Spider

Spider Data Collector User Manual v1.2202

Legal Information

Book Name:	Spider Data Collector 1.2202
Part Number:	SPI-0001-DC012
Product Release Date:	February 2022

Copyright Notice

Intellectual Property

Restricted Rights Legend

Contents

General

In this chapter

1.1 Typographical conventions

This manual uses various formats to highlight certain terms and actions. Specific notes and tips are shown with a different background color, according to their importance.

Format	Description
Bold font	Elements in the software or in the operating system, such as menu items, buttons or elements of a selection list
Italic font	Emphases (important details) and links to other chapters or documents
Triangle symbol "➢"	Instruction step
Angle bracket ">"	Command menu sequences, e.g. **File** > **Open**
`System font`	Directories, code and script samples
CAPITAL LETTERS	Key names, e.g. SHIFT, CTRL, or ALT
KEY+KEY	Key combinations, i.e. the user has to hold one key and press another simultaneously, e.g. CTRL+P or ALT+F4.

Note	Used for notes or tips which facilitate the work or for additional information which enhances understanding for the product.

Important	Information which should be observed by the user, because otherwise problems or additional work may be caused in operation.

Attention	Information which should be observed by the user in order to prevent malfunctions of the system (crashes, data loss, system failure).

1.2 Help

For additional information and support, we recommend the Flexera Community (https://community.flexera.com/). Here you will find product documentation, download links and access to support.

1.3 Abbreviations

For a better understanding the abbreviations in this section are given in full text.

DC Data Collector (previous EDC / External Data Collector)

DR Data Receiver (previous EDC Server)

OSE Operating System Environment

WMI Windows Management Instrumentations

What's new

1.2202.1

- Newly generated password files for the PowerShell connectors are now bound to the system there were generated on.

1.2201.2

- Delivery of version 12.4 for the data center inventory components

1.2111.1

- The vSphere connector now loggs the version of the used PowerShell and PowerCLI module.
- A new diagnostic tool is introduced which detects the vCenter version in use.

1.2108.1

- The communication components of the Spider Data Collector and the recognition module were updated. Extensive security improvements were included. The communication encryption is converted to openSSL 1.1 and a new authentication method is used. As a result of these changes, all Spider Data Collector endpoints must be updated. Otherwise they can no longer send data to the recognition servers successfully.
- A new version of Columbus/Spider Inventory is being delivered (version: 7.6.5.21214). The Inventory Agent can install itself automatically if configured so. The inventory scanner must be updated manually.
- The Adobe Portal connector has been enhanced to support TLS. This can be switched on with the parameter "TLS = true" when calling the connector.
- Fix: When importing Adobe portal data, the import of users is no longer aborted if e-mail addresses are assigned to several user data records.

1.2107.1

- vCenter connector Fix: Due to an error, clusters that were reported simultaneously by the vCenter connector and via the Datacenter Appliance could not be recognized as the same cluster in some cases, which led to redundant devices in the Recognition module and thus also to duplicate Assets in Spider Asset.

1.2106.1

1.2105.1

1.2104.1

- An error in the AD connector that could occur in connection with GetADComputer has been fixed.

1.2103.1

- The vCenter connector has been improved further. The connector will be updated automatically during the execution of the Spider Data Collector (SDC) update.

- Only "Powered On" guests are now considered in host-guest relationships. The previous behavior lead to problems under certain constellations.

1.2102.1

- A problem with the HEAT connector has been fixed.
- The vCenter connector has been improved further. The connector will be updated automatically during the execution of the Spider Data Collector (SDC) update.
  - The new run parameter OnlyWindows controls that details are limited to running virtual instances with Windows as an operational system. Other virtual instances (e.g. Linux based system) are not included. The enabled details enable that Spider creates an Asset, based on the provided data.
  - Due to problems with previous versions, the vCenter connector no longer delivers IP addresses.
  - For VDI infrastructures the vCenter connector provides the host name for device identification for Windows client operating systems (Windows 7, Windows 8, or Windows 10) only.

1.2012.1

- SCCM connector has been improved by the following issues:
  - SQL Server 2019 version detection
  - Windows 10 build versions
  - File export for Visual Studio
- The VCenter connector has been further improved. Only connected hosts are queried, disconnected hosts are ignored. Guests with an empty hostname could lead to an error. This has been fixed.

1.2011.1

- Columbus Inventory Agent and Columbus Inventory Scanner were made available as new versions. In addition to resolving the identified security problem, the performance of the inventory scanner has also been improved.
- Fix: Under certain circumstances, the vCenter Connector did not return any data because the writing of the SWRD file could not be completed.
- Since the September version, the vCenter Connector has also been delivering data from running virtual systems and thus delivering more data to Spider. Accordingly, assets with some system data are visible, even if no additional inventory is made. If this functionality leads to problems, the extension of the guest details can be deactivated.

1.2009.1

- The vCenter interface (connector) has been expanded with additional guest information: the host name, domain, operating system, CPU information and other fields are now also transmitted for the guests. Thus, guest devices can also be created as an Asset in Spider if they are not captured by the hardware inventory.

1.2006.1

- Additional file scan filters have been added to all database-based data connectors. Files that are not required are no longer exported any more, which makes the export files smaller and speed up their processing. The filtering is not carried out on the databases which previously led to significant performance problems.
- In this version, the database-based connectors also support the encrypted passwords, as was already possible with the API-based connectors
- The Hyper-V connector has been expanded. Server hostnames of other Windows-based Hyper-V hosts can be specified via an additional configuration file so that they are inventoried one after the other with just one execution of the connector.

1.2005.1

- SCCM connector: Due to performance issues, the additional filter for files we delivered with the March update has been removed.

- The delivery for the data center inventory for Linux and Unix has been changed in the multi-platform inventory agent, which transfers the inventory results directly to the Spider Data Center Appliance. The previous "cis" agent has been removed from the delivery.

1.2004.1

- Adobe Online: For delivered "Single App" entries, the profile names will be processed and displayed in Spider.

1.2003.1

- The SCCM connector has been improved to reduce the size of the output file by avoiding irrelevant file scans.

1.2001.2

- Improved recognition of Citrix Remote Desktop Services: Cascaded accesses are now evaluable.

1.2001.1

- Bugfix on FileScan_Columbus on large Files implemented.
- Bugfix of possible arithmetic overflow error in ivanti connector implemented.

1.1912.1

- New: Connector prioritizing now possible.

1.1911.1

- Exchange Connector is now delivered by default.
- Bugfix in recognition rules of some Adobe products.
- Improved support of eRunbook.
- Improvements on intune connector.

1.1910.1

- Improved intune connector: change to SerialNo-based URN.
- Improved anonymization on data exports.

1.1909

- Older versions of the Adobe Online Connector were replaced by newer version.
- Changes to Microsoft Intune Connector: device identification is now based on field SerialNo .

1.1908

- Microsoft Intune connector has been improved. Due to customer feedback, beta-labeling could be omitted.

1.1907

- Altiris connector: Due to customer feedback, beta-labeling could be omitted.

1.1906

- The Columbus Datacenter Inventory Connector will be continued.

1.1905

- Altiris Connector performance improved.

1.1812

- Added section about the SFTP server configuration

1.1811

- Altiris Connector (Beta)

1.1809

- XEN Connector (Beta)

1.1806

- ESX/vCenter connector Datacenter-Module
- Linux and Unix Inventory in addition to MAC Inventory

1.1805

- GDPR Information
- Microsoft Exchange Connector (Beta)

1.1804

- Columbus Inventory Scanner for Mac OS
- Reorganization of this document

1.1803

- The Spider Data Collector User Manual is now also available in German
- Azure connection with application/certificate (instead of user/password)
- Empirum Workplace Management (beta)

1.1802

- Microsoft App-V Connector (beta)
- Baramundi Connector (beta)
- Export Metering Information with SCCM Connector

Install and configuration of the Data Collector

3.1 System Requirements

3.1.1 Software

- Operating System: Windows 2008 Server or greater
- Microsoft NET 4
- Microsoft PowerShell 3.0 or higher
- Must not be installed in the same OSE as Software Recognition (RC)
- Multiple Data Collector installations in the same OSE are not supported

Attention	Starting with Release 1610 all PowerShell (ps1) Scripts are digitally signed.

In case of the PowerShell execution policy being set to "AllSigned", it is necessary to add the publisher of the signing certificate to the Trusted Publishers store of the local machine.

The Setup will ask you to add the publisher to the Trusted Publishers store before continuing, this will only place it in the store for the current user, for the certificate to be valid across all accounts of the local machine please follow the instructions found in Add Publisher to Trusted Publishers (on page 27).

3.1.2 Hardware

It is recommended to use a physical or virtual OSE with the following settings:

Processor: 2 or more current CPUs/Cores

RAM: 4-8 GB

Hard disk

In addition to the space used by the Operating system.

Depending on the amount of machines that are exported, there are the following (rough) estimates:

System	Recommendation
Inventory Agent/Scanner	Zip files range from 10kb to 800kb per machine, the zip files are deleted after transmission.
SCCM	2000 machines, 25MB 560 machines, 3.5MB 32900 machines, 450MB
MAP	1280 machines, 1.2MB

Note Please note that file sizes depend on the amount of queried data, especially file information can take up a lot of space.

3.2 Installation

Download and execute "Spider Data Collector.exe".

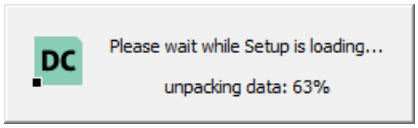


Figure - Extracting the Setup

Select the appropriate installer language.

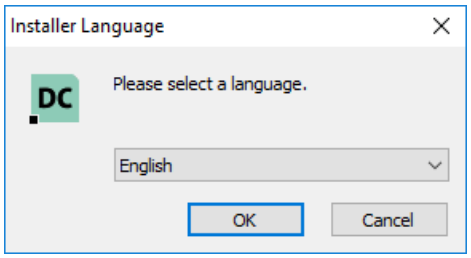


Figure - Choose a language

Click **Next**.

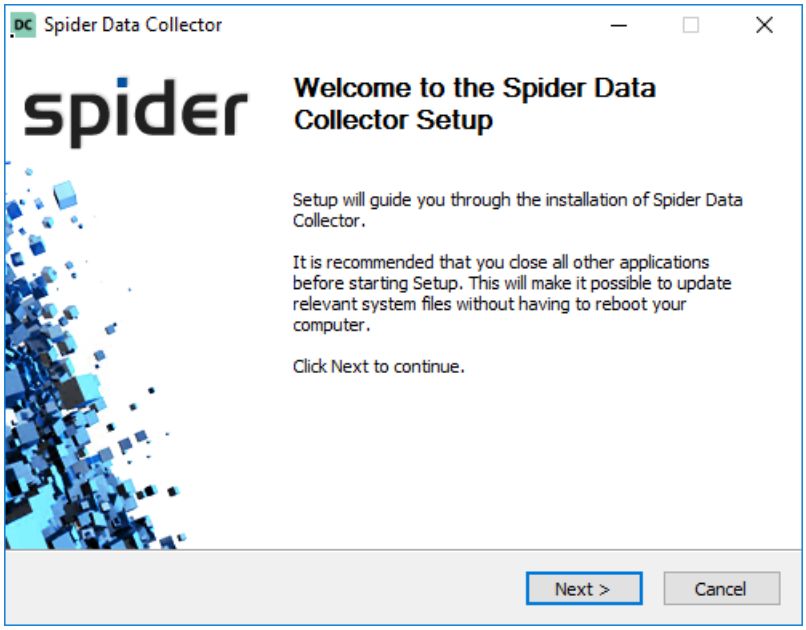


Figure - Welcome Page

Check the box to agree to the EULA Terms and then click next.

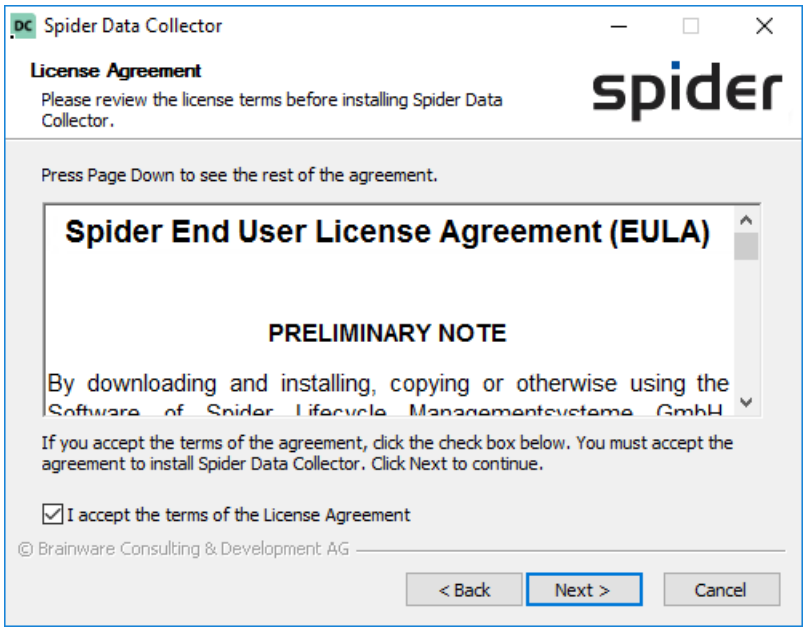


Figure - End User License Agreement

This dialog checks if the system prerequisites are valid, is not it will show you what is not in order. Selecting the line and clicking on "Details" will take you to a knowledgebase article; there you can get help about the problem in question.

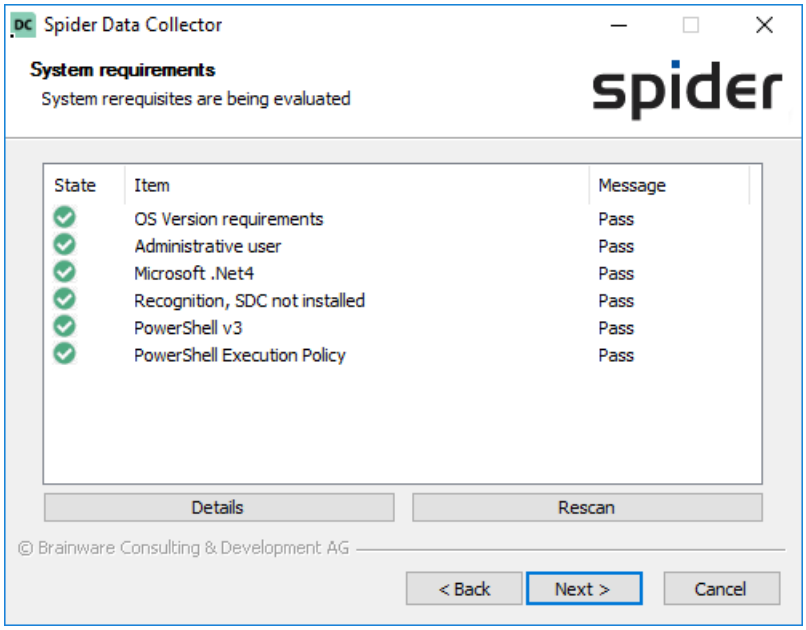


Figure - System prerequisites

Select a location for the installation of the Data Collector. If the default is acceptable, click **Next**.

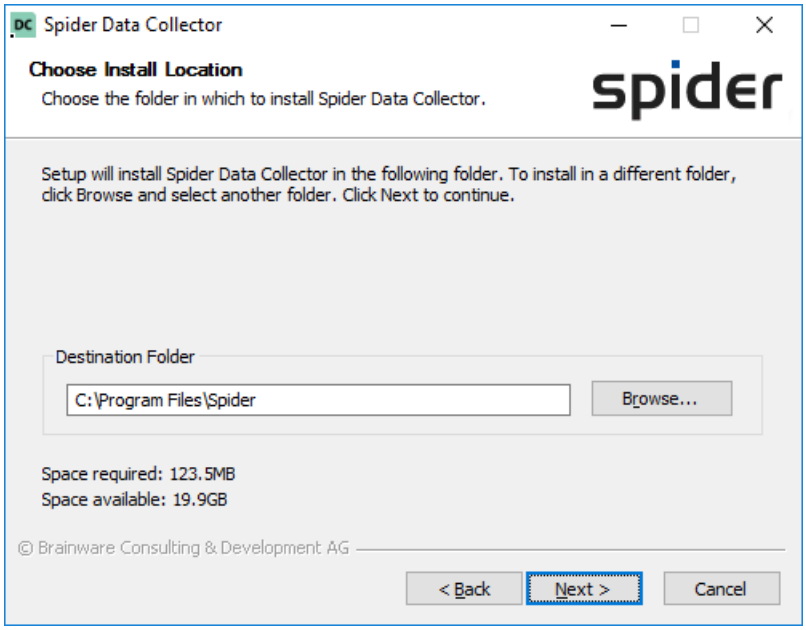


Figure - Destination Folder

On the following screen, the FQDN of the OSE that hosts the Data Receiver Server needs to be specified along with the Data Receiver Port and CustomerID where appropriate.

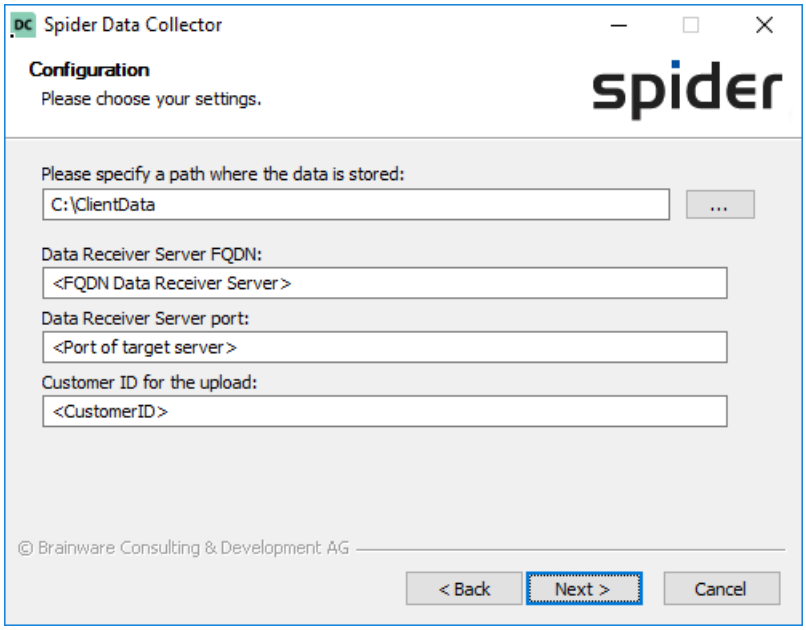


Figure - Configuration

Note For some versions of the Data Collector (for example those downloaded from a SAM Cloud platform), it is possible to preconfigure some values in the dialog and so these may be fixed.

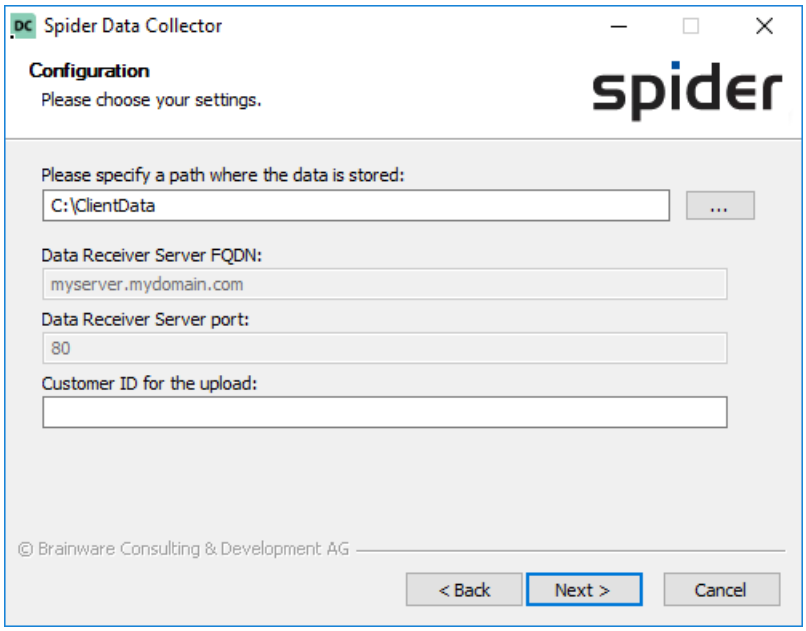


Figure - Configuration with preconfigured values

The Data Collector supports networks that use SOCKS proxy servers using both unauthenticated and authenticated methods. The current Data Collector supports proxies that support version 4, 4A or 5 of the SOCKS protocol. As is shown in the screenshot below, the IP address or DNS name for the proxy server should be specified along with the TCP communication port where the Data Collector must use a proxy. Where proxy authentication is required, the box stating **Proxy needs authentication** has to be checked and the username and password boxes filled out with the appropriate credentials.

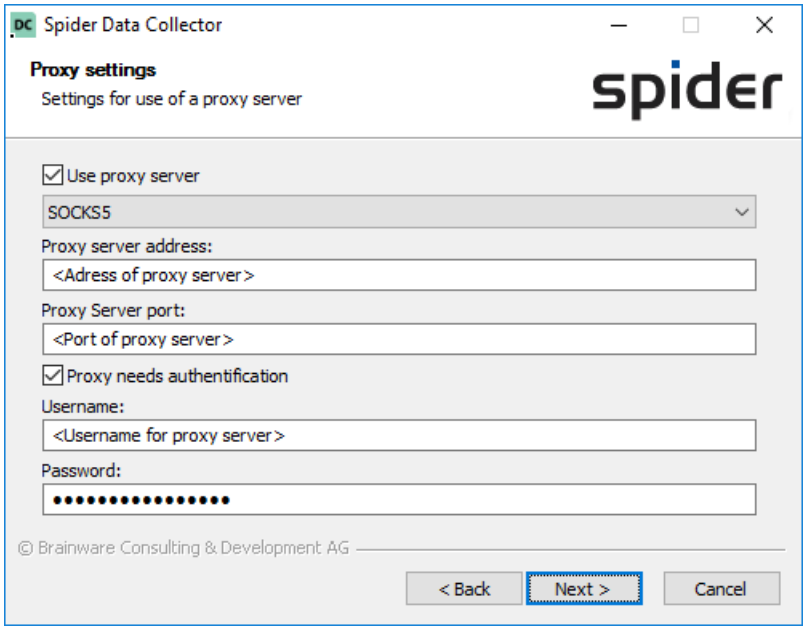


Figure - Proxy detail configuration

The following dialogue presents the schedule that the Data Collector will use to invoke the DC to the data source configured in previous steps and transmit the data to the Data Receiver.

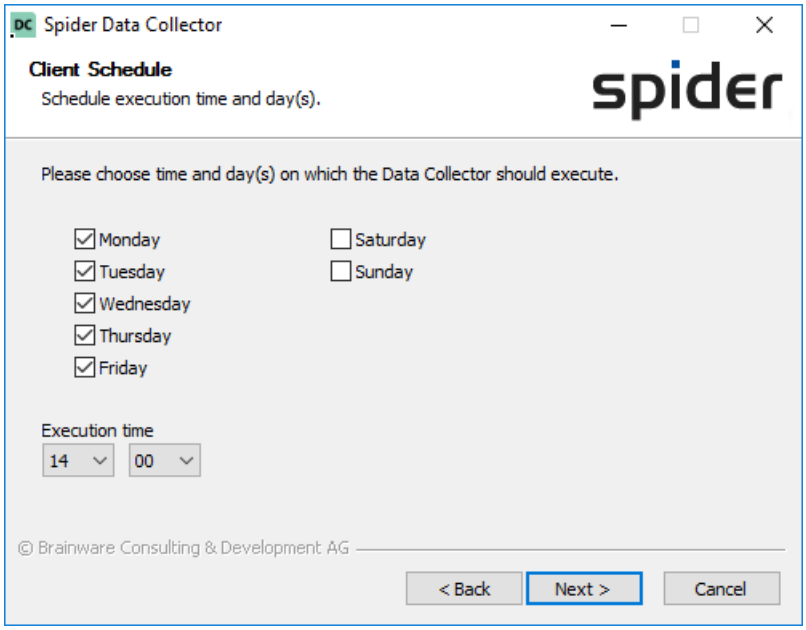


Figure - Schedule

In this step, the FQDN and port for the OTB Server may already be pre-populated by the Data Collector setup. In addition, you can set a time span in which the Inventory Tools will randomly start. Unless another OTB server is in use in the environment, it is recommended to choose a value for start delay and leave the defaults for FQDN and Port and proceed with the installation by clicking **Next**.

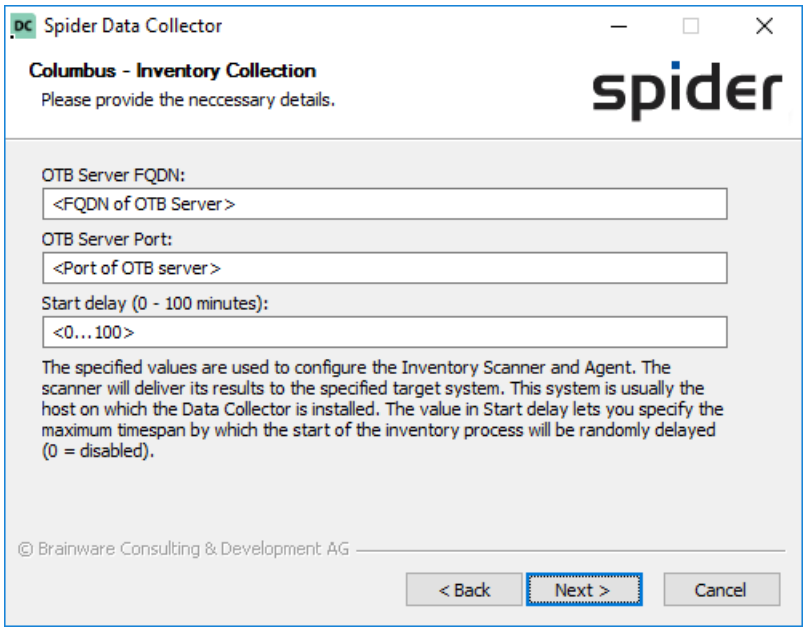


Figure - Columbus OTB Settings

In this step, you can choose which additional connectors you want to configure while executing this setup.

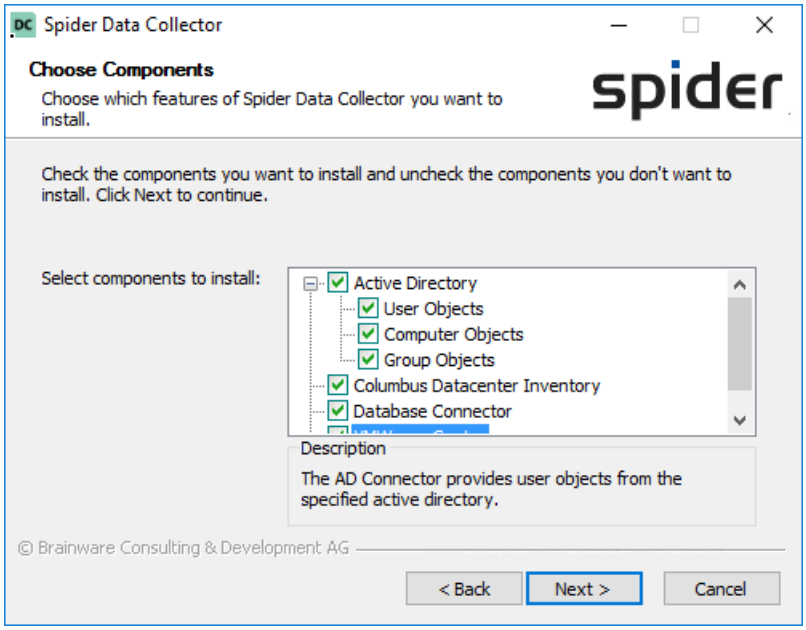


Figure - Choose additional components.

Depending on the chosen connectors, the prerequisites for those that were chosen are checked. If a prerequisite fails (or if you need more information), the item can be selected and then the Button **Details** can be clicked, to be taken to the knowledgebase for further information.

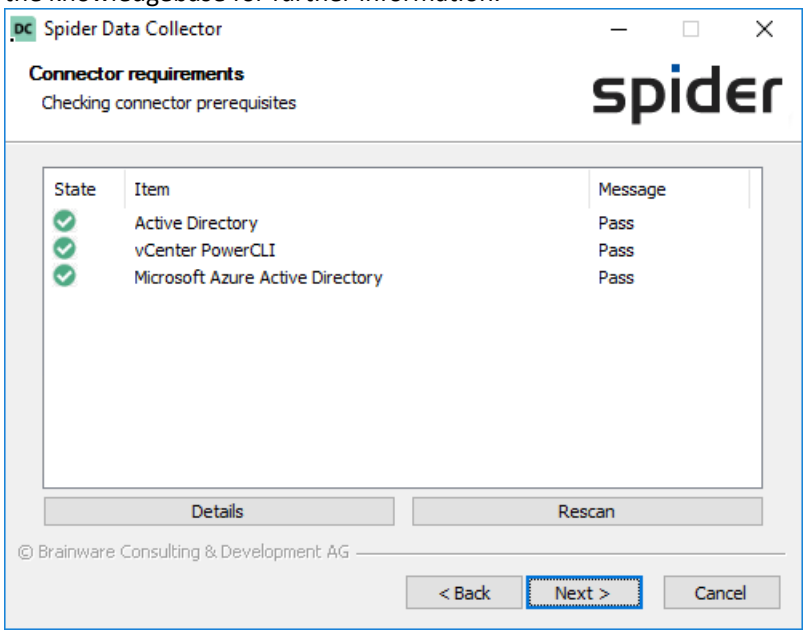


Figure - Connector Prerequisites

If "Active Directory" was selected you are now presented with this dialogue to enter the needed details. In case the necessary server role is not installed, the setup will offer a checkbox enabling the installation of that role during the setup.

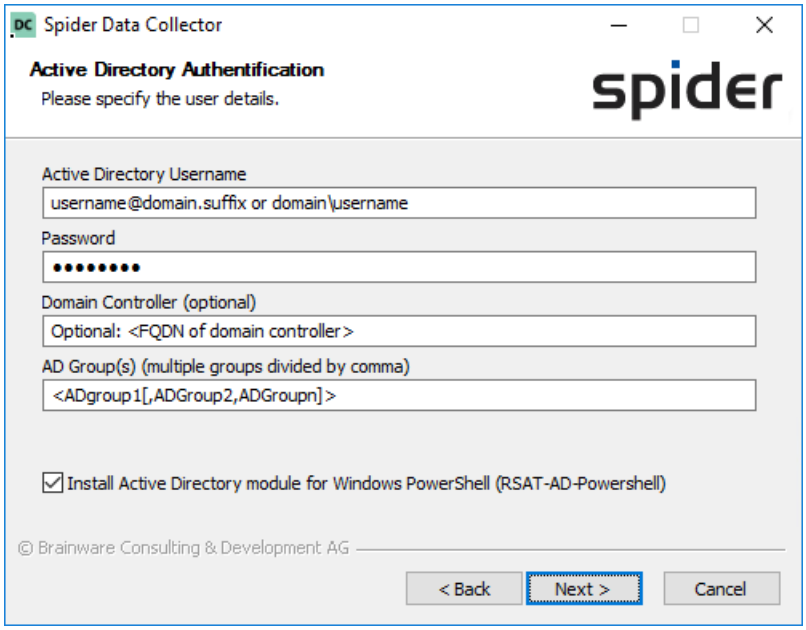


Figure - Configure Active Directory Connector

If Microsoft Azure AD was selected, the user account and password for the connection to Azure have to be entered.

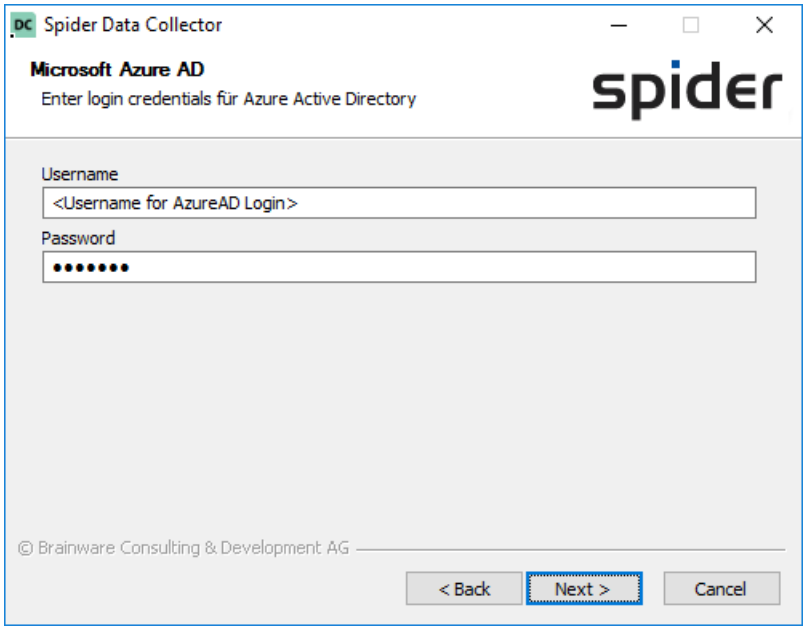


Figure - Microsoft Azure AD

If Adobe Online was chosen, a set of dialogs for entering the access information is now displayed.

Either a preexisting key pair is specified, or a new pair can be generated. The public part of the key is uploaded to the Adobe portal, the private key will be used by the Data Collector..

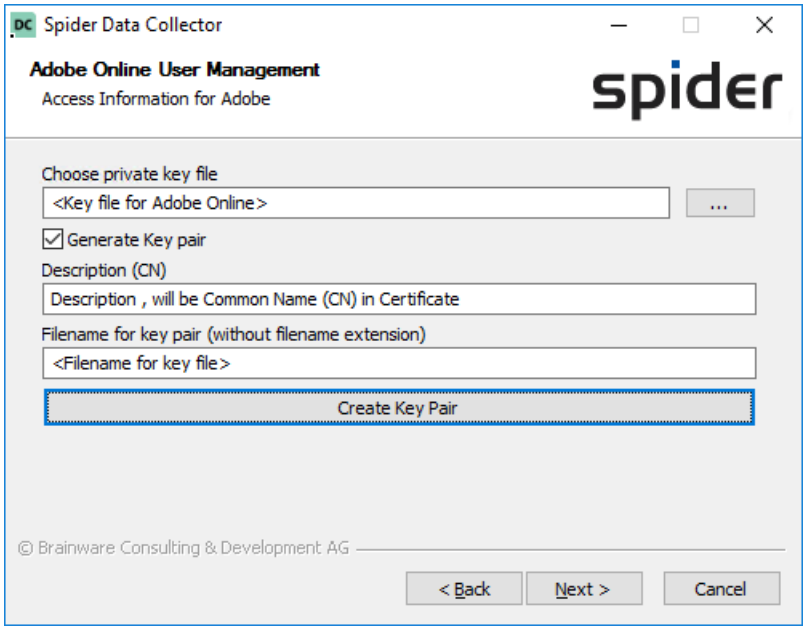


Figure - Adobe Online, Keys

A successful key generation is confirmed by the setup.

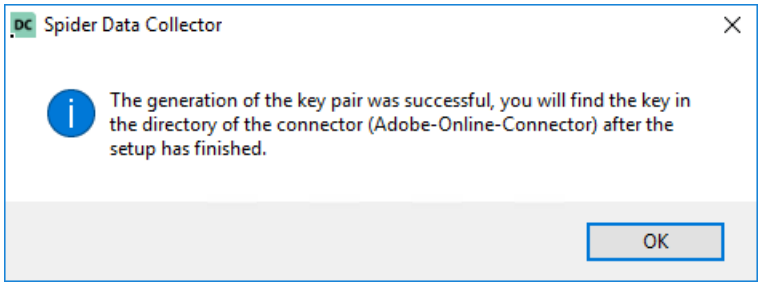


Figure - Confirmation of key generation

After the details for the key have been entered, the details about the Adobe integration have to be specified.

Details about creating the Adobe Integration can be found in the chapter: Adobe Online (on page 67).

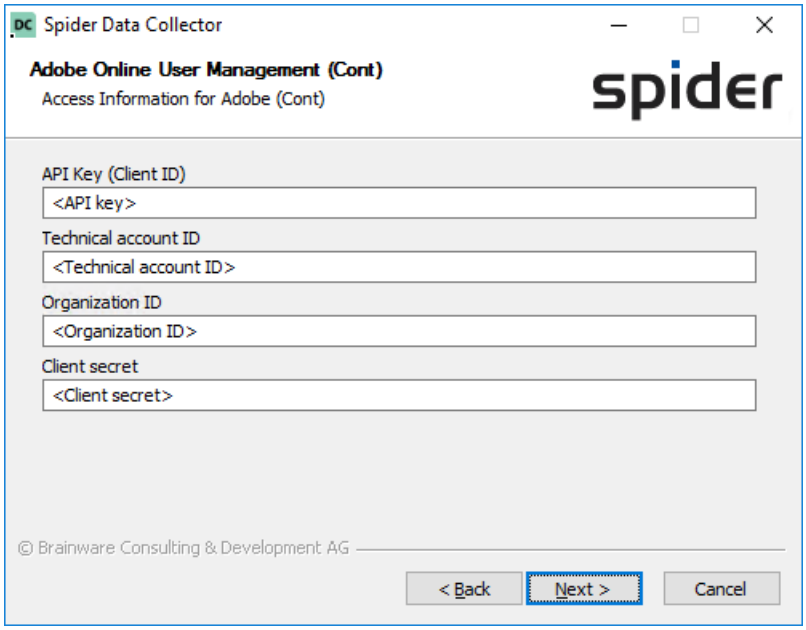


Figure - Adobe Integration details

If "Columbus Datacenter Inventory" was selected, the information to connect needs to be entered in the next dialog. A connection test will then be executed and if successful, the setup continues.

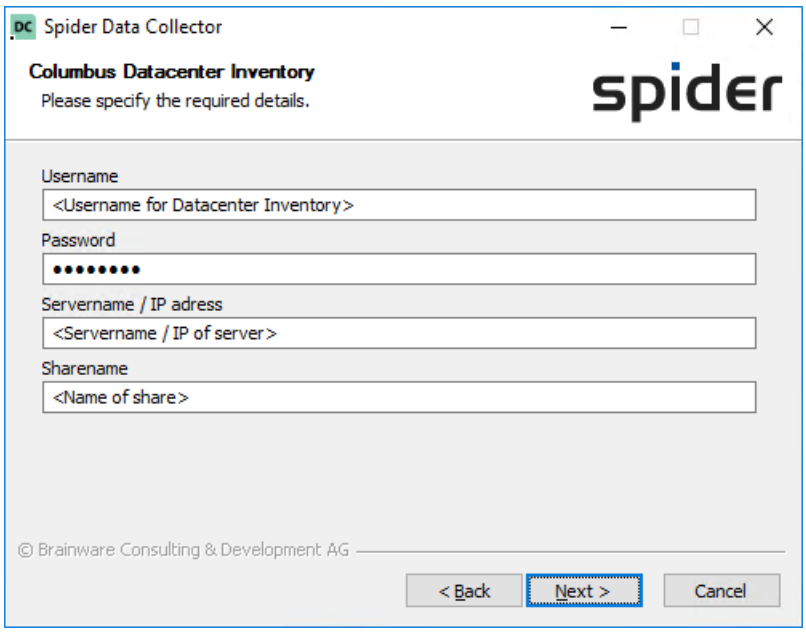


Figure - Configure Columbus Datacenter Inventory

If "Database" was selected, on the following screen, the data system and the SQL Server that hosts the database for the data source that should be connected should be specified along with the instance name of the server where appropriate. If the database resides on the default instance of SQL, only the DNS name or IP address of the SQL Server is required.

Brainwaregroup recommends the use of a service account for the Data Collector to connect to these databases to prevent issues when passwords are changed and to improve administrative security. The account may be either a SQL Se-

curity account defined locally on the destination SQL Server, or a Windows account either defined locally on the SQL Server or in Active Directory.

Select the database of the inventory source that should be connected to and specify the credentials used to make the connection to the specified SQL Server.

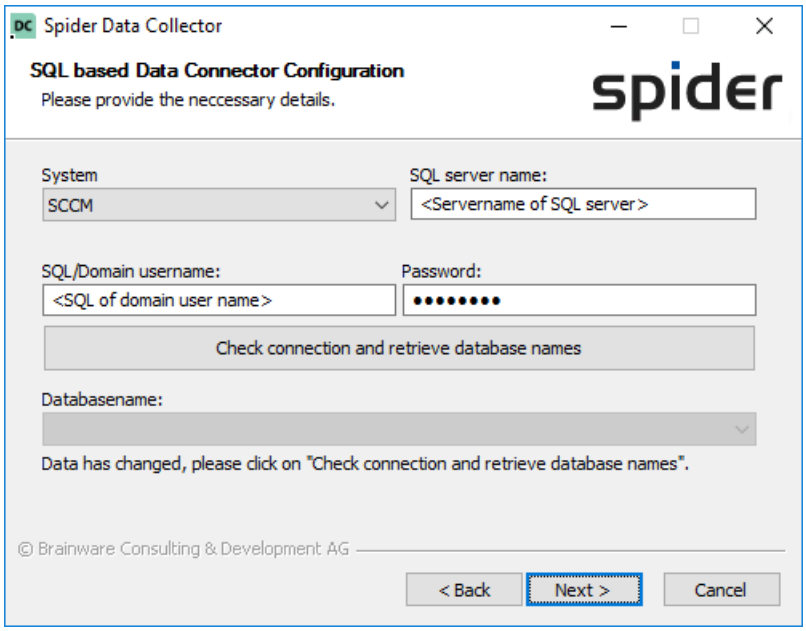


Figure - Configure database connector and choose database

If "VMWare vCenter" was selected the configuration settings must be entered in this dialogue. In case the necessary PowerCLI tools from VMWare are missing, the dialog will show a message.

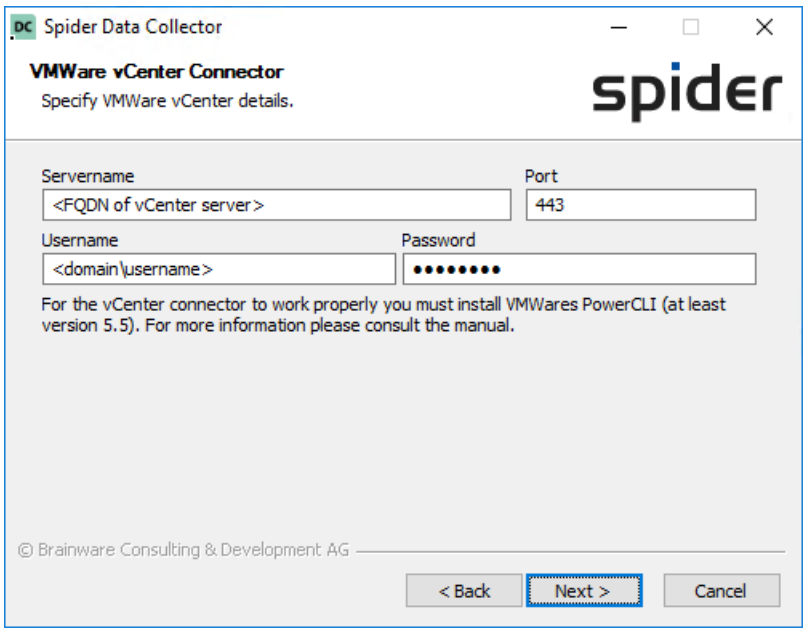


Figure - Configure VMware vCenter Connector

If multiple Connectors were chosen, the next page requires a choice which user is used for the impersonation of the Data Collector service.

Attention	If a domain user was specified as the connecting user for the SQL database, this user will be preselected and cannot be changed. All other user accounts that might have been specified during the setup will have their passwords encrypted and will be put in the configuration.

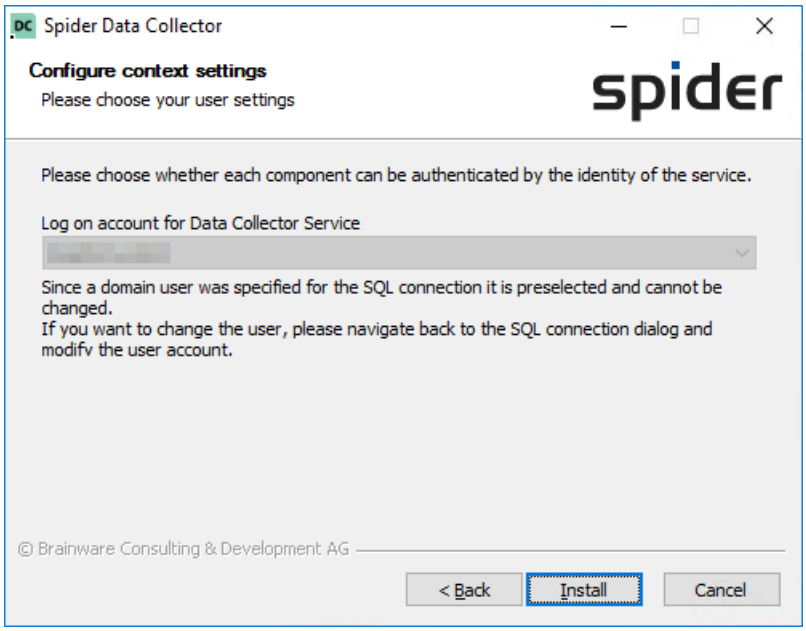


Figure - Configure Account

Once the above settings are configured accordingly, click **Install** to proceed with the installation.

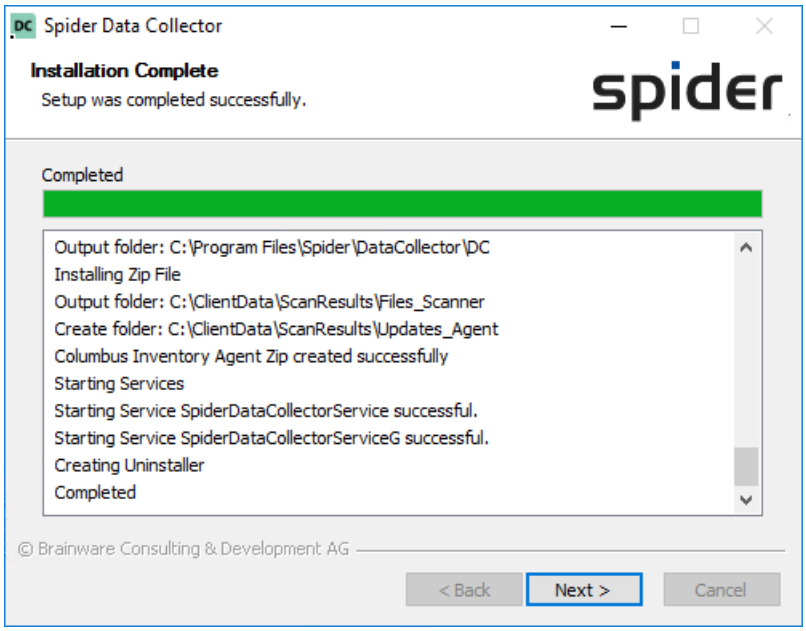


Figure - Installation

Once the installation is finish, click **Finish** to close the installer and end the installation.

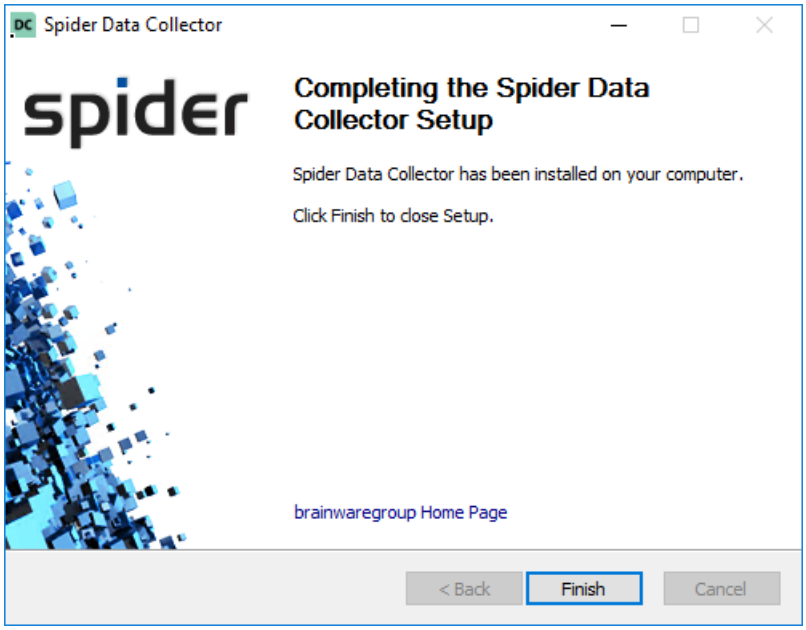


Figure - Installation finished

3.3 Configuration

3.3.1 Data Collector

In many cases, the installation will use the best settings that are appropriate for many networks. However, there are occasions when additional configuration of the Data Collector post installation may be necessary to change the values specified during the installation.

All settings for the Data Collector are held in a configuration file called SpiderDataCollector.cfg. This file resides in the service directory for the Data Collector (..\Data Collector\SpiderDataCollector.cfg).

Section	Parameter	Possible Values	Description
System	CustomerName		Identifies the Data Collector at the server, used to match the data to the Mandator in Spider, which can be retrieved from the download page.
System	SystemName		Name of the machine that Data Collector is installed on. This will be set to the name of the machine if nothing is set.
Connection	OTBHost	<FQDN or IP of the destination OTB server>	The host that will receive the data transmitted by the Data Collector, multiple hosts can be specified, separated by a comma.
Connection	OTBPort	<Valid TCP port numbers (0 - 65535)>	Port on which the data is transmitted.
Connection	ProxyType	0 = Socks5 1 = Socks4A 2 = Socks4	Determines which version of the SOCKS protocol is used.
Connection	ProxyHost	<FQDN or IP of proxy>	IP Address of your proxy server

Section	Parameter	Possible Values	Description
Connection	ProxyPort	<Valid TCP port numbers (0 - 65535)>	Port of the proxy server
Connection	ProxyAuthEnabled	0 = No authentication 1 = Authentication required	Determines if the proxy need authentication in order to access the outside.
Connection	ProxyUser		UserID for proxy access
Connection	ProxyPassword		Encrypted password for proxy access. The password can be encrypted using CryptIt.exe which is in the same directory as SpiderDataCollector.cfg
OTBServer	OTBActive	1 = Active other values = disabled	Determines if the Data Collector listens on the port given in OTBPort for incoming Scan Results from Inventory Scanner or Agent
OTBServer	DataDirectory		Directory where the received inventory zip files will be stored
OTBServer	OTBPort	<Valid TCP port numbers (0 - 65535)>	Port on which the Data Collector listens for transmissions of zip files.
OTBServer	MaxConnections	Integer	Amount of parallel connections possible (Default: 1000)
Schedule	ScheduleTime	0000-2359	Time at which the Data Collector will execute the action defined in Command line and transmit the received data. Must be specified in 24 hour format with no colon e.g. 17:00 has to be written as 1700
Schedule	ScheduleDaysOfTheWeek	0000000-1111111	Each binary digit represents a day; the first being Monday and the last being Sunday. Toggling the appropriate digit enables scheduling on the corresponding day e.g. 0100101 - means run on Tuesday, Friday and Sunday Important - if no schedule is set, no processing will take place.
General	Commandline		is the command line to execute to generate files to upload (environment variables are resolved)
General	DataDirectory		is the data directory to save files in that should be uploaded - default if not set is a folder "Data" where the SpiderDataCollecotr.exe is installed (environment variables are resolved)
General	ExecutionTimeOut		Timeout that defines after how many minutes the execution of the command in Commandline is considered incomplete and transmission continues.

Important	Please note that that only SOCKS is supported as proxy protocol.

Schedule

The Data Collector will check for actions every six minutes.

Directories

After installation all files that are due to be uploaded must be placed into the directory that is specified in Section "General" Parameter "DataDirectory" in SpiderDataCollector.cfg.

Details to the SpiderDataCollector.cfg can be found in the chapter: Data Collector Configuration

Batch Files

There are several batch files that are responsible for exporting and processing data. All of these are in subdirectories of the service directory for the Data Collector.

Directory and name of batch file	Description
..\DataCollector\DC\StartDCsPS.cmd	Main file that is called from Data Collector when the Data Collector is executing its actions. This file will also call the other processes that are needed for exporting and processing data files.

3.3.2 SFTP Server

For updating and receiving data from the Mac Inventory components a SFTP server is necessary. This SFTP server (ColumbusSftpServer.exe) is automatically installed and is started through the SpiderDataCollector.exe. On (first) start the necessary ssh keys will be generated automatically and placed in the SpiderDataCollector.json.

Note: The Inventory for Linux and UNIX is now delivered directly to the Data Center Appliance.

The parameters for the SpiderDataCollector.json are explained in the following table:

Section	Possible Values	Description
"uploads": [],		Reserved for future use.
"sftpSettings": {...},		Configuration section for the SFTP server.

"sftpSettings": {
"Active": true,
...
}
	true|false	Determines if the SFTP server is active (true) or inactive (false)

"sftpSettings": {
"RootDir": "<Directory>",
...
}
		Base folder for the SFTP server, usually the "ClientData" folder specified during the initial SDC setup.

"sftpSettings": {
"Port": 22,,
...
}
	Integer,	Port of the SFTP server.

"sftpSettings": {
"PrivateKey": "<Key>"
...
}
		PrivateKey for the identification of the server to the client.

"Users": [{User1},{Usern}]		Here, users for access to the SFTP sill be defined.

"Users": [
{
"UserName": "<Username>",
...
}
]
		Name of the user allowed to connect to the SFTP server.

Section	Possible Values	Description

"Users": [
{
"HomeDir": "InvData\\cis",
...
}
]
		Home directory of the SFTP user.

"Users": [
{
"PubKey": "<Key>",
...
}
]
		PublicKey of the user, the private key will be distributed with the inventory components.
		If this is value is empty, the service will automatically generate a new key and write it do the .json file. Also a copy of the private key will be placed as cis.prv in the .cis subfolder of the Data Collector.

"Users": [
{
"IsAdmin": false,
...
}
]
	true|false	Determines if the user is an administrator, if set to true the user is allowed to browse all of the sftp directories. If set to false the user can only browse his home directory.

Attention Neither private nor public key can be replaced by self-generated keys. If new keys should be used the values for PubKey and/or PrivateKey need to be emptied ("""). The service will then generate new keys.

If a new PubKey is generated, the private key that is created with it has to be exchanged manually(!) for all Inventory components. The file cis.prv will be provided in the .\cis subfolder of the Data Collector. For the configuration of the Inventory components please see: Columbus Inventory Scanner Configuration (on page 115)

3.4 Resetting last upload date

For testing purposes, it might be necessary to reset the date the Data Collector has last uploaded to its server.

This can be achieved by deleting the registry value

```
Key: HKEY_LOCAL_MACHINE\SOFTWARE\<Wow6432Node>\Brainware\Columbus\7\OTB\Client
Value: LastScheduledActionCompleted
```

Important Even if the above key is deleted, the time specified in the "ScheduleTime" attribute must have passed in order for the Data Collector to execute on its next check of the key (every six minutes).

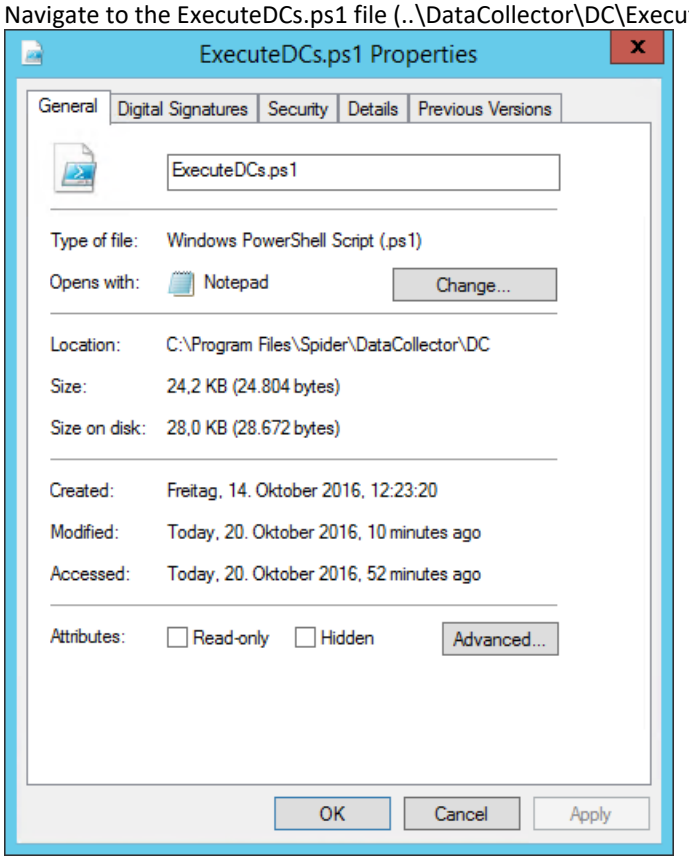
3.5 Add Publisher to Trusted Publishers

In case of the machine policy for PowerShell being set to "AllSigned" it is necessary to include the Publisher of the certificate to the local machines Trusted Publishers store.

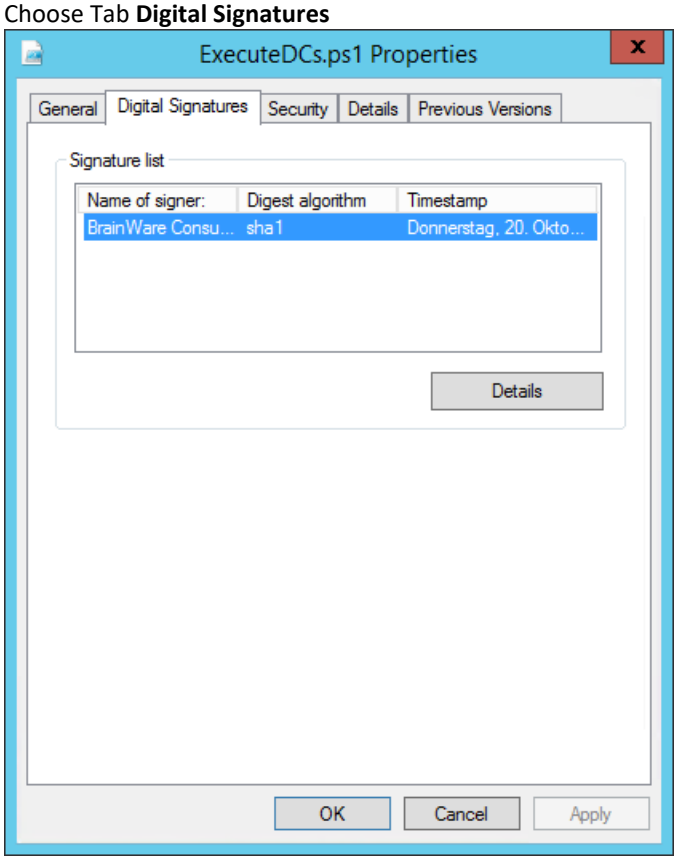
Attention	It is important that the certificate is imported to the Trusted Publishers of the **local machine,** not only the user! If it is only imported into the user store it will not be available if the service is running under another account other than the user the Data Collector was installed with.

In order to add the publisher to the trusted store please execute the following steps.

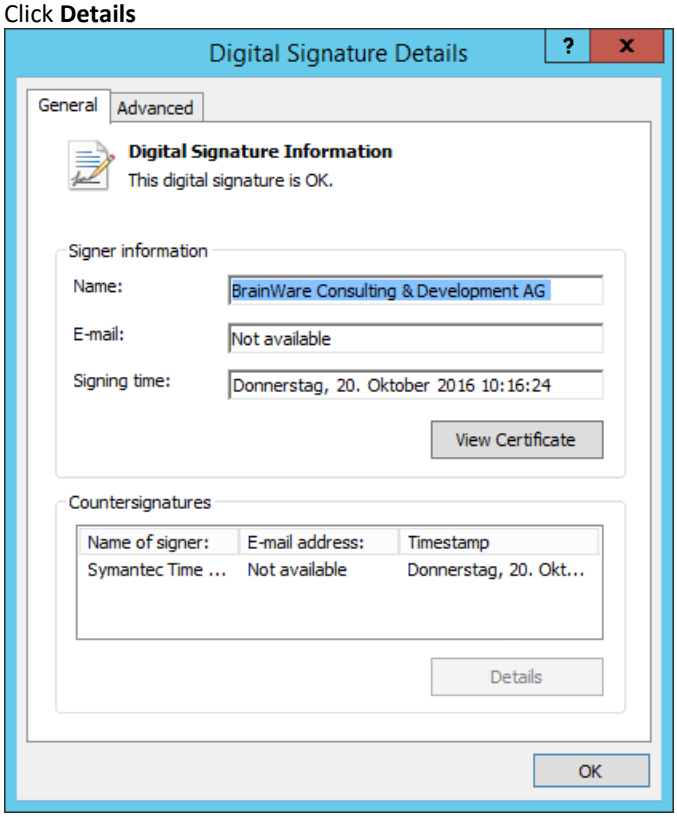
1. Navigate to the ExecuteDCs.ps1 file (..\DataCollector\DC\ExecuteDCs.ps1), right klick and choose **Properties**



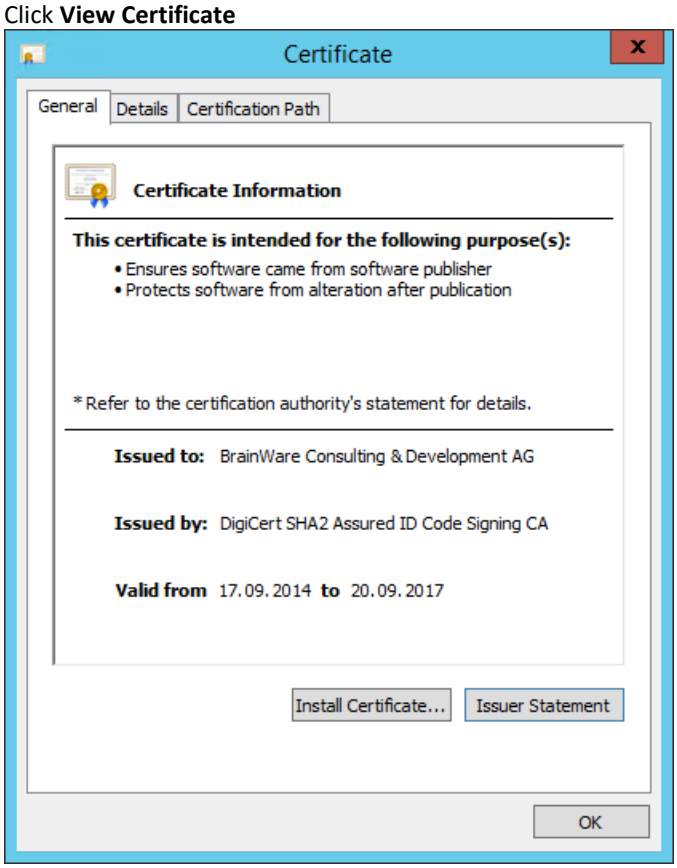
2. Choose Tab **Digital Signatures**



3. Click **Details**

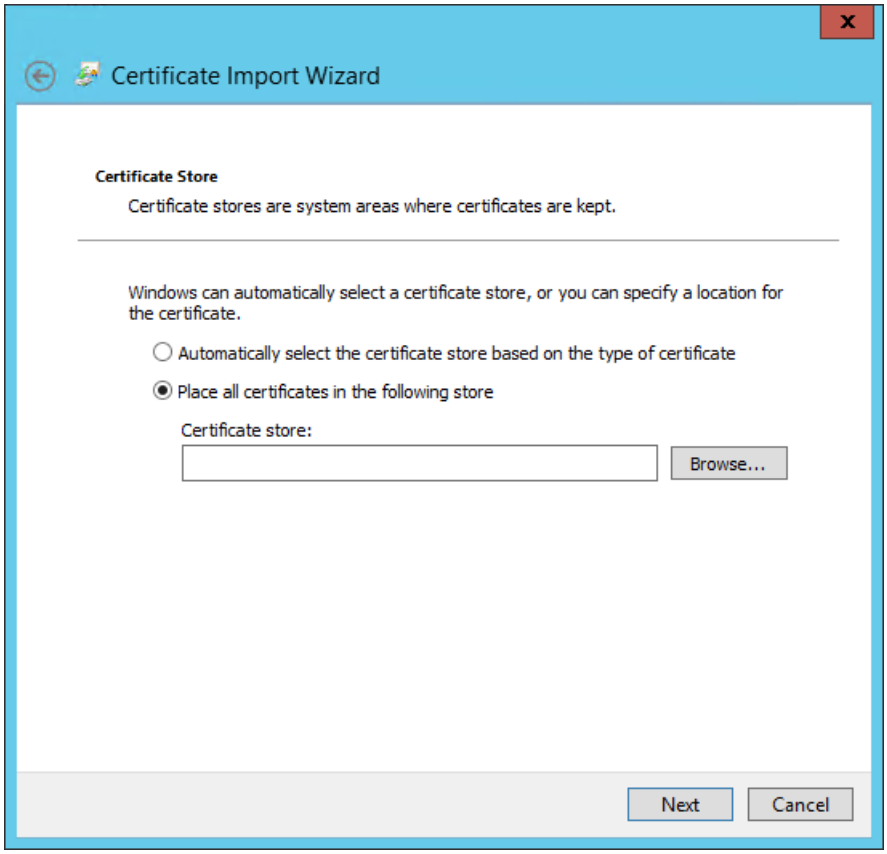


4. Click **View Certificate**

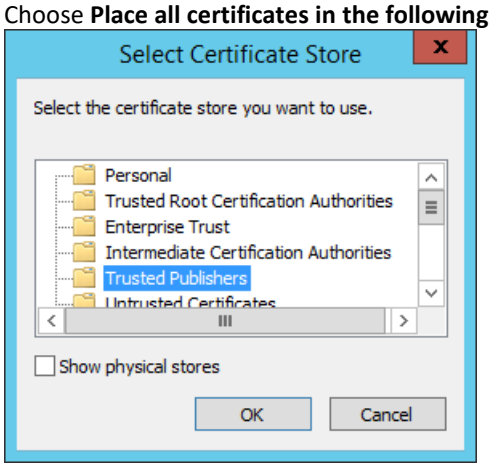


5. Click **Install Certificate**

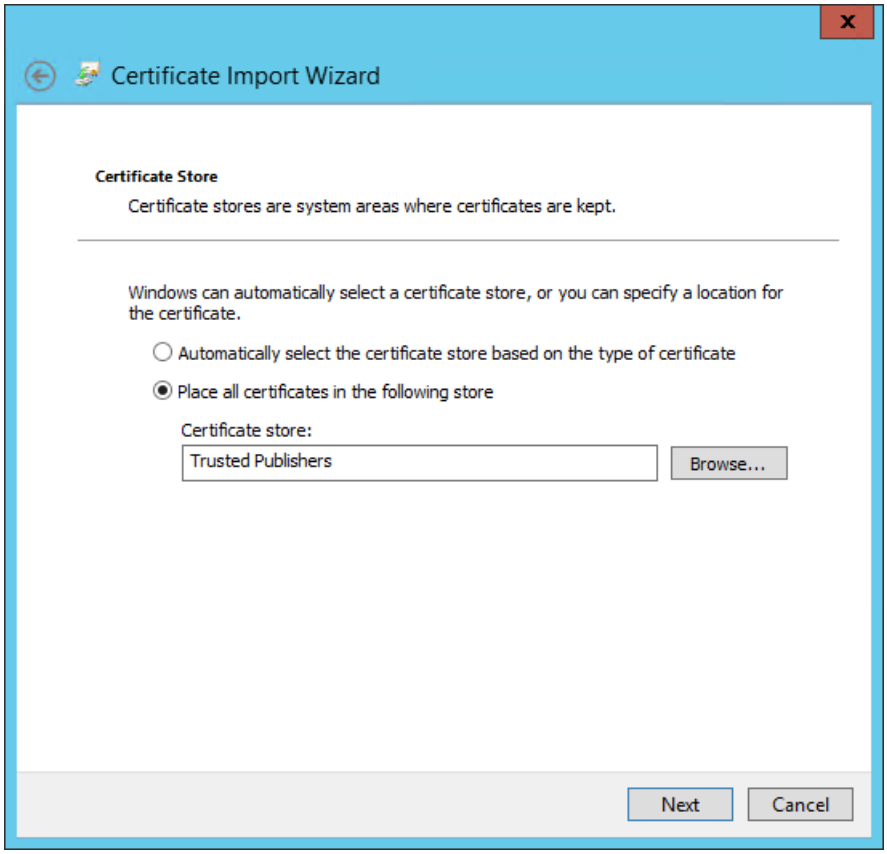
6. Choose **Local Machine** and click **Next**



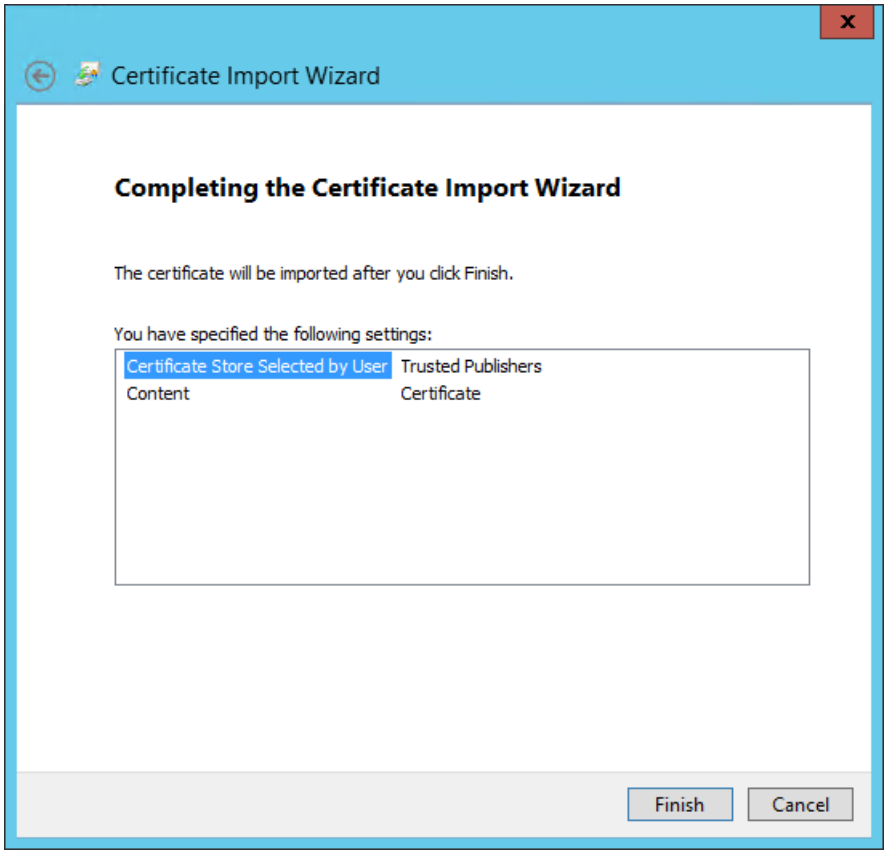
7. Choose **Place all certificates in the following store** and click **Browse...**



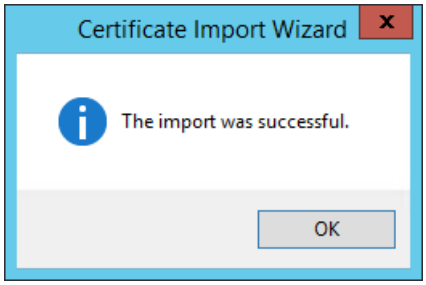
8. Choose **Trusted Publishers** and click **OK**



9. Click **Next**



10. Click **Finish**



3.6 Uninstall

Uninstallation of the Data Collector can be achieved in two ways; either by invoking the "Spider Data Collector_Uninstall.exe" from the path specified during installation, or by using the entry in "Programs and Features".

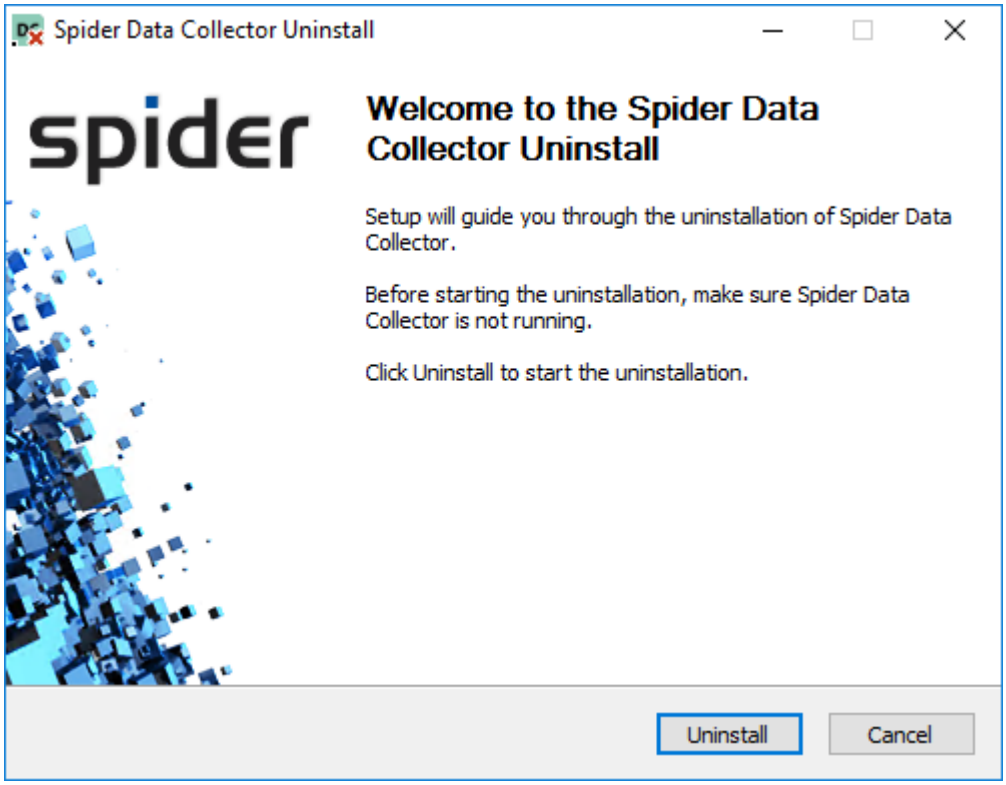


Figure - Welcome Page

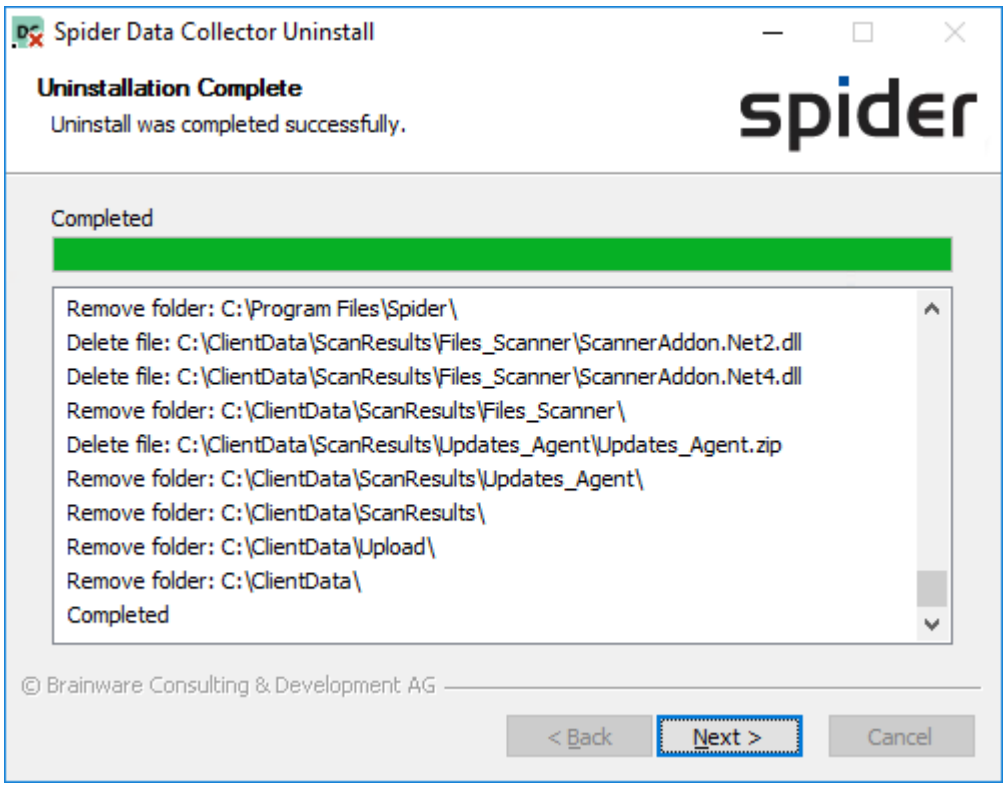


Figure - Removal process

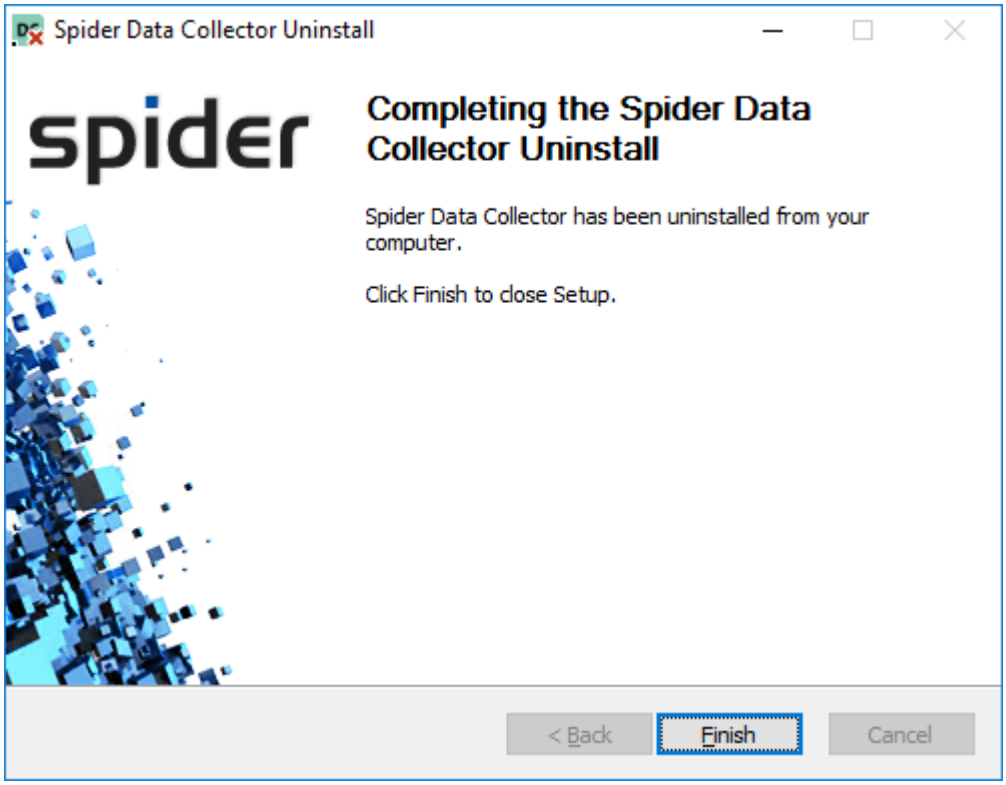


Figure - End of uninstall

3.7 Troubleshooting Connection /Authentication Issues

If your Data Collector does not generate Inventory Data and/or does not upload the results to the server here are some points to check out.

First please check the brainware.log file (%windir% or %ProgramData%\Columbus)

Common Messages

Message	Description
SpiderDataCollector: Connecting to myserver:myport	indicates that the connection to the server is now attempted
SpiderDataCollector: Connection to myserver:myport has succeeded	connection could be established successfully.
Connection from [EDC Client Manager] to [myserver:myport] failed using IP v4 [Socket Error # 10061; Connection refused.], trying IP v6	if errors like this appear, please check if server and port a configured correctly, additionally please check firewall settings in your environment (computer and network) that may prevent connecting to the specified server.
Connection from [EDC Client Manager] to [myserver:myport] failed using IP v6 [Socket Error # 11001; Host not found.]	
SpiderDataCollector: [ERROR] - Problem connecting to the OTB server on myserver:myport with message: Socket Error # 10061; Connection refused.	
SpiderDataCollector: Authenticating with myserver:myport]	Authentication attempt
SpiderDataCollector: Authentication: client [{A74192A6-BF66-49F2-8271-90EEBEE61BDF}: <Customer-ID>:<Servername>:7.5.2.39] is active on the OTB server [<Servername>.<Port>]	Data Collector could successfully authenticate with server.
SpiderDataCollector: Authentication: client [{AE97A9DC-5DFE-442B-B448-56ED1B92BDB1}: <Customer-ID>:<Servername>:7.5.2.39] is new pending registration and activation with the OTB server [<Servername>.<Port>]	indicates that your myCustomerID was registered on the server and awaits activation
SpiderDataCollector: [WARNING] - Failed authentication: Authentication: [{AE97A9DC-5DFE-442B-B448-56ED1B92BDB1}:<CustomerID>:<Servername>:7.5.2.39] has failed to authenticate with the OTB server [<Servername>.<Port>]	Indicates that the authentication on the server has failed, this usually means that the myCustomerID is not known on the server or that it has been deactivated, in this case please check if Software Recognition has been activated for your Mandator and that you have used the correct myCustomerID, in case this is on a hosted system please contact support to check this for you.

Attention On Windows 2012 R2 or later you can use the following Powershell command to test if the machine the DC is installed on can connect to the port on the server running the recognition.
More information can be found here:
https://community.flexera.com/t5/Spider-Knowledge-Base/Operations-Manager-How-to-troubleshoot-Network-connections/ta-p/4791

```
Test-NetConnection -Computername <ServerName> -Port <Port> -InformationLevel Detailed
```

Note The date format in the brainware.log may vary depending on the regional settings of the machine.

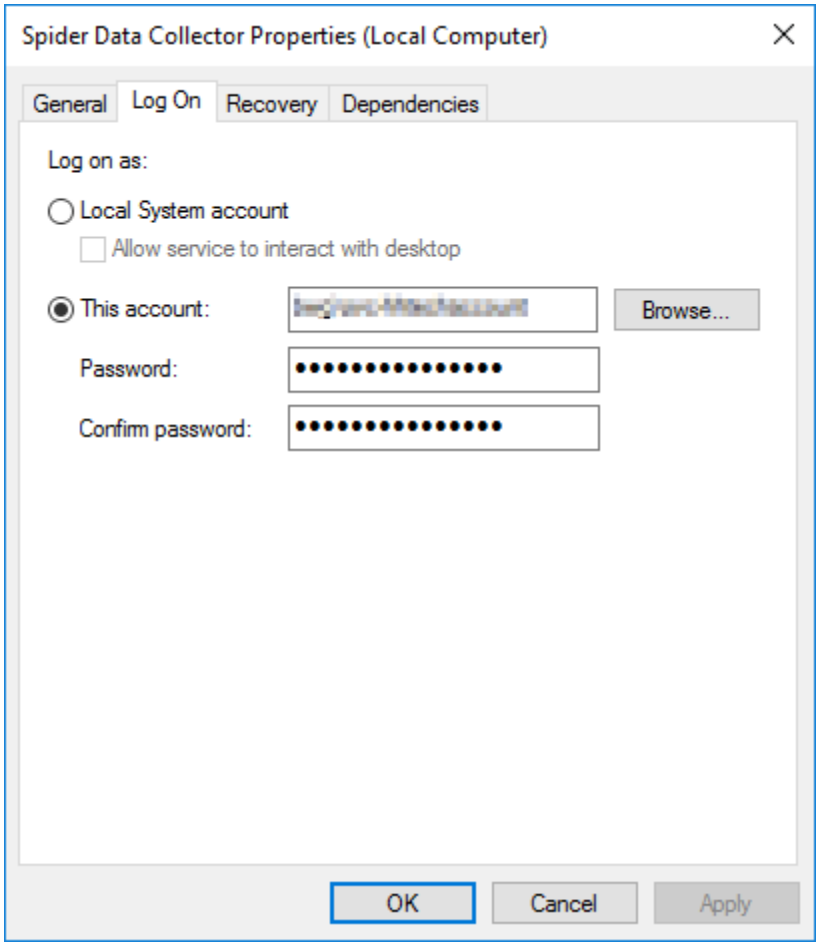
3.8 Data Collector Service Account

When using impersonation (e.g. for SQL database access or Active Directory Connector), the Data Collector Service runs under a user account specified during the setup.

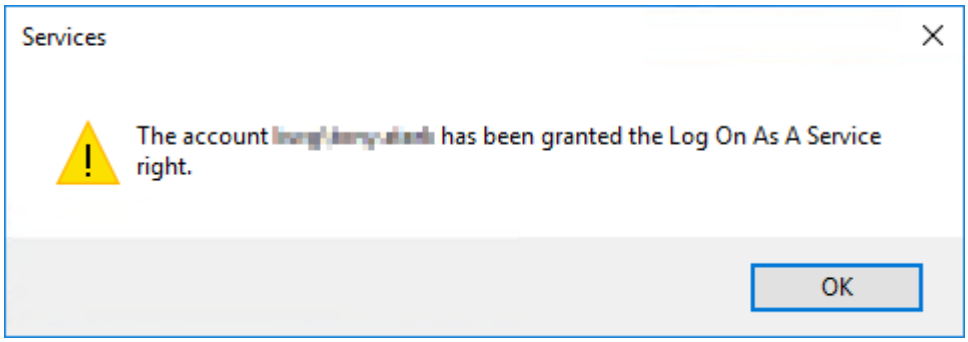
In some situations; a change of this account may be desired in order to do so please execute the following steps:

1. Stop Data Collector Guardian service by issuing "net stop SpiderDataCollectorServiceG" in an administrative command prompt or from service manager.
2. Stop Data Collector service by issuing "net stop SpiderDataCollectorService" in an administrative command prompt or from service manager.

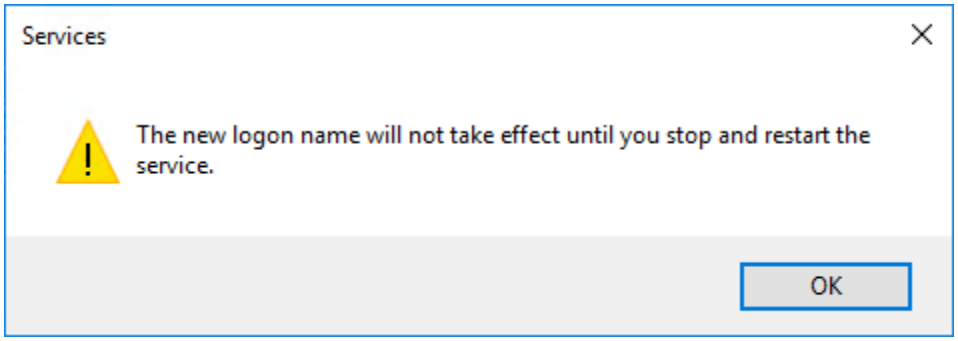
3. Open service configuration in service manager and navigate to the "Log on" tab of the "Spider Data Collector" service.
4. Change the account to the new account information and click on "OK"



5. Confirm the Log On As A Service message



6. Confirm the restart service message



7. Modify the security settings on the Data Collector executable path so the new user has read/write (modify) access to the folder and below. The path where the Data Collector is installed can be found through the registry under

   `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\brainwaregroup\DataCollector Value: Path`

8. Modify the security settings on the Data Collector data path so the new user has read/write (modify) access to the folder and below. The Data Collector data path can be found in the SpiderDataCollector.cfg (Section: [General] Value: Data-directory) in the Data Collector executable directory (see step 5) or in the registry under

   `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\brainwaregroup\DataCollector Value: DataCollectorDataPath`

9. Modify the registry permissions the new user has "Full Control" on the key

   `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\BrainWare\Columbus\7\OTB\Client`

10. Start Data Collector service by issuing "net start SpiderDataCollectorService" in an administrative command prompt or from service manager.
11. Start Data Collector Guardian service by issuing "net start SpiderDataCollectorServiceG" in an administrative command prompt or from service manager.
12. Done!

Important	In case the new user is not an administrative user the location of the brainware.log file will change from the windows directory to %ProgramData%\Columbus\Brainware.log

Configuring execution of connectors

Attention	Starting with Release 1.1609 of the DC, the management of the connectors used to query inventory information will be changed, please read this chapter carefully! The automatic migration of the configured connectors prior to release 1.1609 will only take into account connectors installed by the setup. Any manual changes will be overwritten or left out of the migration.

Up until version 1.1608, the connectors where executed through a set of .cmd files, this has been replaced with a management PowerShell script using a single file for configuration of all connectors.

The file (Connector.config) is located in the ..\DC folder and is an XML structure with one connector configured per <connector> entry.

```
<?xml version="1.0" encoding="utf-8" ?>
<connectors>
    <connector name="DSDC SCCM Inventory" subfolder="DSDC" active="true" scriptname="DSDC.exe" srv="servername" db="database name"
t="SCCM" uid="UserID" pw="Password" h="true" s="true" f="true" sfx="_SCCM" />
    <connector name="vCenter Inventory" subfolder="vCenter" active="true" scriptname="GetvCenter-Details.ps1" srv="servername"
port="port number" uid="domain\username" pwd="password"  h="true" s="true" dr="true" sfx="suffix" />
    <connector name="ADUserObjects" subfolder="ADConnector" active="true" scriptname="GetADUserObjects.ps1"
uid="domain\username" pwd="password"  dc="" sfx="" filter="" ou="" />
    <connector name="ADComputerObjects" subfolder="ADConnector" active="true" scriptname="GetADComputerObjects.ps1"
uid="domain\username" pwd="password"  dc="" sfx="" filter="" ou="" InactiveDays="" />
    <connector name="GetADGroupObjects" subfolder="ADConnector" active="true" scriptname="GetADGroupObjects.ps1"
uid="domain\username" pwd="password"  dc="" sfx="" grp="group[,group]" strict="true" />
    <connector name="DataConnectorColumbus" subfolder="Columbus" active="true" scriptname="DataConnectorColumbus.exe"
noflag="true" nouploaddir="true" />
    <connector name="Columbus Datacenter Inventory" subfolder="DatacenterInventory" active="true"
scriptname="GetDatacenterInventory.ps1" uid="erunbook" server="10.1.2.3" share="spider" sfx="" />
    <connector name="Hyper-V" subfolder="Hyper-V" active="false" scriptname="GetHyper-VDetails.ps1" uid="" pwd="" srv="" sfx=""
/>
</connectors>
```

The configuration files contains common entries (described in the table below) and connector specific entries.

Common entries:

Attribute	Mandatory	Description
name="<Name of the connector>"	Yes	Name of the Connector, can be modified in order to accommodate multiple executions of the same connector. E.g. when querying two or more SCCM databases.
subfolder="<subfolder>"	Yes	Subfolder (relative to ..\DC) where the connector script or executable is located.
active="<true|false>"	Yes	If the connector is active or not.
scriptname="<Name of the script or executable>	Yes	Name of the Script or executable that is used by this connector.
timeout="# of seconds"	No	Timeout after which the connector will be stopped. Be aware that some connectors (depending on the size of the attached inventory environment) may run very long, setting a timeout may interrupt a perfectly well export.
sfx="<FileSuffix>	No	Suffix that will be appended to the export filename (adding "_" if not present)

Attributes other than the common ones described above are directly passed to the chosen connector and have to be retrieved from the connectors description in this manual.

General configuration hints:

- The order of execution is determined by the order the connectors are specified in the Connector.config file.
- The directory parameter from the connectors (-dir or similar) may not be specified in the Connector.config, this one is automatically queried from the SpiderDataCollector.cfg and amended to the command line executing the connector.
- True|False parameters (e.g. /h for hardware scan of DSDC.exe) have to be specified as h="true" in the connector attributes, if it is not needed it may not be specified at all. (so it's either h="true" or nothing).

- The PowerShell based connectors can encrypt their passwords. please refer to Password Encryption for PowerShell based Connectors (on page 60) for details, if you use password encryption pwd=<password>" may not be specified, not even pwd="!
- When using multiple entries of the same connector (e.g. to query two or more SCCM databases), the suffix needs to be adjusted, otherwise results from the second call will overwrite the results of the first call.

Note	The execution of the scripts will be logged to %ProgramData%\ExecuteDCs\ExecuteDCs.log

Escaping special characters

If for some reason you need to use special characters in the Connector.config, some of them need to be escaped, please see the following list:

Character	Escaped character
<	<
>	>
&	&
"	"
'	'

4.1 Password Encryption for the connectors

The connectors support the encryption of the used password into a file.

To aid in creating or updating the password a PowerShell (EncryptPassword.ps1) script has been placed into the ..\DC Folder, you can call it via the command line like this:

```
PowerShell.exe -executionpolicy remotesigned -File EncryptPassword.ps1 -uid "<UserID>" -pwd "<Password>"
```

This will create one file:

- <userid>.pw
  This file is bound to the system it was created on and cannot be used on other machines

To use the old encryption method, you can call it via the command line like this:

```
PowerShell.exe -executionpolicy remotesigned -File EncryptPassword.ps1 -uid "<Benutzer>" -pwd "<Passwort>" – mkey 1
```

This will create two files

- <userid>.key
- <userid>.pwd

Attention	The resulting files have to be placed in either the subdirectory where the actual connector resides or the parent directory!. The password handling module will first look in the subdirectory of the connector before traversing upwards to the parent directory (..\DC)

The Connector.config file needs to be edited and the **pwd="<Password>"** parameter has to be removed. After this the connector will use the encrypted password from the files.

Attention	Please note that you always need the .key AND the .pwd file otherwise decryption will not work. The encryption is reversible so it needs to be made sure that no unauthorized access to the machine is possible.

4.2 SQL based connectors

The settings and requirements for the database-based connectors are listed here.

4.2.1 Discovery Systems Data Connector (DSDC.exe)

The tool used to export inventory data from SQL databases can be found in the subfolder..\DataCollector\DC\DSDC of the Data Collector installation directory.

Specific details about the different connectors are given in the next few chapters.

It accepts the following parameters:

Connection.config	Description
srv="<Server>"	Name of SQL server
db="<Database>"	Name of inventory database
lnk="<LinkedServer>"	Optional: Linked Server name
uid="<User>"	User id for authentication (If not specified, integrated security will be used.)
pw="<Password>"	Password for above user

Connection.config	Description
t="<Inventory type>"	Type of inventory system: SCCM, LANDesk, Map, Discovery, GENERIC
not used	Export directory
sfx="<File suffix>"	Optional: Suffix for export files.
h="true"	Export of hardware scans with default filename.
f="true"	Export of file scans with default filename.
s="true"	Export of software scans with default filename.
tmp="<Tempdir>"	Optional: Directory used for temporary files.
kb="true"	Use this option to suppress Microsoft KB-updates in SoftwareScan.
ad="true"	Export of active directory users (Only available for the inventory type GENERIC)
dr="true"	Export of device relationships (Only available for the inventory type GENERIC)
m="true"	Export of metering (Only available for SCCM connector).
adg="true"	Export of active directory groups (Only available for the inventory type GENERIC)

Note If the DSDC application is to be run in the same user context as the Data Collector, e.g. for exporting data with a domain account instead of an SQL account, the /uid and /pw parameters in the Start.cmd created by the installer should be omitted.

It is not possible to specify domain accounts using the "uid" and "pw" parameters, only SQL user accounts are valid!

4.2.2 Suppress export of MAC and IP information (DSDC.exe.config)

Important Starting with the release 1.1805 the DSDC.exe is delivered with the DSDC.exe.config, with this file the export of MAC and IP information can be controlled.

The file has the following content:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
   <appSettings>
      <add key="IncludeIPAddressFields" value="false" />
      <add key="IncludeMacAddressFields" value="false" />
   </appSettings>
</configuration>
```

To activate the export of IP or MAC addresses, the respective value has to be set to "true".

4.2.3 Microsoft Endpoint Configuration Manager (MECM) formerly known as System Center Configuration Manager (SCCM)

Item	Description
Supported Versions	System Center - Configuration Manager SCCM 2007 to 2012 R2, Build 1511 bis Build 1810.2 Microsoft Endpoint Configuration Manager MECM 1902 to 2107
SQL role in database	db_datareader
Type Entry in Connector.config	t="SCCM"
Tables queried	[dbo].[SetupInfo] [dbo].[v_GS_ADD_REMOVE_PROGRAMS] [dbo].[v_GS_ADD_REMOVE_PROGRAMS_64] [dbo].[v_GS_COMPUTER_SYSTEM] [dbo].[v_GS_COMPUTER_SYSTEM_PRODUCT] [dbo].[v_GS_INSTALLED_SOFTWARE] [dbo].[v_GS_INSTALLED_SOFTWARE_MS] [dbo].[v_GS_LOGICAL_Disk] [dbo].[v_GS_Operating_System] [dbo].[v_GS_PC_BIOS] [dbo].[v_GS_PROCESSOR] [dbo].[v_GS_SoftwareFile] [dbo].[v_GS_SoftwareProduct] [dbo].[v_GS_SYSTEM_ENCLOSURE] [dbo].[v_GS_VIDEO_CONTROLLER] [dbo].[v_GS_WORKSTATION_STATUS] [dbo].[v_GS_X86_PC_MEMORY] [dbo].[v_LU_MSProd] [dbo].[v_R_System] [dbo].[v_R_System_Valid] [dbo].[v_RA_System_IPAddresses]

Note It is recommended to scan for .EXE files on the machines inventoried by System Center to further raise the quality of recognition.

Metering

Metering data can be exported from System Center and processed in Spider. With System Center , only defined files are collected.

Item	Description
Supported Versions	System Center - Configuration Manager SCCM 2012 to 2012 R2, Build 1511 bis Build 1810.2 Microsoft Endpoint Configuration Manager MECM 1902 to 2107
SQL role in database	db_datareader
Type Entry in Connector.config	m="True" f="True"
Tables queried	[dbo].[v_MeterData] [dbo].[v_Users] [dbo].[v_ProductFileInfo]

Custom SQL Server Edition Detection

In order to detect SQL server editions from System Center it is necessary to extend System Center so that it will query certain WMI namespaces on the SQL servers.

The System Center Administrator Console will only allow one namespace to be configured, but there are several namespaces that contain SQL information depending on the installed SQL server version.

Currently known are the namespaces:

- \root\Microsoft\SqlServer\ComputerManagement
- \root\Microsoft\SqlServer\ComputerManagement10
- \root\Microsoft\SqlServer\ComputerManagement11
- \root\Microsoft\SqlServer\ComputerManagement12
- \root\Microsoft\SqlServer\ComputerManagement13
- \root\Microsoft\SqlServer\ComputerManagement14

In order to query all of those an extension of System Center by the use of MOF files is required, a detailed description (including samples) can be found at:
https://sccm-zone.com/sql-version-detection-and-report-sccm-2012-r2-12f299b5e63b

Attention	Please note that the above mentioned website is not a website offered by brainwaregroup, the information contained on this website may change without notice or it may vanish altogether. Please consult your System Center specialist if you have any questions about the configuration of SCCM.

The results of the additional WMI query are likely to be put into additional tables in your System Center environment, since those tables may not have a consistent name throughout all System Center installations a special view on those tables is needed for the SQL recognition of the DSDC to work.

The name of the view that will be queried (only if it exists) has to be:

[dbo].[CUSTOM_SQLSERVEREDITION]

and has to contain the following columns:

Column	Column Type	NULL allowed
MachineID	int	NOT NULL
InstanceKey	int	NOT NULL
TimeKey	datetime	NOT NULL
RevisionID	int	NOT NULL
AgentID	int	NULL
rowversion	timestamp	NOT NULL
IsReadOnly00	int	NULL
PropertyIndex00	int	NULL
PropertyName00	nvarchar(255)	NULL
PropertyNumValue00	int	NULL
PropertyStrValue00	nvarchar(255)	NULL
PropertyValueType00	int	NULL
ServiceName00	nvarchar(255)	NULL
SqlServiceType00	int	NULL

An example of how the view is created is this:

```
CREATE VIEW [dbo].[CUSTOM_SQLSERVEREDITION] As
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_2017_Property_2_0_DATA]
UNION ALL
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_2016_Property_2_0_DATA]
UNION ALL
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_2014_Property_2_0_DATA]
UNION ALL
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_2012_Property_2_0_DATA]
UNION ALL
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_2008_Property_2_0_DATA]
UNION ALL
Select * FROM [CM_BWG].[dbo].[CUSTOM_SQL_Legacy_Property_2_0_DATA]
```

Attention	Please note that neither the table names nor the required columns may exist in your environment, for further information on how to inventory the necessary data please consult the above mentioned website or your System Center consultant.

4.2.4 Columbus Datacenter Inventory

The Columbus Datacenter Inventory imports data from the appliance.

Prerequisites

none

Configuration

Connector.config attribute	Description
server="<IP>"	IP-Address where the connector can query the data
share="<Share name>"	Share name on the specified server
uid="<User>"	UserID needed to connect with the share.
pwd="<Password>"	Password to the above UserID, this can be omitted if you use the PowerShell password encryption feature as described in: Password Encryption for PowerShell based Connectors (on page 60)

Examples

Export with UserID and password

```
<connector name="Columbus Datacenter Inventory" subfolder="DatacenterInventory" active="true"
scriptname="GetDatacenterInventory.ps1" uid="<UserID>" pwd="<Password> server="<Server IP>" share="<Sharename>" sfx="" />
```

Export with UserID and password stored in external file

```
<connector name="Columbus Datacenter Inventory" subfolder="DatacenterInventory" active="true"
scriptname="GetDatacenterInventory.ps1" uid="<UserID>" server="<Server IP>" share="<Sharename>" sfx="" />
```

4.2.5 Heat Discovery

Item	Description
Supported Versions	2014.2 and later
SQL role in database	db_datareader
Type Entry in Connector.config	t="HEAT"
Tables queried	[dbo].[CI] [dbo].[FRS_CIComponent] [dbo].[FRS_IM_FileInstance] [dbo].[FRS_IM_SoftwareFile] [dbo].[SoftwareIdentity] [dbo].[SoftwareType]

4.2.6 Frontrange Discovery

Item	Description
Supported Versions	9.3 and later
SQL role in database	db_datareader
Type Entry in Connector.config	t="DISC"
Tables queried	[dbo].[ClientType] [dbo].[Client] [dbo].[defOSType] [dbo].[defOS] [dbo].[Hardware] [dbo].[Manufacturer] [dbo].[OperatingSystem] [dbo].[Products] [dbo].[SoftwareAud] [dbo].[SoftwareFile] [dbo].[SoftwarePackage] [dbo].[SoftwarePath] [dbo].[SRDFile] [dbo].[SRDVersion] [dbo].[System]

4.2.7 Landesk

Item	Description
Supported Versions	9.x
SQL role in database	db_datareader
Type Entry in Connector.config	t="LANDESK"
Tables queried	[dbo].[AppSoftwareSuites] [dbo].[AppSoftware] [dbo].[BIOS] [dbo].[BoundAdapter] [dbo].[CompSystem] [dbo].[Computer] [dbo].[FileInfo] [dbo].[LogicalDrives] [dbo].[Memory] [dbo].[NetworkSoftware] [dbo].[Operating_System] [dbo].[Processor] [dbo].[VideoAdapter]

4.2.8 Lansweeper

Item	Description
Supported Versions	x
SQL role in database	db_datareader
Type Entry in Connector.config	t="Lansweeper"
Tables queried	[dbo].[tblAssetCustom] [dbo].[tblAssets] [dbo].[tblBIOS] [dbo].[tblComputersystem] [dbo].[tblComputerSystemProduct] [dbo].[tblDiskdrives] [dbo].[tblOperatingsystem] [dbo].[tblProcessor] [dbo].[tblSoftware] [dbo].[tblSoftwareUni] [dbo].[tblSystemEnclosure] [dbo].[tblVideoController] [dbo].[tblSqlServers]

4.2.9 Altiris 7

Item	Description
Supported Versions	7.5

Item	Description
SQL role in database	db_datareader
Type Entry in Connector.config	t="ALT7"
Tables queried	[dbo].[Inv_AddRemoveProgram] [dbo].[vComputer] vc [dbo].[vHWBaseboard] [dbo].[vHWChassis] [dbo].[vHWComputerSystem] [dbo].[vHWDisplayController] [dbo].[vHWProcessor] [dbo].[vSWBIOSElement]

4.2.10 Generic Connector

In Generic mode, the DSDC expects a series of stored procedures to exist in the given database, in order to export data. It will purely rely on the data provided by the stored procedures.

Item	Description
SQL role in database	db_datareader on Tables used Execute rights on Stored Procedures
Type Entry in Connector.config	t="Generic"
Stored Procedures	[dbo].[swrGetWorkList] [dbo].[swrGetHardwareScan] [dbo].[swrGetFileScan] [dbo].[swrGetSoftwareScan] [dbo].[swrGetDeviceRelationship] [dbo].[swrGetADUserObject] [dbo].[swrGetADGroupObject] [dbo].[swrGetADGroupMember] [dbo].[swrGetSwidScan]

Attention Please make sure that the user used to connect to the database is granted the "EXECUTE" right to the stored procedures that are created.

A set of templates can be downloaded from: https://docs.flexera.com/Spider64/GenericDataConnectorTemplates.zip

All of these procedures have mandatory and optional columns all of which and their formats are explained in Generic Connector Stored Procedures (on page 141)

4.2.11 Microsoft Assessment and Planning Toolkit (MAP)

Item	Description
Supported Versions	8.5 9.0, 9.1, 9.2, 9.4
SQL role in database	db_datareader
Type Entry in Connector.config	t="MAP"
Tables queried	[AllDevices_Assessment].[CategorizedDevices] [AllDevices_Assessment].[HardwareInventoryCore] [AllDevices_Assessment].[HardwareInventoryEx] [Core_Inventory].[Devices] [SqlServer_Assessment].[SqlInstances] [UT_Exchange_Inventory].[AdServers] [Win_Assessment].[VirtualMachinesView] [Win_Assessment].[WindowsInstalledSoftwareFull] [Win_Inventory].[ComputerSystemProduct] [Win_Inventory].[DataFile] [Win_Inventory].[LogicalDisks] [Win_Inventory].[Processors] [Win_Inventory].[VideoControllers] For additional recognition the user connecting to the MAP database must be granted the Execute right to the following functions: [UT_Exchange_Reporting].[GetExchangeEditionFromTypeValue] [UT_Exchange_Reporting].[GetExchangeProductNameFromVersionNumber] [UT_SCCM_Reporting].[GetSccmProductName]

The Microsoft Assessment and Planning Tool (MAP) can be installed in two different ways.

The first method is to install it using the bundled application specific and on-demand version of SQL Server known as LocalDB. The benefit of a LocalDB installation is the low time required for the implementation of MAP as everything is self-contained and dependent on the client's speed, can be ready to scan in under five minutes. The drawback is that unfortunately other users will not be able to access the database or indeed that installation of MAP. It also means that the Data Collector will need to be configured to run in the context of the user who installed MAP, as well as requiring some other additional configuration to determine the named pipe used for the connection to the MAP database held on the LocalDB SQL Server

The second method that is recommended by brainwaregroup is to install a SQL Server Express Edition Instance on the machine and configuring it with the following settings during setup:

Instance Name: **MAPS**

SQL Server Collation: **SQL_Latin1_General_CP1_CI_AS**

Installing a SQL Server using the aforementioned settings before executing the MAP setup will then enable MAP to use a fully configurable version of SQL Server that runs as a Windows service when the machine starts and can easily be configured to allow multiple users to access it. This provides a number of benefits when using the Data Collector to collect MAP data as the ease of configuring the system does not require having to perform any queries for the named pipes of the databases on the LocalDB server, nor configure the Data Collector to trigger a server start when it needs to access the database.

If the user used for the export is not an administrator on the database, you need to grant Execute permission for Software Scan Export to work to the following:

- Functions
  - [UT_Exchange_Reporting].[GetExchangeEditionFromTypeValue]
  - [UT_Exchange_Reporting].[GetExchangeProductNameFromVersionNumber]
  - [UT_SCCM_Reporting].[GetSccmProductName]

Important	The above configuration of the SQL instance should be performed during the setup and not configured post setup. Whilst it is possible to have different SQL database collations running on a single SQL Server, the MAP pre-requisite checker will look for the SQL Server collation to be set above and will fail to install if this is not the case.

4.2.12 Matrix42 (beta)

Item	Description
Supported Versions	TBD
SQL role in database	db_datareader
Type Entry in Connector.config	t="MATR"
Tables queried	[dbo].[SPSApplicationClassBase] [dbo].[SPSAssetClassBase] [dbo].[SPSAssetPickupType] [dbo].[SPSComputerClassBase] [dbo].[SPSComputerClassGraphicCard] [dbo].[SPSComputerClassGraphicCard] [dbo].[SPSComputerClassHardDisk] [dbo].[SPSComputerClassOS] [dbo].[SPSInventoryClassApplication] [dbo].[SPSStockKeepingUnitClassBase] [dbo].[SPSSupplierClassBase] [dbo].[SPSUserClassbase]

4.2.13 Empirum Workplace Management (beta)

Item	Description
Supported Versions	15.2
SQL role in database	db_datareader
Type Entry in Connector.config	t="EMPI"
Tables queried	[dbo].[DMISystem] [dbo].[InvComputer] [dbo].[InvFiles] [dbo].[InvSoftware] [dbo].[WMIBattery] [dbo].[WMIProcessor]

4.2.14 Baramundi (beta)

Item	Description
Supported Versions	TBD
SQL role in database	db_datareader
Type Entry in Connector.config	t="BARA"
Tables queried	[dbo].[machine] [dbo].[inventory_nodes] [dbo].[inventory_nodeproperties] [dbo].[InventoriedSoftware] [dbo].[SwDetectionRule]

4.2.15 Snow (beta)

Item	Description
Supported Versions	TBD
SQL role in database	db_datareader
Type Entry in Connector.config	t="SNOW"
Tables queried	[dbo].[vClient] [dbo].[vDisplayAdapter] [dbo].[vLogicalDisks] [dbo].[vMemory] [dbo].[vNetworkAdapter] [dbo].[vOperatingSystem] [dbo].[vProcessor] [dbo].[vSoftware]

4.2.16 Overview of discovered Items

The different Exports and Inventory Tools report back a different set of hardware items, which export/tool exports which inventory items can be found in the table below.

Manufacturer	brainwaregroup Inventory		Microsoft			Ivanti		Lansweeper	Landesk
Product	Scanner	Agent	MAP	SCCM	SCCM 2012	Discovery	Heat		
DomainName	•	•	• (1)	• *1	• (1)	• (1)	• (1)	- (1)	• (1)
BIOSDate	•	•	•	•	•	•	•	•	•
BIOSVendor	•	•	•	•	•	•	•	•	•
BIOSVersion	•	•	•	•	•	•	•	•	•
ChassisType	-	-	-	•	•	-	•	-	-
CorePerCPU	-	-	•	-	•	•	-	•	•
CPUArchitecture	•	•	•	•	•	-	•	•	•
CPUCoreCount	•	•	•	-	•	-	•	-	•
CPUCount	•	•	•	•	•	•	•	•	•
CPULogicalCount	•	•	•	-	•	-	-	•	-
DeviceChassis	•	•	•	-	-	•	-	•	•
DiskFreeMB	•	•	•	•	•	•	•	•	•
DiskTotalMB	•	•	•	•	•	•	•	•	•
DomaineNameNetBios	•	•	- (1)	• (1)	• (1)	- (1)	- (1)	• (1)	- (1)
GraphicAdapter	•	•	-	•	•	•	•	•	•
GraphicMemoryMB	•	•	-	•	•	•	•	•	-
HostName	•	•	• (1)	• (1)	• (1)	• (1)	• (1)	• (1)	• (1)
IPAddressV4	•	•	•	-	•	•	•	•	•
IPAddressV6	•	•	-	-	•	-	-	-	-
LastLoggedOnUser	•	•	•	•	•	•	•	•	•
MAC1	•	•	-	-	-	•	•	•	•
MAC2	•	•	-	-	-	•	-	-	-

Manufacturer	brainwaregroup Inventory		Microsoft			Ivanti		Lansweeper	Landesk
MAC3	•	•	-	-	-	•	-	-	-
MAC4	•	•	-	-	-	-	-	-	-
Manufacturer	•	•	•	•	•	•	•	•	•
MemoryMB	•	•	•	•	•	•	•	•	
Model	•	•	•	•	•	•	•	•	•
OSCaption	•	•	•	•	•	•	•	•	•
OSClass	•	•	-	-	-	•	•	•	•
ProcessorManufacturer	•	•	•	•	•	•	•	•	•
ProcessorSpeed	•	•	•	•	•	•	•	•	•
ProcessorType	•	•	•	•	•	•	•	•	•
ScanDate	•	•	•	•	•	•	•	•	•
Serial	•	•	•	•	•	•	•	•	•
UUID	•	•	• (1)	• (1)	• (1)	• (1)	• (1)	• (1)	- (1)
Worklist									
Identifier	-	-	•	•	•	•	•	•	•
UUID	•	•	•	•	•	•	•	•	-
URN	-	-	-	-	-	-	-	-	-
Hostname	•	•	•	•	•	•	•	•	•
DomainName	•	•	•	•	•	•	•	-	•
DomainNameNetBios	•	•	-	•	•	-	-	•	-
Software Scan	•	•	•	•	•	•	•	•	•
SQL Server Edition	•	•		•	•	-	-	•	-
Autodesk	•	•		-	-	-	-	-	-
Embedded OS	•	•		-	-	-	-	-	-
File Scan	•	•	-	•	•	•	•	-	•
Metering	-	•	-	•	•	-	-	-	-

Manufacturer	brainwaregroup Inventory		Microsoft			Ivanti		Lansweeper	Landesk
SWID Tags	-	-	•	•	•	-	-	-	-

Beta Connectors

Manufacturer	Snow	Matrix42		Baramundi
Product	**Snow Inventory**	**Matrix42**	**Empirum**	**Baramundi**
DomainName	• (1)	• (1)	• (1)	• (1)
BIOSDate	•	•	•	•
BIOSVendor	•	•	-	•
BIOSVersion	•	•	•	•
ChassisType	-	-	-	-
CorePerCPU	•	-	-	•
CPUArchitecture	-	-	•	-
CPUCoreCount	•	-	•	•
CPUCount	•	•	•	•
CPULogicalCount	•	•	•	•
DeviceChassis	•	•	•	•
DiskFreeMB	•	-	-	•
DiskTotalMB	•	•	-	•
DomaineNameNetBios	• (1)	• (1)	• (1)	-
GraphicAdapter	•	•	•	-
GraphicMemoryMB	-	•	-	-
HostName	• (1)	• (1)	• (1)	• (1)
IPAddressV4	•	•	-	•
IPAddressV6	-	-	-	-
LastLoggedOnUser	•	-	•	•
MAC1	•	•	•	•

MAC2	-	-	´-	-
MAC3	-	-	-	-
MAC4	-	-	-	-
Manufacturer	•	•	•	•
MemoryMB	•	•	•	•
Model	•	•	•	•
OSCaption	•	•	•	•
ProcessorManufacturer	•	-	•	•
ProcessorSpeed	•	•	•	•
ProcessorType	•	•	•	•
ScanDate	•	•	•	•
Serial	•	•	•	•
UUID	• (1)	• (1)	• (1)	-
Worklist				
Identifier	•	•	•	• (1)
UUID	-	-	•	-
URN	-	-	-	-
Hostname	•	•	•	• (1)
DomainName	•	•	•	• (1)
DomainNameNetBios	-	•	•	-
Software Scan	•	•	•	•
SQL Server Edition	-	-	-	-
Autodesk	-	-	-	-
Embedded OS	-	-	-	-
File Scan	•	-	•	-
Metering	-	-	-	-
SWID Tags	-	-	-	-

(1) - Data is taken from Worklist.

4.3 API based connectors

4.3.1 Introduction

Proxy Usage

Some connectors need to access the internet in order to fulfill their tasks, these connectors are:

- Microsoft Azure
- Adobe Online

In order the enable these connectors to access the internet via a proxy, you can specify additional attributes for these connectors in the Connector.config

Configuration

Attribute	Mandatory	Description
proxyAddress=<Address of the proxy server>	Yes	Address of the proxy server, must include http(s)://
proxyPort =<Port of the proxy server>	Yes	Port that the proxy server listens for connections
proxyUser=<Username>	No	If needed User Id to authenticate against the proxy server.
proxyUserPassword=<Password>	No	Password for above user.

Note: The password of the proxy authentication user can be encrypted using the standard encryption supplied for the PowerShell connectors, please see Password Encryption for PowerShell based Connectors (on page 60) for details.

Debugging PowerShell Connector Execution

In some cases even after evaluation of the log files, it might become necessary to view a single connectors execution to determine the exact error why it fails.

The best was to do this is to execute the connector outside of the general management that the Data Collector provides through the ExectuteDCs.ps1.

To execute a given connector individually, create a batch file (e.g. <ConnectorName>.cmd) in the directory of the connector at question. Inside the batch file enter the following

```
PowerShell -File .\<Name of the Connectors PowerShell script>.ps1 -dir "<Output directory for the result>" <AdditionAttributes>
Pause
```

Execute the batch file and look for errors.

<AdditionAttributes> can be taken directly from the Connector.config, for example:

Line in Connector.config

```
<connector name="GetADGroupObjects" subfolder="ADConnector" active="true" scriptname="GetADGroupObjects.ps1"
dc="dc01.domain.ocal" sfx="GroupObjects" grp="ManagementGroup,Developers,Sales" strict="false" queryParentGroup="true" />
```

you can omit the attributes "name", "subfolder", "active", "scriptname", all others (if used, some may not be used), for attributes containing a value, prefix the attribute with a "-" and replace the " with a space, and you will get

```
-dc "dc01.domain.ocal" -sfx "GroupObjects" -grp "ManagementGroup,Developers,Sales"
```

for prefixes containing a true or false value, if the attribute value is "true" please prefix the attribute name with a "-" and delete everything starting from the "=", so "strict="false" queryParentGroup="true"" becomes "-queryParentGroup"

So the complete string for the line starting with "PowerShell" will be:

```
PowerShell -File .\GetADGroupObjects.ps1 -dir "<Output directory for the result>" -dc "dc01.domain.ocal" -sfx "GroupObjects" -grp
"ManagementGroup,Developers,Sales" -queryParentGroup
```

Attention:	For <Output directory for the result> its best to us the connectors directory, this way you can immediately see the resulting file, without navigating to the directory you might have specified otherwise.

PSRemoting - Execute commands on remote computers

Some connectors utilize remote sessions in PowerShell to execute commands on remote machines, this is done for PowerShell Modules that are not available on the server the SDC is installed on.

Details about activation and troubleshooting can be found on the Microsoft website: Enable-PSRemoting

A simple example to test the connection can look like this:

```
$computer = "<Remote server>"
$username = "<User to connect to remote server>"
$password = "<Password for above user>"

$psSessionSplat = @{
   computerName = $computer
}
```

```
if($username -and $password) {
   $pw = convertto-securestring -AsPlainText -Force -String $password
   $cred = new-object -typename System.Management.Automation.PSCredential -argumentlist $username,$pw
   $psSessionSplat.Credential = $cred
}

Write-Host @psSessionSplat -ForeGroundColor Yellow
$session = new-pssession @psSessionSplat
Get-PSSession
Remove-PSSession $session
```

PowerShell Execution Policy

Windows PowerShell execution policies let you determine the conditions under which Windows PowerShell loads configuration files and runs scripts.

In some cases you need to adjust your PowerShell Execution Policy, please refer to About Execution Policies to find out which policy is suitable for you, our scripts have been tested with setting the policy to "RemoteSigned".

The current settings of the Execution Policy can be determined by issuing the following command:

```
Get-ExecutionPolicy -List | fl
```

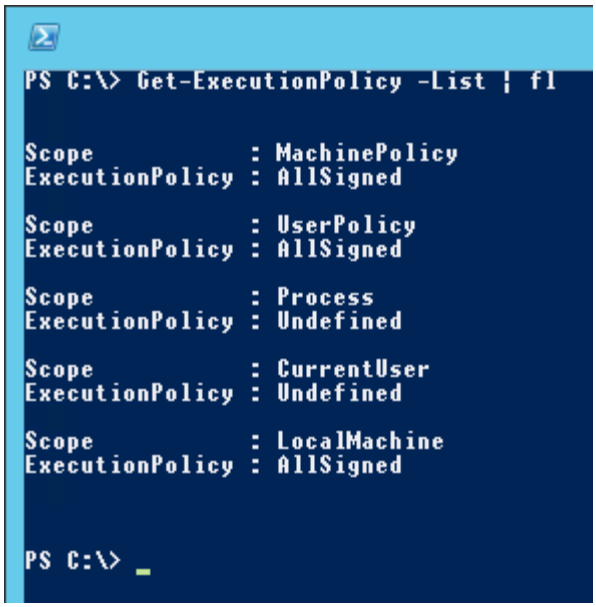


Figure - ExecutionPolicy

In order to change your execution policy you can execute the following in a PowerShell command line:

```
Set-ExecutionPolicy RemoteSigned
```

4.3.2 VMWare vCenter / ESX Server

Prerequisites

In order for the VCenter Connector to work, it is necessary to install the PowerCLI offered by VMWare.

Attention	Starting with version 6.5.1 the PowerCLI will not be provided as an MSI anymore by VMWare, but is offered as a download via the PowerShell Gallery. Details for the installation can be found here: https://blogs.vmware.com/PowerCLI/2017/04/powercli-install-process-powershell-gallery.html It is recommended to keep the PowerCLI up-to-date.

Please note, that in order for the service to be able to use the installed PowerCLI, you need to install the PowerCLI for "AllUsers"

```
Install-Module -Name VMware.PowerCLI –Scope Allusers
```

Sometimes other Modules will have a name that is also used by the PowerCLI, in order to "overwrite" these names you need to add the -AllowClobber Parameter when installing from the PowerShell Gallery

```
Install-Module -Name VMware.PowerCLI –Scope Allusers -AllowClobber
```

Attention	If the certificate or certificate chain is not correct, the resulting warning can be disable by issuing the following in a PowerShell window: **Set-PowerCLIConfiguration -InvalidCertificateAction "Ignore" -Scope AllUsers**

Standard Connector

- This connector will query hardware, software and device relationship information from either a vCenter or standalone ESX server.

For improved support of VDI, with client operating systems (such as “Windows 10”, “Windows 8” or “Windows 7”) the device identification takes place via host name / domain name. In this case no “UUID” is exported.

Configuration

Connector.config attribute	Description
srv="<Server>"	Servername of the vCenter server.
port="<port>	Port of the vCenter or ESX server if it is not port 443.
uid="<User>"	User that is authorized to query the VMware vCenter
pwd="<Password>"	Password of above user.
h="true"	Export hardware related inventory information
dr="true"	Export relationships of devices
s="true"	Export of host license information
SerialNumber="true"	Export serial number of ESX host machines
NoGuests	No Export of guest systems
OnlyWindows	Export Windows Guests only

Examples

```
<connector name="DSDC vCenter Inventory" subfolder="vCenter" active="true" scriptname="GetvCenter-Details.ps1"
srv="vcenter.domain.com" -port="443" uid="domain\username" pwd="password" h="true" s="true" dr="true" onlyWindows=true
sfx="suffix" />
```

Attributes

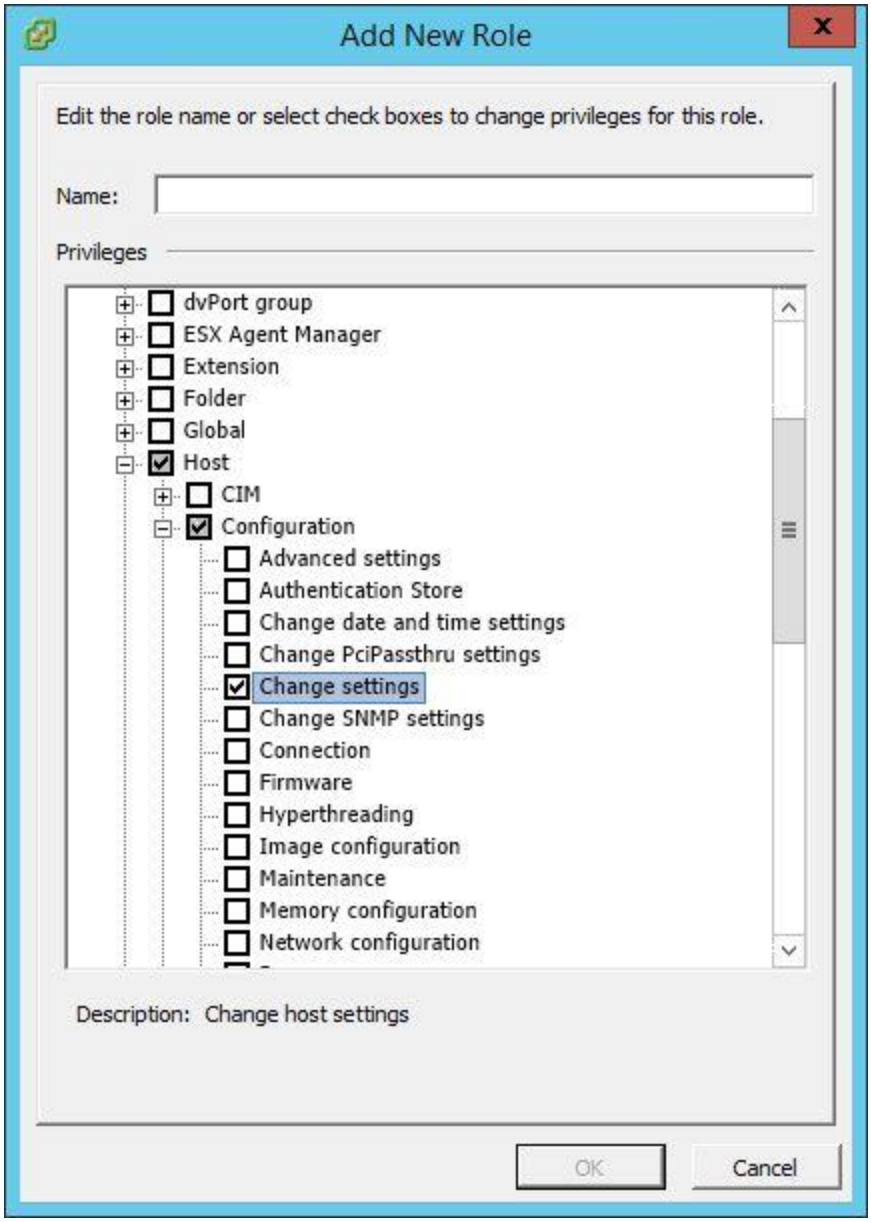
The vCenter Connector queries the following values from a vCenter or ESX server.

- Hardware (of the ESX Hosts)
  - UUID
  - Urn
  - DomainName
  - HostName
  - DomainNetBIOS
  - Manufacturer
  - ScanDate
  - Model
  - Mac1
  - Mac2
  - Mac3
  - Mac4
  - ProcessorManufacturer
  - ProcessorType
  - ProcessorSpeed
  - CPUCount
  - CPUCoreCount
  - CorePerCPU
  - CPULogicalCount
  - DiskTotalMB
  - DiskFreeMB
  - MemoryMB
  - IPAddressV4
  - IPAddressV6
  - OSCaption
  - BIOSVersion
  - BIOSDate
  - InventorySource
  - Class
- Relations
  - DeviceRelationshipTypeID (1 for ESX - Guest relation, 2 for Cluster - ESX relation)
  - ChildDeviceUUID
  - ParentDeviceUUID
  - ParentDeviceURN
  - ScanDate

Serial Number export of ESX host machines

The export of serial numbers for ESX host machines is only possible if the hosts runs ESX Version 5 or above.

In addition, the user account carrying out the export needs a special right to be able to access this information. A simple Read Only user as before is not sufficient for the export of the serial numbers. The user has to be assigned the privilege **"Host.Configuration.Change Settings"**



The export of the serial numbers is only available if the parameter **SerialNumber="True"** is specified in the Connector.config for the vCenter export.

VCenter Diagnostic Tool

The Powershell based tool collects information about the following items, including their status:

- Connected VM Hots
- Disconnecd VM Hosts
- VM Hosts in Maintenance
- Not responding VM Hosts

- running Guests
- Not Runing Guests
- "Powered ON" related VMs
- "Powered Off" related VMS
- Connected Clusters (Master /Child cluster/hosts)

Configuration

Connector.config attribute	Description
srv="<Server>"	Servername of the vCenter server.
port="<port>	Port of the vCenter or ESX server if it is not port 443.
uid="<User>"	User that is authorized to query the VMware vCenter
pwd="<Password>"	Password of above user.

The VCenter Diagnostic Tool queries the versions ob the PowerShell and PowerCli modules.

The hardware scan exports connected hosts and running guest only.

Connector for Datacenter-Module

For the use with the Datacenter-Module this additional connector is necessary to report the ESX/vCenter data.

The connector consists of two scripts, one is used to query the data from the ESX/vCenter servers, the other is used to pack the results into the .CDC format. The data will be transferred to the Datacenter-Module.

Configuration - Query ESX/vCenter data

Connector.config Attribute	Description
server="<Server>"	Name of vCenter or ESX server
port="<port>	Port of vCenter or ESX server
username="<Username>"	Username to connect with
password="<Password>"	Password for above user

Configuration - Packing the results

This connector has no parameters.

Example

```
<connector name="vCenter Inventory" subfolder="vCenter" active="true" scriptname="get_esxihosts_vm.ps1"
server="vcenter.domain.com" -port="443" username="domain\username"  password=password"  />
<connector name="vCenter Inventory novaratio CDC" subfolder="vCenter" active="true" scriptname="get_esxihosts_vm_to_CDC.ps1" />
```

Attention: The script get_esxihosts_vm_to_CDC.ps1 MUST always be executed AFTER the connector that queries the data, a change in order will prevent the results to be packed correctly!

4.3.3 Adobe Online

The Adobe Online Connector is used to export user based licensing data from Adobe Online, e.g. Photoshop CC, Illustrator CC, All Apps Plan.

The connector version one is automatically updated to version two. The older version is still shipped as "Version legacy" and is intended for environments that conflict with version two. The Adobe API used in version one is deprecated.

Prerequisites

Attention	For the connector to work it is necessary that the Visual C++ 2010 Redistributable Package (x64) is installed. The Package can be downloaded from: Microsoft Visual C++ 2010 Redistributable Package (x64) .

Attention	In order to create an Integration an Enterprise ID with administrative privileges is needed. Details can be found here: https://www.adobe.io/apis/cloudplatform/console/authentication/gettingstarted.html "Creating an Integration" -> "Service Account Authentication".

Configuration

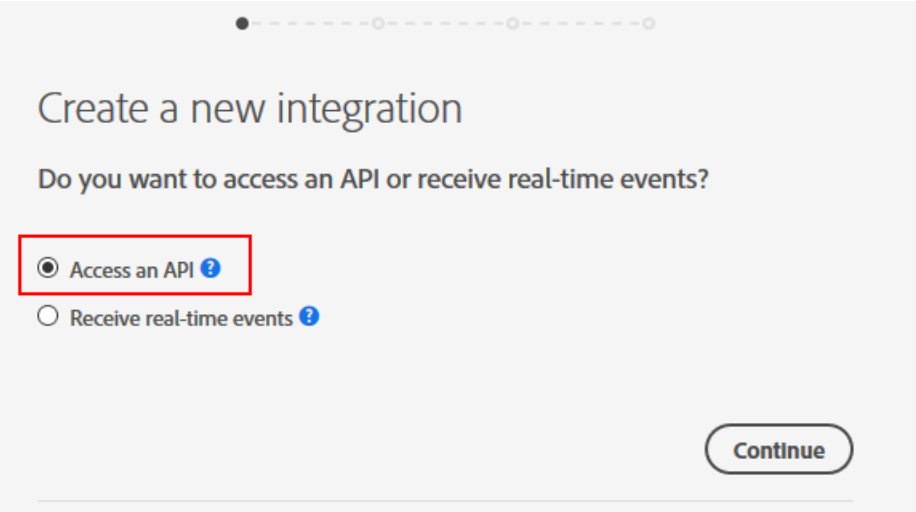
Connector.config attribute	Mandatory	Description
organizationID="<organizationID>"	Yes	organizationID retrieved from the adobe.io console
technicalAccountID="<technicalAccountID>"	Yes	technicalAccountID retrieved from the adobe.io console
apiKey="<apiKey>"	Yes	apiKey retrieved from the adobe.io console
clientSecret="<clientSecret>"	Yes	clientSecret retrieved from the adobe.io console
privateKeyName="<privateKeyName>"	Yes	Name of the file holding your private key.
privateKeyPassword="<privateKeyPassword>"	Yes	Optional: If your key file is protected by a password, please specify here.
proxyAddress="<Proxy>"	No	Address of proxy server (if needed) to connect to the internet. Proxy address including http(s)://
proxyPort="<Port>"	No	Port of proxy server
proxyUser="<User>"	No	User that can authenticate against proxy server, if needed.
proxyUserPassword="<Password>"	No	Password for above user.
TLS="<True|False>"	No	enable|disable encryption

Configuration of Integration

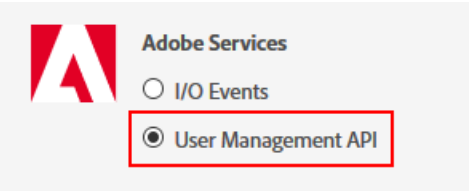
For the connector to work you have to setup an integration in the adobe.io console, to do so please follow these steps:

1. Log on to https://console.adobe.io/integrations

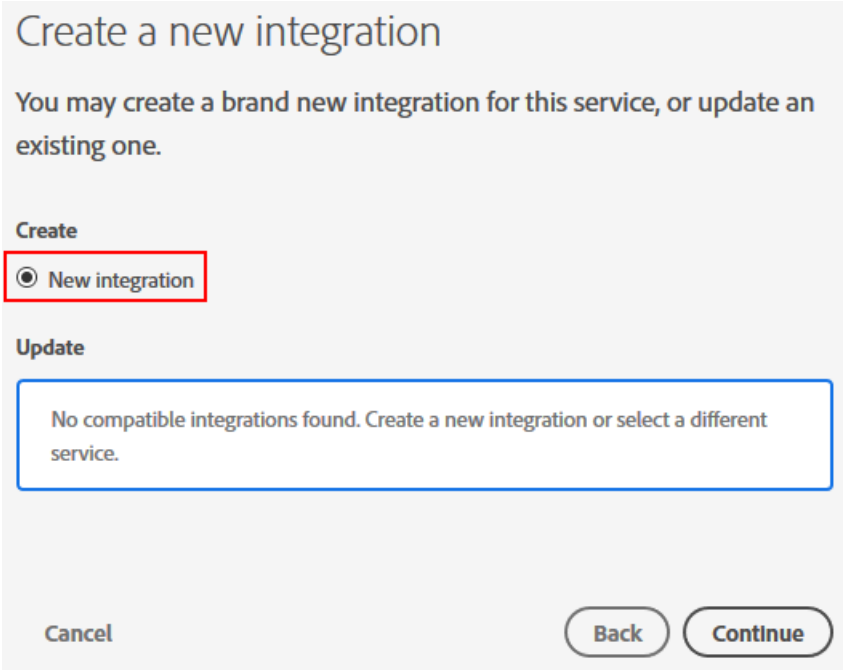
2. Click on "New Integration"
3. Choose **Access an API > Continue**



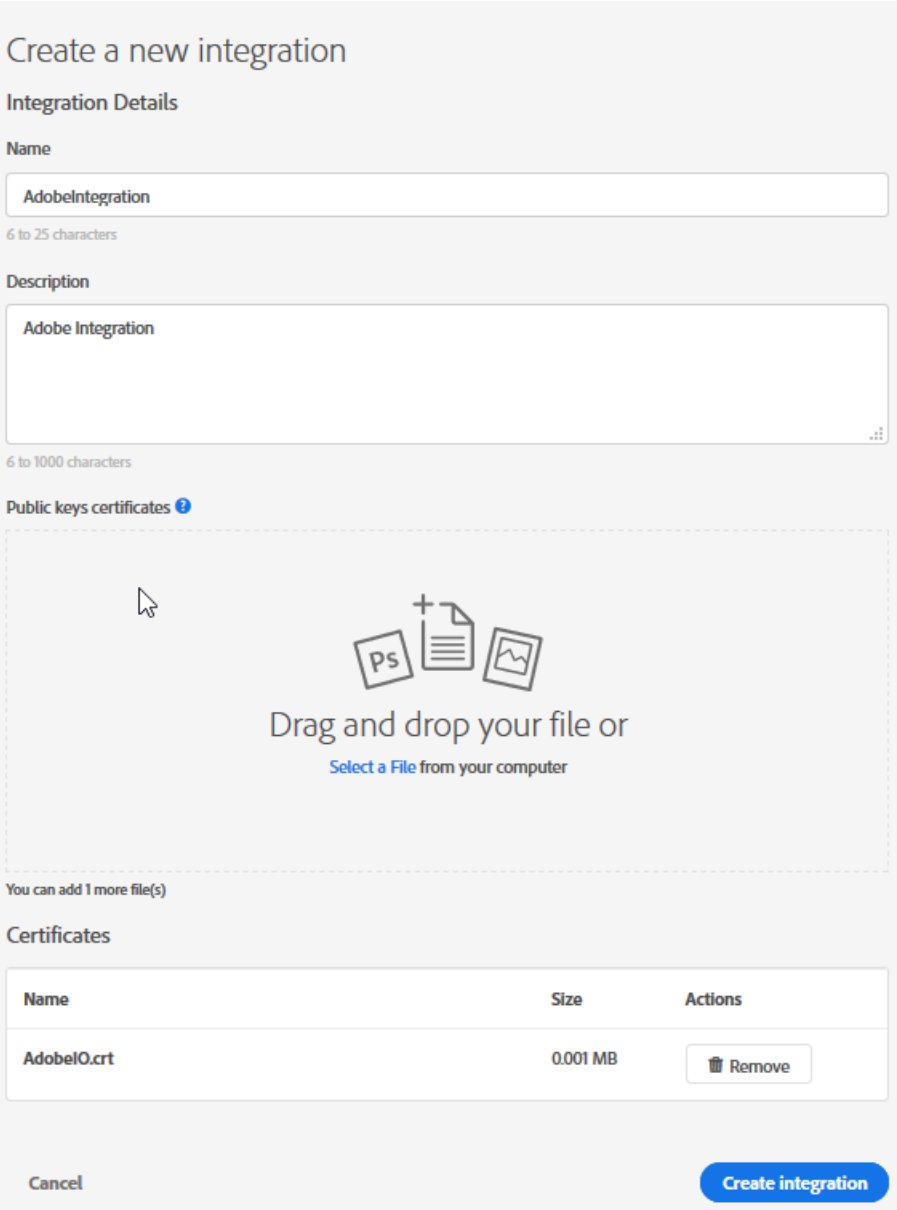
4. Choose **Adobe Services > User Management API > Continue**



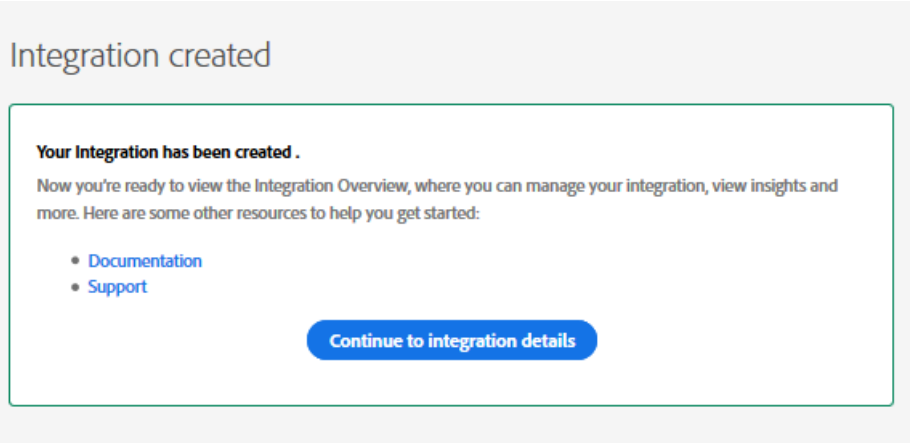
5. Choose **Create > New Integration > Continue**



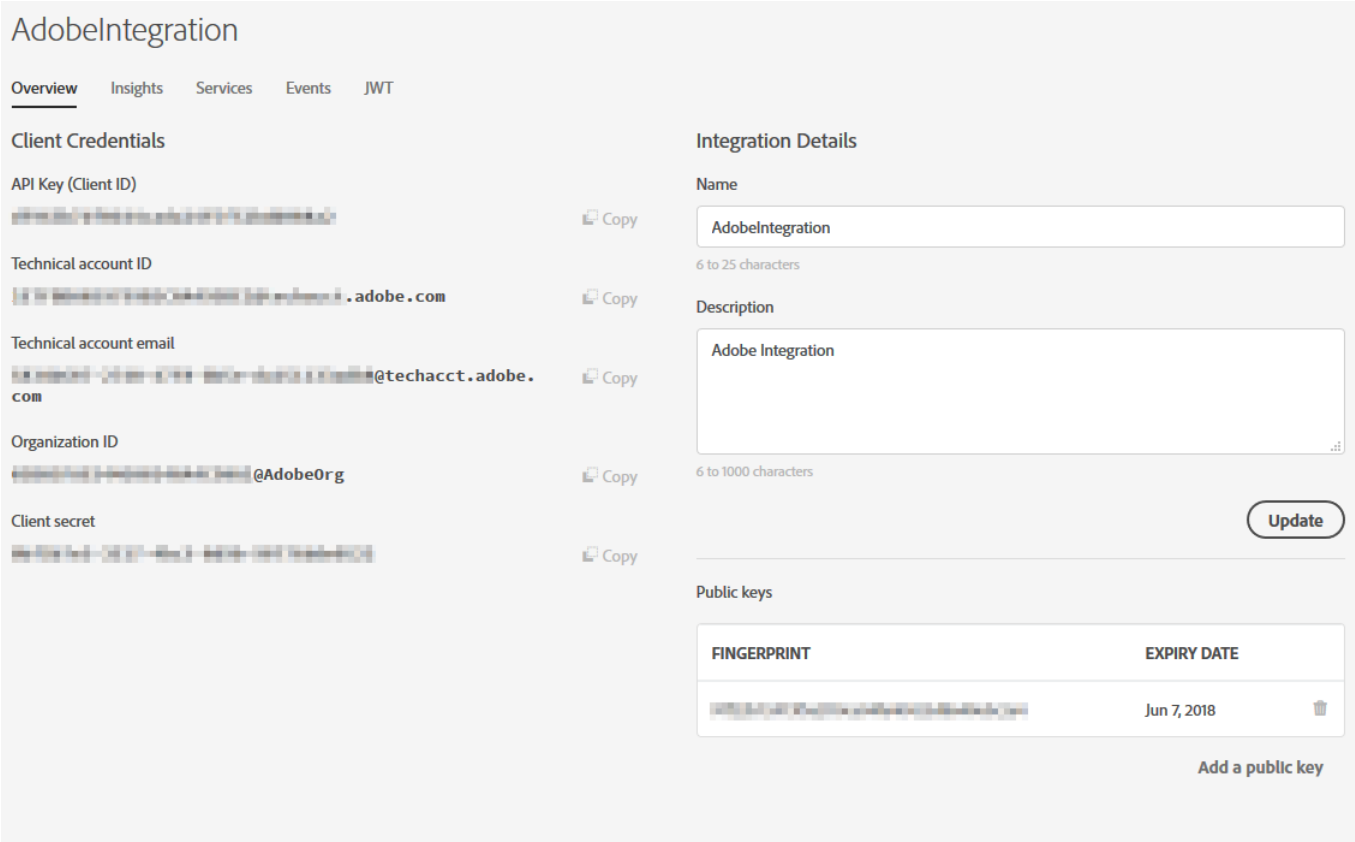
6. **Enter a Name > Enter a Description > Choose public key certificate > Create Integration**



7. Done



Now the integration is done and you can configure the Data collector with the values given by the Integration



Attention	The Setup will let you create your own key pair if you do not have one available, please start the setup and choose the Adobe Online Connector, it will then let you create the certificate and then you can continue setting up the integration. The certificate is placed in the same folder that you started the setup from. After the integration is created, you can then use the values to continue the setup. In case you want to utilize a password protected key file, please continue the setup and create another key pair using the "CreateCertificateAndKeyForAdobeCloud.cmd" in the OpenSSL subfolder of the Adobe Connector. Do not forget to adjust the key file in the Adobe Connector directory, the certificate on the Adobe web site and (if utilized) the password in the configuration.

4.3.4 Microsoft Azure (Microsoft Online)

Prerequisites

Attention	For a user account to be able to retrieve the information, it is only necessary to assign the Directory Role "User" in the Azure user account. For an application to be able to retrieve the information, it needs to have "Read directory data" defined in the "APPLICATION PERMISSIONS" section of the application itself.

For the connectors to work the Windows Module **"AzureAD"** has to be installed

The following PowerShell command can be used to check and install the AzureAD module

```
if (!(Get-Module -ListAvailable | where { $_.Name -eq "AzureAD"}))
{
        Install-Module AzureAd
}
```

A script containing the above code called InstallModule_AzureAD.ps1 is copied to the Microsoft-Azure-Connector folder.

Attention	Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration for all connectors

Note	Microsoft Azure AD consists of multiple connectors, common configuration is described here.

Connector.config Attribute	Mandatory	Description
uid="<User>"	No	User that can query the portal
pwd="<Password>"	No	Password of above user
TenantId="<TenantID>"	No	TenantID for access via an application
ApplicationId="<ApplicationID>"	No	ApplicationID for access via an application
CertificateThumbprint="<ThumbPrint>"	No	Thumbprint for access via an application
proxyAddress="<Proxy>"	No	Address of proxy server (if needed) to connect to the internet. Proxy address including http(s)://
proxyPort="<Port>"	No	Port of proxy server
proxyUser="<User>"	No	User that can authenticate against proxy server, if needed.
proxyUserPassword="<Password>"	No	Password for above user.

Attention	Either "uid" and "pwd" OR "TenantID", "ApplicationID" and "Thumbprint" have to be specified, one method HAS to be specified, BOTH may not be specified.

Examples in Connector.config:

Access by application

```
<connector name="Microsoft-AzureAD" subfolder="Microsoft-Azure-Connector" active="true"
scriptname="Microsoft-AzureAD-Connector.ps1" TenantId="abc" ApplicationId="def" CertificateThumbprint="xyz" sfx="" />
```

Access by user

```
<connector name="Microsoft-AzureAD" subfolder="Microsoft-Azure-Connector" active="true"
scriptname="Microsoft-AzureAD-Connector.ps1" uid="my.user@domain.com" pwd"..." sfx="" />
```

Microsoft AzureAD - User based license information

The AzureAD License Connector is used to export user based licensing data from Microsoft Azure.

E.g. Office 365, EMS, Visio 365, Project 365, ...

This connector replaces the formerly used Microsoft Online Connector.

Configuration

Connector.config Attribute	Mandatory	Description
additionalLicenseDetails="<True|False>"	No	Query additional license details

The Connector can be configured to only export certain attributes of a Tenant AD object; this is done in the file **Microsoft-AzureAD-Connector-TenantAttributeConfig.txt** in the same path as the .ps1 script.

The Microsoft-AzureAD-Connector-TenantAttributeConfig.txt is a csv type file.

Attention	Please note that no changes are allowed to be made in the ADAttribute, SWRDAttribute and Type columns.

If you want to disable the export of a certain attribute, set the value in the process column to "Disabled"

ADAttribute,SWRDAttribute,Type,Process

ADAttribute,SWRDAttribute,Type,Process

DisplayName,DisplayName,System.String,Enabled

ObjectId,ObjectId,System.GUID,Enabled

DirSyncEnabled,DirSyncEnabled,System.String,Enabled

ObjectType,ObjectType,System.String,Enabled

PreferredLanguage,PreferredLanguage,System.String,Enabled

PostalCode,PostalCode,System.String,Disabled

CountryLetterCode,CountryLetterCode,System.String,Disabled

City,City,System.String,Disabled

State,State,System.String,Disabled

Country,Country,System.String,Disabled

TelephoneNumber,TelephoneNumber,System.String,Disabled

Street,Street,System.String,Disabled

Microsoft AzureAD - User Export

The Azure Active Directory User Export is a connector used to export all (!) users from your Azure Active Directory.

The Connector can be configured to only export certain attributes of an AD object; this is done in the file **Microsoft-AzureAD-GetUsersAttributeConfig.txt** in the same path as the .ps1 script.

The Microsoft-AzureAD-GetUsersAttributeConfig.txt is a csv type file.

Attention	Please note that no changes are allowed to be made in the SWRDAttribute and Type columns.

If you want to disable the export of a certain attribute, set the value in the process column to "Disabled"

ADAttribute,SWRDAttribute,Type,Process

OnPremisesSecurityIdentifier,ObjectSid,System.String,Enabled

AccountEnabled,UserAccountControl,System.Int32,Enabled

UserPrincipalName,UserPrincipalName,System.String,Enabled

Mail,EmailAddress,System.String,Enabled

Important	User accounts exported from Azure Active Directory DO NOT contain the SamAccountName, please consider this when using this connector.

Microsoft AzureAD - Group Export

The Azure Active Directory Group Export is a connector used to export all specified groups and their members from your Azure Active Directory.

Configuration

Connector.config attribute	Mandatory	Description
grp="<GroupName(s)>"	Yes	List of groups to query separated by comma
queryChildgroups="<true|false>"	No	Query groups that are contained in the groups you specified to export.

Groups passed in a text file

When specifying the -grp parameter, you have to specify the exact group name, wildcards are not allowed.

Examples

```
-grp "Group1,AdminGropup,CitrixGroup"
```

finds all groups named **Group1, AdminGroup, CitrixGroup.**

```
-grp "File:MyGroups.txt"
```

Will query all groups specified in the file MyGroups.txt, MyGroups.txt has to contain one group per line, and has to be placed in the same directory as the Microsoft-AzureAD-GetGroups.ps1 file.

Attention	**File:** is case sensitive!

Microsoft AzureAD - API Access using an Application

An alternative to using a domain account, is the use of an Azure application. The connection is then established using a (self-signed) certificate. The certificate is in the machine store of the computer that the Data Collector is installed on.

Microsoft AzureAD - Creation of the certificate

A certificate containing a private and a public key are needed for the connection. If such a certificate is not available, it can be created with the script: **Microsoft-AzureAD-CreateSelfSignedCertificate.ps1** which can be found in the folder of the connector.

Attention	The script has to be executed as an Administrator, this way it is ensured that the resulting certificate can be stored in the machine store, instead of the logged on user.

Parameter	Mandatory	Description
-outputPath "<Path>"	No	Path the certificate files are written to, if omitted they will be written to the %TEMP% folder of the executing user.
-pwd "<Password>"	Yes	Password the certificate is protected with.
-dnsName "<DNS Name>"	No	DNS name that is put into the certificate, if no DNS name is provided, "my.domain.local" will be used. This will also serve as the name of the files.
-validfornYears =<xx>	No	Number of years this certificate will be valid, if this is omitted the certificate will be issued for 10 years.

Example

Creation of a certificate with the password "myPassword", the DNS name "name.company.com" and valid for five years.

```
PowerShell -File ".\Microsoft-AzureAD-CreateSelfSignedCertificate.ps1" -outputPath "C:\Temp" -pwd "myPassword" -dnsName
"name.company.com" -validfornYears 5
```

Files created in C:\Temp

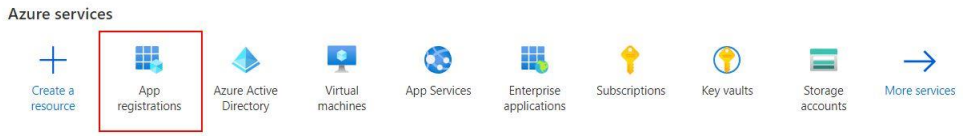
- name.company.com.pfx
- name.company.com.crt

Microsoft AzureAD - Setup of the Application via Azure Portal

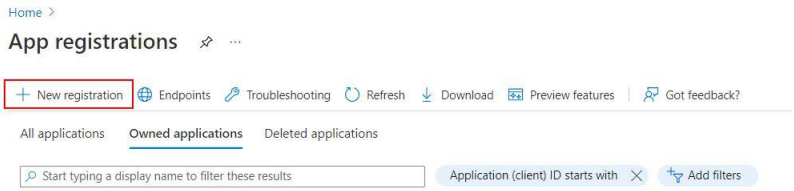
Access to the data is carried out via an application, the application has to be set up in the Azure Portal https://portal.azure.com .

Instructions

1. Logon to the portal

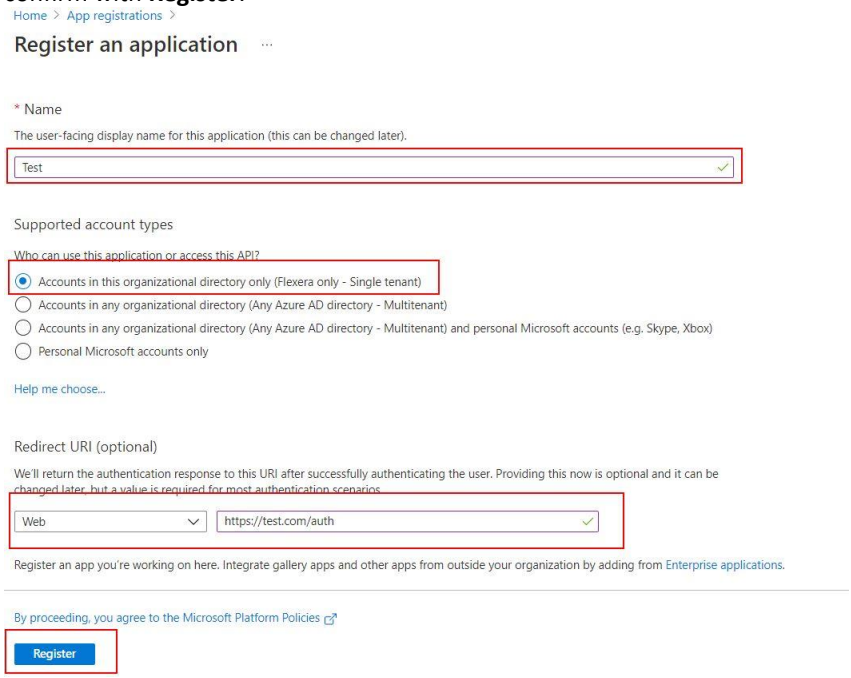


2. Choose, **App registrations**, **All apps, New application registration**

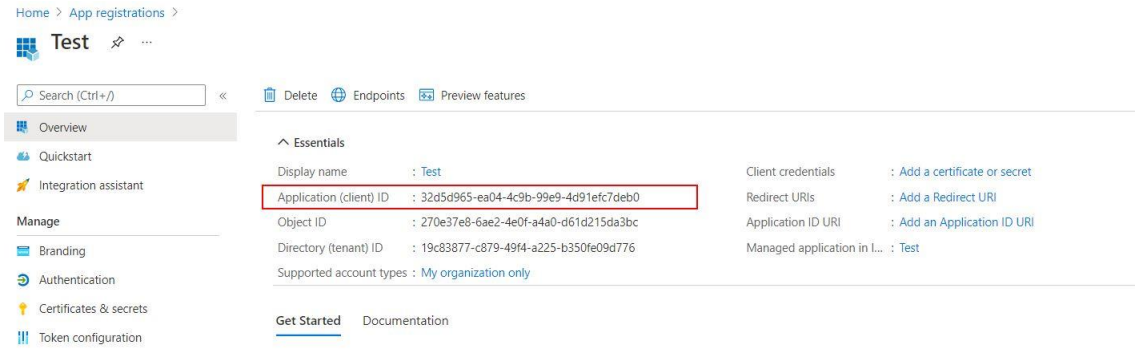


3. Specification of
   Name: can be any name

Supported account type: Choose single tenant
Redirect URI: can be any name, (https:// has to be included)
confirm with **Register**.

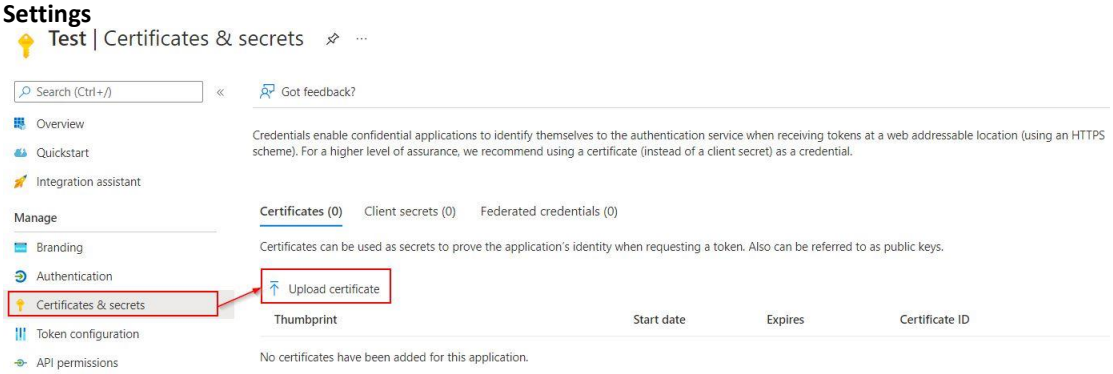


4. Application details

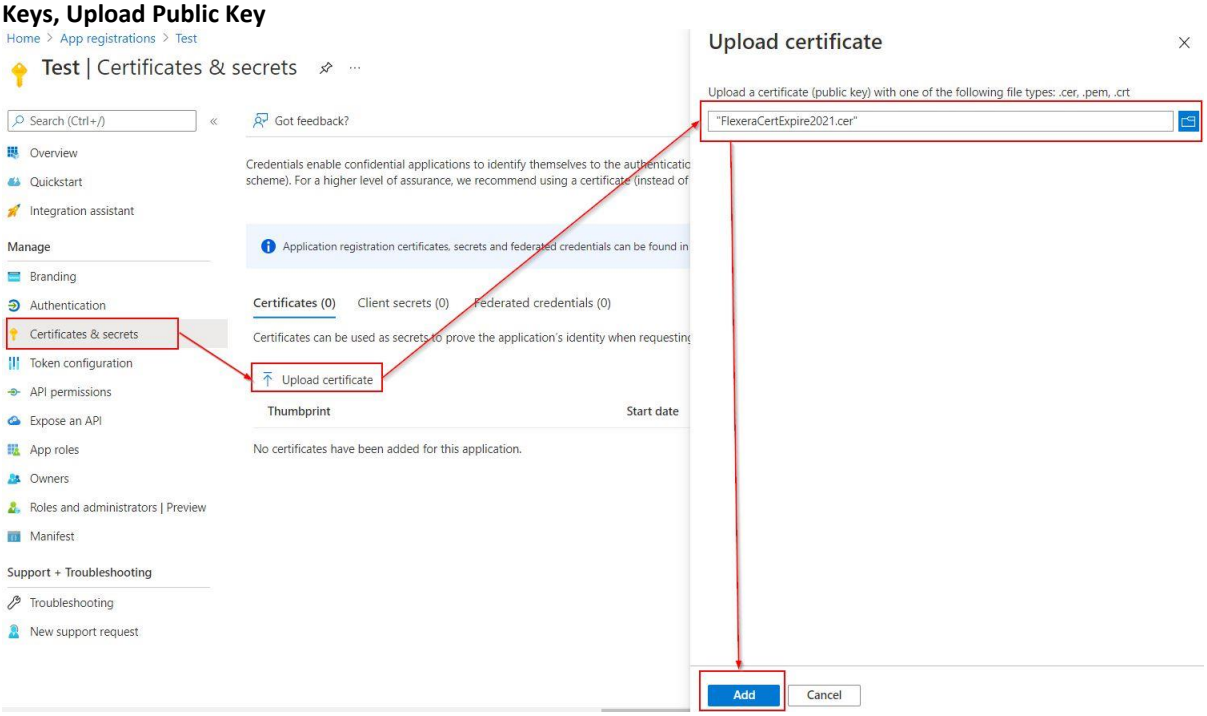


5. Copy the ApplicationID and save for later use

6. **Settings**

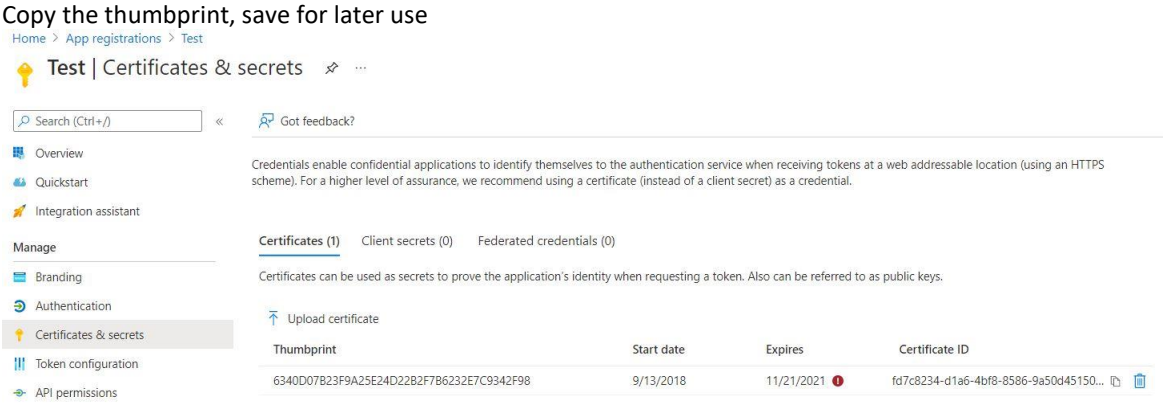


7. **Keys, Upload Public Key**



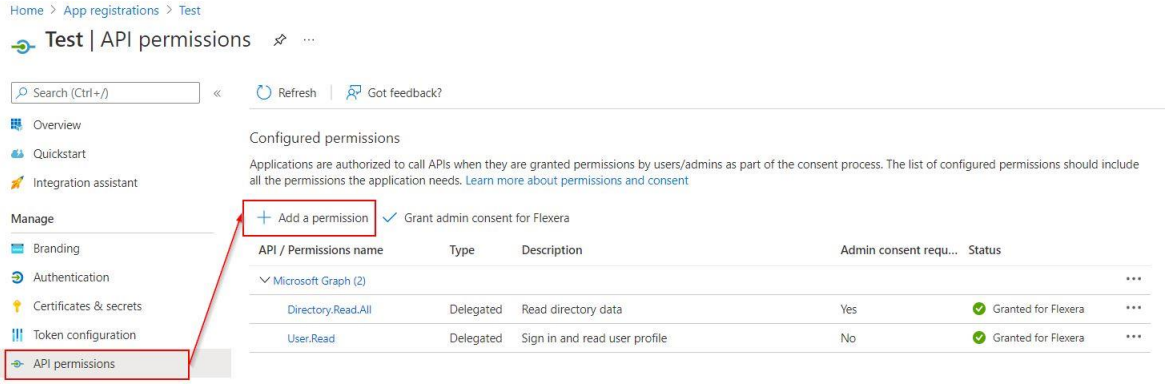
8. Choose key, **Add**

9. Copy the thumbprint, save for later use

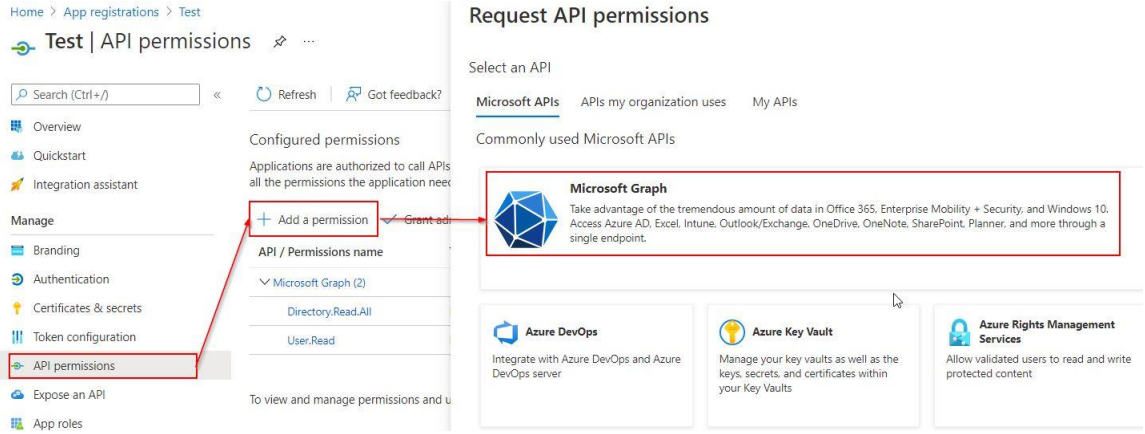


10. **Required Permissions, Windows Azure Active Directory**

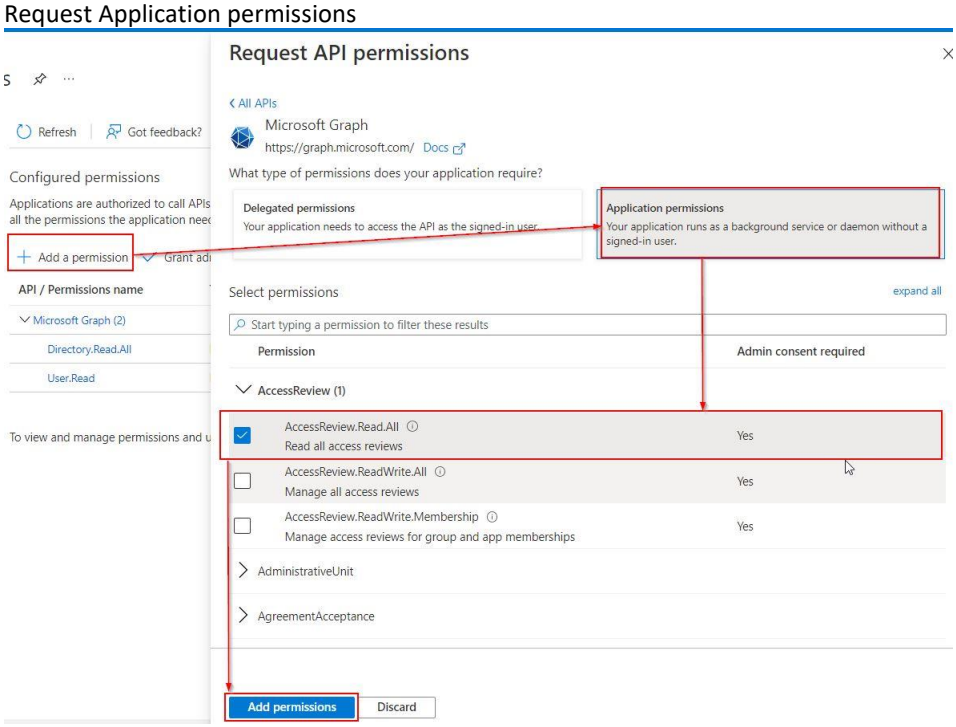
11. In "API PERMISSIONS", choose Add a permission



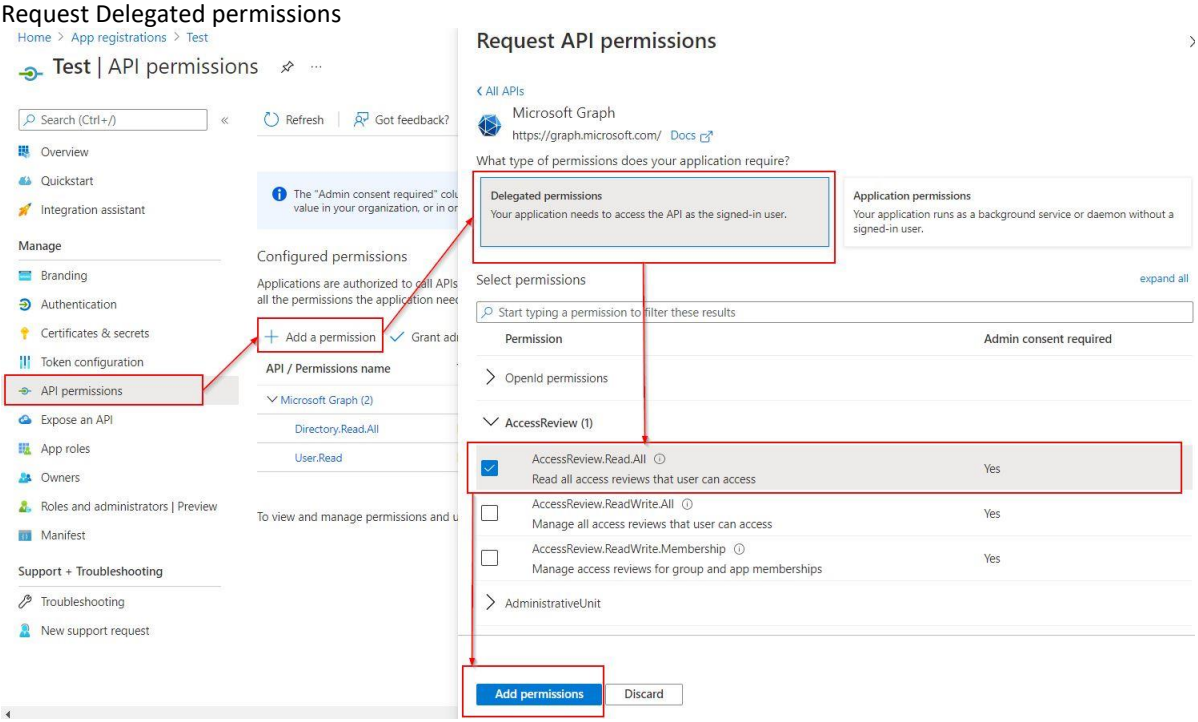
12. Choose Microsoft API



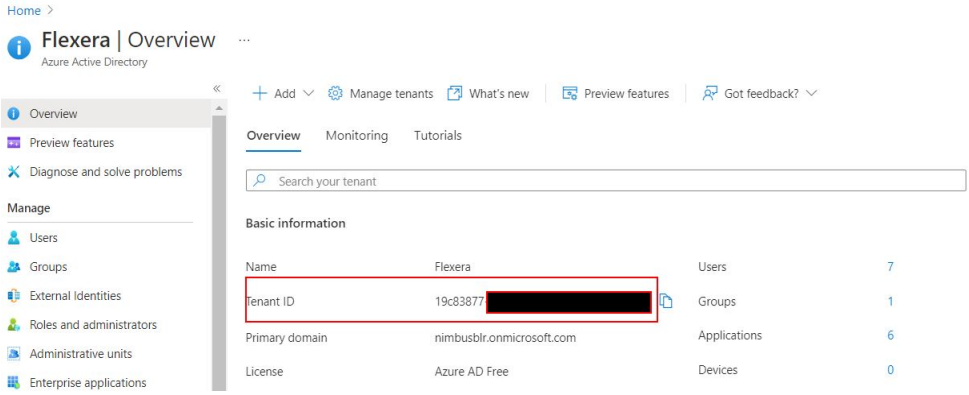
13. Request Application permissions



14. Request Delegated permissions



15. Done



Microsoft AzureAD - Identify TenantID

Azure Active Directory, Properties, note Directory ID for later use.

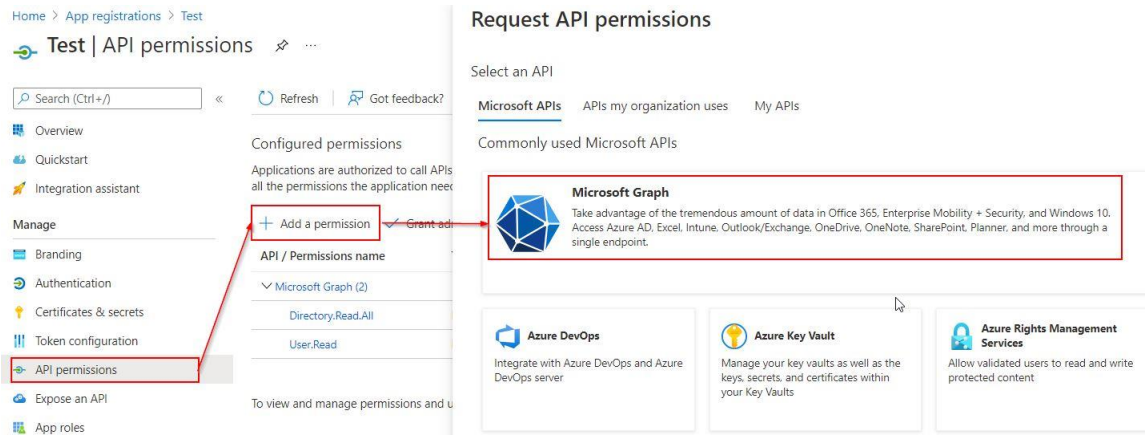


Figure - DirectoryID

4.3.5 Microsoft Intune

The device identification is done with the help of the field SerialNo.

System requirements

For the connectors to work the Windows Module **"AzureAD"** has to be installed

The following PowerShell command can be used to check and install the AzureAD module

```
if (!(Get-Module -ListAvailable | where { $_.Name -eq "AzureAD"}))
{
        Install-Module AzureAd
}
```

A script containing the above code called InstallModule_AzureAD.ps1 is copied to the Microsoft-Azure-Connector folder.

Attention Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration

Connector.config Attribute	Mandatory	Description
uid="<User>"	Yes	User that can query the portal
pwd="<Password>"	No	Password of above user
ApplicationId="<ApplicationID>"	Yes	ApplicationID for Access via Application
h="true|false"	No	Export Hardware
s="true|false"	No	Export Software
proxyAddress="<Proxy>"	No	Address of proxy server (if needed) to connect to the internet. Proxy address including http(s)://
proxyPort="<Port>"	No	Proxy Servers Port
proxyUser="<User>"	No	User that can authenticate against proxy server, if needed.
proxyUserPassword="<Password>"	No	Password for above user.

Examples in Connector.config:

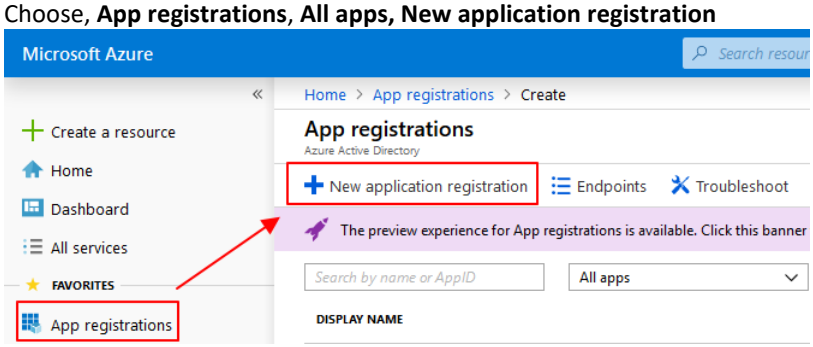
```
<connector name="Microsoft-Intune-Connector" subfolder="Microsoft-Intune-Connector" active="false"
scriptname="Microsoft-Intune-Connector.ps1" uid="user@domain.com" pwd=""
-applicationID="12345678-abcd-efgh-ijkl-mnopqrstuvwxy" />
```

Microsoft Intune - Setup of the Application via Azure Portal

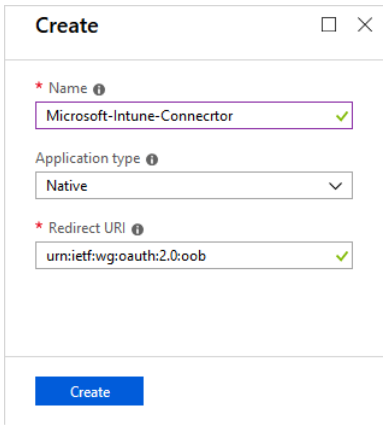
Access to the data is carried out via an application, the application has to be set up in the Azure Portal https://portal.azure.com .

Instructions

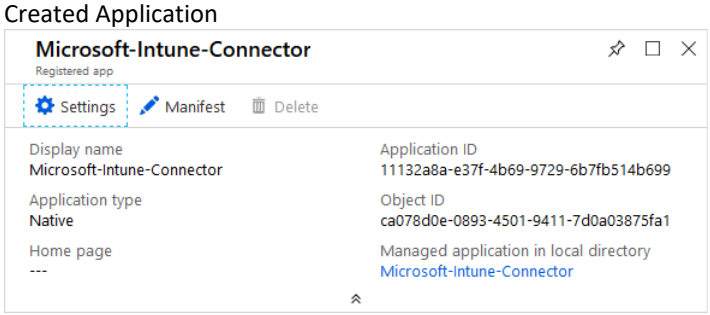
1. Logon to the portal
2. Choose, **App registrations**, **All apps, New application registration**



3. Specify
   Name: Chose any valid name
   Application Type: Native
   Sign-on URI: urn:ietf:wg:oauth:2.0:oob
   confirm with **Create**



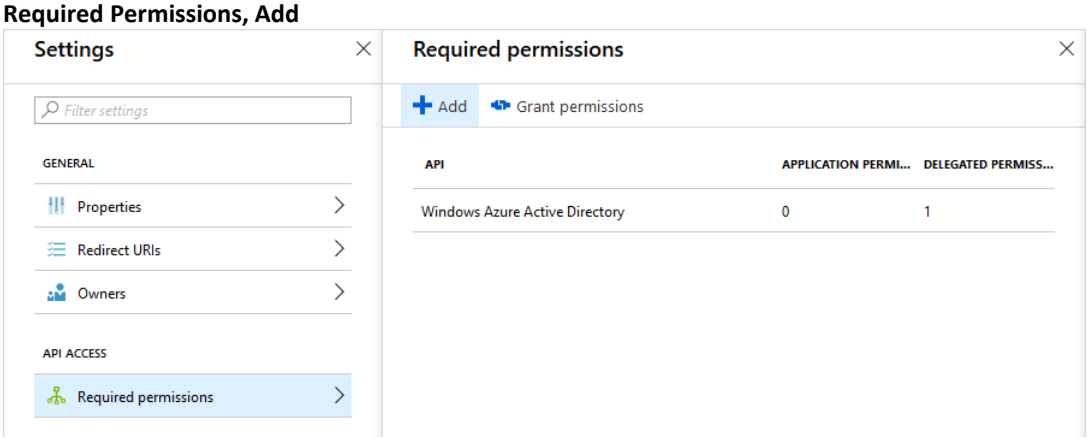
4. Created Application



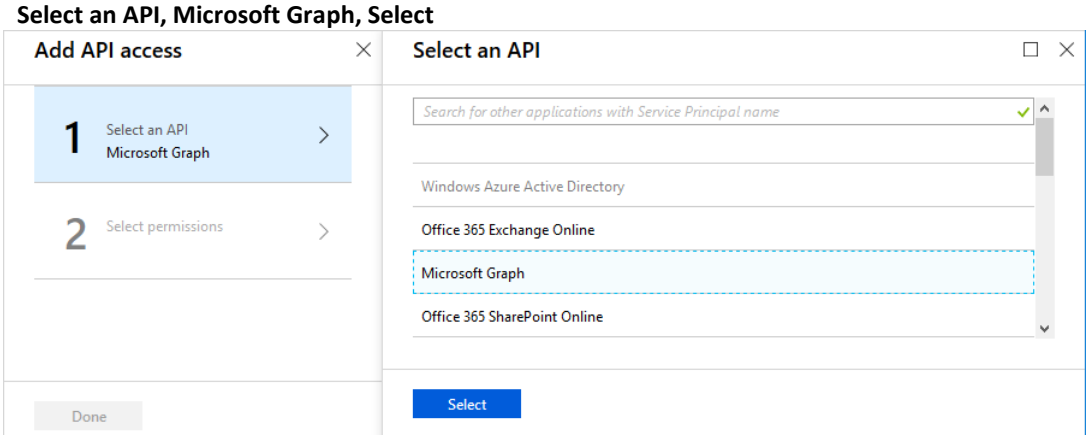
5. Copy the ApplicationID and safe for later use

6. **Settings**

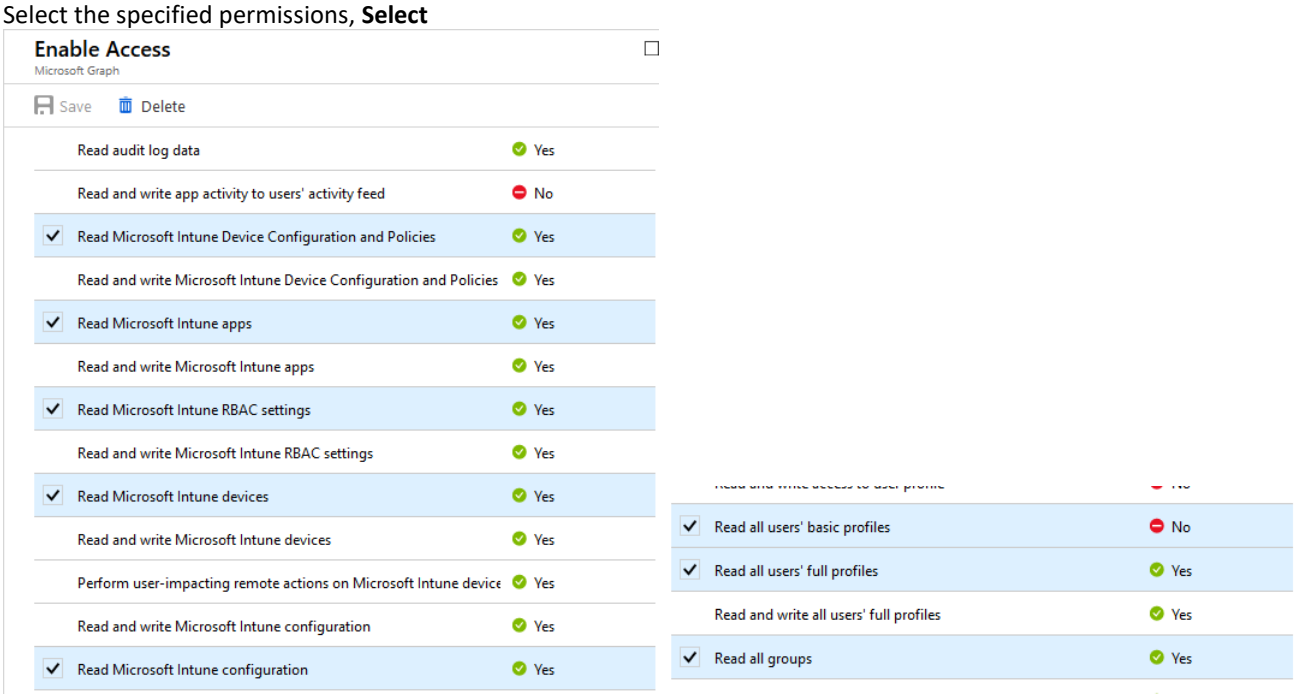
7. **Required Permissions, Add**



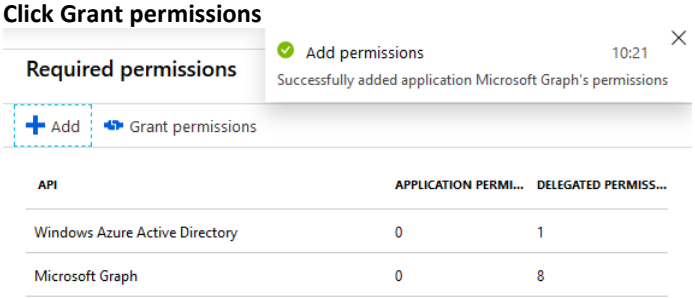
8. **Select an API, Microsoft Graph, Select**



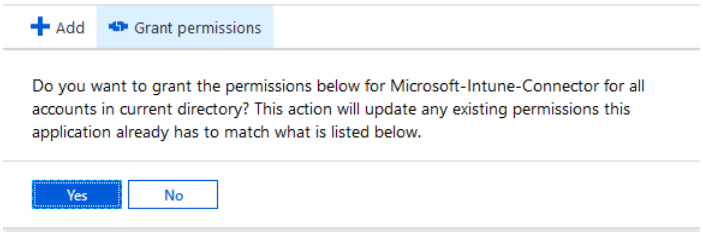
9. Select the specified permissions, **Select**



10. **Click Grant permissions**



11. **Yes**



12. **Done**

4.3.6 Microsoft Active Directory

The Active Directory Connectors are a set of connectors used to export data from Active Directory.

Prerequisites

For the connectors to work the Windows Feature **"RSAT-AD-Powershell"** has to be installed

The following PowerShell command can be used to check and install AD-Domain Services

```
if (Get-WindowsFeature | Where-Object {($_.Name.Trim() -eq "RSAT-AD-Powershell") -and ($_.Installed -eq $False)})
{
        Write-Host "RSAT-AD-Powershell not found, installing...."
        Add-WindowsFeature  -name RSAT-AD-Powershell
}
```

A script containing the above code called InstallWindowsFeature_RSAT-AD-Powershell.ps1 is copied to the ADConnector folder.

Attention Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Note PowerShell uses Active Directory Web Services to access the Active Directory, details can be found here:
https://blogs.msdn.microsoft.com/adpowershell/2009/04/06/active-directory-web-services-overview

The default port used for this is: 9389

User Objects

Configuration

Connector.config attribute	Mandatory	Description
dc="<Domain Controller>"	No	DNS Name of a domain controller may be used if domain name resolution is unreliable.
uid="<User>"	No	User that can query the domain
pwd="<Password>"	No	Password of above user
filter="<Filter string>"	No	Filter for avoiding user accounts you do not want In order to filter unwanted users you have to specify a filter in the command line, more information about filtering can be found here: https://technet.microsoft.com/en-us/library/hh531527.aspx
ou="<OU string>"	No	Only return accounts from given OU in the format OU=user,OU=Test,OU=domain,DC=domain,DC=com

Note Please be aware that the filtering technique used here is Active Directory Filtering and NOT LDAP Filtering. Also some filters are already in place in the .ps1 script, so all filters specified on the command line will be joined with an "and" to that existing filter!"

The Connector can be configured to only export certain attributes of an AD object; this is done in the file **GetADUserObjectsAttributeConfig.txt i**n the same path as the .ps1 script.

The GetADUserObjectsAttributeConfig.txt is a csv type file.

Attention	Please note that no changes are allowed to be made in the SWRDAttribute and Type columns.

If you want to disable the export of a certain attribute, set the value in the process column to "Disabled"

ADAttribute,SWRDAttribute,Type,Process

objectsid,ObjectSid,System.String,Enabled

objectguid,ObjectGUID,System.Guid,Enabled

distinguishedname,DistinguishedName,System.String,Enabled

userprincipalname,UserPrincipalName,System.String,Enabled

Filter Examples

Description	Filter
Only export users that have a first name set:	filter="(givenname -like '*')"
Only export users that have the last name set	filter="(sn -like '*')"
Only export users that have first AND last name set	filter="((givenname -like '*')" -and (sn -like '*'))"
Only export users that have an email address set	filter="(EmailAddress -like '*')"
Only export users that are enabled (=not disabled)	filter="(Enabled -ne $false)"
Only export users that have an email address ending with bwg.testing	filter="(EmailAddress -like '*@bwg.testing')"

Note	Please note the way the quotation marks are set for the above filter arguments!

Computer Objects

The Active Directory Connector for Computer Objects is a script that exports computer objects from the current domain.

Configuration

Connector.config attribute	Mandatory	Description
dc="<Domain controller>"	No	DNS Name of a domain controller may be used if script is called from a workstation that is not joined to a domain of if a foreign domain is to be queried.
uid="<User>"	No	User that can query the domain
pwd="<Password>"	No	Password of above user
InactiveDays="<xx>"	No	Only return machines that changed their password no longer than the given number of days ago
ou="<OU string>"	No	Only return accounts from given OU in the format OU=user,OU=Test,OU=domain, do not include "DC=" OU= has to be capitalized

Connector.config attribute	Mandatory	Description
filter="<Filter string>"	No	Filter for avoiding computer accounts you do not want see: http://www.ldapexplorer.com/en/manual/109010000-ldap-filter-syntax.htm for examples

Group Objects

Configuration

Connector.config attribute	Mandatory	Description
dc="<Domain Controller>"	No	DNS Name of a domain controller, may be used if domain name resolution is unreliable.
uid="<User>"	No	User that can query the domain
pwd="<Password>"	No	Password of above user
grp="<GroupName(s)>"	Yes	List of groups to query separated by comma
strict="<true|false>"	No	Only get information about specified groups but not of related groups (e.g. group(s) that the selected group(s) contain as members)
queryParentGroup="<true|false>"	No	When specified will traverse upwards and query groups that the specified group(s) have as parents.

Wildcards and groups passed in a text file

When specifying the -grp parameter you can choose to use wildcards.

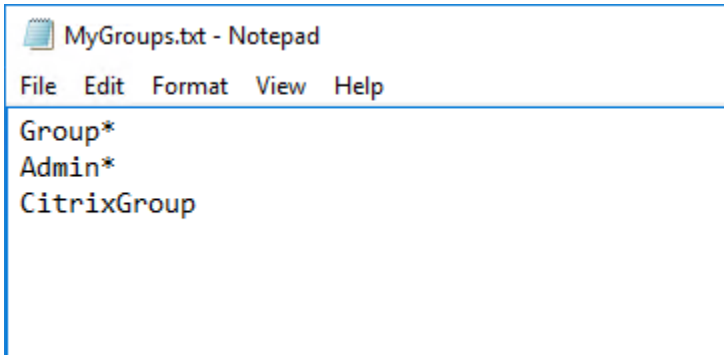
Examples

```
-grp "Group*,Admin*,CitrixGroup"
```

finds all groups starting with **Group**, all groups starting with **Admin** and the group **CitrixGroup.**

```
-grp "File:MyGroups.txt"
```

Will query all groups specified in the file MyGroups.txt, MyGroups.txt has to contain one group per line, and has to be placed in the same directory as the GetADGroupObjects.ps1 file. The * as a wildcard is also permitted in group names within the file.



Attention	**File:** is case sensitive! Microsoft Active Directory limits the number of group members returned to 5.000, details can be found here: https://technet.microsoft.com/en-us/library/dd391908(WS.10).aspx . As a workaround, you can use the Microsoft AzureAD - Group Export which has no known limitation.
Note	An alternate script "GetADGroupObjects2.ps1" is provided for querying large groups, this one utilizes additional AD lookups, so overall performance is affected.

4.3.7 Microsoft Application Virtualization (App-V) Connector

This connector has multiple parts that can be used.

Connector	Description
Microsoft-App-V-Connector.ps1	Connector that utilizes the App-V Powershell Module that is installed alongside the App-V Management Console
Microsoft-App-V-SQL-Connector.ps1	SQL based connector, that will query the same information as the connector above, only it will connect to the management database directly.
Microsoft-App-V-SQL-PackageApplicationUsage.ps1	SQL based connector that export the usage data of the App-V packages, this connector will only work directly with the database.

Important:: To query the App-V packages either "Microsoft-AppV-Connector.ps1" or "Microsoft-AppV-SQL-Connector.ps1" can be used. The differences between those will be described in the following chapters.

Attention Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

App-V Package data, PowerShell based

Prerequisites

For the connector to work it relies on the App-V PowerShell Module for App-V Servers. this is installed alongside the App-V Management Console.

To export the data, the script establishes a remote connection from the machine the SDC is installed on to the machine that holds the App-V server. There the PowerShell Module "AppVServer" that is provided by the App-V server installation is used to query the data.

In order for the remote connection to work, PSRemoting has to be enabled on the remote machine, you can find details about this here: PSRemoting (on page 60).

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User to connect with
pwd="<Password>"	No	Password of above user
server="<App-V Server>"	Yes	Name of the App-V server that hat the Management Console and database installed. If you run the Data Collector on the App-V Server itself, you can omit this entry

Output

The Microsoft App-V Connector will query the following details:

- Packages
- Package Application
- Package Entitlements / AD Groups

App-V Package data, SQL based

Prerequisites

None

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User to connect with
pwd="<Password>"	No	Password of above user
server="<App-V Server>"	Yes	Name of the SQL server that holds the Management database
database="App-V Management database>	Yes	Name of the App-V Management database

Output

The Microsoft App-V Connector will query the following details:

- Packages
- Package Application
- Package Entitlements / AD Groups

App-V Usage Data, SQL based

Prerequisites

None

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User to connect with
pwd="<Password>"	No	Password of above user
server="<App-V Server>"	Yes	Name of the SQL server that holds the Reporting database
database="App-V Management database>	Yes	Name of the App-V Reporting database
days="<Days>"	No	Optional: Day to go back in time for usage data. Default value if omitted is one day.

Output

The Microsoft App-V Connector will query the following details:

- PackageUsage

4.3.8 Hyper-V

Prerequisites

For the connector to work the Windows Feature **"Hyper-V-PowerShell"** has to be installed

The following PowerShell command can be used to check and install Hyper-V-PowerShell

```
if (Get-WindowsFeature | Where-Object {($_.Name.Trim() -eq "Hyper-V-PowerShell") -and ($_.Installed -eq $False)})
{
        Write-Host "Hyper-V-PowerShell not found, installing...."
        Add-WindowsFeature  -name Hyper-V-PowerShell
}
```

A script containing the above code called InstallWindowsFeature_Hyper-V-PowerShell.ps1 is copied to the Hyper-V folder.

Attention Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User that can query the domain
pwd="<Password>"	No	Password of above user
srv="<Hyper-V server>"	Yes	Hyper-V server to be queried.

Output

The Hyper-V Connector will query the following details:

- DeviceRelationshipTypeID
- ChildDeviceUUID
- ParentDeviceUUID
- ParentDeviceURN
- ScanDate

Note The Hyper-V connector will only query Host-Guest relationships; cluster information will be added with a later release

4.3.9 Hyper-V via Virtual Machine Manager

Prerequisites

For the connector to work the module **"virtualmachinemanager"** has to be installed, this module comes with the installation of the Microsoft Virtual Machine Manager.

Attention	Please note that this connector only works if the DC is installed on the machine that runs the Virtual Machine manager.

Attention	Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	User that can query the domain
pwd="<Password>"	No	Password of above user
srv="<Hyper-V VMM server>"	Yes	Hyper-V server running Virtual Machine Manager.

Output

The Hyper-V VMM Connector will query the following details:

- Hardware (of clusters, if present)
  - Information of the Hyper-V cluster if any is present
- Relation
  - DeviceRelationshipTypeID
  - ChildDeviceUUID
  - ParentDeviceUUID
  - ParentDeviceURN
  - ScanDate

4.3.10 Microsoft Exchange Connector (beta)

Attention Please note that this connector is in a beta phase, the results can be incomplete or erroneous.

Prerequisites

For the connector to work it relies on the Windows Feature **"RSAT-AD-Powershell"**.

The following PowerShell command can be used to check and install RSAT-AD-Powershell

```
if (Get-WindowsFeature | Where-Object {($_.Name.Trim() -eq "RSAT-AD-Powershell") -and ($_.Installed -eq $False)})
{
        Write-Host "RSAT-AD-Powershell not found, installing...."
        Add-WindowsFeature  -name RSAT-AD-Powershell
}
```

A script containing the above code called **InstallWindowsFeature_RSAT-AD-Powershell.ps1** was copied to the ADConnector folder.

Attention Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User that can query the domain
pwd="<Password>"	No	Password of above user

Output

The Microsoft Exchange Connector will query the following details:

- ObjectSID
- ObjectGUID
- DistinguishedName
- UUID
- Name
- Edition
- AdminDisplayVersion
- ProductID
- ExchangeVersion

4.3.11 LDAP (beta)

Attention Please note that this connector is in a beta phase, the results can be incomplete or erroneous.

Prerequisites

None

Configuration

Connector.config attribute	Mandatory	Description
uid="<User>"	No	Optional: User that can query the domain
pwd="<Password>"	No	Password of above user
dc="<LDAP server>"	Yes	LDAP server to be queried.
filter="<Filter string>"	No	Filter for avoiding user accounts you do not want see: http://www.ldapexplorer.com/en/manual/109010000-ldap-filter-syntax.htm for examples
ou="<OU string>"	No	Optional: Only return accounts from given OU in the format OU=user,OU=Test,OU=domain,DC=domain,DC=com

The LDAP Connector can be configured to only export certain attributes of an LDAP object, this is done in the file **AttributeConfig.txt** in the same path as the .ps1 script.

The AttributeConfig.txt is a csv type file, there are examples provided for Windows and Novell eDirectory

Attention Please note that no changes are allowed to be made in the SWRDAttribute and Type columns.

If you want to disable the export of a certain attribute, set the value in the process column to "Disabled"

LDAPAttribute,SWRDAttribute,Type,Process

objectsid,ObjectSid,System.String,Enabled

objectguid,ObjectGUID,System.Guid,Enabled

distinguishedname,DistinguishedName,System.String,Enabled

userprincipalname,UserPrincipalName,System.String,Enabled

4.3.12 XEN Server (beta)

Attention	Please note that this connector is in a beta phase, the results can be incomplete or erroneous.

Prerequisites

For the connector to work it is necessary to install the XEN PowerShell module. The module can be found under: http://xenserver.org/partners/developing-products-for-xenserver.html . The XEN PowerShell Module needs Microsoft .NET 4.5 and PowerShell v4.

Attention	Please review your settings for the Powershell Execution Policy, for details please see: PowerShell Execution Policy (on page 61)

Configuration

Connector.config Attribute	Mandatory	Description
uid="<Username>"	No	Username to use to connect to the server.
pwd="<Password>"	No	Password of the above user.
srv="<XEN server>"	Yes	Name of the XEN server to query.

Output

The XEN Connector queries the following details:

- Hardware (of the XEN server)
  - UUID
  - DomainName
  - HostName
  - Manufacturer
  - ScanDate
  - Model
  - ProcessorManufacturer
  - ProcessorType
  - ProcessorSpeed
  - CPUCount
  - CorePerCPU
  - CPUCoreCount
  - CPULogicalCount
  - Mac1
  - Mac2
  - Mac3
  - Mac4
  - MemoryMB
  - IPAddressV4
  - OSCaption
  - BIOSVersion

- SerialNo
- InventorySource

- Relationships
  - DeviceRelationshipTypeID
  - ChildDeviceUUID
  - ParentDeviceUUID
  - ScanDate

Note	The XEN connector only queries the hardware of the XEN server itself and the Host-Guest relationships. Hardware details of the guest machines are not queried.

Spider/Columbus Inventory (Windows / Mac OS)

5.1 Windows

5.1.1 System Requirements for Columbus Inventory

The minimum OS requirements for using Columbus Inventory are:

- Windows XP (32bit) SP3
- Windows 2003 (32/64bit) SP2

5.1.2 DSGVO / GDPR Settings

Starting with version 7.5.5.17 all fields with personal data will not be exported by default anymore..

Inventory	Fields
HardwareScan.csv	LastLoggedOnUser LastLoggedOnSAMUser LastLoggedOnUserSID MAC1 MAC2 MAC3 MAC4 IPAddressV4 IPAddressV6
InventoryItems.csv	OS.System.RegisteredUser OS.System.Organization OS.System.ProductKey

If needed the export of the data can be enabled by changing the configuration for the Inventory Agent (on page 96) and the Inventory Scanner (on page 108).

5.1.3 Columbus Inventory Agent

The Columbus Inventory Agent is the scanning engine that can be installed as a service and will continuously meter and scan the machine it is installed on.

Columbus Inventory Agent Location

The inventory scanner can be found in the chosen directory (Figure - Destination Folder (on page 12)) in the sub directory ..\ColumbusInventoryAgent, it consists of the files:

- ColumbusInventoryAgent.cfg
- ColumbusInventoryAgent.exe
- ColumbusInventoryAgentUpdater.exe
- libeay32.dll
- ssleay32.dll

(The ColumbusInventoryAgentUpdater.exe , is not needed for distribution of the agent, it is only placed in this directory for availability.)

Columbus Inventory Agent Configuration

The scanner is installed as a service and will be started automatically with the machine.

Scanning takes place if:

- the Inventory Agent has been newly installed on the machine and starts for the first time
- the timespan specified in InvScanStartPeriod has passed since the last scan
- the Last Scan Date has been reset

Configuration of the Inventory Agent is achieved using the ColumbusInventoryAgent.cfg which must at least include the Target Server for the transmission of the resulting zip files.

```
[Transmitter]
InvOTB_Host=yourserver.yourdomain.local
```

Section	Config parameter	Default (if empty)	Possible values / Description
Scanner	InvFunction	2	0 = HW, SW, Inv. Items 1 = HW, SW, Inv. Items, File scan 2 = HW, SW, Inv. Items, File scan, Metering
Scanner	InvDrives	All local hard disk drives	CDE (Means, drive C:, D: and E:)
Scanner	InvExtensions	.EXE	List of extensions for which detailed file data will be collected.
Scanner	InvExportPath	%ProgramData%\Columbus	%temp% or %_ExePath%
Scanner	InvUpdateAgent	1	0=disabled 1=enabled
Scanner	InvUpdateEngine	1	0=disabled 1=enabled
Scanner	InvScanStartPeriod	daily	daily, weekly, monthly
Scanner	InvScanStartDelay	0	n minutes (0-100)

Section	Config parameter	Default (if empty)	Possible values / Description
Scanner	InvLastObject	0	1 = Export of personal data • LastLoggedOnUser • LastLoggedOnSAMUser • LastLoggedOnUserSID
Scanner	InvNetwork	0	1 = Export of personal data • MAC1 • MAC2 • MAC3 • MAC4 • IPAddressV4 • IPAddressV6
Scanner	InvLicensee	0	1 = Export of personal data • OS.System.RegisteredUser • OS.System.Organization • OS.System.ProductKey
Transmitter	InvTransmissionMode	3	0 = No Transmission, offline mode 1 = FTP 2 = not used 3 = OTB
Transmitter	InvOTB_Host		FQDN of Data Collector machine
Transmitter	InvOTB_Port	24786	Port of Data Collector machine
Transmitter	InvFTP_Host		FTP-Server hostname
Transmitter	InvFTP_Port		FTP-Server port
Transmitter	InvFTP_User		FTP-Server authentication user (Empty uses anonymous)
Transmitter	InvFTP_Password		FTP-Server authentication password, (Encrypt with cryptit.exe)
DirectoryFilter	InvDirectoryFilter001 ... InvDirectoryFilter999		Windows variables, fixed paths like %windir%* or D:\Data*

Default Filters

The agent comes with a default filter set that is described below:

```
[DirectoryFilter]
InvDirectoryFilter000=*\microsoft system center 2012\dpm\dpm\volumes\*
InvDirectoryFilter001=%windir%\$*_$\*
InvDirectoryFilter002=%windir%\*\$*_$\*
InvDirectoryFilter003=%windir%\Installer\*
InvDirectoryFilter004=%windir%\system32\ccm\cache\*
InvDirectoryFilter005=%windir%\WinSxS\*
InvDirectoryFilter006=%windir%\ServicePackFiles\i386\*
InvDirectoryFilter007=%ProgramData%\App-V\*
InvDirectoryFilter008=%ProgramData%\app-v\*
```

```
InvDirectoryFilter009=%APPDATA%\*
InvDirectoryFilter010=%LOCALAPPDATA%\*
InvDirectoryFilter011=*\AppData\LocalLow\*
```

Columbus Inventory Agent Installation

brainwaregroup recommends the Inventory Agent is installed in the machine's normal "Program Files" directory, e.g. C:\Program Files (x86)\Columbus\InventoryAgent.

The ColumbusInventoryAgent.exe supports the following command line switches

Switch	Function
/Install	Installs the Inventory Agent as a service
/Uninstall	Removes the Inventory Agent service
/Silent	Silent operation for us in scripts or batch files

Examples

Install Inventory Agent as a Service in silent mode:

```
C:\Program Files (x86)\Columbus\InventoryAgent\ColumbusInventoryAgent.exe /Install /Silent
```

Uninstall Inventory Agent service

```
C:\Program Files (x86)\Columbus\InventoryAgent\ColumbusInventoryAgent.exe /Uninstall
```

After the first start of the service the Inventory Agent will read the information from ColumbusInventoryAgent.cfg write those to the registry and delete the ColumbusInventoryAgent.cfg from the directory.

Columbus Inventory Agent Update

If the automatic update option is enabled (default setting - autoupdate enabled), (see Columbus Inventory Agent Configuration) the agent will update itself by checking for available updates every 24 hours.

The files for the update can be found in the "Updates_Agent" directory where the scanner transmits its results files. The folder "Updates_Agent" must contain a zip file called "Updates_Agent.Zip". Within this ZIP file, the following files should be included:

- ColumbusInventoryAgentUpdater.exe (mandatory)
- EDCAgentUpdater.exe (mandatory if you plan to migrate from EDC-Agent to Columbus Inventory Agent)
- ColumbusInventoryAgent.exe (optional)
- ColumbusInventoryAgent.cfg (optional)

If either the "ColumbusInventoryAgent.exe" or "ColumbusInventoryAgent.cfg" files exist within the ZIP file, the scanner will update with this executable and/or change the configuration to the one given in ColumbusInventoryAgent.cfg.

During an update of the Inventory Agent the mentioned zip file is created by the installer and contains "ColumbusInventoryAgentUpdater.exe", "ColumbusInventoryAgent.exe" and "EDCAgentUpdater.exe"

Columbus Inventory Agent Resetting Last Scan Date

For testing purposes it might be necessary to reset the date the agent has last scanned a certain machine.

This is achieved by deleting the registry key (System User):

```
Key: HKEY_LOCAL_MACHINE\SOFTWARE\<Wow6432Node>\BrainWare\Columbus\7\InvAgent
Value: LastRun
```

Columbus Inventory Agent Metering

Inventory Agent does not collect metering data

In case the Inventory Agent does not provide metering data, please check the configuration of the Inventory Agent on the machine that does not provide metering data.

Navigate to the registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\BrainWare\Columbus\7\InvAgent\Config
```

and check for the existence of the REG_SZ Value "InvFunction", **if "Function" exists with any other value than "2" metering will be disabled.**

5.1.4 Columbus Inventory Agent MSI

The Inventory Agent is also available as a MSI based Installation Package.

Columbus Inventory Agent MSI Installation

Even though the ColumbusInventoryAgent.msi provides a graphical user interface, it is recommended that the MSI is distributed by using automated installation tools (such as Columbus, SCCM, LANDESK, Altiris etc.) or by creating a Group Policy Object (GPO) in AD and pushing this to the OUs that contain the machines that should have the Inventory Agent installed.

We recommend you install the Inventory Agent in the regular "Program Files" directory, e.g. C:\Program Files (x86)\Columbus\InventoryAgent.

The ColumbusInventoryAgent.msi supports the following public properties:

Name	Values	Description
INVOTB_PORT	1685	Port that the OTB Server listens on
INVOTB_HOST		FQDN of the OTB Server (aka Data Collector)
AUTOSTARTSERVICE	0 | 1	Automatically start the service after installation (0|1) 1 means yes and is the default when omitted.
SCANNERFUNCTION	0 = HW, SW, Inv. Items 1 = HW, SW, Inv. Items, File scan 2 = HW, SW, Inv. Items, File scan, Metering	Determines if the agent does also meter software usage (2) or not (1 or every other value). Default if empty is 2
INVUPDATEENGINE	0 | 1	Automatically update Scanner Add-on DLLs when running, Default: 1

Name	Values	Description
INVUPDATEAGENT	0 | 1	Automatically update Agent, Default: 1
INVSCANSTARTPERIOD	daily | weekly | monthly	Scan frequency, default when omitted is daily
INVSCANSTARTDELAY	0 - 100	Randomized delay for starting scan, 0 = disabled (default) in minutes.
INVDRIVES	CDE	Defines local drives that are to be scanned, empty (default) means all local fixed drives.

Examples for command line usage:

Install Inventory Agent, Auto start Service, Inventory Only in silent mode:

```
msiexec /i "ColumbusInventoryAgent.msi" /qn INVOTB_HOST="<FQDN of Data Collector machine>" INVOTB_PORT="24999" SCANNERFUNCTION=1
/L*V InventoryAgentInstaller.log
```

Install Inventory Agent, Auto start Service, Inventory+Metering in silent mode:

```
msiexec /i "ColumbusInventoryAgent.msi" /qn INVOTB_HOST="<FQDN of Data Collector machine>" INVOTB_PORT="24999" /L*V
InventoryAgentInstaller.log
```

For more information about using msiexec.exe run "msiexec.exe /?" in a command prompt.

Attention	Please note that in order to activate Metering you either have to omit the SCANNERFUNCTION= from the msi command line or you have to use SCANNERFUNCTION=2, everything else will disable metering.

Columbus Inventory Agent MSI Location

The MSI can be found in the chosen directory (Figure - Destination Folder (on page 12)) in the sub directory ..\ColumbusInventoryAgent-MSI, it consists of the file:

- ColumbusInventoryAgent.msi

Deployment using GPO (Step by Step)

This is a short list of step by step instructions on how to deploy the ColumbusInventoryAgent.msi by using GPOs.

Note	Please note that all paths and location configuration must be adapted to your environment.

Create an MST file for the MSI

1. Download the installer for the Windows Software Development Kit (SDK) for Windows 8.1 from https://msdn.microsoft.com/en-us/windows/desktop/bg162891.aspx and start the installation.
   On the dialogue "Select the features you want to install" Select "MSI Tools"

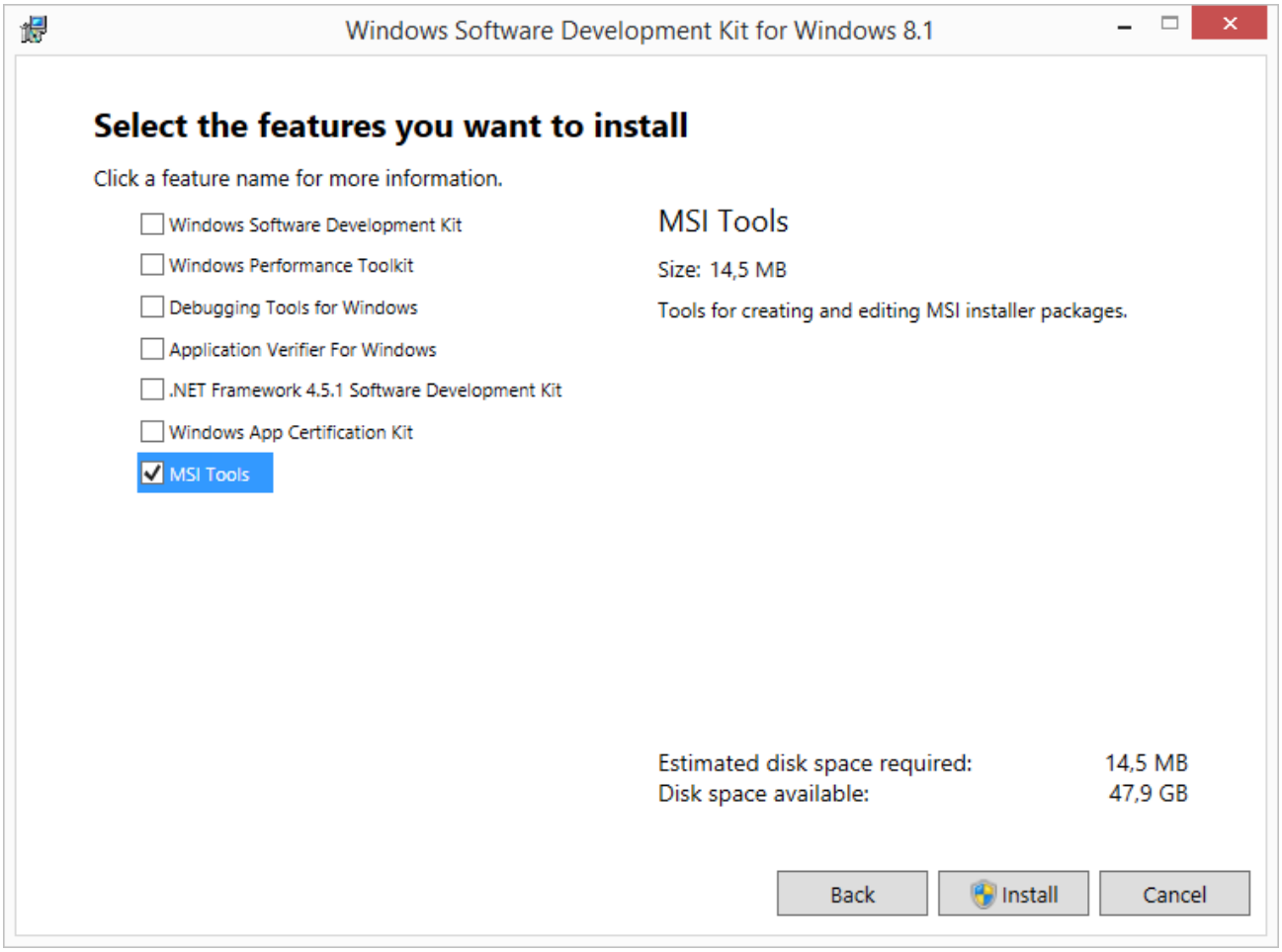


Figure - Select Features to install

2. After installation has finished navigate to C:\Program Files (x86)\Windows Kits\8.1\bin\x86 and execute the ORCA*.msi

3. Open Orca and load the ColumbusInventoryAgent.msi
   Click on "Transform" and then on "New Transform"

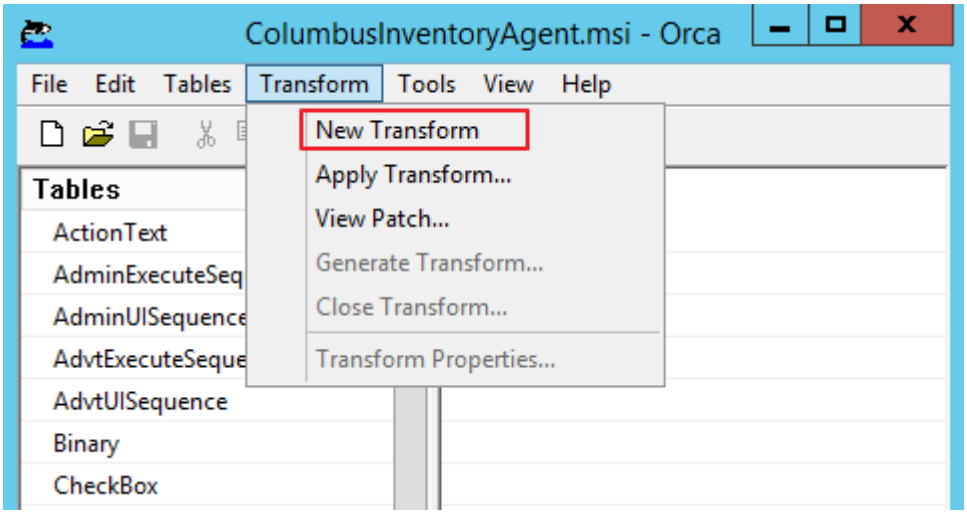


Figure - New Transform

4. Navigate to the "Property" table and change the settings for
   - INVOTB_PORT
   - INVOTB_SERVER
   - AUTOSTARTSERVICE
   - SCANNERFUNCTION
   to the desired values.

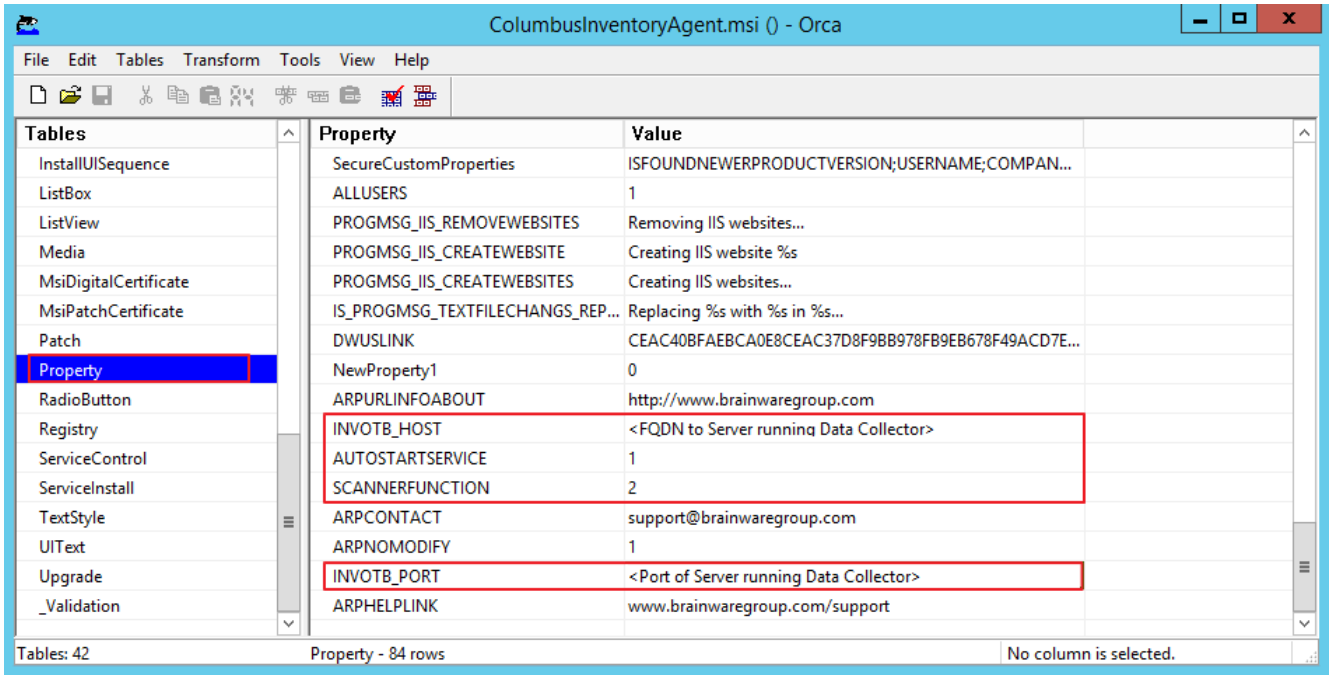


Figure - Property table

Details about the settings can be found in the previous topic

5. Click on Transform > Generate Transform and then select a path for where the MST file should be saved.
   You can then use this MST in the Software Installation settings for the Inventory Agent MSI package in the GPO settings

Creating the GPO

1. Copy the ColumbusInventoryAgent.msi/.mst to a network share. Configure the permissions on the share to ensure that all required users and computers have **Read** access to the installation files.
2. In "Group Policy Management", locate the container on your server (a site, a domain, or an organizational unit (OU)) where you want to advertise the application right click and choose "Create a GPO in this domain, and Link it here..."

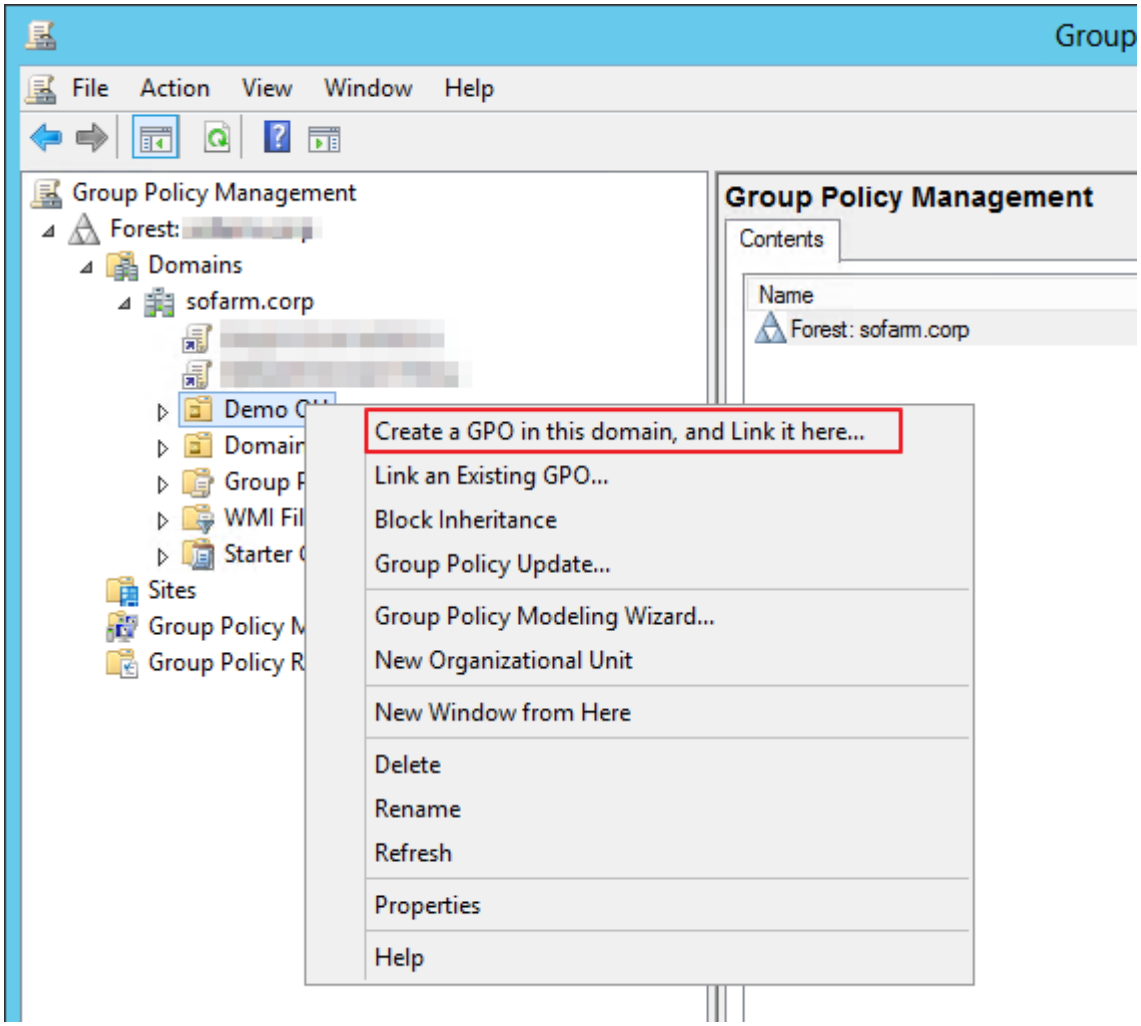


Figure - Create GPO...

3. Give the new GPO a name, in this example "Deploy Columbus Inventory Agent MSI"

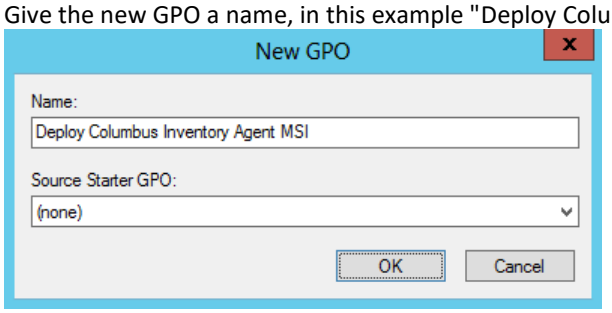


Figure - New GPO

4. Return to "Group Policy Management" and Edit the new GPO

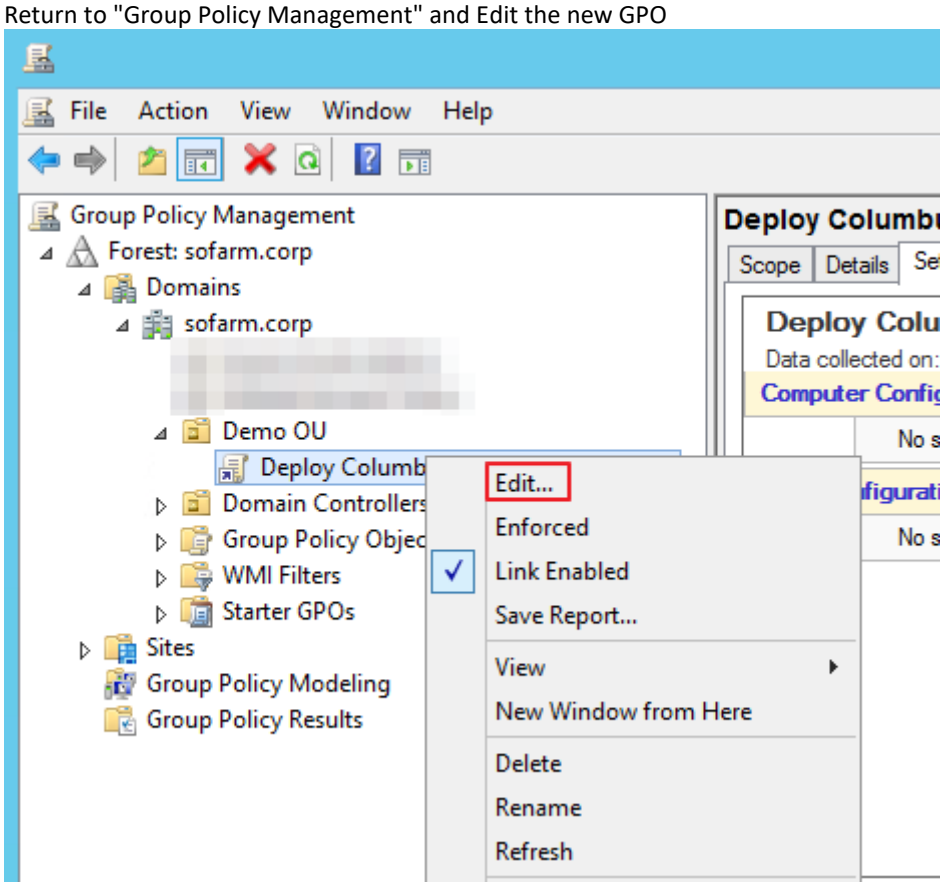


Figure - Edit GPO

5. If you are sharing the application to user accounts, expand the **User Configuration\Software Settings** container in the **Group Policy Management Editor**, right-click **Software Installation,** select **New**, and then select **Package**. If you are sharing the application to computer accounts, expand the **Computer Configuration\Software Settings** container in the **GPO**, right-click **Software Installation**, select **New**, and then select **Package**. See below:

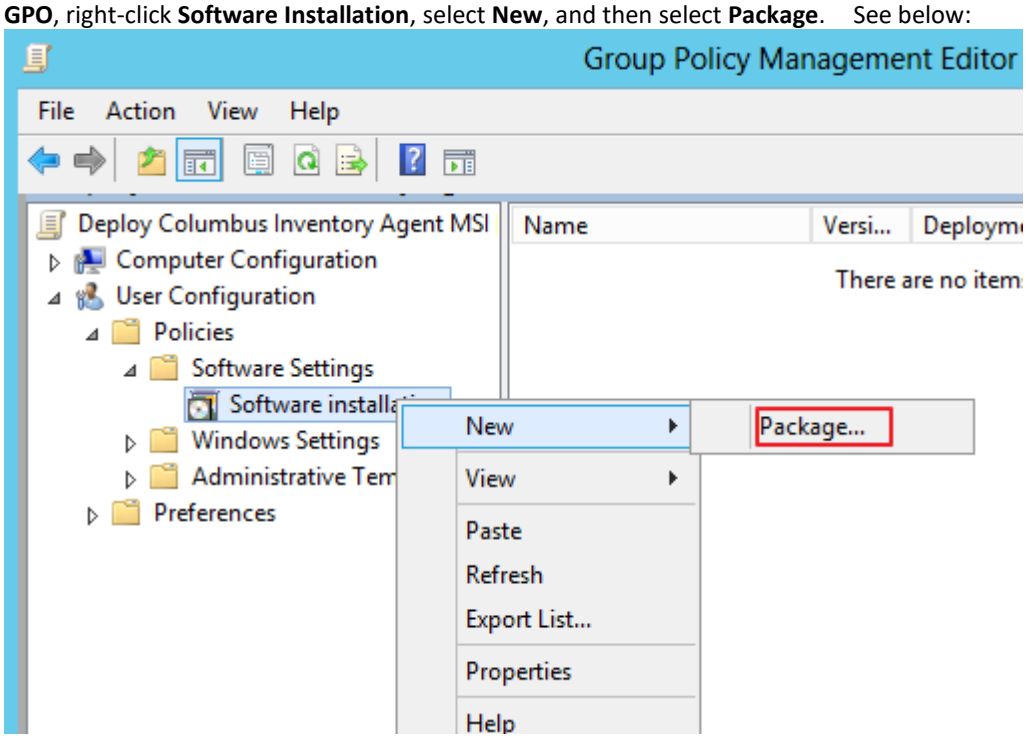


Figure - New Package

6. Select your MSI package then select **Advanced** for the deployment method (see below)

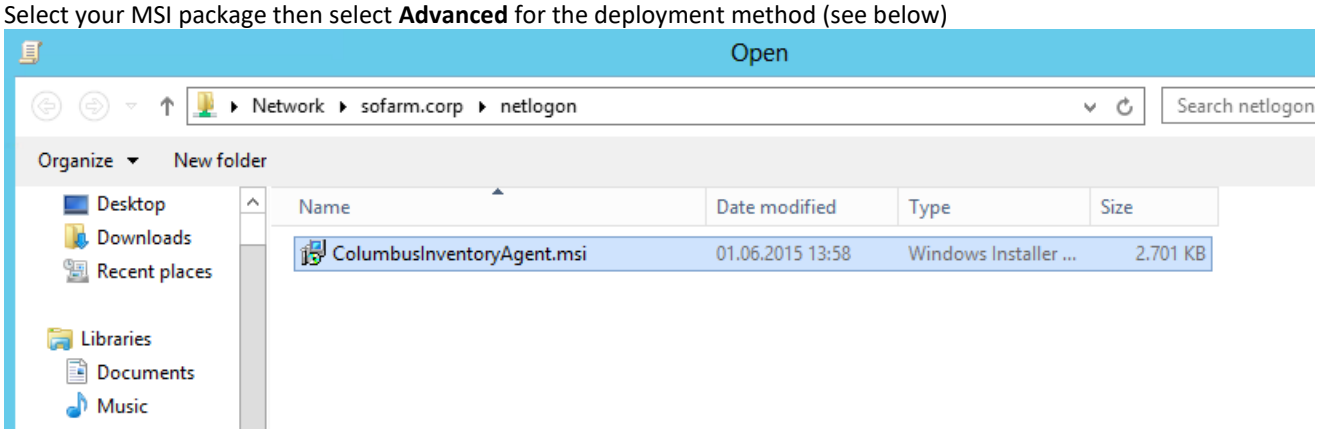


Figure - Choose MSI

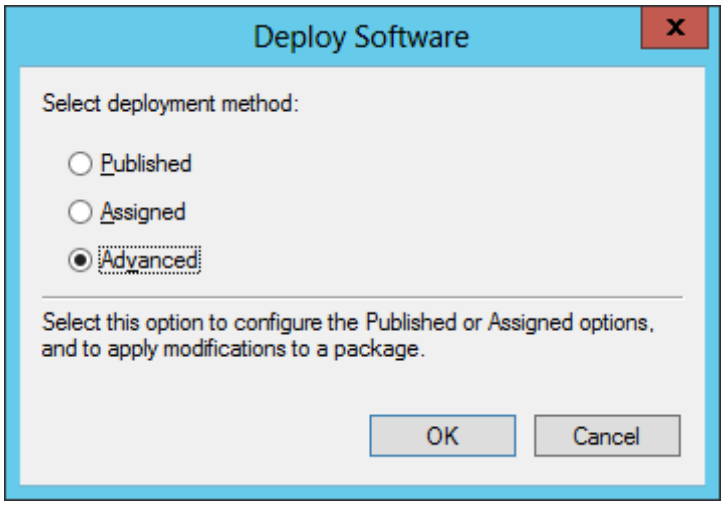


Figure - Deployment Method

7. From the **Deployment** tab, check the Deployment type/Deployment options as seen below (your deployment type/options maybe different depending on your network setup). In this example we use **Assigned** as the deployment type.

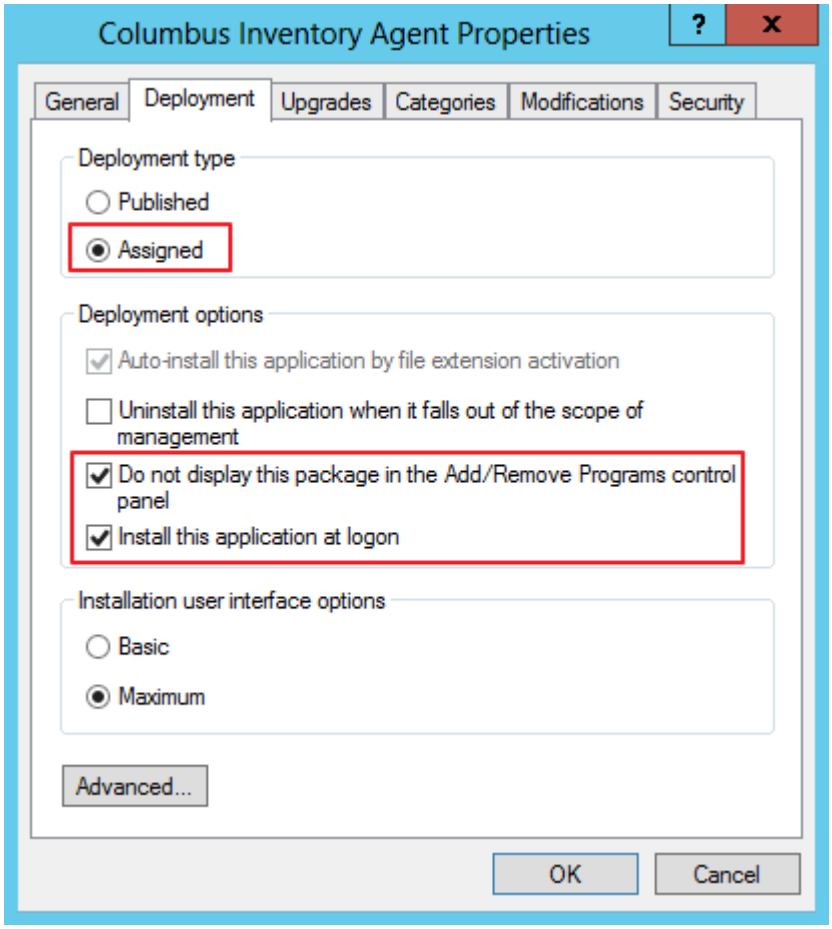


Figure - Deployment

8. From the **Modifications** tab, select your MST file (that customizes your installation) from the network share. See below:

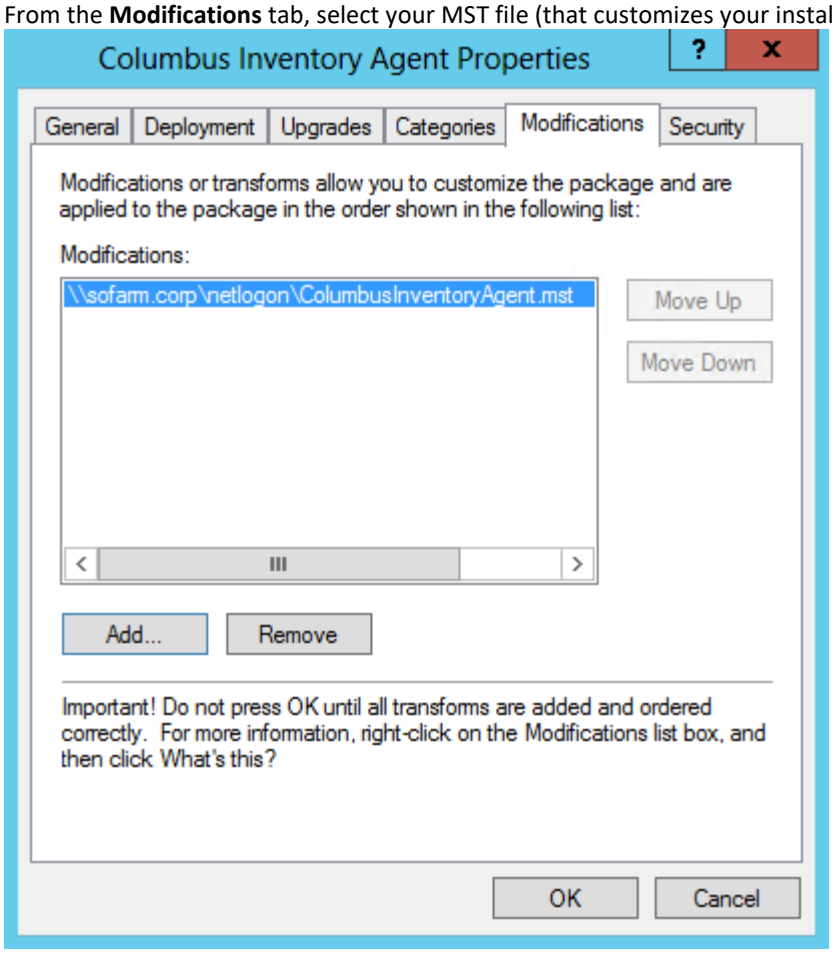


Figure - Modifications

9. Click **OK** to complete the setup.

Attention	This is only an example, please be aware that that some, if not all steps in this guide might be handled differently in your organization, please refer to the responsible member(s) of staff so that deployment can be carried out according to your organizations regulations.

5.1.5 Columbus Inventory Scanner

Columbus Inventory Scanner Location

The inventory scanner can be found in the chosen directory (Figure - Destination Folder (on page 12)) in the sub directory ..\ColumbusInventoryScanner, it consists of the files:

- ColumbusInventoryScanner.cfg
- ColumbusInventoryScanner.exe
- libeay32.dll
- ssleay32.dll
- StartReset.cmd (should not be distributed, will for testing purposes reset the last scan date)

Columbus Inventory Scanner Configuration

The scanner is started by executing the ColumbusInventoryScanner.exe and it is configured with the entries in ColumbusInventoryScanner.cfg.

```
[Scanner]
InvExtensions=.EXE

[Transmitter]
InvOTB_Host=hostname.domain.suffix
InvOTB_Port=24786
```

Section	Config parameter	Default (if empty)	Possible values / Description
Scanner	InvFunction	2	0 = HW, SW, Inv. Items 1 = HW, SW, Inv. Items, File scan
Scanner	InvDrives	All local hard disk drives	CDE (Means, drive C:, D: and E:)
Scanner	InvExtensions	.EXE	List of extensions for which detailed file data will be collected.
Scanner	InvExportPath	%ProgramData%\Columbus	%temp% or %_ExePath%
Scanner	InvUpdateEngine	1	0=disabled 1=enabled
Scanner	InvScanStartPeriod	daily	daily, weekly, monthly
Scanner	InvScanStartDelay	0	n minutes (0-100)
Scanner	InvLastObject	0	1 = Export of personal data • LastLoggedOnUser • LastLoggedOnSAMUser • LastLoggedOnUserSID
Scanner	InvNetwork	0	1 = Export of personal data • MAC1 • MAC2 • MAC3 • MAC4 • IPAddressV4 • IPAddressV6
Scanner	InvLicensee	0	1 = Export of personal data • OS.System.RegisteredUser • OS.System.Organization • OS.System.ProductKey
Transmitter	InvTransmissionMode	3	0 = No Transmission, offline mode 1 = FTP 2 = not used 3 = OTB
Transmitter	InvOTB_Host		FQDN of Data Collector machine
Transmitter	InvOTB_Port	24786	Port of Data Collector machine

Transmitter	InvFTP_Host		FTP-Server hostname
Transmitter	InvFTP_Port		FTP-Server port
Transmitter	InvFTP_User		FTP-Server authentication user (Empty uses anonymous)
Transmitter	InvFTP_Password		FTP-Server authentication password, (Encrypt with cryptit.exe)
DirectoryFilter	InvDirectoryFilter001 ... InvDirectoryFilter999		Windows variables, fixed paths like %windir%* or D:\Data*

After the installation of the Data Collector the scanner is preconfigured and ready to use.

Default Filters

The scanner comes with a default filter set that is described below:

```
[DirectoryFilter]
InvDirectoryFilter000=*\microsoft system center 2012\dpm\dpm\volumes\*
InvDirectoryFilter001=%windir%\$*_$\*
InvDirectoryFilter002=%windir%\*\$*_$\*
InvDirectoryFilter003=%windir%\Installer\*
InvDirectoryFilter004=%windir%\system32\ccm\cache\*
InvDirectoryFilter005=%windir%\WinSxS\*
InvDirectoryFilter006=%windir%\ServicePackFiles\i386\*
```

Columbus Inventory Scanner Execution

The execution of the Inventory Scanner simply requires read access to the two files mentioned in bwScan Location (see "Inventory Scanner Location") and the execution of the ColumbusInventoryScanner.exe

Logon Script

The Inventory Scanner can be triggered to run through a number of automated methods; one of which is by logon scripts. Where a logon script is used, the files for the Inventory Scanner should be copied to a central directory that is easily accessible for all machines such as the NETLOGON directory in AD. When this has been configured, the logon script should be configured to execute ColumbusInventoryScanner.exe.

See the following code-snippet example:

```
Start "\\domain.local\Netlogon\InventoryScanner\ColumbusInventoryScanner.exe"
```

Important	The Inventory Scanner should be executed in a way that does not halt the logon script from further execution of other lines. Failure to do this may result in a delayed logon time for users accessing machines that have been configured to run the script.

Software Distribution

It is possible for any software distribution system to be used to distribute and execute the Inventory Scanner. To do so, simply include the following files in the distribution package (where packaging the files for deploy and execute scenarios), or refer any deployed scripts to a central network location as in the aforementioned section for Logon Scripts:

- ColumbusInventoryScanner.exe
- ColumbusInventoryScanner.cfg

GPO

The execution of the Inventory Scanner is the same as described in the previous section(s) . An example of the GPO settings is shown in the following screenshot.

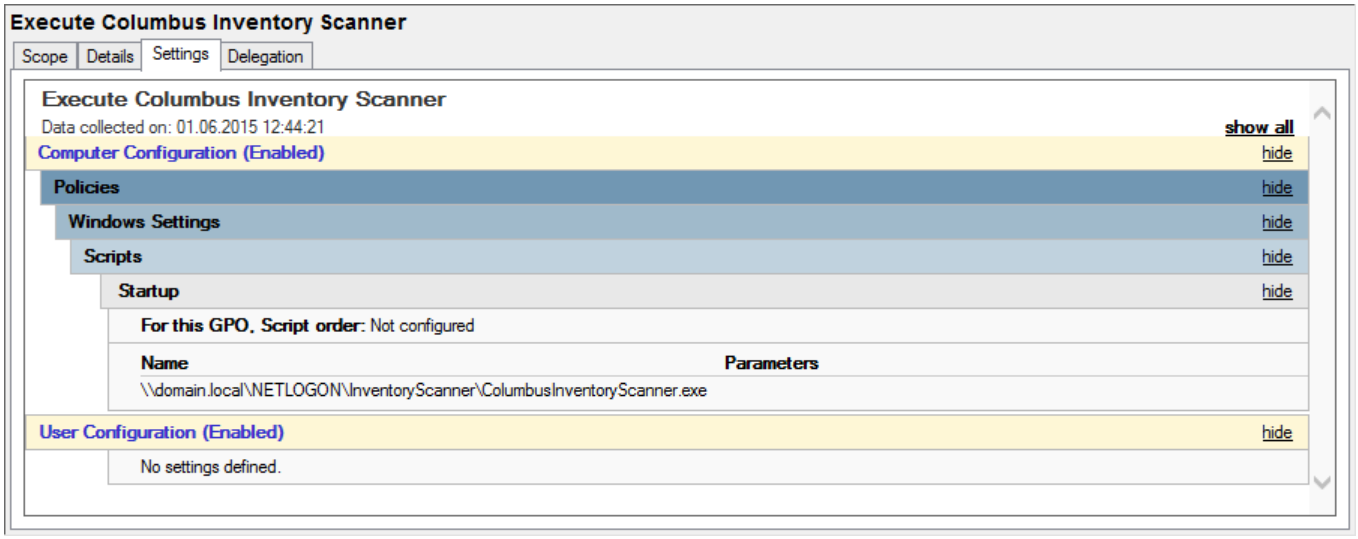


Figure - GPO

Note: It should be observed that group policy execution is synchronous and that the execution of the Inventory Scanner will block further processing of other policies until it has finished. Depending on the machine, the size and number of hard disks and its general performance; this may take a few minutes.

One way to work around this is to use the tool psexec from Sysinternals to start the execution of the Inventory Scanner which once invoked, will allow the GPO processing to continue with processing other policies and run the Inventory Scanner in the background.

Psexec.exe can be placed in the same directory as the Inventory Scanner and then instructed to call a batch file called ColumbusInventoryScanner.cmd with the content psexec /accepteula -d %~dp0.\ColumbusInventoryScanner.exe next to the ColumbusInventoryScanner.exe. This will start the process asynchronously. Psexec can be obtained free of charge from Microsoft: http://technet.microsoft.com/en-us/sysinternals/bb897553.aspx .

Note	The ColumbusInventoryScanner.cmd would then be called from within the GPO instead of calling the Inventory Scanner executable directly.

Scan from USB Stick

In some environments with no network connectivity, it may be necessary to scan machines without transmitting the result directly to a server, e.g. using an USB Stick.

In order to enable this, the Inventory Scanner needs to be instructed to put the resultant scans on the USB stick it is being run from. The following code snippet shows the changes that need to be made to the ColumbusInventoryScanner.cfg file:

```
[Scanner]
InvExportPath="%_ExePath%"
[Transmitter]
InvTransmissionMode=0
```

For information about how to include the Scanner Add-on DLLs into the standalone Scanner please see Advanced Inventory with Scanner Add-on DLLs

Upload of scan results

After the scans have been collected using the USB Inventory Scanner they have to be placed in the ScanResults folder where they will then be processed. The ScanResults folder is a subfolder of the folder specified as data directory during the installation.

You can also find the valid path in the SpiderDataCollector.cfg Section "[OTBServer]" Variable "DataDirectory". The SpiderDataCollector.cfg is in the application directory of the Data Collector.

Attention	The setup will create a folder containing the Inventory Scanner preconfigured for use on an USB stick. It can be found in the installation directory and is named ..\ColumbusInventoryScanner-USB

5.1.6 Columbus Inventory Scanner Resetting Last Scan Date

For testing purposes it may be necessary to reset the date the scanner has last scanned a certain machine.

This is achieved by deleting the registry key (System User):

```
Key: HKEY_USERS\S-1-5-18\Software\BrainWare\Columbus\7\InvScanner
Value: LastRun
```

Alternatively, the registry key may exist in the logged on user's registry hive and so the following key will need to be deleted instead:

```
Key: HKEY_CURRENT_USER\Software\BrainWare\Columbus\7\InvScanner
Value: LastRun
```

5.1.7 Discovered Hardware Items

Hardware Item	Description	Example
DomainName	Full qualified domain name	stark.industries.local
HostName	Hostname	WRK-P-TOST001
Manufacturer	Device Maker	Dell Inc.
Model	Device Model	OptiPlex 7010
MAC1	1st MAC address	F8-B1-56-A3-BE-2F
MAC2	2nd MAC address	
MAC3	3rd MAC address	
MAC4	4th MAC address	
Serial	System Serialnumber	50U8AA2
OSClass	Server, Workstation etc.	Client
DeviceChassis	Notebook, Server etc.	Mini Tower

Hardware Item	Description	Example
ProcessorManufacturer	Processor	Intel
ProcessorType	Processor Type	Core i7-3770
ProcessorSpeed	Processor Speed e.g. 2300	3400
CPUCount	Total of physical CPUs (=Total of instances)	1
CPUCoreCount	Total of CPU cores (from all CPUs)	4
CPULogicalCount	Total of Logical Processors (from all CPUs)	8
UUID	UUID (pretty formatted)	C1FB8C42-E7F7-422E-9211-757E3BFD82F5
InventorySource	Name and Version of Inventory Client	ColumbusInventoryAgent.exe 7.4.0.131
ScanDate	Date the machine was scanned	2014-03-31T10:03:32
DiskTotalMB	Total size of all fixed disks	238472
DiskFreeMB	Total free space	189312
GraphicAdapter	Name of graphic adapter	AMD Radeon HD 7470
GraphicMemory	RAM size of graphic adapter	1024
MemoryMB	Total memory of machine	16338
IPAddressV4	Current IP Address v4 of machine	10.10.20.30
IPAddressV6	Current IP Address v6 of machine	fe80::d8a9:dd4c:619b:ef5
CPUArchitecture	CPU Architecture	amd64
OSCaption	Name of operating system installed	Microsoft Windows 8.1 Enterprise
DomainNetBIOS	NetBIOS Domain of machine	STARKINDUSTRIES
LastLoggedOnUser	Last User that was logged on	STARKINDUSTRIES\Tony.Stark
BIOSVendor	Vendor of the BIOS	Dell Inc.
BIOSVersion	Version of the BIOS	A16
BIOSDate	Date of the BIOS	09.09.2013
URN	for future use	
Class	for future use	
ComputerHomePath	for future use	
LastLoggedOnSAMUser	SAM Account Name of logged on user	STARKINDUSTRIES\Tony.Stark
LastLoggedOnUserSID	SID of logged on user	S-1-5-21-3427917592-4004333369-2915694803-2802

5.1.8 Feature Comparison

The following table shows a feature comparison between the Inventory Agent and the Inventory Scanner inventory mechanisms.

Feature	Inventory Scanner	Inventory Agent
Installable		X
Start with Login script	X	
Auto update (config & engine)		X
HW & SW inventory	X	X

Feature	Inventory Scanner	Inventory Agent
SW Metering		X
Multi-user support		X
Terminal server support		X
Start without user session		X
Support for travelling user		X
Offline usage	X	X

5.1.9 Advanced Inventory with Scanner Add-on DLLs

In order to be able achieve more advanced recognition of software products, the Inventory Agent and Inventory Scanner can utilize DLLs that contain additional logic (e.g. SQL Server detection, Embedded Operating System Recognition).

The DLLs are dependent on .NET and are available for .NET2 and .NET4. In order for the additional recognition to function, either Microsoft .NET 2 or Microsoft .NET 4 must be installed on the scanned machine.

Inventory Scanner and Inventory Agent will automatically update the DLLs from their OTB Server before the scan takes place.

In order for this process to function, the DLLs (ScannerAddon.Net2.dll and ScannerAddon.Net4.dll) must be available in the folder "Files_Scanner" where the scanner transmits its results files. The location of this folder is specified during Data Collector Setup and is written to SpiderDataCollector.cfg, Section [OTBServer] Variable "DataDirectory".

During an update of the Data Collector, the aforementioned files are automatically updated through the setup.

Attention	When using a standalone version of the Inventory Scanner (e.g. on an USB Stick) the DLLs must be placed in the same folder as the Inventory Scanner files (ColumbusInventoryScanner.exe, ColumbusInventoryScanner.cfg).

5.1.10 Additional Inventory Items from Registry

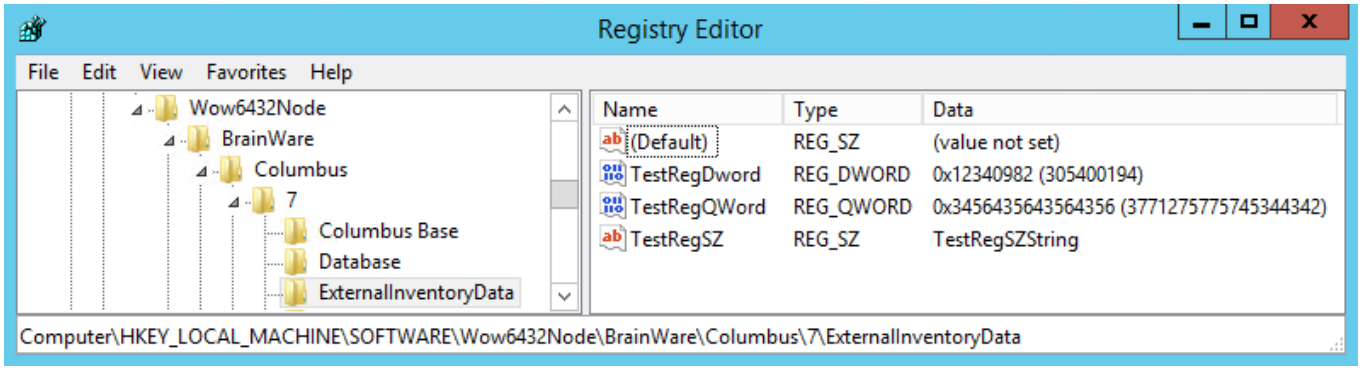
Additional inventory items in the form of registry keys can be placed on every machine that is scanned. Once set those items will be queried when the next Columbus Inventory run occurs.

```
Key: HKEY_LOCAL_MACHINE\SOFTWARE\<Wow6432Node>\Brainware\Columbus\7\ExternalInventoryData
```

Supported Values

- REG_SZ
- REG_DWORD
- REG_QWORD

Example



5.1.11 SSL Secured Transmission

Communication with the Spider Data Collector can be secured by SSL (RSA 2048Bit).

Upon installation of the Data Collector the necessary files are generated and placed in: %ProgramData%\Columbus

SSL usage can be enabled by setting the following registry keys.

```
Key: HKEY_LOCAL_MACHINE\SOFTWARE\<Wow6432Node>\BrainWare\Columbus\7\
"OTBEncryptionUseSSL"="1"
"OTBEncryptionForceUseSSL"="0"
```

5.2 Mac OS

5.2.1 Columbus Inventory Scanner Location

The Columbus Inventory Scanner (CIS) is located in the installation directory (Figure - Installation path (on page 12)) in the subfolder "..\cis", this directory contains the following files:

- cis.prv
- ColumbusInventoryScanner.cfg
- ColumbusInventoryScanner.tar
- setup.sh

5.2.2 Columbus Inventory Scanner Configuration

Before running the CIS for the first time, the configuration file ("ColumbusInventoryScanner.cfg") needs to be adjusted. Most important is the Area "[Transmitter]". With "InvTransmissionMode" the protocol used for the transmission of the result can be specified (1=sftp, 2=ssh, 0=no Communication).

Additionally it hast to specified with which system is being communicated. For this the entries InvTransmissionHost(x), InvTransmissionUser and InvTransmissionKey need to be adjusted.

If the SSH/SFTP key has to be exchanged, this has to be done in the „etc/.ssh“ folder. A new key can only be created via the automatic key generation as described in SFTP Server Configuration (on page 25).

An example is shown below:

```
InvTransmissionMode=1
InvTransmissionHost1=sdc-cis
InvTransmissionPort = 22
InvTransmissionUser = cis
InvTransmissionKey = /path/to/privateKey
```

Attention: All of these settings are preconfigured after the installation, a new key has also been generated. Please use the files that are placed in the same directory as setup.sh!

Most of the settings put in the configuration file can be passed to the binary directly as a runtime argument. The arguments passed directly take precedence to those that are specified in the configuration file. For a list of arguments please run the binary from the "/bin" folder with the argument „--full-help“.

The following table show the configuration items that can be set in „ColumbusInventoryScanner.cfg"

Should this file be empty or non-existing, default values will be used.

Section	Parameter	Default value	Possible Values / Description
Scanner	InvUpdateScanner	1	Update of the binaries 0 = no Update 1 = Update
Scanner	InvUpdatePackage	1	Update of the packages 0 = no Update 1 = Update
Scanner	InvScanStartTime	22:00	Time of execution hh:mm
Scanner	InvScanStartDelay	0	Delay of scanning start by n minutes

Section	Parameter	Default value	Possible Values / Description
Scanner	InvScanStartPeriod	daily	Period of execution daily = once a day weekly = once a week
Scanner	InvPackageExec	$OS	Package that will be executed e.g. „macos“
Transmitter	InvTransmissionMode	1	Type of data transmission 0 = no transmission (offline) 1 = sftp 2 = scp
Transmitter	InvTransmissionHost1		Hostname/IP of target server (empty = offline)
Transmitter	InvTransmissionHost2		Optional. Alternative target server (Hostname/IP)
Transmitter	InvTransmissionHost3		Optional. Alternative target server (Hostname/IP)
Transmitter	InvTransmissionPort	22	Port of transmission hosts
Transmitter	InvTransmissionUser	cis	Username used for transmission host
Transmitter	InvTransmissionKey	etc/.ssh/cis.prv	Private key for authentication (without password!)
Transmitter	InvTransmissionTimeout	60	Timespan in seconds (per host) to try transmission
Transmitter	InvTransmissionRetry	3	Amount of retries
DirectoryFilter	InvStorage	10	Amount of result files that are kept
Runtime	InvLogLevel	2	Log level 0 – 10 0 = only errors 10 = as much output as possible
Runtime	InvLogDir	log/	path for storage of logfiles
Runtime	InvLogFileName	ColumbusInventoryScanner.log	Name of the logfile
Runtime	InvLogOutput	0	1 = additionally log messages to Stdout 0 = no log messages to Stdout
Runtime	InvLogSyslog	0	0 = only log message to logfile 1 = only log messages to syslog 2 = log messages to Stdout and syslog
Runtime	InvHostname	$HOSTNAME	Hostname of the system, this name will be used to generate the output file
Runtime	InvPlatform	$OS	Type of commands that are executed from the package e.g. „macos“
Runtime	InvOutput	output/	path to store the results in

Section	Parameter	Default value	Possible Values / Description
Runtime	InvTimeout	60	Execution timespan for each command executed from package

5.2.3 Columbus Inventory Scanner Installation

The Columbus Inventory Scanner is installed by running the „setup.sh" script. The script has to be called by a user with administrative rights.

Important:	The setup.sh file must be made executable before running the installation.

An example for the installation with the preconfigured values looks like this:

```
sudo ./setup.sh c -k --headless
```

With this call, the tar-archive will be extracted to the default folder „/opt/ColumbusInventoryScanner/". The script will then ask if the settings are correct. During setup the script will report the progress.

```
tw@vm-dev-tw:/export/transfer/tw/CDA/package_build/finished$ sudo ./setup.sh --headless
[sudo] password for tw:
TAR extract to "/opt/ColumbusInventoryScanner"...          ok
get os...                                                  ok
tidy up bin/...                                            ok
tidy up software/...                                       ok
CHOWN "root"...                                            ok
CHMOD "bin"...                                             ok
CHMOD "software"...                                        ok
CHMOD "scanner_wrapper.sh"...                              ok
CHMOD ssh-keys...                                          ok
run scanner_wrapper.sh...                                  ok
cleaning up...                                             ok
```

Figure- Example of installation

Option	Parameter	Function
-a |--archive	<DATEI>	Path and filename to the cis archive. If only one CIS[...].tar.gz archive is available this one will be chosen.
-i | --installdir	<PFAD>	Installation directory, default is /opt/ColumbusInventoryScanner
-c |--config	[<DATEI>]	Path and name of the configuration file. If no filename is specified the ColumbusInventoryScanner.cfg which is placed next to the setup.sh is used.
-k | --key	[<DATEI>]	Path and name to the cis.prv file. This one will be used to overwrite the default file. If no filename is specified the file found next to setup.sh is used.
--headless		No prompts to the user during installation.
-n | --no-run		Do not execute scanner after installation
-h |--help		Display the help text

```
RELEASE: 1.0.2

usage: ./setup.sh  [OPTIONS]
    OPTIONS:
    -a|--archive <FILE>      - Provide the path/name of the CIS archive
                               if only one CIS[..].tar.gz is found; script can default to this one
    -i|--installdir <PATH>   - Installationdirectory (defaults to /opt/ColumbusInventoryScanner)
    -c|--config [<FILE>]     - define ColumbusInventoryScanner.cfg file to overwrite existing file
                               if [<FILE>] is empty ./ColumbusInventoryScanner.cfg will be used
    -k|--key [<FILE>]        - define key-file to overwrite existing cis.prv file
                               if [<FILE>] is empty ./cis.prv will be used
    --headless               - no user confirmation
    -n|--no-run              - don't run the scanner after setup
    -h|--help                - this message
```

Figure - Parameter setup.sh

Note: The argument „--headless“ allows an automatic installation. When this is used **only one** ColumbusInventoryScanner.tar archive may be in the same folder, this one is then installed and the binary is executed.

Important: It is recommended to use the configuration and key files that are placed in ..\cis during the installation. For this copy all the files in the directory to the target system and call the „setup.sh“ script with the arguments „–c“ and „-k“. The configuration file and private key contained in the tar file will then be replaced with the „ColumbusInventoryScanner.cfg“ and „cis.prv“ from the current folder. [As an alternative you can use the parameter „-c /path/to/new/configfile“. In this case „configfile“ will replace the ColumbusInventoryScanner.cfg file in the installation directory.

In the installation directory toe following files/folders can be found:

Element	Type	Description
bin/	Folder	Binaries are placed in this folder
etc/	Folder	Configuration file and ssh key are found in this folder
software/	Folder	Helping libraries are placed here
scanner_wrapper.sh	Script	Script to execute the correct binary

5.2.4 Columbus Inventory Scanner Execution

By default the binary is executed by a Root - Cronjob according the schedule defined in the configuration file (default: daily: 22:00).

The binary can be executed manually with the following command::

```
sudo .bin/ColumbusInventoryScanner_Darwin_x86_64 -p <PackageName>
```

The package name has to correspond with a package from the folder „package/“ or with the „package.tar“ file (e.g. „macos“). If the SFTP/SCP connection is successful the package.tar is retrieved from the server and extracted. If the download does not work, the local package (if exists) is executed.

If no package name is provided, the package (if present) from the config file is executed. If the entry is empty, the package matching the OS-Name (e.g.. „macos“) is executed.

For an easier handling the script „scanner_wrapper.sh“ can be executed:

```
sudo ./scanner_wrapper.sh
```

This script automatically determines $OS and $Architecture and executes the matching binary with the default settings. The wrapper script will report to stdout about the progress.

Process

The binary will execute the following steps.

- Read the current configuration file
- Retrieve a new configuration file from server
- Read the new configuration file
- Create/update cron job
- (optional) Updates binaries and packages
- Execute Package <PackageName> or $OS
- Pack results and send to server
- Clean up

Successful execution returns the exit code „0“. The process can be retraced in the log file in the „log/“ folder. There are additional log options, e.g. „—LogOutput“ will log the process to stdout.

Data Center Inventory (Linux / Unix)

Inventory agents are available for inventorying different server platforms (Unix / Linux / etc.), which can be connected directly to the Spider Data Center Appliance. These do not deliver date to the installed Spider Data Collector.

6.1 Requirements

The following sections explain the prerequisites for using the Data Center Inventory.

6.1.1 Definition of terms

The scan engine of the Spider Data Center platform enables data retrieval from server systems by the Spider Data Center Server. The data query can take place through various mechanisms. This chapter only describes the data query by using the Spider Data Center agent.

Communication between the Spider Data Center Server and the Spider Data Center agents takes place via TCP / IP and two ports reserved for the Spider Data Center protocol.

6.1.2 Network ports

Ports 9616 and **9617** must be enabled bidirectionally for the communication of the Spider Data Center System with the Spider Data Center Agent on the server systems to be queried. This applies both to firewall systems in the network and to firewalls on the target systems.

6.1.3 Server systems

To install the agent packages to the server systems administrator rights are required on these server systems. Local firewalls must have enabled **ports 9616** and **9617** bidirectionally to communicate with the Spider Data Center Server.

6.1.4 UUID-Generator

The UUID-Generator provides two UUIDv4 for each System: One named „Generated" and the other „Machine". The „Generated" UUID is a randomly created UUID by the generator itself, the „Machine" UUID is provided by the system.

If no „Machine" UUID can be determined, the value „Machine: 00000000-0000-0000-0000-000000000000" is returned.

To prevent a new UUID from being generated with each call, the generated UUID is saved in a file "eRunbook.uuid". Before each generation run, a check is made as to whether the file exists. Regeneration is only triggered if no file exists.

The file „eRunbook.uuid" is located by default at:

- Unix/Linux: /var/eRunbook/
- Windows: %AppData%\eRunbook\
  corresponds to %SystemDrive%\ProgramData\eRunbook\ (Windows® 7 and newer) or %SystemDrive%\Documents and Settings\All Users\Application Data\eRunbook\ (Windows® XP/Server2003).

Possible configuration settings in the specific eRunbook.conf. (<agent|scriptmodule>/etc/eRunbook.conf) are:

```
create_uuid_file=<yes|no> (create the „eRunbook_uuid" file? Default is "yes")
dir_uuid_file_win=
dir_uuid_file_unix=
```

The program is delivered as binray for Linux (x86, x64), Windows (x86), Solaris (x86, Sparc), HPUX, AIX and MacOS.

6.2 Oracle databases

The agents determine the license status of the Oracle databases on the queried server systems. In addition to system information, information from the Oracle databases must also be queried.

Note:	The agents do not read out any application-specific data. The agents do not read out customer data.

The database query does not require a dedicated user in the database to be queried in order to collect all necessary information.

All running instances are recognized on the systems. After changing to the process owner's context, read access to the database takes place with his UID. Alternatively, a new database user can be created or an existing user can be used. These database users only need read access.

Grant scripts are provided to create and assign these users.

The database user can be identical on all server systems for all Oracle databases running there.

6.2.1 Execution of the grant scripts

The grant script **MUST** be executed as a SYSDBA (or comparable role) with the right to create users! The grant scripts must be executed once for each database instance to be queried.

The grant script is executed like this:

```
@novaratio_grantscript.sql <user> <password> <tablespace> <ORACLE_SID>
```

The parameters of the grant script have the following meaning:

- <user> is a new username.
  If it already exists, a corresponding message will be returned.
- <password> is the password with which the new user can be logged on.
  The password follows the general rules of the specified database guidelines.
- <tablespace> is an existing table space.
  If this is missing, an error message will be returned and the user will not be created.
- <SID> is the ORACLE_SID of the database where the user is to be created.
  If it does not exist, an error message will be returned and the user will not be created.

Important for Oracle 12c with Pluggable Databases (PDB):

The grant script **MUST be executed in CDB$ROOT**. It automatically creates the transferred user in CDB$ROOT as well as in all PDBs generated for the CDB.

6.2.2 Registration of the credentials

If the data query of the server systems is carried out via a specific database user, the user name and password must be stored.

The administrator must store these credentials in a corresponding credential file on the respective server systems. This file can have multiple credentials, i.e. Valid combinations of user name and password, contain and can be used identically on several server systems.

The entries in the credentials file are line-oriented with a **tab as separator** between user name and password:

```
User1	password1
User2	password2
```

Comment lines are not allowed.

On Windows, the credentials file is expected in the following path after installing the agent:

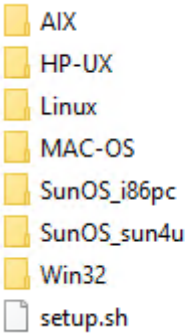
```
%Program Files (x86)%\eRunbook\product\agent\tools\login
```

For Unix / Linux, the credential file is stored after the agent installation in the following path:

```
/opt/eRunbook/product/agent/tools/login
```

6.3 Installation of the agents

The installation files for the respective operating systems are located in the installation directory (Figure - Installation path (on page 12)) in the "Inventory for Data Center" subdirectory. The following further subdirectories are located there:



6.3.1 Linux

Both RPM and DEB packages are available for installation under Linux.

RPM packages

The installation of the signed RPM packages requires root rights. The packages can only be installed if the signature has been recognized. To recognize the signature, the key must be imported before installation.

For this execute e.g.:

```
rpm –import <PFAD>/signatur.key
```

The package can then be installed. The signature can be checked manually with

```
rpm --checksig <Paketname>
```

To switch off the signature check, the --nosignature option of rpm can be used:

```
rpm -Uv --nosignature <Paketname>
```

DEB packages

The installation of the signed DEB packages requires root rights.

```
dpkg -i <Paketname>
```

Here the signature can only be checked if the "debsigs" package has been installed. The signature can then be checked as follows:

```
debsig-verify <Pfad/Paketname>
```

6.3.2 Solaris, HPUX, AIX

The installation files must be stored on the target system in a specific structure that is delivered with the system. The installation of the UNIX agent is started as root user with the following command:

```
./setup.sh --agent
```

The success of the installation can be checked with the following command:

```
#ps –aef | grep eRunbook_agent
```

6.3.3 Mac OS

Note:	For Mac OS, as an alternative to the Columbus Inventory Scanner, this agent can be selected, which delivers directly to the Spider Data Center Appliance.

The installation files must be stored on the target system in a specific structure that is delivered with the system. The installation of the Mac OS Agent is started as root user with the following command:

```
./setup.sh --agent
```

The success of the installation can be checked with the following command:

```
#ps –aef | grep eRunbook_agent
```

It must be ensured that the setup.sh script is executed by the root user. The admin user does not have the necessary rights.

6.3.4 Windows

Note:	If Oracle databases are used on Windows Server, this agent must be used.

The agent is installed under Windows from a DOS shell with administrator rights with the following command:

```
msiexec /i eRunbookAgent-<VERSION>.msi /qn
```

The agent is automatically started as a service after installation.

6.4 VMware vCenter

To calculate Oracle license usage in connection with VCenter, i.a. additional hardware information from the hosts used is required.

The hardware information of the ESXi hosts is queried via the managing VCenter. To do this, the VMware vSphere PowerCLI must be set up on the vCenter servers. An existing user with read rights is required for the query.

The query is carried out by the Windows agent on the VCenter Server.

Registration of the credentials

The login data is stored in the vmlogin file on the respective vCenterServer:

```
%Program Files (x86)%\eRunbook\product\agent\tools
```

The entries in the credentials file are line-oriented with a tab as separator between user name and password:

```
User    password
```

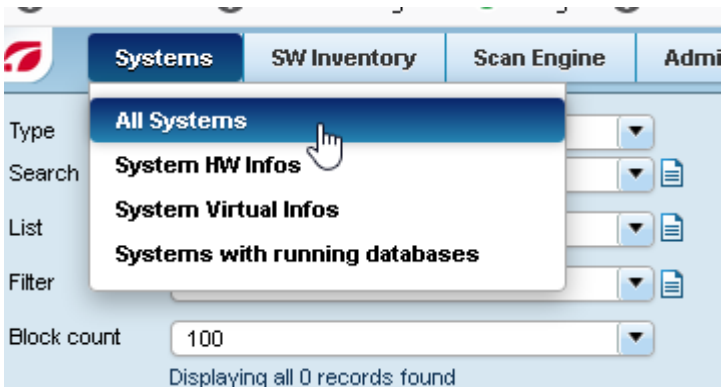
6.5 Set up agents in the Spider Data Center Appliance

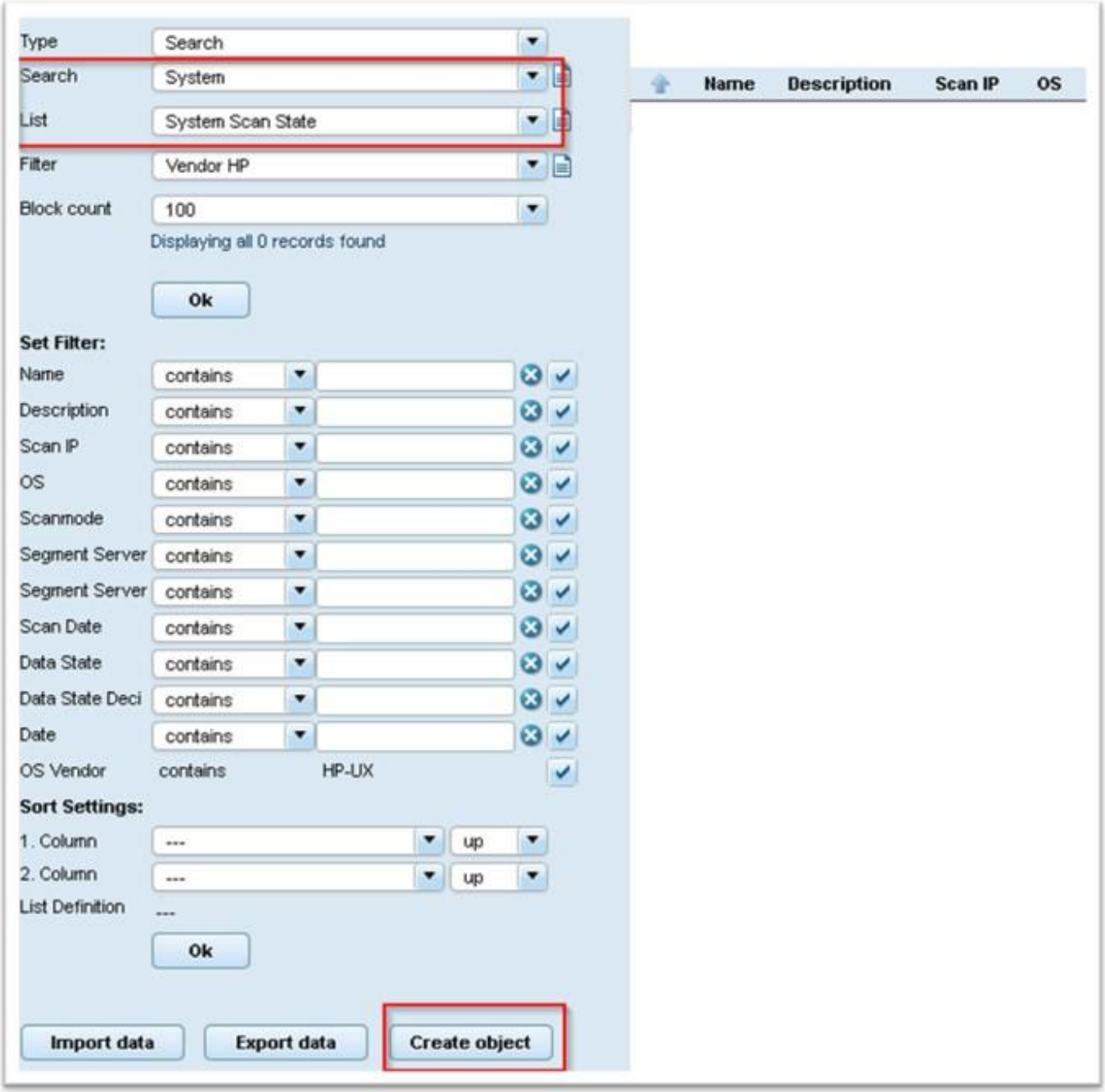
There are two ways to create a system in the Spider Data Center Appliance instance:

1. Create with the editor
2. Create by import

6.5.1 Create with the editor

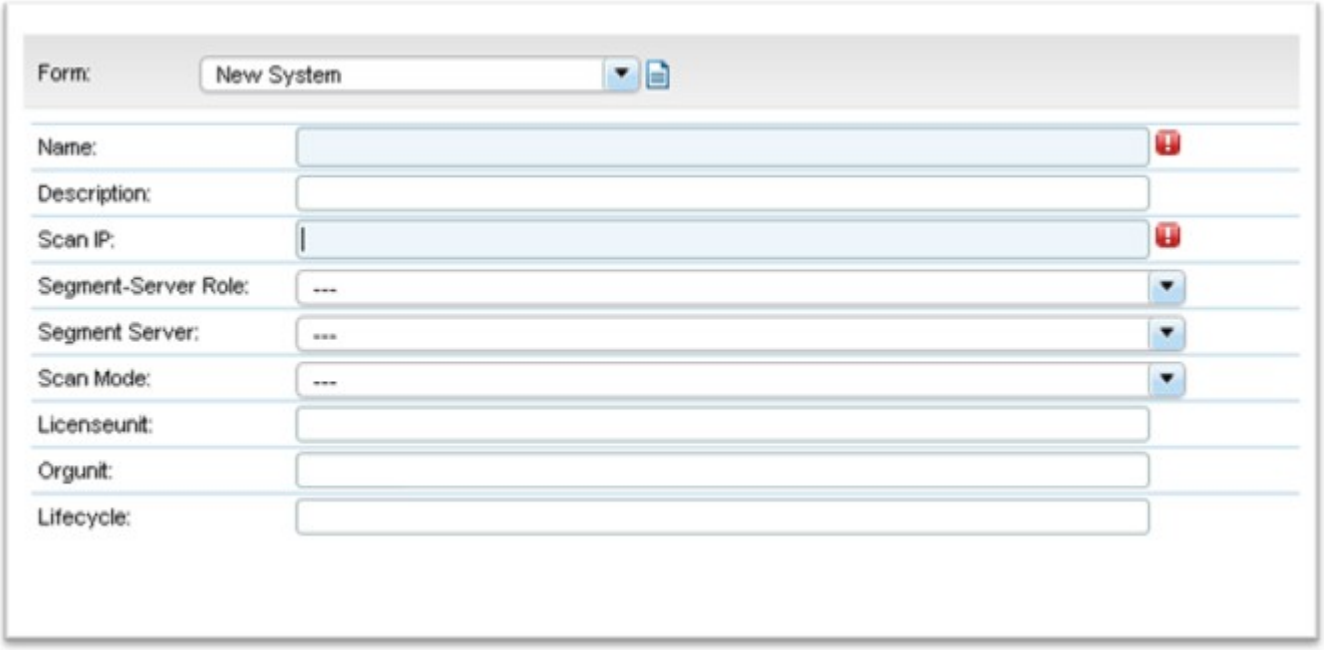
Via the menu item **Systems> All Systems**, the list of configured systems can be displayed. Initially it is empty:




Type
Search
Search
System
List
System Scan State
Filter
Vendor HP
Block count
100
Displaying all 0 records found
Ok
Set Filter:
Name
contains
Description
contains
Scan IP
contains
OS
contains
Scanmode
contains
Segment Server
contains
Segment Server
contains
Scan Date
contains
Data State
contains
Data State Deci
contains
Date
contains
OS Vendor
contains
HP-UX
Sort Settings:
1. Column
---
up
2. Column
---
up
List Definition

Ok
Import data
Export data
Create object
Name
Description
Scan IP
OS

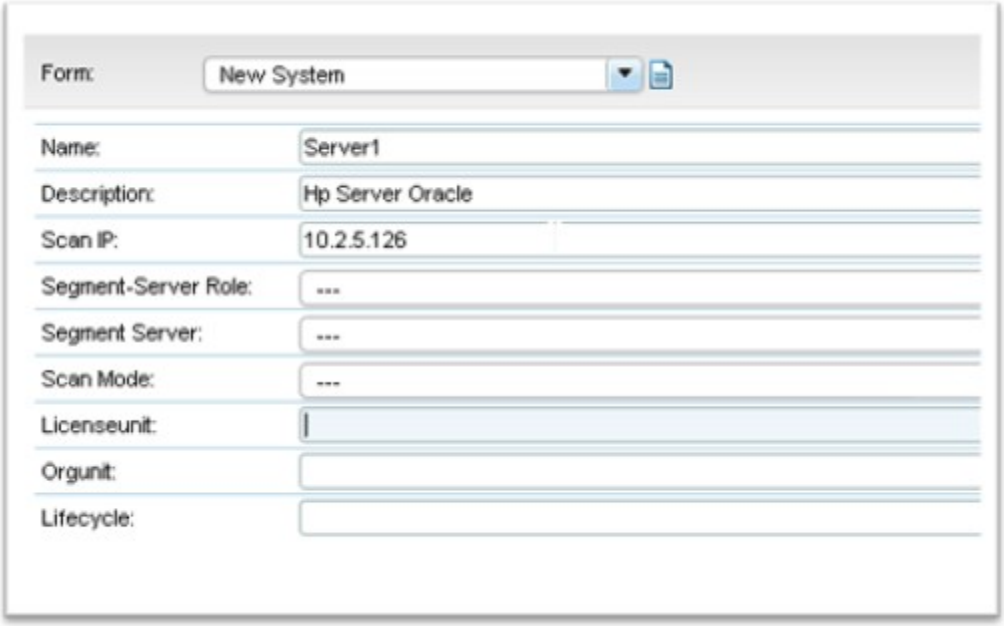
With the button "Create object" at the bottom right, the editor can be opened for entering new systems.



The mandatory fields "System name" and "IP" (IPv4) are highlighted by a red symbol to the right of the input field. Additional fields do not have to be filled, however it is strongly recommended that the "Description" field is not left blank.

Note:	The system name should be the host name of the system. A * can also be entered instead of the IP, if the system names can be resolved via DNS on the appliance.

Example:



6.5.2 Import of large quantities of systems

In order to register several systems at the same time, lists can be imported in Excel format.

Note: The formats XLS and CSV are supported, but not XLSX!

The first line of the Excel spreadsheet **must** contain the column headings:

- Name
- Description
- Scan IP

The second line of the Excel spreadsheet **must** contain the following values:

- $attrib-ute:system:class_system_field_name
- $attrib-ute:system:class_system_field_description
- $attrib-ute:system:class_system_field_scan_ip

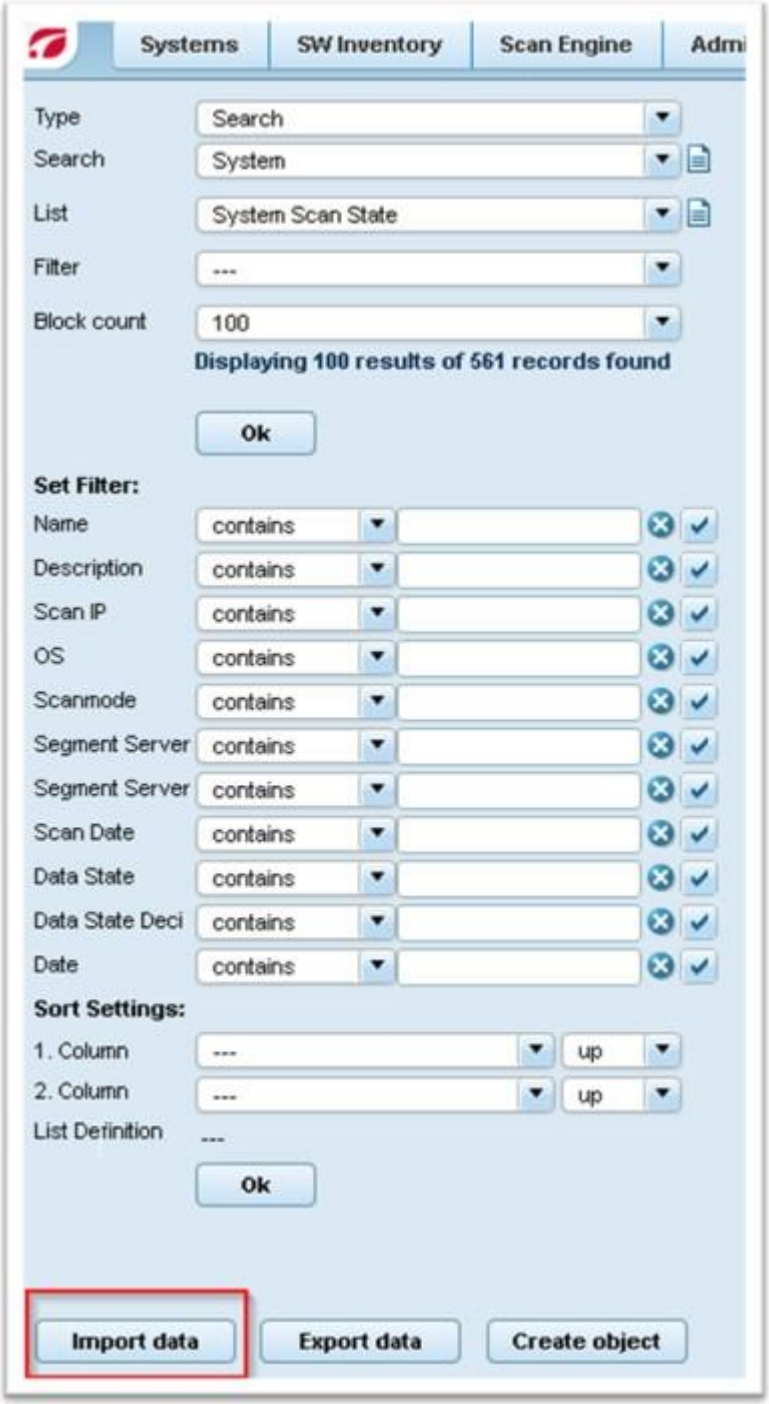
All other lines contain the values of the systems to be imported.

Example:

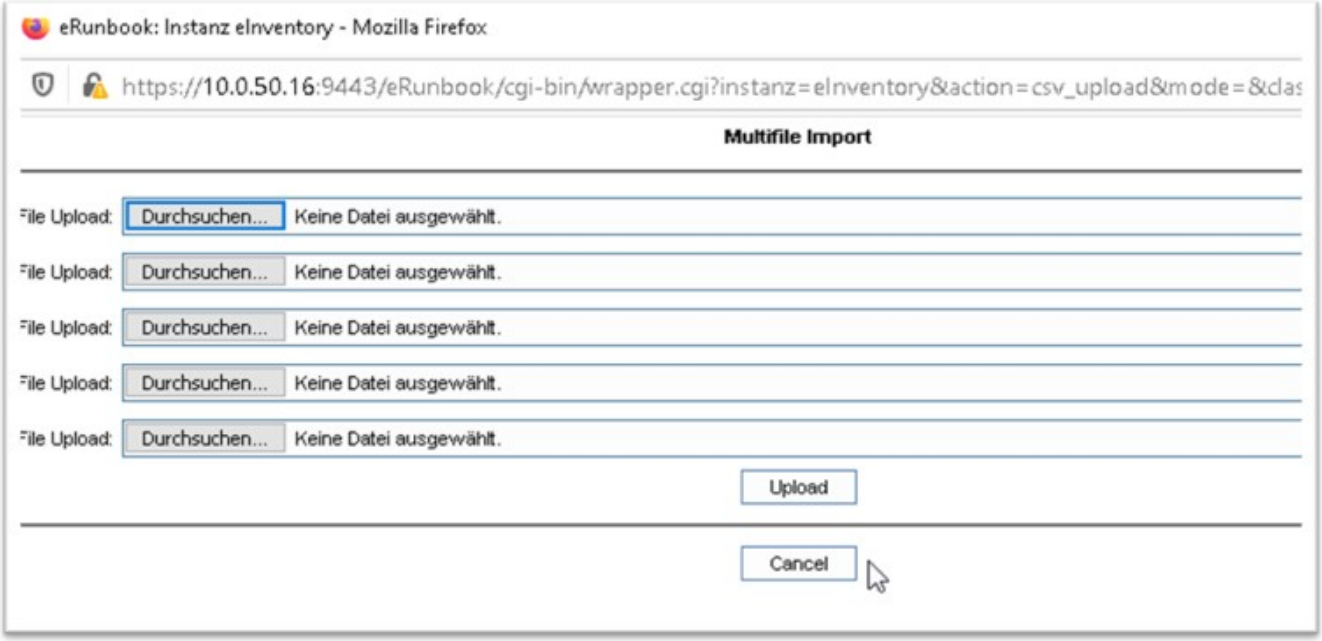
Name	**Description**	**Scan IP**
$at-trib-ute:system:class_system_field_name	$at-trib-ute:system:class_system_field_description	$at-trib-ute:system:class_system_field_scan_ip
jktest	ORA Test	10.0.100.92
jktest2	ORA Test 2	10.0.100.93

Note: The first two lines are the control lines for the correct storage of the data in the appliance. They must NOT be changed!

To get to the file upload and the subsequent import the "Import data" button must be clicked:

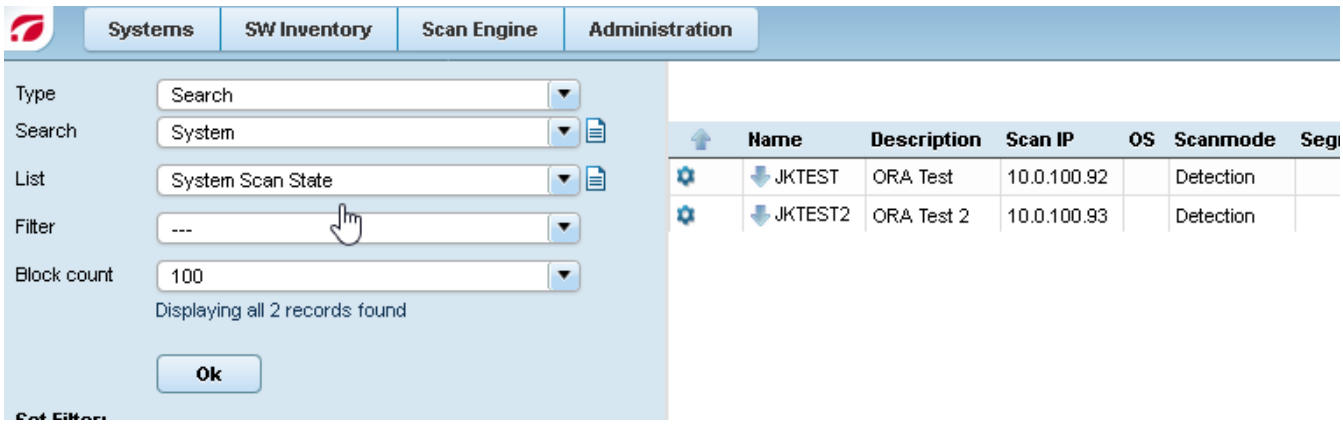


First the location of the import file in the file system must be specified. Up to 5 files can be uploaded at the same time.



Clicking the "Upload" button starts the import. The dialog closes automatically after the import, but can also be closed manually before the process finishes.

After the import, the systems are set up in the appliance:



6.6 Uninstall the agents

If an agent needs to be uninstalled again, different procedures are described here depending on the operating system.

6.6.1 Uninstall on Linux, HPUX, AIX, MacOS

The following commands are to be executed as root user.

Before uninstalling, you can check whether the agent is installed and running as a process. By default, the agent's files are located in "/opt/eRunbook".

The process can be determined with the following command:

```
ps -aef | grep eRunbook_agent
```

When uninstalling, the process must first be stopped with the following command:

```
/etc/init.d/eRunbook stop
```

The eRunbook process should then no longer appear. The following command can be executed again as a check:

```
ps -aef | grep eRunbook_agent
```

The eRunbook file can then be deleted at "/etc/init.d":

```
rm /etc/init.d/eRunbook
```

You can now also delete the agent folder, which is located under "/opt/eRunbook" by default:

```
rm -r /opt/eRunbook
```

The agent is now completely uninstalled.

6.6.2 Uninstall the RPM packages

The following commands are to be executed as root user.

First you should check whether the agent was installed as an RPM package:

```
rpm -qa | grep eRunbook
```

If this is listed here, you can uninstall the RPM package as follows:

```
rpm -e eRunbook-Agent
```

The process ends automatically. However, the eRunbook folder is only moved with a time stamp. This folder must be removed manually.

```
rm -r /opt/eRunbook_<Zeitstempel>
```

The eRunbook start script at "/etc/init.d" must also be deleted afterwards.

```
rm /etc/init.d/eRunbook
```

6.6.3 Uninstall the DEB packages

The following commands are to be executed as root user.

If the agent has been installed as a .DEB package, you can display the exact package name with the following command:

```
dpkg -l | grep erunbook
```

Then this package is uninstalled with the following command:

```
dpkg -r erunbook-agent
```

The process ends automatically and the eRunbook folder is deleted.

6.6.4 Uninstall on Windows

The agent can be removed on Windows using the control panel. The list of installed programs appears via the menu item **System> Programs> Programs and Features**. There you search for **eRunbookAgentStandard** and right-click to initiate the deinstallation.

Compliance with DSGVO/GDPR

What is DSGVO/GDPR and who is affected?

Starting May 25th 2018, the new General Data Protection Regulation (shortened to GDPR or DSGVO) is in effect in all of Europe. This regulation is applied everywhere, where personal data of a person is entered, processed and stored.

The regulation has to be adhered to by the 25th of May 2018. GDPR does not only affect companies within the EU, but also all companies that have business relationships with the EU or that process data of EU-citizens. Also all software products processing personal data are affected.

To ensure the implementation in a timely manner, we have been preparing for this intensively for many months. Special emphasis is put on ensuring that rights of affected persons, data processing, technical and organizational measures, as well as the ability to provide the necessary documentation-, accountability- and declaration duties can be fulfilled correctly.

Up until the 24th of May 2018 we act according to the Bundesdatenschutzgesetz. All Spider products have to fulfill the legal requirements to this point in time. Starting May 25th 2018, we a company, and our products will act according to the new regulation.

7.1 Connectors with personal data

The connectors that export personal data are described in the following chapters.

Starting with the 1.1804 release, the connectors are delivered with a GDPR conforming configuration. In some cases the data is needed for further processing, in other cases the data is for pure comfort and will be marked as such.

7.1.1 API based connectors

VMWare vCenter / ESX Server

This connector does not query personal data.

Columbus Datacenter Inventory

The connector itself does not query any user related data, but the data retrieved from the Datacenter Inventory Appliance can contain such data. The queried data might contain IP Addresses, since the Data Center Inventory is only valid for server OS, these addresses are not retraceable to a single person.

Active Directory

The following personal data is processed when exporting user objects:

AD Attribute	Content	Usage	Exported by default
DistinguishedName	A Distinguished Name represents an object in a hierarchical directory	Detection of a user.	Yes
UserPrincipalName	This attribute contains the UPN that is an Internet-style login name for a user based on the Internet standard RFC 822	Detection of a user.	Yes
EmailAddress	E-Mail-Address of the user	The E-Mail address is used for sending notifications	Yes

AD Attribute	Content	Usage	Exported by default
GivenName	Given name of the user.	First name for display	Yes
Surname	Surname of the user.	Surname for display	Yes
DisplayName	Display Name of the user.	Display Name for display.	Yes
ObjectGUID	Distinct ID uniquely identifying the user in the active directory.	Detection of a user.	Yes
ObjectSid	Distinct SID uniquely identifying the user in the active directory.	Detection of a user.	Yes
UserAccountControl	The Active Directory attribute userAccountControl contains a range of flags which define some important basic properties of a user object	Used to determine if an account is active or not.	Yes
SamAccountType	This attribute contains information about every account type object.	Used to determine type of account	Yes
SamAccountName	In the AD attribute SAMAccountName, the account logon name or the user object is stored		No
telephoneNumber	he AD attribute telephoneNumber can contain the primary telephone number where the user is available at work		No
homePhone	The Active Directory attribute homePhone can contain the private telephone number of the user.		No
mobile	The Active Directory attribute mobile can contain the mobile telephone number of the user.		No
Company	Company name of the user.		No
employeeID	Employee ID of the user.		No
Department	The Active Directory attribute department can be used to store the department name or team label for the regarding user account.		No
physicalDeliveryOfficeName	The Active Directory attribute physicalDeliveryOfficeName is for storing a description for the office, for example the office building/number.		No
Title	Contains the job title of the user.		No
co	Country of the user.		No
StreetAddress	Street address of the user		No
l	City of the user.		No
st	State of the user.		No
PostalCode	Postal code of the user.		No
facsimileTelephoneNumber	Facsimile number of the user.		No
c	County of the user.		No
cn	Common name of the user.		No

Microsoft Azure

The following personal data is processed when exporting user objects:

AD Attribute	Content	Usage	Exported by default
OnPremisesSecurityIdentifier	SID that is synchronized from the local active directory.	Detection of a user.	Yes
UserPrincipalName	This attribute contains the UPN that is an Internet-style login name for a user based on the Internet standard RFC 822	Detection of a user.	Yes
Mail	E-Mail-Address of the user	Detection of a user.	Yes
GivenName	Given name of the user.	Given name for display.	Yes
Surname	Surname of the user.	Surname for display.	Yes
DisplayName	Display name of the user.	Display name for display.	Yes
AccountEnabled	Information if the account is enabled.		Yes
CompanyName	Company name of the user.		No
Country	Country of the user.		No
CreationType	Indicates whether the user account is a local account for an Azure Active Directory B2C tenant.		No
DeletionTimestamp	Timestamp when the object was deleted.		No
DirSyncEnabled	Setting if the account is synchronized from a local active directory.		No
TelephoneNumber	Phone Number		No
ImmutableId	Id of the user		No
IsCompromised	Flag if the account is deemed a security risk.		No
LastDirSyncTime	Last synch time		No
MailNickName	MailNickName of the user		No
Mobile	Can contain the users mobile number.		No
Department	Department of the user.		No
ObjectType	Type of the Azure AD Object.		No
PasswordPolicies	Password policies assigned to the user.		No
PhysicalDeliveryOfficeName	The Active Directory attribute physicalDeliveryOfficeName is for storing a description for the office, for example the office building/number.		No

AD Attribute	Content	Usage	Exported by default
JobTitle	Contains the job title of the user.		No
PreferredLanguage	Preferred language of the user.		No
RefreshTokensValid-FromDateTime	Token Refresh validation time.		No
ShowInAddressList	Flag if the user is shown in public address list.		No
StreetAddress	Street address of the user		No
City	City		No
State	State		No
PostalCode	Postal code		No
FacsimileTele-phoneNumber	Facsimile Number		No
UsageLocation	Usage location of the user.		No
UserType	User type		No

The following personal data is processed when exporting tenant objects:

Tenant Attribute	Content	Usage	Exported by de-fault
DisplayName	Display Name of the tenant.	For Display	Yes
ObjectId	Unique ID of the tenant		Yes
DirSyncEnabled	Setting if the account is synchronized from a local active directory.		Yes
PreferredLanguage	Default language of the tenant.		Yes
ObjectType	Object type of the Tenant		No
PostalCode	Postal Code		No
CountryLetterCode	Country Code		No
City	City		No
State	State		No
Country	Country		No

Tenant Attribute	Content	Usage	Exported by default
TelephoneNumber	Phone number		No
Street	Street address		No

Adobe Online

The following personal data is processed when exporting user objects:

Adobe Attribute	Content	Usage	Exported by default
email	E-Mail address of the user	Detection of a user.	Yes
username	User name	Detection of a user.	Yes
firstName	Given name of the user.	Eventually needed for manual assignment of user	Yes
lastName	Surname of the user.	Eventually needed for manual assignment of user	Yes
status	Status of the user		Yes
domain	Domain of the user.		Yes
countryCode	Country of the user.		Yes
userType	Type of the user.		Yes

*The Adobe connector cannot be configured to export attributes or not.

Microsoft Application Virtualization (App-V)

The following personal data is processed when exporting Application Usage:

App-V Attribute	Content	Usage
UserName	Logon name of the user executing a package	Detection of a user.

7.1.2 Database based connectors

The database connectors all deliver the same set of parameters, those that might contain personal data are listed here:

Field	Content	Usage
Hostname	Unique name of the machine, depending on company policy this might contain the username.	Identification of the computer.
MAC1 ... MAC4	MAC Addresses of the machine.	No technical usage, export is disabled by default starting with release 1.1805.
IPAddressV4	IP v4 address of the machine	No technical usage, export is disabled by default starting with release 1.1805.
IPAddressV6	IP v6 address of the machine	No technical usage, export is disabled by default starting with release 1.1805.

Field	Content	Usage
LastLoggedOnUser	Name of the last logged on user.	Detection of a user.

* You can find details for en-/disabling the export of personal in the chapter: Suppress export of MACC and IP Information (on page 42).

7.2 Inventory Components

7.2.1 Windows

Starting with version 7.5.5.17 all fields with personal data will not be exported by default anymore..

Inventory	Fields
HardwareScan.csv	LastLoggedOnUser LastLoggedOnSAMUser LastLoggedOnUserSID MAC1 MAC2 MAC3 MAC4 IPAddressV4 IPAddressV6
InventoryItems.csv	OS.System.RegisteredUser OS.System.Organization OS.System.ProductKey

If needed the export of the data can be enabled by changing the configuration for the Inventory Agent (on page 96) and the Inventory Scanner (on page 108).

7.3 File locations containing personal data

The data exported and processed by the connectors are put into the folder(s) specified during the setup of the SDC.

The exact locations can be queried from the Configuration file (on page 23) of the Data Collector (SpiderDataCollector.cfg).

Section	Variable	Usage
OTBServer	DataDirectory	Data created by the inventory components.
General	DataDirectory	Data ready for the upload to the Recognition.

7.4 Secure data transport

The data transmission can be secured with a SSL certificate..

Security is always provided and managed by the parent system, e.g. for the inventory components the Data Collector will provide the settings and certificate.

SSL secured Transmission (on page 114)

Security for the transmission from Data Collector to Recognition is provided by the Recognition, details are described in the technical reference.

FAQ

8.1 TCP/IP Socket based communication (OTB)

brainwaregroup uses a proprietary, TCP/IP based, communication backbone named Object Transfer Bus (OTB) which is used for most of its server/client communication.

Some of the key points about the brainwaregroup implementation of OTB include:

- Configurable, single port based transmission
- Data bandwidth and volume limitations on both the client and server side
- Customer specific encryption
- Block Compression
- Block based streaming

As a result we have a single port based communication ability throughout all our software platforms with the ability to be easy administered in a network. brainwaregroup can manage absolute bandwidth of the service independent of the number of nodes, the volume and type of data. Brainwaregroup also supports software controlled checkpoint restart and transmission retry and can be adapted to fit specific encryption requirements for government, military and financial institutions.

8.2 Log file locations

Where support from brainwaregroup is required, please include all of the log files mentioned below.

Program	Executable	Log file	Location(s)
Data Collector	SpiderDataCollector.exe	Brainware.log Brainware_0.log Brainware_1.log Brainware_2.log Brainware_3.log Brainware_4.log	%windir% %ProgramData%\Columbus
Inventory Scanner	ColumbusInventoryScanner.exe		
Inventory Agent	ColumbusInventoryAgent.exe		
DSDC	DSDC.exe	DSDC-<Timestamp>.sil	%ProgramData%\brainwaregroup\DSDC\Logs
Powershell Scripts	<PowerShellScriptName>.ps1	<PowerShellScript-Name><Timestamp>.sil	%ProgramData%\brainwaregroup\<ConnectorName>\Logs

Note The brainware.log file will frequently roll over and create Brainware_0.log, Brainware_1.log and so on.

Note	The *.sil log files are not in a human readable format. In the event this needs to be read, the file should be sent to brainwaregroup support in order to get it evaluated.

8.3 Data Flow

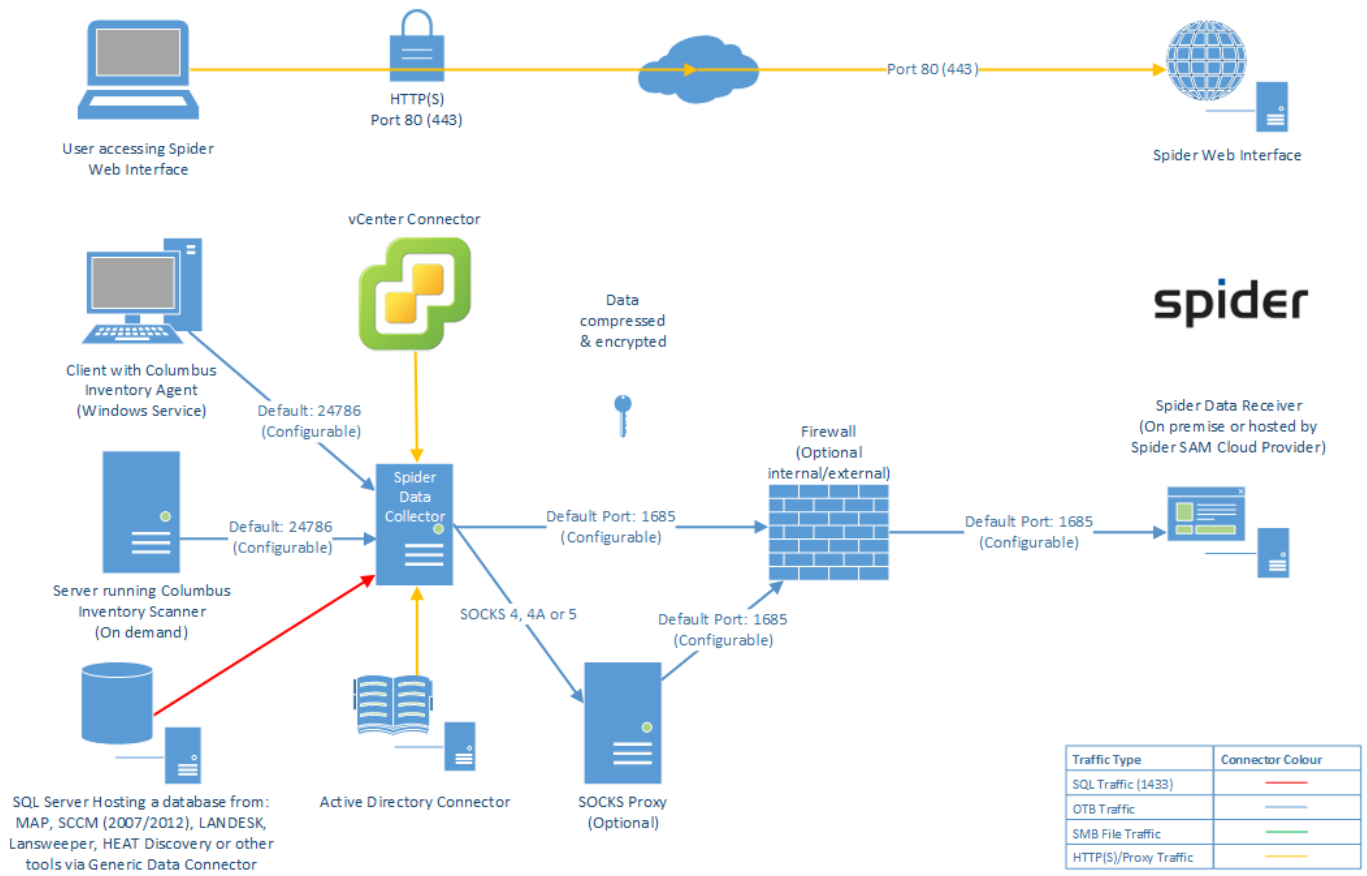


Figure - Data Flow

Appendix

9.1 Powershell Module - bwgTools

Note:	The bwgTools module is introduced with release 1712.2, it was former known as bwgLogging

Logging and some commonly used methods for the PowerShell scripts is realized by placing a module in the paths recommended by Microsoft.

These paths are

for x86: %ProgramFiles(x86)%\WindowsPowerShell\Modules

for x64: %ProgramFiles%\WindowsPowerShell\Modules

Since the Data Collector service is a 32bit executable, it will use the x86 PowerShell installation, if the scripts are executed by hand (e.g. for testing purposes) this will usually be carried out in the x64 environment of PowerShell, therefore the logging module is placed in both paths named above.

Even though the above paths are recommended by Microsoft they only become generally available in the PowerShell environments when PowerShell v4 is installed.

In case PowerShell v3 is used, the paths for the modules have to be added to the PowerShell environment variable "PSModulePath".

The contents of this variable can be queried by issuing:

```
$env:PSModulePath
```

in the Powershell console. It will then list the paths in which Powershell will look for modules, if the above named paths are not part of the variable you can add the path by issuing:

```
$p = [Environment]::GetEnvironmentVariable("PSModulePath", "Machine")
$newPSModulePath = Join-Path $env:ProgramFiles "WindowsPowerShell\Modules"
$p
$newPSModulePath

if($p -match  [regex]::Escape($newPSModulePath))
{
   Write-Host "Found" -ForegroundColor Green
}
else
{
   Write-Host "Not Found" -ForegroundColor Red
   $p += ";$($newPSModulePath)"
   $p
   #[Environment]::SetEnvironmentVariable("PSModulePath",$p, "Machine")
}
```

Attention	Please note that you have to issue the above command in BOTH x86 and x64 Windows PowerShell consoles otherwise one of them might not find the logging module. The Powershell consoles are found in: x86: %windir%\SysWOW64\WindowsPowerShell\v1.0\Powershell.exe x64: %windir%\System32\WindowsPowerShell\v1.0\Powershell.exe

9.2 Generic Connector Stored Procedures

A set of templates can be downloaded from: https://docs.flexera.com/Spider64/GenericDataConnectorTemplates.zip

Attention About SIDs

Many of the following Stored Procedures utilize so called SIDs (Security Identifiers) for the identification of Accounts, Groups, Group Memberships and so on.

In order for all aspects to work correctly it is important that a well formed SID is used when exporting data, an example SID will look like this: S-1-5-21-1004336348-1177238915-682003330-512, the blue part is the domain identifier, the green part is the relative identifier. If you want to anonymize your data but still need to use all features just change the blue part (keep it the same throughout all SIDs) and just increment the green part by one for each object used.

Make sure that all Objects have the same relative identifier they had for the original export, otherwise the results are unpredictable. So if you export Jon Doe with the SID S-1-5-21-1111111111-2222222222-333333333-1000 make sure he gets the same SID with the next export too, the same goes for groups and all other objects utilizing an SID.

For details please see: https://technet.microsoft.com/en-us/library/cc778824(v=ws.10).aspx

9.2.1 dbo.swrGetWorkList

This stored procedure is expected to return a list of devices, the following procedures will be called one by one for each returned item of this stored procedure.

Columns	Data type	Description	Mandatory
Identifier	Variable	Depending on the resulting data the format of this column may vary, it is important that the parameter @identifier of the other stored procedures is of the same type as returned from this stored procedure.	Yes
UUID	uniqueidentifier / GUID	GUID uniquely identifying the machine	No
Urn	nvarchar(100)	Urn of the machine, set to NULL since it is not used yet.	No
Domain	nvarchar(100)	DNS Domain name of the device	No
DomainNameNetBIOS	nvarchar(100)	NetBIOS Domain Name of the machine.	No
Hostname	nvarchar(100)	Hostname of the device	Yes

A device is identified in the following order:

1. UUID
2. URN
3. DomainName + Hostname
4. DomainNameNetBIOS + Hostname

9.2.2 dbo.swrGetHardwareScan

This procedure returns the hardware data of a given device

Parameters: @identifier (format as defined in the results of dbo.swrGetWorkList)

Column	Data type	Description	Mandatory
ScanDate	datetime	Date and time of the last scan of the device.	Yes
Manufacturer	nvarchar(256)	Manufacturer of the device	Yes
Model	nvarchar(100)	Device model	Yes
MAC1	nvarchar(100)	MAC Address of the Network Adapter	No
MAC2	nvarchar(100)	Additional MAC Address	No
MAC3	nvarchar(100)	Additional MAC Address	No
MAC4	nvarchar(100)	Additional MAC Address	No
Serial	nvarchar(100)	Serial Number of the device	No
DeviceChassis	nvarchar(100)	Required for device type recognition. DeviceChassis and ChassisType mutually exclude each other. Values please see table below.	No
ChassisType	int	Required for device type recognition. DeviceChassis and ChassisType mutually exclude each other. Values please see table below.	No
ProcessorManufacturer	nvarchar(100)	Manufacturer of the CPU	No
ProcessorType	nvarchar(256)	Detailed name of the CPU	No
ProcessorSpeed	int	CPU Speed in MHz	No
CPUCount	int	Total amount of physical CPUs in device	No
CPUCoreCount	int	Total amount of all physical CPU cores in system	No
CorePerCPU	int	Count of Cores per single physical CPU	No
CPULogicalCount	int	Total amount of logical CPUs in system	No
DiskTotalMB	int	Total hard disk space of machine (over all physical discs)	No
DiskFreeMB	int	Total free hard disk space of machine (over all physical discs)	No
GraphicAdapter	nvarchar(100)	Name of graphic adapter	No
GraphicMemoryMB	int	Amount of graphic adapter memory	No
MemoryMB	int	Amount of machine memory	No
IPAddressV4	nvarchar(15)	IP Address v4	No
IPAddressV6	nvarchar(50)	IP Address v6	No

Column	Data type	Description	Mandatory
CPUArchitecture	nvarchar(100)	CPU Architecture, e.g. amd64, x86, Itanium	No
OSCaption	nvarchar(100)	Name of Operating System	No
LastLoggedOnUser	nvarchar(100)	Username of the last user logged on to the machine in format Domain\user.name	No
BIOSVendor	nvarchar(100)	Manufacturer of the computer BIOS	No
BIOSVersion	nvarchar(100)	Version of the computer BIOS	No
BIOSDate	datetime	Date of the Computer BIOS	No
InventorySource	nvarchar(100)	Identifier of the DC (type and version)	No
Class	nvarchar(100)	Asset Type, this will override any automatic detection of the device chassis, possible values are: Cluster Desktop Mobile Device Laptop Unknown Printer Router Server Switch Tablet Thin Client Virtual Client Virtual Server Network Device	No
LegalEntity	nvarchar(500)	Legal Entity Path	No
Parameter01	nvarchar(100)	Additional Parameter 1	No
Parameter02	nvarchar(100)	Additional Parameter 2	No
Parameter03	nvarchar(100)	Additional Parameter 3	No
Parameter04	nvarchar(100)	Additional Parameter 4	No
Parameter05	nvarchar(100)	Additional Parameter 5	No

Values for DeviceChassis and ChassisType

ChassisType	DeviceChassis	ChassisType	DeviceChassis
1	Other	15	Space-Saving
2	Unknown	16	Lunch Box
3	Desktop	17	Main System Chassis
4	Low Profile Desktop	18	Expansion Chassis

ChassisType	DeviceChassis	ChassisType	DeviceChassis
5	Pizza Box	19	Sub Chassis
6	Mini Tower	20	Bus Expansion Chassis
7	Tower	21	Peripheral Chassis
8	Portable	22	Storage Chassis
9	Laptop	23	Rack Mount Chassis
10	Notebook	24	Sealed-Case PC
11	Hand Held	99	Virtual
12	Docking Station	100	Thin Client
13	All in One	105	Mobile Device
14	Sub Notebook	110	AzureVM

9.2.3 dbo.swrGetFileScan

This procedure returns the scanned files of a given device

Parameters: @identifier (format as defined in the results of dbo.swrGetWorkList)

Column	Data type	Description	Mandatory
Manufacturer	nvarchar(256)	Manufacturer of the file	Yes
ProductName	nvarchar(256)	Product Name queried from file	Yes
ProductVersion	nvarchar(256)	Product Version queried from file	Yes
FileName	nvarchar(256)	Executable name	Yes
FileDescription	nvarchar(256)	Description of the file	Yes
FileVersion	nvarchar(256)	Version of the file	No
FileSize	bigint	Size of the file	Yes
FilePath	nvarchar(512)	Path where file was found	Yes

9.2.4 dbo.swrGetSoftwareScan

This procedure returns the installed programs (uninstall information) of a given device

Parameters: @identifier (format as defined in the results of dbo.swrGetWorkList)

Column	Data type	Description	Mandatory
Manufacturer	nvarchar(256)	Manufacturer of the installed program	Yes

Column	Data type	Description	Mandatory
SoftwareName	nvarchar(256)	Name of the installed program	Yes
SoftwareVersion	nvarchar(256)	Version of the installed program	Yes
LicenceRequirement	decimal(18,4)	Required License amount	No
SerialNo	nvarchar(100)	Software Serial Number	No
InstallDate	date	Date of installation	No

Operating Systems

Since the operating system is not contained in the Add/Remove Programs list, this information needs to be added to the output also. A list of Operating Systems is given in the following table.

SoftwareName	SoftwareVersion	Manufacturer
Microsoft Windows 2000 Advanced Server	5.0	Microsoft Corporation
Microsoft Windows 2000 Professional	5.0	Microsoft Corporation
Microsoft Windows 2000 Professionnel	5.0	Microsoft Corporation
Microsoft Windows 2000 Server	5.0	Microsoft Corporation
Microsoft Windows XP Professional	5.1	Microsoft Corporation
Microsoft Windows XP Professionnel	5.1	Microsoft Corporation
Microsoft Windows Server 2003	5.2	Microsoft Corporation
Microsoft Windows 7 Enterprise	6.1	Microsoft Corporation
Microsoft Windows 7 Enterprise K	6.1	Microsoft Corporation
Microsoft Windows 7 Enterprise N	6.1	Microsoft Corporation
Microsoft Windows 7 Professional	6.1	Microsoft Corporation
Microsoft Windows 7 Professional N	6.1	Microsoft Corporation
Microsoft Windows 7 Professionnel	6.1	Microsoft Corporation
Microsoft Windows 7 Ultimate	6.1	Microsoft Corporation
Microsoft Windows Server 2008 R2 Datacenter	6.1	Microsoft Corporation
Microsoft Windows Server 2008 R2 Enterprise	6.1	Microsoft Corporation
Microsoft Windows Server 2008 R2 Foundation	6.1	Microsoft Corporation
Microsoft Windows Server 2008 R2 Standard	6.1	Microsoft Corporation
Microsoft Windows 8 Enterprise	6.2	Microsoft Corporation
Microsoft Windows 8 Pro	6.2	Microsoft Corporation

SoftwareName	SoftwareVersion	Manufacturer
Microsoft Windows Server 2012 Datacenter	6.2	Microsoft Corporation
Microsoft Windows Server 2012 Standard	6.2	Microsoft Corporation
Microsoft Windows 8.1 Enterprise	6.3	Microsoft Corporation
Microsoft Windows 8.1 Pro	6.3	Microsoft Corporation
Microsoft Windows Server 2012 R2 Standard	6.3	Microsoft Corporation
Microsoft Windows 10 Enterprise	10.0	Microsoft Corporation
Microsoft Windows 10 Enterprise 2015 LTSB	10.0	Microsoft Corporation
Microsoft Windows 10 Enterprise Edition	10.0	Microsoft Corporation
Microsoft Windows 10 Home	10.0	Microsoft Corporation
Microsoft Windows 10 Home K	10.0	Microsoft Corporation
Microsoft Windows 10 Professional	10.0	Microsoft Corporation
Microsoft Windows 10 Professionnel	10.0	Microsoft Corporation
Microsoft Windows 10 Pro	10.0	Microsoft Corporation
Microsoft Windows 10 Pro N	10.0	Microsoft Corporation
Microsoft Windows Server 2016 Datacenter	10.0	Microsoft Corporation
Microsoft Windows Server 2016 Datacenter Edition	10.0	Microsoft Corporation
Microsoft Windows Server 2016 Standard	10.0	Microsoft Corporation
Microsoft Windows Server 2016 Standard Edition	10.0	Microsoft Corporation

SQL Server Edition Detection

To help that SQL Server editions are handled better, you can create special Software Scan entries, these have to adhere to the following rules.

Column	Value	Description
Manufacturer	Microsoft Corporation	This value is fixed and cannot be named otherwise.
SoftwareName	*Microsoft SQL Server <XXXX> <YYYY>	The Software name has(!) to start with an asterisk, <XXXX> is the 4 digit Year Number of the SQL Server and <YYYY> is the Edition.
SoftwareVersion	<MajorVersion>.<MinorVersion>	Major and Minor version of the SQL server, separated by a dot and without leading zeros.

Examples

SoftwareName	Version

SoftwareName	Version
*Microsoft SQL Server 2005 Express Edition	9.3
*Microsoft SQL Server 2008 Developer Edition	10.3
*Microsoft SQL Server 2008 Developer Edition	10.4
*Microsoft SQL Server 2008 Express Edition	10.3
*Microsoft SQL Server 2008 R2 Developer Edition (64-bit)	10.51
*Microsoft SQL Server 2012 Developer Edition	11.1
*Microsoft SQL Server 2012 Developer Edition	11.3
*Microsoft SQL Server 2012 Developer Edition (64-bit)	11.1
*Microsoft SQL Server 2012 Enterprise Edition	11.1
*Microsoft SQL Server 2012 Enterprise Edition: Core-based Licensing	11.0
*Microsoft SQL Server 2012 Enterprise Edition: Core-based Licensing (64-bit)	11.0
*Microsoft SQL Server 2012 Express Edition	11.0
*Microsoft SQL Server 2012 Express Edition	11.3
*Microsoft SQL Server 2012 Express Edition (64-bit)	11.0
*Microsoft SQL Server 2014 Developer Edition	12.0
*Microsoft SQL Server 2014 Developer Edition	12.2
*Microsoft SQL Server 2014 Express Edition	12.0
*Microsoft SQL Server 2014 Express Edition	12.2
*Microsoft SQL Server 2014 Standard Edition	12.0
*Microsoft SQL Server 2014 Standard Edition	12.1
*Microsoft SQL Server 2014 Standard Edition	12.2
*Microsoft SQL Server 2016 Developer Edition	13.0
*Microsoft SQL Server 2016 Enterprise Edition: Core-based Licensing	13.0
*Microsoft SQL Server 2016 Express Edition	13.0
*Microsoft SQL Server 2016 Express Edition	13.1

9.2.5 dbo.swrGetDeviceRelationship

This procedure returns the guest host relationships between devices.

Column	Data type	Description	Mandatory

Column	Data type	Description	Mandatory
ChildDeviceUUID	uniqueidentifier	UUID of the child device	No
ChildDeviceUrn	nvarchar(100)	URN of the child device	No
ChildDeviceDomainName	nvarchar(100)	Domain name of the child device	No
ChildDeviceHostName	nvarchar(100)	Hostname of the child device	No
ChildDeviceDomainNetBIOS	nvarchar(100)	NetBIOS domain name of the child device	No
ParentDeviceUUID	uniqueidentifier	UUID of the parent device	No
ParentDeviceUrn	nvarchar(100)	URN of the parent device	No
ParentDeviceDomainName	nvarchar(100)	Domain name of the parent device	No
ParentDeviceHostName	nvarchar(100)	HostName of the parent device	No
ParentDeviceDomainNetBIOS	nvarchar(100)	NetBIOS domain name of the parent device	No
DeviceRelationshipTypeID	int	Relationship type: 1: Guest-Host 2: Host-Cluster	No
ScanDate	datetime	Scan date	Yes

9.2.6 dbo.swrGetADUserObject

This procedure returns user objects.

Column	Data type	Mandatory
ObjectGUID	uniqueidentifier	One of these columns is needed for identification purposes.
ObjectSid	nvarchar(184)	
DistinguishedName	nvarchar(255)	
UserPrincipalName	nvarchar(1024)	
EmailAddress	nvarchar(254)	No
SamAccountName	nvarchar(20)	No (but recommended)
NetbiosDomainName	nvarchar(16)	No (but recommended)
Firstname	nvarchar(100)	No
Lastname	nvarchar(100)	No
DisplayName	nvarchar(256)	No
PhoneNo	nvarchar(100)	No
PrivatePhoneNo	nvarchar(100)	No

Column	Data type	Mandatory
MobilePhoneNo	nvarchar(100)	No
Company	nvarchar(64)	No
StaffNo	nvarchar(100)	No
Department	nvarchar(100)	No
PhysicalDeliveryOfficeName	nvarchar(128)	No
JobTitle	nvarchar(100)	No
Country	nvarchar(100)	No
StreetAddress	nvarchar(1024)	No
Location	nvarchar(100)	No
State	nvarchar(128)	No
PostalCode	nvarchar(100)	No
FaxNo	nvarchar(100)	No
CountryCode	nvarchar(2)	No
Name	nvarchar(255)	No
UserAccountControl	int	No
SamAccountType	int	No

Attention For matching to work correctly it is necessary that SamAccountName AND NetbiosDomainName are specified!

9.2.7 dbo.swrGetADGroupObject

This procedure returns group objects.

Column	Data type	Mandatory
ObjectGUID	uniqueidentifier	Yes
ObjectSid	nvarchar(184)	Yes
DistinguishedName	nvarchar(256)	Yes
Name	nvarchar(256)	Yes
SamAccountName	nvarchar(256)	Yes

Attention This procedure only delivers correct data if dbo.swrGetADGroupMember is also implemented.

9.2.8 dbo.swrGetADGroupMember

This procedure returns group member objects.

Column	Data type	Mandatory
GroupObjectSID	nvarchar(184)	Yes
GroupObjectGUID	uniqueidentifier	Yes
GroupDistinguishedName	nvarchar(256)	Yes
MemberObjectGUID	uniqueidentifier	Yes
MemberObjectSID	nvarchar(184)	Yes
MemberName	nvarchar(256)	Yes
MemberDistinguishedName	nvarchar(256)	Yes
MemberSamAccountName	nvarchar(256)	Yes
MemberObjectClass	nvarchar(256)	Yes, either "user" or "group"

Attention This procedure only delivers correct data if dbo.swrGetADGroup is also implemented.

Also the connected user accounts need to be exported using the AD User export.

9.2.9 dbo.swrGetSwidScan

This procedure returns SWID tag information.

Column	Data type	Mandatory
SWCreatorName	nvarchar(256)	Yes
SWCreatorRegID	uniqueidentifier	Yes
Product_title	nvarchar(256)	Yes
Product_version	nvarchar(256)	Yes
SWLicensorName	nvarchar(256)	Yes
SWLicensorRegID	nvarchar(256)	No
SoftwareUnique	nvarchar(256)	Yes
SoftwareRegID	nvarchar(256)	No
TAGCreatorName	nvarchar(256)	No
TAGCreatorRegID	nvarchar(256)	No
LicenseActivation	nvarchar(256)	No
LicenseChannel	nvarchar(256)	Yes

Column	Data type	Mandatory
LicenseCustomer	nvarchar(256)	No
SerialNumber	nvarchar(256)	Yes

.

9.3 Inventory using MAP Toolkit

For those that are unfamiliar, Microsoft Application and Planning (MAP) is a tool which provides inventory, assessment, and reporting that will help you assess your current IT infrastructure status and determine the right Microsoft technologies for your IT needs and environment. It can be a very valuable tool as it is agentless, and has the ability discover machines on your network that may be unknown.

The information gathered by MAP can also be used by the Data Collector.

Resources:

Product page: http://technet.microsoft.com/en-us/solutionaccelerators/dd537566

Download: http://www.microsoft.com/en-us/download/details.aspx?&id=7826

System requirements: https://www.microsoft.com/en-us/download/details.aspx?&id=7826

9.3.1 Database

By default, the MAP Toolkit will install SQL Server 2012 Express LocalDB during setup. You may also use an existing installation of SQL Server 2008, SQL Server 2008 R2, or SQL Server 2012 if you create an instance named "MAPS" before running the MAP Toolkit installer.

Note	The MAP Toolkit requires the collation order of the database engine to be set to "SQL_Latin1_General_CP1_CI_AS".

It is possible to use an existing SQL Database, this is done by installing SQL Server Express Edition Instance on the machine and configuring it with the following settings during setup:

Instance Name: MAPS

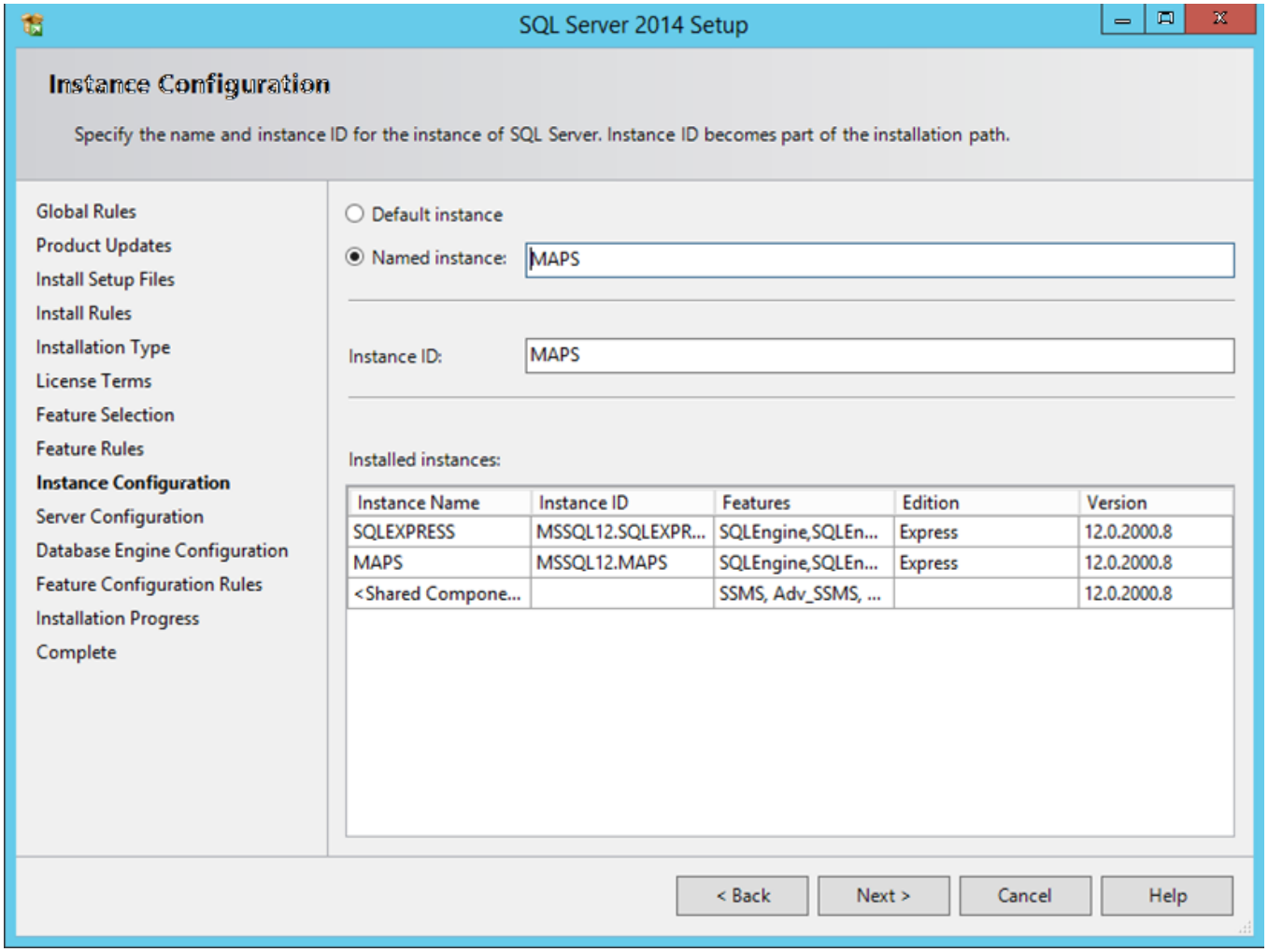


Figure - Instance Name

SQL Server Collation: SQL_Latin1_General_CP1_CI_AS

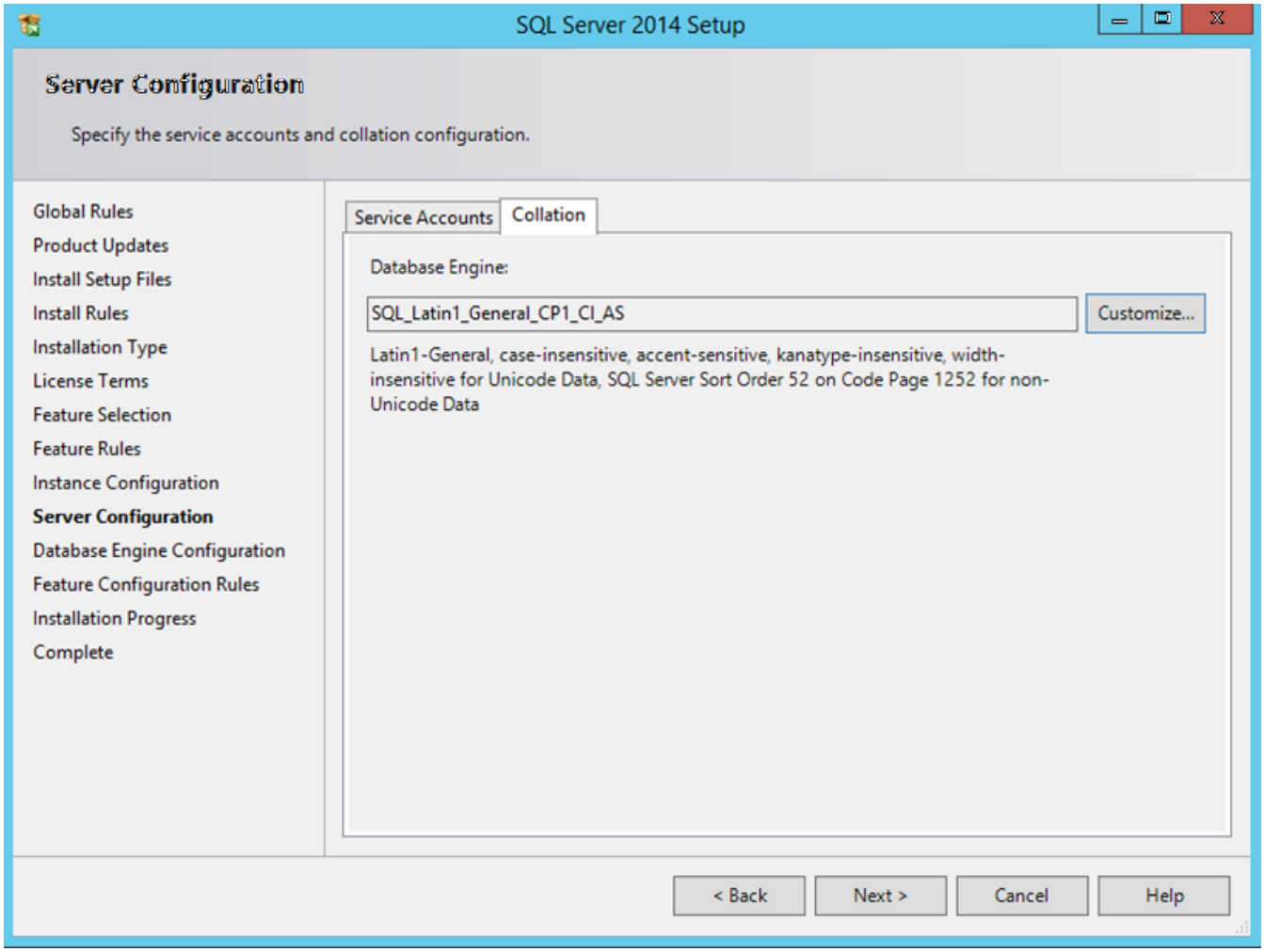
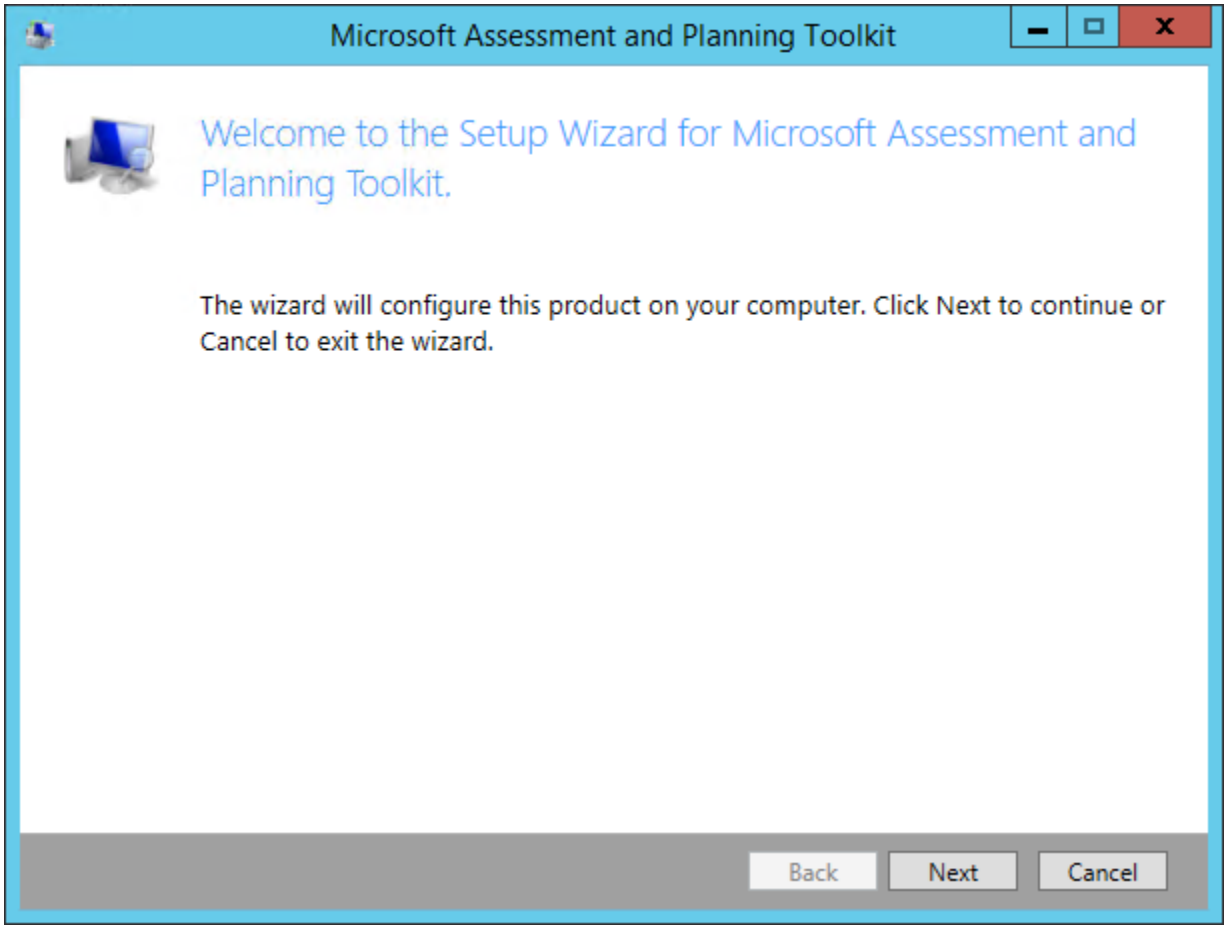


Figure - SQL Server Collation

Installing a SQL Server using the aforementioned settings before executing the MAP setup will then enable MAP to use a fully configurable version of SQL Server that runs as a Windows service when the machine starts and can easily be configured to allow multiple users to access it.
This provides a number of benefits when using the Data Collector to collect MAP data as the ease of configuring the system does not require having to perform any queries for the named pipes of the databases on the LocalDB server, nor configure the Data Collector to trigger a server start when it needs to access the database.

9.3.2 Installation

Run the "MapSetup.exe" program.



1. To get started, click Next.
2. The first step is a pre-req checker. If any are not met you must correct these before continuing.
3. Accept the license agreement and click Next.
4. Accept or change the installation path and click Next.
5. Select a choice for the Customer Experience Improvement Program and click Next.
6. Click Install.

When completed, click Finish to open the MAP toolkit.

9.3.3 Configuration

When the MAP toolkit opens for the first time – you must create a database to store our collected inventory. Give the DB a name, such as "MAPDB" and click OK to create the DB.

Using LocalDB

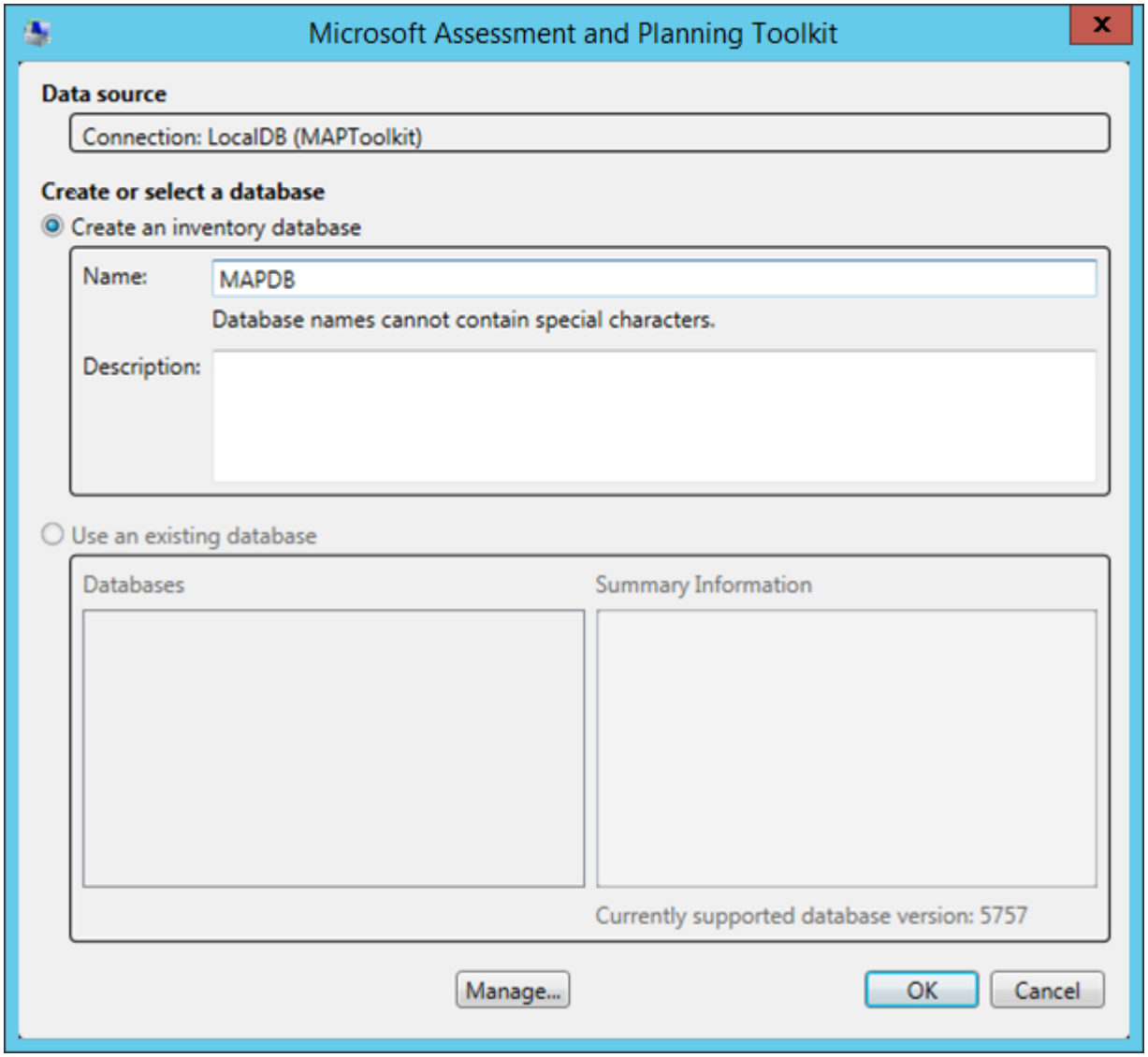


Figure - LocalDB

Alternative: Using MAPS instance

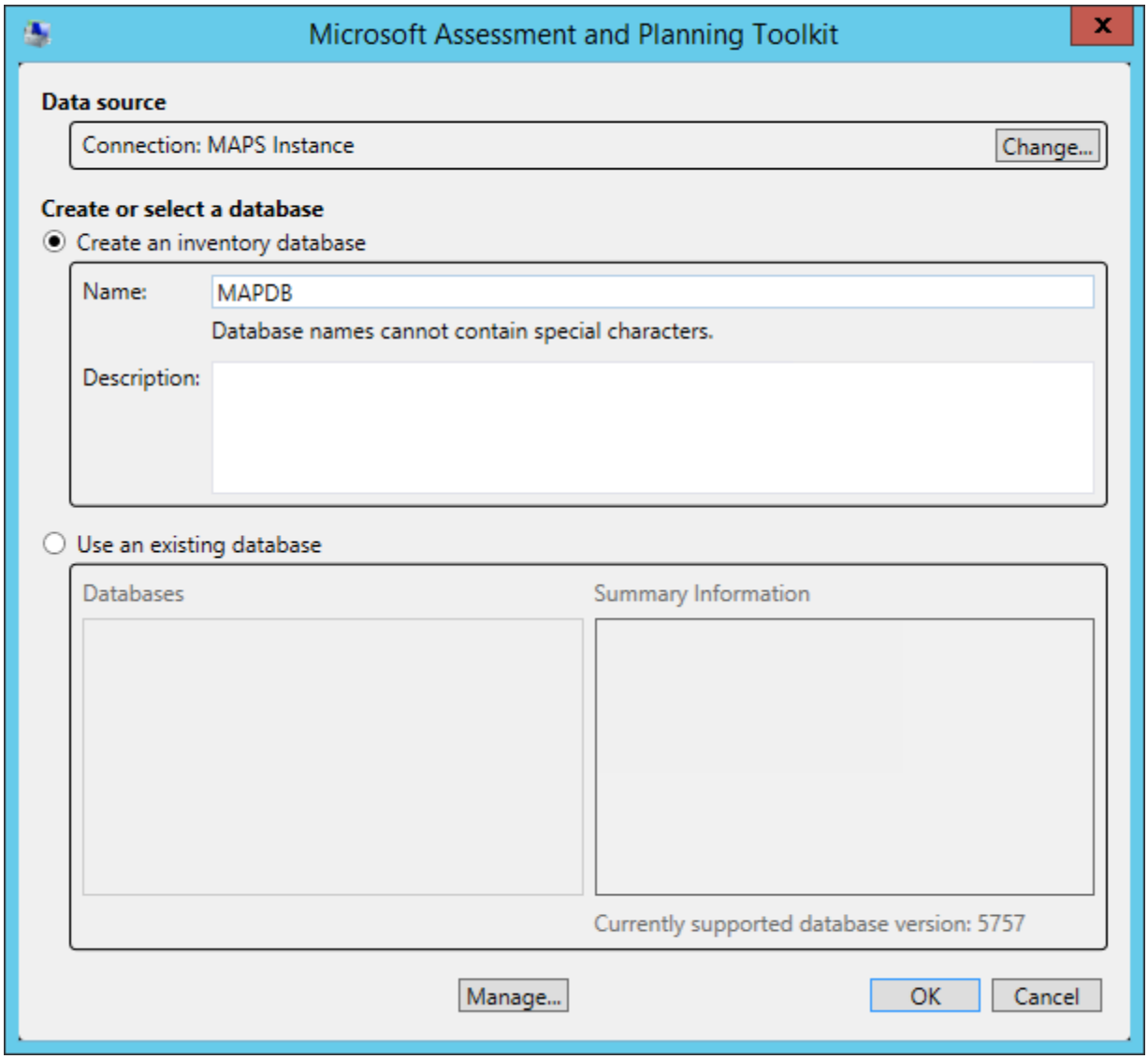


Figure - MAPS instance

9.3.4 Collecting inventory data

Requirements to gather data:

MAP uses WMI to gather the inventory data. You need to ensure that the server/workstation that is running the MAP collection has access to all servers via any hardware firewalls, and if the servers are running Windows Firewall that exceptions allow the MAP workstation to contact all servers on those ports. Detailed information is available at: http://social.technet.microsoft.com/wiki/contents/articles/8657.map-prepare-the-environment-wmi.aspx

In the left pane – click "Environment" and then select "Collect Inventory Data".

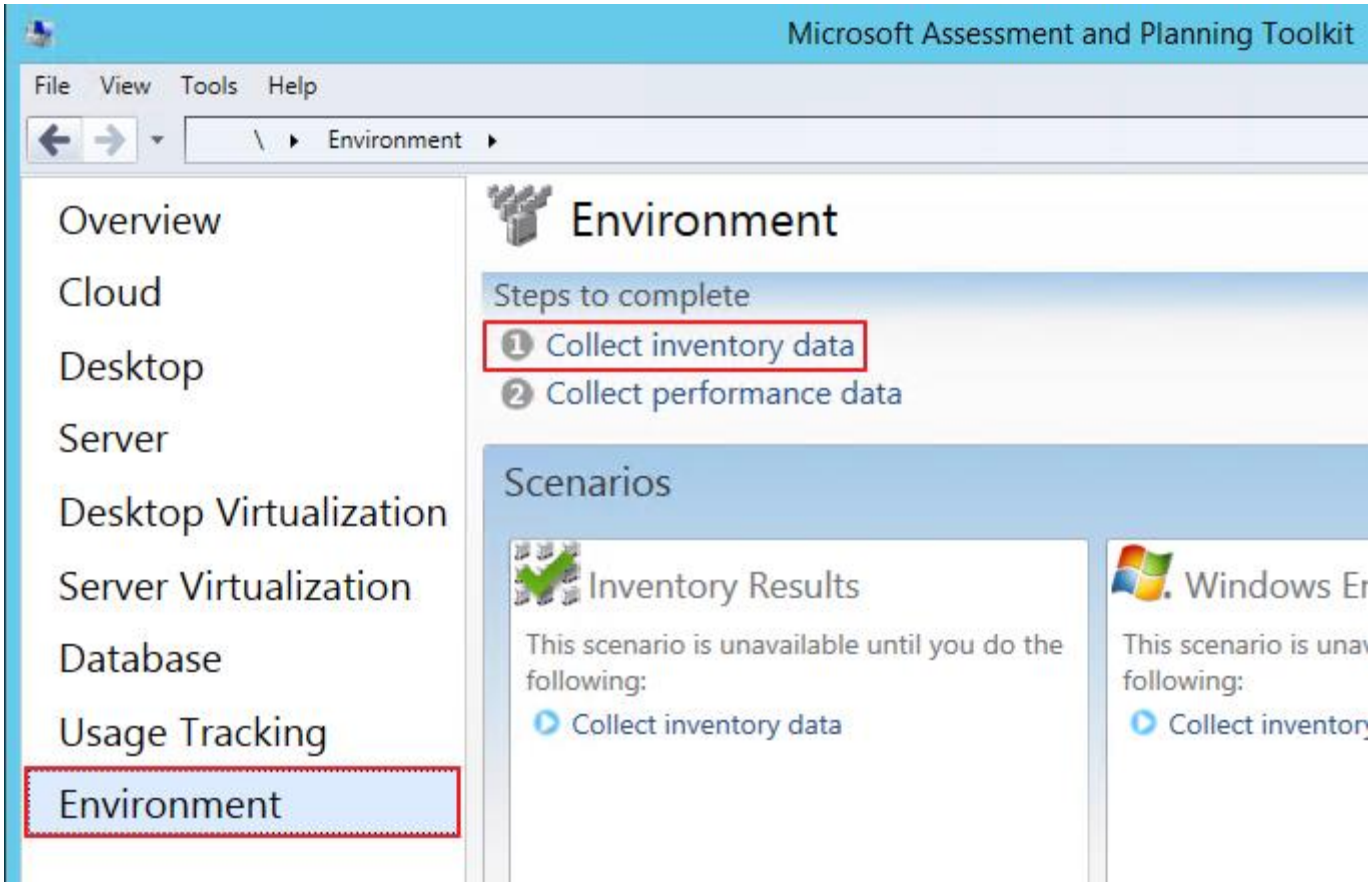


Figure - Collect inventory data

Select all that apply and click Next.

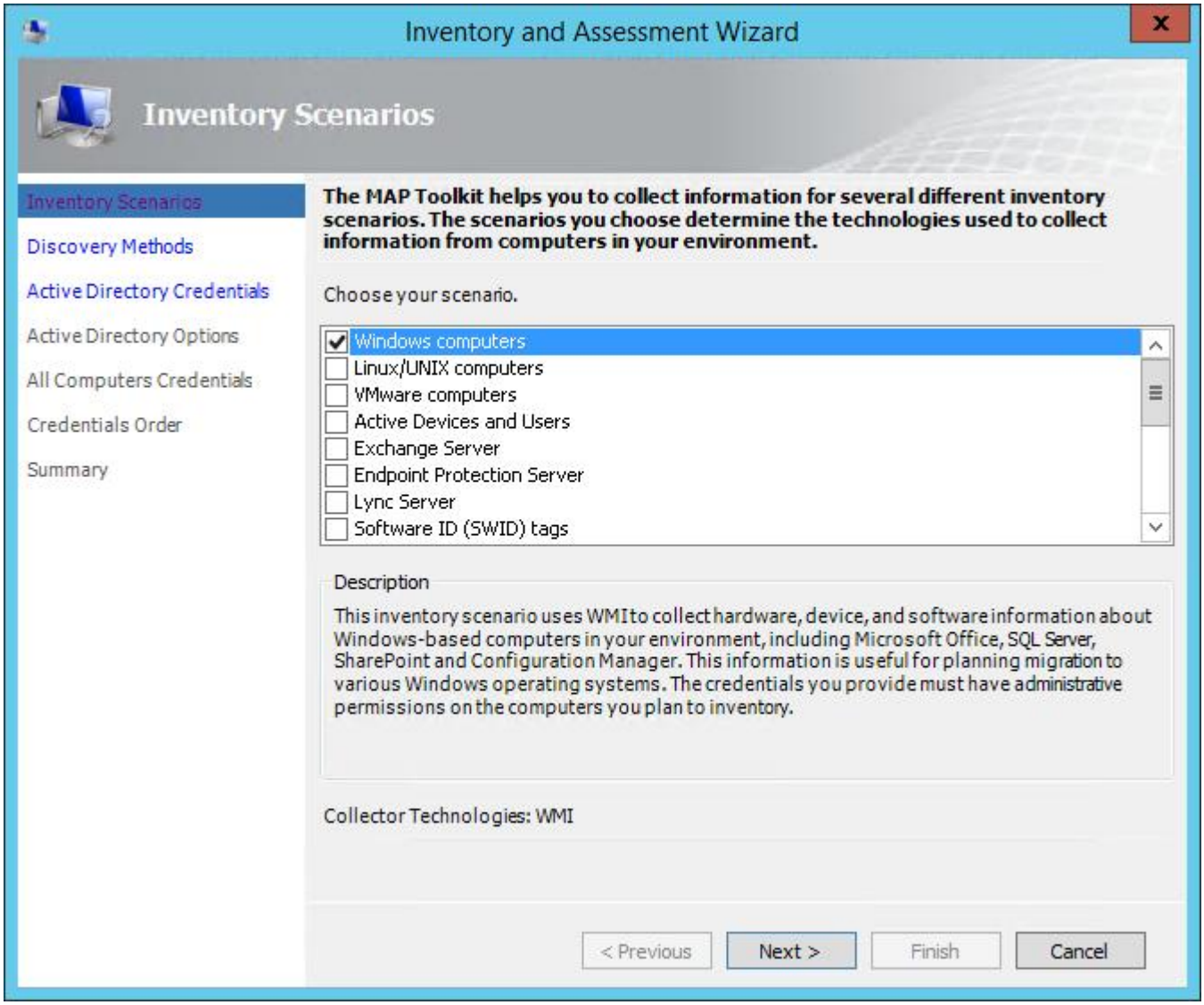


Figure - Choose scenario

Choose to leverage Active Directory to discover from. Additionally you can leverage alternative methods to discover machines not found in AD. Click Next.

We must provide domain credentials that have rights to be able to query active directory. Input the data and click Next.

On the AD options, you can select the default to scan the entire domain, or if all servers are in known OU's, you can select specific AD containers to search in. Click Next.

On the credentials page, we need to input a credential that has local administrator rights on all machines in the domain. This is required as MAP will connect to each machine and inventory details from WMI. For this purpose a domain administrator account works best, or a domain account that is a member of the local administrators group of each server in the domain. Click "Create" and input the credentials. You can input multiple credentials here and all will be attempted if one fails, however, this could extend the time required to run the inventory. When complete, click Next.

On the Credentials Order screen, you can change the order of multiple credentials if entered. Click Next. Click Finish.

Inventory will start immediately. Querying the data from AD will occur rather quickly. However, connecting to each server on the network via WMI will take considerable time, even days, depending on how large the environment. Allow this to complete, as below:

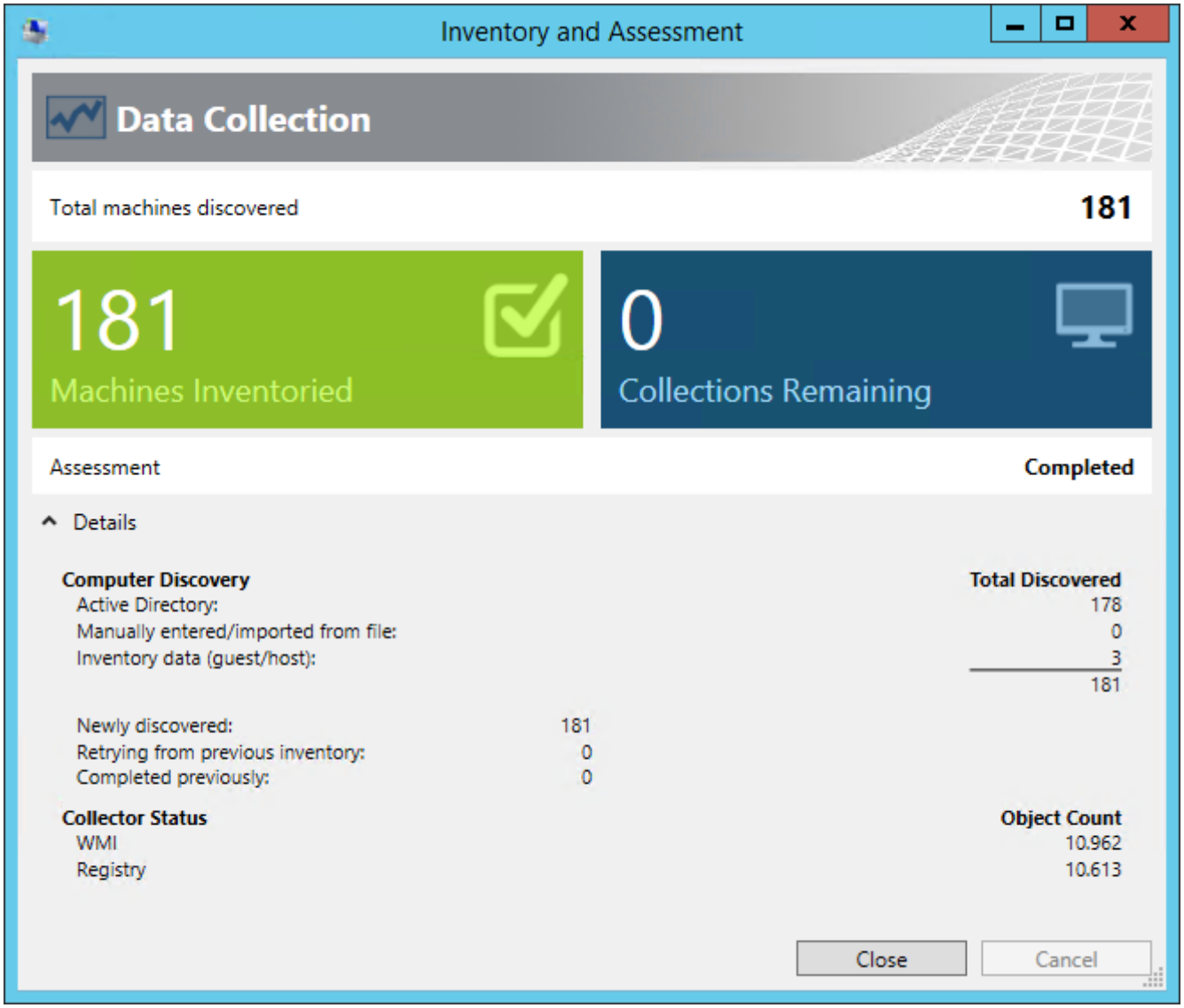


Figure - Data Collection

Once this is complete – you will see the amount of success/failure between what was queried from AD, and what was actually reachable via WMI.

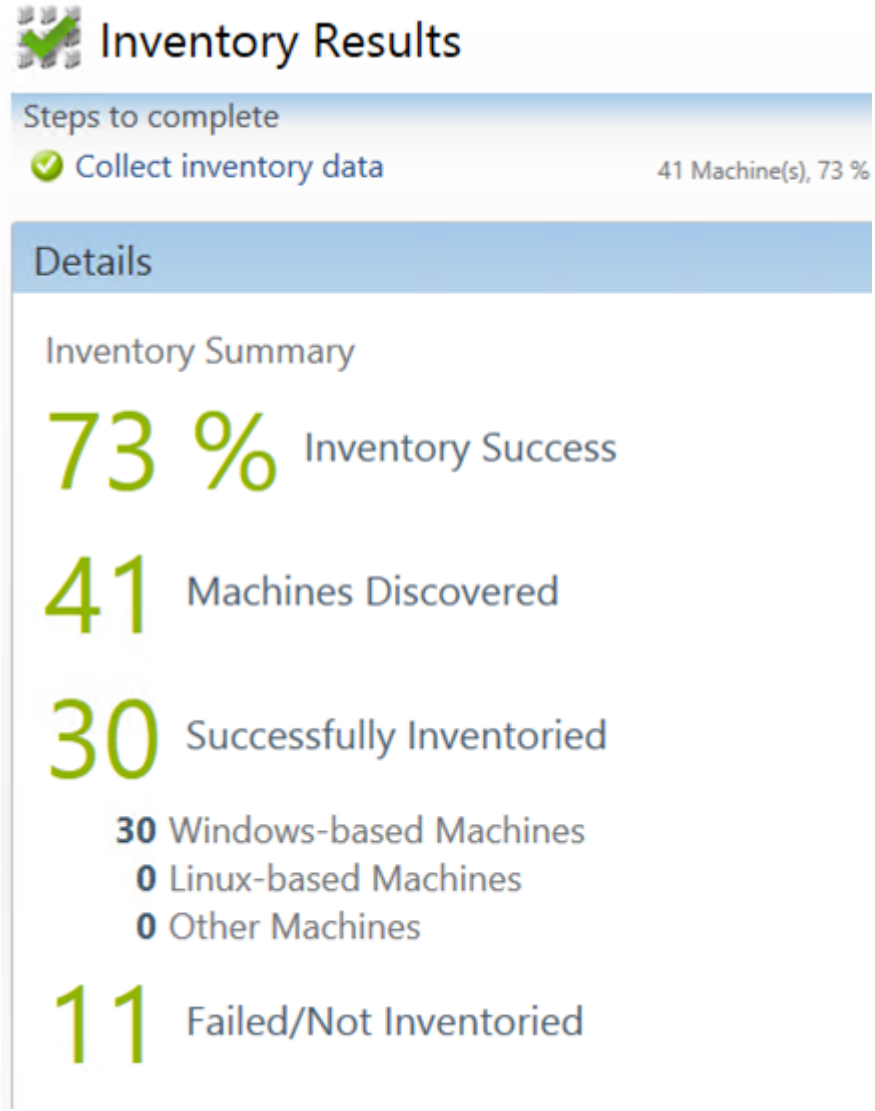


Figure - Inventory Results

Gathering data from VMware

This is covered at: http://social.technet.microsoft.com/wiki/contents/articles/12160.map-prepare-the-environment-vmware.aspx

In the inventory collection check the box for VMware:

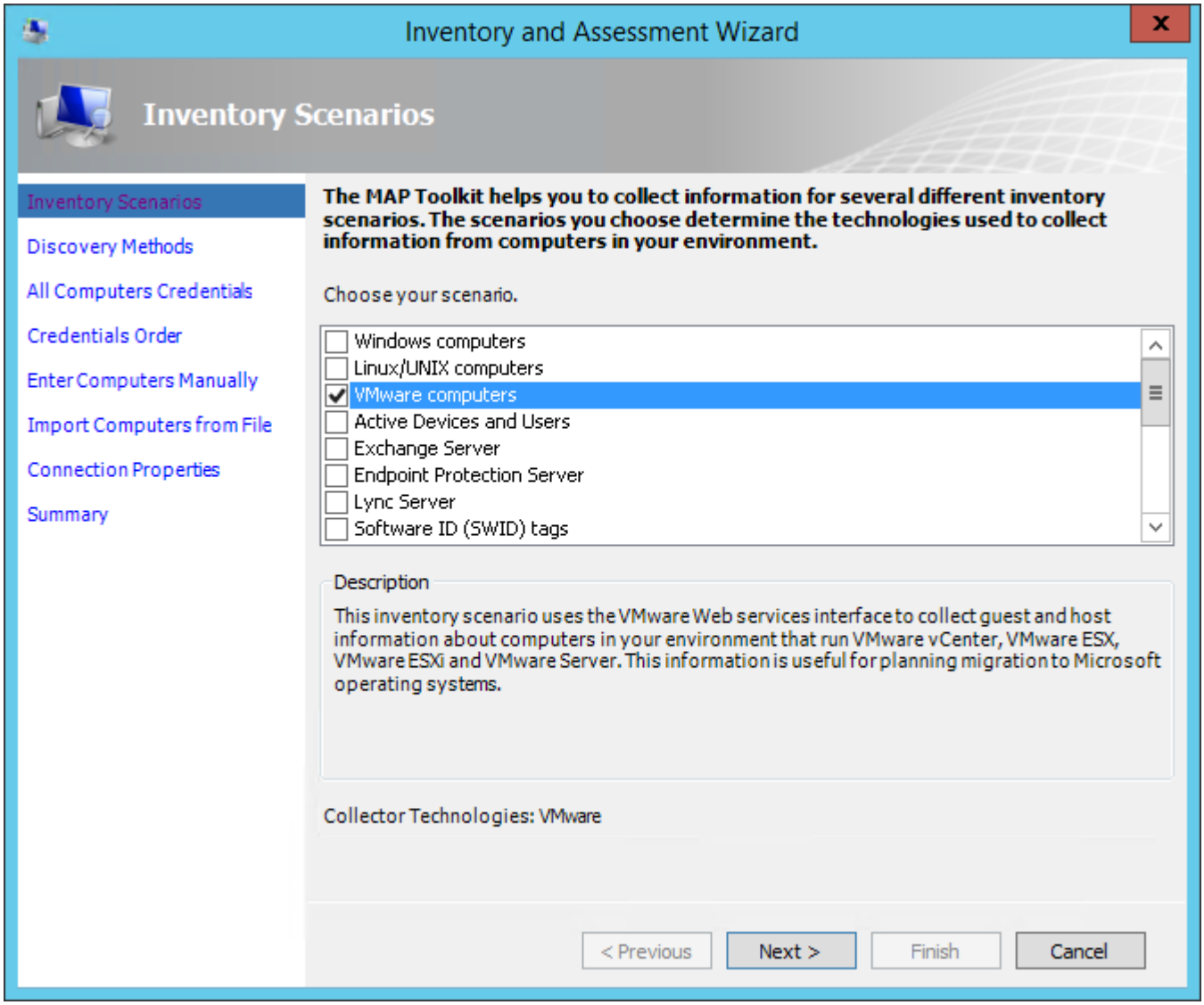


Figure - VMware Computers

Choose to manually provide a list of vCenter server names.

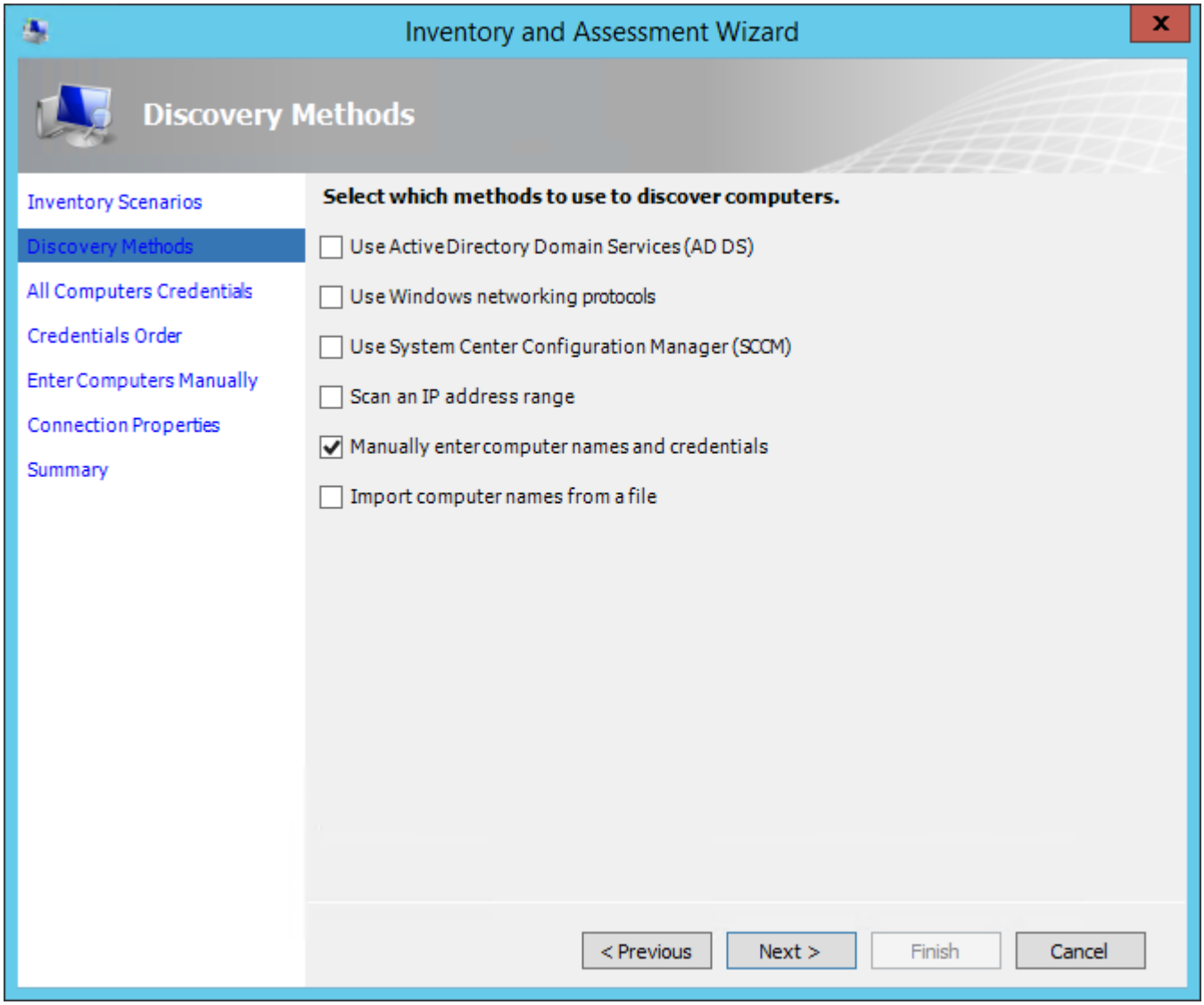


Figure - Manually enter...

Provide credentials that have access to the vCenter servers:

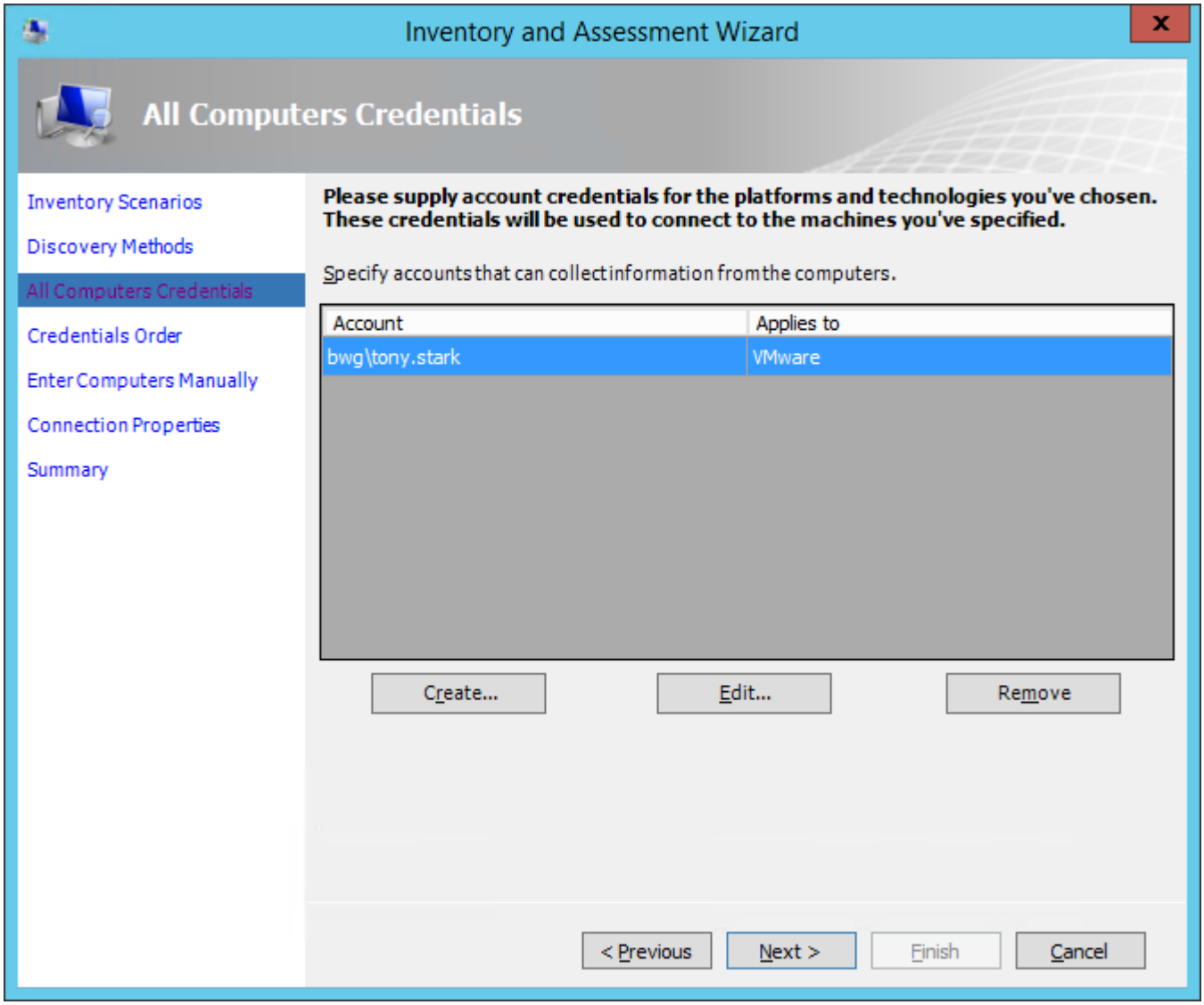


Figure - Credentials

If multiple credentials are used specify their order.

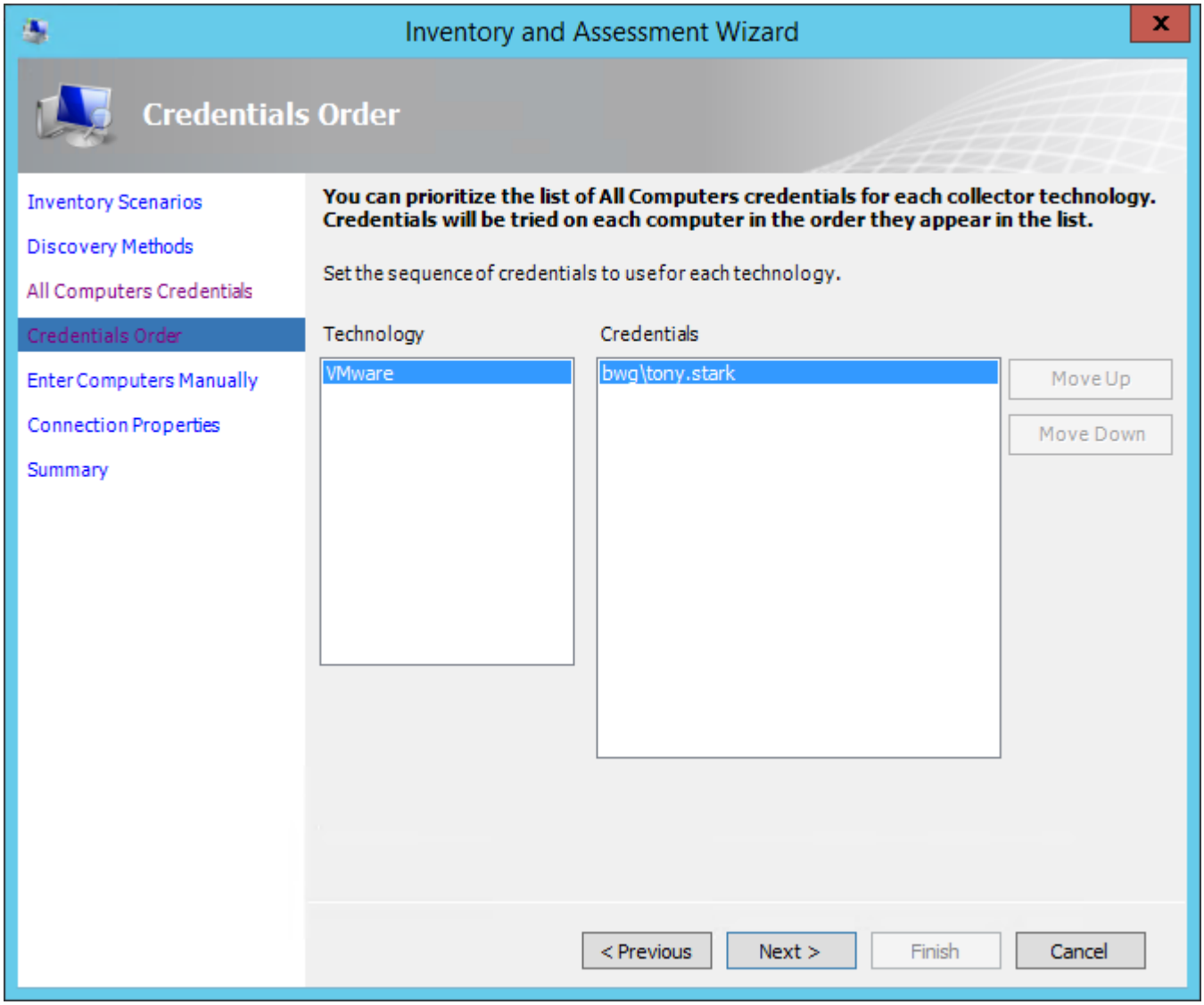


Figure - Credentials Order

Provide a list of server names that run vCenter:

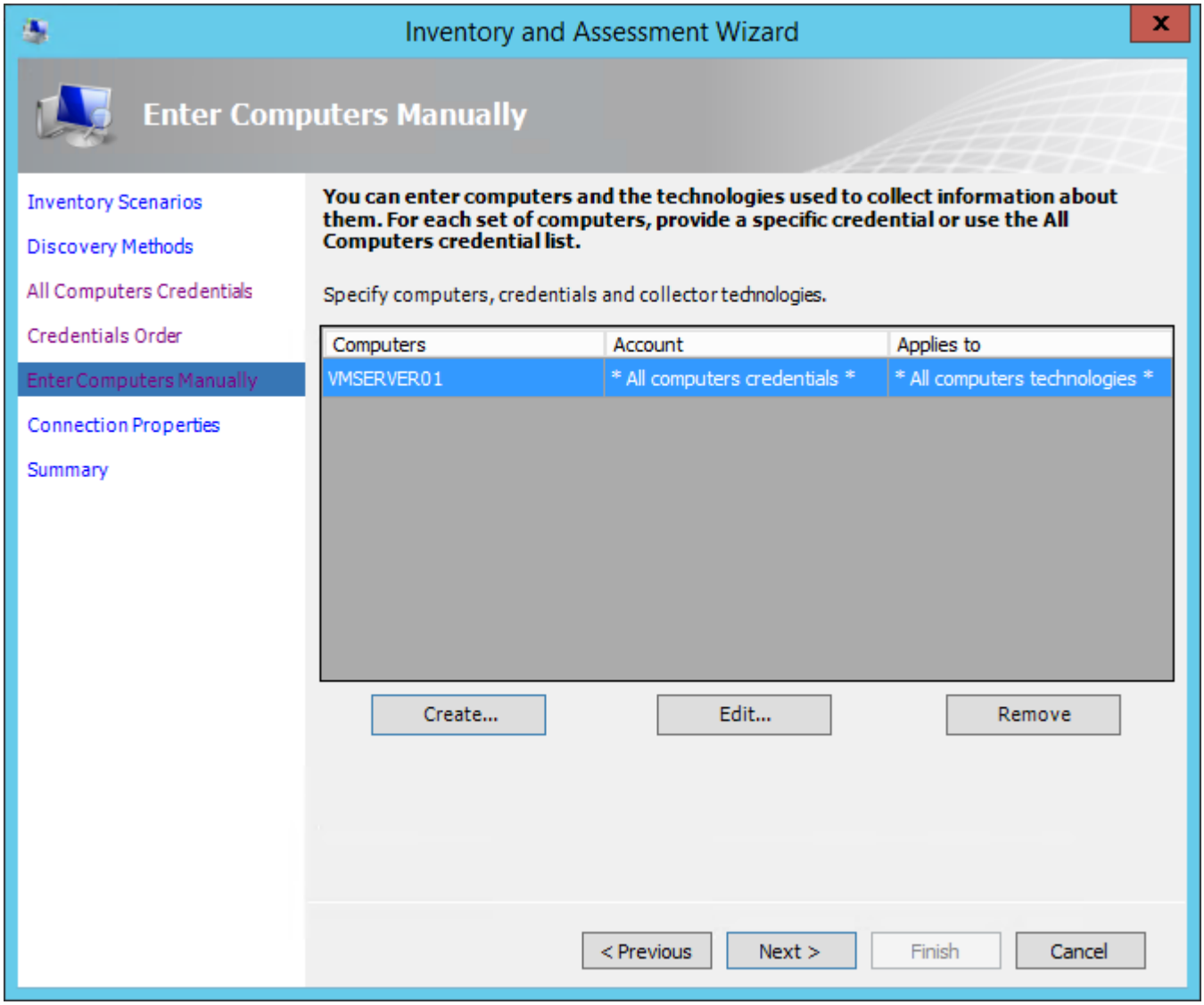


Figure - vCenter Servers

Configure the properties of your vCenter servers:

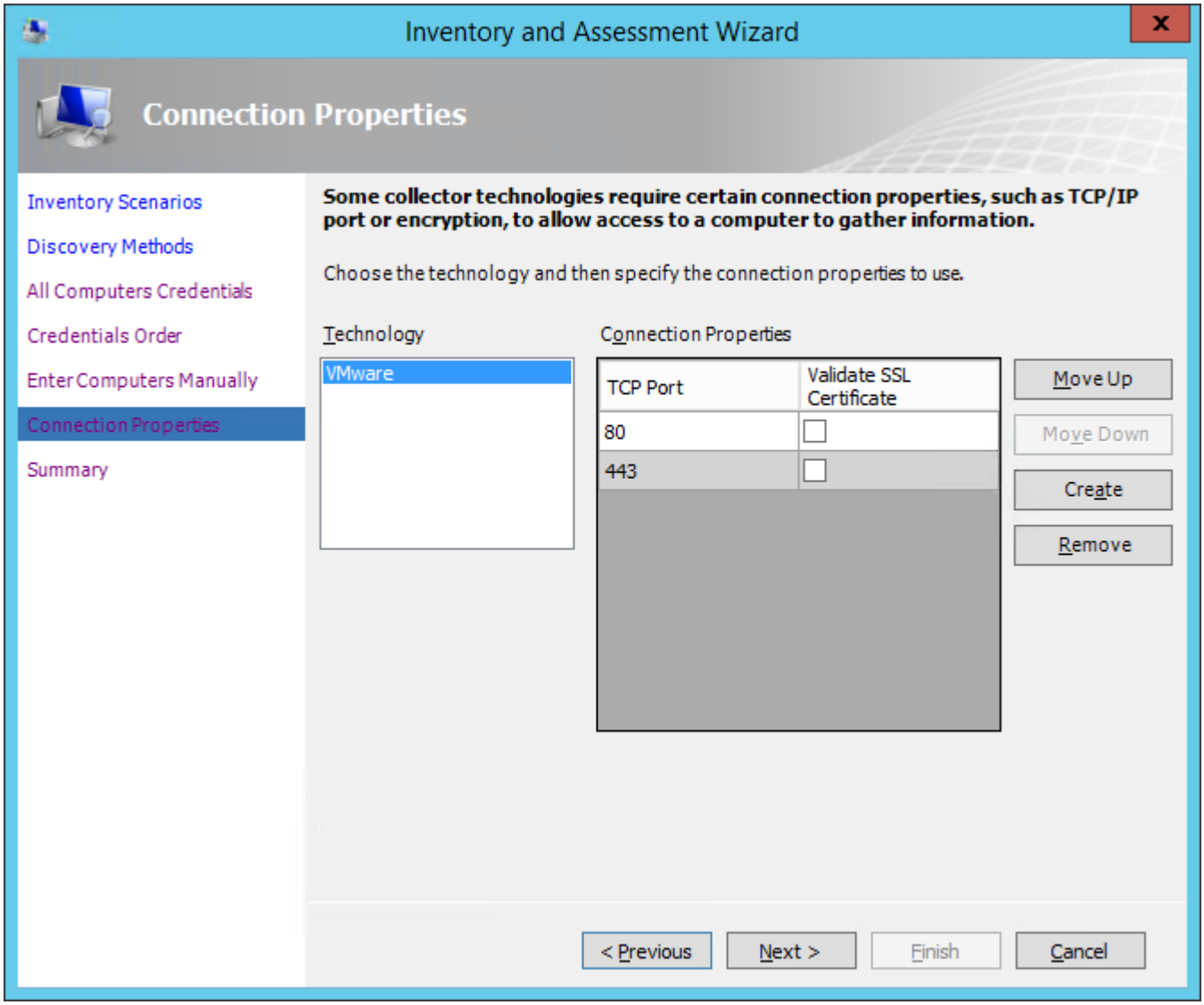


Figure - vCenter configuration

Check if all options are correct in the summary and click Finish.

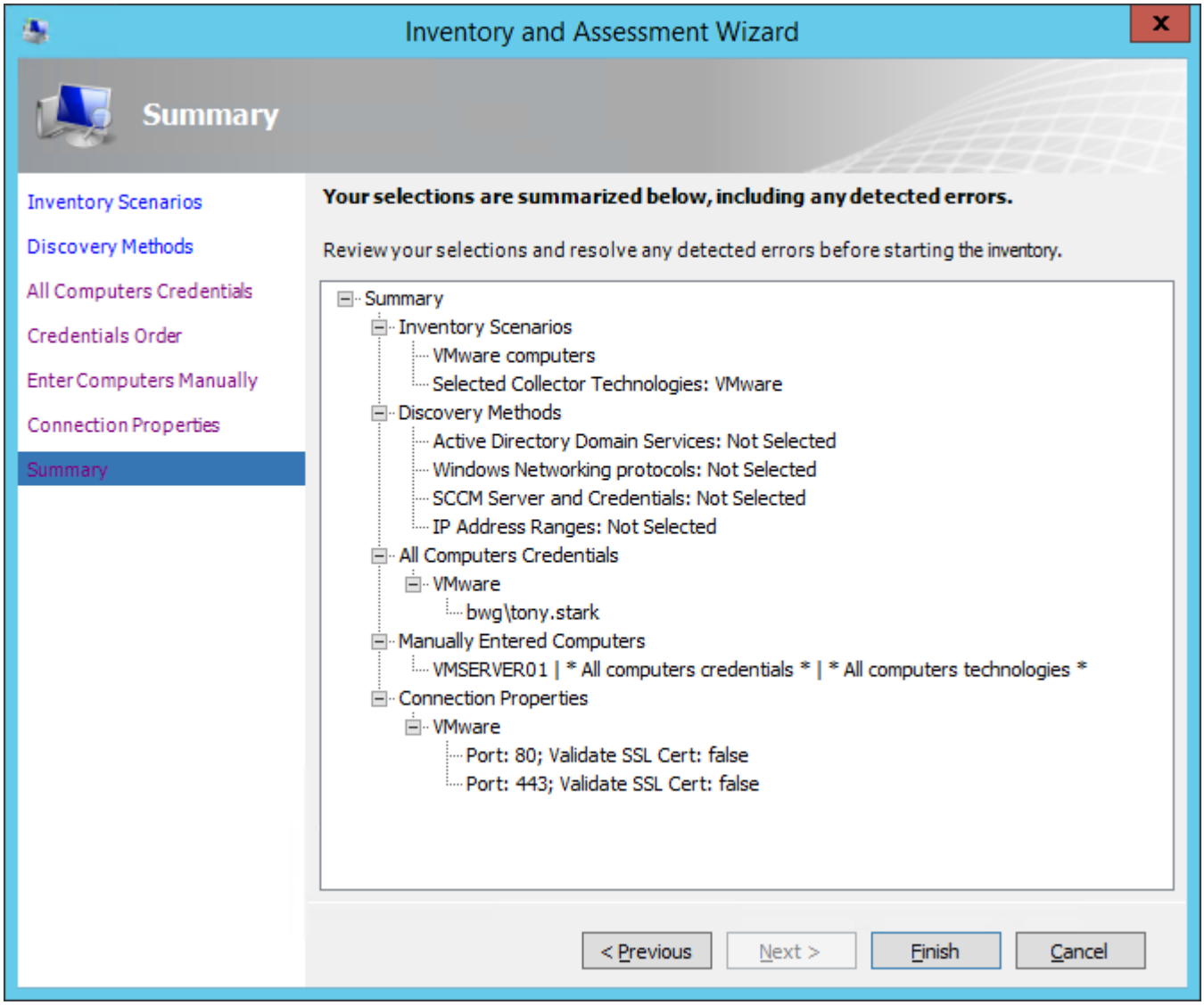


Figure - Summary

After the collection has finished you can then work with the data.