task.

11. With both exports now scheduled, run a test of each by clicking **Execute Now** on the associated **Scheduled**

**Script Execution** properties page.

(This same button can be used to run the export if you chose the **Run On Demand** setting for the schedule.) Use the test to assess the time taken to complete the export and finish a well-formed XML file on the MID server, as a check on your scheduling plans. The test also allows you to troubleshoot any problems.

**12. Monitor overall progress in ServiceNow by going to **Flexera Integration > Export History > Export Runs**.**

The **State** column on this view shows the progress of export, and the **Stage** column displays **File Completion** when the XML file is done.

**13. On the MID server, check for well-formed (complete) XML files.**

The file names are of the form `x_fls_flexera_fnms_ExportType.xml`, where the placeholder *ExportType* is either **Asset** or **Contract**.

This completes the configuration of ServiceNow for data flows from ServiceNow to IT Asset Management. These same data flows also require some configuration of IT Asset Management to receive the data. This side of the configuration is covered next.

## Configuring FlexNet Beacon for Import

IT Asset Management (Cloud)

If you are not already familiar with importing business-related data to IT Asset Management, the following terminology summary may help:

- A "business adapter" is an XML file that allows configuration of a connection to a source of business data, as well as mapping of that data from the source format to the database tables and columns required in IT Asset Management. The data from ServiceNow is handled as business data, and therefore data transfers make use of business adapters.
- The "Business Adapter Studio" is a utility available on each inventory beacon for customizing and testing business adapters.
- The "Business Importer" is the process that exercises the business adapters to the schedule you define, and controls each import as defined in its business adapter.

In previous tasks, you have configured ServiceNow to run two exports that queue data about hardware assets and contracts and send it to the MID server of your choice. On the MID server, the data is written into XML files. Once these have been completed, two business adapters running on your inventory beacon collect this data and upload it to your central operations databases. You must use the following process to configure these business adapters before they can operate.



**Tip:** Clearly, the inventory beacon and the ServiceNow MID server must access a common folder for the transfer of the XML files. It is possible that the FlexNet Beacon software and the ServiceNow MID server are installed on the same physical machine; but if they are on separate machines, network access allowing a file share is required. This means you may need to grant at least read access for the share on your MID server to two accounts on your inventory beacon:

- The local administrator account with which you will log in now for configuring and testing your business adapters
- The account that runs your FlexNet Beacon Service, which will exercise the business adapters in production (by default, this is the local **SYSTEM** account on your inventory beacon).

As well as access to the MID server, your inventory beacon requires access to the unzipped archive you downloaded for the ServiceNow integration application to complete this process.



**To configure an inventory beacon for importing data for IT Asset Management:**

1. On your chosen inventory beacon, log in as a local administrator.
2. If necessary, validate that, from this server, you can access the file share where the MID server saves the completed XML files.
3. In the unzipped archive you downloaded for the ServiceNow integration application, open the Importer folder and locate the two business adapters `ServiceNowAssets.xml` and `ServiceNowContracts.xml`.
4. Copy these two files to the business adapters folder, `%CommonAppData%\Flexera Software\Beacon\BusinessAdapter`, on the inventory beacon.
5. Start the FlexNet Beacon software from the Windows start menu on the inventory beacon, using the **Run as administrator** option.
6. In the interface for FlexNet Beacon, select the **Business Importer** page from the **Connections** group (in the left-hand navigation pane).

The two adapters appear in the list of available business adapters.

7. For each of these adapters in turn, select the adapter from the list, and click **Edit...** In the following dialog, click **Edit Adapter**.

The Business Adapter Studio opens, editing this adapter.

8. In the Business Adapter Studio, select the top level import node (like `ServiceNowAssets` or `ServiceNowContracts`) and locate the **File name** field. Complete the file path and file name for the appropriate XML file on the MID server. If you have already exported data from ServiceNow to the MID server, you may click the **...** button and browse to the respective XML file in the folder on the MID server.

*The following example from the assets adapter shows a path when the MID server and inventory beacon are installed on the same physical computer:*

```
C:\ServiceNow\MyMIDServerFolder\agent\x_fls_flexera_fnms_asset.xml
```

*The following example from the contracts adapter shows a path when the MID server is separate, using UNC format:*

```
\\MyMIDServer\ServiceNow\MyMIDServerFolder\agent\x_fls_flexera_fnms_contracts.xml
```

9. When the file path and name are identified, click **Refresh** (just below the **...** button).
10. **Save** the modified adapter; and repeat the process for the other adapter.
11. When both are completed and saved, close Business Adapter Studio.
12. If you have not already created the schedule(s) on which you want to run these adapters, switch to the **Scheduling** page of the FlexNet Beacon interface, and create one or two now.

You may run these two business adapters on the same schedule, as the inventory beacon queues them and

runs them one after the other.



**Important:** Remember that there is a delay between when the scheduled job starts executing in ServiceNow, and when the MID server finishes assembling the transmitted chunks into a valid XML file. You can test the elapsed time for your environment by triggering each export, and waiting until, in the **Export Runs** view (in ServiceNow), it reaches the *Succeeded* state. Allow a safety buffer for future data growth, and set the schedule for the business adapters to run after the exports and assembly of the XML files are completed.

**13.** In the **Business Importer** page of the FlexNet Beacon, select each adapter in turn, and:

- a. Click **Schedule....**
- b. From the list of available schedules, select the one for each adapter.
- c. Click **OK**.
- d. Ensure that the adapter is marked as **Enabled**.
- e. Repeat for the other adapter.

**14.** At the bottom of the inventory beacon interface, click **Save** to store your changes.

**15.** Optionally, if you have test data waiting for a system check, you can select each adapter in turn, and click **Execute Now**.

The Business Importer takes the raw data from XML files, transforms it into the formats required for IT Asset Management, and uploads it to the central application server.

To check the status of imports, in the web interface of IT Asset Management, navigate to the system menu (⚙️ ▼ in the top right corner) and choose either **Data Inputs** (in the **Business Data** tab) or **System Health > System Tasks**.



**Tip:** If you need to interrupt the export from ServiceNow, disable the scheduled jobs in ServiceNow, and disable the relevant schedule(s) in FlexNet Beacon.

## 4

# Operational Details

## IT Asset Management (Cloud)

The exchange of data between ServiceNow and IT Asset Management happens in a pair of independent processes driven by schedules (either Windows scheduled tasks or schedules built in to the products).

As these schedules by default run on a weekly cycle, you may expect the data to be synchronized after the weekly runs and up until either system is subsequently updated. If you require an intermediate synching after an important update to one system or the other, you can also trigger either process manually.

The process flows are described in detail in the following topics. Notes about configuration and customization are included.

## Process for Exports from IT Asset Management to ServiceNow

### IT Asset Management (Cloud)

A Windows scheduled task `Export to ServiceNow` triggers exports to ServiceNow at 3am every Sunday morning.

If you are in an administrator role, you can also trigger an export through the web interface for IT Asset Management. Go to the IT Asset Management Settings **General** page (**Administration > IT Asset Management Settings > General**) and select the **Integrations** tab. Expand the **ServiceNow** section.

### Process Details

1. On the central application server, when the export is triggered, a utility is launched that:
  - Checks whether you have licensed the ServiceNow integration option from Flexera (if not, it shows an error in the same page of the web interface).
  - Tests your ServiceNow credentials in the **ServiceNow export settings** section of the **Integrations** tab in the IT Asset Management Settings **General** page (**Administration > IT Asset Management Settings > General**).
2. Next, the utility checks whether ServiceNow is able to accept an import at this time:
  - It checks whether a record in `Scheduled Data Imports`, and another in `Data Sources`, have been created

by the integration application. (Links to the relevant pages where you can inspect the records in ServiceNow are under **Flexera Integration > Advanced**.)

- It ensures that no prior import is already in process, checking that there is *no* record in the Import Runs table or the Import Transactions table that are incomplete (that is, have a State other than Succeeded or Failed). Both lists are available for inspection in the **IT Asset Management Integration History** group.

If either of the checks fails, the utility abandons the export and displays an appropriate error.

3. The utility checks the connection with the operations databases for IT Asset Management. (If there is any failure, the utility abandons the export and displays an appropriate error.)
4. All being well, the data export, transfer, and import processes are run for all data types (unless any of them have been specifically excluded) in the following order:
  - a. Hardware inventory
  - b. Contracts
  - c. Installed applications



**Caution:** In ServiceNow, contracts and installed applications both have dependencies on hardware inventory. It is recommended that you do not exclude the hardware inventory, as this may result in unpredictable gaps in other records when the correct dependencies cannot be established.

The data collected is differential (that is, collecting only additions and changes since the last export). To prevent timeout issues, each data set is split into segments for transfer to ServiceNow. Each segment is identified with a transaction ID, transaction type, and the data in an XML chunk. ServiceNow returns a return code for each segment of data.

- If the return code shows a failure, the segment is retransmitted for a maximum number of tries, after which the utility exits (skipping any remaining exports) and displays an appropriate error.
  - While success continues, the utility waits for each data type to complete before commencing the next data type.
5. In ServiceNow, as the first data chunk is received, the integration application creates a record in the Import Runs table. (For details of the main columns in the Import Runs table, see [Import Runs Columns](#).)




**Note:** If you have selected to store installed application data to SAM Foundation Plugin tables but the SAM plugin is not found, then ServiceNow logs an error in the Application Log and returns an error code. In IT Asset Management, the utility abandons the export and displays an appropriate error in the log file.

6. For each data chunk (including the first), ServiceNow creates a record in the Import Transactions table. (For details of the main columns in the Import Transactions table, see [Import Transactions Columns](#).)
7. When the transaction record is created, it returns a success code to IT Asset Management, which then processes and transmits the next data chunk. This loop continues until all the records of a particular export type have been transmitted.
8. In ServiceNow, the inbound data chunks are written to staging tables (visible in the navigation panel under **Flexera Integration > Staging Records**) based on the transaction type:

Transaction type	Import Set Table
inventory_export	Inventory Imports (note that this is hardware inventory)
application_export	Application Imports
contract_export	Contract Imports
connection_test	Not applicable
export_status	Not applicable

9. In ServiceNow, the integration application uses the Scheduled Data Imports and Data Sources records to process each Import Transaction record. The integration application queues the transaction records, and as each record is processed, it sets its State to Succeeded. When all the transaction records for an Import Run are completed, the integration application also sets its State to Succeeded.
10. During this processing loop, IT Asset Management continues polling the API for status. Only when it receives a successful completion message does it resume the process with the next selected data type.
11. When the data has been collated from the individual transactions to the staging tables, transform maps are executed to map the fields in the import set tables to fields in ServiceNow database tables. The final tables for data from each transaction type are shown below. (For a complete set of the transform mappings, see [Transform Maps for ServiceNow Integration](#).)

Transaction type	Ultimate Data Tables in ServiceNow
application_export	Software Model, Discovery Model, Software Installation, and Software Instance. The tables used are the SAM tables for installation.
contract_export	Contract and Lease Instance
 <b>Tip:</b> Check the <b>Contract used by</b> tab near the bottom of the page of contract properties. This references all computers linked to the contract.	
inventory_export	Product Model, Virtual Machine Instance, Computers

At the completion of this process, data exported from IT Asset Management is reflected in your ServiceNow data set.

## Import Runs Columns

IT Asset Management (Cloud)

These are the major properties of the entries in the **Import Runs** view created as ServiceNow commences each data import. Go to **Flexera Integration > Import History > Import Runs**.



Column	Notes
Number	An automatically-generated sequential numbering of each created record in this listing.
Import Id	A GUID created by IT Asset Management as it exports the data, used to track activity and exchanges for this particular import into ServiceNow.
Import Type	The type of data being imported, being one of Inventory, Contract, or Application.
State	<p>Values include:</p> <ul style="list-style-type: none"> <li>• <b>New</b> — This default state is set when a record is created in the <b>Import Runs</b> listing.</li> <li>• <b>Waiting</b> — ServiceNow has received the final data chunk for the particular Import Type.</li> <li>• <b>Succeeded</b> — All data chunks for this Import Run have been processed by ServiceNow.</li> <li>• <b>Failed</b> — For some reason (unspecified here), the import has failed.</li> </ul>

## Import Transactions Columns

IT Asset Management (Cloud)

These are the major properties of the entries in the **Import Transactions** view created as ServiceNow receives each data segment for import. Go to **Flexera Integration > Import History > Import Transactions**.

Column	Notes
Number	An automatically-generated sequential numbering of each created record in this listing.
Import Run	The ID of the import run of which this transaction is a part.
Import Type	The type of data being imported, being one of Inventory, Contract, or Application.

Column	Notes
State	<p>Values include:</p> <ul style="list-style-type: none"> <li>• <b>New</b> — This default state is set when a record is created in the <b>Import Transactions</b> listing.</li> <li>• <b>Waiting</b> — ServiceNow is processing another transaction received earlier.</li> <li>• <b>Processing</b> — Set when the integration application invokes import of this transaction.</li> <li>• <b>Succeeded</b> — ServiceNow has completed the import of this transaction.</li> <li>• <b>Failed</b> — For some reason (unspecified here), the import of the transaction has failed.</li> </ul>
Payload	The segment of data, in XML format, that makes up this transaction.

## Transform Maps for ServiceNow Integration

### IT Asset Management (Cloud)

These transform maps map hardware, license, and contract data collected from IT Asset Management into ServiceNow.

Data collected from IT Asset Management is initially held in staging tables within ServiceNow, and must be transformed for insertion in the operational tables in your ServiceNow implementation. For example, the FNMS Inventory -> Computer set has columns that are matched with the Computers table in ServiceNow. Contract data is first mapped into the Contracts table in ServiceNow, and a second transform map is run to provide links to relevant Configuration Items (Computers).

The following tables show the standard transformations from the staging tables (the source) to operational tables (the target) within ServiceNow. Items marked [Script] involve a data transformation, which may involve finding the foreign key to another record, or conversion of units such as bytes to MB.

In each transform, the "(key)" fields are used for record matching. If a record already exists with identical key values for fields so marked, it is updated; and if not, a new record is created.



**Note:** There are integration properties that define:

- Whether a new record should be created in ServiceNow when a match for incoming data is not found
- Whether an existing record should be updated when a match is found

These properties can be modified in ServiceNow by navigating to **Flexera Integration > Integration Properties**.

The transform maps in this topic are separated into the following respective data group that they are used for:

- [Transform maps for inventory data](#)
- [Transform maps for installed application data](#)
- [Transform maps for contracts data](#)

## Transform maps for inventory data

### Computer Model Transform

- Original data from IT Asset Management: Inventory
- Target table in ServiceNow: Product Model [cmdb\_model]

Source Display Name	Source Field	Target Display Name	Target Field
ModelNo (key)	u_modelno	Model number	model_number
ChassisType	u_chassistype	Type	type
ComputerType	u_computertype	Model categories	cmdb_model_category
Manufacturer	u_manufacturer	Manufacturer	manufacturer
ModelNo	u_modelno	Name	name

### Virtual Machine Transform

- Original data from IT Asset Management: Inventory.
- Target table in ServiceNow: Virtual Machine Instance [cmdb\_ci\_vm\_instance]

Source Display Name	Source Field	Target Display Name	Target Field
ComputerID (key)	u_computerid	FlexNet Computer ID	x_fls_flexera_fnms_computer_id
[Script]	[Script]	Is deleted	x_fls_flexera_fnms_is_deleted
[Script]	[Script]	Correlation ID	correlation_id
[Script]	[Script]	Class	sys_class_name
[Script]	[Script]	Disks size (GB)	disks_size
[Script]	[Script]	Memory (MB)	memory
[Script]	[Script]	Fully qualified domain name	fqdn
[Script]	[Script]	Sys ID	sys_id
ComputerName	u_computername	Name	name
DiscoveredDate	u_discovereddate	First discovered	first_discovered
Domain	u_domain	Domain	sys_domain
InventoryDate	u_inventorydate	Most recent discovery	last_discovered
IPAddress	u_ipaddress	IP Address	ip_address
MACAddress	u_macaddress	MAC Address	mac_address
Manufacturer	u_manufacturer	Manufacturer	manufacturer
ModelNo	u_modelno	Model ID	model_id

Source Display Name	Source Field	Target Display Name	Target Field
ModelNo	u_modelno	Model number	model_number
NumberOfProcessors	u_numberofprocessors	CPUs	cpus
SerialNo	u_serialno	Serial number	serial_number

### Computer Transform

- Original data from IT Asset Management: Inventory
- Target table in ServiceNow: Computer [cmdb\_ci\_computer]

Source Display Name	Source Field	Target Display Name	Target Field
ComputerID (key)	u_computerid	FlexNet Computer ID	x_fls_flexera_fnms_computer_id
[Script]	[Script]	RAM (MB)	ram
[Script]	[Script]	Correlation ID	correlation_id
[Script]	[Script]	Disk space (GB)	disk_space
[Script]	[Script]	Sys ID	sys_id
[Script]	[Script]	Is Virtual	Virtual
AssetID	u_assetid	FlexNet Asset ID	x_fls_flexera_fnms_asset_id
CalculatedUser	u_calculateduser	Calculated User	x_fls_flexera_fnms_calculated_user
ChassisType	u_chassistype	Chassis type	chassis_type
ComputerName	u_computername	Name	name
ComputerStatus	u_computerstatus	Status (hardware_status)	hardware_status
ComputerType	u_computertype	Subcategory	subcategory
DiscoveredDate	u_discovereddate	Discovered date	x_fls_flexera_fnms_discovered_date
DiscoveredDate	u_discovereddate	First discovered	first_discovered
Domain	u_domain	Domain	sys_domain
[script]	[script]	Fully qualified domain name	fqdn
InventoryConnection Name	u_inventoryconnection name	Inventory Connection	x_fls_flexera_fnms_inventory_connection
InventorySource	u_inventorysource	Inventory Source	x_fls_flexera_fnms_inventory_source
IPAddress	u_ipaddress	IP Address	ip_address
IsDeleted	u_isdeleted	Is deleted	x_fls_flexera_fnms_isdeleted

Source Display Name	Source Field	Target Display Name	Target Field
LastLoggedInUser	u_lastloggedinuser	Last logged in user	x_fls_flexera_fnms_last_logged_in_user
MACAddress	u_macaddress	MAC Address	mac_address
Manufacturer	u_manufacturer	Manufacturer	manufacturer
MaxClockSpeed	u_maxclockspeed	CPU speed (MHz)	cpu_speed
ModelNo	u_modelno	Model ID -> model_number	model_id
ModelNo	u_modelno	Model number	model_number
InventoryDate	u_inventorydate	Most recent discovery	last_discovered
NumberOfCores	u_numberofcores	CPU core count	cpu_core_count
NumberOfProcessors	u_numberofprocessors	CPU count	cpu_count
NumberOfThreads	u_numberofthreads	CPU core thread	cpu_core_thread
OperatingSystem	u_operatingsystem	Operating System	os
ProcessorType	u_processortype	CPU type	cpu_type
SerialNo	u_serialno	Serial number	serial_number

### Network Adapter Transform

- Original data from IT Asset Management: Inventory
- Target table in ServiceNow: Network Adapter [cmdb\_ci\_network\_adapter]

Source Display Name	Source Field	Target Display Name	Target Field
[Script]	[Script]	Name	name
MACAddress	u_macaddress	MAC Address	mac_address
Manufacturer ID	u_manufacturer_id	Mac manufacturer	mac_manufacturer
Netmask	u_netmask	Netmask	netmask
IPAddress	u_ipaddress	IP Address	ip_address
Is deleted	u_is_deleted	Is deleted	x_fls_flexera_fnms_is_deleted
[Script]	[Script]	Configuration Item	cmdb_ci
Data source	u_data_source	Data source	x_fls_flexera_fnms_data_source
Data token	u_data_token	Data token	x_fls_flexera_fnms_data_token
DHCP Enabled	u_dhcp_enabled	DHCP Enabled	dhcp_enabled
[Not applicable]	[Not applicable]	Created	created

### IP Address Transform

- Original data from IT Asset Management: Inventory
- Target table in ServiceNow: IP Address [cmdb\_ci\_ip\_address]

Source Display Name	Source Field	Target Display Name	Target Field
IPAddress	u_ipaddress	IP Address	ip_address
[Not applicable]	[Not applicable]	IP Version	ip_version
[Not applicable]	[Not applicable]	NIC	nic
[Script]	[Script]	Configuration Item	ci_item
DiscoveredDate	u_discovereddate	First discovered	x_fls_flexera_fnms_discovered_date
DiscoveredAt	u_discoveredat	Most recent discovery	x_fls_flexera_fnms_last_discovered



**Note:** Script is run to query the cmdb\_ci\_computer table. If the CI is found, the script returns the sys\_id for REF. If the CI is not found, this field is left blank.

## Transform maps for installed application data

### Software Transform



**Note:** This transform map is not applicable when the Software Asset Management (SAM) Foundation plugin is installed.

- Used when you use CMDB tables for Installed Application data.
- Original data from IT Asset Management: Application.
- Target table in ServiceNow: Software [cmdb\_ci\_spkg]

Source Display Name	Source Field	Target Display Name	Target Field
FlexeraID (key)	u_flexeraid	Flexera Unique ID	x_fls_flexera_fnms_id
ApplicationID	u_applicationid	FlexNet Application ID	x_fls_flexera_fnms_application_id
ApplicationName	u_applicationname	Package name	package_name
ApplicationVersion	u_applicationversion	Version	version
Classification	u_classification	Subcategory	subcategory
ProductName	u_productname	Name	name
Publisher	u_publisher	Manufacturer	manufacturer

### Software Instance Transform



**Note:** This transform map is not applicable when the Software Asset Management (SAM) Foundation plugin is installed.

- Used when you opt to use CMDB tables for Installed Application data.
- Original data from IT Asset Management: Application.
- Target table in ServiceNow: Software Instance [cmdb\_software\_instance]



**Tip:** When application data is imported into ServiceNow, the installed application records are linked to their computer record based on FlexNet Computer ID. This is recommended and the most reliable way to link records. However, if for any reason you need to link records based on Computer Serial Number; note that the installed application records do have Computer Serial Number available and therefore you can manually edit the transform map in ServiceNow to change the key to be Computer Serial Number.

Source Display Name	Source Field	Target Display Name	Target Field
FlexeraID (key)	u_flexeraid	Software -> Product Name	Product Name
[Script] (key) [Note 1]	[Script]	Installed on	installed_on
[Script]	[Script]	Is deleted	x_fls_flexera_fnms_is_deleted
DiscoveredAt	u_discoveredat	Discovered at	x_fls_flexera_fnms_discovered_at
DiscoveredBy	u_discoveredby	Discovered by	x_fls_flexera_fnms_discovered_by
FlexeraID	u_flexeraid	Flexera Unique ID	x_fls_flexera_fnms_id
LastScanned	u_lastscanned	Last scanned	x_fls_flexera_fnms_last_scanned
ProductName	u_productname	Name	name

Note:

1. An installed application record defines the installation of an application on a computer. A source record from FlexNet Manager Suite in the Application Import [x\_fls\_flexera\_fnms\_application\_import] table has these values.

The import process creates or updates such a record in ServiceNow in the Software Instance [cmdb\_software\_instance] table. The record links an application record in the Software [cmdb\_ci\_spkg] table and a computer record in Computer [cmdb\_ci\_computer].

Field map, [Script] (key), finds a computer record in the Computer table based on source ComputerID [u\_computerid] in the target FlexNet Computer ID [x\_fls\_flexera\_fnms\_computer\_id] field. Another field map, FlexeraID (key) finds an application record in the Software table based on the source FlexeraID [u\_flexeraid] field in the target Flexera Unique ID [x\_fls\_flexera\_fnms\_id] field.

### Software Model Transform

- Used when you opt to use SAM tables for Installed Application data.
- Original data from IT Asset Management: Application
- Target table in ServiceNow: Software Model [cmdb\_software\_product\_model]

Source Display Name	Source Field	Target Display Name	Target Field
FlexeraID (key)	u_flexeraid	FlexNet Manager Id	x_fls_flexera_fnms_id
ApplicationName	u_applicationname	FlexNet Application Name	x_fls_flexera_fnms_application_name
ApplicationVersion	u_applicationversion	Version	version
Classification	u_classification	Type	type
FlexeraID	u_flexeraid	Short description	short_description
ProductName	u_productname	Name	name
Publisher	u_publisher	Manufacturer	manufacturer

### Software Installation Transform

- Used when you opt to use SAM tables for Installed Application data.
- Original data from IT Asset Management: Application
- Target table in ServiceNow: label [cmdb\_sam\_sw\_install]



**Tip:** When application data is imported into ServiceNow, the installed application records are linked to their computer record based on *FlexNet Computer ID*. This is recommended and the most reliable way to link records. However, if for any reason you need to link records based on *Computer Serial Number*; note that the installed application records do have *Computer Serial Number* available and therefore you can manually edit the transform map in ServiceNow to change the key to be *Computer Serial Number*.

Source Display Name	Source Field	Target Display Name	Target Field
ApplicationID (key)	u_applicationid	FlexNet Application ID	x_fls_flexera_application_id
[Script] (key) [Note 1]	[Script]	Installed on	installed_on
[Script]	[Script]	Is deleted	x_fls_flexera_fnms_is_deleted
ApplicationVersion	u_applicationversion	Version	version
DiscoveredAt	u_discoveredat	Last Discovered	x_fls_flexera_fnms_last_discovered
DiscoveredBy	u_discoveredby	Discovered by	x_fls_flexera_fnms_discovered_by
DisplayName	u_displayname	Display name	display_name
FlexeraID	u_flexeraid	Discovery model -> prod_id	discovery_model
FlexeraID	u_flexeraid	Prod id	prod_id
LastScanned	u_lastscanned	Last scanned	last_scanned
Publisher	u_publisher	Publisher	publisher

Note:



1. An installed application record defines the installation of an application on a computer. A source record from FlexNet Manager Suite in the Application Import [x\_fls\_flexera\_fnms\_application\_import] table has these values.

The import process creates or updates such a record in ServiceNow in the Software Installation [cmdb\_sam\_sw\_install] table. The record links an application record in the Software Discovery Model [cmdb\_sam\_sw\_discovery\_model] table and a computer record in Computer [cmdb\_ci\_computer].

Field map, [Script] (key), finds a computer record in Computer table based on source ComputerID

[u\_computerid] in the target FlexNet Computer ID [x\_fls\_flexera\_fnms\_computer\_id] field.

Another field map, FlexeraID (key) finds an application record in the Software Discovery Model table based on the source FlexeraID [u\_flexeraid] field in the target Prod ID [prod\_id] field.

## Transform maps for contract data

### Contracts Transform

- Original data from IT Asset Management: Contracts
- Target table in ServiceNow: Contract [ast\_contract]

Source Display Name	Source Field	Target Display Name	Target Field
ContractNumber (key)	u_contractnumber	Contract number	vendor_contract
ContractName	u_contractname	Description	short_description
ContractStatus	u_contractstatus	State	state
ContractType	u_contracttype	Short Description	short_description
EndDate	u_enddate	Ends	ends
IsDeleted	u_isdeleted	Is deleted	x_fls_flexera_fnms_is_deleted
StartDate	u_startdate	Starts	starts
Vendor	u_vendor	Vendor	vendor

### Contract Instance Transform

- Original data from IT Asset Management: Contracts
- Target table in ServiceNow: Lease Instance [ast\_contract\_instance]

Source Display Name	Source Field	Target Display Name	Target Field
ContractNumber (key)	u_contractnumber	Contract	ast_contract -> vendor_contract
ContractType	u_contracttype	Contract Type	contract_type
See note.	See note.	Configuration Item	ci_item



**Note:** Script is run to query cmdb\_ci\_computer table. If the CI is found, the script returns the sys\_id for REF. If the CI is not found, this field is left blank.

# Process for Exports from ServiceNow to IT Asset Management

IT Asset Management (Cloud)

How the import of data from ServiceNow works.

## Process Details

1. Two scheduled jobs are run separately (and not at the same time) in ServiceNow.



**Note:** These jobs are visible in **Configuration > Scheduled Jobs** as *Export Assets from ServiceNow* and *Export Contracts from ServiceNow*. Click the job names to make them active or inactive, to modify the schedule, or to execute either one immediately.

2. When either job is executed, ServiceNow creates an XML file on the MID server with the data from the database view. Go to **Flexera Integration > Export Configuration > Database Views**. The filename is `x_fls_flexera_fnms_asset.xml` for assets or `x_fls_flexera_fnms_contract.xml` for contracts. The files are overwritten with every export.



**Tip:** Within the **Database Views**, you can specify additional tables to join to the database view, or within a table, you can add additional columns to join to the database view. To preview the records for an export (without actually invoking the export), click the view name in that page to display the properties of the view; and under the **Related Links** heading, click *Try It*.

3. After the export is executed in ServiceNow, a record is created under **Export Runs**. Go to **Flexera Integration > Export History > Export Runs**. ServiceNow updates the state of this record to *Succeeded* when the export is successfully completed.
4. Two independent business adapters from IT Asset Management (on potentially independent schedules) connect from your inventory beacon to your MID server, and collect the latest XML files saved there, converting them into the intermediate data form required for business data uploads. (For details of the mapping between source data in ServiceNow and destination fields in the IT Asset Management database, see [Business Adapter Mappings](#).)



**Important:** The business adapters will fail if they are run while the XML files on the MID server are incomplete. Be sure that the schedules for export from ServiceNow, and the running of the business adapters, allow sufficient time for the export to be completed. For large data sets, this may be several hours.

5. As the adapters complete their run, the data set is uploaded immediately to the application server for IT Asset Management. (There is also a catch-up upload task that runs overnight to retry any failed uploads.)
6. On a separate schedule, the batch server starts a business import job. This imports data from the data package into the IT Asset Management database.

At the completion of this process, data exported from ServiceNow is reflected in your IT Asset Management data set.

# Properties for Export Columns

IT Asset Management (Cloud)

These are the major properties of the entries in the **Export Runs** view created as ServiceNow commences each data export. Go to **Flexera Integration > Export History > Export Runs**.

Column	Notes
Number	An automatically-generated sequential numbering of each created record in this listing.
Export Type	The type of data being exported, either Contract or Asset.
Stage	<p>May have the following values:</p> <ul style="list-style-type: none"> <li>File Creation — Displayed when the integration application sends the opening XML tag to the MID Server to be commence writing the XML file.</li> <li>Data Collection — Displayed while the integration application gathers the data and sends it to MID Server to be written into the XML file. The data is sent in multiple chunks of 500 records each.</li> <li>File Completion — Displayed when the closing XML tag has been sent to the MID Server.</li> </ul>
State	<p>Provides additional insight into the Data Collection stage in particular. Values include:</p> <ul style="list-style-type: none"> <li>New — This default state is set when a record is created in the <b>Export Runs</b> listing.</li> <li>Processing — The integration application is reading from the ServiceNow database and compiling a chunk of data.</li> <li>Processed — All database records required for the current chunk (maximum: 500) have been prepared.</li> <li>Waiting — The data chunk is being transferred to the MID server.</li> <li>Ready — The MID server has notified that the data chunk has been added to the XML file. At this point, the integration application sets the state back to Processing and prepares the next chunk of data.</li> <li>Succeeded — All data chunks have been transferred, and no more are required.</li> <li>Failed — For some reason (unspecified here), the export has failed.</li> </ul>
Counter	The number of database records that the integration application has read and processed. Because the data is transferred to the MID server in a number of chunks, the counter value is used as a cursor for reading the correct records from the database for the next data chunk.

# Business Adapter Mappings

IT Asset Management (Cloud)

Two business adapters are supplied as a standard part of the ServiceNow integration application for IT Asset Management, and are used in the process of transferred asset and contract data from ServiceNow to IT Asset Management. The following lists show the source data from ServiceNow, and the target (destination) fields in IT Asset Management, for each of these adapters in turn.



**Tip:** For special purposes, mappings can be added, removed, or changed by editing the business adapter(s) in Business Adapter Studio on your inventory beacon.

In each listing, the first field shown is used for matching existing records (a key). During import, if the value in this field already exists in the compliance database, the record is updated with the other values imported for this item. Otherwise, a new record is automatically created in IT Asset Management.

In both listings below, "display" names are the names visible in the two products' web interfaces; and the **Source Field** column shows a compound name made up of:

- An abbreviated reference to a ServiceNow database table name (see below)
- An underscore character
- The field name in the relevant database table (which name may contain additional underscores).

The mapping of the abbreviated references to the actual database table names in the ServiceNow database is:

- acntrct represents ast\_contract
- ahrdwr represents alm\_hardware
- cicomp represents cmdb\_ci\_computer
- cmdbmdl represents cmdb\_model
- sysusr represents sys\_user.

## Assets Imports

- Business adapter file name: ServiceNowAssets.xml
- Original data source: ServiceNow database view x\_fls\_flexera\_fnms\_asset
- Destination table in IT Asset Management database: Asset

Source Display Name	Source Field	Target Display Name	Target Field
<b>Serial number</b>	cicomp_serial_number	<b>Serial Number</b>	SerialNumber (key)
<b>Asset tag</b>	ahrdwr_asset_tag	<b>Asset Tag</b>	AssetTag
<b>Asset Status</b>	ahrdwr_install_status	<b>Asset Status</b>	AssetStatus
<b>Configuration Item</b>	ahrdwr_ci	<b>Name</b>	ShortDescription
<b>Installed</b>	cicomp_install_date	<b>Installed on</b>	InstallationDate

Source Display Name	Source Field	Target Display Name	Target Field
<b>Manufacturer</b>	cicomp_manufacturer	<b>Manufacturer</b>	Manufacturer
<b>ModelNo</b>	cicomp_model_number	<b>ModelNo</b>	ModelNo

Notes:

1. AssetStatusID is a foreign key to the AssetStatus table, from which display values are drawn.



**Note:** *AssetStatus* is a mandatory field for asset records in IT Asset Management. The out of the box business adapter does explicit mapping between ServiceNow **Status** values and IT Asset Management *AssetStatus* values, as shown in the following table. If needed, you can modify these mappings in the business adapter or if you have custom mappings defined in ServiceNow, you must include them here. Note that if you are also 'round-tripping' this data (that is, copying it back from to ServiceNow at some future point, to maintain synchronization), you need to provide a similar mapping in the relevant transform map as well.

ServiceNow	IT Asset Management
On order	Purchased
In stock	In Storage
In transit	Purchased
In use	Installed
Consumed	Installed
In maintenance	Other
Retired	Retired
Missing	Other

## Contracts Imports

- Business adapter file name: ServiceNowContracts.xml
- Original data source: ServiceNow database view x\_fls\_flexera\_fnms\_contract
- Destination table in IT Asset Management database: Contract

Source Display Name	Source Field	Target Display Name	Target Field
<b>Contract number</b>	acntrct_vendor_contract	<b>Contract Number</b>	ContractNo (key)
<b>Description</b>	acntrct_description	<b>Information</b>	Comments
<b>Display name</b>	cmdbmdl_display_name	<b>Contract type</b>	ContractTypeID [Note 1]
<b>Ends</b>	acntrct_ends	<b>Expiry date</b>	EndDate
<b>Short description</b>	acntrct_short_description	<b>Contract Name</b>	ContractName

Source Display Name	Source Field	Target Display Name	Target Field
<b>Starts</b>	acntrct_starts	<b>Start date</b>	StartDate
<b>State</b>	acntrct_state	<b>Status</b>	ContractStatusID [Note 2]

Notes:

1. ContractTypeID is a foreign key to the ContractType table, from which display values are drawn. Also see note below.
2. ContractStatusID is a foreign key to the ContractStatus table, from which display values are drawn.



**Note:** *ContractTypeID is a mandatory field for contract records in IT Asset Management. If a record in ServiceNow has a **Display name** value that does not exist in IT Asset Management, the record is imported with the contract type General. To work around this type change, you can modify the business adapter to provide an explicit mapping between ServiceNow **Display name** values and IT Asset Management *ContractTypeID* values. Note that if you are also 'round-tripping' this data (that is, copying it back from IT Asset Management to ServiceNow at some future point, to maintain synchronization), you need to provide a similar mapping in the relevant transform map as well.*

## 5

# Appendices

## IT Asset Management (Cloud)

You can find additional information about using the integration application in the following topics:

- [Integration Properties](#)
- [Import Properties](#)
- [Additional ServiceNow Indexes for Performance](#)
- [Removing a Legacy Integration Application](#)
- [Configuring the Software Asset Management Foundation Plugin](#)
- [Deleting Records from ServiceNow](#)

## Integration Properties

### IT Asset Management (Cloud)

The **Integration Properties** page lets you specify export integration properties. This page is accessible from **Flexera Integration > Integration Properties**. The integration properties let you choose whether to accept IT Asset Management as the source of truth for various data types. Choosing **No** for any option sets ServiceNow as the source of truth for that property and choosing **Yes** for any option sets IT Asset Management as the source of truth for that property.

The descriptions in the following sections show you which properties are for use with exports from ServiceNow and which properties are for use with exports from IT Asset Management.

### Properties used when exporting from ServiceNow for input to IT Asset Management

The following table provides descriptions of properties to use when exporting from ServiceNow for input to IT Asset Management.

Setting/Property	Description
<b>Exclude virtual machine assets</b>	Choose whether to exclude virtual machine assets from the export from ServiceNow.

Setting/Property	Description
<b>MID server name</b>	Enter the full name of the MID server. If you have multiple MID servers, enter their names here in a comma-separated list (no spaces).
<b>Export file path</b>	Optionally, enter a file path for where to store the exported XML data file from ServiceNow (instead of storing it in the default MID server location). When using multiple MID servers, enter the paths for the MID servers as a comma-separated list (no spaces) in the same order they appear in the <b>MID server name</b> field. Because the FlexNet Beacon reads this file using the business adapter, FlexNet Beacon needs access to this file location.
<b>Configure the log verbosity (View the Application logs at System Logs - System Log - Application logs)</b>	Choose the desired level of log verbosity: <ul style="list-style-type: none"> <li>• <b>Error</b></li> <li>• <b>Warning</b></li> <li>• <b>Information</b></li> <li>• <b>Debugging</b></li> <li>• <b>Tracing</b></li> </ul>
<b>Use IRE in Flexera Data Integration</b>	Choose whether to use the Identification and Reconciliation Engine (IRE) in Flexera Data Integration: <ul style="list-style-type: none"> <li>• <b>Yes:</b> IT Asset Management will use IRE in Flexera Data Integration.</li> <li>• <b>No:</b> IT Asset Management will not use IRE in Flexera Data Integration.</li> </ul>

## Properties used when exporting from IT Asset Management for input to ServiceNow

The following table provides descriptions of properties to use when exporting from IT Asset Management for input to ServiceNow.



Setting/Property	Description
<b>IT Asset Management is source of truth for:</b>	<p>For any properties in the following list, choose <b>Yes</b> to accept IT Asset Management as the source of truth, or choose <b>No</b> to specify ServiceNow as the source of truth:</p> <ul style="list-style-type: none"> <li>• <b>Adding inventories</b></li> <li>• <b>Updating inventories</b></li> <li>• <b>Updating inventory class name (sys_class_name)</b></li> <li>• <b>Adding installations</b></li> <li>• <b>Updating installations</b></li> <li>• <b>Adding contracts</b></li> <li>• <b>Updating contracts</b></li> </ul> <p>The following two properties are set to <b>No</b> by default. If you would like to have all the MAC and IP address information associated to a computer to be included in the import, choose <b>Yes</b> for both properties. This will possibly create multiple records for each computer.</p> <ul style="list-style-type: none"> <li>• <b>Allow Flexera to insert records into the network adapter table</b></li> <li>• <b>Allow IP address records (Requires network adapter creation)</b></li> </ul>
<b>Configure the log verbosity (View the Application logs at System Logs - System Log - Application logs)</b>	<p>Choose the desired level of log verbosity:</p> <ul style="list-style-type: none"> <li>• <b>Error</b></li> <li>• <b>Warning</b></li> <li>• <b>Information</b></li> <li>• <b>Debugging</b></li> <li>• <b>Tracing</b></li> </ul>

## Additional ServiceNow Indexes for Performance

IT Asset Management (Cloud)

Creating additional indexes on your ServiceNow instance substantially improves the intake of data exported from IT Asset Management.

ServiceNow enables you to create your own database indexes. A worked example suggests that, with the following

indexes added, you can expect better than four-fold performance increase in the initial data transforms to finish the import of data that has been exported from IT Asset Management.

Database table	New column indexed
Contract [ast_contract]	vendor_contract
Software Model [cmdb_software_product_model]	x_fls_flexera_fnms_id
Product Model [cmdb_model]	model_number
Computer [cmdb_ci_computer]	x_fls_flexera_fnms_computer_id
Software Installation [cmdb_sam_sw_install]	x_fls_flexera_fnms_application_id

## Removing a Legacy Integration Application

IT Asset Management (Cloud)

The integration application for ServiceNow (from version 3.0 and later of the adapter) is a scoped application. The previous integration applications used global scope. While two integration applications can be present in ServiceNow at the same time (with different scopes), it is recommended that you remove any previously-installed integration application for IT Asset Management after installing the newer version and successfully running the migration script.



**Note:** Using the following process to remove all the components of the legacy integration application (including its menu, tables, scripts, and user interface components). As expected, this process does not remove any records from ServiceNow core tables.



**To remove a legacy integration application:**

1. In ServiceNow, navigate to **System Applications > Applications**, and click **FlexNet Manager Suite Integration**.
2. On the page that appears, click **Delete**.

A confirmation dialog appears asking you to type in the word “delete”.

3. Enter delete in the dialog, and click **Ok**.

The application deletion progress dialog appears, and the integration application is deleted.



**Tip:** Sometimes deleting the integration application fails to remove the Scheduled Jobs it created. To check, or to delete them manually:

- a. In ServiceNow, navigate to **System Scheduler > Scheduled Jobs > Scheduled Jobs**.
- b. Filter the list of scheduled jobs to find those with Name starting with Export.
- c. If the following scheduled jobs exist, delete them manually:
  - Export Assets from ServiceNow
  - Export Contracts from ServiceNow.

# Configuring the Software Asset Management Foundation Plugin

## IT Asset Management (Cloud)

When you export application data from IT Asset Management for use in ServiceNow, the export utility `fnmp_servicenow_export.exe` makes use of tables that are part of a plugin for ServiceNow. The naming of this plugin, and your method of enabling it, have changed considerably across various releases of ServiceNow. Choose the appropriate step below depending on your current version of ServiceNow.



### **Summary of procedures across ServiceNow releases:**

1. Up to and including the Helsinki release of ServiceNow, the following applies:
  - The plugin name is `Software Asset Management`, and its ID is `com.snc.software_asset_management`.
  - This plugin has a dependency on a second one named `Software Asset Management Core` with the ID `com.snc.sam.core`.
  - Your ServiceNow administrator can make the `Software Asset Management` plugin Active in the **System Plugins** page of your ServiceNow instance. This action automatically makes the `Software Asset Management Core` plugin active as well.
2. For the Istanbul release of ServiceNow, the following applies:
  - If you had activated the `Software Asset Management` plugin in an earlier version of ServiceNow, and your ServiceNow instance has now been upgraded to Istanbul, there should be no need to activate the plugin a second time.
  - The plugin feature is called "Software Asset Management Template", but still accessed through the **System Plugins** page of your ServiceNow instance.
  - The plugin name is `Software Asset Management`, and its ID is `com.snc.software_asset_management`.
  - (There is no visible dependency on any other plugin, as had been the case for earlier releases of ServiceNow.)
  - Your ServiceNow administrator can make the `Software Asset Management` plugin Active in the **System Plugins** page of your ServiceNow instance.
3. For the Jakarta release of ServiceNow, the following applies:
  - If you had activated the `Software Asset Management` plugin in an earlier version of ServiceNow, and your ServiceNow instance has now been upgraded to Jakarta, there should be no need to activate the plugin a second time.
  - Confusingly, there is a new and different plugin named `Software Asset Management Premium`, and its ID is `com.snc.samp`. This is used internally by a new SAM application that is subscription based.



**Important:** This new subscription-based application and its plugin are not relevant to the integration with IT Asset Management, and this subscription is not required for the integration. Do not be confused by the similarities. Be aware that, when current ServiceNow documents refer to "Software Asset Management" or

---

"SAM", they are generally referring to the new application and the premium plugin it uses, and this is not relevant to your project for integration with IT Asset Management.

- The plugin from previous releases, named Software Asset Management or Software Asset Management Template, with its ID `com.snc.software_asset_management`, is no longer visible in the **System Plugins** page of your ServiceNow instance. However, it is still supported for the Jakarta release, and is still required for the integration between ServiceNow and IT Asset Management.



**Tip:** Some documents from ServiceNow refer to this plugin as the "Base SAM plugin" or the "old SAM plugin", and the like. Be prepared to be precise, and refer to its ID `com.snc.software_asset_management` to be sure the correct plugin is identified.

- To activate the "base SAM plugin", send a request to ServiceNow Support to activate the Software Asset Management Template plugin with the ID `com.snc.software_asset_management` on your ServiceNow instance.
4. For the Kingston release of ServiceNow, the following applies:
- For the Kingston release, do *not* activate the earlier plugin named Software Asset Management, with its ID `com.snc.software_asset_management`. This legacy plugin is only applicable to the Jakarta and earlier releases.
  - Instead, for Kingston, activate the Software Asset Management Foundation plugin, with its ID `com.snc.sams`. This plugin does not require any additional licensing.
  - Contact ServiceNow Support to request activation of the Software Asset Management Foundation plugin ( `com.snc.sams`).

## Deleting Records from ServiceNow

IT Asset Management (Cloud)

If an asset or application record exists both in IT Asset Management and ServiceNow but is then deleted in IT Asset Management, the records are marked for deletion in ServiceNow during the next export from IT Asset Management. However, the records are not automatically deleted. To delete the records from ServiceNow, you need to create a Global business rule and include a script with a condition: `current.state == 'Succeeded'`.



**To delete records in ServiceNow using a global business rule:**

1. In ServiceNow, navigate to **System Definition**, and click **Business Rule**.

The **Business Rules** page appears.

2. Click **New**.
3. Do the following.
  - a. In the **Name** field, enter a name for the rule:
  - b. From the **Table** field, select `x_fls_flexera_fnms_import_transaction`.

- c. Ensure that the **Application** field is set to **Global**.
- d. Select the **Active** check box.
- e. Select the **Advanced** check box.
- f. On the **Advanced** tab, enter the following in the **Condition** field:

```
current.state == 'Succeeded'
```

- g. On the **Advanced** tab, enter the following into the **Script** field:

```
var util = new FNMSUtil();

util.removeIsDeletedRecordsFromTable('cmdb_ci_computer');

util.removeIsDeletedRecordsFromTable('cmdb_ci_vm_instance');

util.removeIsDeletedRecordsFromTable('cmdb_sam_sw_install');
```

# XVIII

## ServiceNow Inventory Adapter

### IT Asset Management (Cloud)

Separately from the sharing of information about contracts and assets described in the previous section, it is also possible to use ServiceNow as an inventory source, importing data to help in your software asset management and, in particular, license consumption calculations. To do this, you need to configure an *inventory connector* to collect and import the additional information.

This inventory connector is quite different and separate from the "ServiceNow Integration" discussed in [ServiceNow Integration with IT Asset Management](#) (and its following topics). It is quite possible to run both kinds of data exchange within the same overall implementation, but it is very helpful to keep the concepts clearly separated. The key differences are:

Functionality	SN Inventory Connector	SN Integration
Data flow	From ServiceNow into IT Asset Management	Bi-directional
Data type	Hardware and raw software inventory	<ul style="list-style-type: none"><li>• Computer, normalized application data, and contract records into ServiceNow</li><li>• Asset and contract records into IT Asset Management</li></ul>
Data app	Flexera Integration, from the ServiceNow Store (common app)	Flexera Integration, from the ServiceNow Store (common app)
Data import	Simple inventory connector	Two business adapters are required for the imports into IT Asset Management.

### Limitations

ServiceNow is not currently strong in collecting inventory from (or, in ServiceNow terminology, "discovery of") partitioning technologies that provide virtualization on several UNIX-like platforms. If you are supporting these platforms, best practice is to install the FlexNet Inventory Agent locally on them, reporting directly to your inventory beacon(s), thereby allowing collection of all details needed for your license consumption calculations. This additional information integrates smoothly with the information collected from ServiceNow, with details collected by FlexNet Inventory Agent taking precedence.

Because the ServiceNow inventory connector is strictly an *importer* of inventory discovered by ServiceNow, it can be conveniently implemented on an instance of ServiceNow already running the Flexera Integration application for integration with a separate Flexera tool that *exports* data to ServiceNow. The limitation is that only *one* of these exporting tools (IT Visibility, Data Platform, or the IT Asset Management integration for contracts and assets as described in the previous section) may be implemented at one time on the same instance of ServiceNow. However, once you have selected just *one* of these exporting tools, it can operate in parallel with the ServiceNow inventory connector described in this section.

## What you will find

This section includes the following chapters:

- [ServiceNow Inventory Connector](#) covering all prerequisites and the architecture that underlies the use of the connector.
- [Setting up ServiceNow for the Inventory Connector](#) provides details on all the setup required on the ServiceNow side to prepare the inventory information for collection by the inventory connector. Many of these detail are very similar to those described for ServiceNow configuration in the previous part; and if you are configuring *both* kinds of integration, you only need the ServiceNow configuration done once. Details are repeated (and fine tuned) in this section for those choosing to implement only the inventory connector.
- [Configuring the Inventory Connector on the Beacon](#) covers the IT Asset Management side of the inventory connector. Again, this is familiar territory for those who have set up several connectors; but details are provided here again for those not yet practised in this area.
- Technical appendices for this section give details of the data mapping on both sides of the inventory connector.

## 1

# ServiceNow Inventory Connector

## IT Asset Management (Cloud)

The ServiceNow inventory connector allows you to import into IT Asset Management inventory collected from target devices by ServiceNow (ServiceNow refers to this process as "discovery", whereas in IT Asset Management, "discovery" refers to just finding devices on the network, and "inventory" refers to gathering information about the software and hardware of the discovered device). With ServiceNow as an inventory source, the imported information includes both hardware information and raw evidence of installed applications. Once imported, the information is shown in IT Asset Management with a source type of either ServiceNow or ServiceWatch.

This introductory chapter gives useful orientation, and includes:

- [Prerequisites](#)
- [Purpose and Architecture](#).

## Prerequisites

### IT Asset Management (Cloud)

To import from the ServiceNow inventory connector, you need all of the following:

- Access to the current SaaS implementation of IT Asset Management.
- An installation of FlexNet Beacon 16.0.0 or later (supplied with IT Asset Management 2020 R2), with good network access to your chosen ServiceNow MID server on the one hand, and to the central application server for IT Asset Management on the other.



**Tip:** It is possible to install your ServiceNow MID server on your inventory beacon, provided that there are common communications requirements — for example, that you don't have a case where one requires a proxy server setting for reaching remote servers that the other cannot use. (No proxy is required or possible between the inventory beacon and the MID server, since data exchange at this level uses a simple file share.)

The inventory beacon you choose:

- Must have PowerShell 5.1 or later running on Windows Server 2012 or later, or on Windows 7 SP1 or later.
- Must have the Microsoft Windows environment variable PSModulePath set to include the following path:



```
%ProgramFiles%\WindowsPowerShell\Modules
```

- An implementation of ServiceNow release 5.3.316 or later.
- At least one of the following ServiceNow plugins (enabling the collection of the raw inventory information):
  - Discovery plugin – data is imported only when discovered by ServiceNow, so this one is mandatory; and it is also a prerequisite for the following:
  - SAM plugin – this plugin provides additional information in the file `flexera_software_install_export_set.csv`, and without this additional plugin, that `.csv` file remains empty.
- Flexera Integration application version 5.0.316 (or later) installed on ServiceNow (obtaining and installing this application are described in [Setting up the Integration Application](#)).

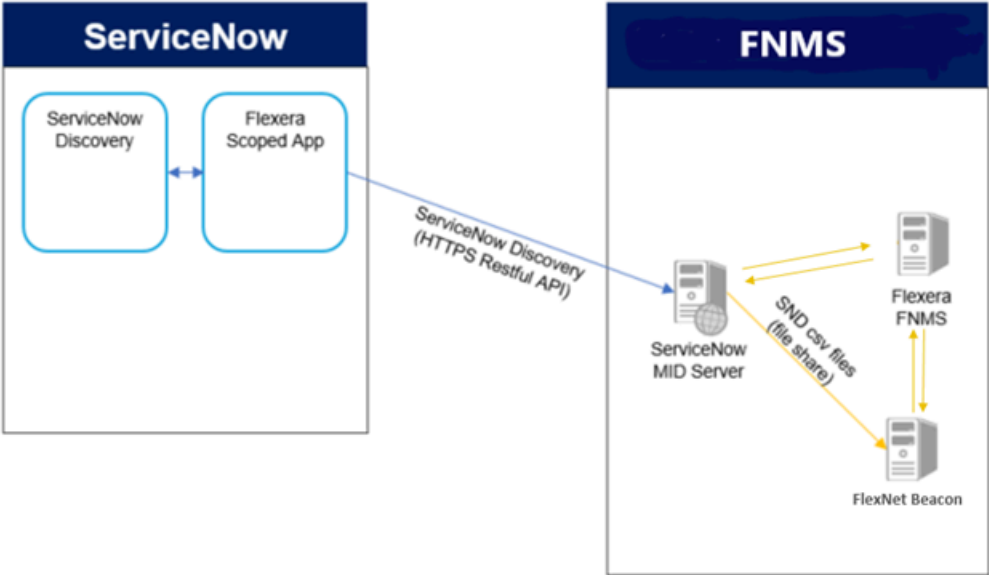
## Purpose and Architecture

### IT Asset Management (Cloud)

The ServiceNow inventory connector lets you use ServiceNow as an inventory source for IT Asset Management. With it, you can import raw evidence of software installations, along with hardware details of discovered devices, for use in your central software asset management (SAM) views and license consumption calculations. The connector does *not* import normalized data about what applications are installed – only the installation *evidence* is imported, so that, as normal, you rely on the Application Recognition Library to map that imported evidence to individual applications.

The inventory information discovered by ServiceNow first needs to be extracted and made accessible. This is done with the Flexera Integration app, available through the ServiceNow store (see installation and configuration instructions in [Setting up the Integration Application](#)). The Flexera Integration app saves the data in the `.csv` file format on your chosen ServiceNow MID server.

On your chosen inventory beacon, the ServiceNow inventory connector connects to the MID server and extracts the data from the `.csv` files, and uploads the inventory to the central application server for IT Asset Management.



# 2

## Setting up ServiceNow for the Inventory Connector

### IT Asset Management (Cloud)

Much of the setup on the ServiceNow side is almost identical for the inventory connector as it is for the integration of asset and contract data discussed in the previous section ([ServiceNow Integration with IT Asset Management](#)). Both require use of the Flexera Integration application, both need a suitable account to run that app, and both require scheduling and set-up on a ServiceNow MID server. If you are using *both* forms of data sharing, you may already have in place much that is covered in this chapter; but should still cherry-pick the details you may need to adjust. For those whose only sharing of ServiceNow data is through the inventory connector, you may find all the details in this chapter helpful. These include:

- [Setting up the Integration Application](#)
- [Creating a ServiceNow Integration User](#)
- [Setting Up a MID Server](#)
- [Scheduling Inventory Exports.](#)

## Setting up the Integration Application

### IT Asset Management (Cloud)

Use this process either for a first time installation of the Flexera Integration application in ServiceNow, or to upgrade to the latest version of that integration app.



**To install the integration application for ServiceNow from the ServiceNow store:**

1. Log into ServiceNow Store with ServiceNow HI credentials.
2. In the **Search** box, enter Flexera Integration and press Enter.  
  
Application information appears in the search results.
3. From the search results, click **Flexera Integration**.

Information about the application displays.

4. In the right pane of the application window, view the **Compatibility** section to ensure that the integration application is compatible with the version of ServiceNow that you require. If the version of ServiceNow that you require is not listed, contact IT Asset Management Technical Support.

5. In the right pane of the application window, click the **Get** button.

A notice appears.

6. At the bottom of the notice, click **Continue**.
7. On the **Purchase of Flexera Integration** page, choose the **Make available on specific instances** radio button, and click the **Select** button.

A **Select Instances** dialog appears.

8. In the **Available Instances** pane, double-click the instance that you want the app to be available on and click **OK**.
9. Select the **I accept** check box (to accept terms of use and subscription terms and conditions) and click **Get**.
10. Log into the ServiceNow instance where you made the integration application available. (You selected this instance in *step 8*).
11. In the Filter navigator, enter `System Applications` and then under **System Applications**, click **Applications**.

The **Applications** list appears.

12. Click the **Downloads** tab.

The **Downloads** list appears.

13. Scroll to the **Flexera Integration** application, and click the **Install** button.

The application now appears in the ServiceNow menu, displayed in the left-hand navigation panel as **Flexera Integration**. While you are logged into ServiceNow, complete the next process to set up the appropriate operating account (if you do not already have one.)

## Creating a ServiceNow Integration User

IT Asset Management (Cloud)

This simple process sets up the user account used to run the Flexera Integration app in Service Now, and assigns the user to the appropriate role. Complete this task while you are still logged into ServiceNow as an administrator.



**To create the user account in ServiceNow:**

1. In ServiceNow, go to **User Administration > Users**.
2. Click **New**, complete the properties for your new user, and click **Submit**.



**Tip:** Take note of the user ID and password, which you will need again later in the set-up processes.

3. Once the creation process finishes, click the hyperlinked **User ID** for this user, and scroll down to **Roles**.
4. Click **Edit**, and select `x_fls_flexera_fnms.admin` from the collection, and add to **Roles List**.

5. Click **Save**.
6. Close the user properties page.

## Setting Up a MID Server

IT Asset Management (Cloud)

You may already have an operational MID server (accessible from your chosen inventory beacon) that you wish to use for inventory transfers from ServiceNow to IT Asset Management. If so, and the MID server is validated in ServiceNow, you may skip this topic.

On the other hand, you may prefer to set up a specific MID server especially for this inventory connector. If so, follow these steps while you are still logged into ServiceNow with administrator privileges.



**To set up a MID server that stages data from ServiceNow:**

1. In the navigation bar, expand **MID Server** and click **Downloads**.
2. Click the link appropriate for your platform.
3. In the navigation bar, in the **MID Server** group, click **Installation Instructions**, and click the link appropriate to your version of ServiceNow.

The documentation opens in a new tab.

4. Step through those instructions to complete installation of your MID server.

Take particular note of the exact name you give this MID server, as you will need it again soon. (It is also available through **MID Server > Servers**.)

5. Be sure to validate your MID server (see the **Validate the MID Server** link on the MID Server installation page).

## Scheduling Inventory Exports

IT Asset Management (Cloud)

Now that you have a MID server installed, two further configuration points for ServiceNow remain:

- Identifying the MID server as the one to use for the inventory exports to IT Asset Management
- Scheduling the exports from ServiceNow.

Continue the following while still logged into ServiceNow with administrator privileges.



**To schedule inventory exports from ServiceNow:**

1. In ServiceNow, go to **Flexera Integration > Integration Properties**.
2. In the list of **Integration Properties**, scroll down to the **MID server name** field (in the **ServiceNow → FNMS** section), and enter the full names of the MID servers. If you have multiple MID servers, enter their names here in a comma-separated list (no spaces).
3. Optionally, in the **Export file path** field, enter an alternative file path location to store the .csv files.

By default, the .csv files are saved in a folder called `export` under the installation folder of the MID server software. You may use this folder, or another. Keep in mind that the inventory connector on the inventory beacon must have access to this file location, in order to read the .csv files. Take note of the name used so that you can correctly configure the connection from the inventory beacon.

4. Scroll down (if necessary), and click **Save**.
5. Still in ServiceNow, go to **Flexera Integration > Advanced > Scheduled Data Exports**.

The **Scheduled Data Exports** node includes standard jobs to export inventory from ServiceNow, including:

- Flexera Scheduled Computer Export
- Flexera Scheduled Software Export
- Flexera Scheduled Software Install Export.

Each of these jobs needs to be configured for your environment.

6. Click the first export task to schedule (such as Flexera Scheduled Computer Export). (Check the alerts bar at the top, as you may need an additional click there to allow editing the values for your chosen export task.)

The **Scheduled Data Export** properties are displayed. As you consider your timing, remember that this export forms only a part of the overall process, which also includes the inventory connection from your inventory beacon, the following upload to the central application server, and the import in time for the nightly license reconciliation calculations.

7. Change the value in the **Run** field to your desired schedule.

These data exports may be large, based on the size of your ServiceNow repository. A common practice would be to choose `Weekly` exports.



**Tip:** ServiceNow by default limits its exports from the database to 10,000 records. You can modify the maximum number of records exported from ServiceNow by having your ServiceNow administrator add the property `glide.db.max_view_records` and setting a new limit. For instructions on adding the property, see [http://wiki.servicenow.com/index.php?title=Adding\\_a\\_Property#gsc.tab=0](http://wiki.servicenow.com/index.php?title=Adding_a_Property#gsc.tab=0).

8. Use the additional fields that appear to complete configuring your scheduled job.

For example, for `Weekly` exports, set the **Day**, and **Time** of day, when the export should occur. An off-peak time is preferable.

9. Click **Update** to save your changes.

Repeat the scheduling for the Flexera Scheduled Software Export and Flexera Scheduled Software Install Export tasks.

10. With all exports now scheduled, run a test of each by clicking **Execute Now** on the associated **Scheduled Script Execution** properties page.

(This same button can be used to run the export if you chose the **Run On Demand** setting for the schedule.) Use the test to assess the time taken to complete the export and finish a well-formed .csv file on the MID server, as a check on your scheduling plans. The test also allows you to troubleshoot any problems.

11. Monitor overall progress in ServiceNow by going to **Flexera Integration > Export History > Export Runs**.

The **State** column on this view shows the progress of export, and the **Stage** column displays File Completion when the .csv files are done.

12. On the MID server, check for complete .csv files.



**Note:** *If you are not using the SAM plugin with ServiceNow, the exported `flexera_software_install_export_set.csv` file will be empty.*

This completes the configuration of ServiceNow for data flows from ServiceNow to IT Asset Management. The final stage is to configure IT Asset Management to import the inventory information.

## 3

# Configuring the Inventory Connector on the Beacon

IT Asset Management (Cloud)

If you are familiar with setting up inventory connections in FlexNet Beacon, the ServiceNow inventory connector is simply importing the data already saved by ServiceNow on the MID server. If you are not familiar with this task, check the instructions in the following two topics:

- [Setting the Import Schedule](#) — how to choose the appropriate schedule.
- [Setting Connector Details, and Validation](#) — how to configure the connector.

## Setting the Import Schedule

IT Asset Management (Cloud)

The schedule for the inventory connector on the inventory beacon forms the 'bridge' between the completed exports from ServiceNow on the one hand, and the nightly inventory imports and compliance (license consumption) calculations on the other. For example:

- If you have chosen a weekly schedule for export of inventory data from ServiceNow (see [Scheduling Inventory Exports](#)), then there is no point in scheduling anything more frequent for the import from the inventory beacon – repetitive uploads from unchanged .csv files just create unnecessary system load.
- Ideally, if you want to see the latest inventory from ServiceNow as early as possible after it has been updated, your inventory beacon schedule should allow for data collection, upload to the application server, and import, all prior to the overnight license consumption calculations, so that results are ready for you next morning.

These considerations may mean that you need a custom schedule for import from the ServiceNow inventory connector. It is convenient to have such a schedule configured before you set up the actual inventory connector, so that you can include the schedule within its configuration (rather than revisiting later).



---

### *To choose or configure a schedule for import:*

1. Run the inventory beacon interface (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).





**Tip:** Remember that you must run the inventory beacon software with administrator privileges.

2. From the **Data collection** group in the navigation bar, choose **Schedules**.
3. Review the list of available schedules (and in particular, the **Summary** column), to see whether you have an existing schedule than can be re-used for the ServiceNow inventory connector, taking into account the schedule optimizations mentions above.
  - If so, your work here is done: skip ahead to configuring the connector itself.
  - If not, continue here to create a purpose-built schedule.

4. Click **New...**

The **Edit Schedule** dialog appears.

5. Complete the details:

- a. Make the **Schedule name** distinct within the first few characters, such as SNIInvWkly.
- b. Select the radio button for the kind of schedule to align with your ServiceNow exports (such as weekly) and set the day and time the schedule triggers the inventory connector.

Remember that this is local time on the inventory beacon. Take into account any time differences between this local time and time on the MID server on one hand, and the central application server on the other.

6. Click **OK** to write the schedule into the list on the **Scheduling** page.

The list of schedules now contains unsaved information, shown by the trailing asterisk on the **Scheduling\*** entry in the navigation bar. You may save now, or you can apply your newly-created schedule to the new ServiceNow inventory connector (as follows) before saving.

## Setting Connector Details, and Validation

IT Asset Management (Cloud)

Ensure that your chosen inventory beacon meets all the prerequisites listed in [Prerequisites](#). Step through the following process while you are still logged into FlexNet Beacon as administrator.



**To configure the ServiceNow inventory connection on the inventory beacon:**

1. In FlexNet Beacon, in the **Data collection** group, click **Inventory systems**.  
The list of current connections for this inventory beacon appears.
2. Below the list, select the down arrow at the right of the **Next...** button, and from the drop-down options, choose **PowerShell**.  
The **Create PowerShell Source Connection** dialog appears.
3. Create a **Connection Name**.  
Make the name meaningful within the first few characters, as it will appear in lists with fairly narrow columns.
4. From the **Source Type** drop-down, choose **ServiceNow**.

On inventory beacon versions 16.0.0 and later, two additional sets of controls (for test mode, and filter for overlapping records) appear in the lower part of the same dialog. (The proxy server settings remain disabled, as these cannot apply for this connector.)

5. Ensure that the **Source Folder** field contains the correct path to the folder on the MID server where the .csv files are saved.

By default, this is the export folder under the MID server installation directory. You may have customized this path (see [Scheduling Inventory Exports](#)). Ensure that the exact path is specified here.

6. If your inventory beacon is version 16.0.0 or later, in the **Overlapping Inventory Filter** section, choose what action the importer should take if inventory for the same server(s) is returned from another source as well as from ServiceNow.

A likely example is if you have some UNIX-like servers with a locally-installed FlexNet Inventory Agent returning inventory, while the same servers are included in inventory collected by ServiceNow. Also keep in mind that there may be large differences in the default schedules for different inventory sources: for example, FlexNet inventory may be collected daily, and ServiceNow inventory may be imported only once every 7 days. For reasons like these, a typical choice for the ServiceNow inventory connector is **Import the inventory from this source for possible merging**, where you may also have inventory incoming from locally-installed FlexNet Inventory Agents, and you have given the IT Asset Management inventory source a higher priority – this allows it to improve the inventory gathered from partitioned technologies on UNIX-like platforms.

7. When all data is complete, click **Test connection**.

You cannot save the connection details if the connection test fails. Either:

- If the **Test connection** fails and displays an error message:
  - a. Click **OK** to close the message.
  - b. Review and correct the connection details.
  - c. Retest the connection (and repeat until your changes are successful, in which case click **OK** and resume from the next step).
  - d. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details, and seek further assistance.
- If the inventory beacon can successfully access the MID server folder using the details supplied, a **Test connection succeeded** message displays. Click **OK** to close the message, so that your new connector is displayed in the underlying list of connectors.

8. When your connection test is successful, select your new connection from the displayed list, and click **Schedule...**

9. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.

10. At the bottom of the **FlexNet Beacon** interface, click **Save**, and if you are done, also click **Exit**. Your work here is done.



#### **To validate your configuration:**

11. Be sure to wait until after the next scheduled execution of the ServiceNow export through the Flexera Integration application.

12. On the ServiceNow MID server, use the File Explorer to check the contents of the output folder (by default, the export folder under the MID Server software installed directory, although you may have customized this location).

You should see three .csv files (these file are not cleaned up between exports, and are over-written next time):

- flexera\_computer\_export.csv
- flexera\_software\_export.csv
- flexera\_software\_install\_export\_set.csv.

The presence of these well-formed .csv files confirms that all the configuration elements on the ServiceNow side are correct.

13. After the overnight inventory import and license consumption calculations, navigate in IT Asset Management to the **All Inventory** page, and apply a simple filter to the **Last inventory source** column, where the column contains ServiceNow.

The first-time appearance of even one entry with that value means that the export from ServiceNow, the collection by the inventory connector, the upload from the inventory beacon to the application server, the import, and the compliance calculations have all succeeded. Remember that some servers which were in fact discovered by ServiceNow may not be visible in your filtered list – if they were also inventoried by a higher-priority inventory source (such as a locally-installed FlexNet Inventory Agent), they will be listed against that higher-priority source (and may, for example, display IT Asset Management).

4


# Appendices: Data Flows

IT Asset Management (Cloud)

The following appendices are for those wanting a deep technical dive into the properties extracted from ServiceNow and imported as inventory into IT Asset Management.

The Flexera Integration application (for configuration details, see [Setting up the Integration Application](#)) is scheduled to produce three .csv files in a folder on your chosen MID server (see [Setting Up a MID Server](#), where you may have customized the folder, or used the default path of export under the ServiceNow MID Server installation directory).

Each of these tasks runs independently, and each is documented in a separate topic below.

 **Important:** *The outputs depend on which ServiceNow plugins you are using:*

- *Discovery plugin – data is imported only when discovered by ServiceNow, so this one is mandatory; and it is also a prerequisite for the following:*
- *SAM plugin – this plugin provides additional information in the file flexera\_software\_install\_export\_set.csv, and without this additional plugin, that .csv file remains empty.*

*It is also important to know that the connector validates data availability from all of the following output files, cross-referencing to ensure that no devices "slip through the cracks", even in corner cases such as the time interval between when the SAM plugin was enabled and when the next ServiceNow discovery (and data export) was run.*

This table summarizes the individual tasks (as they are listed within ServiceNow), the source of properties exported by each task, and the .csv file that saves the properties. In the following topics, the individual properties as listed in the .csv files are identified, along with a mapping to the results in IT Asset Management.

ServiceNow Task/Source	.csv File
Flexera Scheduled Computer Export (independent of plugins)	
cmdb_ci_computer.list	flexera_computer_export.csv
Flexera Scheduled Software Export (when only the Discovery plugin is enabled)	
cmdb_software_instance.list	flexera_software_export.csv
Flexera Scheduled Software Install Export (when the SAM plugin is also enabled)	

ServiceNow Task/Source	.csv File
cmdb_sam_sw_install.list	flexera_software_install_export_set.csv

For more information, see the following topics (in the same order as the table above):

- [Imported Computer Property Mapping](#)
- [Imported Installer Evidence \(with Discovery only\)](#)
- [Imported Installer Evidence \(with SAM\)](#)

## Imported Installer Evidence (with Discovery only)

IT Asset Management (Cloud)

These details apply when *only* the Discovery plugin is used with ServiceNow (if the SAM plugin is also in use, see next topic).

- ServiceNow scheduled task: Flexera Scheduled Software Export
- ServiceNow source list: cmdb\_software\_instance.list
- Saved file name: flexera\_software\_export.csv (properties from this file listed alphabetically below)
- Staging table in IT Asset Management: ImportedInstallerEvidence
- Sample locations in web interface: **Installer Evidence** tab of the **All Evidence** page; application properties on the **Evidence** tab

ServiceNow property	FNMS imported property	FNMS presentation/Notes
cicomp_sys_id	ExternalID (after processing through ImportedStringMapping)	Not visible in listings. Inventory device properties <b>General</b> tab > <b>Machine ID</b> (allows linking evidence to the individual device).
cispkg_manufacturer	Publisher	<b>Publisher</b>
cispkg_name	DisplayName	<b>Name</b>
cispkg_sys_id	ExternalInstallerID	Not visible in listings or properties: used internally for linking installer evidence to individual devices.
cispkg_version	Version	<b>Version</b>

# Imported Computer Property Mapping

## IT Asset Management (Cloud)

- ServiceNow scheduled task: Flexera Scheduled Computer Export
- ServiceNow source list: cmdb\_ci\_computer.list
- Saved file name: flexera\_computer\_export.csv (properties from this file listed alphabetically below)
- Staging table in IT Asset Management: ImportedComputer
- Sample locations in web interface: **All Inventory**, inventory device properties

ServiceNow property	FNMS imported property	FNMS presentation/Notes
cicomp_assigned_to	LastLoggedOnUser	<b>Last logged on user</b>
cicomp_cpu_core_count	NumberOfCores	<b>Cores</b>
cicomp_cpu_core_thread	NumberOfLogicalProcessors	<b>Processors</b>
cicomp_cpu_count	NumberOfProcessors	<b>Processors</b>
cicomp_cpu_name	ProcessorType (If cicomp_cpu_name is empty,take the value from cicomp_cpu_type.)	<b>Processor type</b>
cicomp_cpu_speed	MaxClockSpeed	<b>Clock speed (MHz)</b>
cicomp_cpu_type	ProcessorType (Only used if cicomp_cpu_name is empty.)	<b>Processor type</b>
cicomp_discovery_source	InventoryAgent	<b>Last inventory source</b> (displays the value ServiceNow for this import)
cicomp_disk_space	TotalDiskSpace	<b>Disk (GB)</b>
cicomp_dns_domain	Domain	<b>Domain name</b>
cicomp_last_discovered	InventoryDate	<b>Last inventory date</b>
cicomp_manufacturer	Manufacturer	<b>Manufacturer</b>
cicomp_model_id	ModelNo	<b>Model</b>
cicomp_name	ComputerName	<b>Name</b>
cicomp_os	OperatingSystem	<b>Operating system</b>
cicomp_os_service_pack	ServicePack	<b>Service pack</b>
cicomp_ram	TotalMemory	<b>RAM (GB)</b>

ServiceNow property	FNMS imported property	FNMS presentation/Notes
cicomp_serial_number	SerialNo	<b>Serial number</b>
cicomp_sys_id	ExternalID (after processing through ImportedStringMapping)	Not visible in listings. Inventory device properties <b>General</b> tab > <b>Machine ID</b>

## Imported Installer Evidence (with SAM)

IT Asset Management (Cloud)

These details apply when *both* the Discovery plugin and the SAM plugin are used with ServiceNow.

- ServiceNow scheduled task: 3) Flexera Scheduled Software Install Export
- ServiceNow source list: cmdb\_sam\_sw\_install.list
- Saved file name: flexera\_software\_install\_export\_set.csv (properties from this file listed alphabetically below)
- Staging table in IT Asset Management: ImportedInstallerEvidence
- Sample locations in web interface: **Installer Evidence** tab of the **All Evidence** page; application properties on the **Evidence** tab

ServiceNow property	FNMS imported property	FNMS presentation/Notes
cicomp_sys_id	ExternalID (after processing through ImportedStringMapping)	Not visible in listings. Inventory device properties <b>General</b> tab > <b>Machine ID</b> (allows linking evidence to the individual device).
swinstall_display_name	DisplayName	<b>Name</b>
swinstall_install_date	InstallDate (in the ImportedInstalledInstallerEvidence table)	Not directly visible in evidence listing. Applies to individual devices and visible in the <b>Applications</b> tab of inventory device properties.
swinstall_publisher	Publisher	<b>Publisher</b>
swinstall_sys_id	ExternalInstallerID	Not visible in listings or properties: used internally for linking installer evidence to individual devices.
swinstall_version	Version	<b>Version</b>

# XIX

## Tanium Adapter

### IT Asset Management (Cloud)

This part introduces the Tanium adapter, provided by Flexera, which allows you to export inventory data from Tanium into IT Asset Management using both Credential based authentication as well as Token based authentication. Tanium Asset collects a complete inventory of your hardware and software assets, including servers, laptops, and desktops.


IT Asset Management uses the software and hardware inventory data collected from Tanium Asset to calculate license consumption. As always, the inventory records must be recognized by the Application Recognition Library, and you must have the resulting application records linked to the appropriate license, for compliance calculations to proceed.

### Supported versions

The Tanium adapter supports inventory import from the following releases of Tanium:

- Tanium Server version 7.2 or later
- Tanium Asset versions:
  - For the adapter included in the inventory beacon released with IT Asset Management 2020 R1.1 (internal version number 15.1.0) or later, supported versions of Tanium Asset are 1.8.0.0075–1.28.187.
  - If you have an *earlier* release of Tanium Asset (from release 1.6.3) in use and cannot consider upgrading it, you must continue to use the adapter version installed on an inventory beacon version 15.0.0 (or earlier), available with the previous IT Asset Management 2020 R1 release. Do not upgrade this inventory beacon until you also upgrade Tanium Asset.

---

 **Warning:** Upgrading the inventory beacon that hosts the Tanium adapter also upgrades the adapter itself. Switching to the new adapter (on 15.1.0 or later) with an older version of Tanium Asset (before 1.8.0.0075) will cause recognition of Microsoft SQL Server to fail. The product change made to Tanium Asset at that version, reporting SQL Server data in different ways, requires that you must match either old adapter with old Tanium Asset, or new adapter with new Tanium Asset.



# 1

## Prerequisites

IT Asset Management (Cloud)

The following topics provide the software prerequisites that are required and the configuration steps that must be undertaken **before** you can create a connection to Tanium.

- [FlexNet Beacon Server Prerequisites](#)
- [Tanium Server Prerequisites](#)
- [Creating a Flexera SQL Edition view within Tanium Asset](#)

## FlexNet Beacon Server Prerequisites

IT Asset Management (Cloud)

### FlexNet Beacon software prerequisites

The following FlexNet Beacon server prerequisites are required for the Tanium adapter:

- An inventory beacon running the appropriate release of FlexNet Beacon software matched to your version of Tanium Asset (which fundamentally changed the way its API reports details of Microsoft SQL Server):
  - For versions of Tanium Asset *earlier* than 1.8.0.0075, continue to use an inventory beacon running a version of FlexNet Beacon in the range of 14.2 (shipped with 2019 R2.2) to 15.0.0 (shipped with 2020 R1), and do not allow an upgrade of that inventory beacon until you also upgrade Tanium Asset to a later version.
  - For versions of Tanium Asset from 1.8.0.0075, you must use an inventory beacon running FlexNet Beacon 15.1.0 or later (shipped with 2020 R1.1). This brings an updated Tanium adapter that works with the changed API to collect information about Microsoft SQL Server.
- PowerShell 5.1 or later
- A web browser that can access IT Asset Management
- The ability to login to the inventory beacon, and run FlexNet Beacon, using an account with administrator privileges.

# Tanium Server Prerequisites

IT Asset Management (Cloud)

## Tanium server software prerequisites

The following Tanium server software prerequisites are required to use the Tanium adapter:

- An operational Tanium instance
- Tanium Server version 7.2 or later
- Tanium Asset version 1.8.0.0075–1.28.187 (for use with FlexNet Beacon version 15.1.0 or later – if you have an earlier version of Tanium Asset, also preserve an earlier installation of FlexNet Beacon with the earlier adapter matching the previous API of Tanium Asset, as described in [FlexNet Beacon Server Prerequisites](#))
- A web browser
- The ability to log into Tanium web interface, using an account with administrator privileges.



**Note:** The Tanium adapter leverages the REST APIs provided by Tanium to export asset inventory data from Tanium Asset and import it into IT Asset Management.



**Important:** To ensure recognition of Microsoft SQL Server from Tanium Asset v1.8.0.0075 and later, you will need to use the Tanium adapter on FlexNet Beacon 15.1.0 (or later), shipped with IT Asset Management 2020 R1.1.

## Tanium server configuration prerequisites



**Important:** For Tanium Asset version 1.11 or later (running on Tanium Server 7.3.0 or later), these configuration prerequisites are not required. Instead, skip ahead directly to [Creating a Flexera SQL Edition view within Tanium Asset](#).

For earlier versions of Tanium Asset, complete all of the following configuration tutorials which are documented in the Tanium Asset User Guide on the <https://docs.tanium.com/asset/asset/exportasset.html?Highlight=flexera> page:

1. "Configure Flexera staging database"



**Tip:** You must configure your staging database to allow a Flexera destination in Tanium. Although the Tanium adapter will not use the staging database to import inventory data, the staging database is required to add a Flexera destination in Tanium.

2. "Add Flexera destination"
3. "(Optional) Enable file evidence content"

Once the above configurations are complete, you are ready to create a Flexera SQL Edition view. See [Creating a Flexera SQL Edition view within Tanium Asset](#).

## Creating a Flexera SQL Edition view within Tanium

# Asset

IT Asset Management (Cloud)

You must manually create the Flexera SQL Edition view in Tanium Asset to enable the Tanium adapter to correctly recognize Microsoft SQL Server, and retrieve related evidence from Tanium for import into IT Asset Management.



**To create the Flexera SQL Edition view in Tanium Asset:**

1. Login to Tanium Asset.



**Tip:** As a shortcut, you can, from the Asset menu, click **Views > Create Flexera Views**, and then check for the *Flexera SQL Edition* view in the list of **Asset Views**. If you choose to do this, skip the rest of this process. Otherwise, to specify your own view, continue below.

2. Expand the left-hand menu and navigate to **Views**.

The list of existing destinations displays.

3. Click **New View**.

The **Create View** page displays.

4. In the **View Name** field, enter Flexera SQL Edition.

This exact name must be entered.

5. In the **Description** field, you can enter any description.

6. In the **View Attributes** section, click the **Select None** link to deselected all of the check boxes and then select the following attributes:

- **Computer Name**
- The following attributes which are in the **Asset Internal** section:
  - **Asset Id**
  - **Created At**
  - **Updated At**
- The following attributes which are in the **Asset SQL Server Details** section:
  - **SQLServerEdition**
  - **SQLProductVersion**
  - **SQLProductLevel**
  - **SQLDisplayName**
  - **SQLInstanceName**

7. Click **Create View**.

The Flexera SQL Edition view displays in the list of **Asset Views**.

## 2

# Configure FlexNet Beacon to Connect with Tanium

IT Asset Management (Cloud)



**To configure FlexNet Beacon to connect with Tanium:**

1. Log into the inventory beacon interface as an administrator (for example, in the Windows Start menu, search for **FlexNet Beacon**, right-click it, and select **Run as administrator**).



**Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.

2. Select the **Inventory systems** page and click the down arrow on the right of the **New...** split button, and choose **PowerShell**.



**Tip:** Alternatively, you can edit a connection you have defined previously, by selecting it from the list of connections and clicking **Edit....**

3. In the dialog that appears, complete (or modify) the following required fields:
  - **Connection Name:** The name you give this inventory connection is also used in the web interface of IT Asset Management to name the data import task. The name may contain alphanumeric characters, underscores or spaces, but it must start with either a letter or a number, for example **Tanium**.
  - **Source Type:** Select **Tanium** from the drop-down list.
4. Optionally, if your enterprise uses a proxy server to enable communication between FlexNet Beacon and Tanium, select the **Use Proxy** check box, and complete the following additional details:
  - **Proxy Server:** Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format `https://ProxyServerURL:PortNumber`, `http://ProxyServerURL:PortNumber`, or `IPAddress:PortNumber`. If the protocol is omitted, it defaults to `http:`. If the port number is omitted, it defaults to `:80` for `http`, or `443` for `https`.
  - **Username and Password:** If your enterprise is using an authenticated proxy, specify the credentials to access

the proxy server you just identified.

5. In the **Tanium** section, complete the following fields:

#### ApiToken

- a. **Tanium Server:** Enter the address of your Tanium Cloud instance, for example, [flexerapartner-api.cloud.tanium.com](https://flexerapartner-api.cloud.tanium.com).

Do not include the `http://` or `https://` protocol with the server address.

- b. **Token:** Provide the API Token for the Tanium instance.



**Note:** The **Refresh** button, when selected, will only refresh the existing token with a new token if it is active. If the new token has expired, then selecting **Refresh** will not refresh the token. In this case, you will need to go to the Tanium site and reactivate the expired token or create a new token.

6. Click **Test Connection** to make sure that your specifications are correct (adjusting as necessary for success).

When FlexNet Beacon has successfully accessed the Tanium APIs using the details supplied, a **Test connection succeeded** message displays.

7. Click **OK** to close this dialog, and you may also click **Save**.

The new connection is added to the connections list in the **Inventory Systems** tab.

8. If you do not already have a suitable schedule for this connection available on this inventory beacon, select the **Scheduling** page and click **New...** to open the **Edit Schedule** dialog and define one.

For more details see *Scheduling a Connection* in the online help.

9. Back on the **Inventory systems** page, select the new connection in the list, click **Schedule...** to select your appropriate schedule in the **Select schedule** dialog and then click **OK** to confirm your selection.

10. In the main **FlexNet Beacon** interface, click **Save** to store your newly scheduled connection.

11. Select the newly-added connection and click **Execute Now**.

The **Execute Now** confirmation dialog displays and the import of Tanium inventory data commences.

12. Click **OK** to confirm.

The resulting import (which follows immediately) may take some time to complete. You may monitor progress in the web interface for IT Asset Management — go to the **System Tasks** page (**Data Collection > IT Assets Inventory Status > System Tasks**), and periodically refresh this page. When the process is complete, you can inspect the inventory imported from Tanium in the **All Inventory** page on IT Asset Management.



**Tip:** In the **All Inventory** page:

1. Add the **Connection name** column to the page.
2. Display the simple filter bar.
3. Filter for the name you have your Tanium connection (the example given was *Tanium*).

After a successful data import, the hardware and software inventory data from Tanium Asset are visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** page of FlexNet Beacon, see *Inventory Systems Page* in the online help. For scheduling data imports through this connection, see *Scheduling a Connection*, also in online help.

## 3

# Validating Imports to the Inventory Beacon

IT Asset Management (Cloud)

This process traces the path of data imported from Tanium through an inventory beacon, to the central application server of IT Asset Management. You may start this process at whichever stage is suitable to your needs.



## **To validate imports to the inventory beacon:**

1. On the inventory beacon that connects to Tanium, log into FlexNet Beacon using an account with administrator privileges.
2. On the **Inventory Systems** page, select your connection to Tanium, and click **Execute Now**.  
A confirmation shows that the extract has started.
3. In the top-left corner of the FlexNet Beacon interface, right-click the menu icon, and select **Open log file folder**.  
Windows File Explorer opens.
4. Ensure that you are viewing `ProgramData\Flexera Software\Compliance\Logging\` and then navigate to the `ComplianceReader` folder.
5. In the text editor of your choice, open the text document `importer-[nnnn]` that was modified on today's date, and scroll to the bottom to determine whether data reading is complete. If not, exit the text editor, and try again in a few minutes.

A successful process shows the following at the end of the log file (with each line prefixed with the date and time):

```
[INFO] 0 source data warnings
```

```
[INFO] 0 errors, 0 warnings
```

```
[INFO] Import has been completed successfully
```

If there are warnings or errors in the log file, try to correct the issues. If you are unable to correct the error, save the log file and attach it to a support ticket. Typical issues at this stage may include:

- Proxy or firewall settings preventing access to the Internet
- Intermittent issues with the Tanium server
- Incorrect details for the account name or password accessing Tanium
- The connection has not been regularly scheduled and so succeeds only when executed manually.



**Note:** Immediately after a successful import is completed, the inventory beacon attempts to upload the archive of imported data to its parent (the parent may be another inventory beacon in the hierarchy, or the application server):

- When the upload succeeds, the inventory beacon saves a copy in your default drive at `<Drive>:\ProgramData\Flexera Software\Beacon\IntermediateData\Uploaded` and preserves this copy for 14 days
- If the upload fails, the imported archive stays in `<Drive>:\ProgramData\Flexera Software\Beacon\IntermediateData`.



**Tip:** If you are working on a disconnected inventory beacon (one that cannot access the application server), this `IntermediateData` folder contains the data files that you must transfer to another location where uploads are possible.

You may now want to review or debug the imported data. To review the data imported from Tanium:

- Navigate to `<Drive>:\ProgramData\Flexera Software\Beacon\IntermediateData\Uploaded`, and inspect the most recent relevant .zip file there. If the zip archive remains in the `IntermediateData` folder (on a connected inventory beacon), there is an upload problem, as discussed below.

#### **To debug uploads from this inventory beacon to its parent:**

- Review `C:\ProgramData\Flexera Software\Compliance\Logging\ComplianceUpload\upload`. Log for details of any networking issues and note the URL of the next server in the upload chain. If this is another inventory beacon, repeat these reviews there, and so on until you exhaust the hierarchy of inventory beacons.

The upload URL is in a log line that starts with `dateTime [Upload.UploadProcess ] [INFO] Uploading to http`

If you are unable to resolve problems, collect your logs together in a zip archive. Ask your registered support contact (a designated person within your enterprise who has access rights and login details) to open a new support case at <https://community.flexera.com/t5/forums/postpage/board-id/@support>, including a clear description of the issue.

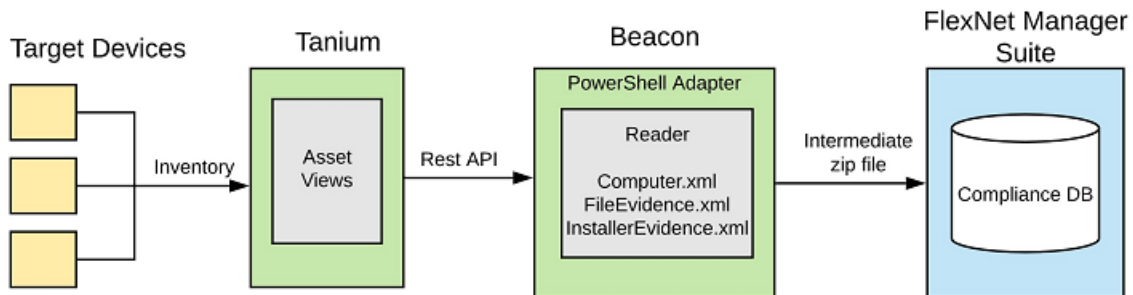


## 4

# Tanium Adapter Architecture

IT Asset Management (Cloud)

This topic provides a useful framework for understanding the workflow of the Tanium adapter in more detail.



The above diagram shows the workflow for the adapter:

## Tanium

- Creating the Flexera Destination creates most of the required views and reports automatically
- Tanium views named as Flexera \* are used for importing data from Tanium Asset to IT Asset Management Imported\* tables, with the data later moved into the compliance database tables.
- The following Tanium views used for importing data:
  - Flexera Computer — contains Computer, User, Domain and WMI evidence data
  - Flexera Software Installations — contains installer evidence data
  - Flexera File Evidence — contains file evidence data
  - Flexera Network Adapter — contains IP address and mac address data for each computer
  - Flexera SQL Edition — contains SQL-related installer evidence data.



**Note:** You must create this view manually, if the view does not exist. See [Creating a Flexera SQL Edition view within Tanium Asset](#) for instructions.



**Tip:** Flexera reports in Tanium Asset enable crosschecking of the data between Tanium and IT Asset Management once the import is complete.

### Beacon

- The inventory beacon connects to Tanium to perform the inventory upload
- Data is collected from the above-mentioned views through REST API calls, and the inventory beacon creates an intermediate zip file.

### IT Asset Management

- The intermediate zip file is pushed from the inventory beacon hierarchy to IT Asset Management, and the database writers move the uploaded data into the compliance database).
- Once data is imported, this inventory is matched to the Application Recognition Library, and license compliance is calculated.

## 5

# Investigating Tanium Connection Errors

## IT Asset Management (Cloud)

You can troubleshoot FlexNet Beacon failures or errors by reviewing the error messages that display in the FlexNet Beacon web interface, or by looking in the log files for any failures, errors or warning messages. If you are unable to resolve problems, collect your logs together in a zip archive. Ask your registered support contact (a designated person within your enterprise who has access rights and login details) to open a new support case at <https://community.flexera.com/t5/forums/postpage/board-id/@support>, including a clear description of the issue. The following table provides help with potential issues that you may encounter when attempting to connect to Tanium.

Error, Failure or Message	Description
Value is required	<p>One or more mandatory fields are missing a value in the <b>PowerShell Source Connection</b> dialog when configuring the adapter on FlexNet Beacon.</p> <p>You can resolve this error by ensuring all mandatory field are completed and retesting and save the connection. For details, see <a href="#">Configure FlexNet Beacon to Connect with Tanium</a>.</p>
Value is required	<p>One or more mandatory fields are missing a value in the <b>PowerShell Source Connection</b> dialog when configuring the adapter on FlexNet Beacon.</p> <p>You can resolve this error by ensuring all mandatory field are completed and retesting and save the connection. For details, see <a href="#">Configure FlexNet Beacon to Connect with Tanium</a>.</p>

Error, Failure or Message	Description
View missing: Flexera SQL Edition	<p>If the Flexera SQL Edition view is missing, this message displays when you test the connection between in the FlexNet Beacon and the Tanium server.</p> <p>You can resolve this error by creating the Flexera SQL Edition view. See <a href="#">Creating a Flexera SQL Edition view within Tanium Asset</a>.</p>
The remote name could not be resolved: '<[fully qualified domain name of Tanium server] of the Tanium server>'	<p>This message is most likely to occur because there is no Internet connection available to the location where the inventory beacon is installed.</p> <p>It may also occur for the following reasons:</p> <ul style="list-style-type: none"> <li>• The remote name could not be resolved</li> <li>• The Tanium server is blocked by a firewall or server is currently down</li> <li>• The fully qualified domain name of the Tanium server and the IP address have not been added in the host file.</li> </ul>
Unable to connect to the remote server	<p>This message may appear when <code>http://</code> or <code>https://</code> is included as part of the IP address to the Tanium Server. You can resolve this error by removing the <code>http://</code> or <code>https://</code> protocol from the <b>Tanium Server</b> field. For details, see <a href="#">Configure FlexNet Beacon to Connect with Tanium</a>.</p>

6

# Appendix: Technical Data

IT Asset Management (Cloud)

The following information is not required for set-up or operation of the Tanium adapter.

The connection to Tanium lists imported hardware and software inventory. Details of imported data can be viewed on the **All Inventory** page by filtering results by the **Connection name** you chose for your connection. You can then select the **Name** to view full list of properties that have been imported for a single inventory device.

In the inventory device properties, imported inventory data displays in the **General**, **Hardware**, **Applications**, and **Evidence** tabs as appropriate. See online help for full details.

Hardware inventory data Imported by Tanium Adapter

An inventory device record is automatically created for your hardware inventory (with the resulting property sheet). The following properties (listed alphabetically) are collected for hardware inventory through the Tanium Adapter.

Tanium View Name	Columns	FNMS Table Name/Column Name
Flexera Computer	Id	
	asset_ad_short_domain	ImportedDomain /FlatName
	asset_last_logged_in_user	ImportedComputer/ LastLoggedOnUser ImportedUser/SAMAccountName ImportedUser/Domain
	asset_number_of_cpu_socket	ImportedComputer/ NumberOfSockets
	chassis_type	ImportedComputer/ChassisType
	computer_name	ImportedComputer/ComputerName
	cpu_core	ImportedComputer/NumberOfCores
	cpu_name	ImportedComputer/ProcessorType

Tanium View Name	Columns	FNMS Table Name/Column Name
	cpu_processor	ImportedComputer/ NumberOfProcessors
	cpu_speed	ImportedComputer/MaxClockSpeed
	disk_total_space	ImportedComputer/TotalDiskSpace
	display_adapter_count	ImportedComputer/ NumberOfDisplayAdapters
	domain_name	ImportedDomain/QualifiedName
	manufacturer	ImportedComputer/Manufacturer
	model	ImportedComputer/ModelNo
	number_of_fixed_drive	ImportedComputer/ NumberOfHardDrives
	number_of_logical_processor	ImportedComputer/ NumberOfLogicalProcessors
	operating_system	ImportedComputer/ OperatingSystem
	os_platform	ImportedWMIEvidence /ClassName
	serial_number	ImportedComputer/SerialNo
	service_pack	ImportedComputer/ServicePack
	system_uuid	ImportedComputer/UUID
	ram	ImportedComputer/TotalMemory
	updated_at	ImportedComputer/ ServicesInventoryDate
<b>Flexera Network Adapter</b>	<b>Id</b>	
	ipv4_address	ImportedComputer/IPAddress
	mac_address	ImportedComputer/MACAddress
<b>Flexera File Evidence</b>	<b>Id</b>	
	company	ImportedFileEvidence/Company
	description	ImportedFileEvidence/Description
	file_name	ImportedFileEvidence/FileName
	file_path	ImportedFileEvidence/FilePath
	file_size	ImportedFileEvidence/FileSize
	file_version	ImportedFileEvidence/FileVersion
	language	ImportedFileEvidence/Language

Tanium View Name	Columns	FNMS Table Name/Column Name
	product_name	ImportedFileEvidence/ProductName
	product_version	ImportedFileEvidence/ ProductVersion
<b>Flexera Software Installations</b>	<b>Id</b>	
	ci_installed_application_name	ImportedInstallerEvidence/ DisplayName
	ci_installed_application_source	ImportedInstallerEvidence/Evidence
	ci_installed_application_version	ImportedInstallerEvidence/Version
	ci_installed_application_vendor	ImportedInstallerEvidence/Publisher
<b>Flexera SQL Edition</b>	<b>Id</b>	
	ci_asset_sql_server_sqldisplayname	ImportedInstallerEvidence/ DisplayName
	ci_asset_sql_server_sqlproductversion	ImportedInstallerEvidence/Version
	ci_asset_sql_server_sqlserveredition	ImportedInstallerEvidence/ DisplayName



# VMware Horizon Adapter

IT Asset Management (Cloud)

The **VMware Horizon** virtual desktop infrastructure (VDI) adapter, provided by Flexera, allows you to collect data from VMware Horizon non-persistent VDIs and import it into IT Asset Management.

Imported VDI data and application evidence collected by the FlexNet Inventory Agent will give you full IT visibility of inventory running on VMware Horizon platform, and provide licensing capabilities for all applications used by end-users on non-persistent VDIs.



**Note:** Virtual desktop infrastructure (VDI) is a technology that refers to the use of virtual machines to provide and manage virtual desktops. VDI hosts desktop environments on a centralized server and deploys them to end-users on request.

## The difference between non-persistent and persistent VDIs

Once a user logs off a virtual machine (VM), that VM and associated data is destroyed. This is referred to as a non-persistent VDI. Any changes made on the VM will not persist when the VM is destroyed. Because of this, non-persistent VMs are very difficult to license. This is the problem that the VDI adapters provided by Flexera solve.

Unlike non-persistent VMs, persistent VMs are not destroyed once a user logs off. Persistent VMs are similar to that of a standard VM, and when set up, administrators can give access to groups/users who can continue to use it until the administrator deletes it.

## What will you see in the UI

In IT Asset Management, you can view a VDI template for each gold image that has been configured in VMware Horizon. This template allows you to see the full list of applications on all VDIs deployed from that template.



**Tip:** The "VDI template" terminology is used in the IT Asset Management UI. For customers not familiar with this terminology, the "VDI template" can also be referred to as the "golden image virtual machine".

For each user who has access to any of the VDIs that were deployed from a VDI template, the application evidence is mapped over to any inventory device for which that user is the primary user.

For any end-user who can access a VDI template who is not a primary user of any inventory device, a remote device will be created for them, and the application evidence will be mapped.



## Supported versions

- VMware Horizon version 8.0.0 or higher, if you are not using Cloud Pod Architecture
- If you are using Cloud Pod Architecture, version 2012 or higher
- The VMware Horizon adapter is supported by version 2022 R1 of IT Asset Management onwards.

## What's not supported

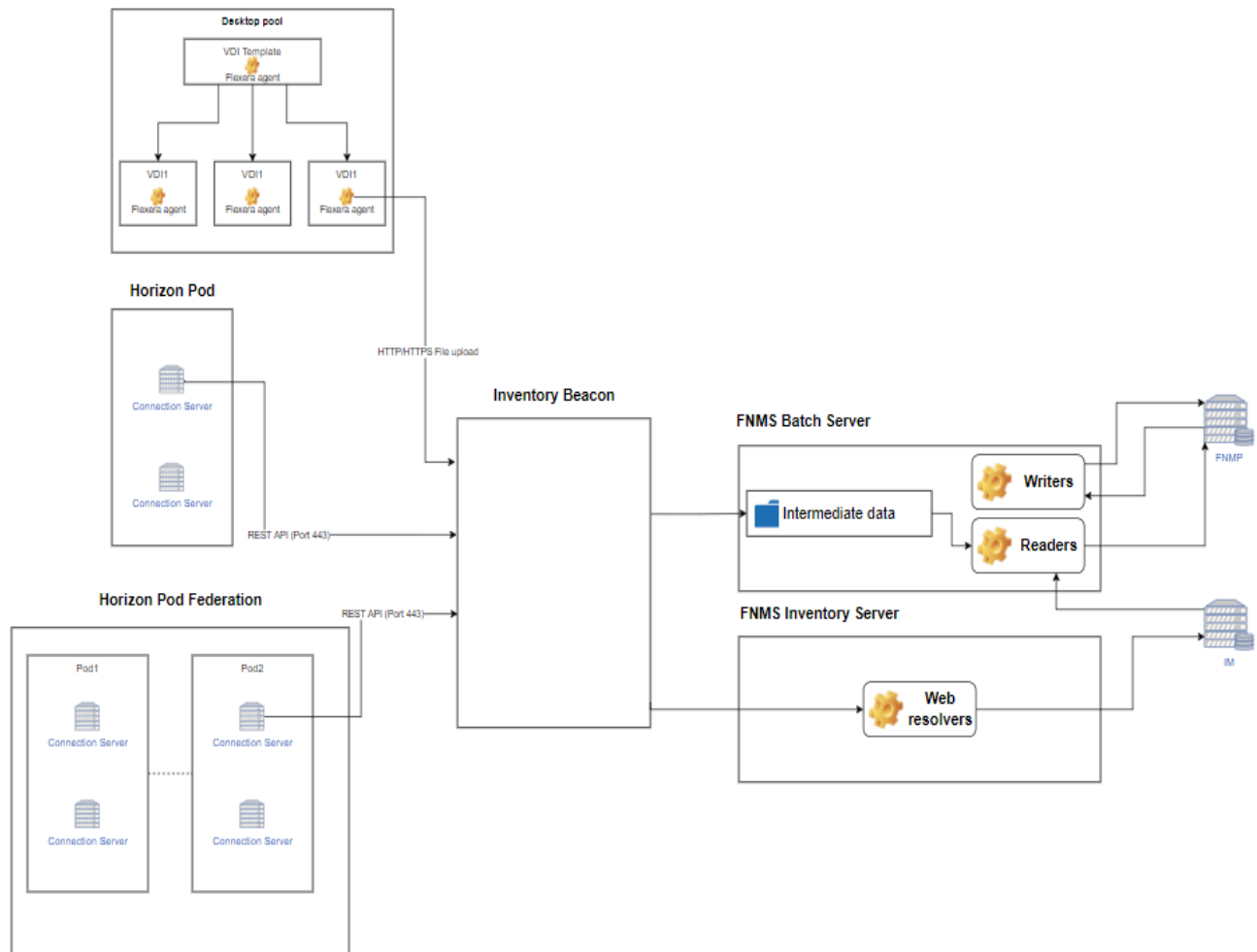
- Layered Images in VMware Horizon is currently not supported. This technology allows you to dynamically assign applications to a specific VDI instance outside of the VDI template.

## 1

# Architecture and Operation

## IT Asset Management (Cloud)

The following diagram shows the operational architecture for the VMware Horizon adapter.



## Summary

The VMware Horizon adapter has been created to collect supplementary VDI data. This data will show:

- Existing VDI devices and templates
- Existing desktop pools in Horizon where these VDI devices and templates are installed
- What users have access to these desktop pools.

The FlexNet Inventory Agent which is installed on the VDI template, collects application evidence from each of the VDI devices purported by the VMware Horizon adapter. This application evidence shows all of the software that end-users have access to.

To import the collected supplementary VDI data into IT Asset Management, the VMware Horizon adapter uses PowerShell to query the REST APIs available on each connection server. The connection server acts as a broker and is the main component that fetches the virtual desktop or application(s) and delivers it to the end-user.

VMware Horizon documentation pertaining to the REST API used for gathering application evidence on the connection server is available [here](#).

There are 5 main components in the above diagram:

- **Desktop pool:** A collection of existing virtual machines. The FlexNet Inventory Agent collects the application evidence from these machines which is then mapped to users who have access to that desktop pool. Note: Access to a desktop pool is defined in Active Directory.
- **Pod / Pod Federation:**
  - A Pod is a collection of existing connection servers. The connection server in VMware Horizon acts as the broker and is the main component that fetches the virtual desktop or application(s) and delivers it to the end-user. The connection server verifies what each user can access by checking the group and user permissions defined in Active Directory. To be able to pull the data into IT Asset Management, the connection server is queried by the VMware Horizon adapter which is set up on the inventory beacon. The VMware Horizon adapter then collects the information needed to represent the VMware Horizon inventory in IT Asset Management.
  - A Pod Federation is a collection of existing connection server pods.
- **Inventory Beacon:** Connects to a single connection server in each pod, or in the case of a pod federation a single connection server in that federation. Inventory is then uploaded to the Batch and Inventory Servers. The inventory beacon also imports data from Active Directory, including groups (and their members), users, and computers, and the security identifiers for each item within Active Directory. (These security identifiers, or SIDs, are the same identifiers that the VMware Horizon adapter reports for usage of the applications delivered by VMware Horizon).
- **Inventory Server:** Is where the application evidence (.NDI file from each VDI device) is received, processed and imported to the IM inventory database. .NDI files are produced by running the FlexNet Inventory Agent on the VDI.
- **Batch Server:** Is where data from the IM Inventory Database is processed and imported to the IT Asset Management Compliance database which in turn drives the VDI template UI. Note: The VMware Horizon adapter has been configured as a new compliance connection. VDI data is sent to the Batch server as intermediate data files which are then processed (matched/merged) with data from other compliance connections to produce a single view of the data and imported to the IT Asset Management database.

## What data is retrieved

The data listed below is retrieved by means of running functions in the PowerShell reader that is used to connect to the VMware Horizon REST API.

Functions	Retrieved data
Site name	<p>Returns a string that represents the site name associated with the data from the connection server. If Cloud Pod Architecture (CPA) is in use, the name of the Pod Federation is used: (/rest/federation/v1/cpa - name).</p> <p>If CPA is not in use, the cluster name which represents a group of connection servers sharing the same configuration is used: (/rest/config/v1/environment-properties - cluster_name).</p>
Desktop pools	<p>Returns each desktop pool along with the following properties for each pool.</p> <p>/inventory/v2/desktop-pools - source</p> <p>/inventory/v2/desktop-pools - provisioning_settings - base_snapshot_id</p> <p>/rest/inventory/v2/desktop-pools - name</p> <p>/rest/inventory/v2/desktop-pools - id</p>
Machines	<p>Returns a list of VDI's associated with a desktop pool or delivery group and the corresponding properties for that VDI.</p> <p>/rest/inventory/v1/machines - name</p> <p>/rest/inventory/v1/machines - dns_name</p> <p>/rest/inventory/v1/machines - desktop_pool_id</p>
User access	<p>Returns the Active Directory SID for a user or group that has access to a desktop pool.</p> <p>/rest/entitlements/v1/desktop-pools- ad_user_or_group_ids</p> <p>If CPA is in use: /entitlements/v1/global-desktop-entitlements</p>
Test connection	<p>A test connection button is available in the FlexNet Beacon UI. Selecting test connection will show a successful test if the configured user is able to successfully log into the API, going through any configured proxy.</p> <p>If the connection fails, the relevant error is fed back to the user.</p>

# 2

## Prerequisites

### IT Asset Management (Cloud)

The VMware Horizon adapter is available in the FlexNet Beacon UI from IT Asset Management 2022 R1 onwards.

The VMware Horizon adapter requires the following:

- VMware Horizon version 8.0.0 or higher.
- VMware Horizon version 2012 or higher, if you are using cloud pod architecture.
- Create at least one desktop pool that will provide non-persistent VDI to users.
- Install the FlexNet Inventory Agent. It is recommended that you install the FlexNet Inventory Agent onto the VDI template to ensure inventory from at least one VDI device can be collected per desktop pool, unless another inventory source is providing software inventory from the VDI devices.



**Tip:** The "VDI template" terminology is used in the IT Asset Management UI. For customers not familiar with this terminology, the "VDI template" can also be referred to as the "golden image virtual machine".

- An inventory beacon (or multiple if required) that collects Active Directory data from the domain where the users are located who have been given access to a desktop pool.
- Configure a connection in the FlexNet Beacon UI to a single connection server in each pod, or in the case of a pod federation a single connection server in that federation. See [Creating the VMware Horizon Connection](#).
- PowerShell 5.0 or higher on the beacon where the VMware Horizon adapter is set up.

3


# Creating the VMware Horizon Connection

IT Asset Management (Cloud)

Use the following procedure to create a connection to VMware Horizon on the FlexNet Beacon. A connection is required to be configured for a single connection server in each pod, or in the case of a pod federation, a single connection server in that federation. The inventory beacon is responsible for uploading the data to the central operations databases of IT Asset Management.


 **To create the VMware Horizon connection in the FlexNet Beacon UI:**

1. Log into your selected inventory beacon.

 **Tip:** Starting the FlexNet Beacon interface requires that you are logged in with administrator privileges.


2. In the navigation pane on the left, select the **Inventory systems** page. To create a new connection, click the down arrow on the right of the **New** split button, and choose **Powershell**.

The **Create PowerShell Source Connection** dialog appears.

 **Tip:** The **New...** button defaults to creating a connection for Microsoft SQL Server. If you use the down arrow on the split button, you can choose between *SQL Server*, *Spreadsheet*, *PowerShell*, and *Other* connections. However, while you are creating a connection to a Microsoft SQL Server database (regardless of the **Source Type** of the connection), use only the *SQL Server* option.

3. Complete the values in the dialog, as follows:

Control	Comments
Connection Name	A descriptive name for this connection, such as Horizon VDI Data. The name may contain alphanumeric characters, underscores or spaces, but must start with either a letter or a number. When the data import through this connection is executed, the data import task name is same as the connection name.
Source Type	Select <b>Horizon</b> from the list.

Control	Comments
<b>Use Proxy</b>	Optionally, if you use a proxy server to enable Internet access, complete (or modify) the values in the <b>Proxy Settings</b> section of the dialog box in order to configure the proxy server connection.
<b>Proxy Server</b>	Enter the address of the proxy server using HTTP, HTTPS, or an IP address. Use the format <code>https://ProxyServerURL:PortNumber</code> , <code>http://ProxyServerURL:PortNumber</code> , or <code>IPAddress:PortNumber</code> . This field is enabled when the <b>Use Proxy</b> check box is selected.
<b>Username and Password</b>	If your enterprise is using an authenticated proxy, specify the <b>username</b> and <b>password</b> of an account that has credentials to access the proxy server that is specified in the <b>Proxy Server</b> field. These fields are enabled when the <b>Use Proxy</b> check box is selected.
<b>Horizon Connection Server</b>	<p>You need to provide the VMware Horizon connection server URL. The connection server is the broker (equal to the Citrix delivery controller). Importantly, you only need to configure a single connection server per pod.</p> <hr/> <p> <b>Note:</b> Essentially, you can load balance connection servers and all of the data will be replicated between each one. You only need to target a single connection server. Or if you are using a pod federation, you only need a single connection server in that pod federation to provide the results.</p> <p>Provide the VMware Horizon connection server URL using the format <code>https://ConnectionServerURL</code> or <code>https://IPAddress</code>.</p> <p>The internet protocol must be secure and begin at the start of the connection server URL, and consist only of the domain and no paths to specific pages. For example <code>https://ConnectionServerURL/admin</code> is incorrect.</p>
<b>Domain, Username and Password</b>	Enter a <b>domain</b> , <b>username</b> and <b>password</b> . Note: the account needs to have full read access to all objects in VMware Horizon.
<b>Disable certificate check</b>	This allows you to disable verifying the TLS certificate. This is turned off by default.

Control	Comments
<b>Connection is in test mode (do not import results)</b>	<p>Controls the uploading and importing of data from this connection:</p> <ul style="list-style-type: none"> <li>When this check box is clear, the connection is in production mode, and data collected through this adapter is uploaded to the central server and (in due course) imported into the database there.</li> <li>When the check box is set: <ul style="list-style-type: none"> <li>The adapter for this connection is exercised, with data written to the intermediate file in the staging folder on the inventory beacon (%CommonAppData%\Flexera Software\Beacon\IntermediateData)</li> <li>The immediate upload that normally follows data collection is suppressed, so that you can inspect the contents of the file</li> <li>The catch-up process that retries stalled uploads, normally scheduled overnight, runs as usual and uploads the file to the central server</li> <li>At the central server, the file contents are discarded (and not imported into the central database).</li> </ul> </li> </ul>
<b>Overlapping Inventory Filter</b>	This control does not apply to VMware Horizon adapter, and you may leave it at the default setting.

#### 4. Click **Test Connection**

This will make sure that you can successfully connect to both the endpoint and the specified **username** and **password** are correct. Note: a request to log into the API is part of the test connection.

- If the connection is successful, click **OK** to close the message. Click **Save** to complete the addition. The connection is added to (or updated in) the list.
- If the connection is unsuccessful, the appropriate error message will display. Click **OK** to close the message. Edit the connection details and retest the connection.  
You cannot save the connection details if the connection test fails. If you cannot get the connection test to succeed, click **Cancel** to cancel the addition of these connection details.

5. In the FlexNet Beacon PowerShell Source Connection dialog, click **Save** to save the connection.

6. Select your new connection from the displayed list, and click **Schedule....**

7. In the dialog that appears, select the name of your chosen schedule for inventory collection through this connection, and click **OK**.

8. At the bottom of the FlexNet Beacon interface, click **Save**, and if you are done, also click **Exit**.

After a successful data import, existing VDI devices and templates, existing desktop pools where these VDI devices and templates are installed and what user groups have access to these desktop pools are all visible in the appropriate pages of IT Asset Management.



**Note:** To know more about the operations available on the **Inventory Systems** page of FlexNet Beacon, see the *Inventory Systems Page* in the online help. For scheduling data imports through this connection, see *Scheduling a*



---

*Connection, also in the online help.*