



RAYVENTORY®

The most comprehensive
Solution for Discovery and
Inventory of Software
and Hardware

User Guide
12.2



**Copyright © Raynet GmbH (Germany, Paderborn HRB 3524). All rights reserved.
Complete or partial reproduction, adaptation, or translation without prior written permission is prohibited.**

User Guide

Raynet and RayFlow are trademarks or registered trademarks of Raynet GmbH protected by patents in European Union, USA and Australia, other patents pending. Other company names and product names are trademarks of their respective owners and are used to their credit.

The content of this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Raynet GmbH. Raynet GmbH assumes no responsibility or liability for any errors or inaccuracies that may appear in this document. All names and data used in examples are fictitious unless otherwise noted.

Any type of software or data file can be packaged for software management using packaging tools from Raynet or those publicly purchasable in the market. The resulting package is referred to as a Raynet package. Copyright for any third party software and/or data described in a Raynet package remains the property of the relevant software vendor and/or developer. Raynet GmbH does not accept any liability arising from the distribution and/or use of third party software and/or data described in Raynet packages. Please refer to your Raynet license agreement for complete warranty and liability information.

Raynet GmbH Germany
See our website for locations.

www.raynet.de

Contents

| | |
|--------------------------------|----|
| Introduction | 8 |
| About this Guide | 8 |
| Documentation Requests | 8 |
| Installation | 9 |
| Prerequisites | 9 |
| License Manager | 10 |
| Getting Started | 11 |
| Software Architecture | 11 |
| Credential Store | 12 |
| Discovering the Network | 12 |
| Discovering Oracle Databases | 13 |
| Manually Adding Devices | 14 |
| Running Inventory | 15 |
| Uploading and Reporting | 15 |
| Automation | 16 |
| Maintenance | 17 |
| Inventory Agent | 17 |
| Home Screen | 18 |
| About Screen | 20 |
| Troubleshooting | 20 |
| Devices + Services | 22 |
| Overview | 22 |
| Devices | 24 |
| Recent Scan Details | 26 |
| Viewing Inventory Details | 31 |
| Software | 33 |
| Hardware | 33 |
| Services | 34 |
| Server features | 35 |
| Docker | 35 |
| Raw data | 36 |
| Adding a New Device | 37 |
| Device Capabilities | 39 |
| Editing Devices | 41 |
| Removing Devices | 42 |
| Working with Custom Attributes | 42 |
| Defining custom attributes | 42 |

| | |
|---|-----------|
| Editing custom attributes for devices | 43 |
| Displaying and data-shaping | 44 |
| Oracle | 45 |
| Recent Scan Details | 48 |
| Viewing Inventory Details | 48 |
| Adding New Database Connections | 49 |
| Database Connection Capabilities | 50 |
| Editing Database Connections | 52 |
| Removing Database Connections | 54 |
| Support for Review Lite Script | 55 |
| Support for Database Feature Usage Script | 55 |
| ESX / vSphere | 56 |
| Recent Scan Details | 59 |
| Viewing Inventory Details | 59 |
| Adding a new ESX / vSphere Connection | 61 |
| vSphere Connection Capabilities | 63 |
| Editing vSphere Connections | 65 |
| Removing vSphere Connections | 66 |
| SNMP | 66 |
| Recent Scan Details | 69 |
| Viewing Inventory Details | 69 |
| Adding a New SNMP Connection | 70 |
| SNMP Connection Capabilities | 72 |
| Editing SNMP Connections | 74 |
| Removing SNMP Connections | 75 |
| Hyper-V | 75 |
| Users / Groups | 76 |
| Import AD Users and Groups Wizard | 77 |
| Credential Store | 79 |
| Credential Type Windows | 81 |
| Credential Type SSH | 81 |
| Credential Type Oracle | 83 |
| Credential Type vSphere / ESX | 83 |
| Credential Type SNMP | 85 |
| Adding New Credentials | 85 |
| Discovery Wizard | 86 |
| Discovering New Devices | 86 |
| Methods | 87 |
| Active Directory | 87 |

| | |
|--|------------|
| Network Scan | 90 |
| Services | 91 |
| Inventory | 92 |
| Summary | 94 |
| Progress and Results | 94 |
| Discovering Services on Existing Devices | 96 |
| Services | 96 |
| Inventory | 97 |
| Progress and Results | 98 |
| Inventory Wizard | 100 |
| Target | 102 |
| Select Devices | 105 |
| Filtering | 105 |
| Inventory Overview | 107 |
| Summary | 108 |
| Progress and Results | 109 |
| Notification Center | 111 |
| Settings | 113 |
| General | 113 |
| Discovery | 114 |
| Inventory | 114 |
| Inventory Methods | 114 |
| Zero-Touch | 116 |
| Windows | 116 |
| Linux + UNIX | 118 |
| vSphere + ESX | 119 |
| Oracle | 120 |
| Remote Execution | 122 |
| Inventory Agent | 123 |
| Tagging | 125 |
| Parallelism | 125 |
| Health Assessment | 126 |
| Custom properties | 128 |
| HTTP Services | 128 |
| Server | 128 |
| Upload Location | 130 |
| Task Scheduler | 131 |
| Credential Store | 132 |
| Scheduling | 133 |

| | |
|---|------------|
| Triggers | 134 |
| Scheduled Operations | 135 |
| Remote OS Inventory | 135 |
| VMware Inventory | 137 |
| Oracle Inventory | 138 |
| SNMP Inventory | 139 |
| Discovery | 140 |
| Upload | 141 |
| Chaining Operations | 142 |
| Composite Operations | 142 |
| Inventory Agent | 144 |
| Installation | 144 |
| Configuration | 145 |
| Usage | 146 |
| Command-line | 146 |
| Usage agent | 147 |
| Scheduling | 149 |
| Logging | 151 |
| Commands | 151 |
| encrypt | 151 |
| getconfig | 151 |
| horizon | 152 |
| oracle | 153 |
| saas | 153 |
| schedule | 153 |
| settings | 153 |
| upload | 153 |
| Advanced Topics | 155 |
| Receiving Uploads from Remote Scans | 155 |
| Uploading Results to Parent Servers | 155 |
| Inventory Methods Overview | 156 |
| Windows Inventory | 157 |
| Linux / UNIX Inventory | 161 |
| Oracle Audit | 163 |
| Oracle Inventory | 163 |
| SNMP Inventory | 168 |
| ESX / vSphere Inventory | 169 |
| Application of Credentials | 170 |
| Custom Windows Scans | 171 |

| | |
|---|-----|
| Using Remote Execution Plugins | 176 |
| Commandline Tools | 177 |
| Remote Inventory for Windows (RIW) | 177 |
| Remote Inventory for Unix / Linux (RIU) | 179 |
| VMware Inventory | 182 |
| ORATRACK | 184 |
| NDTRACK | 190 |
| SNMP Tracker | 198 |
| PowerShell Automation | 200 |
| Get-DeviceConnections | 200 |
| Send-Inventory | 202 |
| OsConnection | 202 |

Introduction

RayVentory Scan Engine is a component of the RayVentory HAM / SAM solutions. A RayVentory implementation may use RayVentory Scan Engine as a source for RayVentory data where this data is processed by a RayVentory server installation.

About this Guide

This guide is an operation manual for RayVentory Scan Engine intended for end-users. The manual also covers certain aspects of RayVentory Scan Engine which are relevant to software architects and for the implementation of RayVentory Scan Engine.

Documentation Requests

We welcome suggestions and input on the various documentation resources available for RayVentory Scan Engine and its components. Feedback and other concerns can be forwarded through local Raynet support representative.

Installation

The installation sources for RayVentory Scan Engine are provided as an MSI package.

Prerequisites

RayVentory Scan Engine can be installed on Microsoft Windows 7 or later. It does not require a Microsoft Windows Server Edition, but can be installed on those as well.

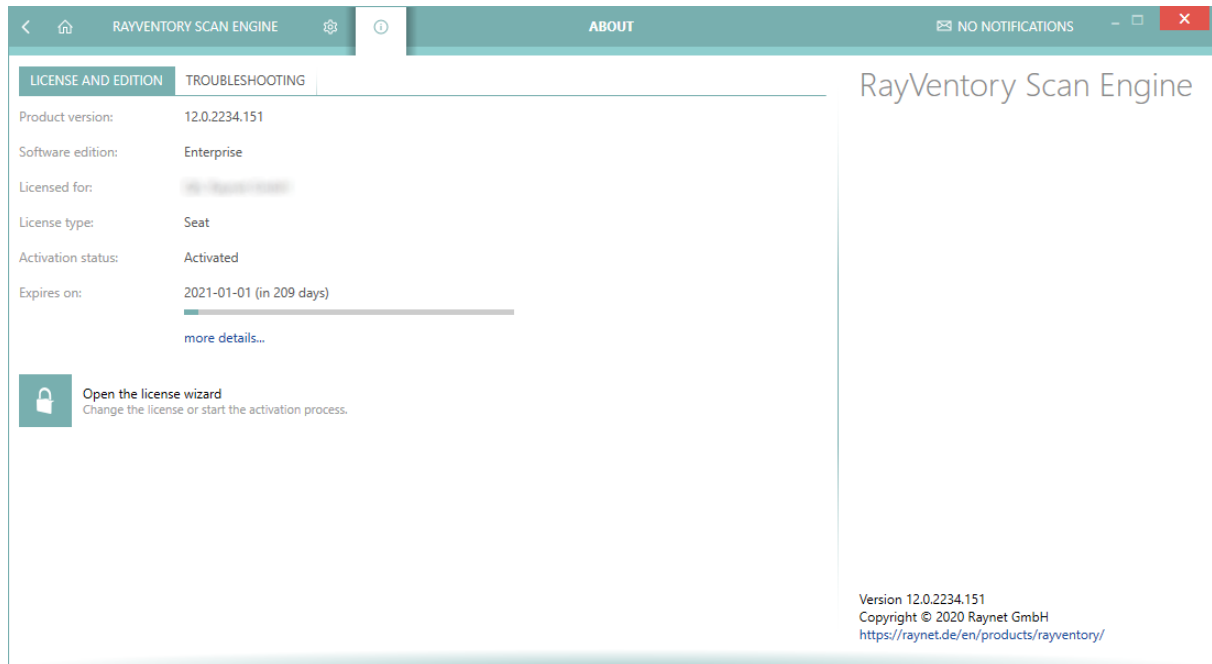
Furthermore, RayVentory Scan Engine requires the .NET Runtime Environment to run.

In order to use the Oracle Inventory feature of RayVentory Scan Engine, at least a Java SE 6 compatible runtime environment is required. RayVentory Scan Engine automatically detects the installed Java Runtime Environments and prompts the user to pick one if necessary.

The Oracle Audit feature (support for running the Oracle Review Lite Script) and the Oracle Database Feature Usage Statistics feature (support for running the Oracle Database Feature Usage Statistics script) require SQLplus to be installed.

License Manager

RayVentory Scan Engine is being controlled by a license. The product can be activated by using the built-in license manager with either a license file or an order number. The license controls the available features and it may have an expiration date.



Getting Started

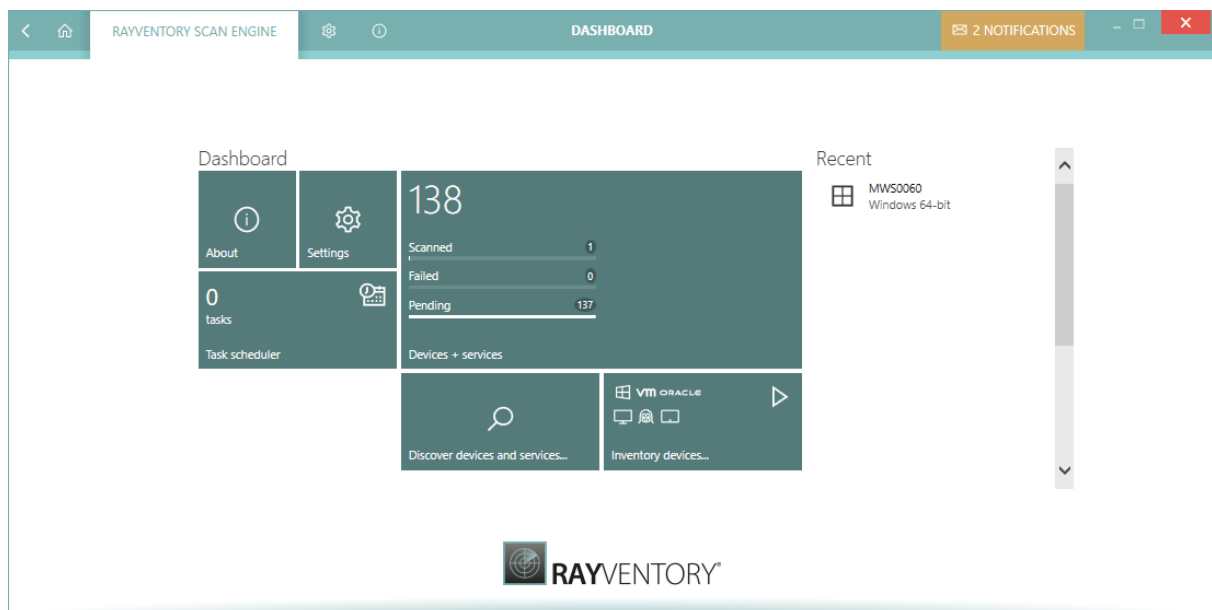
RayVentory Scan Engine is intended to easily gather inventory data on computers in the network. The many components available for implementation in RayVentory Scan Engine allow to suit a wide range of different requirements.

This chapter describes how RayVentory Scan Engine ideally is operated. For simplicity, one of the simpler solutions is discussed, the one that only uses the components RayVentory Scan Engine and RayVentory Server.

Software Architecture

The simplest approach to work with RayVentory Scan Engine is to use it in conjunction with the RayVentory server. In this case, RayVentory Scan Engine discovers devices / services and gathers inventory data, which are afterwards uploaded to the Server. In turn, the RayVentory Server processes the discovery and inventory data to provide reports for the hard- and software asset management. The intelligence from these reports can be used to support management decisions.

A typical RayVentory implementation consists of a RayVentory Server and one RayVentory Scan Engine instance for each site. All RayVentory Scan Engine instances must be able to directly upload to the RayVentory Server or to indirectly upload to the RayVentory Server by uploading to another RayVentory Scan Engine instance which, in this case, acts as a relay.



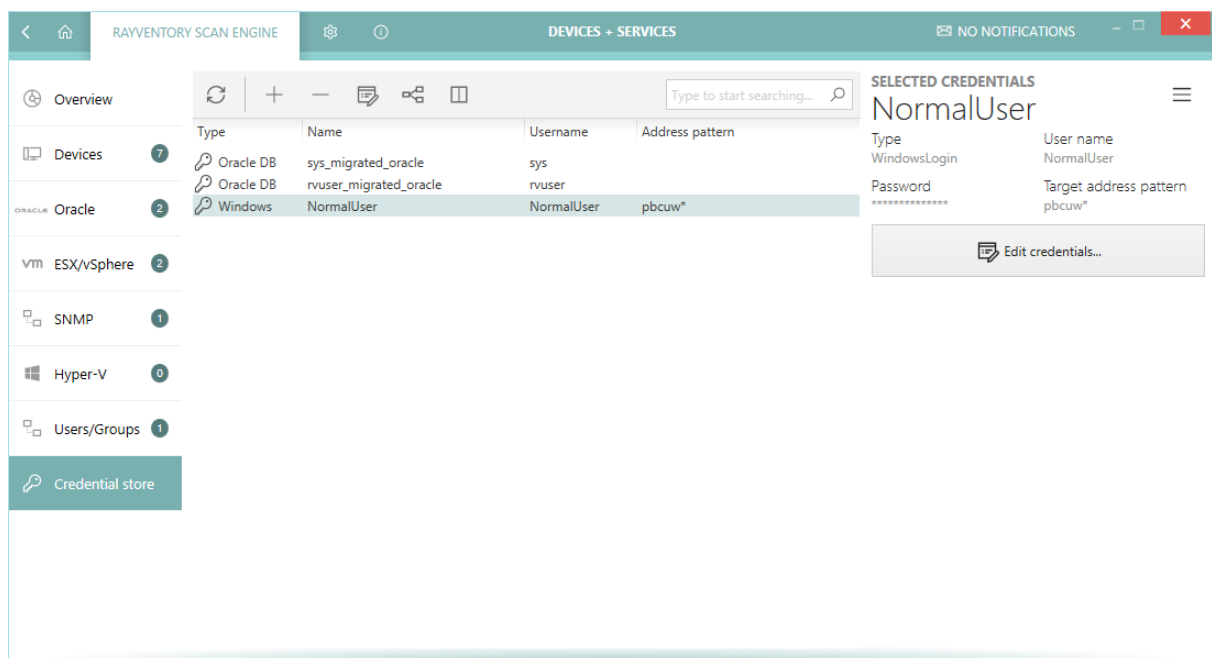
Unless a VPN for cross-site connectivity is present, the upload to RayVentory Server would have to use a WAN connection. For an upload via WAN, it is recommended to use RayVentory Server with HTTPS instead of HTTP. For using HTTPS a certificate is needed to be installed on the

RayVentory Server host and IIS must be configured to use it. If the certificate has not been created / published by a certificate authority known to the RayVentory Scan Engine hosts then the public part of the certificate must be installed there too.

Credential Store

The account information that are needed for scanning are stored in the Credential Store.

The Credential Store can be opened by clicking on the **Devices + Services** tile on the Dashboard, and then selecting **Credential Store** tab. Clicking on **+** icon will open the credential wizard. Choose the login type of the new account and click on **Accept** to proceed. Fill in the necessary information into the fields that define the credentials.



| Type | Name | Username | Address pattern |
|-----------|------------------------|------------|-----------------|
| Oracle DB | sys_migrated_oracle | sys | |
| Oracle DB | rvuser_migrated_oracle | rvuser | |
| Windows | NormalUser | NormalUser | pbcuw* |

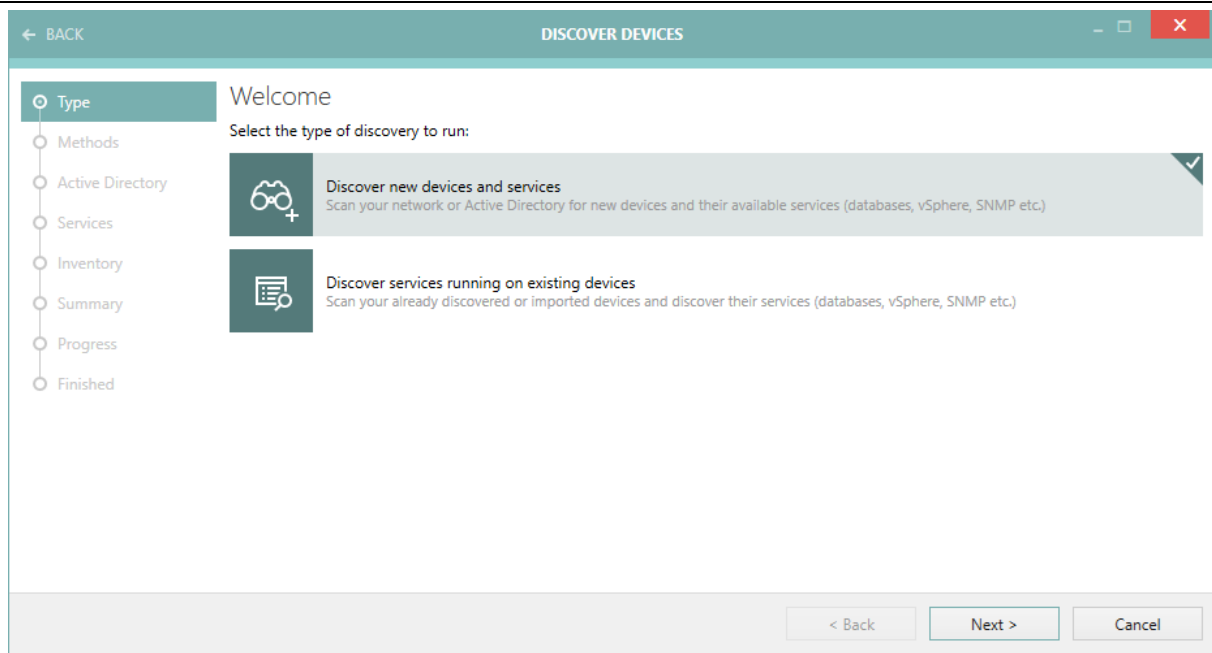


Note:

Windows credentials are needed for OS inventories on Windows based hosts. SSH credentials are needed for OS inventories on hosts running Linux and Unix-like platforms. DB credentials are needed for inventories of Oracle databases and vSphere / ESX credentials are needed for inventories on VMware vSphere / ESX virtual infrastructures.

Discovering the Network

Before discovering devices and services the inventory scope needs to be defined: For what devices and services is data needed and what devices and services are expected.



Currently, RayVentory Scan Engine gathers data on computers and network devices from Active Directory, network scans, and VMware ESX / vSphere infrastructures. RayVentory Scan Engine gathers device specific data on OS, platform, hardware, and installed software for Windows, a range of Linux distributions, and certain unix-like OSs like HP-UX (11i v1-v3), AIX (7.1, 7.2), OSX / MacOS, and later versions of Solaris / SunOS. It gathers service specific data on Oracle Databases from version 9 to 12 and MS SQL Server. Furthermore, hardware and configuration data is collected for a range of SNMP enabled devices.


Hint:

More options like custom inventory of WSMAN enabled devices or the support of virtual infrastructures managed by OpenStack are available for RayVentory Server.

The Discovery Wizard allows to configure and to perform device and service discovery. The discovery offers an overview on what computers, Oracle Databases, ESX / vSphere hosts, and SNMP devices exist in the infrastructure. Later, the inventory collects and offers details.

Successive discovery runs are cumulative – later results are merged into existing discovery results and may update hostnames and IP addresses by DNS query and target OS type by port probing or Active Directory attributes.

More information:

- Discovery Wizard

Discovering Oracle Databases

Discovering Oracle Databases during a discovery run can be achieved by two different methods. The default method requires knowledge of the ports that TNSListeners is listening to, for probing hosts for indication of a potential Oracle Database and credentials of type Windows resp. SSH, for

authentication as privileged users to the Databases' host systems. The credentials are used for reading the configuration by remote execution, resp. WMI query. Such users need at least permissions to read `/etc/oratab` on a Linux / unix-like host systems and to query `Win32_Process` by WMI on a Windows host system.

The advanced Oracle Database discovery option, named **Use Remote-Execution based Oracle discovery** in the wizard does not need knowledge on the TNSListeners' ports for discovery. This method also needs credentials of a privileged user to perform remote execution. The users for Linux / unix-like host systems should have permissions for reading environment variables, files in the Databases' home directories like `listener.ora`, `tnsnames.ora`, and `sqlnet.ora`, the files `/var/opt/oracle/oratab`, `/etc/oratab`, `.oratab` and to execute `lsnrctl`. The users for Windows like host systems should have permissions for reading environment variables, files in the Databases' home directories like `listener.ora`, `tnsnames.ora`, and `sqlnet.ora`, to execute `lsnrctl` and to query the registry HKLM.

**Tip:**

The advanced Oracle Database discovery option can also be run on specific devices from within the Devices list, by the context menu option **Oracle Discovery**.

The discovery is able to discover database connections on its own, but this feature is still very limited and requires a user with sufficient privileges to read certain configuration files on a unix-like host or to query running processes by WMI on a Windows host. Adding connections manually is also an option that is worth considering.

More information:

- Settings for Oracle Zero-Touch Scan
- Overview of Oracle Instances
- Discovery Wizard

Manually Adding Devices

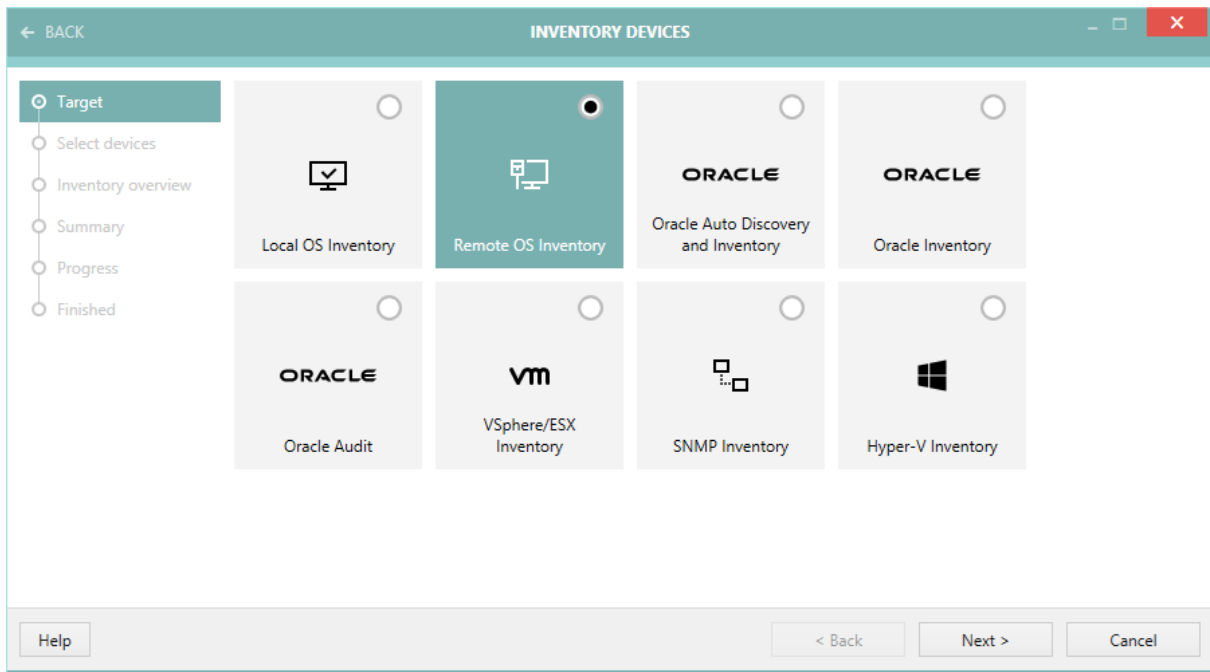
It is possible to manually add hosts and services using the related screens (devices and services like **vSphere**, **Oracle databases**, and **SNMP**).

To access the overview of the connections, open the **Devices + Services** screen directly from the dashboard. The overview is common for all base types and each has a dedicated tab on the left side that contains a dedicated data grid.

Press the **+** icon to add a host or click **Edit** to change the properties of an existing host. This opens the properties dialog which can be used to set the type of operating system, the hostname, or network address and optionally choose the credentials that are tried first when authenticating to this host for running the inventory. Finally, the dialog allows for the configuration of the device-specific capabilities which will later be used to determine how to access it when doing the inventory scan.

Running Inventory

The discovery wizard offers an option to directly run an inventory on newly found devices and services. Later, an inventory can be run on all or specific targets by using the inventory wizard, which can be started from the home screen and from within the context menu and side-bar in the device and service specific tabs of the **Devices + Services** screen.



By default, if there are multiple options, RayVentory Scan Engine is configured to find a suitable inventory method on its own. Currently, there are multiple options for Device and Oracle inventories. RayVentory Scan Engine shows its reasons for considering or discarding certain inventory methods in the inventory wizards. See [Application of Inventory Options](#) for requirements, prerequisites, and options.

RayVentory Scan Engine will save and show error messages for each inventory target and the failed inventory methods to assist troubleshooting. Additionally, RayVentory Scan Engine logs its activities and errors regarding discovery and inventory.

More information

- [Inventory Wizard](#)
- [Inventory Methods Overview](#)
- [Configuration of Inventory](#)

Uploading and Reporting

The intention of RayVentory Scan Engine is to gather data. It offers a basic reporting on inventory health and status to assist in managing an inventory campaign. The report **Summary of inventory health** integrates the inventory wizard to support workflows regarding inventory

updates and troubleshooting. The tiles on the report, which show number that represent certain sets of devices, are clickable. When clicked then the inventory wizard will be opened with the respective devices selected.

The devices and service lists offer views on the gathered inventory data including summaries of certain devices or service specific facts.

More in-depth reports that will show discovery and inventory data are available on the RayVentory Server. There are facts shown, that RayVentory Scan Engine will not show in its summaries, for example MS SQL Server instances or data on Oracle Database options.

The data gathered by RayVentory Scan Engine must be uploaded to RayVentory Server in order to fill the discovery and inventory database for reporting.

In the RayVentory Scan Engine settings, the upload endpoints for inventory and (optionally) discovery must be set. Uploading discovery data is needed to see the gap between devices known without inventory data and devices known with inventory data in the reports that the RayVentory Server offers. See **SETTINGS > HTTP Services > UPLOAD LOCATION URL to upload inventory files** and **URL to upload legacy discovery data**.

Usually, the URL is of the following form: `http://yourRVServerAddress/ManageSoftRL/inventories` **OR** `http://yourRVServerAddress/ManageSoftRL/discovery`.

More information:

- Uploading Results to Parent Servers
- Receiving Uploads from Remote Scans
- Settings for Server and Upload Location

Automation

For a continuous inventory, it is recommended to frequently run discovery, inventory, and upload to keep the inventory data up to date.

If a network discovery is needed instead of or in addition to the Active Directory import then such a discovery should be run at different times during the day, to cover devices which are rarely active or not active during the whole day like workstations and desktop computers. For the same reason, device inventories should be run at different times too.

All inventory operations can be limited to a certain set of targets either based on an explicit list of targets or implicitly, by a regular expression which is matched against the target hostname / address / URL. The Oracle inventory operation also allows to set regular expressions for matching SIDs / service names and ports. If no regular expression is set for a **Target definition by filter**, then all available targets will be addressed by the inventory operation.

Uploads may be run on a weekly or monthly basis.

RayVentory Scan Engine's built-in task scheduler enables automation of these operations. Each task may consist of an operation like upload, devices import, discovery, inventory, oracle discovery, or a sequence of these operations.

More information:

- PowerShell Automation
- Scheduled Operations
- Command-line Tools

Maintenance

Once RayVentory Scan Engine and RayVentory Server are configured to frequently run all necessary operations, the system health can be monitored by the inventory health related reports. Then RayVentory Scan Engine inventory health summary, the saved target status / error messages, the logs of RayVentory Scan Engine, and RayVentory Server can be used for troubleshooting.

From time to time updates and fixes / patches for RayVentory Scan Engine and RayVentory Server are released.

A knowledge base for self-service regarding troubleshooting is also available.

Inventory Agent

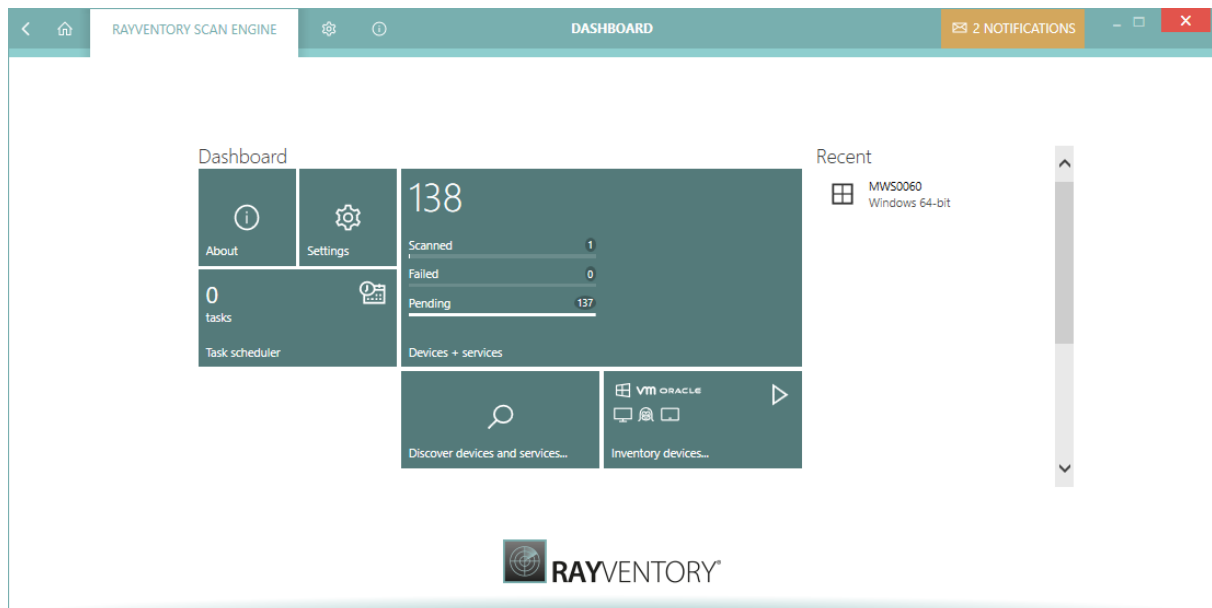
RayVentory Scan Engine includes a bundle for RayVentory Inventory Agent. It is designed to continuously deliver hard- and software inventory and usage data from computer systems running Windows, Linux or Unix.

The agent must be installed or deployed separately, as it runs remotely from the RayVentory Scan Engine.

For more information about how to install and set-up the Inventory Agent, refer to the dedicated chapter: Inventory Agent.

Home Screen

The home screen, also called **Dashboard**, grants access to the different functions and screens which are part of RayVentory Scan Engine. The tiles on the **Dashboard** can be used to start certain operations or to navigate to other screens where more operations and information will be available. Some of the tiles on the **Dashboard** also show information. An example for this is the tile showing the number of hosts which are currently known to RayVentory Scan Engine. Furthermore, the recent inventory results can be shown in the **Recent** panel.



There is a handful of buttons that lead to various screens containing data and settings of RayVentory Scan Engine.

About

Opens the **About** screen with license and troubleshooting data.

Settings

Opens the **Settings** screen with various options for inventory, execution, scanning, uploads, etc.

Task scheduler

Opens the view which shows scheduled tasks and operations

Devices + Services

The main view of RayVentory Scan Engine, which provides both, insights and reports on the already discovered and scanned assets, as well as, the ability to perform various operations like the import of devices, the execution of new discoveries and inventories, database maintenance,

etc.

Discover devices and services...

Opens the **Discovery** wizard, which allows for the import of new devices and services from Active Directory or from a network scan.

Inventory devices

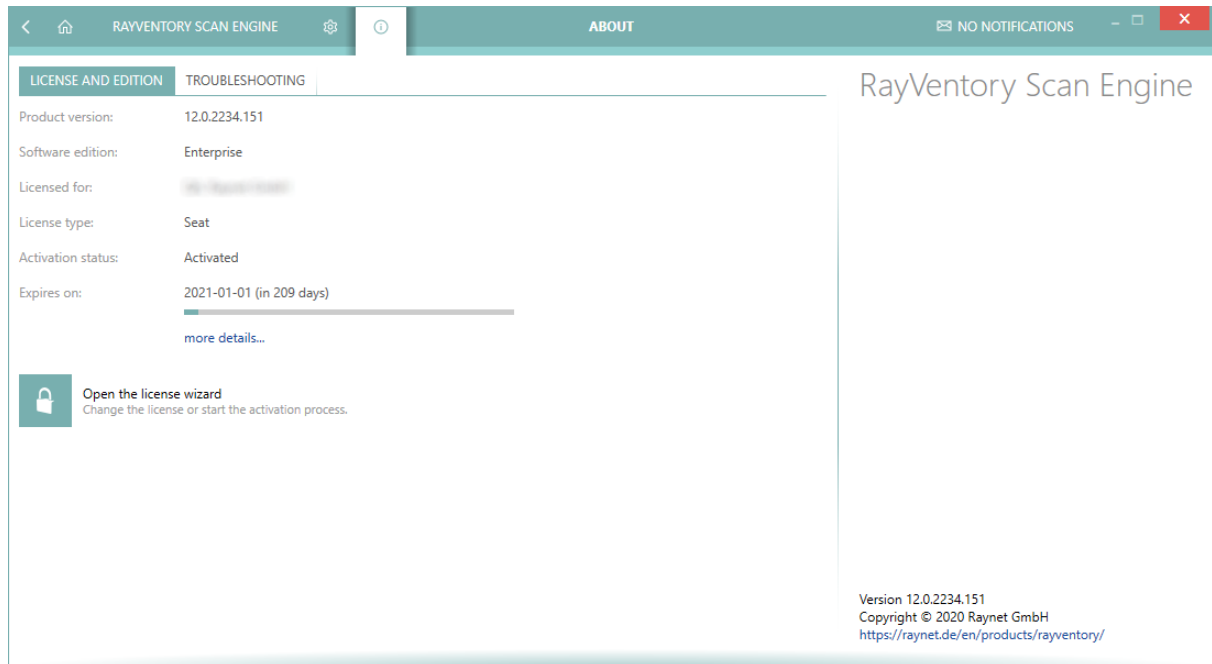
Opens the **Inventory** wizard, which allows to scan the already imported computers and services for a new software and hardware asset.

Recent

The view of recently inventoried devices and services. Clicking these items will jump to the corresponding place where more details can be seen.

About Screen

The **About** screen shows the version of the RayVentory Scan Engine installation. It also allows for access to the License Manager and the directory with the log files.



The **About** view shows basic information about the license, its expiration date, and the hardware ID. It is possible to view the details of the current Raynet license and / or apply a new license by pressing the **Open the license wizard** button.

Troubleshooting

The **TROUBLESHOOTING** tab contains various performance, diagnostics, and logging information.

Logs and diagnostic information

RayVentory Scan Engine writes its log files (by default) into a text file, which path is displayed in the UI. You can press the button **Open logs folder** to jump to this location.

Versioning

This section shows various sub-components belonging to RayVentory Scan Engine and their current version. The table can be easily copied to the clipboard.

**Note:**

Both information about the product version and the current log file may be required when opening a Support ticket for RayVentory Scan Engine.

Devices + Services

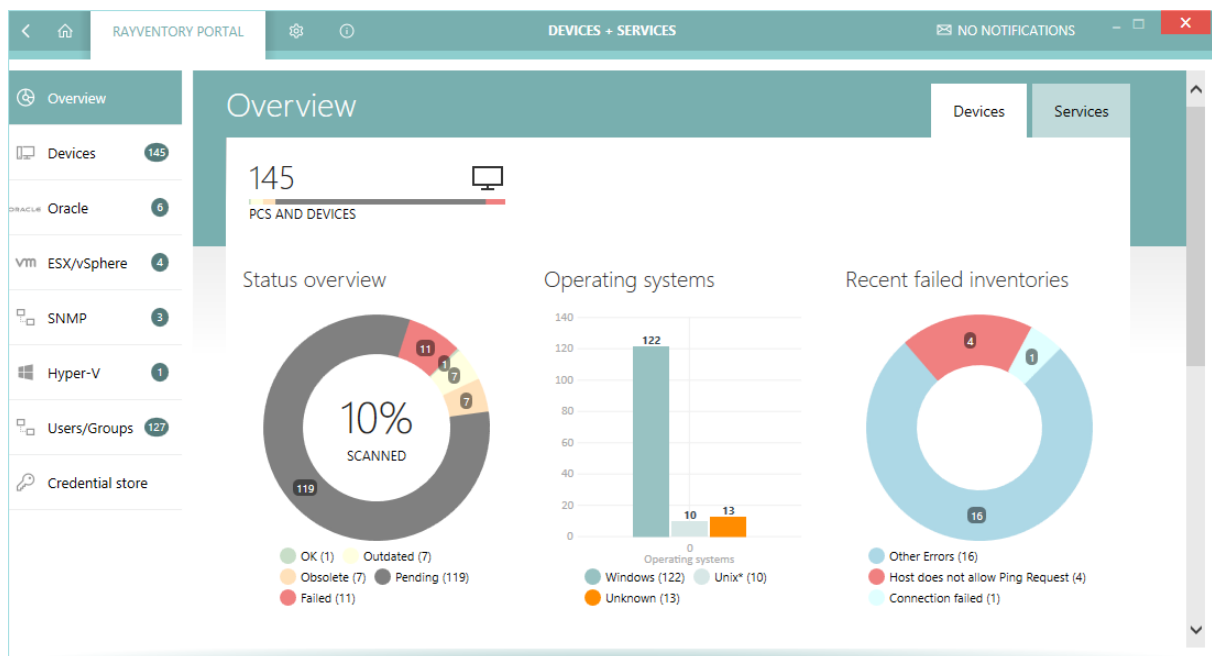
This is the core view of RayVentory Scan Engine. In this view the following options are available:

- Adding, importing, and managing connections (physical devices, virtual devices, databases, and services)
- Triggering discovery and inventory on your asset
- Identifying old and failed scans
- Managing credentials
- Browsing inventory results of scanned assets

Overview

The **Overview** section is the dashboard which aggregates data from various sources and shows them in a simplistic style for an easier identification of:

- Successful, pending, and failed inventories
- Share of operating systems
- Top reasons of failed inventories
- A summary of inventory health



This view has two tabs on the top, which toggle between **Devices** and **Services** data sources. Operating systems (hardware and software scans) are contained within the first tab while services (Oracle, SNMP, vSphere / ESX) are part of the second tab.

Working with Inventory Health Overview





Each section (**Devices** and **Services** respectively) has its own dedicated table that identifies the "health" of recent inventories. An inventory file is considered "healthy" if (both must be true):

- The inventory finished successfully and
- The inventory was executed not more than 30 days ago

Inventory jobs that failed are presented in the **Failed** columns. Devices that have not been scanned yet are grouped under the **Pending** column. The three remaining columns (**0-30 days**, **30-90 days**, **90+ days**) show inventory jobs which succeeded sorted by their time frame.

As a rule of thumb, inventory results older than 90 days should be treated as already outdated and requiring attention.

Inventory summary

| | OK | Outdated | Obsolete | Pending | Failed |
|---|----|----------|----------|---------|--------|
|  Windows | 1 | 3 | 3 | 112 | 3 |
|  Unix* | - | 4 | - | 6 | - |
|  Other | - | - | 4 | 5 | 4 |
|  Total | 1 | 7 | 7 | 123 | 7 |

This view is interactive. It is possible to click on the cells with numbers inside to open the **Inventory Wizard** and automatically preselect the devices or services which match the corresponding criteria. For example (see the figures in the above picture):

- Pressing "5" in the row **Other** and the column **Pending** opens the Inventory Wizard with 5 unscanned devices with an unknown system preselected.
- Pressing "3" in the row **Windows** and the column **Failed** opens the Inventory Wizard with the Windows devices that failed preselected. The one device that succeeded will not be selected.



Note:

The rules for colouring and assessing the status are customizable. Refer to the chapter Health Assessment for more information.

Executing Common Tasks

The lower part of the overview contains shortcuts to some common operations:

Devices Tab

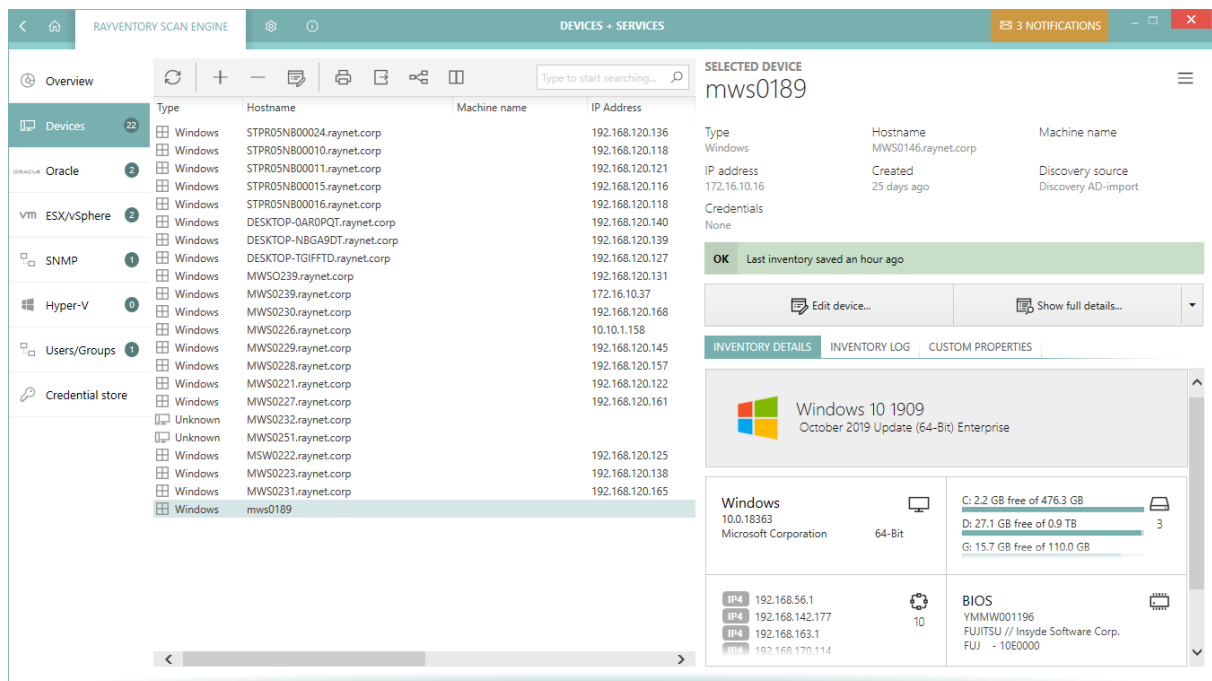
- **Scan your infrastructure for new or changed devices...** - opens the **Discovery** wizard.
- **Import devices connections from CSV file...** - opens the wizard to import data from .csv file.
- **Export device connections to XML file...** - opens a file open dialog prompting where to save the exported .xml file.

Services Tab

- **Export Oracle connections to XML file...** - opens a file open dialog prompting where to save the exported .xml file with all Oracle connections.
- **Export vSphere connections to XML file...** - opens a file open dialog prompting where to save the exported .xml file with all vSphere connections.
- **Export SNMP connections to XML file...** - opens a file open dialog prompting where to save the exported .xml file with all SNMP connections.

Devices

This view aggregates discovery and inventory data of your devices.



The screenshot shows the RAYVENTORY SCAN ENGINE interface. The top bar includes navigation icons, the title 'RAYVENTORY SCAN ENGINE', and a 'DEVICES + SERVICES' tab. On the left, a sidebar lists various categories: Overview, Devices (22), Oracle (2), ESX/vSphere (2), SNMP (1), Hyper-V (0), Users/Groups (1), and Credential store (1). The main area displays a table of devices with columns for Type, Hostname, Machine name, and IP Address. The table lists various Windows machines, including STPR05NB00024.raynet.corp, STPR05NB00010.raynet.corp, STPR05NB00011.raynet.corp, STPR05NB00015.raynet.corp, STPR05NB00016.raynet.corp, DESKTOP-QAR0PQT.raynet.corp, DESKTOP-NBGA9DT.raynet.corp, DESKTOP-TGIFFTD.raynet.corp, MWSO239.raynet.corp, MWSO239.raynet.corp, MWSO230.raynet.corp, MWSO226.raynet.corp, MWSO229.raynet.corp, MWSO228.raynet.corp, MWSO221.raynet.corp, MWSO227.raynet.corp, MWSO232.raynet.corp, MWSO251.raynet.corp, MWSO222.raynet.corp, MWSO223.raynet.corp, MWSO231.raynet.corp, and mws0189. The right sidebar shows the 'SELECTED DEVICE' details for 'mws0189', including its Type (Windows), Hostname (MWS0146.raynet.corp), Machine name, IP address (172.16.10.16), Created date (25 days ago), Discovery source (Discovery AD-import), and Credentials (None). Below this, there are buttons for 'Edit device...' and 'Show full details...'. The bottom section of the right sidebar shows 'INVENTORY DETAILS' for 'Windows 10 1909', including the operating system version, update information, and disk space usage (C: 2.2 GB free of 476.3 GB, D: 27.1 GB free of 0.9 TB, G: 15.7 GB free of 110.0 GB). It also displays BIOS information (YMMW001196, FUJITSU // Insyde Software Corp., FUJ - 10E0000).

The view is divided into three parts:

- **Toolbar** - showing buttons to perform quick operations like adding, editing, and removing

entries.

- **Main grid** - showing the list of all saved devices (physical and virtual).
- **Sidebar** - showing the details of the current selection, including inventory data if available.

Columns

The grid supports a predefined set of columns, only some of which are visible by default. It is possible to select more columns by pressing the **Column chooser** button in the grid toolbar. The following columns are available:

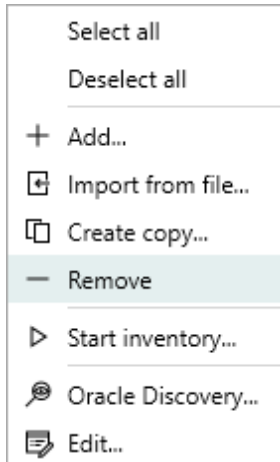
- **Type** - The device family (Windows / Unix / Unknown). This determines the inventory methods available for each device.
- **Hostname** - Either the DNS hostname of a device or it is empty if no name has been configured (in that case the IP address will be non-empty).
- **IP Address** - Either the IP address of a device or it is empty if the IP address is resolved automatically from the DNS name.
- **Status** - The inventory status. There are three possible values:
 - n/a - The device has not been inventoried yet
 - OK - The device has been inventoried and returned some results
 - In any other case, the column **Status** contains a short description of the most recent issue
- **Discovery source** - A text value determining the import source (for example *Discovery AD-importor* or *Discovery ping-sweep*). Manually added devices have an empty value in this column.
- **Last inventory attempt** - The last time the inventory has been started (whether successful or not).
- **Credentials** - The logical names of credentials used when scanning this device.
- **Capabilities** - The short version of enumeration of allowed inventory capabilities for this device. The string consists of two-letter tokens, with the following meaning:
 - *ZT*= Zero-Touch
 - *RE*= Remote-Execution
 - *FS*= Access to File System
 - *SMB*= Upload to SMB Shares
 - *WMI*= Windows Management Instrumentation (WMI) Queries
 - *WSM*= Windows Service Manager
- **Last inventory attempt (time)** - The last time the inventory has been started (whether successful or not) (contains full date and time).
- **Last inventory** - The last time the inventory has been successfully started (only date part).
- **Last inventory (time)** - The last time the inventory has been successfully started (contains full date and time).
- **Last successful inventory method** - The name of the last inventory method that worked for this device. This method will be preferred for future scans.
- **Last failed inventory methods** - The names of the inventory methods that failed the last time the device was scanned.
- **Last failed inventory method (details)** - Details about the methods that failed when the device was scanned for the last time.
- **Show inventory** - Shows the details of the inventory (only available if an inventory has been

already performed).

- **Created** - The date when the device was created or imported.

Context Menu

Pressing the Right-Mouse-Button after highlighting an item opens a context menu for it.



- **Select all** - Selects all visible entries in the grid.
- **Deselect all** - Deselects all visible entries in the grid.
- **Add...** - Opens the New Device Dialog.
- **Import from file...** - Opens the Import Wizard.
- **Create copy...** - Opens the New Device Dialog where the default values are automatically set to the values from the current selection.
- **Remove** - Removes the currently selected devices (see Removing Devices).
- **Start inventory...** - Opens the Inventory Wizard for the currently selected devices.
- **Oracle Discovery...** - Scans for the presence of Oracle instances on the currently selected devices.
- **Edit...** - Opens the Edit Device Dialog.



Note:

Some options may be conditionally hidden or disabled (for example the menu item **Edit...** is disabled if more than one device is selected).

Recent Scan Details

The details of the last scan are represented in two tabs **INVENTORY DETAILS** and **INVENTORY LOG**.

If the selected devices have not been scanned yet...

For devices that have not been scanned yet, both tabs show initially no data.

SELECTED DEVICE

MWS0191.raynet.corp

| | | |
|---------------|---------------------|----------------|
| Type | Hostname | IP address |
| Windows | MWS0191.raynet.corp | 172.16.210.116 |
| Created | Discovery source | Credentials |
| one month ago | Discovery AD-import | None |

Edit device...

Start inventory...

INVENTORY DETAILS

INVENTORY LOG

CUSTOM PROPERTIES

There is no data available yet. Perform inventory on this item to see the last inventory operation details.

[Run Inventory Wizard to get started](#)

Press **Run Inventory Wizard to get started** to open the Inventory Wizard, which will guide you through the process of collecting the data.

If the selected devices have already scanned...

For devices that have been scanned, the **INVENTORY DETAILS** show the details of the last successful scan.

SELECTED DEVICE

ubuntu



| | | |
|----------------------------|------------------------|------------------|
| Type UNIX* | Hostname ubuntu | Machine name |
| IP address 172.16.10.16 | Created 25 days ago | Discovery source |
| Credentials None | | |

Outdated Last inventory saved 2 months ago



Edit device...



Show full details...



INVENTORY DETAILS

INVENTORY LOG

CUSTOM PROPERTIES



Ubuntu 18.04
64-bit

| | |
|---|---|
| <p>Ubuntu 18.04</p> <p>64-bit</p> | <p>sda 0 B free of 20.0 GB</p> <p>sr0 0 B free of 1.0 GB</p> <p>/dev 0.9 GB free of 0.9 GB</p> |
| <p>IP4 127.0.0.1</p> <p>IP4 172.17.0.1</p> <p>IP4 172.25.0.1</p> <p>IP4 192.168.174.128</p> | <p>BIOS</p> <p>VMware-56 4d 5d 58 33 4d 7e 9e-3c 48 35 5f 15 f0 50 28</p> <p>VMware, Inc.</p> |
| <p>CPU</p> <p>1x Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz [1 core(s)]</p> | <p>Physical memory (RAM)</p> <p>2.0 GB</p> |

The exact content of the tab varies, depending on the target operating system. For many popular systems, RayVentory Scan Engine shows a logo and uses a dedicated key colour for an easy identification. Some basic details, including the list of IP addresses, CPU, BIOS, RAM are also displayed. The tiles will be stacked if the width of the sidebar is too small to fit them in two

columns.

INVENTORY LOG tab is a summary of recent successful and failed scans. Typically, once the inventory results are available, the view would have the following content:

| INVENTORY DETAILS INVENTORY LOG CUSTOM PROPERTIES | | | | |
|---|----------|------------------------------|-----------|--------------------|
| Name | Status | Message | Runnin... | Started |
| ame2017 | ✓ Su... | The device has been succe... | 00:00:26 | 4/9/2020 5:54:0... |
| Zero-Touch Wind... | ✓ Su... | Successfully scanned. | 00:00:26 | 4/9/2020 5:54:0... |
| AdminWin | ✓ Su... | Successfully scanned. | 00:00:25 | 4/9/2020 5:54:0... |
| Remote Execution... | ⏸ Ski... | Not required anymore. | | |
| Remote Execution... | ⏸ Ski... | Not required anymore. | | |

These details are specific to a concrete device from the current selection. The tree view uses several levels to distinguish between the following actions, that were executed during the inventory scan:

- Targeting a device (root entry),
- Used inventory method (for example Zero-Touch Windows),
- (Optionally) Sub-methods and variants of execution paths (for example using particular version of JRE),
- Used credentials (for example AdminWin). This is the display name from credential store, or a default string "Current user" if the default credentials were used.

For every entry, its date, duration time and status are shown. The items may have different statuses, and the logic that RayVentory Scan Engine applies when interpreting them varies. For example, if the first credentials from the list fail, the program tries the next ones available and so on until it finds matching ones. This is going to be represented as a series of sub-nodes from a particular method, with statuses indicating whether the credentials failed or succeeded. A similar logic is applied to the methods selection - RayVentory Scan Engine executes them from top to the bottom and stops on the first one that succeeded. Other methods are marked as **Not required anymore** if any of previous method returned the expected results.

If a method failed due to a non-critical error (for example closed port, missing credentials or access denied) the method result is also shown, which simplifies troubleshooting. For example, the following inventory log:

| Name | Status | Message | Runni... | Started |
|------------------------|-------------------|--|-----------|--------------------|
| ame2017 | Connection failed | We could not connect to this device. | 00:01:... | 6/5/2020 10:50:... |
| Zero-Touch Windows... | Connection failed | Connection to WMI failed. Port 135 se... | 00:00:... | 6/5/2020 10:50:... |
| Remote Execution Wi... | Skipped | Skipped due to a dependency on a ser... | | |
| Remote Execution Wi... | Connection failed | Connection to SMB failed. Cannot con... | 00:00:... | 6/5/2020 10:51:... |
| AdminWin | Connection failed | Connection to SMB failed. Cannot con... | 00:00:... | 6/5/2020 10:51:... |

Can be interpreted as it follows:

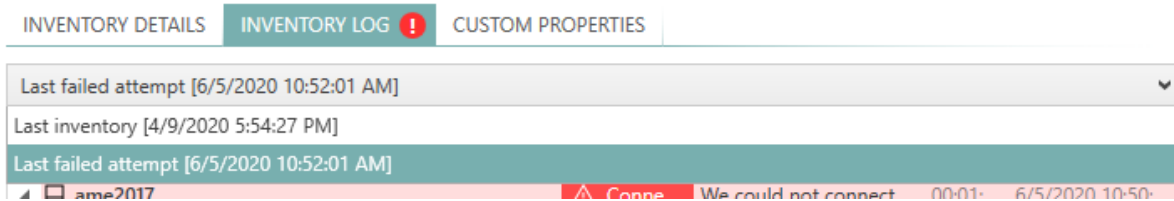
- RayVentory Scan Engine executed an inventory scan of machine **RVP10** and it failed using all available methods.
- The following methods were used: Zero-Touch Windows, Remote Execution Windows [WMI/SMB], Remote Execution Windows [ServiceManager/SMB].
- The Zero-Touch scan on Windows failed due to port 135 being closed. Since Zero-Touch on Windows relies on WMI, not being able to communicate with the machine on that port lead to an error for that particular method.
- The Remote-Execution Windows [WMI/SMB] was not executed at all. RayVentory Scan Engine was aware that a similar method using WMI failed, and it implied that since WMI was not possible in the previous step, there was no necessity to perform it again. Thus, the method never executed and reported the status **Skipped**, and the message indicates that the reason was indeed the WMI check which had failed previously.
- The last method executed, and it start probing the credentials store. It picked first the best-match **Windows**. After running for a few seconds, it also failed with the message that `ADMIN$` share could not be accessed, which indicates unavailability of the machine. Since no more credentials were available, it stopped and returned the error back to the caller.
- Overall, since no method returned a positive result, RayVentory Scan Engine summarized the results and picked the status **Connection failed** as the best one that describes the issue. IT Pro would be now able to address the issue by accessing the machine (ensuring it is up and running) and verifying whether the port 135 is open. After that, the scan should be repeated.

Working with the last positive and failed result

If the last result of the inventory scan was positive, only its details are shown in the sidebar. However, consider the following scenario:

- The machine has been scanned successfully at least once,
- But due to configuration changes it is not available anymore, and all new inventory results report various connection issues.

In this case, RayVentory Scan Engine shows both the last failed and positive result of scan. You can determine if this is the case by observing the content of the **INVENTORY LOG** tab:

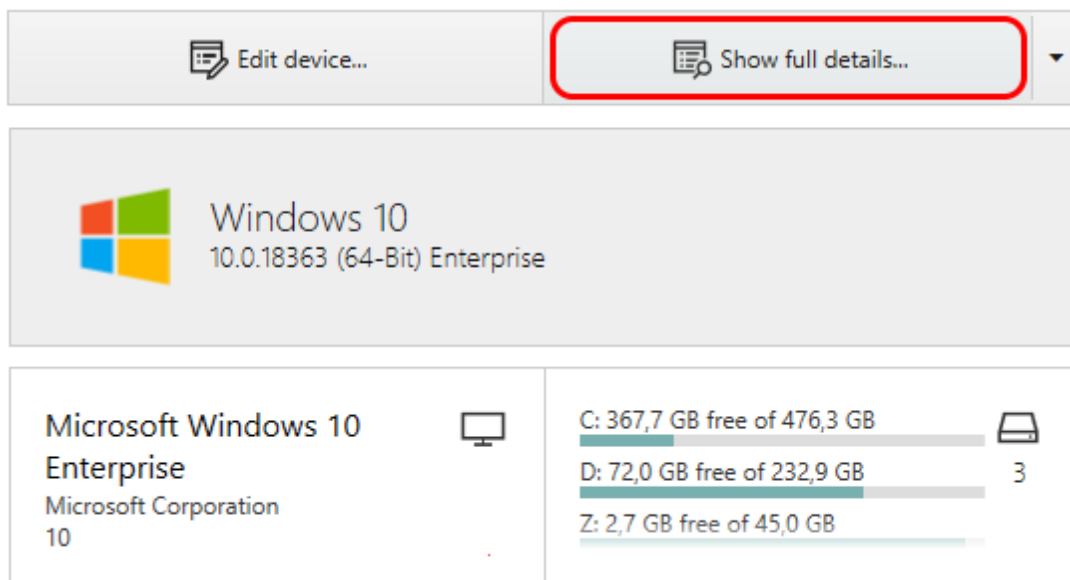


The exclamation mark drawn next to the tab indicates, that the last scan failed. You will then be able to use the drop-down menu below to select which results are to be analyzed. By default, the most recent log is shown (this provides on the other hand a quick overview of the relevant data, because usually it is the last failed scan that is most important when troubleshooting connection or permission issues).




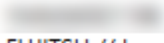



There is still a way to see the previous success result, which will give some additional information, including which method succeeded for the last time, when the last time was and which credentials were used at that time. This provided a valuable information, indicating whether the change in machine availability has been caused by changes in the configuration, password store, firewall and so.

Viewing Inventory Details

Once a device has been successfully scanned for its software and hardware, a button will be shown on the sidebar allowing the user to show the content of a specified machine:



The button can be pressed to open a detailed overview of the machine software and hardware. For convenience, the summary of the machine asset is also shown directly in the sidebar:

| | |
|---|--|
| <p>Microsoft Windows 10 Enterprise Microsoft Corporation 10.0.17134</p>  | <p>C: 367,7 GB free of 476,3 GB </p> <p>D: 72,0 GB free of 232,9 GB 3</p> <p>Z: 2,7 GB free of 45,0 GB</p> |
| <p>IP4 192.168.120.147 IP6 fe80::680c:ebd:e82e:41d9 IP4 192.168.254.1 IP6 fe80::7090:b2ff:f04e:70c6</p>  6 | <p> FUJITSU // Insyde Software Corp. FUJ - 10A0000 </p> |
| <p>1x Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz [6 core(s)]  1</p> | <p>UUID </p> |

For popular operating systems and distributions, RayVentory Scan Engine is able to show additional details and product logo, for example:

The details inventory overview contains several more tabs, which are contextually sensitive and display various information depending on the connection type.

Some most-used tabs are:

- SOFTWARE
- HARDWARE
- SERVICES
- SERVER FEATURES
- DOCKER
- RAW DATA

Undocking the Inventory View

It is possible to "undock" the view to a separate window which can be freely moved and stacked with other windows. In order to do that, press the little **undock** button next to the **CLOSE** button of the inventory overlay.

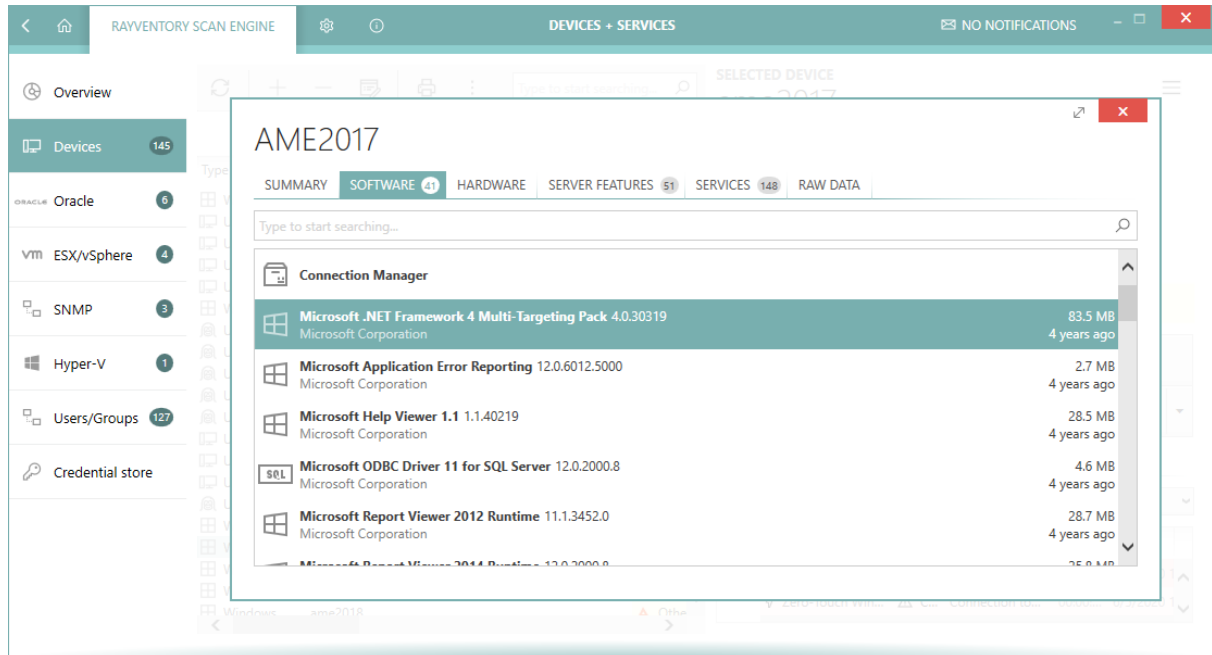


Note:

After undocking, the window cannot be docked in the main window anymore.

Software

The **SOFTWARE** tab contains an aggregated information about the software and packages, which were detected during the last inventory scan.



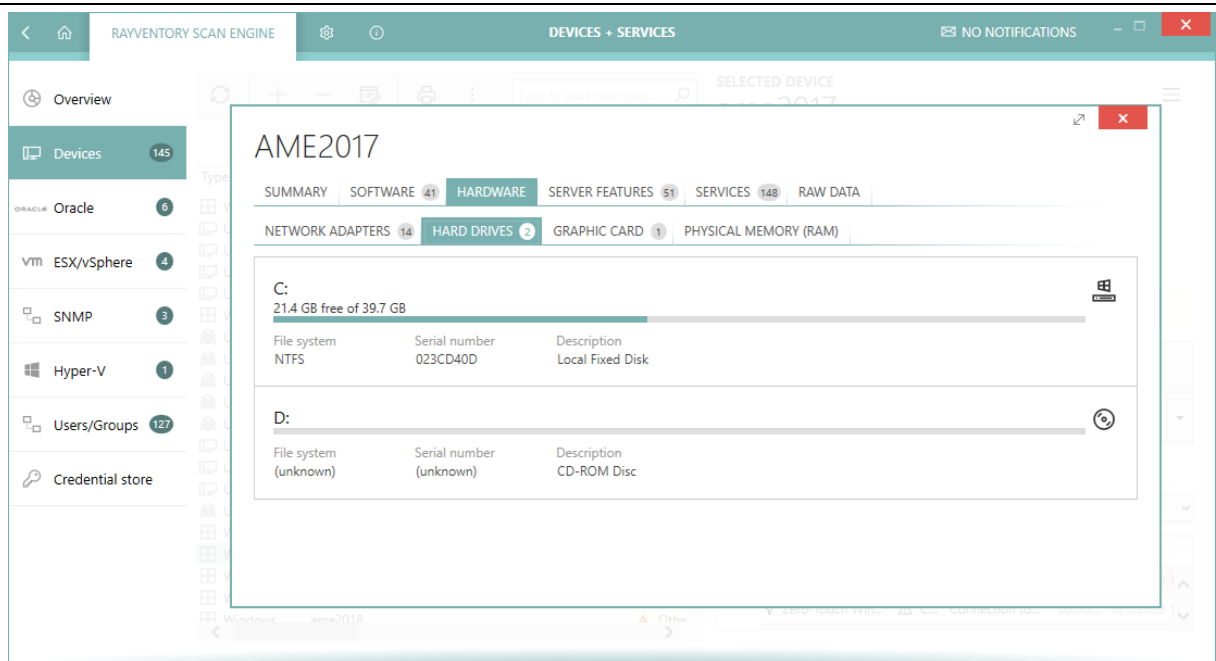
Depending on the target system, this view may differ and show various information, applicable to its source.

- **Windows systems:** This is a merged view of the ARP information (Add/Remove Programs registry keys) and the MSI evidence (Windows Installer Database). The duplicates are eliminated. Some products (for example SQL Server, Java, Visual Studio etc.) may already have dedicated icons. Estimated size, the actual version and installation date are shown, based on Windows evidence (usually ARP registry entries).
- **Non-Windows systems:** Usually only the data from package managers is shown, depending on the actual distribution being scanned. Due to differences between data reported back by various systems, the information here may vary. For most of supported package managers, at least name, publisher, size and versions are displayed.

You can use the built-in search box to filter the list and find the application of interest.

Hardware

This tab shows the list of recognized hardware features of the target device.

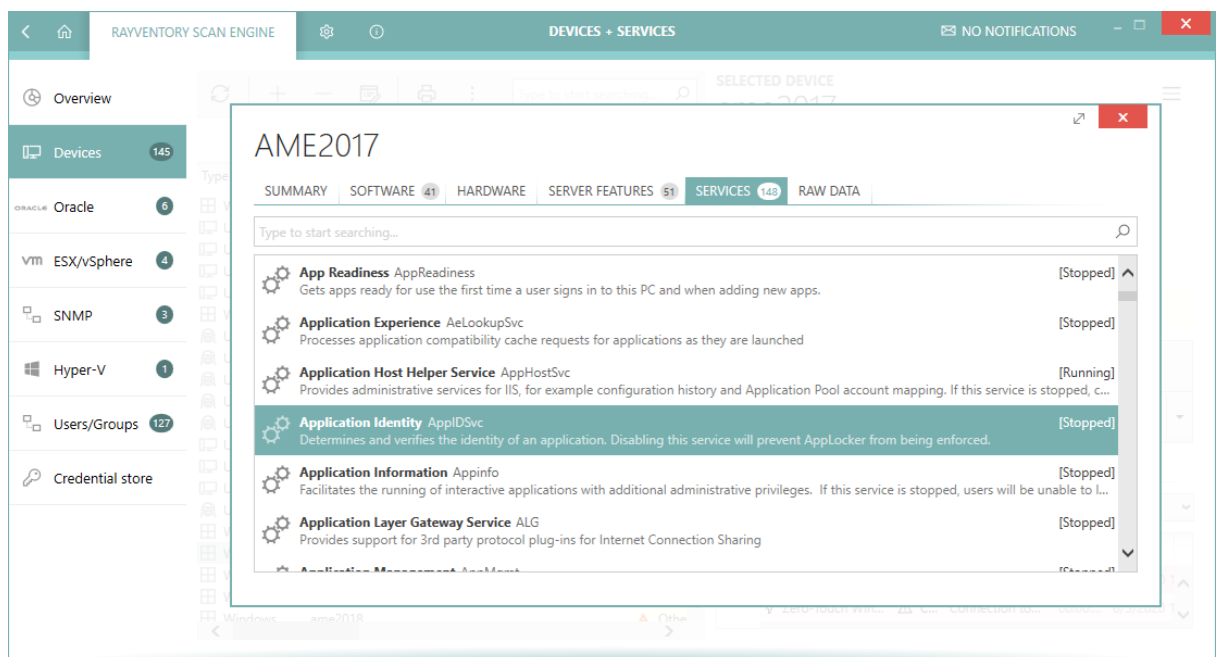


For most of supported devices, the following information should be available:

- The list of network adapters and their MAC addresses + IP addresses,
- The list of hardware drives, partitions, and mount points,
- The information about the graphic card,
- The amount of physical memory (RAM).

Services

This tab shows the list of Windows services which were present on the machine at the time of the last inventory scan. Both started and stopped services are included.



You can use the built-in search box to filter the list and find the service of interest.

Server features

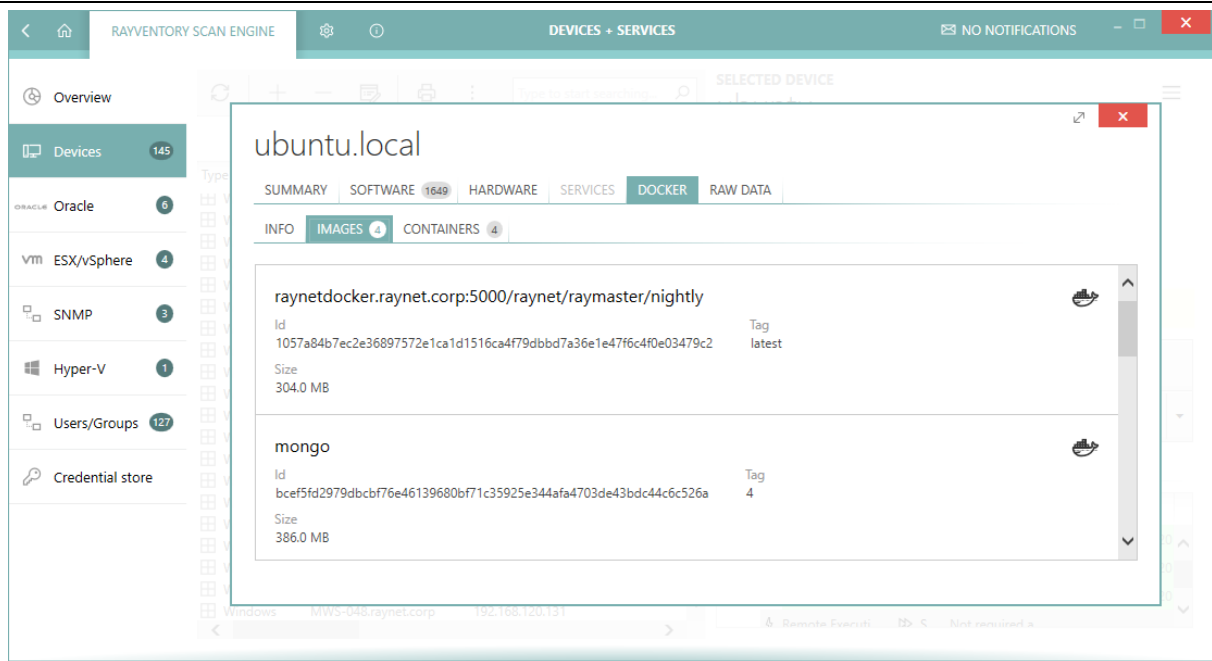
This tab shows the list of Windows Server features which were enabled at the time of the scan.



You can use the built-in search box to filter the list and find the feature of interest.

Docker

The **DOCKER** tab contains an aggregated information about the available images, containers and meta data of a Docker instance, found on the particular device.



- **INFO**
This is the place where client information is shown, for example the type of docker engine, its version and the architecture.
- **IMAGES**
Locally available images are listed in this tab. For each image, the name, SHA-256 ID, size and tag are displayed.
- **CONTAINERS**
This shows the list of containers, both running and stopped. Each container is identifier by the image it was run from, an entry command, exposed ports and information about whether it was running at the time of the scan.



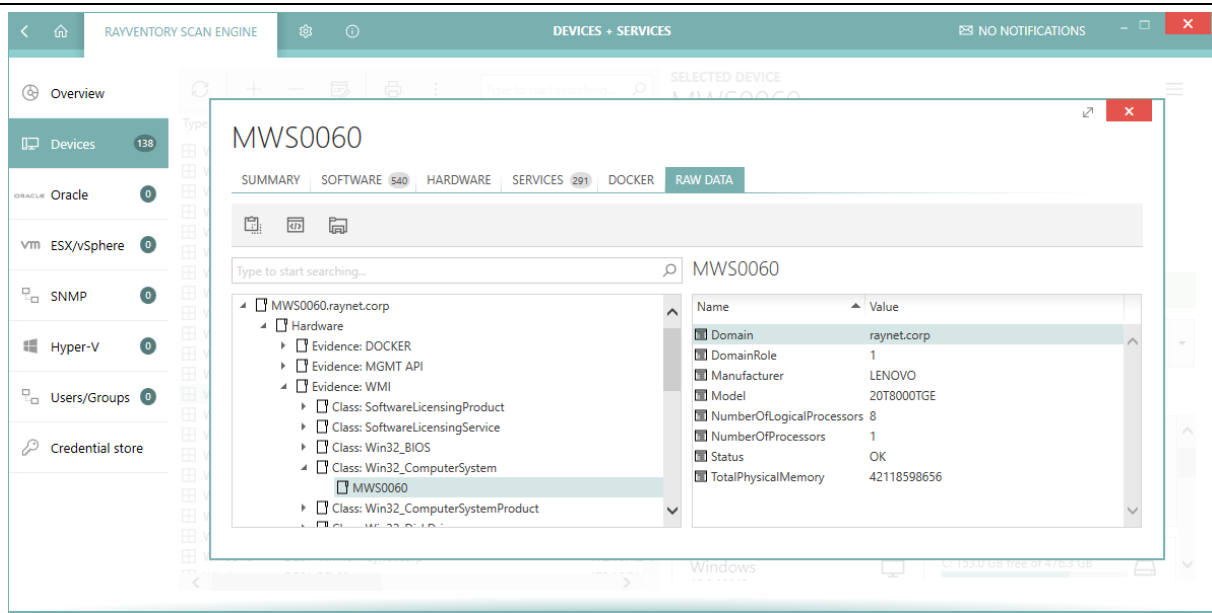
Note:

This and much more information about docker entities can be found in the RAW view.

Raw data

Working with Raw Data

IT professionals and administrators having experience with the data structures of the RayVentory Scan Engine object domain can also work directly with the underlying inventory files (NDI).



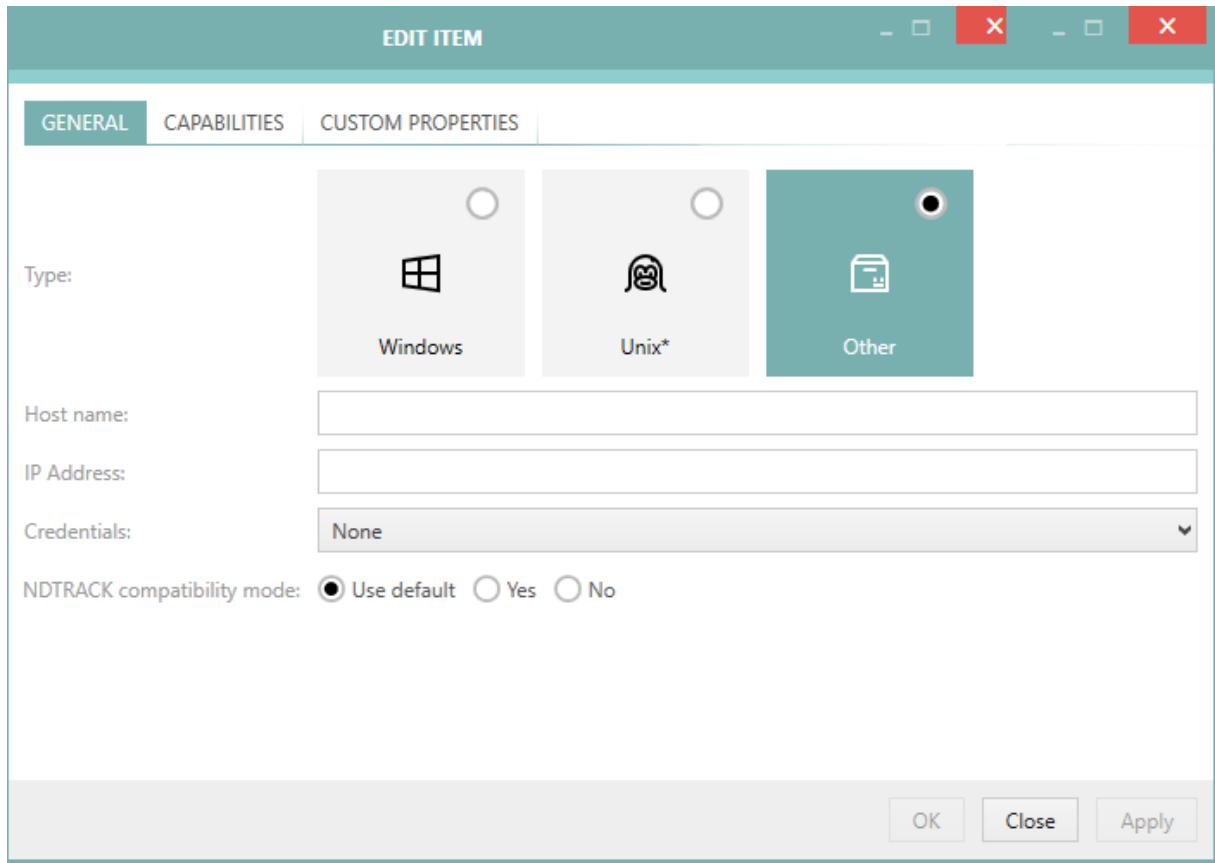
To see the raw content:

1. In the inventory overview select the last tab which is called **RAW DATA**.
2. The tree shows the logical structure of the underlying XML data contained within the **.ndi** file.
3. The trees can be expanded in order to reach the node containing the required content.
4. The three buttons in the toolbar have the following meaning:
 - Copy the full path to the **.ndi** file to the clipboard.
 - Open the **.ndi** file in the default editor.
 - Open Windows Explorer and highlight the **.ndi** file.

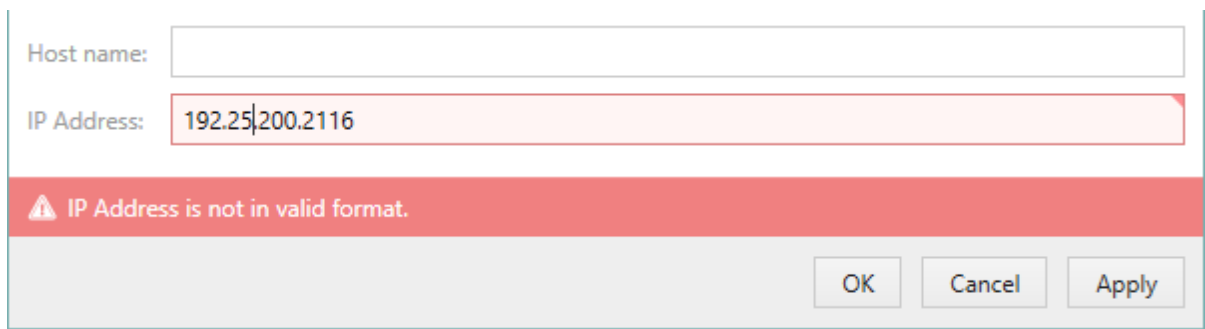
Adding a New Device

In Order to Add a New Device

1. Press the **+** button in the top toolbar or click the **Add..** menu item from the context menu.
2. A new empty dialog will be shown.



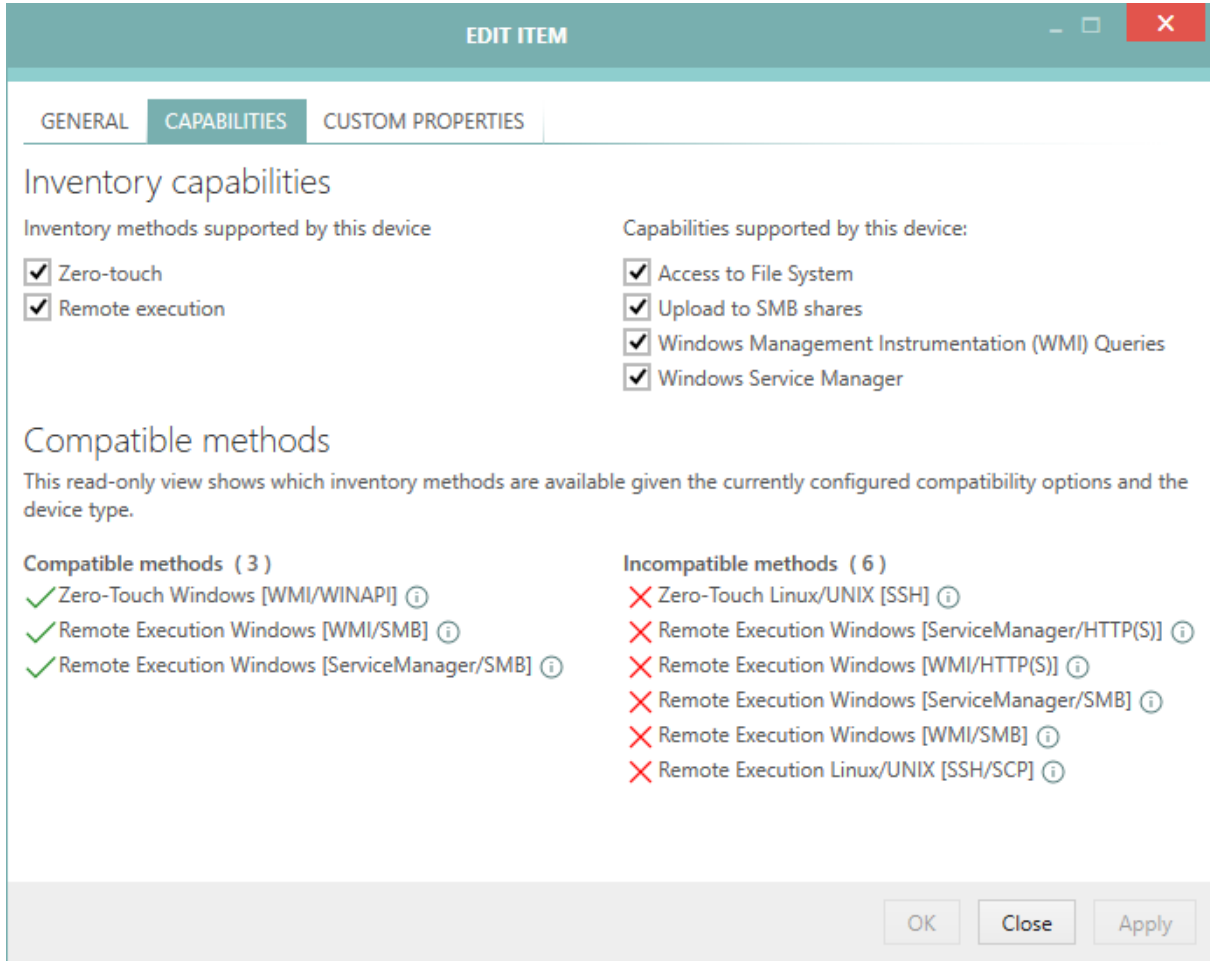
3. At minimum, specify the device DNS name and / or IP address. You can also select the device family type (Windows / Unix / other) which will be respected by RayVentory Scan Engine later during the discovery step.
4. Optionally, it is possible to select the preferred credentials used by this device. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.
5. In the **CAPABILITIES** tab, the capabilities of the newly added device can be limited.
6. In the **CUSTOM PROPERTIES** tab the device-specific attributes can be configured.
7. Press **OK** to accept the changes and close the window or **Apply** to save them immediately.
8. If any mandatory field is not specified or is in the wrong format a validation error is shown:



Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

Device Capabilities

The **CAPABILITIES** section allows users to precisely configure which low-level capabilities each device supports.



EDIT ITEM

GENERAL **CAPABILITIES** CUSTOM PROPERTIES

Inventory capabilities

Inventory methods supported by this device

- ☒ Zero-touch
- ☒ Remote execution

Capabilities supported by this device:

- ☒ Access to File System
- ☒ Upload to SMB shares
- ☒ Windows Management Instrumentation (WMI) Queries
- ☒ Windows Service Manager

Compatible methods

This read-only view shows which inventory methods are available given the currently configured compatibility options and the device type.

Compatible methods (3)

- ✓ Zero-Touch Windows [WMI/WINAPI] ⓘ
- ✓ Remote Execution Windows [WMI/SMB] ⓘ
- ✓ Remote Execution Windows [ServiceManager/SMB] ⓘ

Incompatible methods (6)

- ✗ Zero-Touch Linux/UNIX [SSH] ⓘ
- ✗ Remote Execution Windows [ServiceManager/HTTP(S)] ⓘ
- ✗ Remote Execution Windows [WMI/HTTP(S)] ⓘ
- ✗ Remote Execution Windows [ServiceManager/SMB] ⓘ
- ✗ Remote Execution Windows [WMI/SMB] ⓘ
- ✗ Remote Execution Linux/UNIX [SSH/SCP] ⓘ

OK Close Apply

The combination of supported capabilities is used by the **Inventory** Wizard to determine which methods are compatible with which target. For example, unchecking the Zero-Touch option for a device will render it scannable by remote execution methods only.

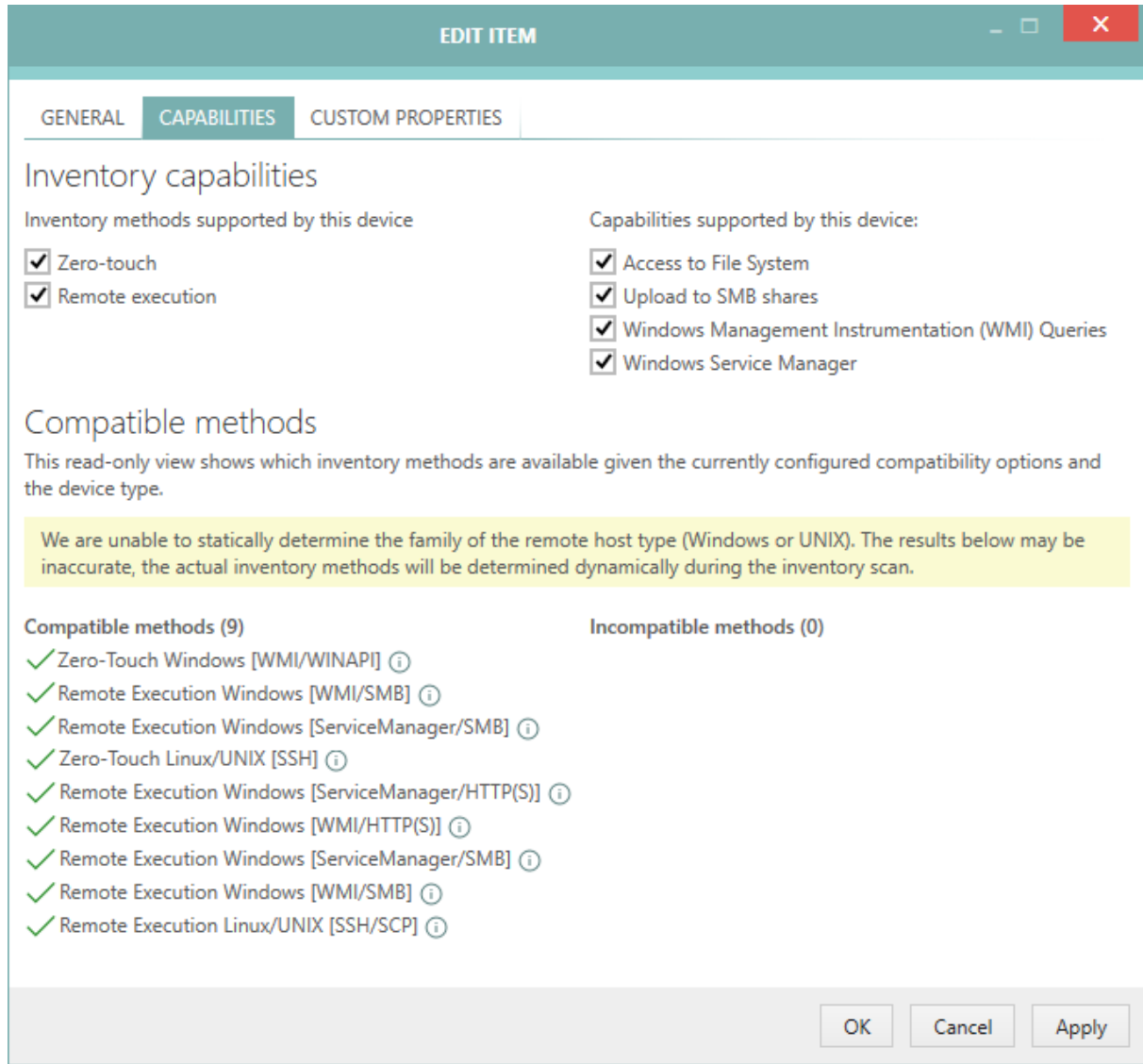
The tab consists of three panels:

- The selection of general inventory methods supported by this device (Zero-Touch and / or Remote-Execution).
- More low-level capabilities which determine whether particular features should be supported by this device.
- The read-only view showing in real-time which inventory methods are applicable given the current state of the device.

The Zero-Touch execution does not require any additional capabilities other than certain RayVentory Scan Engine settings. On the other hand, the Remote-Execution may have different

prerequisites depending on the method type. By ticking and unticking the checkboxes next to the methods, the lower view is refreshed and shows dynamically which methods are applicable for a given device. By hovering a mouse over the info icon next to a name, a detailed information is shown explaining which factors affect the static availability of the current inventory method.

For some devices, the view may not be shown accurately. For example, if the device type is **Unknown**, RayVentory Scan Engine is unable to determine which set of methods (Windows- or Unix-based) should be used. In this case, they are all shown as compatible methods, but a warning like below may be shown:



EDIT ITEM

GENERAL **CAPABILITIES** CUSTOM PROPERTIES

Inventory capabilities

Inventory methods supported by this device

- ☒ Zero-touch
- ☒ Remote execution

Capabilities supported by this device:

- ☒ Access to File System
- ☒ Upload to SMB shares
- ☒ Windows Management Instrumentation (WMI) Queries
- ☒ Windows Service Manager

Compatible methods

This read-only view shows which inventory methods are available given the currently configured compatibility options and the device type.

We are unable to statically determine the family of the remote host type (Windows or UNIX). The results below may be inaccurate, the actual inventory methods will be determined dynamically during the inventory scan.

Compatible methods (9)

- ✓ Zero-Touch Windows [WMI/WINAPI] ⓘ
- ✓ Remote Execution Windows [WMI/SMB] ⓘ
- ✓ Remote Execution Windows [ServiceManager/SMB] ⓘ
- ✓ Zero-Touch Linux/UNIX [SSH] ⓘ
- ✓ Remote Execution Windows [ServiceManager/HTTP(S)] ⓘ
- ✓ Remote Execution Windows [WMI/HTTP(S)] ⓘ
- ✓ Remote Execution Windows [ServiceManager/SMB] ⓘ
- ✓ Remote Execution Windows [WMI/SMB] ⓘ
- ✓ Remote Execution Linux/UNIX [SSH/SCP] ⓘ

Incompatible methods (0)

OK Cancel Apply

The options set here affect which methods are available later on when doing inventory scanning. Refer to the advanced topic [Inventory Methods Overview](#) to find out about the prerequisites and required settings.

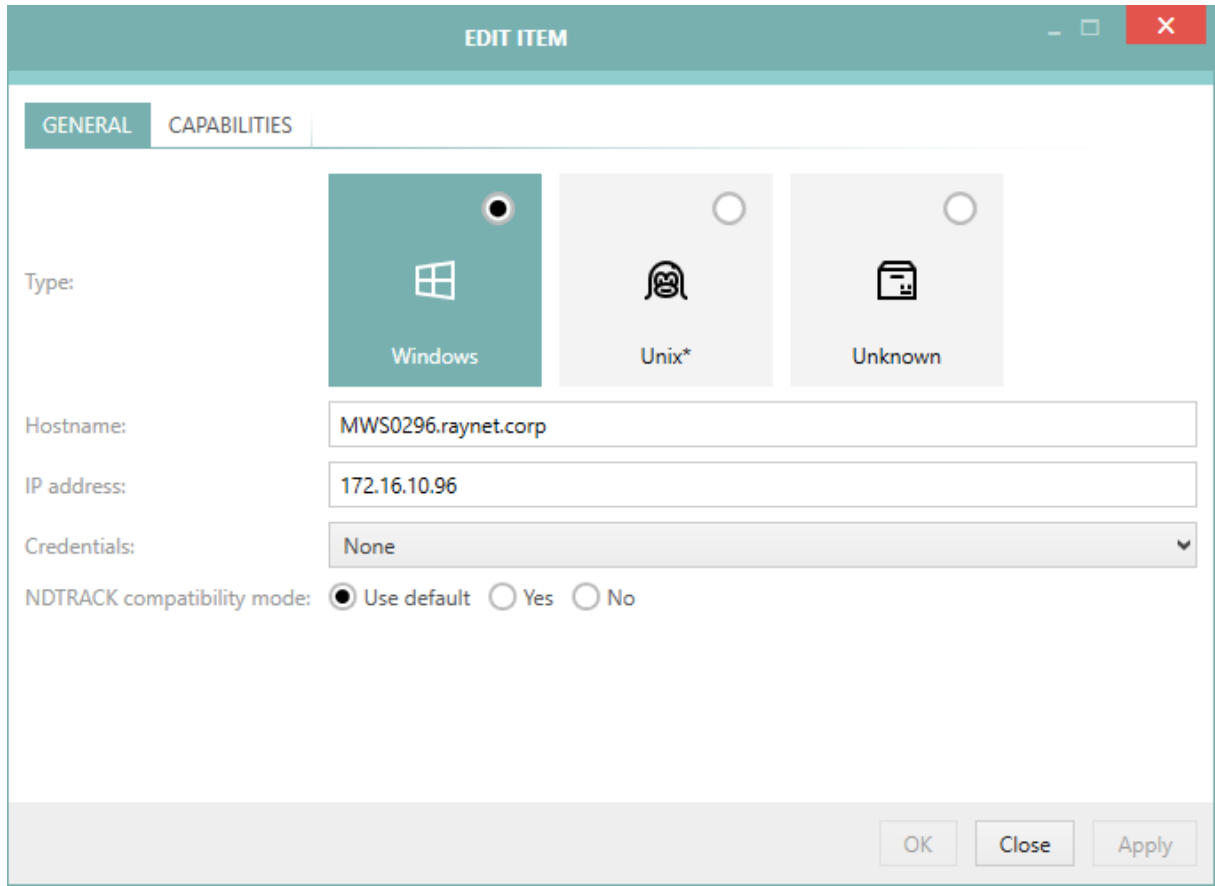
Preventing a Device from Being Scanned

You can opt-out for any further scans of the device, thus preserving its current inventory state. To do that, disable the Zero-Touch and the Remote-Execution inventory. The list that is shown underneath these options should reflect this by saying that currently there is no compatible method. This effectively means, that when the user executes an inventory job (from the Inventory wizard, PowerShell command let or from a scheduled task), the device is never scanned and its current inventory files and details are not affected by the new scan.

Editing Devices

In Order to Edit a Device

1. Highlight an entry in the list and press the **Edit selected...** button, the **Add..** menu item from the context menu, or the **Edit device...** button from the sidebar.
2. A new dialog will be shown with the details of the current selection:



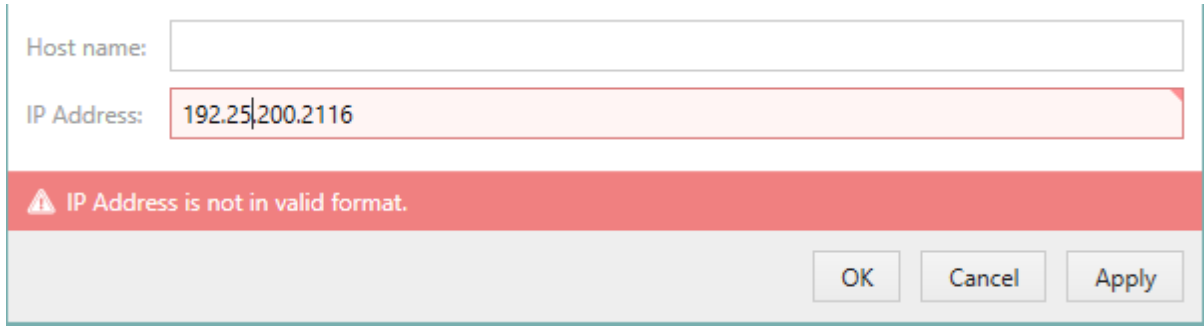
The screenshot shows the 'EDIT ITEM' dialog box. It has a title bar with standard window controls. Below the title bar are two tabs: 'GENERAL' and 'CAPABILITIES'. The 'GENERAL' tab is active. It contains the following fields:

- Type:** Three radio buttons are shown: 'Windows' (selected), 'Unix*', and 'Unknown'.
- Hostname:** A text input field containing 'MWS0296.raynet.corp'.
- IP address:** A text input field containing '172.16.10.96'.
- Credentials:** A dropdown menu showing 'None'.
- NDTRACK compatibility mode:** Three radio buttons: 'Use default' (selected), 'Yes', and 'No'.

At the bottom right of the dialog are three buttons: 'OK', 'Close', and 'Apply'.

3. Before the device can be saved, at minimum the device DNS name and / or IP address must be specified. You can also select the device family type (Windows / Unix / other) which will be respected by RayVentory Scan Engine later at the discovery step.
4. You can optionally select preferred credentials used by this device. If you leave this empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.

5. In the **CAPABILITIES** tab, you can also limit the capabilities of the edited device.
6. In the **CUSTOM PROPERTIES** tab the device-specific attributes can be configured.
7. Press **OK** to save the change and close the window or **Apply** to immediately save them.
8. If any mandatory field is not specified or is in the wrong format, a validation error is shown:



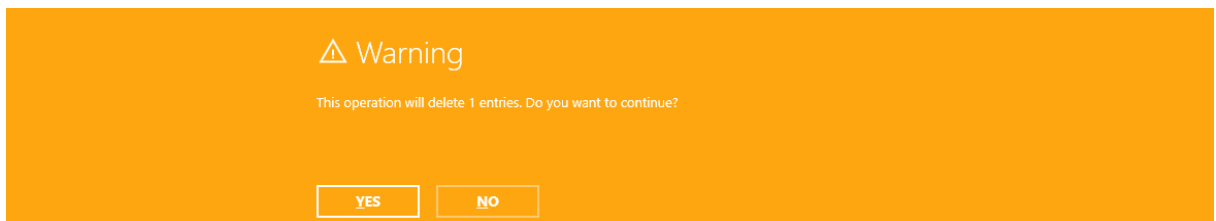
The screenshot shows a configuration window with two input fields: "Host name:" and "IP Address:". The "IP Address:" field contains the text "192.25|200.2116" and is highlighted with a red border. Below the fields, a red error bar displays the message "IP Address is not in valid format." with a warning icon. At the bottom right, there are three buttons: "OK", "Cancel", and "Apply".

Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

Removing Devices

In Order to Remove a Device

1. Highlight an entry in the list and press the - button or click the **Remove...** menu item from the context menu.
2. Confirm the deletion by pressing **YES** in the confirmation dialog.



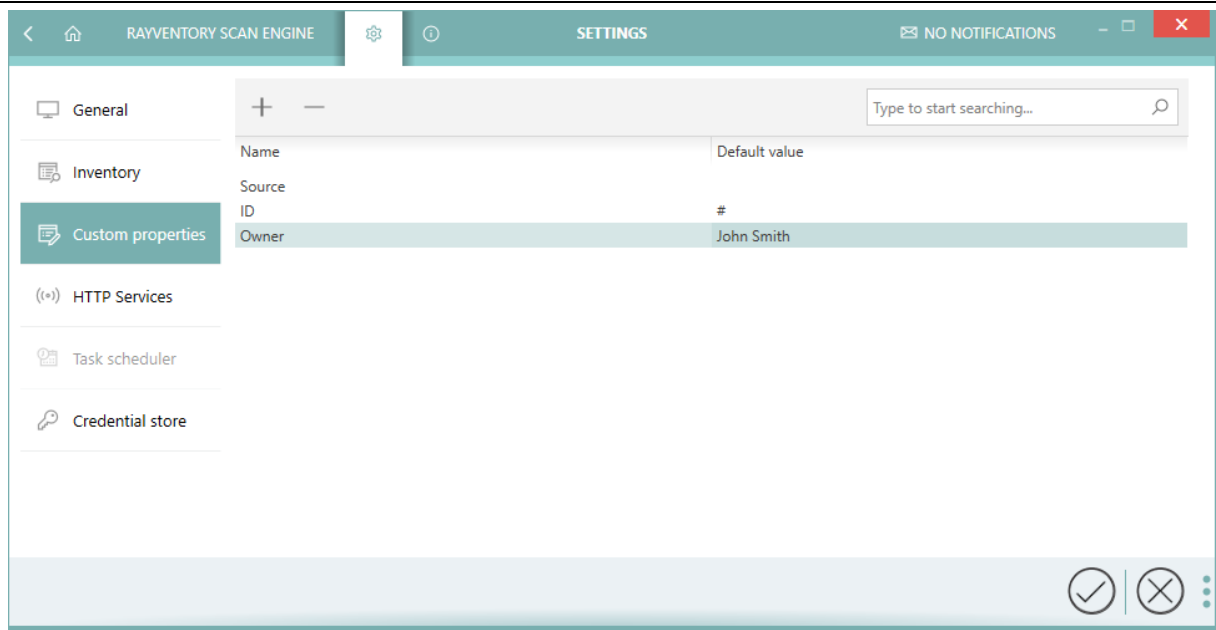
Note:

This operation is irreversible. Any existing inventory files which were assigned to the device will stay, though.

Working with Custom Attributes

Defining custom attributes

Custom attributes can be managed from the **Settings screen**, tab **Custom attributes**.



Each custom attribute consists of two fields:

- **Name**

This is the name that will be used for value headers. When combined with NDI Upload, the custom attributes are written using the name as the key. This field is required and must be unique.

- **Default value**

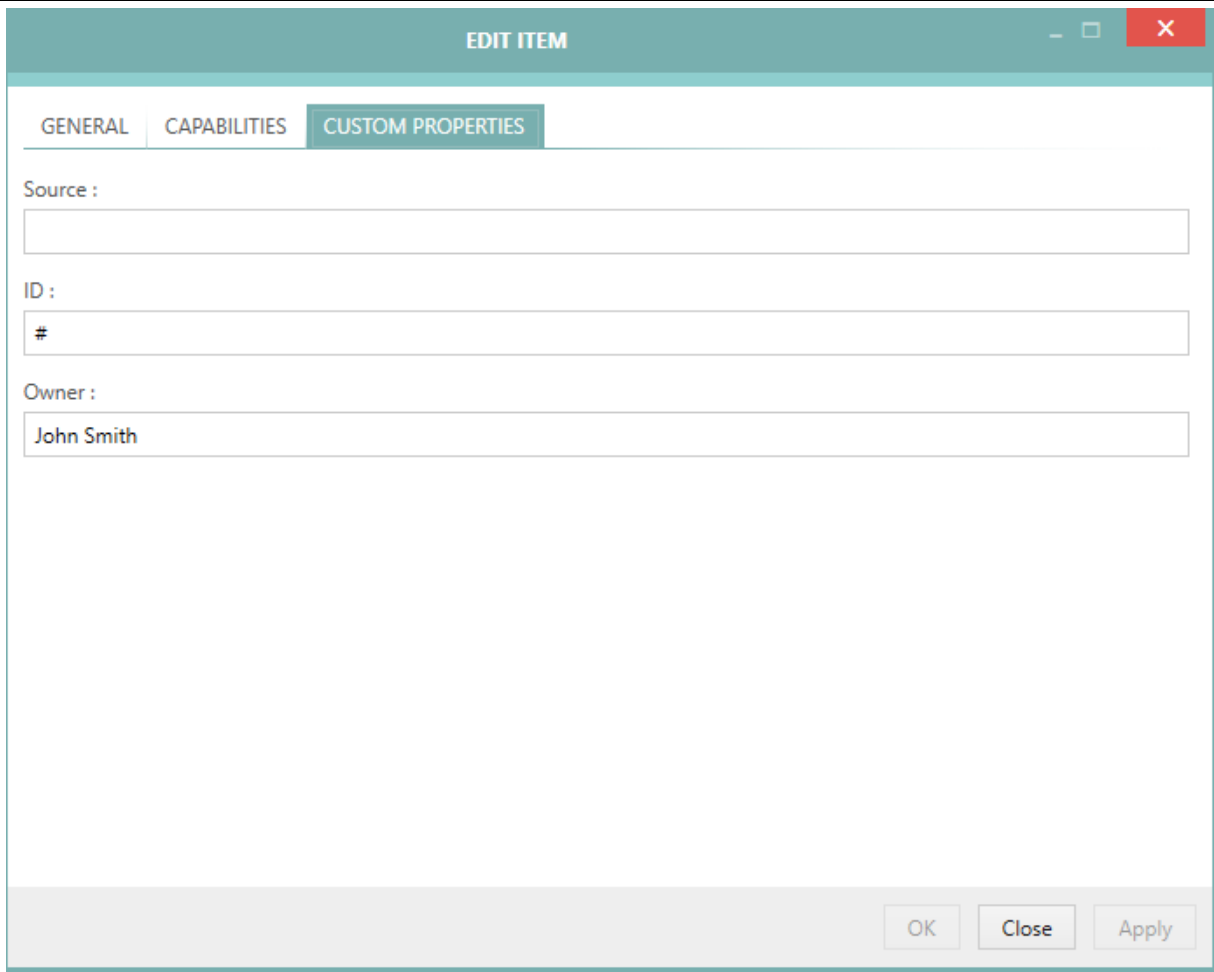
This is the default value used for attributes, for which the user did not provide anything yet. This field can be left empty to indicate that a simple empty string should be used as a default value.

The toolbar contains function buttons: + to add a new custom attribute, and - to remove the currently selected one.

Editing custom attributes for devices

Once a set of custom attributes is defined, it is possible to define the actual values on a device level.

The editor is available in the **Edit device dialog**, tab **Custom attributes**.



The screenshot shows a window titled "EDIT ITEM" with a teal header bar. Below the header are three tabs: "GENERAL", "CAPABILITIES", and "CUSTOM PROPERTIES", with the last one being active. The form contains three labeled text input fields: "Source :" (empty), "ID :" (containing "#"), and "Owner :" (containing "John Smith"). At the bottom right, there are three buttons: "OK", "Close", and "Apply".

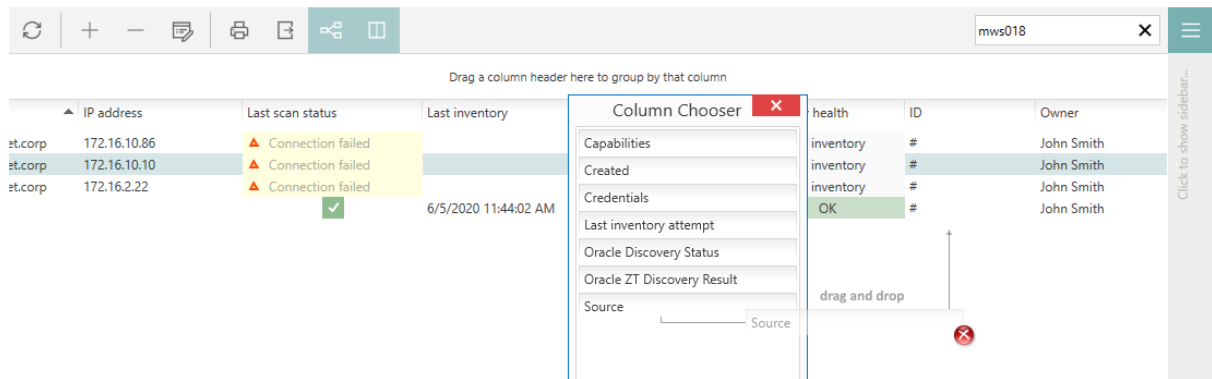
The number of fields and their names will be different, based on your current configuration.

If a default value is defined, for every device without a value configured explicitly by the user, the default one will be shown instead.

All custom properties are optional.

Displaying and data-shaping

The device grid supports displaying custom attributes in a tabular way.



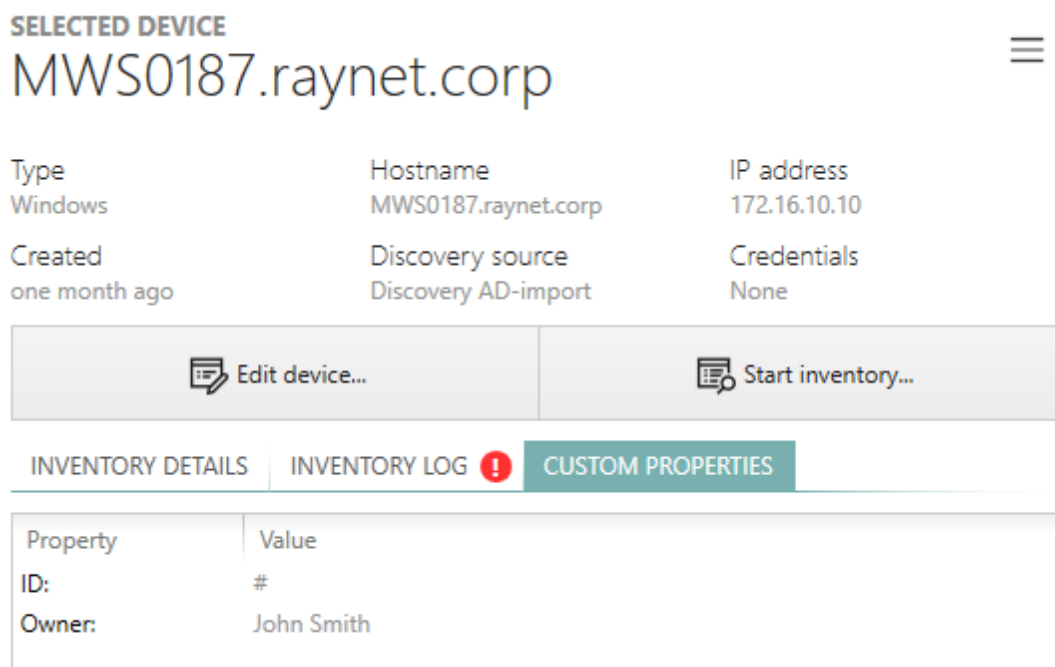
The screenshot shows the RAYVENTORY interface with a table of devices. The table has columns for IP address, Last scan status, and Last inventory. A 'Column Chooser' dialog box is open, allowing users to select columns to display. The dialog box lists various attributes such as Capabilities, Created, Credentials, Last inventory attempt, Oracle Discovery Status, Oracle ZT Discovery Result, and Source. A 'drag and drop' instruction is visible in the dialog box.

| IP address | Last scan status | Last inventory |
|----------------------|-------------------|----------------------|
| st.corp 172.16.10.86 | Connection failed | |
| st.corp 172.16.10.10 | Connection failed | |
| st.corp 172.16.2.22 | Connection failed | 6/5/2020 11:44:02 AM |

The attributes should be by default visible in the grid, each having a separate column. You can order, filter and search them using the same way as all other predefined columns.

To hide the custom column, drop it to the column chooser. In order to show the column again, drag it from the column chooser into the required place and release the mouse button to drop it.

Custom properties are also displayed in the device sidebar:

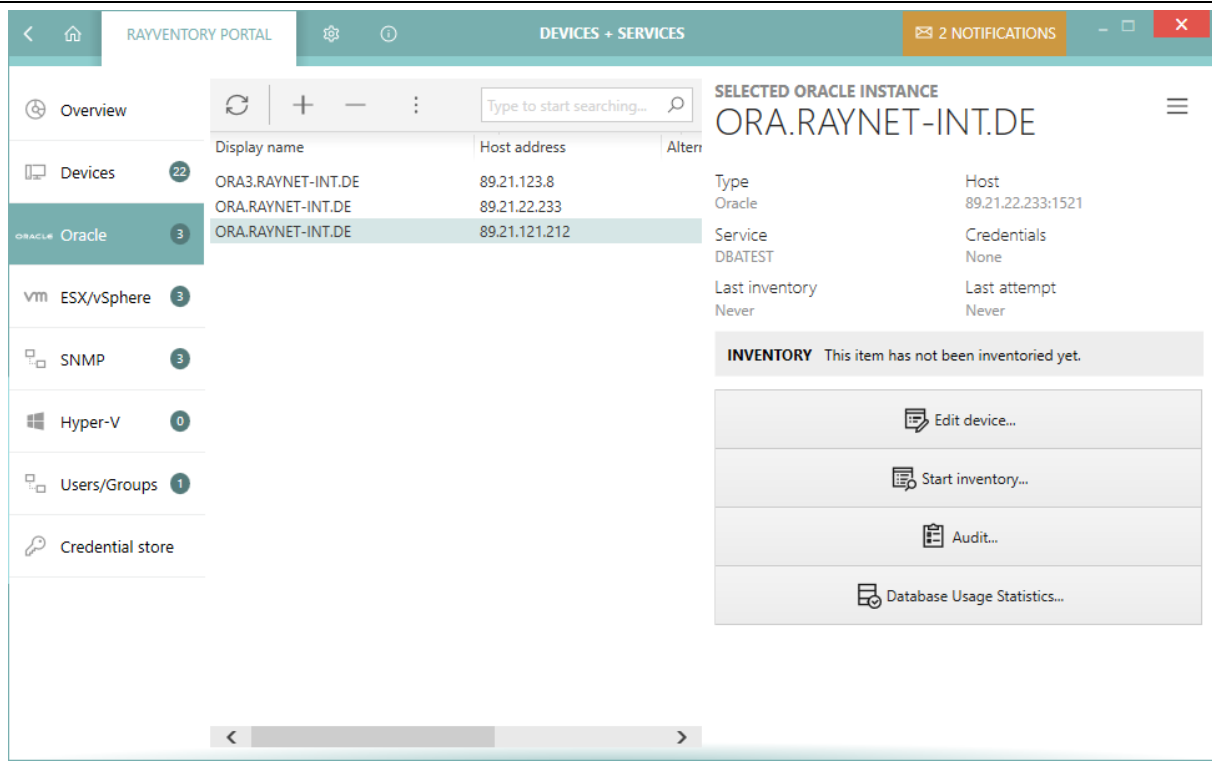


The screenshot shows the 'SELECTED DEVICE' sidebar for MWS0187.raynet.corp. The sidebar displays various device details and actions. The 'CUSTOM PROPERTIES' tab is selected, showing a table of properties and values.

| Property | Value |
|----------|------------|
| ID: | # |
| Owner: | John Smith |

Oracle

This view aggregates the discovery and inventory data of your Oracle databases.



The view is divided into three parts:

- **Toolbar** - showing buttons to perform quick operations like adding, editing, and removing entries.
- **Main grid** - showing the list of all saved devices (physical and virtual).
- **Sidebar** - showing the details of the current selection including inventory data if available.

Columns

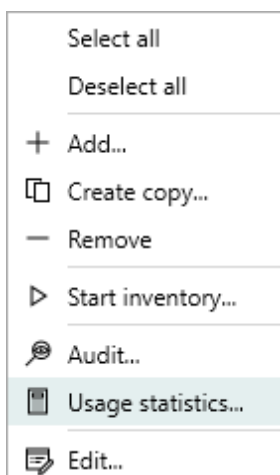
The grid supports a predefined set of columns. Out of these columns a handful is visible by default. It is possible to select more columns by pressing the **Column chooser** button in the grid toolbar. The following columns are available:

- **Type** - A static column with a connection icon (Oracle).
- **Host name** - The DNS hostname or IP address of the connection.
- **Service** - The name of the service.
- **Port** - The port used to communicate with the service.
- **Status** - The inventory status. There are 3 possible values:
 - n/a - The device has not been inventoried yet.
 - OK - The device has been inventoried and returned some results.
 - In any other case, the column **Status** contains a short description of the most recent issue.
- **Last inventory attempt** - The last time the inventory has been started (whether successful or not).

- **Credentials** - The logical names of credentials used when scanning this device.
- **Capabilities** - The short version of enumeration of allowed inventory capabilities for this device. The string consists of two-letter tokens, with the following meaning:
 - *ZT*= Zero-Touch
 - *RE*= Remote-Execution
 - *FS*= Access to File System
 - *SMB*= Upload to SMB Shares
 - *WMI*= Windows Management Instrumentation (WMI) Queries
 - *WSM*= Windows Service Manager
- **Last inventory attempt (time)** - The last time the inventory has been started (whether successful or not) (contains full date and time).
- **Last inventory** - The last time the inventory has been successfully started (only date part).
- **Last inventory (time)** - The last time the inventory has been successfully started (contains full date and time).
- **Last successful inventory method** - The name of the last inventory method that worked for this device. This method will be preferred in future scans.
- **Last failed inventory methods** - The names of inventory methods that failed the last time the device was scanned.
- **Last failed inventory method (details)** - Details about the methods that failed when the device was scanned for the last time.
- **Show inventory** - Shows the inventory details (only available if an inventory has been already performed).
- **Created** - The date when the device was created or imported.

Context Menu

Pressing Right-Mouse-Button after highlighting an item opens a context menu for it.



- **Select all** - Selects all visible entries in the grid.
- **Deselect all** - Deselects all visible entries in the grid.
- **Add...** - Opens the New Database Dialog.

- **Create copy...** - Opens the New Database Dialog where the default values are automatically set to the values from the current selection.
- **Remove** - Removes the currently selected devices (see Removing Oracle connections).
- **Start inventory...** - Opens the Inventory Wizard for the currently selected devices.
- **Audit...** - Starts an audit for the currently selected connections.
- **Usage statistics...** - Starts **Database Feature Usage Statistics** for the currently selected connections.
- **Edit...** - Opens the Edit Database connection

**Note:**

Some options may be conditionally hidden or disabled (for example the menu item **Edit...** is disabled if more than one device is selected).

Recent Scan Details

The results of the most recent scan are shown here. If the last scan was not successful, the last successful scan is also shown and can be selected from the drop-down.

For more information about working with Inventory log, refer to the following chapter: Recent Scan Details

Viewing Inventory Details

Once a database has been scanned successfully, a button which allows to show the details of scanned inventory assets will be shown in the sidebar. The button can be used to open a detailed overview of Oracle database instance details. For convenience, the summary of the database connection is also shown directly in the sidebar.

The **Details Inventory** overview contains several more tabs which are context-sensitive and which display various information depending on the connection type.

Undocking the Inventory View

It is possible to "undock" the view to a separate window which can be moved freely and stacked with other windows. In order to do that, press the little **undock** button located next to the **CLOSE** button of the inventory overlay.

**Note:**

After undocking, the window cannot be docked in the main window anymore.

Working with Raw Data

IT professionals and administrators having experience with data structures of RayVentory Scan Engine object domain can also work directly with underlying inventory files (NDI).

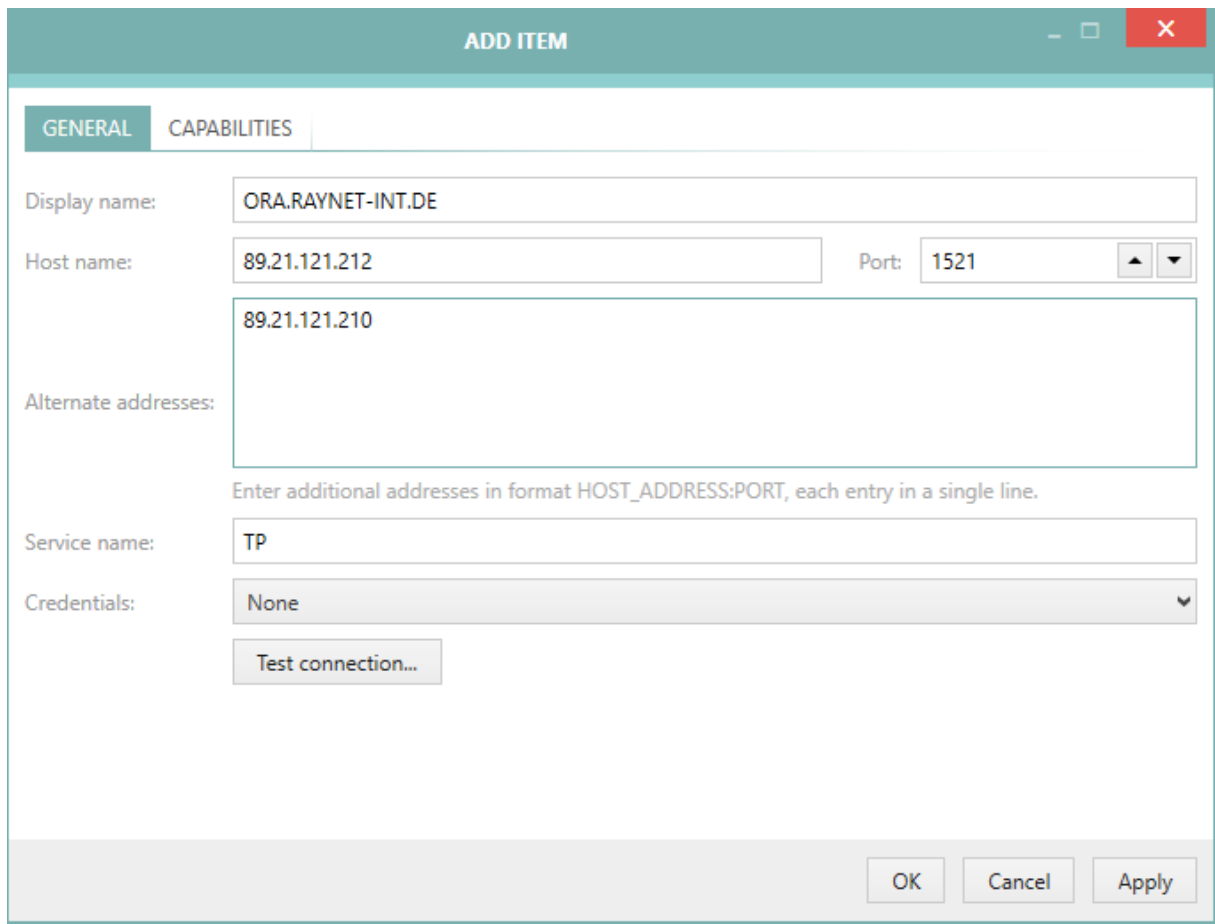
To see the raw content:

1. In the inventory overview select the last **RAW DATA** tab.
2. The tree shows the logical structure of the underlying XML data contained within the `.ndi` file.
3. It is possible to expand the trees to reach the node containing the required content.
4. The three buttons in the toolbar have the following meaning:
 - Copy the full path to `.ndi` file to the clipboard.
 - Open the `.ndi` file in the default editor.
 - Open Windows Explorer and highlight the `.ndi` file.

Adding New Database Connections

In Order to Add a New Database Connection

1. Press **+** button in the top toolbar or click **Add..** menu item from the context menu.
2. A new empty dialog will be shown.



ADD ITEM

GENERAL | CAPABILITIES

Display name: ORA.RAYNET-INT.DE

Host name: 89.21.121.212 Port: 1521

Alternate addresses: 89.21.121.210

Enter additional addresses in format HOST_ADDRESS:PORT, each entry in a single line.

Service name: TP

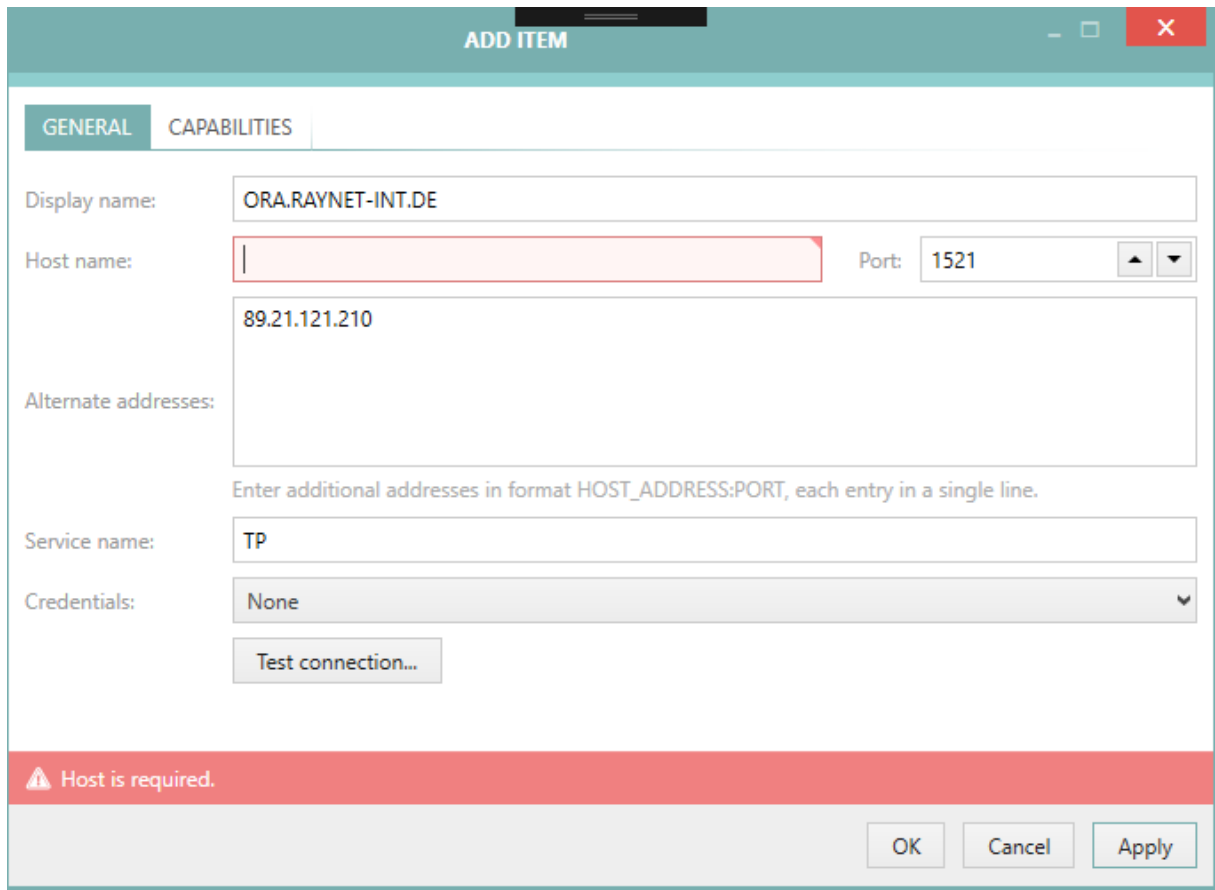
Credentials: None

Test connection...

OK Cancel Apply

3. At the minimum, specify the host name and the service name.
4. It is possible to optionally select the preferred credentials used by this database connection. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.
5. In the **CAPABILITIES** tab, it is possible to limit the capabilities of the newly added device.
6. Press **OK** to accept the changes and close the window, or click **Apply** to immediately save them.

7. If any required field is not specified or is in the wrong format a validation error is shown:




The screenshot shows a window titled "ADD ITEM" with two tabs: "GENERAL" and "CAPABILITIES". The "GENERAL" tab is active. It contains the following fields and controls:

- Display name:** A text box containing "ORA.RAYNET-INT.DE".
- Host name:** A text box that is empty and has a red border, indicating a validation error.
- Port:** A spinner box set to "1521".
- Alternate addresses:** A text box containing "89.21.121.210". Below it is a hint: "Enter additional addresses in format HOST_ADDRESS:PORT, each entry in a single line."
- Service name:** A text box containing "TP".
- Credentials:** A dropdown menu set to "None".
- Test connection...:** A button located below the credentials dropdown.

A red error bar at the bottom of the dialog contains a warning icon and the text "Host is required." At the bottom right are three buttons: "OK", "Cancel", and "Apply".

Fix the issues indicated by the red error bar and press **OK / Apply** to apply the changes.

 **Note:** The connection properties dialog that is being used to create and edit a vSphere / EX connection features a **Test connection...** button that allows to test the connection (and the credentials in the credential store) immediately.

Database Connection Capabilities

The capabilities section allows users to precisely configure which low-level capabilities each device supports.

EDIT ITEM

GENERAL
CAPABILITIES

Inventory capabilities

Inventory methods supported by this device

- ☒ Zero-touch
- ☒ Remote execution

Capabilities supported by this device:

- ☒ Access to File System
- ☒ Upload to SMB shares
- ☒ Windows Management Instrumentation (WMI) Queries
- ☒ Windows Service Manager

Compatible methods

This read-only view shows which inventory methods are available given the currently configured compatibility options and the device type.

We are unable to statically determine the family of the remote host type (Windows or UNIX). The results below may be inaccurate, the actual inventory methods will be determined dynamically during the inventory scan.

Compatible methods (8)

- ✓ RE by ServiceManager upload HTTP(S) on Windows ⓘ
- ✓ RE by WMI upload HTTP(S) on Windows ⓘ
- ✓ RE by ServiceManager upload SMB on Windows ⓘ
- ✓ RE by WMI upload SMB on Windows ⓘ
- ✓ RE by ServiceManager/SMB local files on Windows ⓘ
- ✓ RE by WMI/SMB local files on Windows ⓘ
- ✓ RE by SSH/SCP local files on Linux/Unix ⓘ
- ✓ ZT audit by SQLPLUS ⓘ

Incompatible methods (1)

- ✗ ZT by ORATRACK ⓘ

OK
Close
Apply

The combination of supported capabilities is used by the **Inventory** Wizard to determine, which methods are compatible with which target. For example, unchecking Zero-Touch options for a device will render it only scannable by remote execution methods.

The tab consists of three panels:

- The selection of general inventory methods supported by this device (Zero-Touch and / or Remote-Execution).
- More low-level capabilities, determining whether particular features should be supported by this device.
- The read-only view showing in real-time which inventory methods are applicable given the current state of the device.

Zero-Touch execution does not require any additional capabilities other than certain RayVentory

Scan Engine settings. On the other hand, the Remote-Execution may have different prerequisites depending on the method type. By ticking and unticking the checkboxes next to the methods the lower view is refreshed and shows dynamically which methods are applicable for a given device. By hovering a mouse over the info icon next to a name a detailed information is shown explaining which factors affect the static availability of the current inventory method.

The target platform cannot be statically determined for Oracle connections. The methods that depend on a particular host type are always shown as compatible as long as their other requirements are met.

The options set here affect which methods are available later on when doing inventory scanning. Refer to advanced topic Inventory Methods Overview to find out about the prerequisites and required settings.

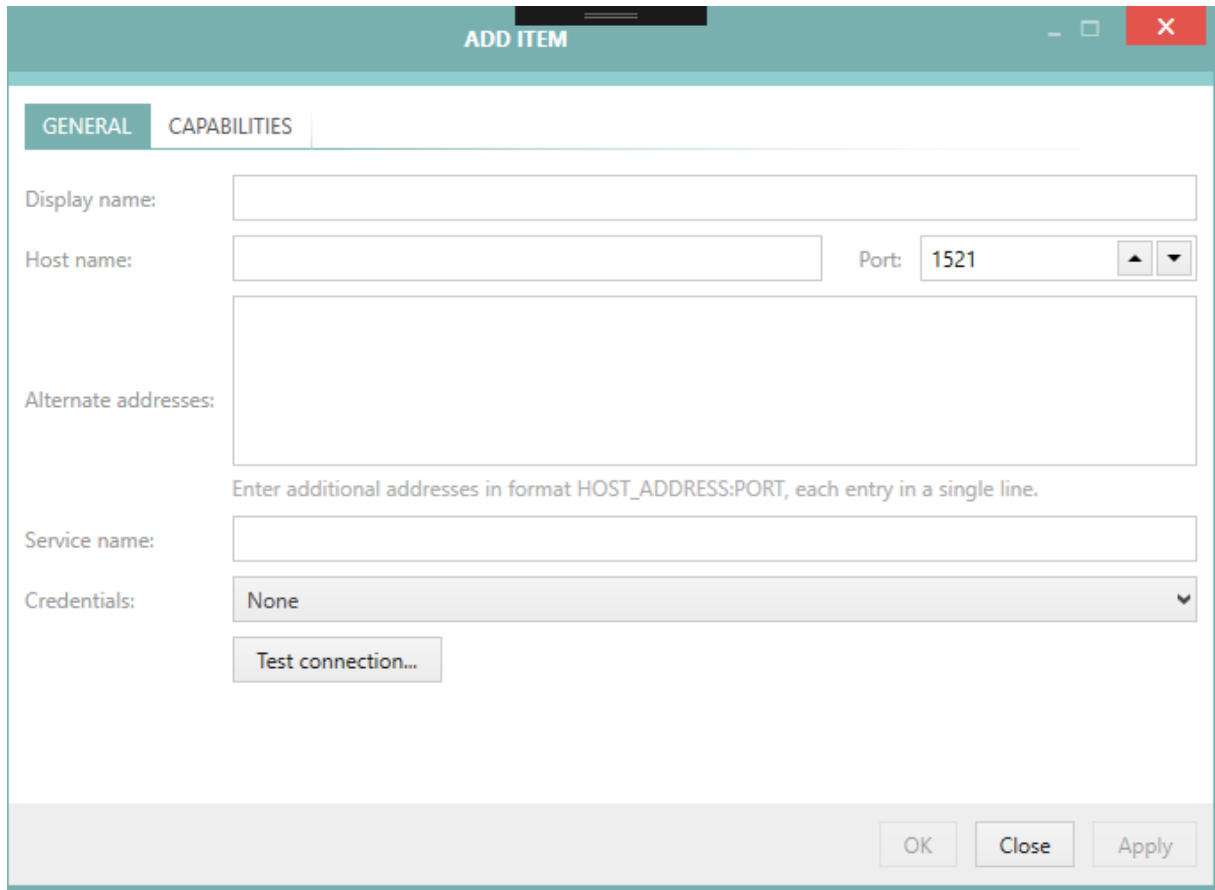
Preventing a Connection from Being Scanned

You can opt-out for any further scans of the connection, thus preserving its current inventory state. To do that, disable the Zero-Touch and the Remote-Execution inventory. The list that is shown underneath these options should reflect this by saying that currently there is no compatible method. This effectively means that, when the user executes an inventory job (from the Inventory wizard, PowerShell command let or from a scheduled task), the database is never scanned and its current inventory files and details are not affected by the new scan.

Editing Database Connections

In Order to Edit a Database Connection

1. Highlight an entry in the list and press the **Edit selected...** button, click the **Add..** menu item from the context menu, or press the **Edit device...** button in the sidebar.
2. A new dialog will be shown with the details of the current selection:



ADD ITEM

GENERAL CAPABILITIES

Display name:

Host name: Port: 1521

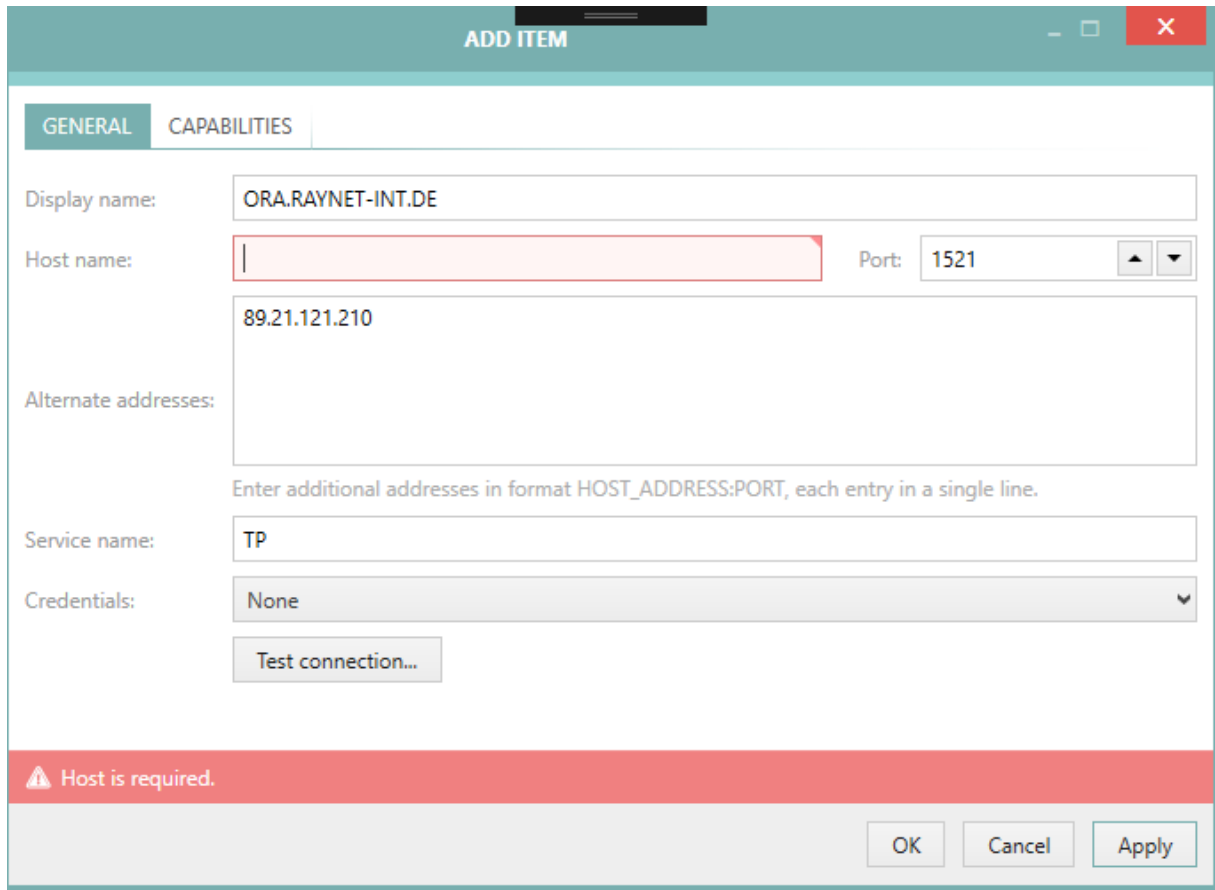
Alternate addresses:

Enter additional addresses in format HOST_ADDRESS:PORT, each entry in a single line.

Service name:

Credentials: None

3. Before the connection can be saved it is necessary to specify at least the device host name and the service name.
4. Optionally, the preferred credentials used by this database can be selected. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.
5. In the **CAPABILITIES** tab, it is also possible to limit the capabilities of the edited database connection.
6. Press **OK** to save the change and close the window or **Apply** to immediately save them.
7. If any mandatory field is not specified or is in the wrong format a validation error is shown:



Fix the issues indicated by the red error bar, and press **OK** / **Apply** to apply the changes.



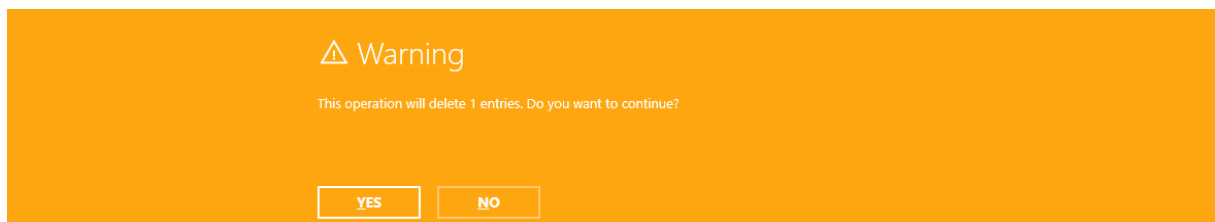
Note:

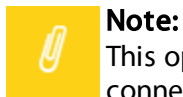
- The connection properties dialog that is being used to create and edit a vSphere / EX connection features a **Test connection...** button that allows to test the connection (and the credentials in the credential store) immediately.

Removing Database Connections

In Order to Remove a Database Connection

1. Highlight an entry in the list and press the - button or click the **Remove...** menu item from the context menu.
2. Confirm the deletion by pressing **YES** in the confirmation dialog.




Note:

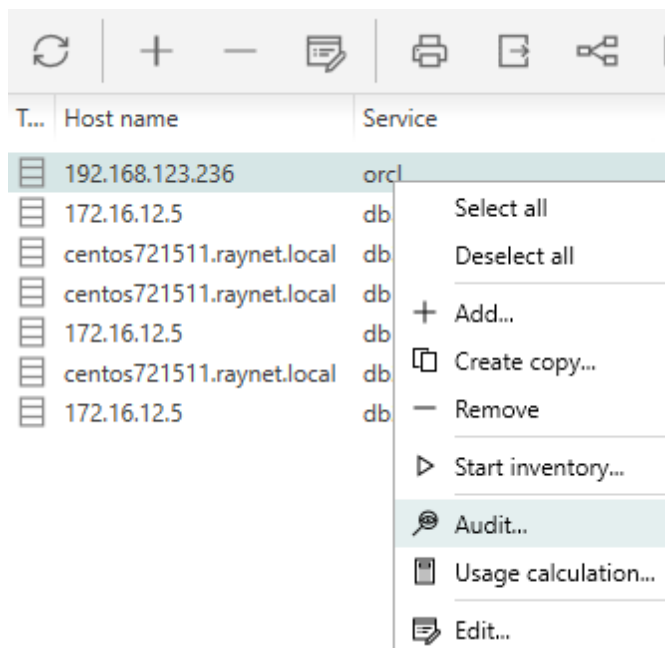
This operation is irreversible. Any existing inventory files which were assigned to the connection stay, though.

Support for Review Lite Script

With RayVentory Scan Engine, it is possible to run the **Oracle's Review Lite Script** on many target databases at once and collect the output files.

To use this option in the settings under **Review Lite Script path** the path to the copy of the **Review Lite Script** needs to be set. Furthermore, the path to the SQLplus executable in the local Oracle client installation needs to be set (see Oracle settings for more information on that).

To perform an audit on one or more Oracle databases, select them in the Oracle view, press the **Right Mouse Button** to open a context menu, and select **Audit....**



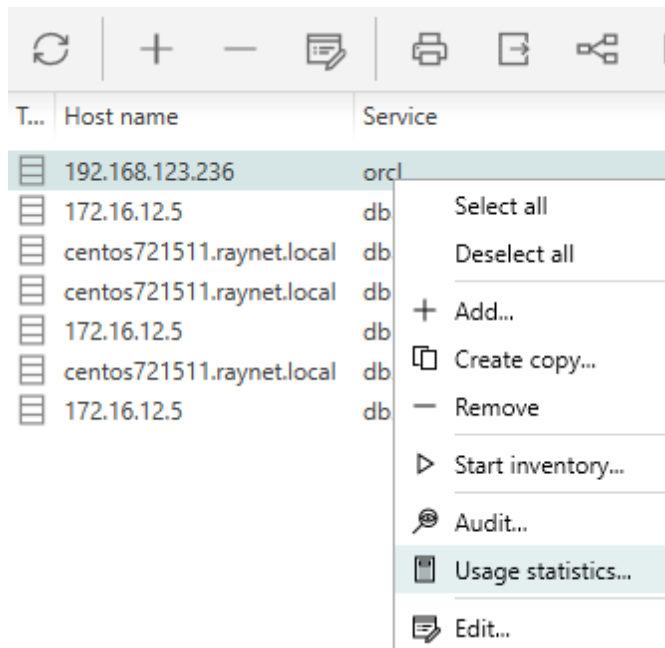
Alternatively, it is possible to invoke the same functionality by pressing the button **Audit** from the right hand sidebar. This applies to the currently selected item(s) as well.

Support for Database Feature Usage Script

RayVentory Scan Engine can be used to more easily run the **Oracle's Database Feature Usage Statistics** script (DFUS) on multiple target databases at once and collect the output as `.ndi` file.

To use this option in the settings under **Oracle Database Feature Usage Statistics script path** the path to a copy of the DFUS script needs to be set. Furthermore, it is necessary to set a path to the SQLplus executable located in the local Oracle client installation.

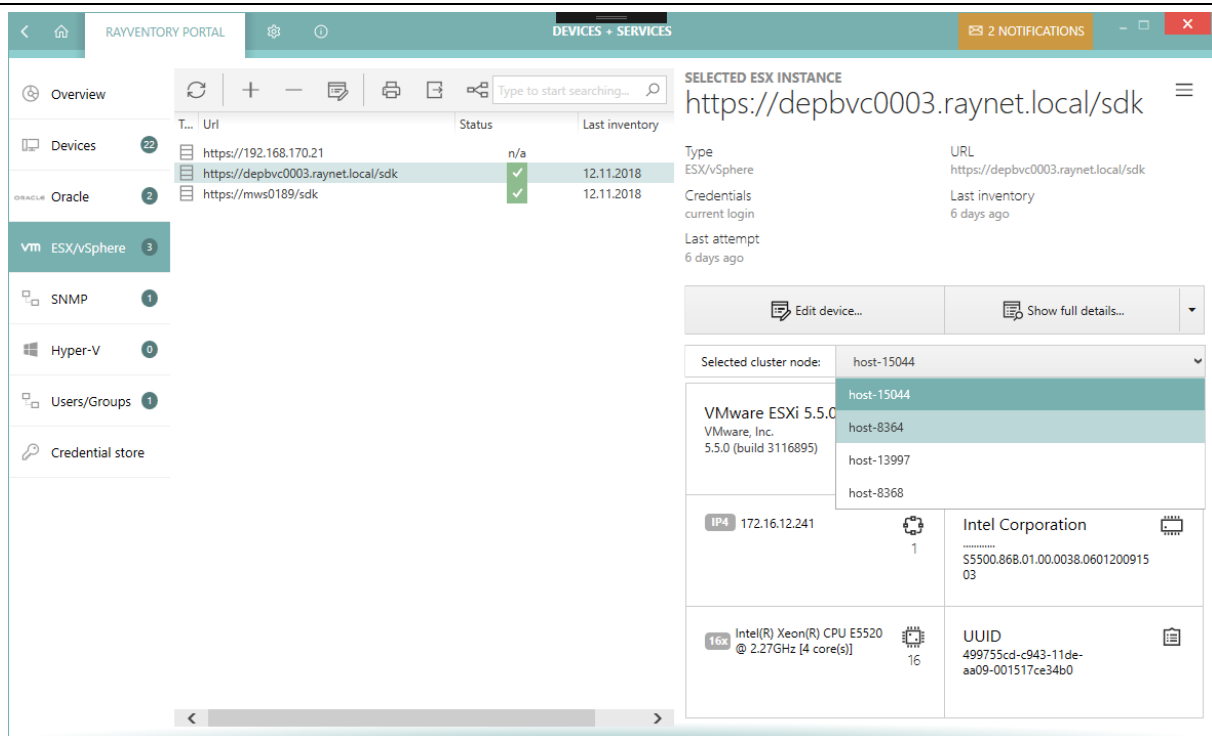
To execute feature usage calculation on one or more Oracle databases select them in the Oracle view, press the **Right Mouse Button** to open a context menu, and select **Usage statistics....**



Alternatively, it is possible to invoke the same functionality by pressing the button **Database Usage Statistics...** from the right hand sidebar. This applies to the currently selected item(s) as well.

ESX / vSphere

This view aggregates discovery and inventory data of your vSphere / ESX instances.



The view is divided into three parts:

- **Toolbar** - showing buttons to perform quick operations like adding, editing, and removing entries.
- **Main grid** - showing the list of all saved devices (physical and virtual).
- **Sidebar** - showing the details of the current selection, including inventory data if available.

Columns

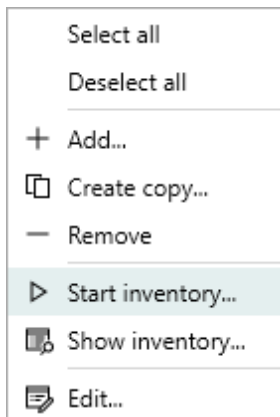
The grid supports a predefined set of columns, out of these columns a handful is visible by default. It is possible to select more columns by pressing the **Column chooser** button in the grid toolbar. The following columns are available:

- **Type** - A static column with the connection icon (vSphere / ESX).
- **URL** - The DNS hostname or the IP address of the instance.
- **Status** - The inventory status. There are 3 possible values:
 - n/a - The device has not been inventoried yet.
 - OK - The device has been inventoried and returned some results.
 - In any other case, the column **Status** contains a short description of the most recent issue.
- **Last inventory attempt** - The last time the inventory has been started (whether successful or not).
- **Credentials** - The logical names of the credentials used when scanning this device.
- **Capabilities** - The short version of enumeration of allowed inventory capabilities for this instance. The string consists of two-letter tokens, with the following meaning:

- *ZT*= Zero-Touch
- *RE*= Remote-Execution
- *FS*= Access to File System
- *SMB*= Upload to SMB Shares
- *WMI*= Windows Management Instrumentation (WMI) Queries
- *WSM*= Windows Service Manager
- **Last inventory attempt (time)** - The last time the inventory has been started (whether successful or not) (contains full date and time).
- **Last inventory** - The last time the inventory has been successfully started (only date part).
- **Last inventory (time)** - The last time the inventory has been successfully started (contains full date and time).
- **Last successful inventory method** - The name of the last inventory method that worked for this instance. This method will be preferred for future scans.
- **Last failed inventory methods** - The names of inventory methods that failed the last time the instance was scanned.
- **Last failed inventory method (details)** - Details about the methods that failed when the instance was scanned for the last time.
- **Created** - The date when the connection was created or imported.

Context Menu

Pressing the Right-Mouse-Button after highlighting an item opens a context menu for it.



- **Select all** - Selects all visible entries in the grid.
- **Deselect all** - Deselects all visible entries in the grid.
- **Add...** - Opens the New vSphere Dialog.
- **Create copy...** - Opens the New vSphere Dialog where the default values are automatically set to the values from the current selection.
- **Remove** - Removes the currently selected devices (see Removing vSphere connections).
- **Start inventory...** - Opens the Inventory Wizard for the currently selected devices.
- **Show inventory** - Shows the details inventory (only available if an inventory has been performed).
- **Edit...** - Opens the Edit vSphere connection connection.



Note:

Some options may be conditionally hidden or disabled (for example the menu item **Edit...** is disabled if more than one device is selected).

Recent Scan Details

The results of the most recent scan are shown here. If the last scan was not successful, the last successful scan is also shown and can be selected from the drop-down.

For more information about working with Inventory log, refer to the following chapter: Recent Scan Details

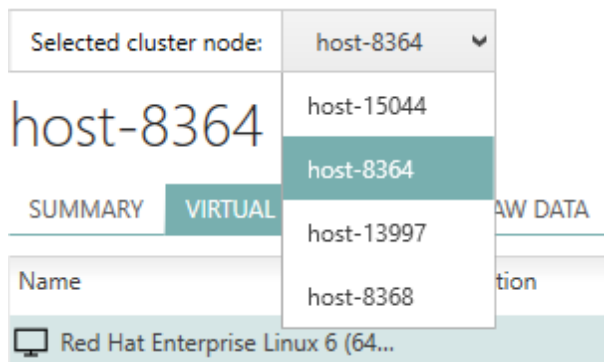
Viewing Inventory Details

Once an vSphere / ESX instance has been successfully scanned, a button will be shown in the sidebar allowing the user to show the details of the scanned inventory asset including the virtual machines running on it. It is possible to use the button to open a detailed overview of machine software, hardware, and virtualized guests. For convenience, the summary of the vSphere / ESX instance is also shown directly in the sidebar.

The details inventory overview contains several more tabs, which are contextually sensitive and display various information depending on the connection type.

Working with Clusters

For vSphere / ESX connections a selector of clusters is available if there is more than one cluster in the inventory data. The inventory view always shows the details of the current cluster. To change the current cluster, select its name from the drop down menu.



Accessing Details about Guest Virtual Machines

Once in the inventory details, the tab **VIRTUAL MACHINES** can be used to view the details of hosted virtual guests:

Selected cluster node: host-8364

X

host-8364

SUMMARY
VIRTUAL MACHINES 55
RAW DATA

| Name | Association | State | DataStore | Uuid | Network c... | CPU |
|------------------------------------|-------------|--|----------------------------------|-----------------------|--------------|-----|
| Red Hat Enterprise Linux 6 (64... | | ■ OFF | [System_HDD_99] CentOS 6 x6... | 4209f657-3101-32dc... | | 0 |
| Microsoft Windows Server 201... | | ■ OFF | [System_HDD_99] Win2k16_64... | 42099f08-9d84-6115... | | 0 |
| Microsoft Windows 8 (64-bit) | | ■ OFF | [devtemplates] Neue virtuelle... | 4209f35a-bdbe-c4b2... | | 2 |
| Microsoft Windows Server 200... | | ▶ Running | [S3_LUN1_5000] RV-Win10-Cli... | 4209c75a-7a35-d3fa... | | 2 |
| Microsoft Windows 8 (32-bit) | | ■ OFF | [S3_LUN2_5000] Manuel_MD3... | 4209efb0-1077-a860... | | 1 |
| Microsoft Windows Server 201... | | ■ OFF | [S3_LUN2_5000] Win2k16 Test... | 4209f8d0-69e5-debc... | | 0 |
| Microsoft Windows Server 200... | | ■ OFF | [Templates] Win2k8R2_AIO_ED... | | | 0 |
| SUSE Linux Enterprise 11 (64-... | | ■ OFF | [S3_LUN2_5000] Marlon SLES... | 42099c24-ef3c-efac... | | 1 |
| Microsoft Windows Server 200... | | ▶ Running | [S3_LUN3_5000] PBPR03SVDC... | 4209ca50-8d02-f3e6... | | 10 |
| Microsoft Windows 8 (64-bit) | | ■ OFF | [Templates] Win10x64/Win10x... | 4209a2a2-b7de-3d5... | | 1 |
| Microsoft Windows Server 201... | | ▶ Running | [S3_LUN3_5000] RV Server 11/... | 4209dd90-1e9d-82bf... | | 1 |
| Red Hat Enterprise Linux 5 (32-... | | ■ OFF | [System_HDD_99] CentOS 5/C... | 4209a5dc-8e64-044b... | | 0 |
| Microsoft Windows Server 200... | | ■ OFF | [RAID 6 (DATA)] David Win2k8... | 4209d84d-9a27-af90... | | 1 |
| VMware ESX 4.x | | ■ OFF | [Templates] ESX41/ESX41.vmtx | 42358ee6-3a5b-2d9... | | 2 |

Undocking the Inventory View

It is possible to "undock" the view to a separate window which can be freely moved and stacked with other windows. In order to do that, press the little **undock** button next to the **CLOSE** button of the inventory overlay.

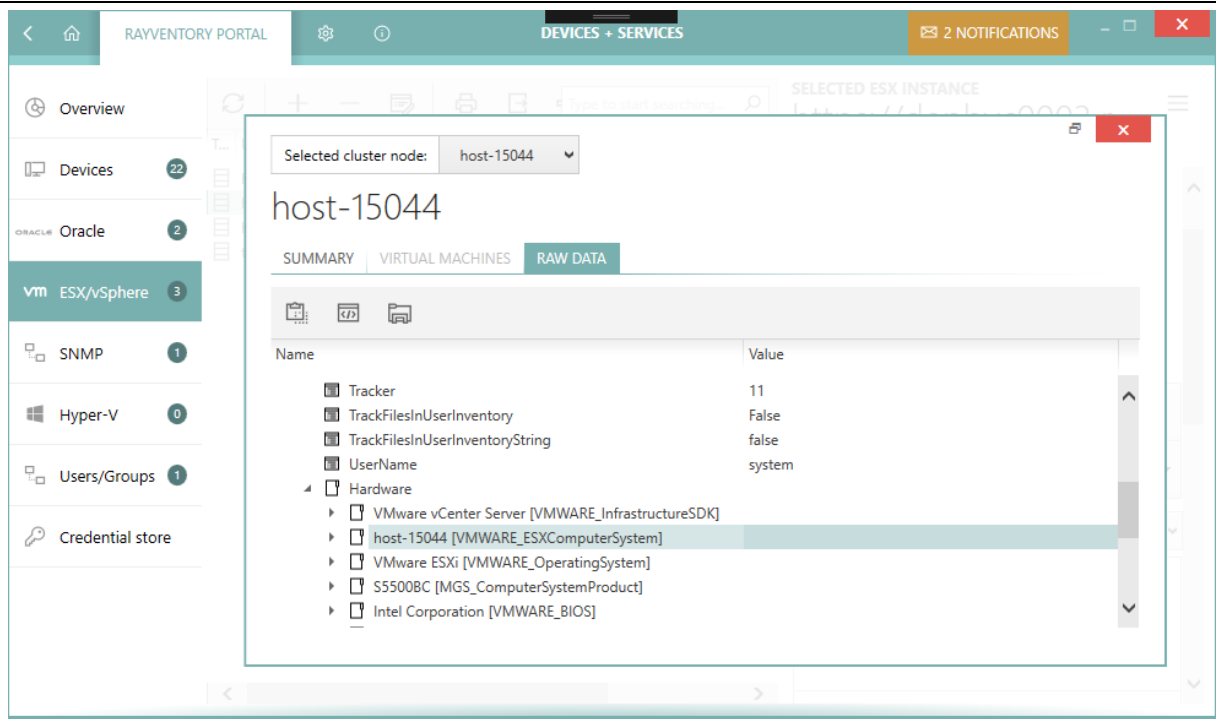


Note:

After undocking, the window cannot be docked to the main window anymore.

Working with Raw Data

IT professionals and administrators having experience with data structures of the RayVentory Scan Engine object domain can also work directly with the underlying inventory files (.ndi).



To see the raw content:

1. In the inventory overview select the last tab **RAW DATA**.
2. The tree shows the logical structure of the underlying XML data contained within the `.ndi` file.
3. It is possible to expand the trees to reach the node containing the required content.
4. The three buttons in the toolbar have the following meaning:
 - Copy the full path to the `.ndi` file to the clipboard.
 - Open the `.ndi` file in the default editor.
 - Open the Windows Explorer and highlight the `.ndi` file.



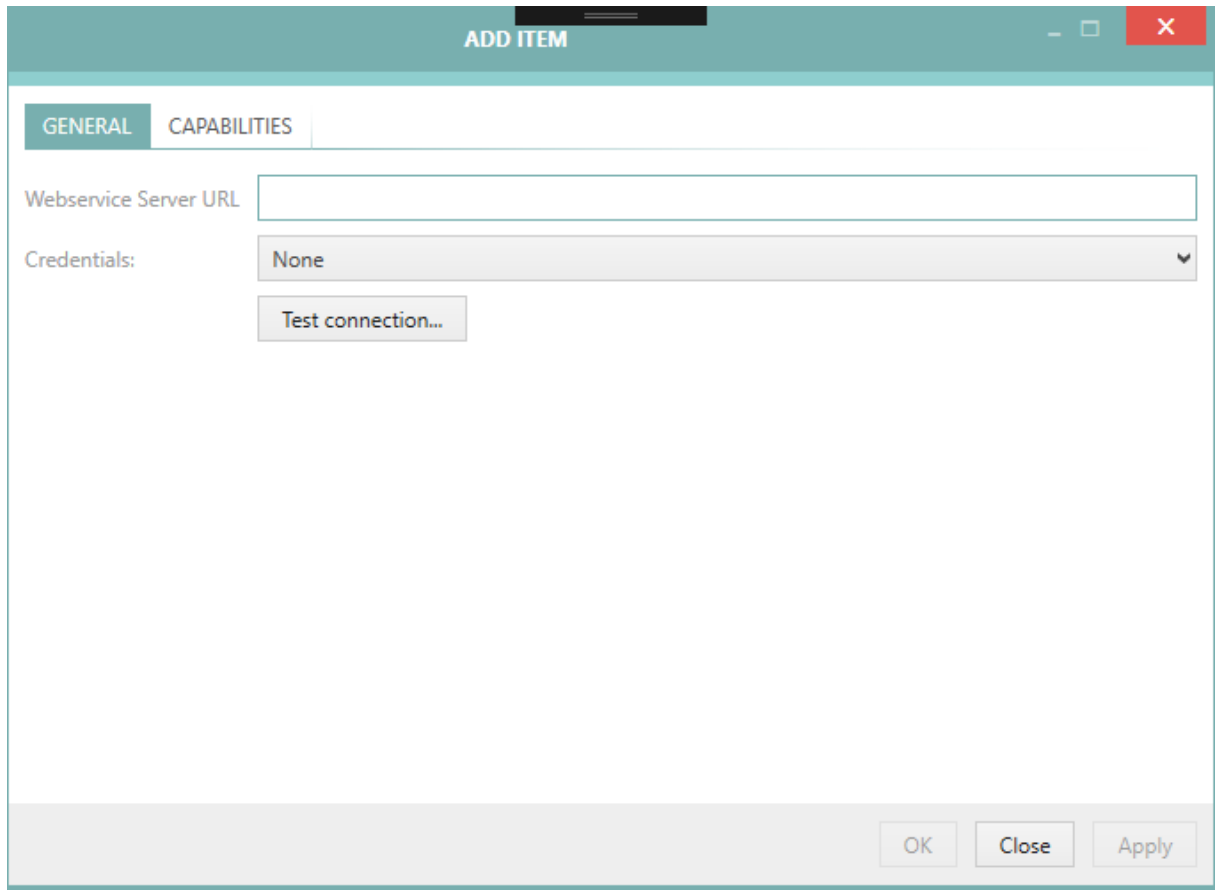
Note:

After undocking, the window cannot be docked to the main window anymore.

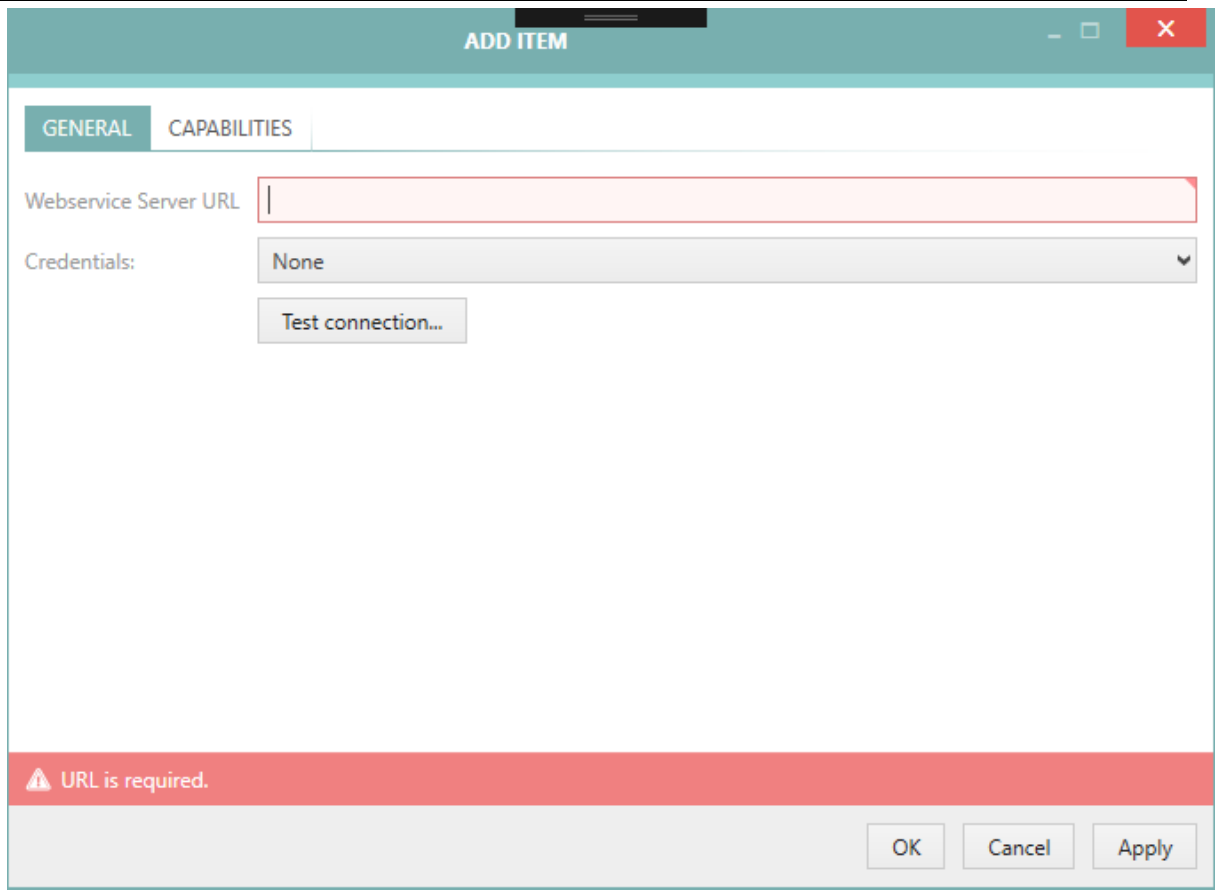
Adding a new ESX / vSphere Connection

In Order to Add a New vSphere Connection

1. Press the **+** button in the top toolbar or click the **Add..** menu item from the context menu.
2. A new empty dialog will be shown.



3. At least the URL address needs to be specified.
4. Optionally, the preferred credentials used by this vSphere connection can be selected. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.
5. In the **CAPABILITIES** tab, the capabilities of the newly added device can be limited.
6. Press **OK** to accept the changes and close the window, or **Apply** to immediately save them.
7. If any mandatory field is not specified or is in the wrong format, a validation error is shown:



Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

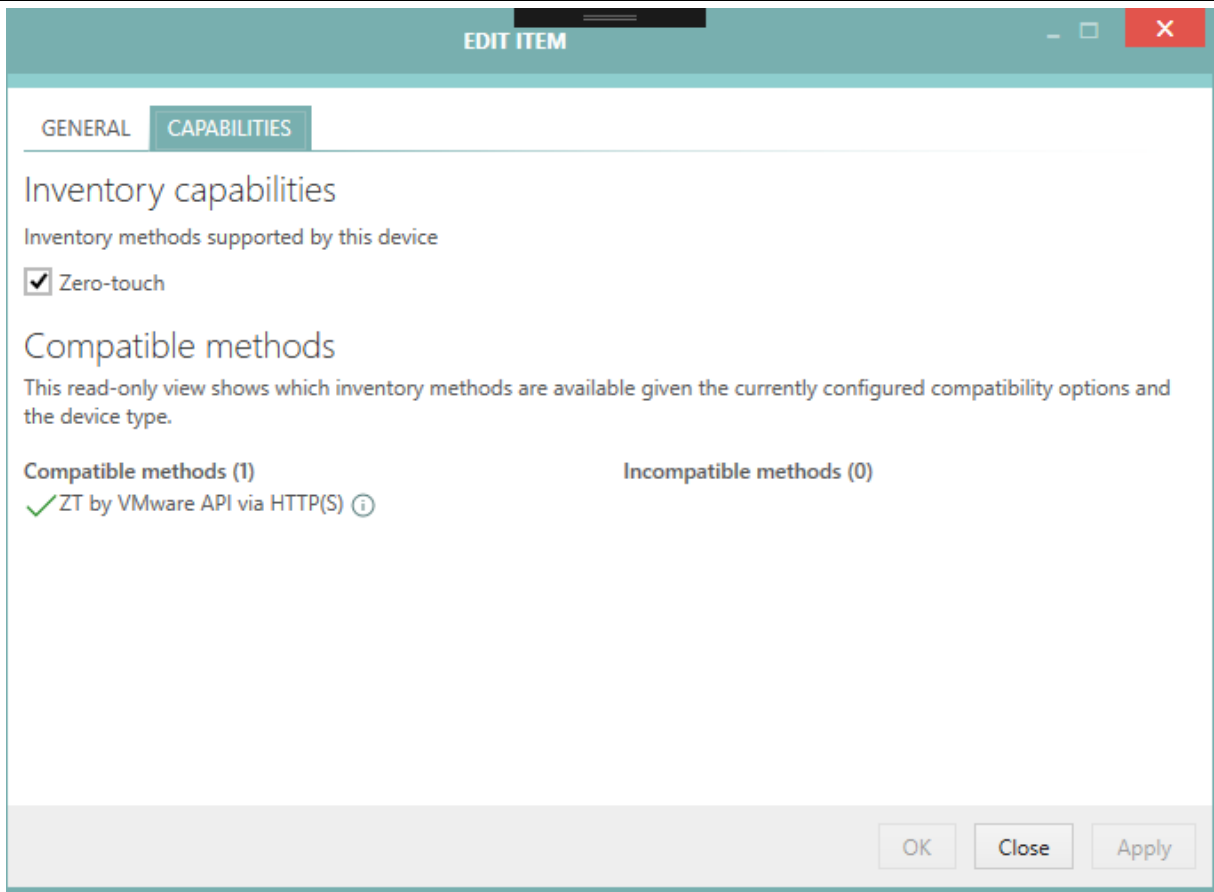


Note:

- The URL for the connection to a vSphere / ESX SDK service endpoint usually has the following form: <https://yoursxhost/sdk>.
- The connection properties dialog that is being used to create and edit a vSphere / EX connection features a **Test connection...** button that allows to test the connection (and the credentials in the credential store) immediately.

vSphere Connection Capabilities

The capabilities section allows users to precisely configure which low-level capabilities each device supports. Currently, only Zero-Touch execution is available for vSphere / ESX connections.



Whether Zero-Touch is enabled or not is used by the Inventory Wizard to determine whether the device can be scanned.

The tab consists of three panels:

- The selection of general inventory methods supported by this device (Zero-Touch and / or Remote-Execution).
- More low-level capabilities determining whether particular features should be supported by this device.
- The read-only view showing in real-time which inventory methods are applicable given the current state of the device.

Refer to the advanced topic [Inventory Methods Overview](#) to find out about the prerequisites and required settings.

Preventing a Connection from Being Scanned

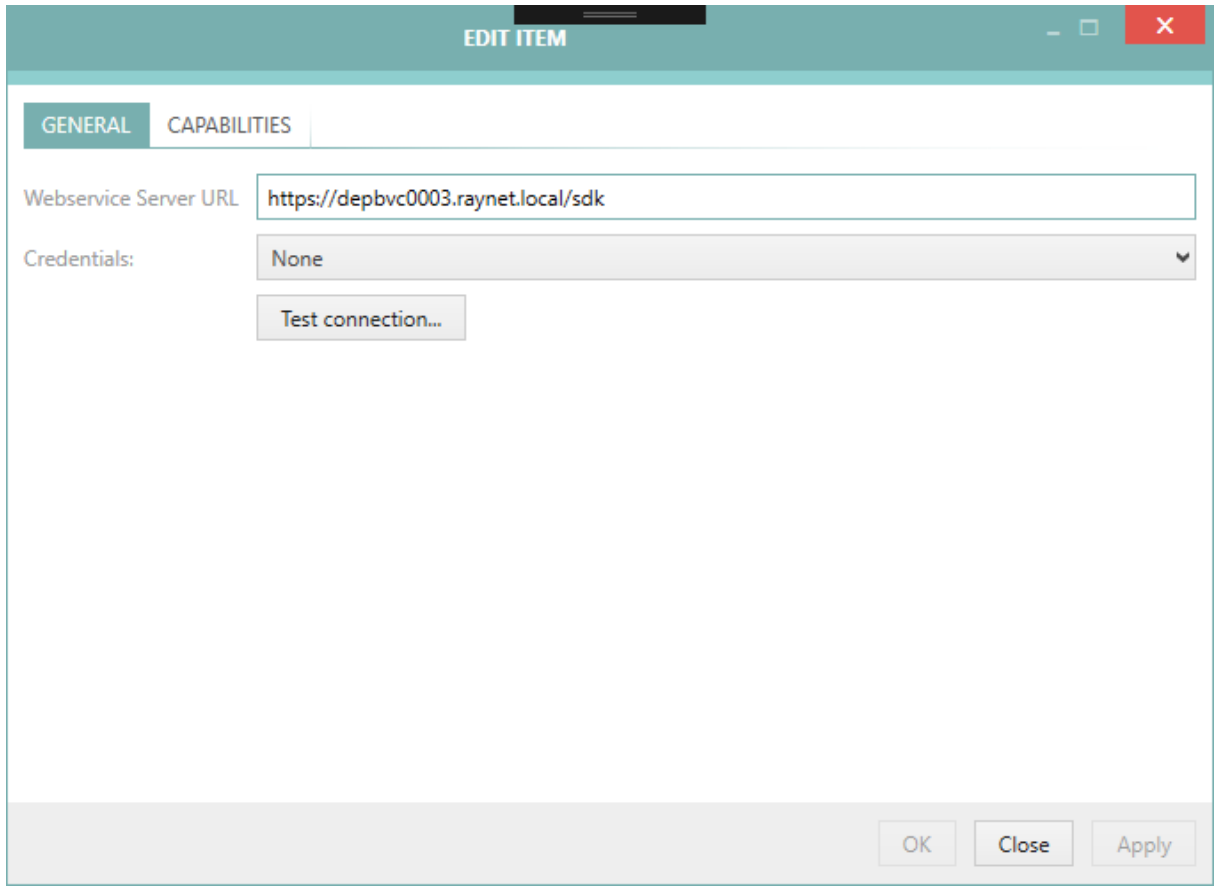
It is possible to opt-out for any further scans of the connection, thus preserving its current inventory state. To do that, disable Zero-Touch scanning. The list that is shown underneath these options should reflect this by saying that currently there is no compatible method. This effectively means, that when the user executes an inventory job (from the Inventory wizard,

PowerShell command let, or from a scheduled task), the instance is never scanned and its current inventory files and details are not affected by the new scan.

Editing vSphere Connections

In Order to Edit a vSphere Connection

1. Highlight an entry in the list and press the **Edit selected...** button, click the **Add..** menu item from the context menu, or press the **Edit device...** button in the sidebar.
2. A new dialog will be shown with the details of the current selection:



3. Before the connection can be saved, at least the device host name and the service name must be specified.
4. Optionally, the preferred credentials used by this database can be selected. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the advanced topics.
5. In the **CAPABILITIES** tab, it is possible to limit the capabilities of the edited database connection.
6. Press **OK** to save the change and close the window or **Apply** to immediately save them.
7. If any required field is not specified or is in the wrong format, a validation error is shown. Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

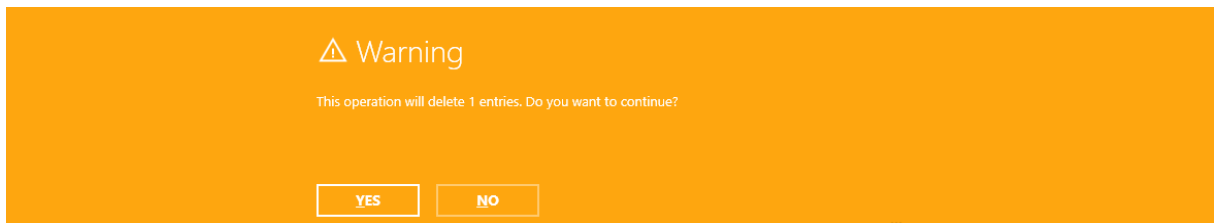
**Note:**

- The URL for the connection to a vSphere / ESX SDK service endpoint usually has the following form: `https://youresxhost/sdk`.
- The connection properties dialog that is being used to create and edit a vSphere / EX connection features a **Test connection...** button that allows to test the connection (and the credentials in the credential store) immediately.

Removing vSphere Connections

In Order to Remove a vSphere Connection

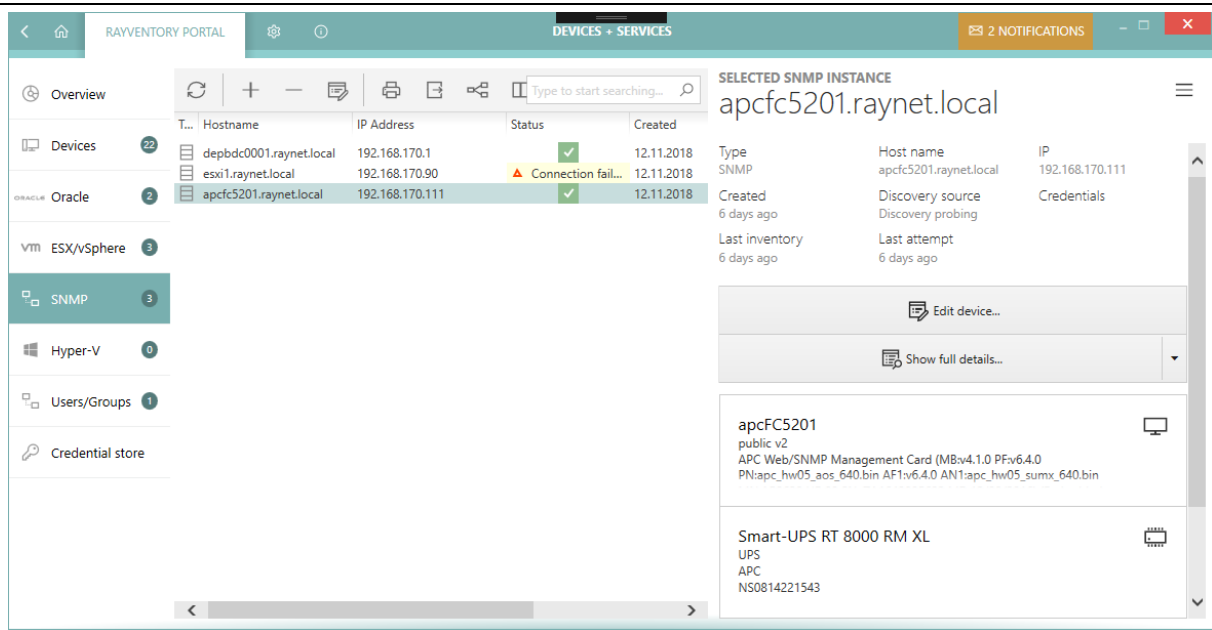
1. Highlight an entry in the list and press the - button or click **Remove...** menu item from the context menu.
2. Confirm the deletion by pressing **YES** in the confirmation dialog.

**Note:**

This operation is irreversible. Any existing inventory files which were assigned to the connection stay, though.

SNMP

This view aggregates discovery and inventory data of your SNMP connections.



| T... | Hostname | IP Address | Status | Created |
|--------|-------------------------|-----------------|----------------------|------------|
| | depbdc0001.raynet.local | 192.168.170.1 | ✓ | 12.11.2018 |
| | esxi1.raynet.local | 192.168.170.90 | ⚠ Connection fail... | 12.11.2018 |
| ORACLE | apcfc5201.raynet.local | 192.168.170.111 | ✓ | 12.11.2018 |

SELECTED SNMP INSTANCE
apcfc5201.raynet.local

| | | |
|------------------------------|---------------------------------------|-----------------------|
| Type SNMP | Host name apcfc5201.raynet.local | IP 192.168.170.111 |
| Created 6 days ago | Discovery source Discovery probing | Credentials |
| Last inventory 6 days ago | Last attempt 6 days ago | |

apcFC5201
public v2
APC Web/SNMP Management Card (MB:v4.1.0 PF:v6.4.0
PN:apc_hw05_aos_640.bin AF1:v6.4.0 AN1:apc_hw05_sumx_640.bin

Smart-UPS RT 8000 RM XL
UPS
APC
NS0814221543

The view is divided into three parts:

- **Toolbar** - showing buttons to perform quick operations like adding, editing, and removing entries.
- **Main grid** - showing the list of all saved devices (physical and virtual).
- **Sidebar** - showing the details of the current selection including inventory data if available.

Columns

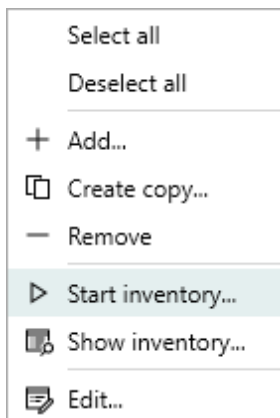
The grid supports a predefined set of columns, only a handful of which are visible by default. It is possible to select more columns by pressing the **Column chooser** button in the grid toolbar. The following columns are available:

- **Type** - A static column with the connection icon (SNMP).
- **Hostname** - The DNS hostname of the instance.
- **IP Address** - The IP Address of the instance.
- **Status** - The inventory status. There are 3 possible values:
 - n/a - The device has not been inventoried yet.
 - OK - The device has been inventoried and returned some results.
 - In any other case, the column **Status** contains a short description of the most recent issue.
- **Last inventory attempt** - The last time the inventory has been started (whether successful or not).
- **Credentials** - The logical names of credentials used when scanning this instance.
- **Capabilities** - The short version of enumeration of allowed inventory capabilities for this instance. The string consists of two-letter tokens, with the following meaning:
 - **ZT**= Zero-Touch
 - **RE**= Remote-Execution

- *FS* = Access to File System
- *SMB* = Upload to SMB Shares
- *WMI* = Windows Management Instrumentation (WMI) Queries
- *WSM* = Windows Service Manager
- **Last inventory attempt (time)** - The last time the inventory has been started (whether successful or not) (contains full date and time).
- **Last inventory** - The last time the inventory has been successfully started (only date part).
- **Last inventory (time)** - The last time the inventory has been successfully started (contains full date and time).
- **Last successful inventory method** - The name of the last inventory method that worked for this instance. This method will be preferred in future scans.
- **Last failed inventory methods** - The names of inventory methods that failed the last time the instance was scanned.
- **Last failed inventory method (details)** - Details about the methods that failed when the instance was scanned for the last time.
- **Created** - The date when the connection was created or imported.

Context Menu

Pressing Right-Mouse-Button after highlighting an item opens a context menu for it.



- **Select all** - Selects all visible entries in the grid.
- **Deselect all** - Deselects all visible entries in the grid.
- **Add...** - Opens the New SNMPDialog.
- **Create copy...** - Opens the New SNMPDialog where the default values are automatically set to the values from the current selection.
- **Remove** - Removes the currently selected devices (see Removing SNMP connections).
- **Start inventory...** - Opens the Inventory Wizard on the currently selected devices.
- **Show inventory** - Shows the details inventory (only available if an inventory has been already performed).
- **Edit...** - Opens the Edit SNMP connection connection.

Note: Some options may be conditionally hidden or disabled (for example the menu item **Edit...** is disabled if more than one device is selected).

Recent Scan Details

The results of the most recent scan are shown here. If the last scan was not successful, the last successful scan is also shown and can be selected from the drop-down.

For more information about working with Inventory log, refer to the following chapter: Recent Scan Details

Viewing Inventory Details

Once a device has been successfully scanned for its software and hardware, a button will be shown in the sidebar allowing the user to show the content of specified machine:

SUMMARY

RAW DATA

| | |
|---|---|
| DEPBDC0001 public v2 Hardware: Intel64 Family 6 Model 44 Stepping 2 AT/AT COMPATIBLE - Software: |  <div> <div>IP4 127.0.0.1</div> <div>IP4 192.168.170.1</div> </div> <div>  2 </div> |
|---|---|

The button can be used to open a detailed overview of the machine software and hardware. For convenience, the summary of machine assets is also shown directly in the sidebar.

The details inventory overview contains several tabs which are contextually sensitive and display various information depending on the connection type.

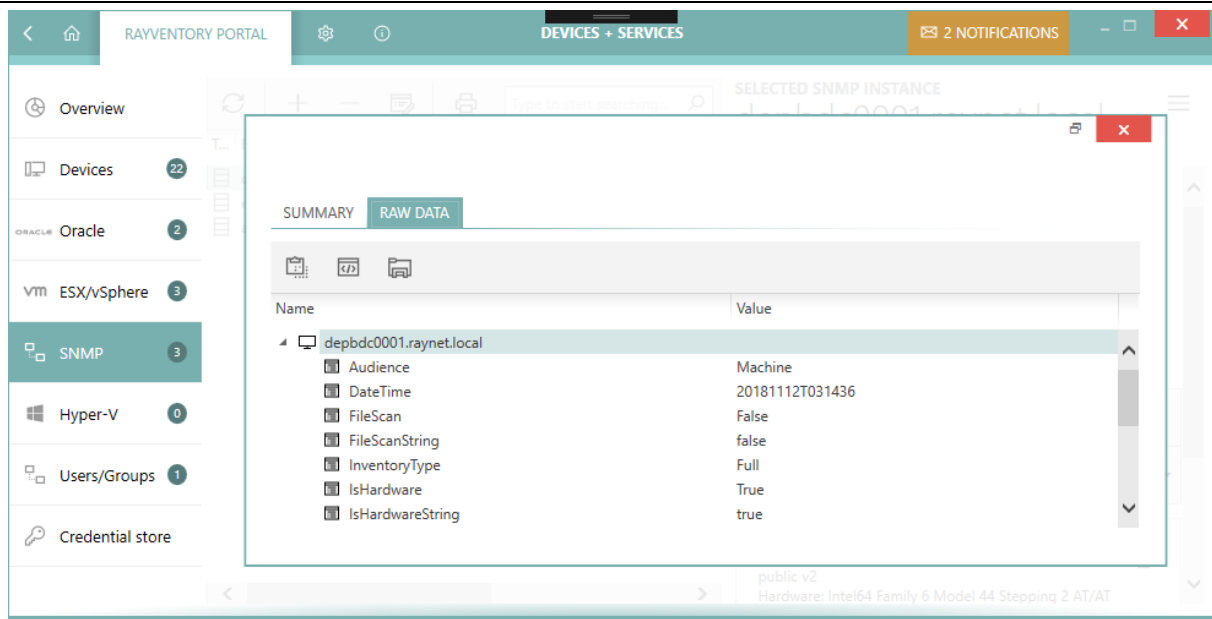
Undocking the Inventory View

It is possible to "undock" the view to a separate window which can be freely moved and stacked with other windows. In order to do that, press the little **undock** button next to the **CLOSE** button of the inventory overlay.

Note: After undocking, the window cannot be docked to the main window anymore.

Working with Raw Data

IT professionals and administrators having experience with the data structures of the RayVentory Scan Engine object domain can also work directly with underlying inventory files (.ndi).



To see the raw content:

1. In the inventory overview select the last tab **RAW DATA**.
2. The tree shows the logical structure of the underlying XML data contained within the `.ndi` file.
3. The trees can be expanded to reach the node containing the required content.
4. The three buttons in the toolbar have the following meaning:
 - Copy the full path to the `.ndi` file to the clipboard.
 - Open the `.ndi` file in the default editor.
 - Open Windows Explorer and highlight the `.ndi` file.



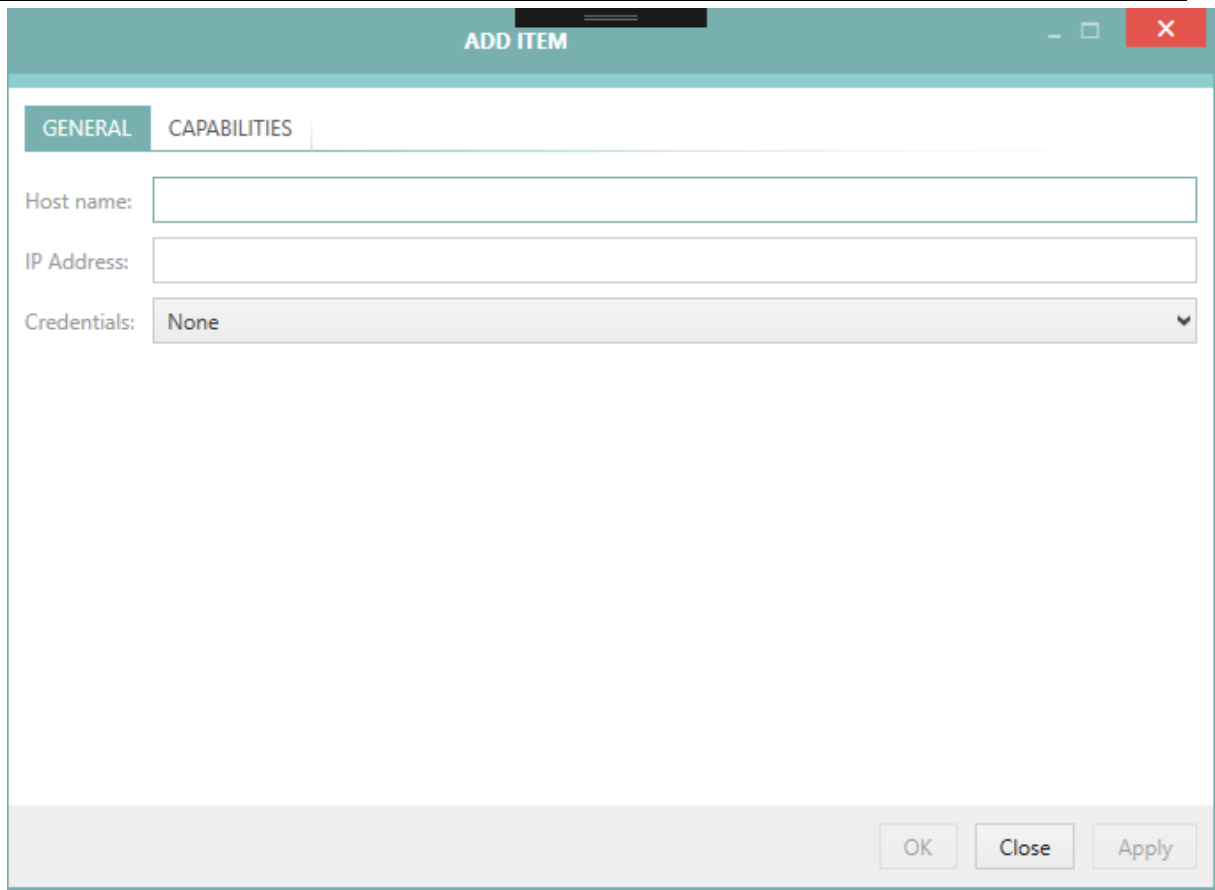
Note:

After undocking, the window cannot be docked to the main window anymore..

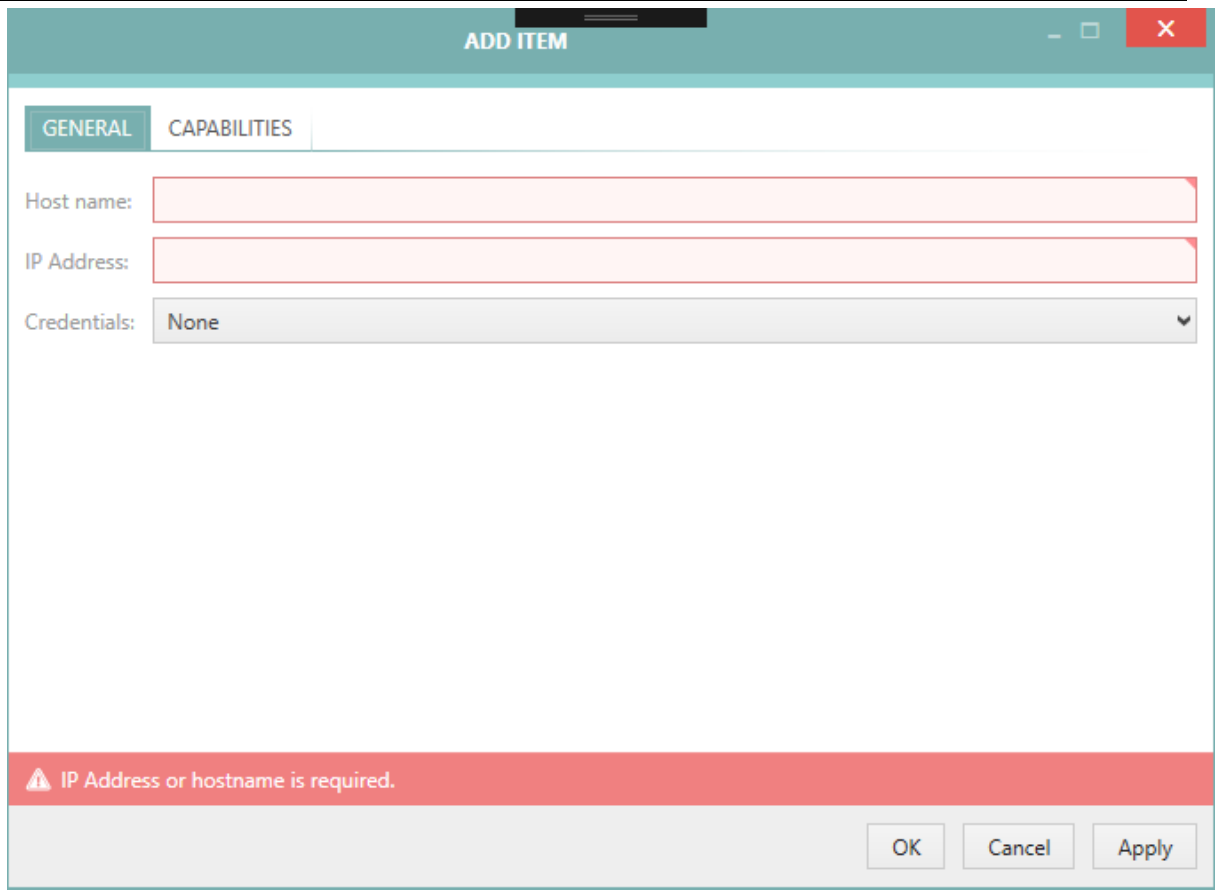
Adding a New SNMP Connection

In Order to Add a New SNMP Connection

1. Press the **+** button in the top toolbar or click the **Add..** menu item from the context menu.
2. A new empty dialog will be shown.



3. At least, the Host name and / or the IP Address need to be specified.
4. Optionally, the preferred credentials used by this vSphere connection can be selected. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the chapter Advanced Topics.
5. In the **CAPABILITIES** tab, it is possible to limit the capabilities of the newly added device.
6. Press **OK** to accept the changes and close the window, or **Apply** to immediately save them.
7. If any required field is not specified or is in the wrong format, a validation error is shown:



ADD ITEM

GENERAL CAPABILITIES

Host name:

IP Address:

Credentials: None

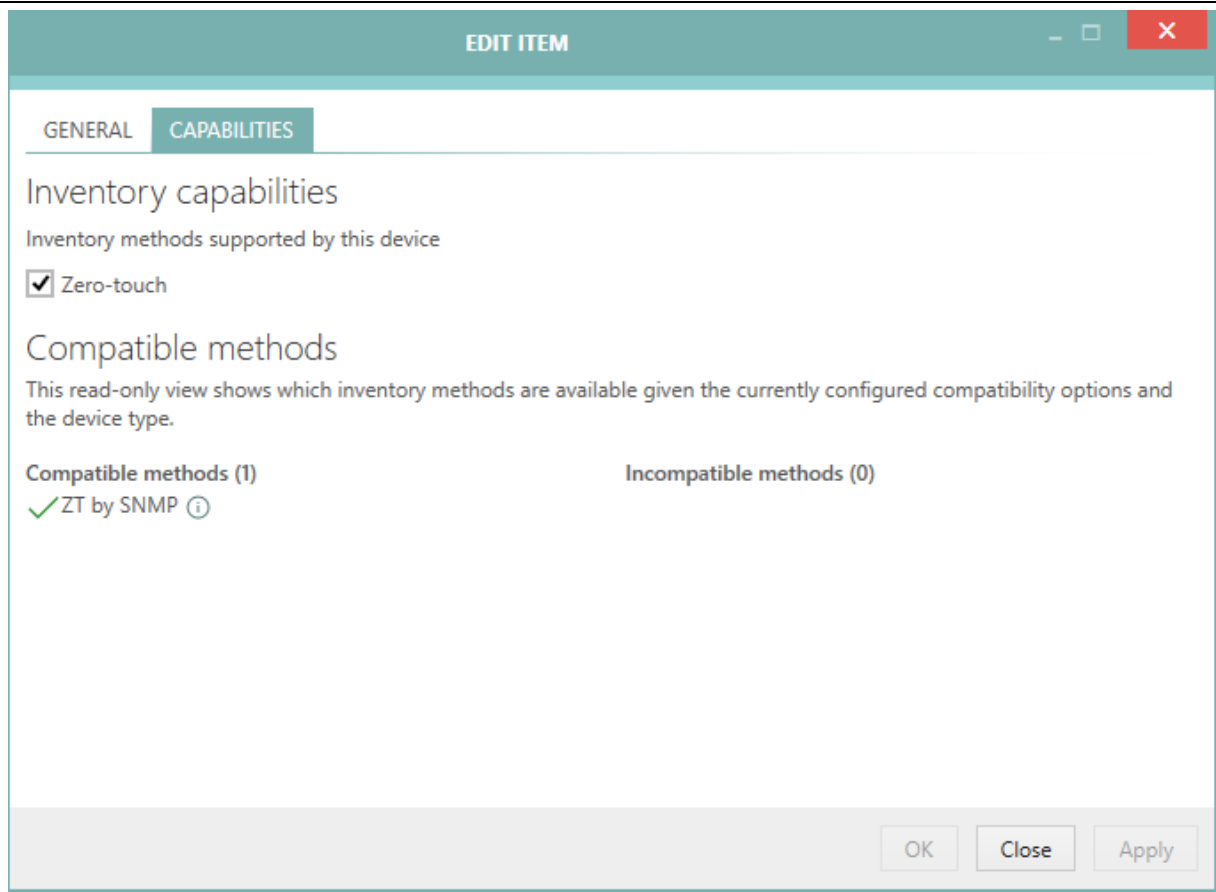
⚠ IP Address or hostname is required.

OK Cancel Apply

Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

SNMP Connection Capabilities

The capabilities section allows users to precisely configure which low-level capabilities each device supports. Currently, only Zero-Touch execution is available for SNMP connections.



Whether Zero-Touch is enabled or not is used by the Inventory Wizard to determine if the device can be scanned.

The tab consists of three panels:

- The selection of general inventory methods supported by this device (Zero-Touch and / or Remote-Execution).
- More low-level capabilities, determining whether particular features should be supported by this device.
- The read-only view showing in real-time which inventory methods are applicable given the current state of the device.

Refer to advanced topic Inventory Methods Overview to find out about the prerequisites and required settings.

Preventing a Connection from Being Scanned

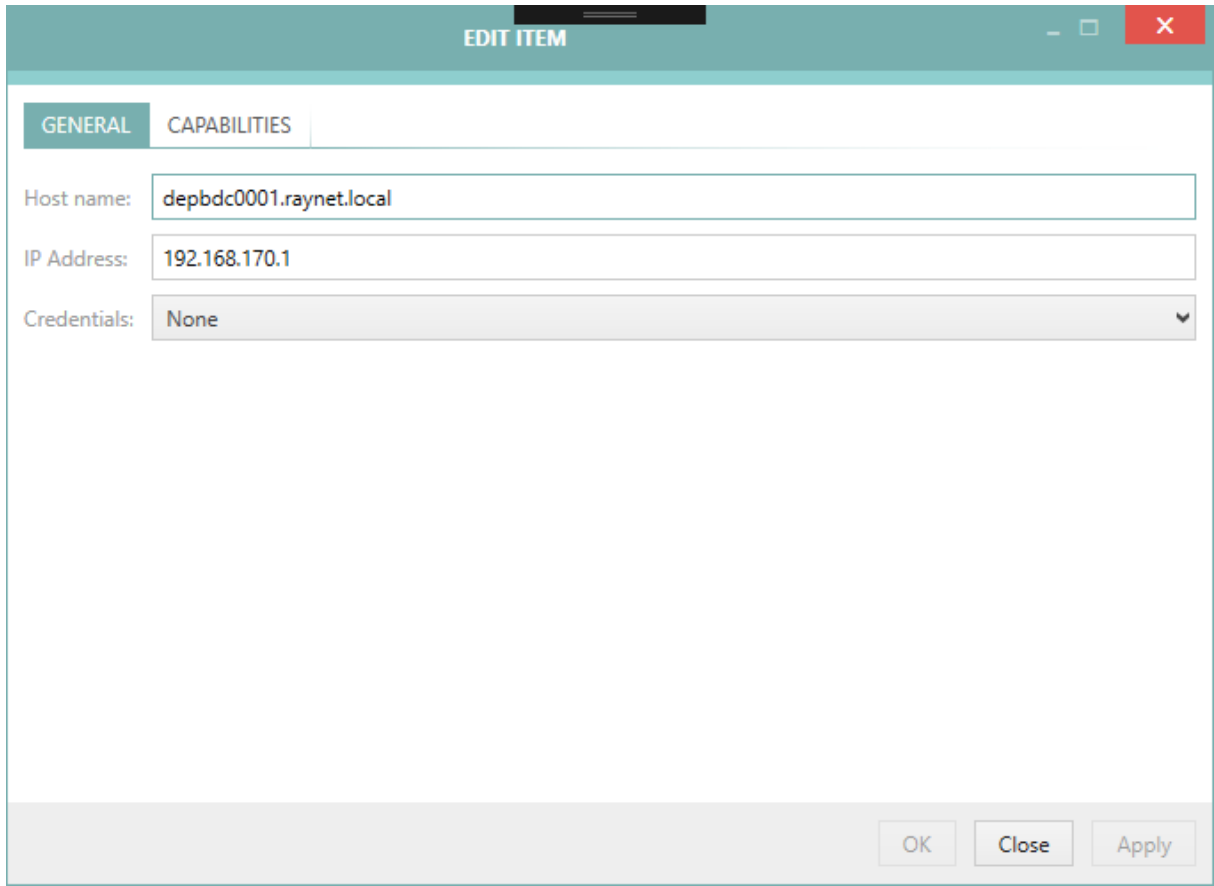
It is possible to opt-out for any further scans of the connection, thus preserving its current inventory state. To do that, disable Zero-Touch scanning. The list that is shown underneath these options should reflect this by saying that currently there is no compatible method. This effectively means, that when the user executes an inventory job (from the Inventory wizard,

PowerShell command let, or from a scheduled task), the instance is never scanned and its current inventory files and details are not affected by the new scan.

Editing SNMP Connections

In Order to Edit an SNMP Connection

1. Highlight an entry in the list and press the **Edit selected...** button, click the **Add..** menu item from the context menu, or press the **Edit device...** button in the sidebar.
2. A new dialog will be shown with the details of the current selection:



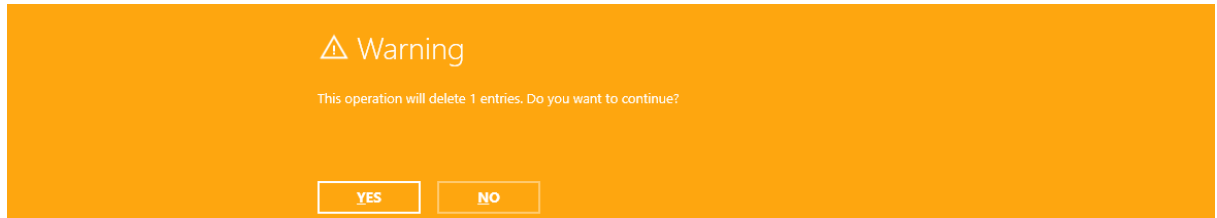
The screenshot shows a dialog box titled "EDIT ITEM". It has two tabs: "GENERAL" and "CAPABILITIES". The "GENERAL" tab is selected. Inside the dialog, there are three input fields: "Host name" with the value "depbdc0001.raynet.local", "IP Address" with the value "192.168.170.1", and "Credentials" with a dropdown menu showing "None". At the bottom right of the dialog, there are three buttons: "OK", "Close", and "Apply".

3. Before the connection can be saved, at least the device host name and the service name must be specified.
4. Optionally, the preferred credentials used by this database can be selected. If this is left empty, RayVentory Scan Engine applies a special logic which is described in the advanced topics.
5. In the **CAPABILITIES** tab, it is possible to limit the capabilities of the edited database connection.
6. Press **OK** to save the change and close the window or **Apply** to immediately save them.
7. If any required field is not specified or is in the wrong format, a validation error is shown. Fix the issues indicated by the red error bar and press **OK** / **Apply** to apply the changes.

Removing SNMP Connections

In Order to Remove an SNMP Connection

1. Highlight an entry in the list and press the - button or click the **Remove...** menu item from the context menu.
2. Confirm the deletion by pressing **YES** in the confirmation dialog

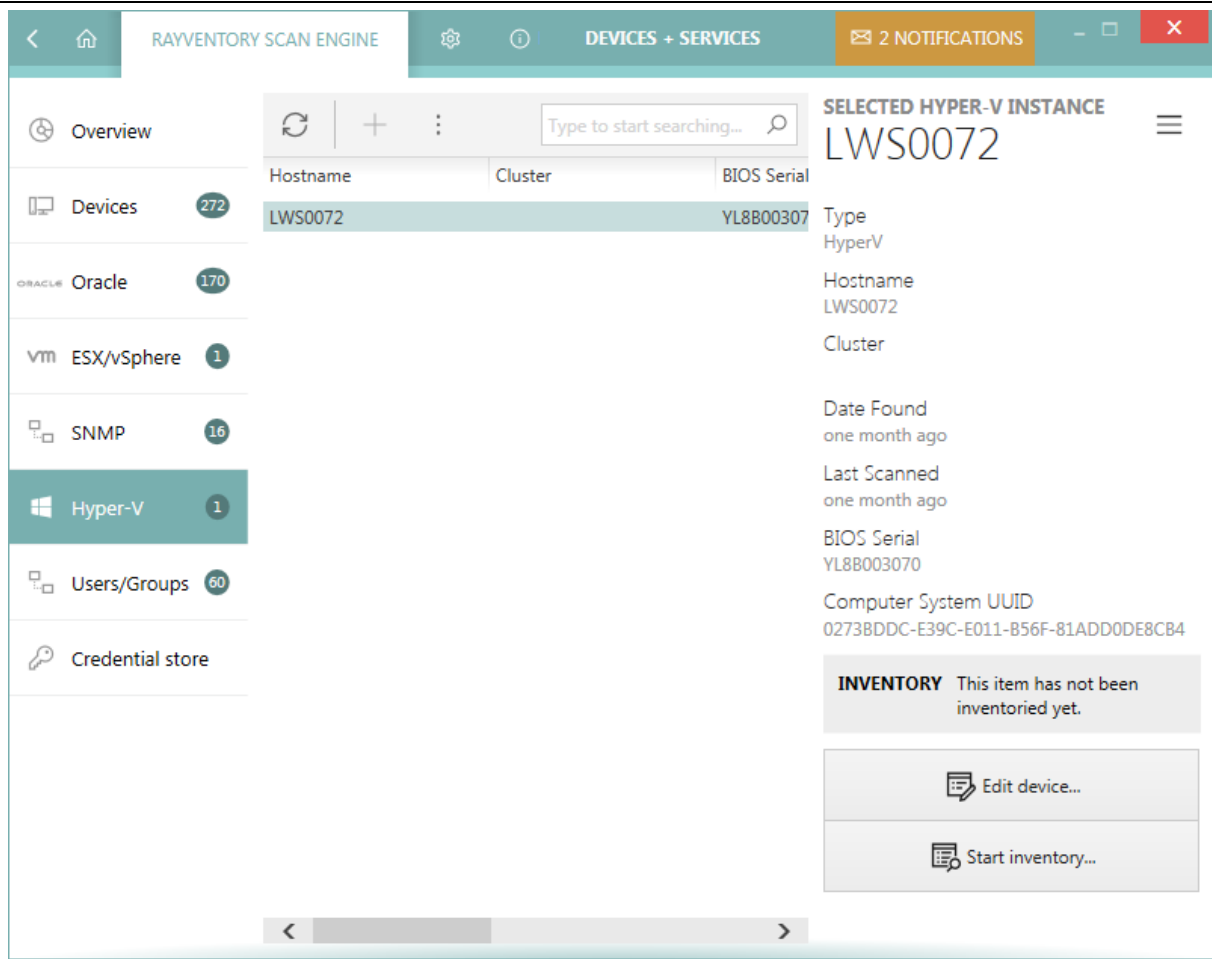


Note:

This operation is irreversible. Any existing inventory files which were assigned to the connection stay, though.

Hyper-V

This view aggregates discovery and inventory data of existing Hyper-V connections.



This screen will be populated automatically. Therefore, it is not possible to add, edit, or remove any entries. This screen only shows the data that has been gathered from the OS related inventory data. The data will be present in RayVentory Server after an OS inventory import no matter if an explicit Hyper-V inventory has been executed or not.

Users / Groups

This view lists all imported **Users** and **Groups**.

RAYVENTORY SCAN ENGINE

DEVICES + SERVICES

2 NOTIFICATIONS

Overview

Type

Name

Path

Member Of

Members

Created

Attributes

Devices

272

Group

Administrators

LDAP://CN=Administrat...

CN=Domain Admins,CN=...

5/14/2019

ORACLE

170

Group

Users

LDAP://CN=Users,CN=B...

CN=SomeUser,CN=User...

5/14/2019

Group

Guests

LDAP://CN=Guests,CN=...

CN=Domain Guests,CN=...

5/14/2019

Group

Print Operators

LDAP://CN=Print Operat...

5/14/2019

Group

Backup Operators

LDAP://CN=Backup Ope...

5/14/2019

VM ESX/vSphere

1

Group

Replicator

LDAP://CN=Replicator,C...

5/14/2019

Group

Remote Desktop Users

LDAP://CN=Remote Des...

5/14/2019

SNMP

16

Group

Network Configuration...

LDAP://CN=Network Co...

5/14/2019

Group

Performance Monitor Us...

LDAP://CN=Performance...

5/14/2019

Group

Performance Log Users

LDAP://CN=Performance...

5/14/2019

Hyper-V

1

Group

Distributed COM Users

LDAP://CN=Distributed...

5/14/2019

Group

IIS_IUSRS

LDAP://CN=IIS_IUSRS,CN...

CN=S-1-5-17,CN=Foreig...

5/14/2019

Group

Cryptographic Operators

LDAP://CN=Cryptograph...

5/14/2019

Group

Event Log Readers

LDAP://CN=Event Log R...

5/14/2019

Users/Groups

60

Group

Certificate Service DCOM...

LDAP://CN=Certificate S...

5/14/2019

Group

RDS Remote Access Serv...

LDAP://CN=RDS Remote...

5/14/2019

Group

RDS Endpoint Servers

LDAP://CN=RDS Endpoi...

5/14/2019

Group

RDS Management Servers

LDAP://CN=RDS Manag...

5/14/2019

Group

Hyper-V Administrators

LDAP://CN=Hyper-V Ad...

5/14/2019

Group

Access Control Assistanc...

LDAP://CN=Access Contr...

5/14/2019

Group

Remote Management Us...

LDAP://CN=Remote Ma...

5/14/2019

Group

Server Operators

LDAP://CN=Server Oper...

5/14/2019

Group

Account Operators

LDAP://CN=Account Op...

5/14/2019

Group

Pre-Windows 2000 Com...

LDAP://CN=Pre-Window...

CN=S-1-5-11,CN=Foreig...

5/14/2019

Group

Incoming Forest Trust Bu...

LDAP://CN=Incoming Fo...

5/14/2019

Group

Windows Authorization...

LDAP://CN=Windows Au...

CN=S-1-5-9,CN=Foreign...

5/14/2019

Group

Terminal Server License...

LDAP://CN=Terminal Ser...

5/14/2019

User

Administrator

LDAP://CN=Administrat...

CN=MGS Distributors,C...

5/14/2019

User

Guest

LDAP://CN=Guest,CN=U...

CN=Guests,CN=BuiltIn,D...

5/14/2019

User

krbtgt

LDAP://CN=krbtgt,CN=U...

CN=Denied RODC Passw...

5/14/2019

User

Test User

LDAP://CN=Test User,CN...

5/14/2019

User

SomeUser

LDAP://CN=SomeUser,C...

CN=Domain Admins,CN...

5/14/2019

All **Users** and all **Groups** imported from the **Active Directory** are shown here. It is possible to add, remove, and edit entries. Two optional text files which can be used to customize the Active Directory Users and Groups features are available in RayVentory Scan Engine.

It is possible to import and view additional attributes from the Active Directory by creating the optional files ADImportUserAttributes.lst and ADImportGroupAttributes.lst and adding the attribute names to the files (one per line).

The following links contain lists of attribute names that can be used for the attribute file lists:

- <https://docs.microsoft.com/en-us/windows/desktop/adschema/c-user#windows-2000-server-attributes>
- <https://docs.microsoft.com/en-us/windows/desktop/adschema/c-group#windows-2000-server-attributes>

Import AD Users and Groups Wizard

The **Import AD Users and Groups** wizard will show a preview of the data for the current query.

← BACK

IMPORT AD USERS AND GROUPS

✖

○ Query

○ Summary

○ Progress

○ Finished

Query

DC=candy,DC=mountain

...

PREVIEW OF THE CURRENT QUERY

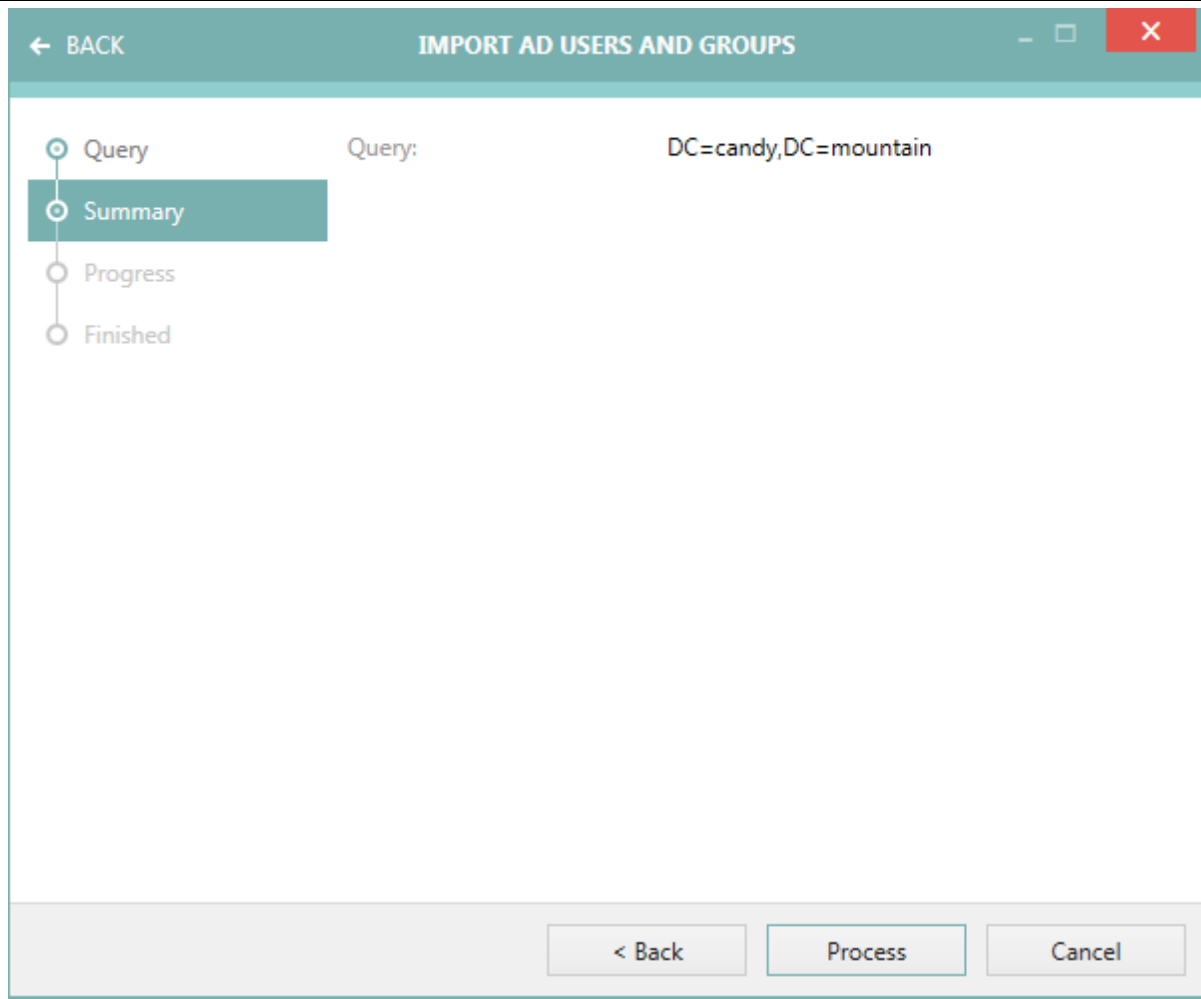
| Type | Name | Path | Members |
|-------|--------------------|-------------------|----------|
| User | Administrator | LDAP://CN=Ad... | |
| User | Guest | LDAP://CN=Gue... | |
| User | krbtgt | LDAP://CN=krbt... | |
| User | Test User | LDAP://CN=Test... | |
| User | SomeUser | LDAP://CN=Som... | |
| User | Big BC. Chungus | LDAP://CN=Big... | |
| Group | WinRMRemote... | LDAP://CN=Win... | |
| Group | HelpLibraryUpda... | LDAP://CN=Help... | |
| Group | SQLServer2005S... | LDAP://CN=SQL... | |
| Group | Administrators | LDAP://CN=Ad... | CN=Doma |
| Group | Users | LDAP://CN=User... | CN=Somel |
| Group | Guests | LDAP://CN=Gue... | CN=Doma |

< Back

Next >

Cancel

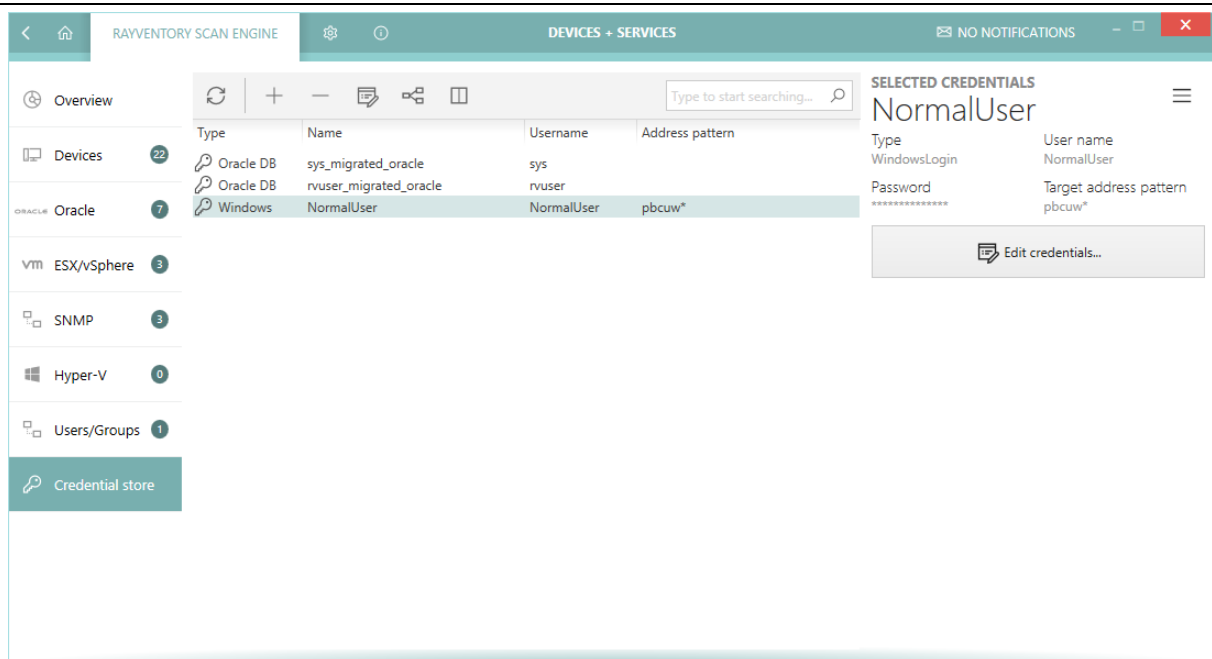
To create a query from the Active Directory browser, click on the **Browse** button [...]. To continue to the next step, click on the **Next >** button.



In the **Summary** step of the wizard the query will be shown once more. Click on the **Process** button to execute the query.

Credential Store

In the credential store all credentials that are used by RayVentory Scan Engine are consolidated. The credentials are stored in an encrypted form. By default, the store uses its default encryption key. It is possible to use the custom encryption key as described in the **Settings > General** section.



The credential store distinguishes between different types of credentials:

- Windows
- SSH
- Oracle
- vSphere / ESX
- SNMP

Different types of credentials are used for different inventory operations and the Windows credentials are additionally used for the upload operation.

All credentials have a logical name that is used in the drop-down menus which are available in the connection properties dialogs. The drop-down menu in the **Credentials** field which is present in all connection properties dialogs is used to configure the preferred credentials for a connection. This means that the selected credentials are the first ones that are being tested. If no preferred credentials have been defined, the credentials are tested in the order in which they are shown in the credential store screen and filtered by matching the optional **Target Address Pattern** field to all credentials.

All credentials have a **Target Address Pattern** field. This field is optional and it is used to filter the credentials for the usage with a certain host or set of hosts. If the field is empty, then the credentials are applicable for all hosts. The field can be set to a specific hostname or address and will then be applied to this hostname or address only. It is also possible to use a regular expression for the **Target Address Pattern** in order to match multiple hosts that fit this expression.

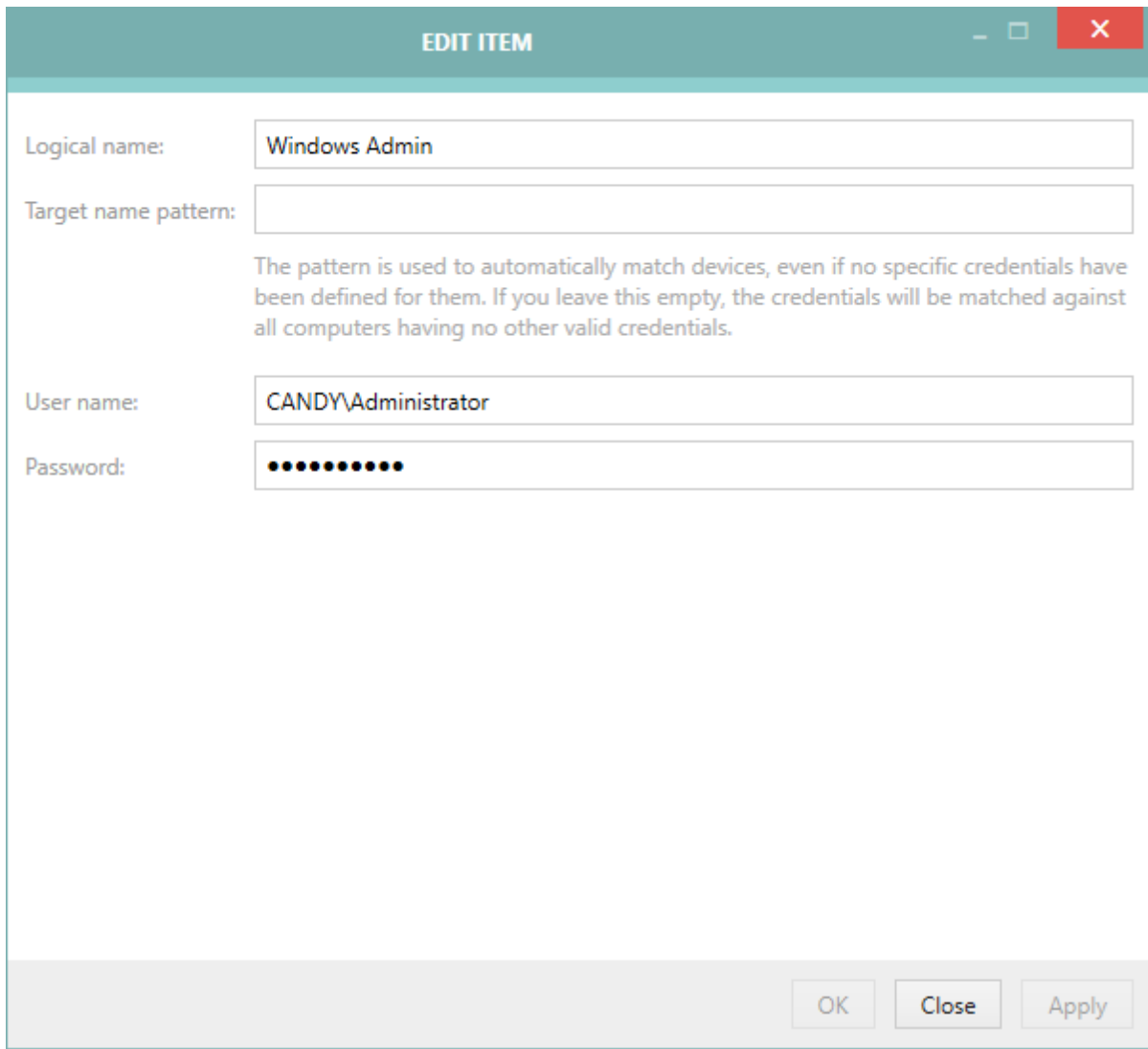
Credential Type Windows

When running an inventory on a remote Windows host, the Windows credential are used for authentication.

**Note:**

There is an option to use a Windows type login for authentication for the upload operation.

Windows credentials consist of a user name and a password. The user name might include a domain name (NT domain name). An example for this is `mydomain/myusername`.



The screenshot shows a dialog box titled "EDIT ITEM" with a teal header bar containing standard window controls. The dialog has a light gray background and contains the following fields and text:

- Logical name:** A text box containing "Windows Admin".
- Target name pattern:** An empty text box. Below it, a note states: "The pattern is used to automatically match devices, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all computers having no other valid credentials."
- User name:** A text box containing "CANDY\Administrator".
- Password:** A text box filled with 12 black dots.

At the bottom right, there are three buttons: "OK", "Close", and "Apply".

Credential Type SSH

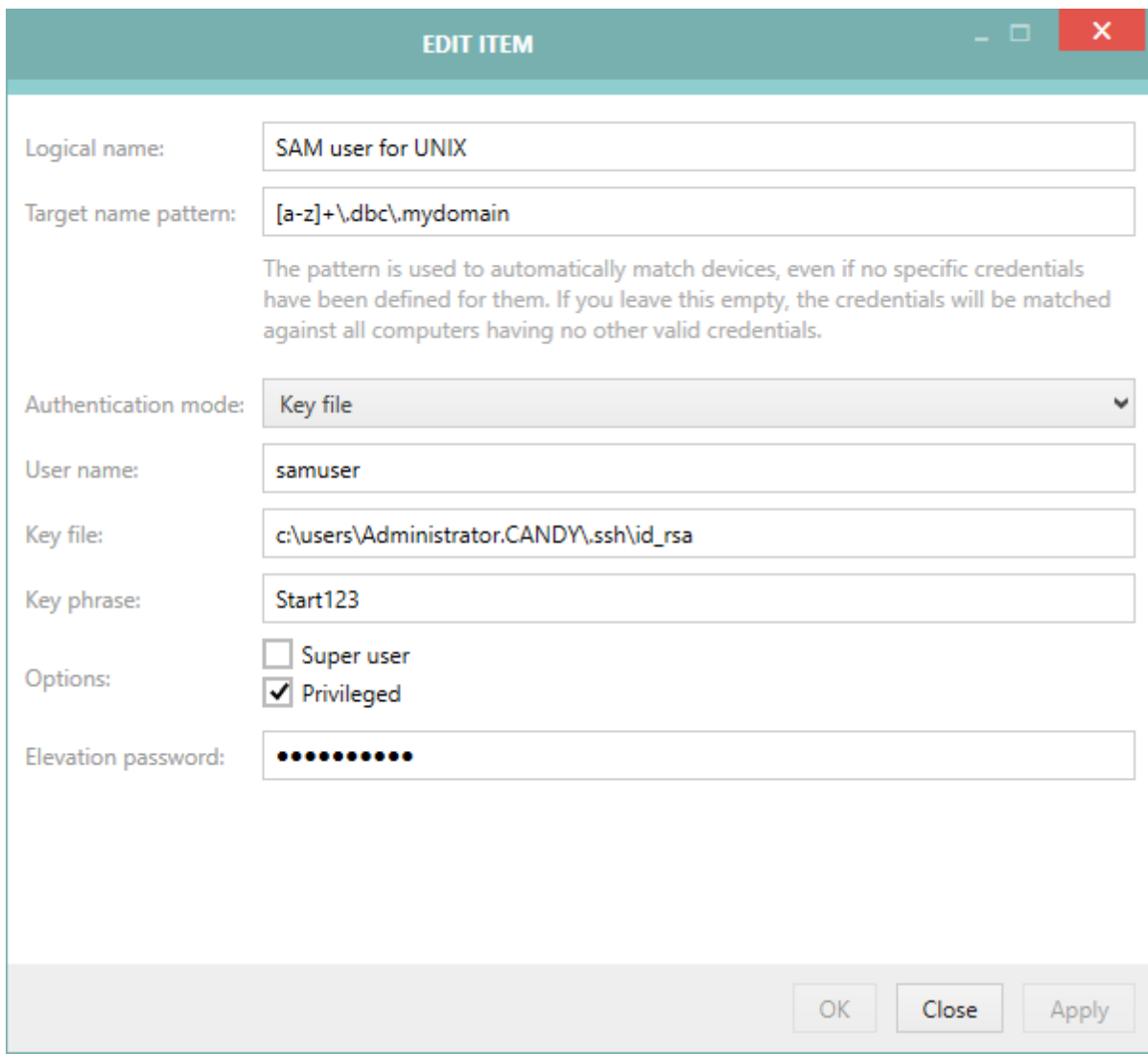
The SSH credentials are used for a remote inventory on Linux and unix-like platforms.

The credential store offers three different types of credentials for SSH authentication:

- user name / password
- user name / private key
- user name / private key file

A private key or a private key file can be secured by a passphrase. A passphrase can be set in the **Authentication Key Passphrase** field. The passphrase is stored in the credential store in an encrypted form. A private key without a passphrase is stored in the credential store in an encrypted form.

The SSH credentials include the information on how the elevation of privileges is done or if the user specified for credentials is assumed to be a super user on the target host. It is possible to specify a password for the elevation of privileges for targets where, for example, sudo would prompt for a password when using it to run a command with elevated privileges.



The screenshot shows a window titled "EDIT ITEM" with a teal header bar containing standard window controls. The main area contains several fields and options for configuring an item:

- Logical name:** A text field containing "SAM user for UNIX".
- Target name pattern:** A text field containing "[a-z]+\dbc\mydomain". Below this field is a descriptive text: "The pattern is used to automatically match devices, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all computers having no other valid credentials."
- Authentication mode:** A dropdown menu currently set to "Key file".
- User name:** A text field containing "samuser".
- Key file:** A text field containing "c:\users\Administrator.CANDY\.ssh\id_rsa".
- Key phrase:** A text field containing "Start123".
- Options:** Two checkboxes: "Super user" (unchecked) and "Privileged" (checked).
- Elevation password:** A text field containing ten black dots, indicating a masked password.

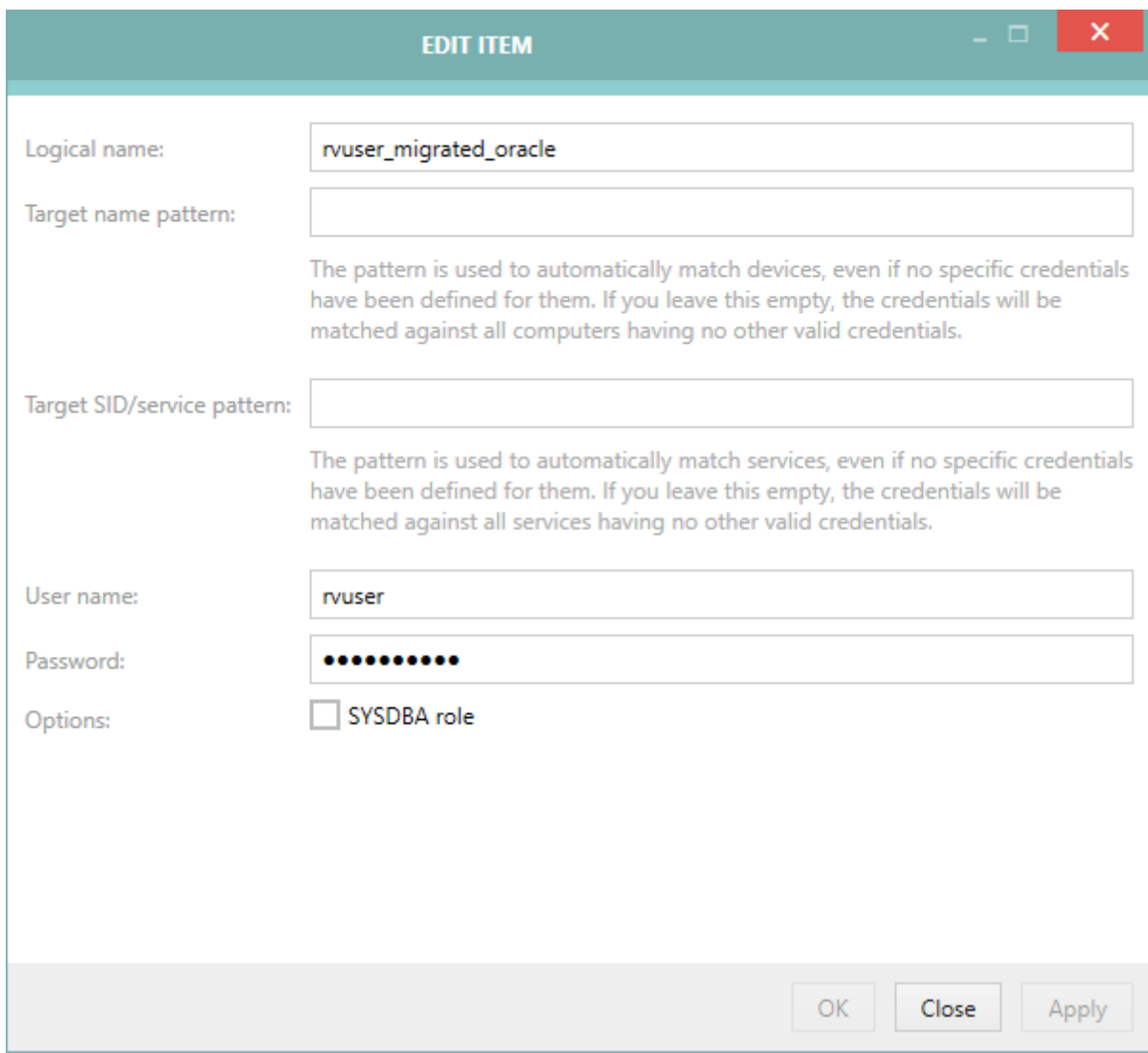
At the bottom right of the dialog are three buttons: "OK", "Close", and "Apply".

Credential Type Oracle

The Oracle credentials are used for inventory operations on Oracle databases and are for using the Review Lite Script or DFUS script support.

The credentials consist of a user name / password pair. The password is optional. For special logins like the build-in user SYS the flag SYSDBA role can be set.

In addition to the Target Address Pattern, the DB credentials have another optional field. The **Target SID/Service Pattern** field which matches with the server name or the SID. Just like the Target Address Pattern, this field may stay empty, a concrete SID / service name, or a regular expression which matches multiple SIDs / service names can be enter.



The screenshot shows a dialog box titled "EDIT ITEM" with a teal header bar containing standard window controls. The dialog contains several input fields and a checkbox:

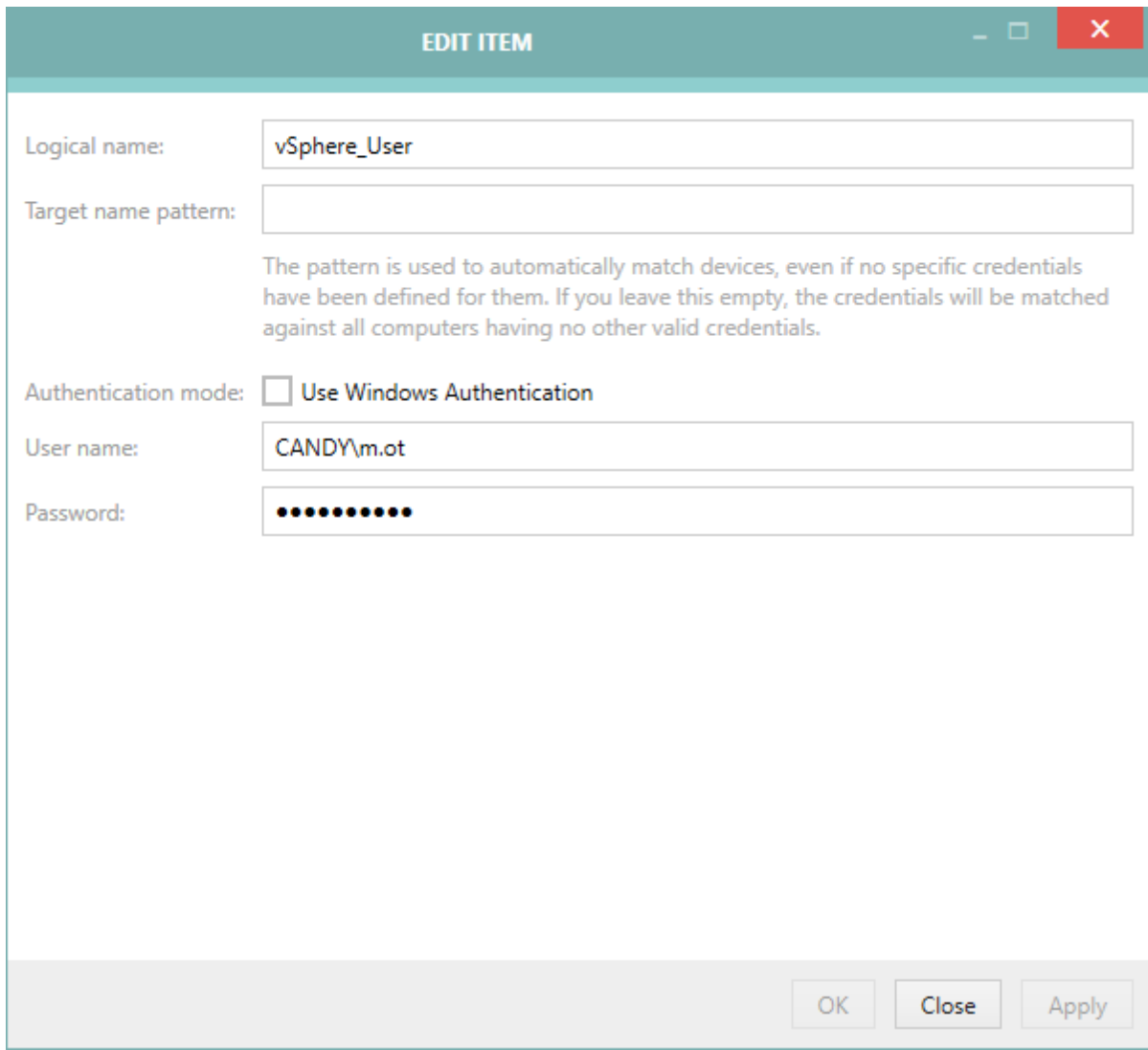
- Logical name:** A text field containing "rvuser_migrated_oracle".
- Target name pattern:** An empty text field. Below it is a descriptive text: "The pattern is used to automatically match devices, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all computers having no other valid credentials."
- Target SID/service pattern:** An empty text field. Below it is a descriptive text: "The pattern is used to automatically match services, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all services having no other valid credentials."
- User name:** A text field containing "rvuser".
- Password:** A text field with masked characters (dots).
- Options:** A checkbox labeled "SYSDBA role" which is currently unchecked.

At the bottom right of the dialog are three buttons: "OK", "Close", and "Apply".

Credential Type vSphere / ESX

The vSphere / ESX credentials are used for authentication when running an inventory on a vSphere / ESX infrastructure.

A set of vSphere / ESX credentials consists of a user name and a password. In case that the implementation involves a Windows Domain Controller, the user name may include a domain name (NT domain name). An example for this is `mydomain/myusername`. User names which are local to a vSphere / ESX infrastructure should work without any prefix or suffix like `username@esxhost`.



EDIT ITEM

Logical name:

Target name pattern:

The pattern is used to automatically match devices, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all computers having no other valid credentials.

Authentication mode: ☐ Use Windows Authentication

User name:

Password:

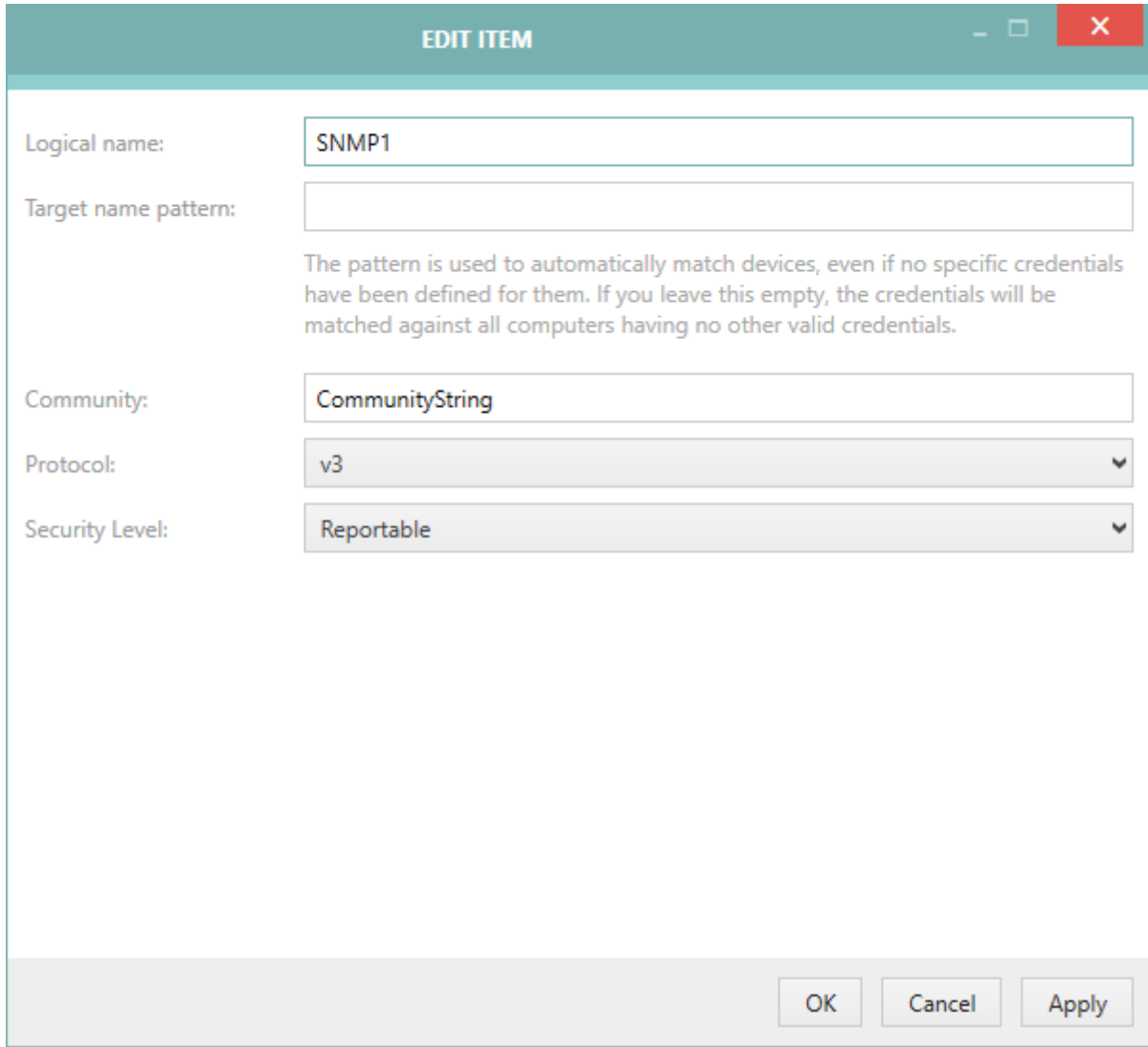
OK Close Apply

It is possible to set the **Use Windows session credentials** flag instead of entering a username and password if all the following conditions are met:

- It is an implementation that allows for authentication to be delegated to a Windows Domain Controller.
- The RayVentory host is part of the domain.
- The user that runs RayVentory Scan Engine and its scheduling service is a domain user.

Credential Type SNMP

The SNMP credentials are used for authentication when running an inventory on an SNMP connection.



The screenshot shows a dialog box titled "EDIT ITEM" with a teal header bar containing standard window controls. The dialog has a light gray background. It contains several input fields and dropdown menus:

- Logical name:** A text input field containing "SNMP1".
- Target name pattern:** An empty text input field. Below it is a descriptive text: "The pattern is used to automatically match devices, even if no specific credentials have been defined for them. If you leave this empty, the credentials will be matched against all computers having no other valid credentials."
- Community:** A text input field containing "CommunityString".
- Protocol:** A dropdown menu with "v3" selected.
- Security Level:** A dropdown menu with "Reportable" selected.

At the bottom right of the dialog, there are three buttons: "OK", "Cancel", and "Apply".

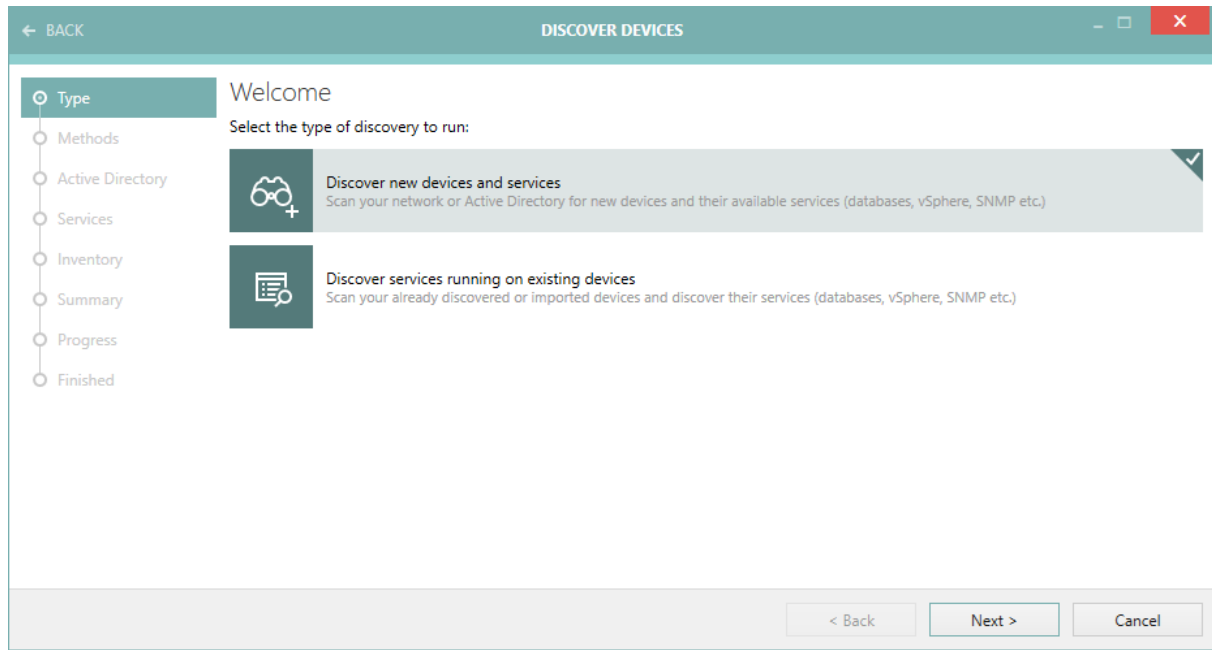
Adding New Credentials

To add new credentials press the **+** button in the toolbar or right click the credentials grid and choose **Add...** from its context menu (visible after pressing **Right Mouse Button**).

RayVentory Scan Engine uses a wizard-based approach guiding you through all steps and ensuring the correctness of gathered information.

Discovery Wizard

The **Discovery** wizard is the primary tool for finding running devices and services and / or importing them from Active Directory.



There are two mutually exclusive options available in the first page of the **Discovery** Wizard.

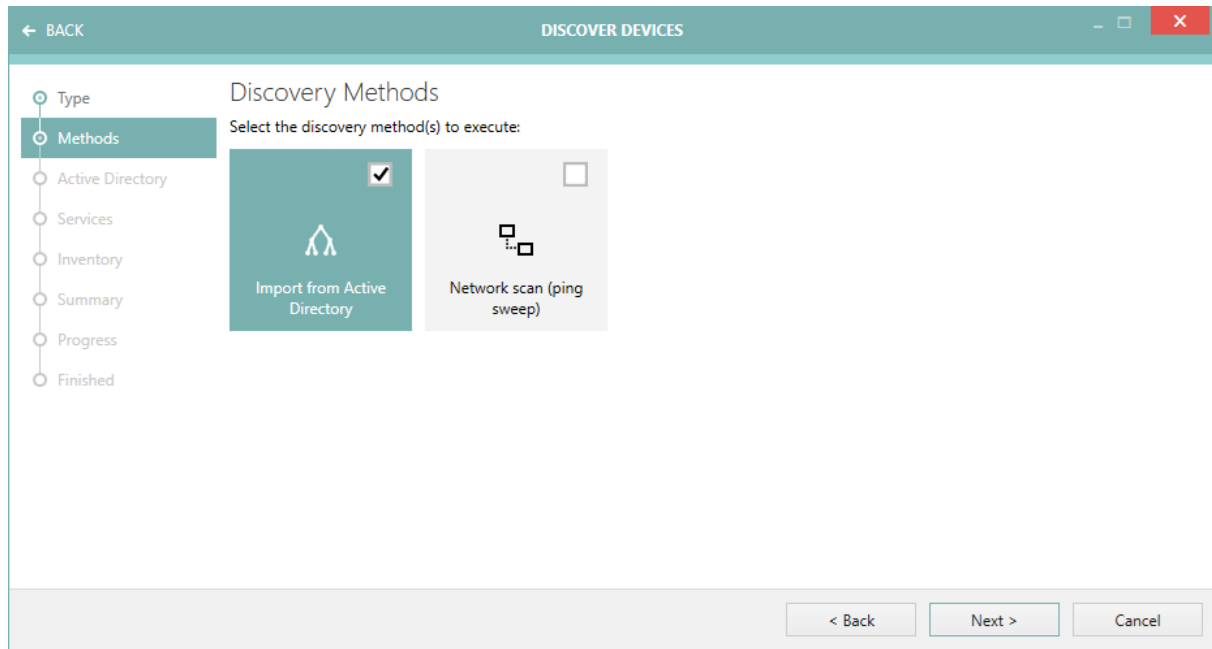
- **Discover new devices and services** - Select this option to perform a discovery of new devices and / or services.
- **Discover services running on existing devices** - Select this option to perform a discovery of services on devices that have already been added to the **Devices** lists.

Depending on the choice selected, the number and the content of wizard pages that are following may vary.

Discovering New Devices

This section describes the steps of the **Discovery** wizard, provided that the user selected the option **Discover new devices and services** as his required Discovery type.

Methods



There are two complementary options which determine the source of the data for the discovery operation. In order to continue to the next page, at least one of them needs to be selected.

- **Import from Active Directory** - use this option to import devices from your Active Directory. You will be able to specify the domain and the filtering in the next step.
- **Network scan (ping sweep)** - use this option to perform a ping sweep on your network. You will be able to specify the IP address range in the next step.



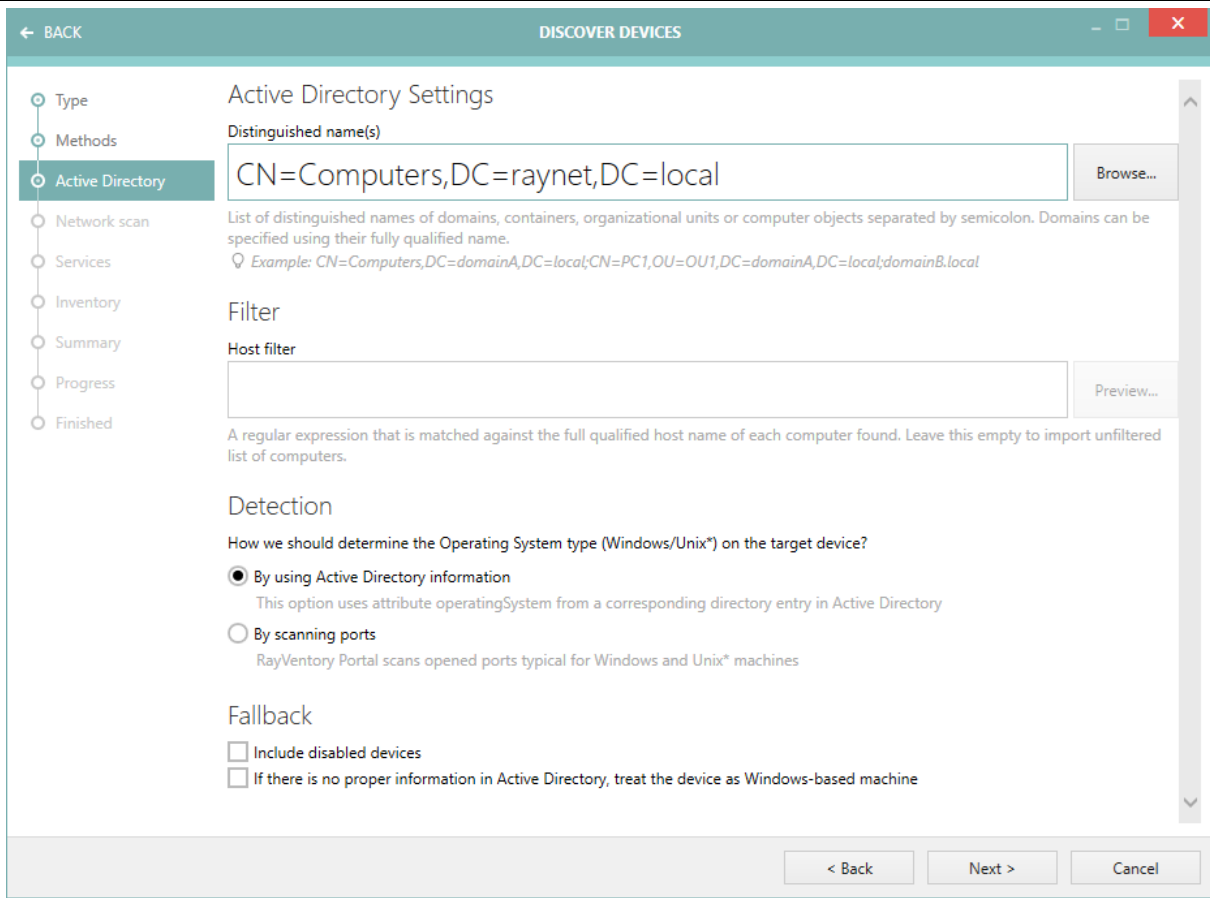
Note:

The network scan tries to contact each device in the specified network range and checks for predefined ports to determine the device family (Windows or UNIX). Contact your network administrator before executing scans on a wide range of addresses.

Selecting **Network scan (ping sweep)** adds an extra page **Network scan** to the list, where the user can configure the IP address range. Selecting **Import from Active Directory** adds an extra page **Active Directory** where the user can configure scanned domain(s), filters, and default behaviors. The options are complementary which means that both can be selected at the same time. In this case, RayVentory Scan Engine merges information from both sources and discovers devices respecting both ping sweep results and extra bits of information from the Active Directory.

Active Directory

This page is shown only if the user selected the **Import from Active Directory** option on the previous screen.



The screenshot shows a software window titled "DISCOVER DEVICES" with a sidebar on the left containing a tree view with items: Type, Methods, Active Directory (selected), Network scan, Services, Inventory, Summary, Progress, and Finished. The main area is titled "Active Directory Settings". It contains a text field for "Distinguished name(s)" with the value "CN=Computers,DC=raynet,DC=local" and a "Browse..." button. Below this is a description: "List of distinguished names of domains, containers, organizational units or computer objects separated by semicolon. Domains can be specified using their fully qualified name. Example: CN=Computers,DC=domainA,DC=local;CN=PC1,OU=OU1,DC=domainA,DC=local;domainB.local". There is a "Filter" section with a "Host filter" text field and a "Preview..." button. Below that is a "Detection" section with the question "How we should determine the Operating System type (Windows/Unix*) on the target device?". It has two radio buttons: "By using Active Directory information" (selected) and "By scanning ports". The "By scanning ports" option has a sub-note: "RayVentory Portal scans opened ports typical for Windows and Unix* machines". There is a "Fallback" section with two checkboxes: "Include disabled devices" and "If there is no proper information in Active Directory, treat the device as Windows-based machine". At the bottom are three buttons: "< Back", "Next >", and "Cancel".

Distinguished Name(s)

The domain name (dotted domain) or a LDAP query sting into the Active Directory Domain field. It is possible to enter multiple domain names or LDAP queries by separating them using a semicolon. You can also press the **Browse...** button to open a simple LDAP browser, that enables you to visually select the required container.



Be aware:

The LDAP browser supports multiselection. Hold thr **CTRL** button when selecting items to select more than one item at once.

Filter

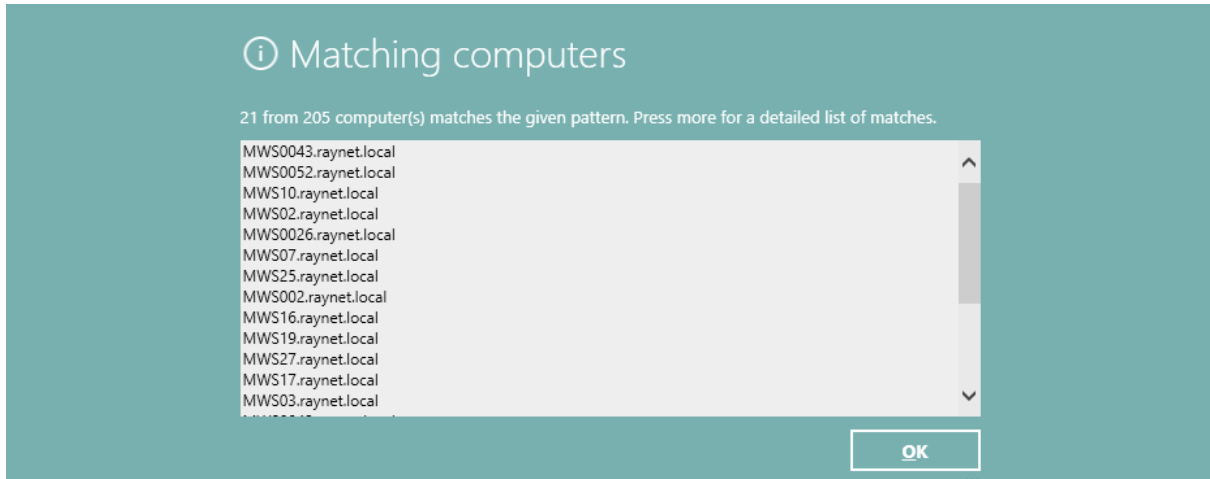
This is an optional field, which can be used to perform extra filtering of devices. Regular expressions are supported. If the field is left empty, all entries from the Active Directory will be considered when performing the import, otherwise only the devices with matching full qualified host names will be imported.



Be aware:

Regular expressions can be complex to write and validate. If an invalid regular expression is entered, nothing will be imported. Also, pay attention to the fact that some characters and sequenced may have special meaning, for example a dot "." means any character inside the Regular Expression. To escape special characters, prepend them with backslash (\).

You can always test the filter string by pressing the **Preview...** button. It will scan the currently selected node(s) and report back with a list of devices that matched your filter. For example, for a filter `^mws` the following is reported:



(as a side note: `^mws` means any computer name that starts with the string `mws`).

The regular expression engine is case insensitive and does not enforce string boundaries (you have to enforce them by `^` and `$` respectively).

Detection

This setting determines how RayVentory Scan Engine figures out which Operating System is installed on the imported device. There are two different ways of solving this:

- Select **By using Active Directory information** to use Active Directory member attributes and resolve Operating System from their values. This is recommended if you want to avoid physical scans of the network and it is much faster than the other method.
- Select **By scanning ports** if your Active Directory does not contain the required information OR if you want the results to be reliable based on the actual status of open / closed ports. If this option is selected, RayVentory Scan Engine checks whether typical ports for SSH or RPC are opened and based on their status the right device family is assigned.



Note:

Option **Scanning ports** tries to contact each device in the specified network range and checks for predefined ports to determine the device family (Windows or UNIX). Contact your network administrator before executing scans on wide range of addresses.

Fallback

These options define the behavior of RayVentory Scan Engine should an unrecognized or disabled device be found.

- **Include disabled devices** - Select this option to include disabled devices. If you leave it unchecked, the devices that are disabled in Active Directory will not be imported

(recommended).

- **If there is no proper information in Active Directory, treat the device as Windows-based machine** - Select this option to automatically fallback to Windows, if the information in Active Directory is not sufficient to determine the device family. If you uncheck this option, the devices will be imported as "unknown" types. This option is only available if the detection is set to **By using Active Directory information**.
- **If device is unreachable, treat it as Windows-based machine** - Select this option for an automatically fallback to Windows if the set of open / closed ports was not enough to identify the device family. If you uncheck this option, the devices will be imported as "unknown" types. This option is only available if the detection is set to **By scanning ports**.

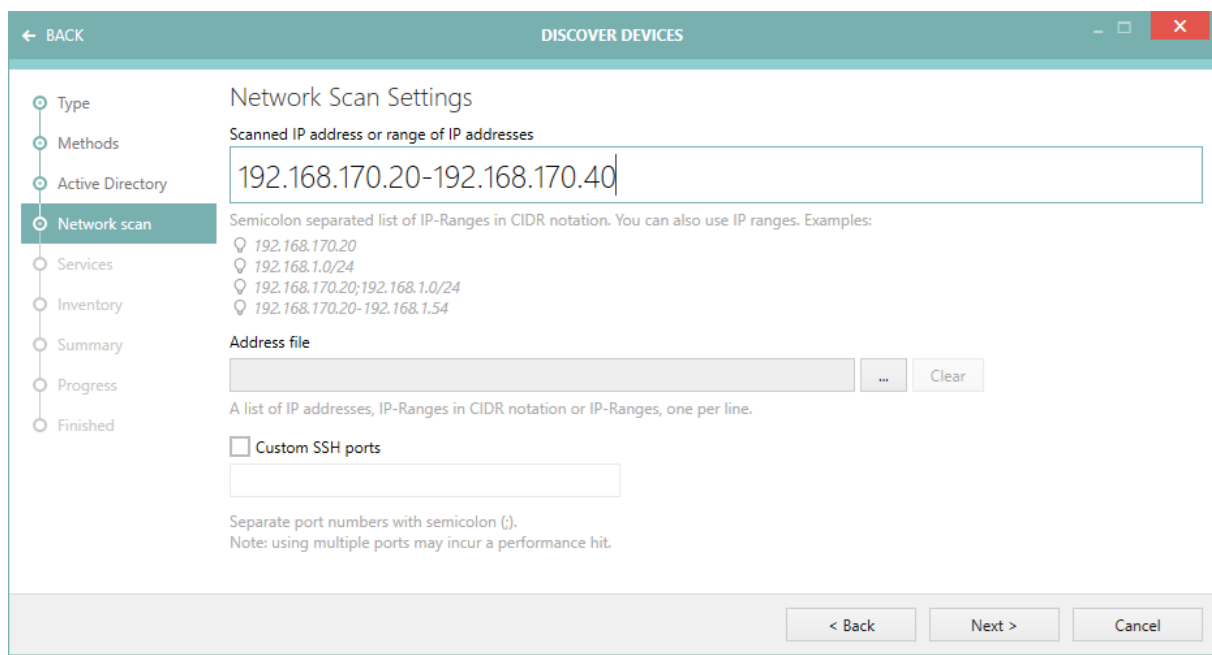


Note:

Make sure that the current user has read permissions for the Active Directory.

Network Scan

This page is shown only if the user selected the **Network scan (ping sweep)** option on the previous screen.



Scanned IP Address or Range of IP Addresses

This is a value which is represented in one of the following notations:

- CIDR notation (for example 192.168.170.0/24),
- Single IP address (for example 192.168.170.21),
- IP address range (for example 192.168.170.1-192.168.172.200),
- Any combination of previous three separated by semicolon.

The discovery will try to reverse-lookup the DNS name for the IP addresses that have been found responding during the scan.

CIDR Examples

| CIDR | First IP | Last IP |
|------------------|---------------|----------------|
| 192.168.40.0/24 | 192.168.40.0 | 192.168.40.255 |
| 192.168.40.10/32 | 192.168.40.10 | 192.168.40.10 |

Address file

When specified, the list of IP addresses is read from a plain text file (where each line is a separate address to scan).

Custom SSH ports

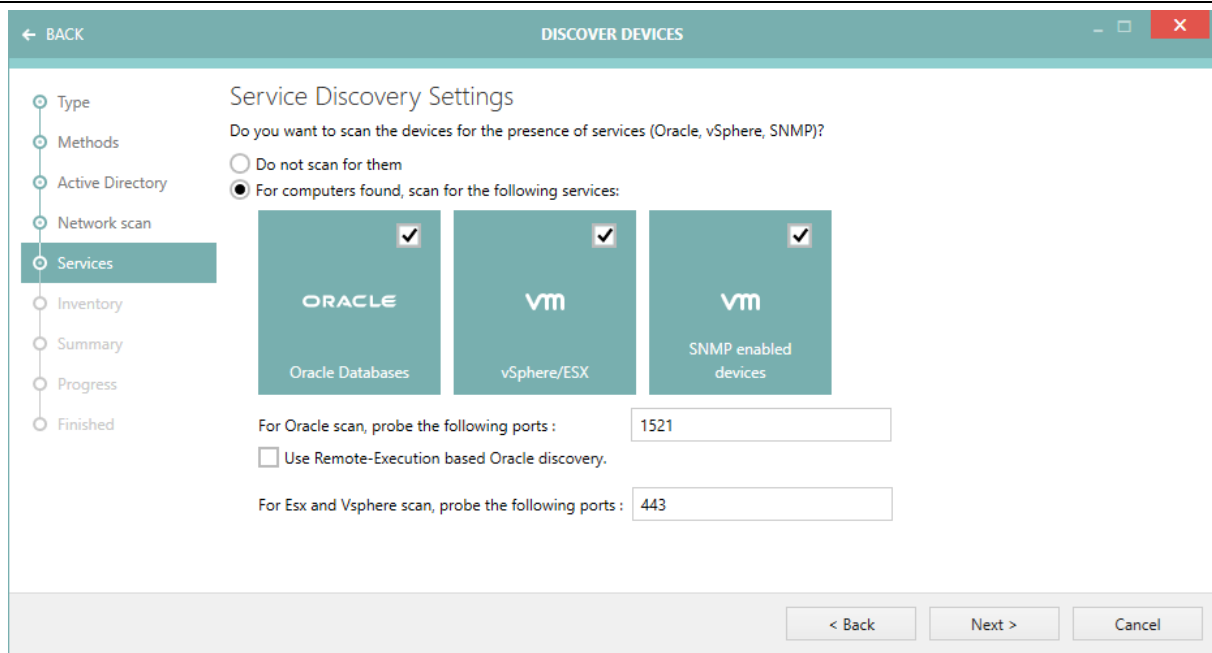
If a non-standard SSH port should be used for non-Windows devices, it should be entered here. More than one value can be entered by using a semicolon.

**Note:**

Providing more than one port may have negative impact on the scan speed.

Services

This page contains a configuration of an optional step, allowing the user to perform extra-scanning of services likes vSphere / ESX instances, SNMP devices, and Oracle databases on newly found devices.



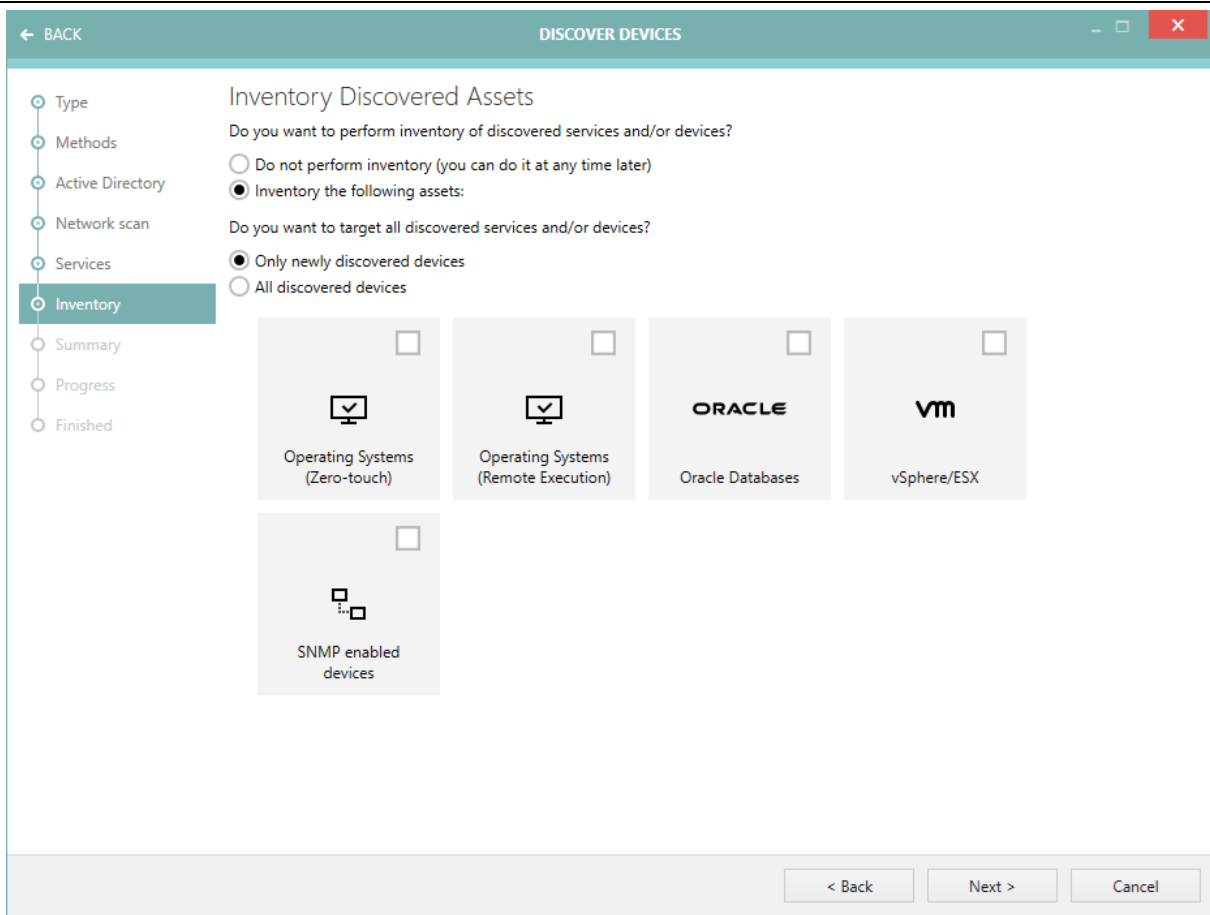
In order to scan for the services, tick the radio button **For computers found, scan the following services** and then select the options for the scan. The options are complementary, if the radio button is checked then at least one option must be enabled in order to continue. By default, none of these options are selected.

For Oracle inventory, there are two extra options on this page that can be configured:

- **For Oracle scan probe the following ports** - This is the port that will be used for probing for Oracle TNS listeners. Multiple ports can be probed by entering a list of port numbers, separated by commas or semicolons. If not defined otherwise, the default port **1521** is going to be used.
- **For Esx and vSphere scan, prove the following port:** This is the port that will be used for probing for vSphere. Multiple ports can be probed by entering a list of port numbers, separated by commas or semicolons. If not defined otherwise, the default port **443** is going to be used.
- **Use Remote-Execution based Oracle discovery** - By default, the discovery of the instance will be done in the "zero-touch" mode. Using this option enables remote execution of scanning utilities on a target server, which usually provides better scanning results. For more information about differences between inventory methods, refer to the Inventory Methods chapter.

Inventory

This page contains a configuration of an optional step which allows users to perform an inventory of software and hardware on newly found devices.



In order to scan the software and the hardware of new devices, tick the radio button **Inventory the following assets** and then select the options which are to be scanned. The options are complementary, if the radio button is checked then at least one option must be enabled in order to continue. By default, these options are not selected.

The number of options may vary, depending on which services were selected on the Services page. For the **Operating Systems** scan, two options are available:

- **Zero-touch** - which uses agentless, zero-touch execution leaving no traces on the target device,
- **Remote Execution** - which uses agentless execution of scanning utilities which run on the target device and report the results back to RayVentory Scan Engine.

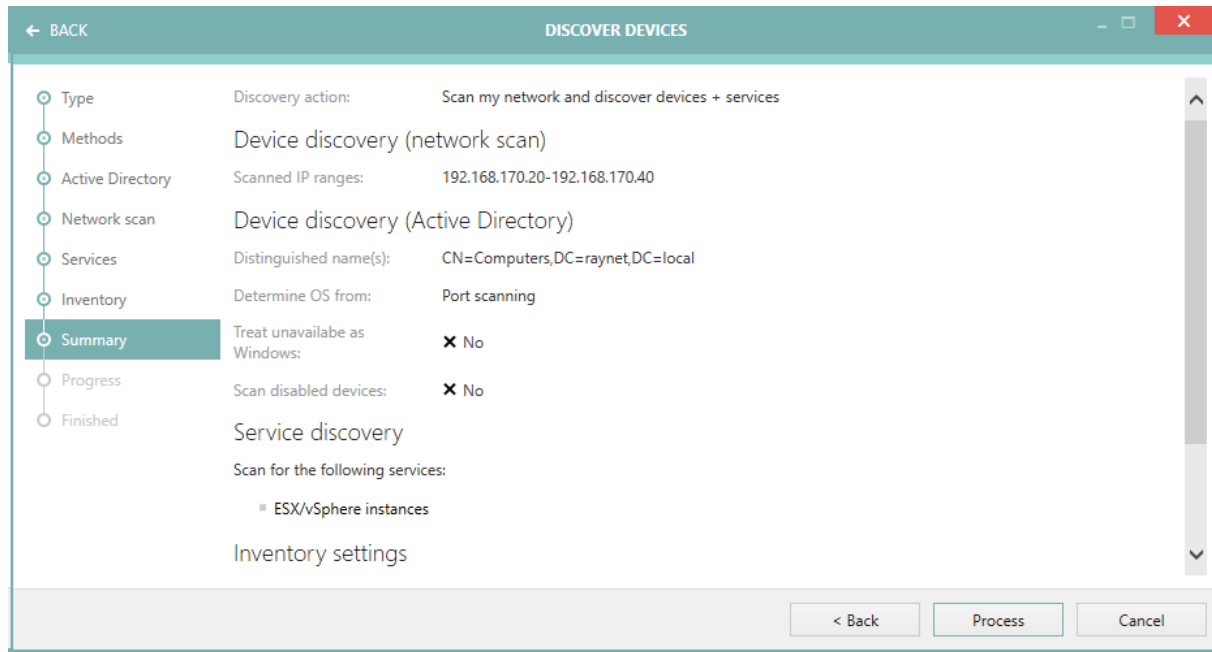
Since the devices that are to be found are not known when the wizards starts, the Zero-Touch and Remote-Execution are offered separately. For more controls on how an inventory is executed on target devices, we recommend to skip the inventory at the discovery page and instead let it run and discover the devices. Then, adjust the devices and / or settings and start the Inventory Wizard for fine-grained control on all inventory aspects.

If the optional scan for ESX / vSphere has been enabled, the option **Populate devices** will be shown. In the summary screen of the wizard this will be shown as **Populate the devices table**

from ESX/vSphere guests.

Summary

The **Summary** page shows the overview of all choices defined in the previous pages. You can click on the settings to go back to their corresponding places and change them.

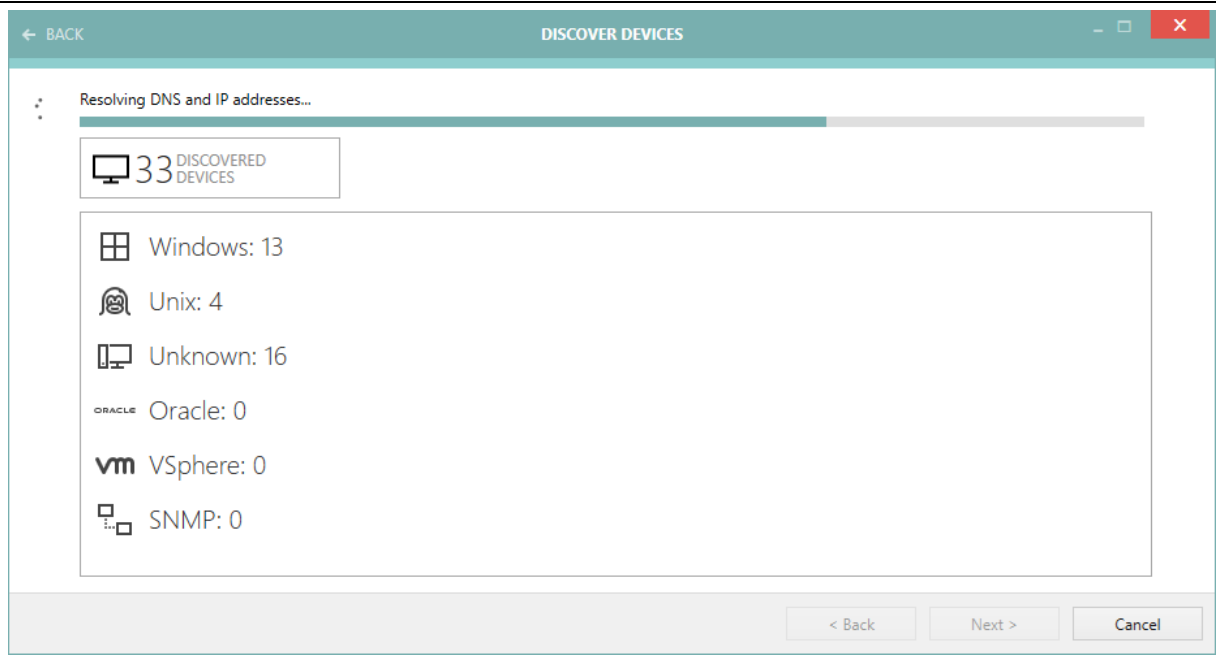


| DISCOVER DEVICES | |
|---|--|
| ← BACK | DISCOVER DEVICES |
| Type | Discovery action: Scan my network and discover devices + services |
| Methods | Device discovery (network scan) |
| Active Directory | Scanned IP ranges: 192.168.170.20-192.168.170.40 |
| Network scan | Device discovery (Active Directory) |
| Services | Distinguished name(s): CN=Computers,DC=raynet,DC=local |
| Inventory | Determine OS from: Port scanning |
| Summary | Treat unavailable as Windows: <input checked="" type="checkbox"/> No |
| Progress | Scan disabled devices: <input checked="" type="checkbox"/> No |
| Finished | Service discovery |
| | Scan for the following services: |
| | ▣ ESX/vSphere instances |
| | Inventory settings |
| <input data-bbox="1044 1035 1172 1066" type="button" value=" < Back "/> <input data-bbox="1187 1035 1315 1066" type="button" value=" Process "/> <input data-bbox="1331 1035 1459 1066" type="button" value=" Cancel "/> | |

Press **Process** to start the discovery.

Progress and Results

The **Progress** page shows the current activity and real-time results of the scanning.

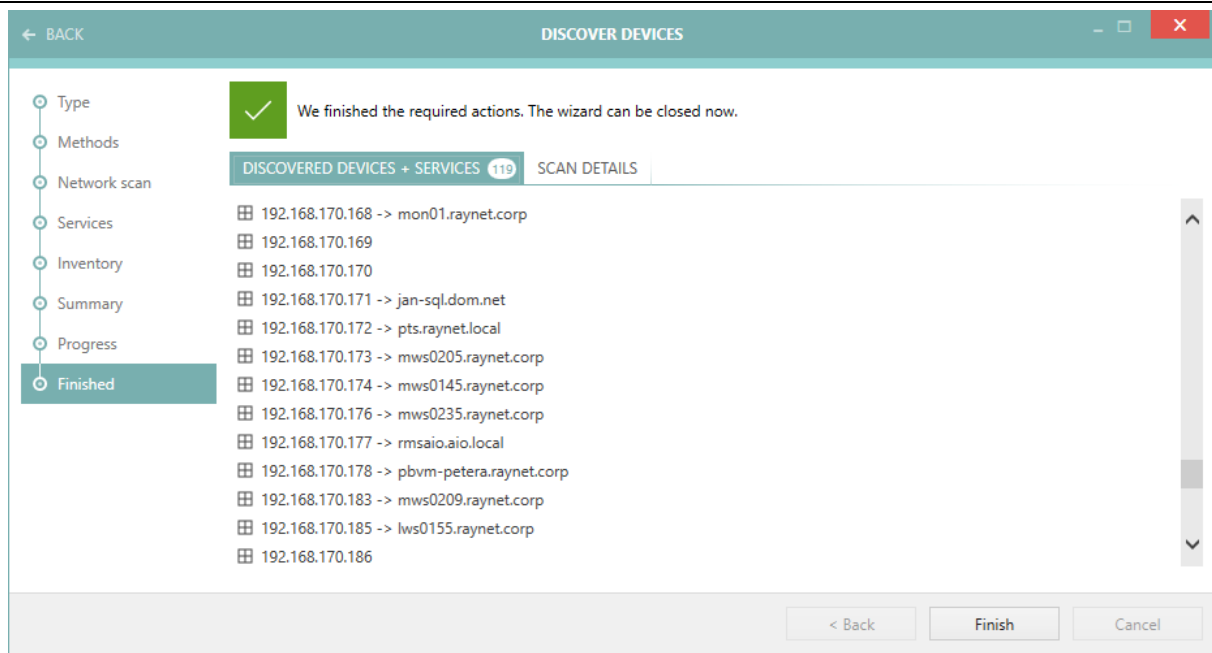


The grid shows the IP addresses and / or host names of devices (depending on the options selected on previous pages) and it runs basically in two steps:

1. Ping sweep and / or Active Directory import of devices.
2. DNS lookup / Reverse lookup.

Once an IP address is resolved to a host name (or vice versa), the content of the entry is updated (see picture above for resolved IP addresses). The icon next to the IP address identifies the device family (Windows or UNIX).

Once the work is done, a confirmation page is shown:



This shows the overview of the discovered devices and extra scan details plus the detailed descriptions in case of errors during scanning.

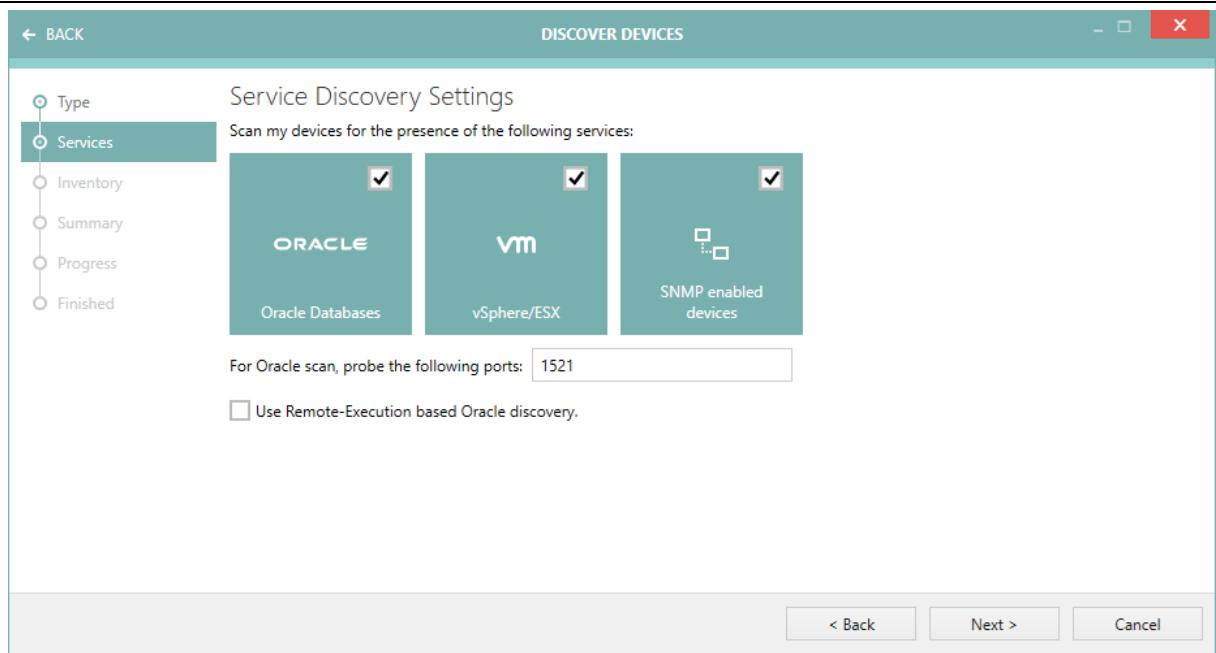
Once the wizard is closed, all discovered devices are automatically imported into the **Devices** view. All discovered services are imported to the **Oracle**, **SNMP**, or **vSphere / ESX** views.

Discovering Services on Existing Devices

This section describes the steps of the **Discovery** wizard, provided that the user selected the option **Discover services running on existing devices** as his required Discovery type.

Services

This page contains a configuration of an optional step, allowing the user to perform scanning of services likes vSphere / ESX instances, SNMP devices, and Oracle databases on already existing devices.



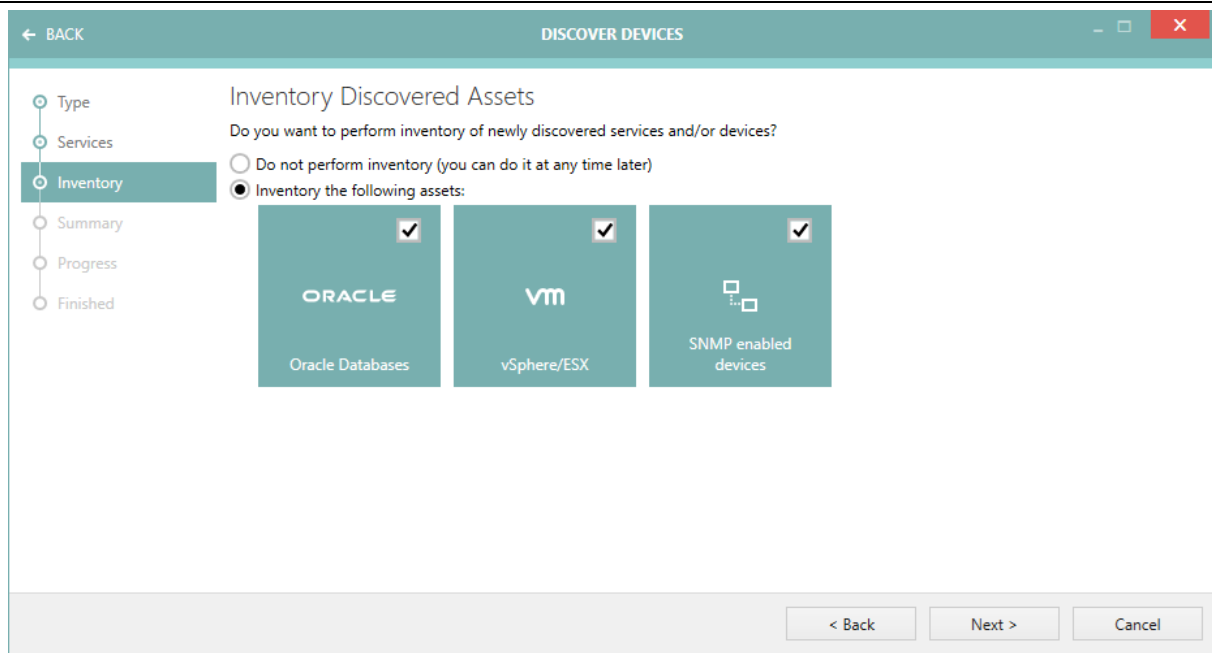
The options are complementary, if the radio button is checked then at least one option must be enabled in order to continue. By default, these options are not selected.

For the Oracle inventory, there are two extra options to configure on this page:

- **For Oracle scan, probe the following ports** - This is the port that will be used for probing for Oracle TNS listeners. Multiple ports can be probed by entering a semicolon separated list of port numbers. If not defined otherwise, the default port **1521** is going to be used.
- **Use Remote-Execution based Oracle discovery.** - By default, the discovery of the instance will be done in the "zero-touch" mode. Using this option enables remote execution of scanning utilities on a target server, which usually provides better scanning results. For more information about the differences between inventory methods, refer to the Inventory Methods chapter.

Inventory

This page contains a configuration of an optional step, allowing users to perform inventories of newly found services.



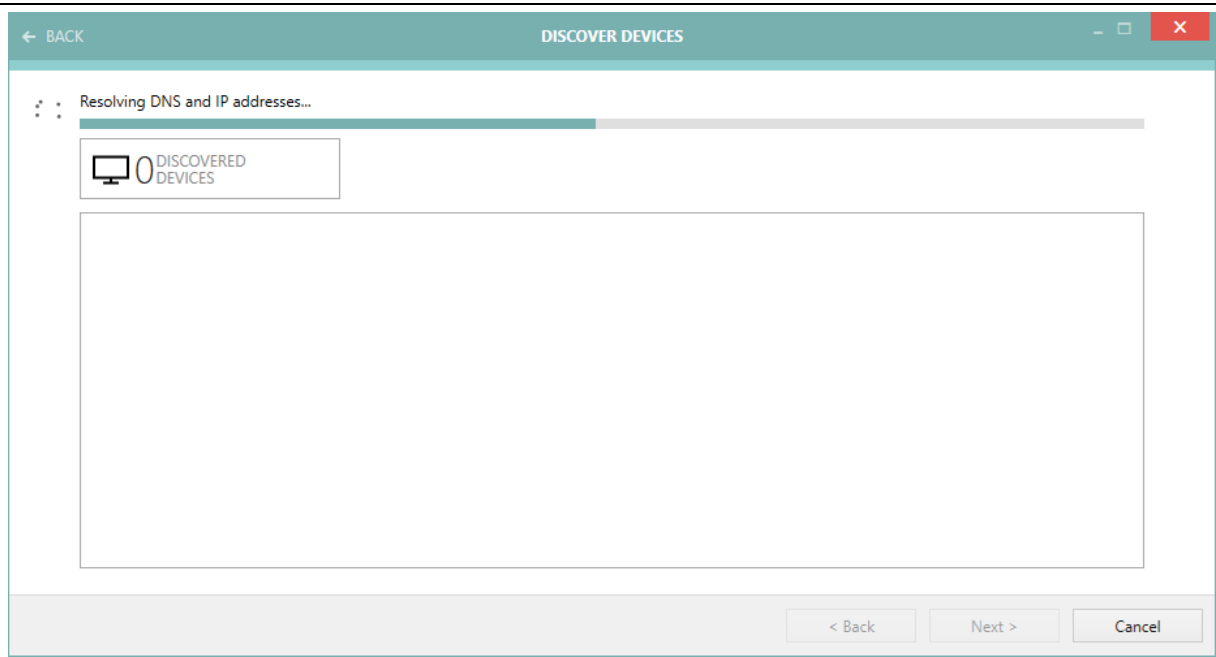
In order to scan software and hardware of new devices, tick the radio button **Inventory the following assets** and then select the options to scan. The options are complementary, if the radio button is checked then at least one option must be enabled in order to continue. By default, no options are selected.

The number of options may vary, depending on which services were selected on the **Services** page.

For more controls on how the inventory is executed on the target devices, we recommend to skip the inventory at the discovery page and instead let it run and discover the devices. Then, adjust the devices and / or settings and start the Inventory Wizard for fine-grained control on all inventory aspects.

Progress and Results

The progress page shows the current activity and real-time results of the scanning.



Once the work is done, a confirmation page is shown. Once the wizard is closed, all discovered services are automatically imported into the **Oracle**, **SNMP**, or vSphere / ESX views.

Inventory Wizard

The **Inventory** wizard is a primary tool of scanning for software and hardware assets on existing devices and services.

There are different inventory methods which are offered by RayVentry Scan Engine and which will generate data that can be imported by the RaVentry Server for further processing and reporting.

Starting the Wizard

You can start the **Inventory** wizard from the start / home screen by pressing the tile **Inventory devices...** or from the context menu or the right side-bar of the **Devices + Services** screen. You may also run an inventory as part of the discovery on newly found devices / hosts and services.

Introduction to Inventory Methods

RayVentry Scan Engine supports Zero-Touch inventory for all inventory target types. Zero-Touch is a general term for number of techniques which use exposed APIs and management functions without executing own code on the target system. Using Zero-Touch inventory usually involves faster scans, simplistic configuration, and zero impact on the target device (no leftovers, no code execution). As a trade-off, some functions must be enabled and some permissions must be set on the target device in order for the remote zero-scanning being possible at all.

Alternatively, RayVentry Scan Engine offers remote-execution based inventory methods for device / OS and Oracle inventory. Remote-Execution denotes that custom scan utilities must be executed on the scanned environment. This already implies a certain impact on the target device, even though all temporary files are gone once the process is done. Remote scans are more complex to configure, but may deliver more precise scanning results, as by executing a code locally there is usually a better access to lower-level features and APIs.

What inventory methods may be applied to the targets, can be restricted per device or for all devices in general. Restriction on a per-device base is done implicitly by disabling what technologies / capabilities of the target devices may be used. For example: Disabling the capability **Windows Service Manager** for a device will disable all remote-execution based inventory methods that use the Windows service manager. For completely excluding a device / service from inventory, disable all capabilities.



Hint:

For the current set of OS / device inventory and Oracle inventory methods, it is sufficient to disable the capabilities 'Zero-touch' and 'Remote-execution' to disable all respective inventory methods.

The restriction of inventory methods for all devices / services is done explicitly, by disabling methods in the **Settings > Inventory > Inventory Methods**.

If an inventory is started and multiple inventory methods are available for a target, RayVentory Scan Engine will try each inventory method until it succeeds or runs out of available inventory methods.

Currently, RayVentory Scan Engine offers only one inventory method (zero-touch) for **Oracle Audit, vSphere, and SNMP**.

The inventory / discovery method **Oracle Auto Discovery and Inventory** is a remote-execution based discovery and inventory method. This method tries to find all available Java installations and determines their versions in order to pick the right version of ORATRACK for the combinations of Oracle DB versions and Java that are to be expected on a target device. This method is not restricted by capabilities and cannot be disabled in the **Settings** screen as it is not part of the inventory methods that will be automatically applied by RayVentory Scan Engine. This inventory / discovery method can only explicitly be triggered by the context menu in the **Devices** list or during discovery by enabling the option **Use Remote-Execution based Oracle discovery**.

Usual Precedence of Inventory Methods

The inventory methods for device / OS inventory and Oracle inventory are tried in a specific order. They are usually sorted by their impact on the target device (for example Zero-Touch Inventory considered to have the least impact on the target devices are always tried first, unless they are disabled or incompatible).

Operating System Inventory Methods for Windows Devices

- Zero-Touch by WMI / WINAPI on Windows
- Remote Execution by ServiceManager upload HTTP(S) on Windows
- Remote Execution by WMI upload HTTP(S) on Windows
- Remote Execution by ServiceManager upload SMB on Windows
- Remote Execution by WMI upload SMB on Windows
- Remote Execution by WMI / SMB local files on Windows
- Remote Execution by ServiceManager / SMB local files on Windows

Operating System Inventory Methods for UNIX Devices

- Zero-Touch by SSH on Linux / Unix
- Remote Execution by SSH / SCP local files on Linux / Unix

Oracle Inventory Methods

- Zero-Touch by ORATRACK
- Remote Execution by ServiceManager upload HTTP(S) on Windows
- Remote Execution by ServiceManager upload HTTP(S) on Windows
- Remote Execution by ServiceManager upload SMB on Windows
- Remote Execution by WMI upload SMB on Windows
- Remote Execution by ServiceManager / SMB local files on Windows

- Remote Execution by WMI / SMB local files on Windows
- Remote Execution by SSH / SCP local files on Linux/Unix

Special Precedence of Inventory Methods

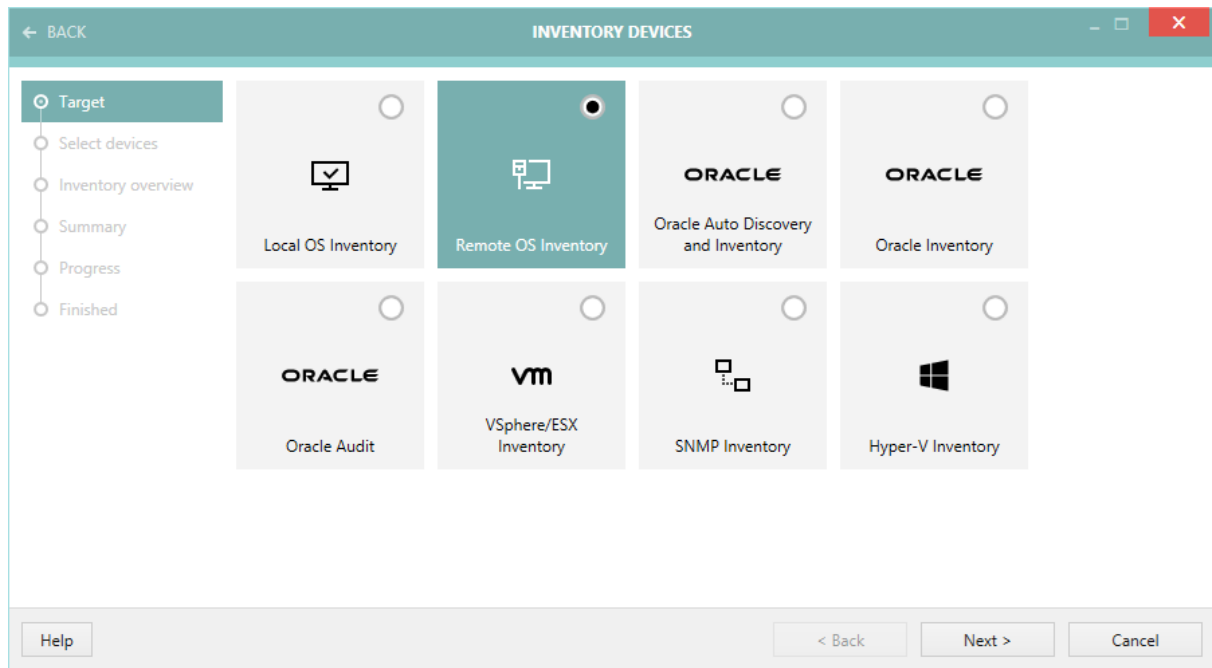
RayVentory Scan Engine remembers which inventory method worked for a each scanned device. If the previous scan was successful and RayVentory Scan Engine is able to determine which method worked for a device, that method is going to be preferred in the future and overrides the precedence outlined in the above lists. If the last successful method fails, then all other methods will be tried in the usual precedence. You can see the technical name for that method in the column **Last successful inventory method**. The column is hidden by default and must be selected from the column chooser. Similarly, technical names of failing methods are available in column **Last failed inventory methods**. The columns can be shown in the respective views in the **Devices + Services** screen.

Optimization on Consecutive Runs

Certain inventory methods for device / OS inventory and Oracle inventory are platform-specific. If an inventory succeeds for a target of undetermined type, then the implied target type is set for its respective device. The next time an inventory is run, certain checks that are needed to determine which of the available inventory methods are considered (like which ports are open) will be skipped.

Target

The first wizard page asks the user to select the type of inventory he would like to perform.



Based on the selected method, subsequent pages will receive slightly different context. For example, once **Remote OS Inventory** is selected, the next page shows the list of devices to be scanned. On the other hand, selecting **SNMP Inventory** shows the list of SNMP connections on the next page, while **Local OS Inventory** completely hides the second page.

The following options are available:

Local OS Inventory

This runs a local inventory which will give an `.ndi` file as result. This inventory file includes data regarding the hardware and software of the host. The extend of the inventory can be controlled by adjusting the file `wmitrack.ini` (query by WMI) or by adding certain registry entries (including the results of special Visual Basic scripts).

This is the only method that can be started without any devices or services configured in the **Devices + Services** screen.



Be aware:

Running the local inventory is equivalent of starting `ndtrack.exe` manually with the following parameters `-t machine -o ShowIcon=false -o NetworkSense=false -o Upload=false -o InventoryDirectory="XXX"` where XXX is the path to the output folder.

This option is kept in RayVentory Scan Engine for compatibility reasons. The results of the scan cannot be seen in the **Devices** view. If you want to perform a scan on the local system and have the results managed by the Devices screen trigger an inventory on a target device with DNS name localhost or IP address of the current host.

Remote OS Inventory

This creates an inventory of a remote machine using either pure Zero-Touch technique or Remote-Execution on the target device. RayVentory Scan Engine automatically chooses the right method for each device, based on various settings, capabilities and previous runs. After selecting remote devices, you will be able to verify which methods get executed on which device by visiting the **Inventory Overview** page.

Local and remote OS inventories support the following platforms:

- Windows
- Linux*
- HP-UX
- IBM AIX
- OSX
- Solaris

* - conditions apply, support for different Linux distribution may vary.

**Be aware:**

There may be certain versions or architectures of platforms that are not supported.

Oracle Inventory

The actual inventory runs certain queries on an Oracle database in order to retrieve data and hints on license relevant configuration details. In conjunction with the OS inventory of the database host, this data is the source for the reports that help to figure out what kind of Oracle license is needed. If the path to the Oracle's DBFUS script is set, this inventory will also include results from this script. If such an inventory is being imported by a RayVentory Server, this will allow for the comparison on what database features were found to be license relevant by the Oracle Inventory and by the DBFUS script. See Support for Database Feature Usage Script for the requirements that are necessary to enable the DBFUS execution option.

**Be aware:**

In order to use this option a Java Runtime Environment (JRE) compatible with at least Java SE 6 must be installed.

Oracle Auto Discovery and Inventory

Discovers database instances installed on a host and gathers data on enabled options and usage on Oracle database instances.

Oracle Audit

With RayVentory Scan Engine, it is possible to run the **Oracle's Review Lite Script** on many target databases at once and collect the output files. To use this option, in the setting **Review Lite Script** path, the path to the copy of the **Review Lite Script** needs to be set. Furthermore, the path to the SQLplus executable in the local Oracle client installation needs to be set.

vSphere / ESX Inventory

This gathers data regarding host / guest relationships and the host and guest configuration for VMware ESX or vSphere virtual infrastructures. This inventory method requires VMware type credential in the credential store. Addressing the SDK service endpoint of an ESX will retrieve data on all hosts and guests in the cluster.

SNMP Inventory

Gathers basic data on network devices that support the SNMP protocol. This is intended for devices like printers, UPSs, and network equipment that does not expose its Operating Systems or a standardized interface besides SNMP.

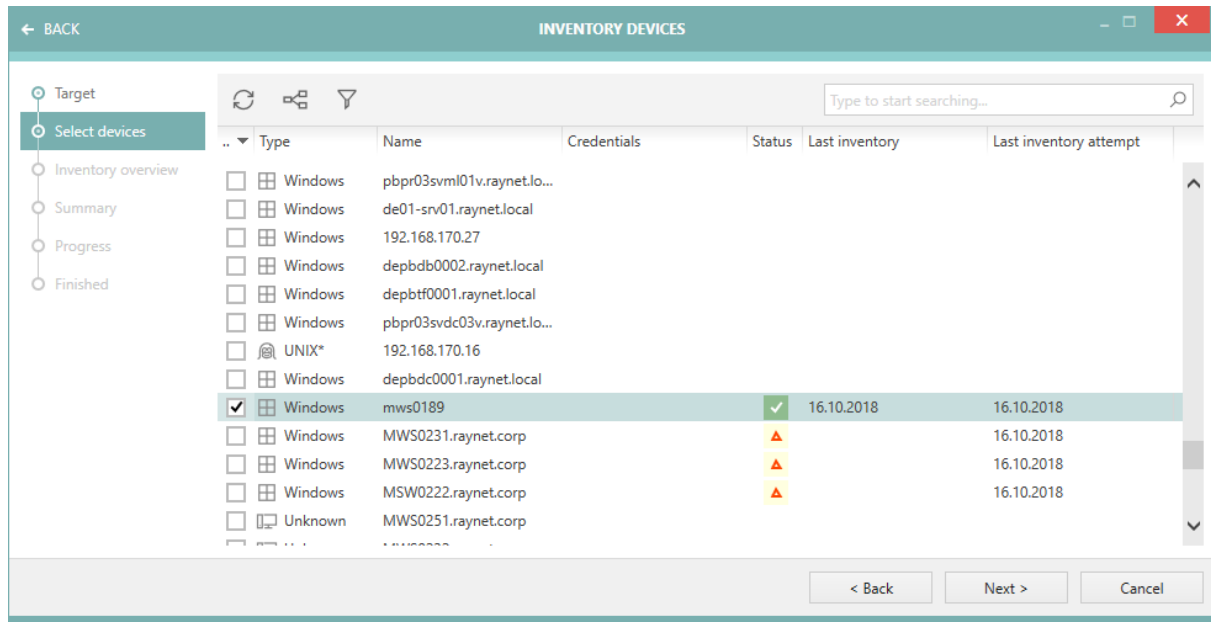
Performing Oracle Audit and Discovery

There are two methods are not available in the Wizard, but can be called from other places:

- The inventory method **Oracle Audit** is not available from the inventory wizard or as an option in the discovery wizard but it is accessible in the context menu of the **Oracle** list as **Audit....**
- The discovery / inventory method **Oracle Discovery** is not available in the inventory wizard but as an option in the **Discovery** wizard and in the context menu of the **Devices** list as **Oracle Discovery....**

Select Devices

This page allows the user to select which devices are to be scanned.



The entries and layout seen here may vary based on the inventory type selected on the first page. You can use similar filtering, searching, and grouping capabilities as in the **Devices** view. The same chapter describes how to interpret the values visible in the **Status** column.

Unlike the **Devices** view, the grid here does not support multiselection by highlighting items. Instead, a separate checkbox column is used. You can select all, deselect all, and invert the currently visible selection by accessing the grid context menu and choosing one of three options.

In order to go to the next page, at least one item should be selected.



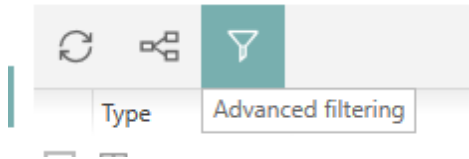
Note:

RayVentory Scan Engine provides a usability shortcut - if no device is selected in this view and the user presses the **Next>** button, all devices will be included automatically.

Filtering

The list of available devices can be filtered using advanced capabilities of the **Advanced filtering**.

To filter the devices, press the **Advanced filtering** button:



The **Advanced Filtering** dialog consists of the following parts:

- **Expression editor**

A multi-line text field supporting basic syntax highlighting. The content of this field can be edited manually, or changed by interacting with four functional buttons: Properties, Features, Operators and Snippets.

- **Properties dropdown**

This is the list of properties belonging to devices. You can insert the property by selecting it and choosing the comparison/equality operator, followed by pressing the **Add to condition** button. Properties may be compared against values entered manually, or generated by the **Features** button.

- **Features**

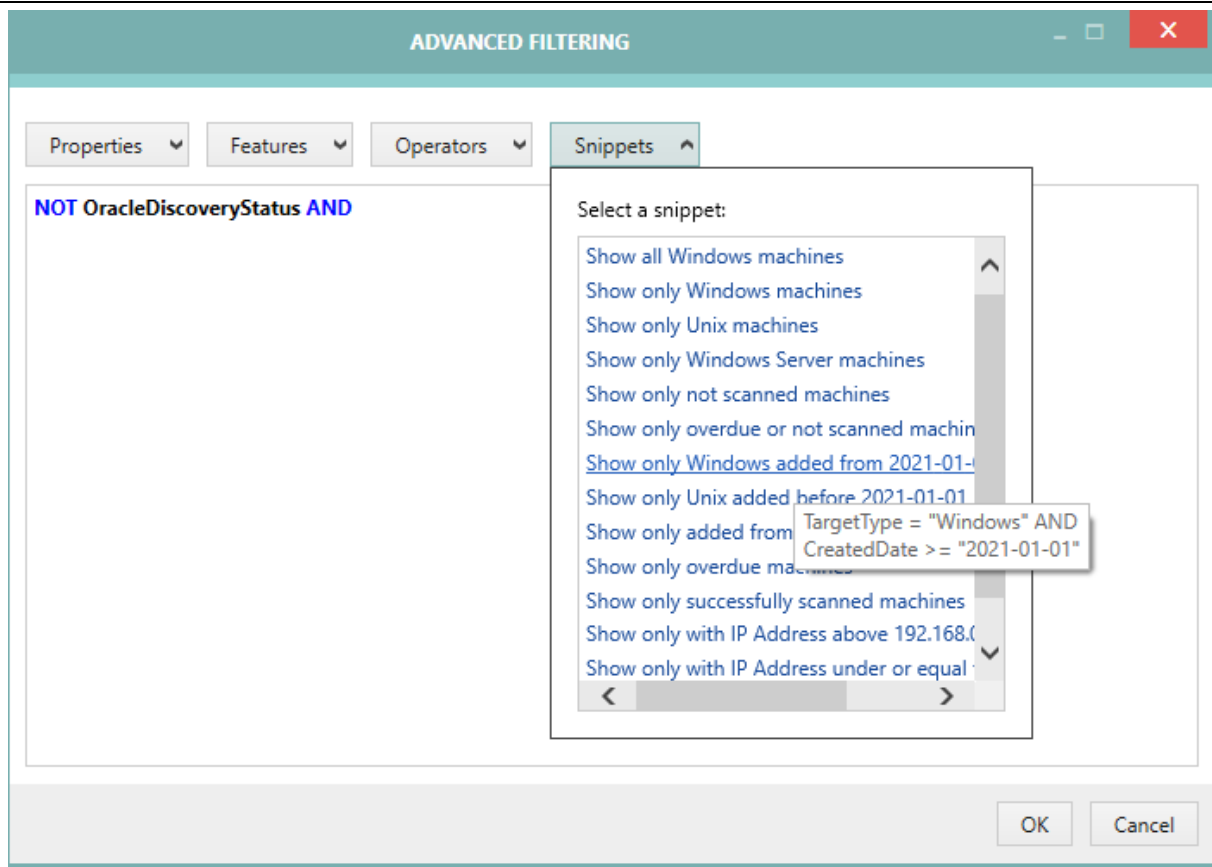
This is a generator of values for various properties. Using this editor ensures, that dates, boolean values, names etc. are used properly, with a right context and in the right format.

- **Operators**

Various boolean operators for joining conditions.

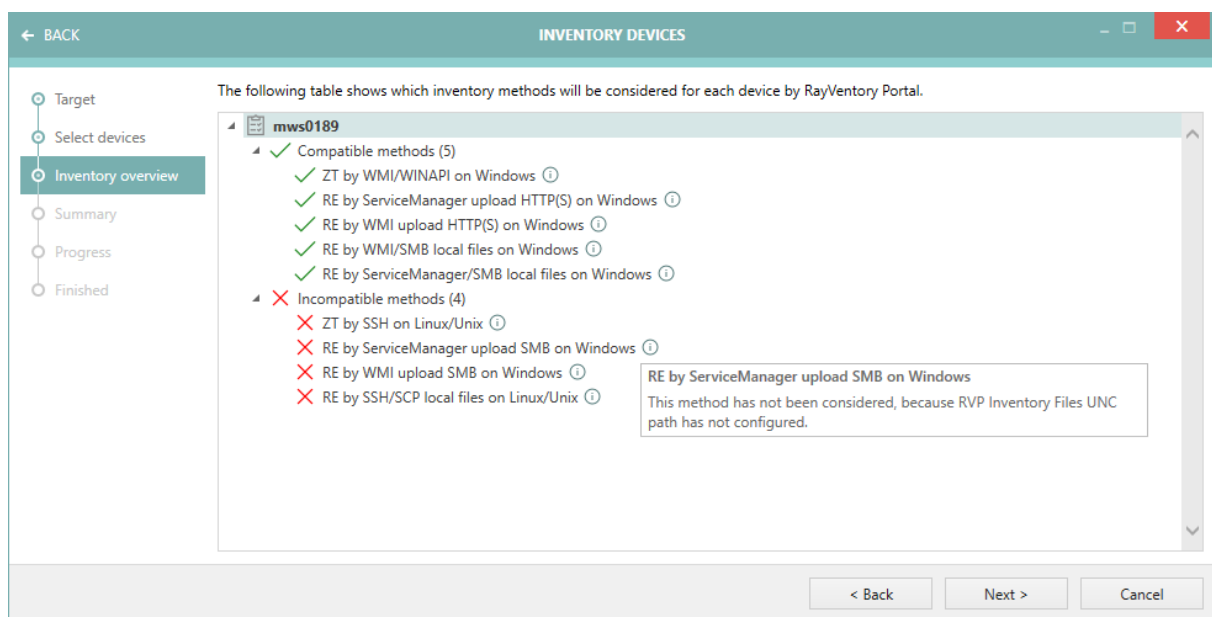
- **Snippets**

This is the list of predefined pieces of conditions, which provides some common conditions and serves as a basis and get-started help for customizing conditions.



Inventory Overview

This page contains a detailed technical summary of compatible and incompatible methods for each selected device.



The main purpose of this view is to identify issues before the scan is even started. The administrator can utilize this information to find out devices which:

- ... have no compatible methods and thus will not be scanned at all,
- ... have certain required methods disabled,
- ... are otherwise misconfigured or use unwanted scanning methods.

For each method, RayVentry Scan Engine provides a tooltip help which explains why a given method was considered as **Incompatible**. The tooltip has also secondary function - for methods that are considered **OK** for a given device, the tooltip information explains why the method has been chosen (for example: the target OS is Windows and its capabilities are sufficient to handle the task).

To determine which methods are compatible 3 sources are primarily concerned:

- Current RayVentry Scan Engine settings, for example globally excluded inventory methods, HTTP server, remote execution settings, etc.
- Device properties (for example type) and capabilities,
- Additional information gathered from port scanning (only if device type is **Unknown**).

You can ignore recommendations from this wizard page and continue to the next page, even if loading the data has not been finish yet. Keep in mind, that for every device of the type **Unknown** a separate port scan will be executed to determine which methods (Windows- or Unix-based) will be used. In order to avoid this extra ping sweep on your network, make sure to always set the device type to either Windows or Unix in the device properties.

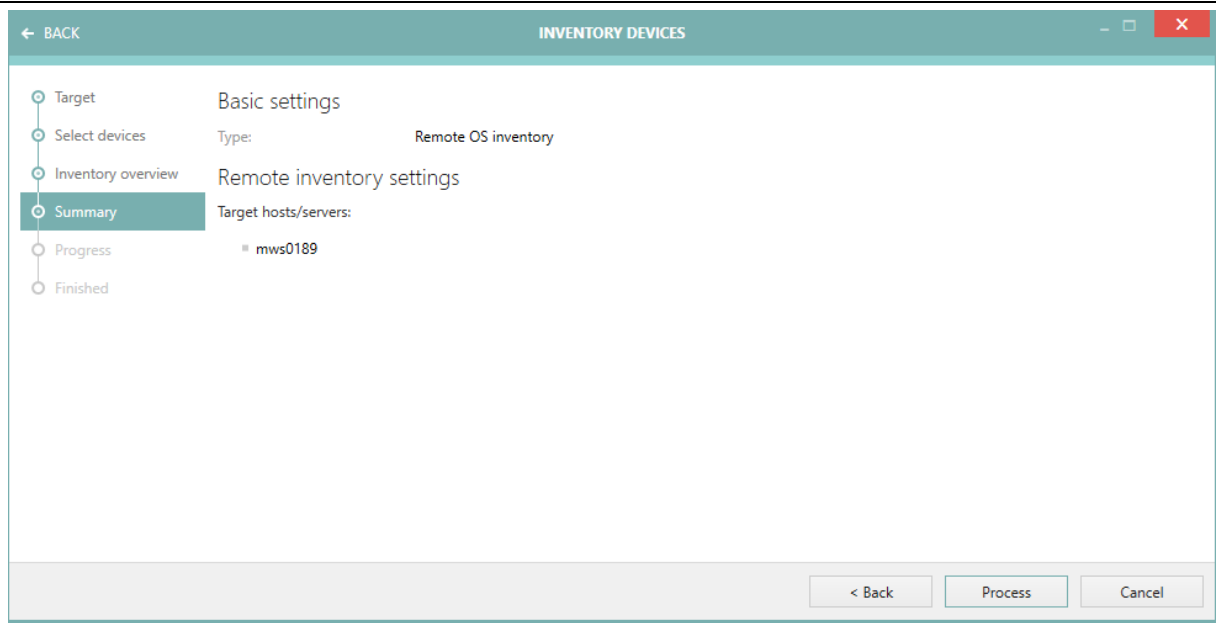


Be aware:

This view provides read-only information. It is not possible to change or configure RayVentry Scan Engine or target devices from here.

Summary

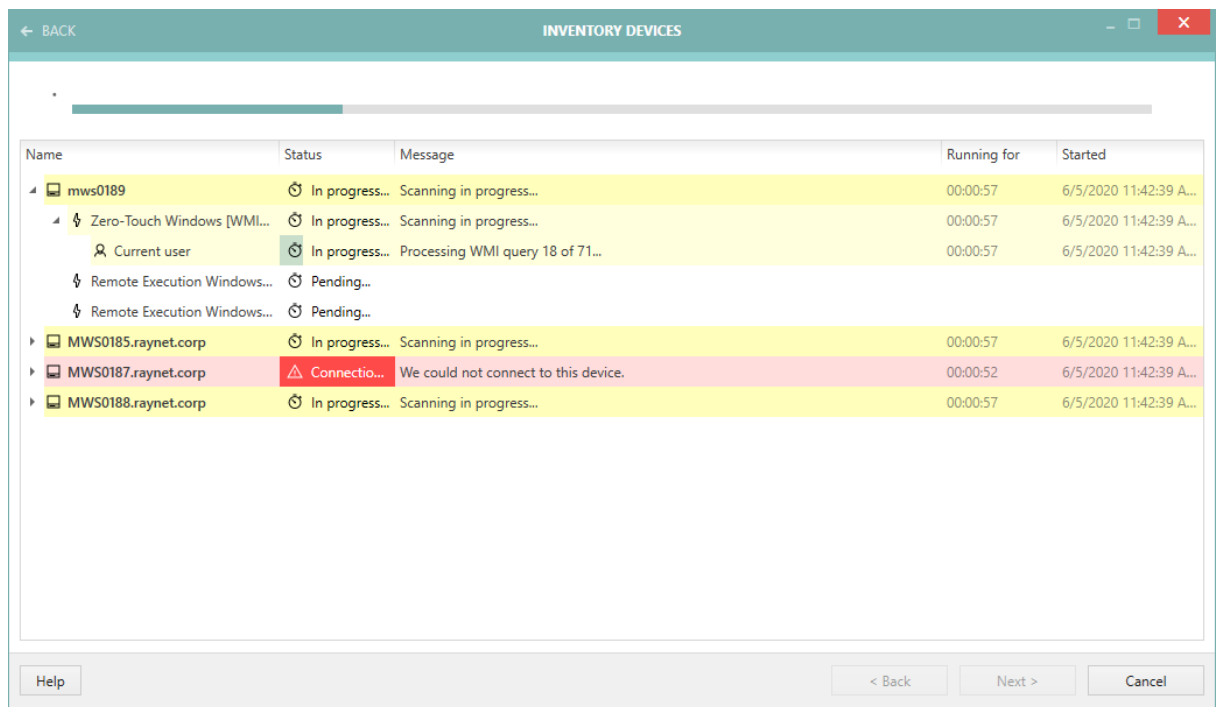
The **Summary** page shows the overview of all choices defined in the previous pages. You can click on the settings to go back to their corresponding places and change them.



Press **Process** to start the inventory scan.

Progress and Results

The progress page shows the current activity and real-time results of the scan.



The progress is live, which means the status, time and progress/error feedback is immediately visible.

- Yellow cell - the item is currently being processed.
- Green cell - the item has been successfully processed and returned the inventory results back.
- Grey cell - the item has been skipped or canceled.
- Red cell - the item has failed.

You can cancel the inventory at any time by pressing the **Cancel** button. Note that some operations may still need a few seconds to cancel properly and release the resources, close connections etc.



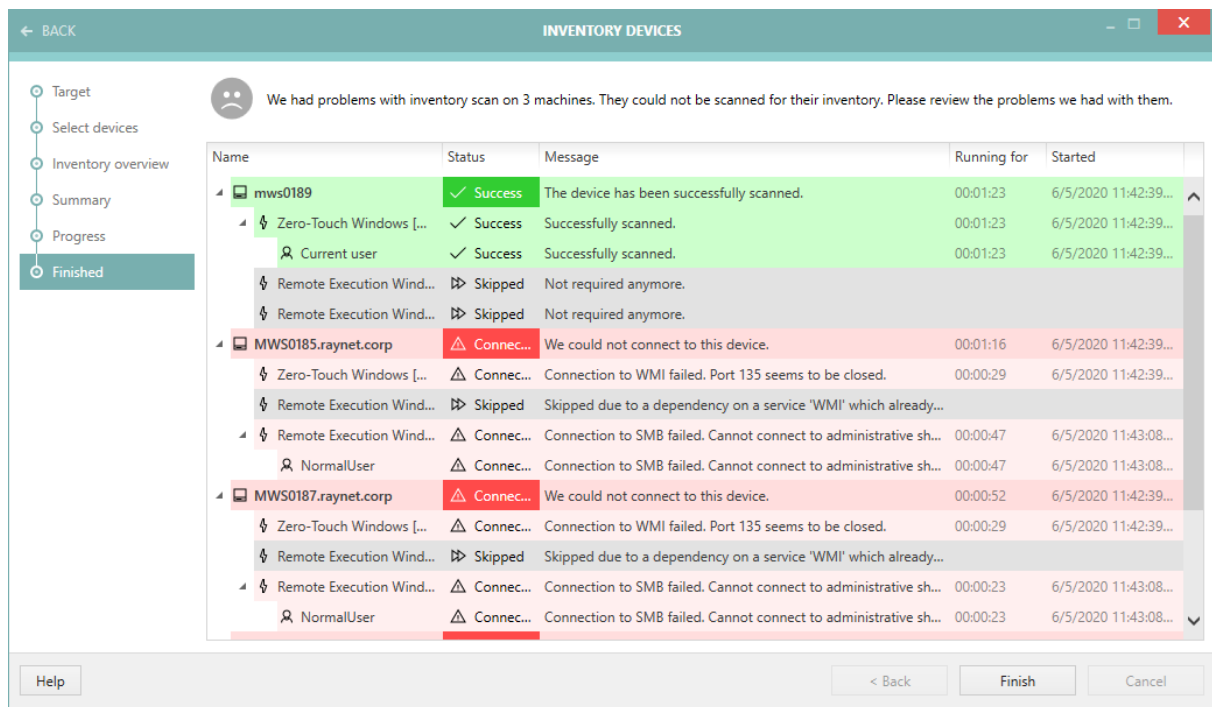
Note:

Chapter Recent Scan Details contains various tips how to interpret the results displayed in the inventory view.

By default, the treeview of device data is partially collapsed, which means only root entries (the devices itself) are displayed. You can expand the items using the little arrow icons to reveal more details about each of the item.

Once the scan is over, press **Finish** to close the wizard. If any errors are encountered during the scanning, they will be listed here.

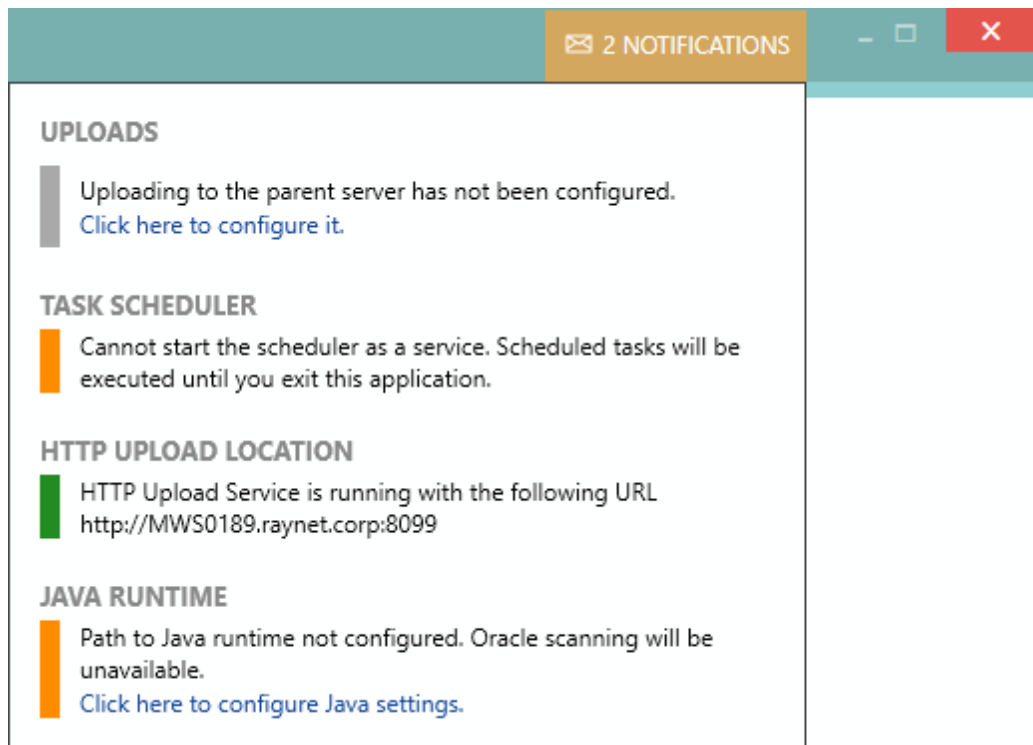
The finished page contains a partially expanded view of the results. This way it is possible to review the results gathered during the finished inventory scan.



| Name | Status | Message | Running for | Started |
|--------------------------|-----------|--|-------------|----------------------|
| mws0189 | Success | The device has been successfully scanned. | 00:01:23 | 6/5/2020 11:42:39... |
| Zero-Touch Windows [...] | Success | Successfully scanned. | 00:01:23 | 6/5/2020 11:42:39... |
| Current user | Success | Successfully scanned. | 00:01:23 | 6/5/2020 11:42:39... |
| Remote Execution Wind... | Skipped | Not required anymore. | | |
| Remote Execution Wind... | Skipped | Not required anymore. | | |
| MWS0185.raynet.corp | Connec... | We could not connect to this device. | 00:01:16 | 6/5/2020 11:42:39... |
| Zero-Touch Windows [...] | Connec... | Connection to WMI failed. Port 135 seems to be closed. | 00:00:29 | 6/5/2020 11:42:39... |
| Remote Execution Wind... | Skipped | Skipped due to a dependency on a service 'WMI' which already... | | |
| Remote Execution Wind... | Connec... | Connection to SMB failed. Cannot connect to administrative sh... | 00:00:47 | 6/5/2020 11:43:08... |
| NormalUser | Connec... | Connection to SMB failed. Cannot connect to administrative sh... | 00:00:47 | 6/5/2020 11:43:08... |
| MWS0187.raynet.corp | Connec... | We could not connect to this device. | 00:00:52 | 6/5/2020 11:42:39... |
| Zero-Touch Windows [...] | Connec... | Connection to WMI failed. Port 135 seems to be closed. | 00:00:29 | 6/5/2020 11:42:39... |
| Remote Execution Wind... | Skipped | Skipped due to a dependency on a service 'WMI' which already... | | |
| Remote Execution Wind... | Connec... | Connection to SMB failed. Cannot connect to administrative sh... | 00:00:23 | 6/5/2020 11:43:08... |
| NormalUser | Connec... | Connection to SMB failed. Cannot connect to administrative sh... | 00:00:23 | 6/5/2020 11:43:08... |

Notification Center

The Notification Center is a central place where important messages and status are shown.



You can access it at anytime by clicking on its icon residing in the upper right corner of the window. Depending on the status, the notification center can contain the counter of important messages (for example **2 NOTIFICATIONS**) or just simply a text **NOTIFICATION CENTER** if there are no pending messages. Additionally, the color of the badge determines the importance of the messages:

- Green (transparent) - no urgent messages.
- Yellow - important messages (for example missing non-critical configuration, pending uploads etc.).
- Red - critical messages (errors and critical problems with product configuration).

The notification center is the place where file uploads are triggered. If your upload path is configured in **Settings > HTTP Services > Upload Location**, then the button to upload any pending files is displayed in the **UPLOADS** section.

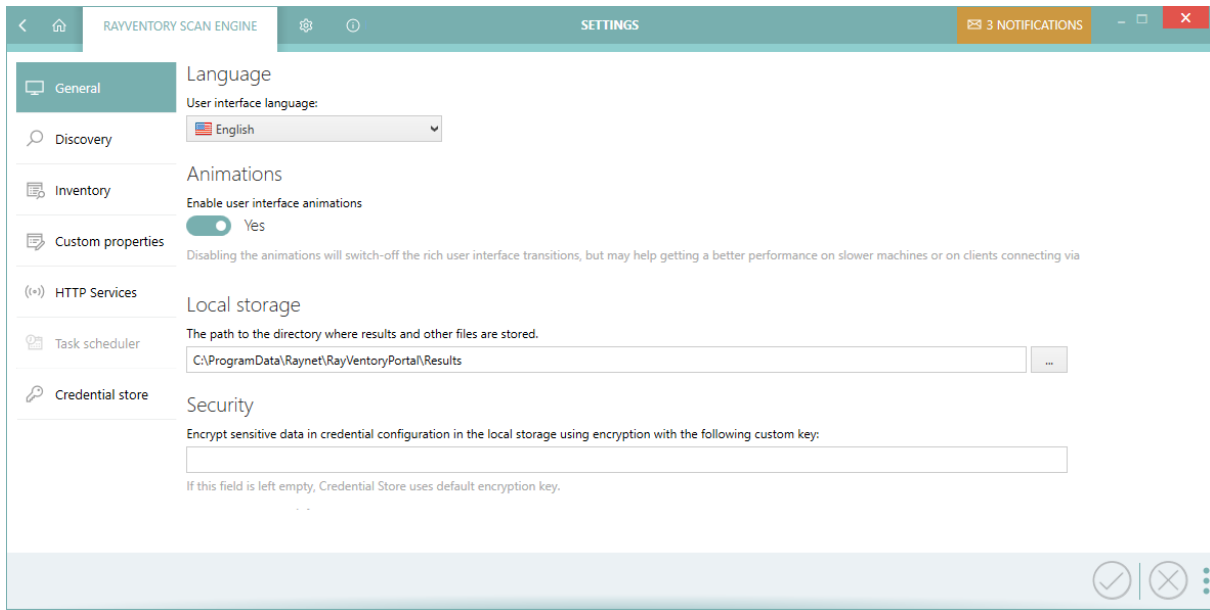
The notification center also informs about:

- The status of the task scheduler (whether it is running as a Windows Service or as private process).

-
- The status of the HTTP Upload location exposed to endpoints (and its full address).
 - The status of the Java runtime (required for Oracle scans).

Settings

In the **Settings** screen further settings for the customization of the inventory methods and the customization of RayVentory Scan Engine can be found.



The **Settings** screen is split into the following categories:

- **General**
- **Discovery**
- **Inventory**
- **HTTP Services**
- **Task scheduler**
- **Credential store**

General

In the **GENERAL** section there are two settings which apply to the user interface of all the features.

User interface language

This is used to change the language of RayVentory Scan Engine UI. In the current version, three languages are available (English, German and Russian).

Animations

This is used to enable or disable the user interface animations like the transition between the different screens.

Local Storage

This setting defines where RayVentory Scan Engine saves the results of software and hardware inventories. The default value is `C:\ProgramData\RaynetVentoryPortal\Results`.



Note:

It is not recommended to change the local storage after any inventory wizards are available.

Security

This setting defines the encryption key used to encrypt sensitive data (like passwords and keys) inside the configuration files. If this value is left empty, a default encryption key (which is the same for all RayVentory Scan Engine users) will be used.



Note:

It is not recommended to change the local storage after any inventory wizards are available. Once the encryption key is changed, all credentials and keys must be corrected again, because no automatic decryption and encryption will take place once the key is changed.



WARNING

The encryption key is saved in plain text in the RayVentory configuration file. Therefore, it is necessary to ensure that the permissions of the file system suit the respective security needs.

Discovery

Enter topic text here.

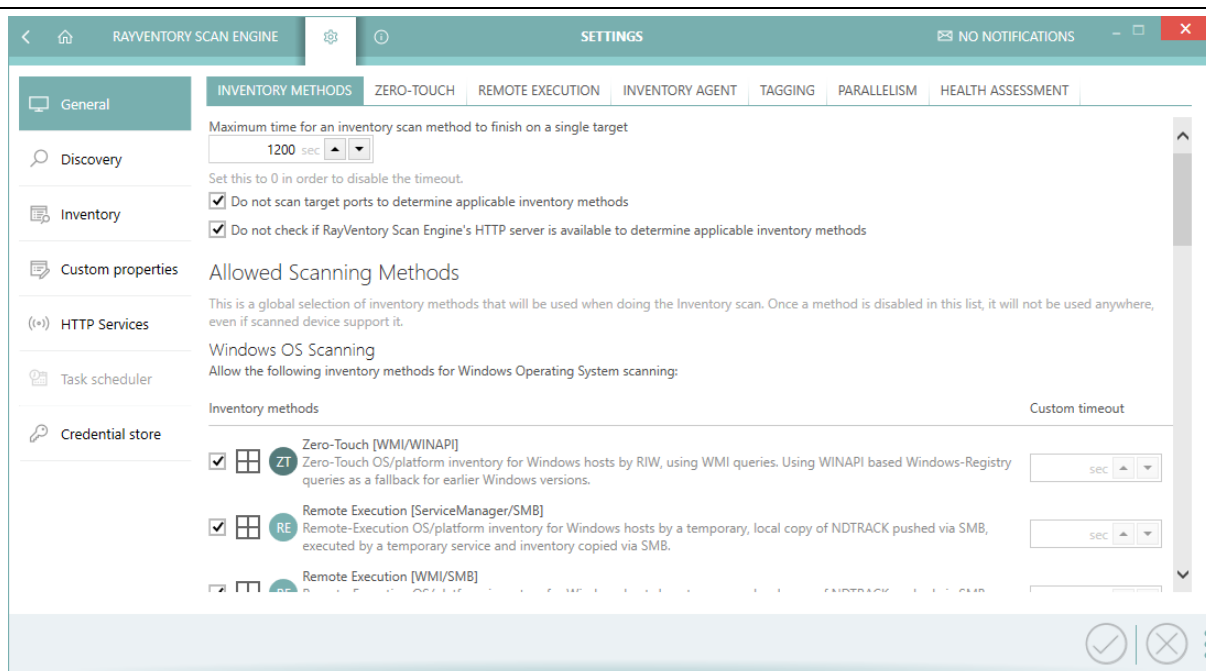
Inventory

This tab contains three sub-tabs that control which inventory methods are active and defines the settings that are used by the Zero-Touch and the Remote-Execution inventory scans.

This is also the place where the configuration of Inventory Agent can be managed and created.

Inventory Methods

This tab is used to configure which inventory methods are globally enabled or disabled.



RayVentory Scan Engine supports up to 20 different methods which differ in:

- ... where the work is being done (zero-touch from remote machine, or remote execution on the actual target machine).
- ... the type of the Operating System (different set of techniques for Windows and Unix).
- ... the way results are received back (via HTTP, file share, file access etc.).
- ... the way in which remote execution is triggered (portable executable from share, a service, file access etc.).

The description next to each icon contains information about the target device family (in form of icon and textual description, either Windows or Unix), its name, and main characteristics of how it is actually working and where potential security pitfalls may lie. Within every group, the methods are sorted from what is agreed to be the "safest" method to some more invasive methods.

As a general rule:

- The methods described as "Zero-Touch" have minimal or zero impact on the target system. They usually connect to the target device via publicly available APIs or services and extract the data they need. No RayVentory Scan Engine process are executed on the host system. Methods from this family are less "intrusive" than remote execution, but may sporadically be limited by functionality of underlying APIs, services, and exposed endpoints. As a rule of thumb, you should try to start with zero-touch methods and verify whether the incoming results are satisfactory and move to the remote-execution in the next step, should some devices or networks require them.
- Methods described as "Remote-Execution" may potentially have impact on the target system to some varying degree. Depending on the method type, some of them may try to install a service and almost all of them execute scan utilities bundled with RayVentory Scan Engine.

After each execution a necessary clean-up is done (for example removing the temporary service and / or files), but the machine may be impacted in one or more ways just because of some processes being physically started on it. While these methods are considered more "intrusive", they tend to deliver better results as they have better access to the necessary resources and databases.

**Note:**

You can read more about differences between the different methods in the following chapter: Inventory Methods Overview. You may review them with your system network administrator to find out which methods should be allowed or disallowed in your environment.

Enabling or disabling any method is done by a mouse click:

- Tick a checkbox next to the name of the method to enable it.
- Untick it to disable it.

The icon and badge are dimmed if the given method is disabled.

Once the method is disabled here, it will no longer be considered by any inventory scan, even if the target device supports it. For example, if you disable remote execution on Windows and Unix from this screen, only zero-touch methods will be considered for your devices.

Time-out per method

It is possible to define a time-out, which - regardless of global settings - will be then used for a particular family of methods. To define it, enter the required value into the **Custom timeout** column. Leave the column empty to not use any method-specific timeout.

Zero-Touch

This tab contains four sub-tabs that control how Zero-Touch inventory is being executed on the following platforms:

- Windows
- Linux + UNIX
- vSphere + ESX
- Oracle

Windows

This tab controls the Zero-Touch scan settings for Windows machines.

Scan Settings

- **Use verbose logging for remote commands**

When this options is active, extra information about executed WMI queries are logged to the program log and / or console. You can use this option for the troubleshooting of custom Zero-Touch Windows scans.

- **Maximum time for the inventory scan to finish on a single computer**

The value (in seconds) which denotes the timeout for a single operation. For a task to finish successfully, it must return to the caller within the specified time range. Setting this value to 0 means that RayVentory Scan Engine waits indefinitely for the results.

- **Maximum number of scanned computers by a single task**

Within the concept of RIW, this number denotes how many computers can be scanned in parallel by a single task.

- **Connection test time out**

This number (expressed in milliseconds) denotes the timeout for a connection attempt. If the machine does not answer within this time range, the computer is considered to be offline or unavailable.

- **Delay between connection tests**

This denotes the delay between repetitions of unsuccessful connection attempts. RayVentory Scan Engine performs up to 3 attempts before returning a failed result if none of the attempts succeeded.

Extra Logging

- **Log failed connection attempts to the following file**

Specifies the extra logging path for failed connection attempts. New entries are appended to the bottom of the file content.

**Note:**

This option is not equivalent to the various RayVentory Scan Engine log options.

- **Log successful connection attempts to the following file**

Specifies the extra logging path for successful connection attempts. New entries are appended to the bottom of the file content.

**Note:**

This option is not equivalent to the various RayVentory Scan Engine log options.

Custom Configuration

- **Custom inventory configuration file**

This extra configuration file defines the custom scanning options. Configuring custom scans is a complex task, which is described in the chapter Custom Windows Scans.

Options for Legacy Systems

- **When reading the registry, prefer WINAPI over WMI**

When active, this indicates that RayVentory Scan Engine should prefer the WINAPI strategy

over WMI for querying for registry entries. The option is designed for legacy systems where WMI is unavailable.

Linux + UNIX

This tab controls the Zero-Touch scan settings for Linux machines.

Scan Settings

- **Maximum number of scanned computers by a single task**
Within the concept of RIU, this number denotes how many computers can be scanned in parallel by a single task.
- **Maximum time for the inventory scan to finish on a single computer**
The value (in seconds) which denotes the timeout for a single operation. For a task to finish successfully, it must return to the caller within the specified time range. Setting this value to 0 means that RayVentory Scan Engine waits indefinitely for the results.

Extra Logging

- **Log failed connection attempts to the following file**
Specifies the extra logging path for failed connection attempts. New entries are appended to the bottom of the file content.



Note:

This option is not equivalent to the various RayVentory Scan Engine log options.

- **Log failed authentication attempts to devices to the following file**
Specifies the extra logging path for failed authentication attempts. New entries are appended to the bottom of the file content.



Note:

This option is not equivalent to the various RayVentory Scan Engine log options.

- **Log successful connection attempts to the following file**
Specifies the extra logging path for successful connection attempts. New entries are appended to the bottom of the file content.



Note:

This option is not equivalent to the various RayVentory Scan Engine log options.

Scan options

- **Inventory items**
A list of comma-separated items of items to collect.

- **Enable Filescan**

Activate this switch to enable file scanning on UNIX devices.

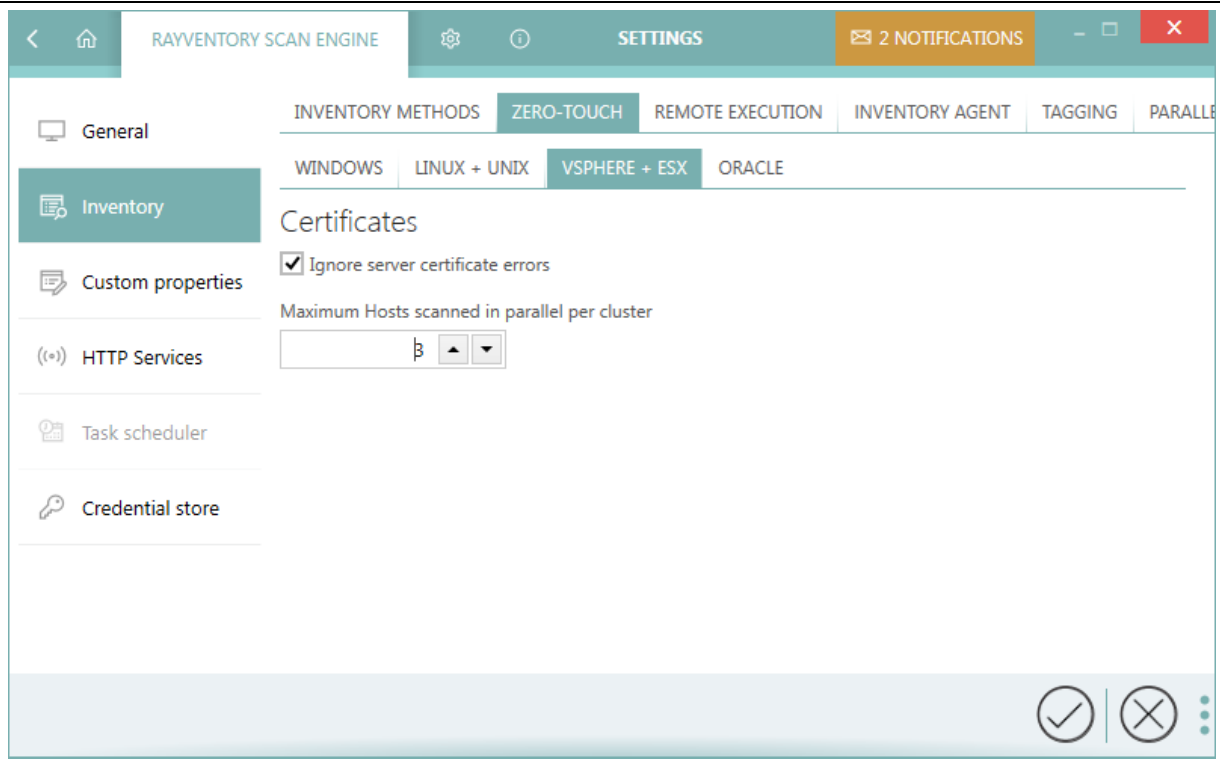
File Scan Options

This section controls file scan on UNIX systems. Each setting has a dedicated help text, which can be revealed upon hovering with mouse on the question mark icon. To change these settings, enable file scan first.

| WINDOWS | LINUX + UNIX | VSPHERE + ESX | ORACLE |
|---------|--|--|--------|
| | Embed File Content Max Size | 25000 | |
| | Exclude Directories | | |
| | Exclude Embed File Content Directories | | |
| | Exclude Extensions | | |
| | Exclude Files | Comma separated list of file extensions which should be excluded from the file scan. | |
| | Exclude MD5 | | |
| | Include Directories | / | |
| | Include Extensions | sys,sys2,jar,sh | |
| | Include Files | version.txt,versions.txt,notes.ini | |
| | Include MD5 | | |
| | Include Network Drives | <input type="checkbox"/> | |

vSphere + ESX

This tab controls the vSphere / ESX Zero-Touch scan settings.



Ignore Server Certificate Errors

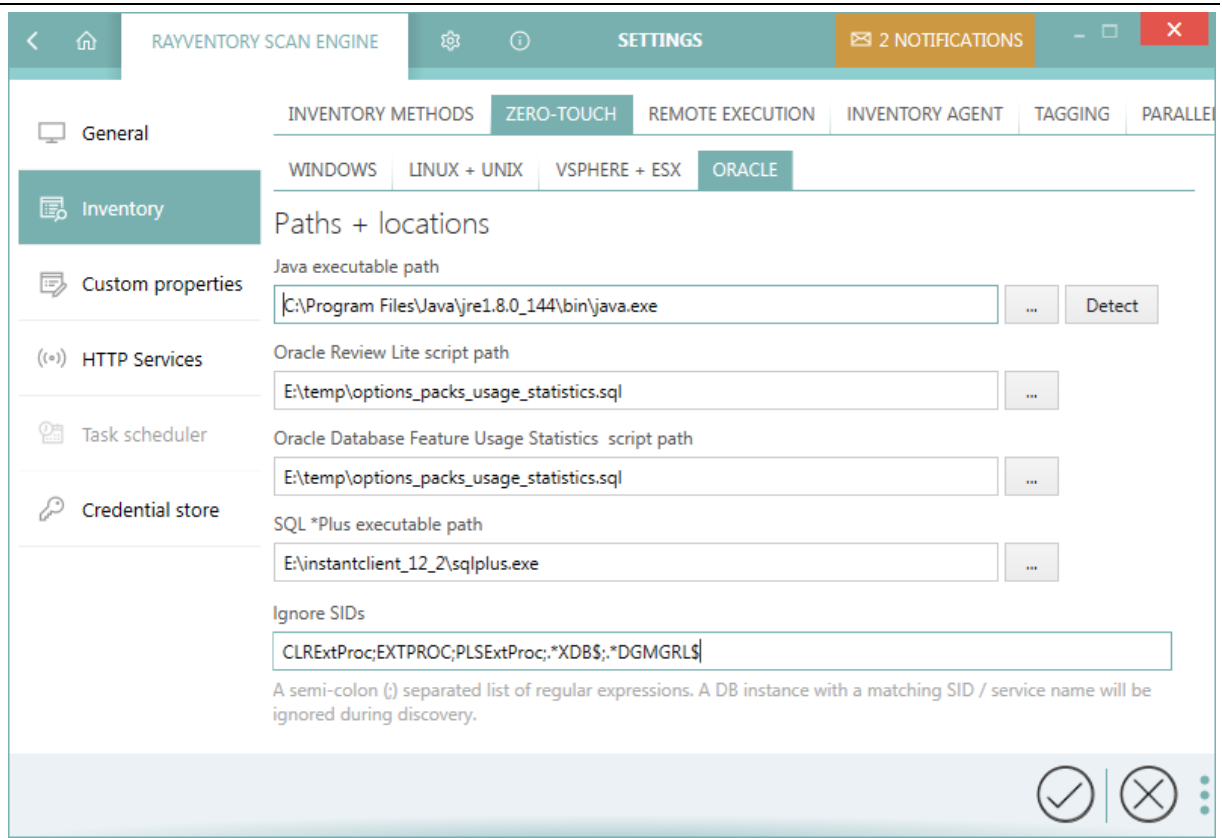
When this option is enabled, invalid, missing, or expired SSL certificates of the vSphere server will be ignored by the scanning agent. If this option is deactivated, then the certificate of the server (in case of HTTPS connection) will be checked and must be therefore: valid, trusted by trusted authority on the current machine, and not expired.

Maximum Hosts Scanned in Parallel per Cluster

This option enables the user to limit the maximum number of ESX hosts scanned in parallel per target (cluster).

Oracle

This tab is used to configure various Oracle-related paths.



RAYVENTORY SCAN ENGINE

SETTINGS

2 NOTIFICATIONS

INVENTORY METHODS ZERO-TOUCH REMOTE EXECUTION INVENTORY AGENT TAGGING PARALLEL

WINDOWS LINUX + UNIX VSPHERE + ESX ORACLE

General

Inventory

Custom properties

HTTP Services

Task scheduler

Credential store

Paths + locations

Java executable path

C:\Program Files\Java\jre1.8.0_144\bin\java.exe

Oracle Review Lite script path

E:\temp\options_packs_usage_statistics.sql

Oracle Database Feature Usage Statistics script path

E:\temp\options_packs_usage_statistics.sql

SQL *Plus executable path

E:\instantclient_12_2\sqlplus.exe

Ignore SIDs

CLRExtProc;EXTPROC;PLSExtProc;*XDB\$;*DGMGRL\$

A semi-colon (;) separated list of regular expressions. A DB instance with a matching SID / service name will be ignored during discovery.

Java Executable Path

The full path to the java runtime executable. You can type the path directly, use the ... button to pick a file from your host, or use the auto-detection by pressing the **Detect** option.

If the Java runtime path is not configured here, RayVentory Scan Engine tries to look for it anyway any time it needs it. By providing a custom value in this field, you can cover the following use cases:

- There are multiple instances of Java and you want to use a specific one for Oracle tasks or...
- You have a "private" instance of Java which is not registered in the system-wide location and you want to use that instance.



Note:

All Oracle-related methods require that Java is available to the RayVentory Scan Engine scan utilities, otherwise these method will fail.

Oracle Review Lite Script Path

The full path to the Oracle Review Lite script, which gets executed when performing an audit on Oracle databases. You can type the path directly or use the ... button to pick a file from your host.

Oracle Database Feature Usage Statistics Script Path

The full path to the Oracle Database Feature Usage Statistics script, which gets executed when performing an DBFUS on Oracle databases. You can type the path directly or use the ... button to

pick a file from your host.

SQL *Plus Executable Path

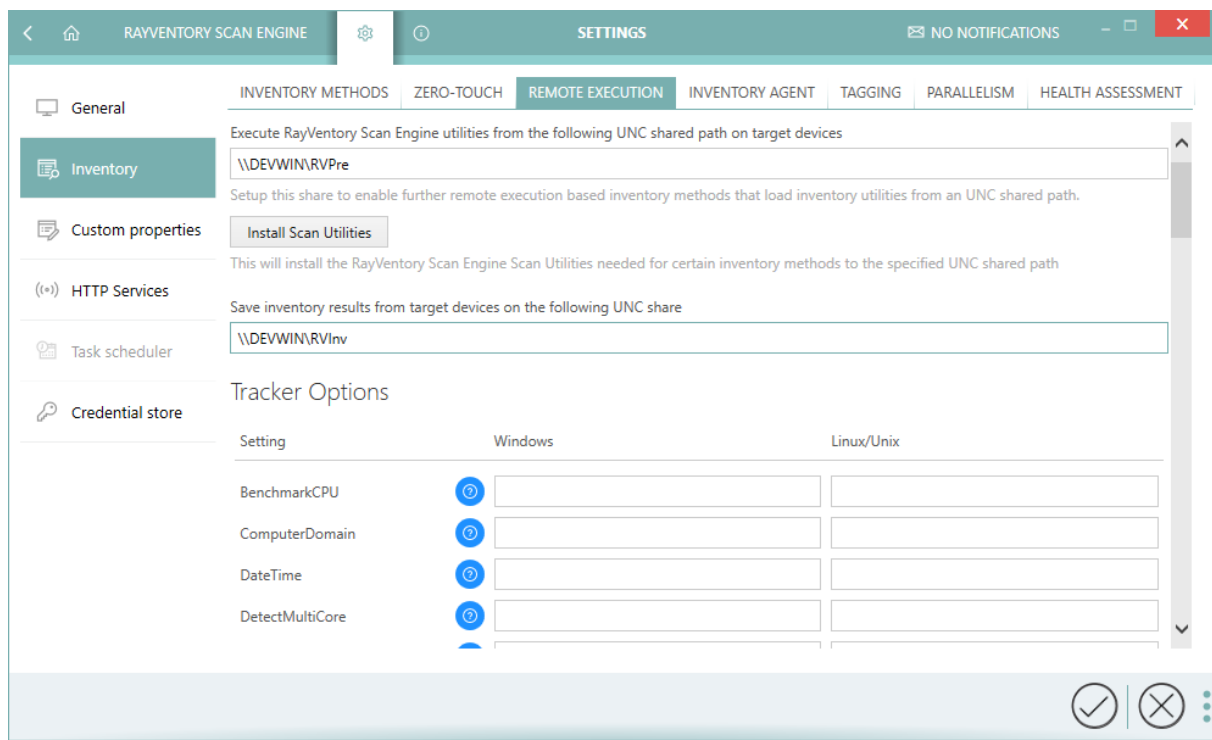
The full path to the SQL *Plus executable. This path is required to perform audit tasks (see more on that in the following chapter Oracle Audit).

Ignore SIDs





The list of semicolon separated regular expressions, used to determine which DB instances (based on their SIDs) will be ignored during OracleDB Inventory/Discovery operations. The defaults should be reasonable for most of use-cases, but you may finetune your results by adding some more excluded items here.

Remote Execution

This tab contains settings which are relevant for all remote-execution-based scans.



The screenshot shows the 'RAYVENTORY SCAN ENGINE' application window with the 'SETTINGS' tab selected. The 'REMOTE EXECUTION' sub-tab is active. The interface includes a left sidebar with options: General, Inventory, Custom properties, HTTP Services, Task scheduler, and Credential store. The main content area is divided into sections: 'Execute RayVentory Scan Engine utilities from the following UNC shared path on target devices' with a text field containing '\\DEVWIN\\RVPre'; 'Setup this share to enable further remote execution based inventory methods that load inventory utilities from an UNC shared path.' with an 'Install Scan Utilities' button; 'Save inventory results from target devices on the following UNC share' with a text field containing '\\DEVWIN\\RVInv'; and 'Tracker Options' which is a table with columns for 'Setting', 'Windows', and 'Linux/Unix'. The table lists settings: BenchmarkCPU, ComputerDomain, DateTime, and DetectMultiCore, each with a status icon and input fields for both operating systems.

| Setting | Windows | Linux/Unix |
|-----------------|--|----------------------|
| BenchmarkCPU |  <input type="text"/> | <input type="text"/> |
| ComputerDomain |  <input type="text"/> | <input type="text"/> |
| DateTime |  <input type="text"/> | <input type="text"/> |
| DetectMultiCore |  <input type="text"/> | <input type="text"/> |

Execute RayVentory Scan Engine Utilities from the Following UNC Shared Path on Target Device

This is a UNC share path from which RayVentory Scan Engine is available. The default installation does not install the tools, it is the responsibility of the administrator to set up a file share which is accessible by the machines scanned by the **Remote Execution Inventory Scan**.

Once the path is entered in the text field, the content of the share should be initialized. This process copied the required files (scan utilities) into the share specified by the user. To initialize the share, press **Install Scan Utilities**. This is a onetime operation, you can simply copy over the content of the folder and reuse it for another scan tools source paths.

Save inventory results from Target Devices on the Following UNC Share

This is the place where scanned devices save their inventory results. The default installation does not set up a file share for uploads, it is the responsibility of the administrator to configure it.

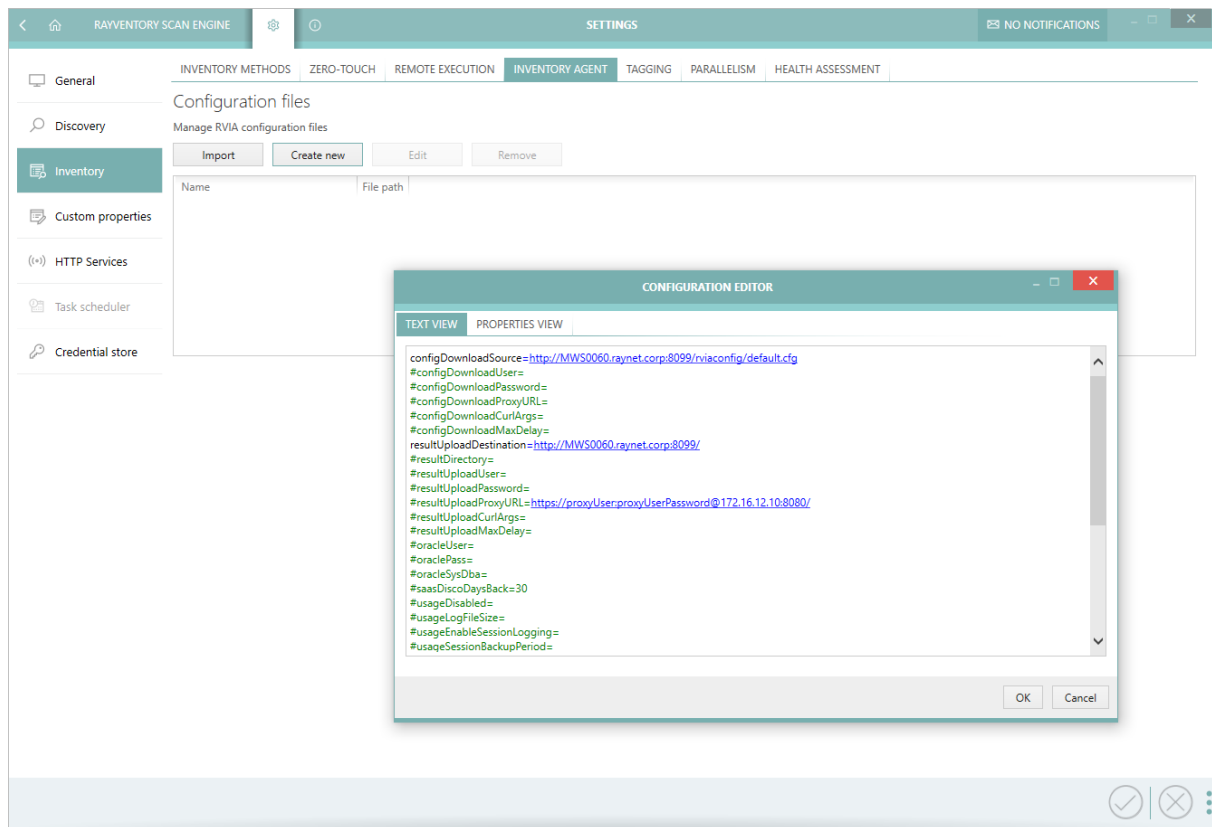


Note:

Certain inventory methods may require that all of some settings from this page are configured. Failing to configure them will render these inventory methods incompatible. You can find more details about the dependencies and requirements for each method in the following chapter: Inventory Methods Overview.

Inventory Agent

These settings control the configuration, which is served for the purpose of the RayVentory Inventory Agent.



RayVentory Scan Engine serves default values, which are reasonable for most of simple use cases. To further control the settings, new configurations may be added.

Creating or importing new configuration

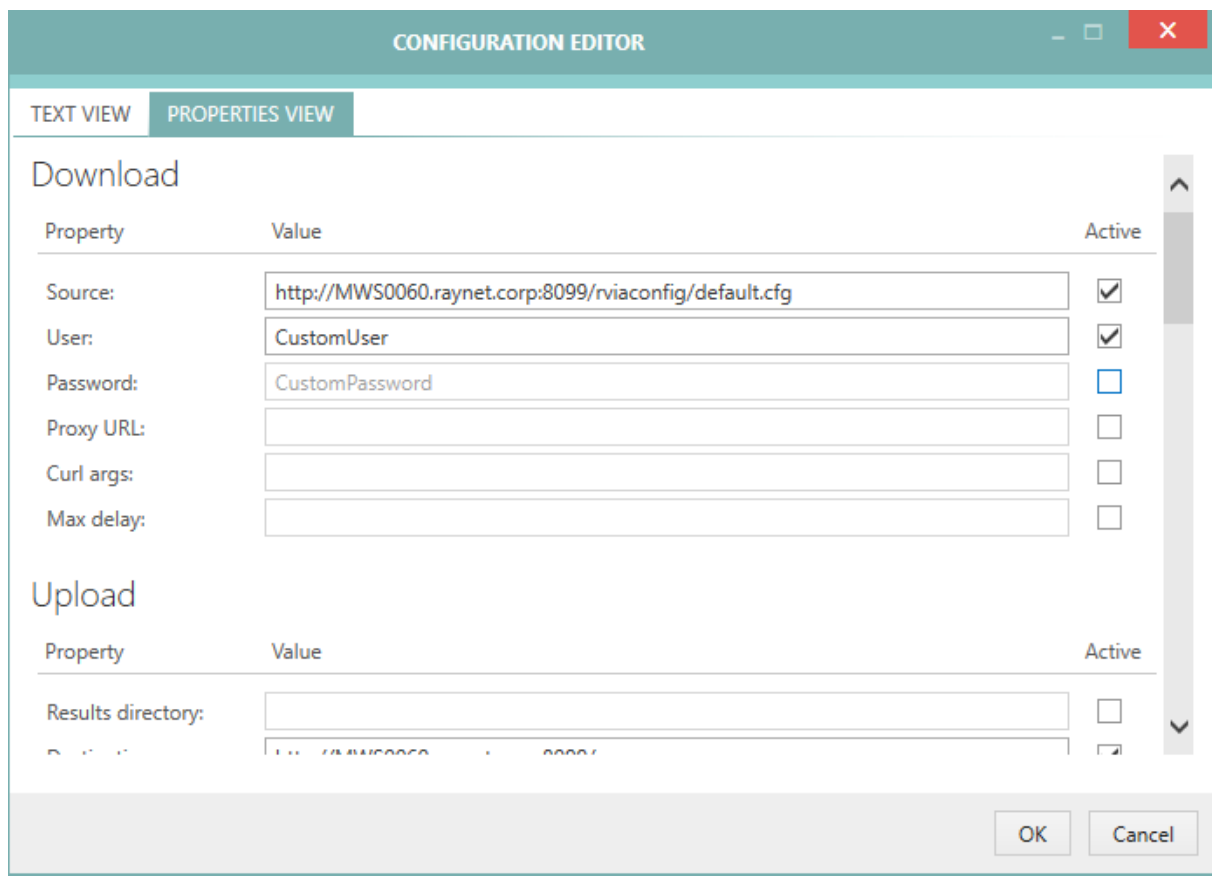
To add a new configuration:

1. Go to the **Settings** screen > **Inventory** > **Inventory Agent**
2. Press **Import** to import an existing file, or **Create new** to create a new one from a template
3. Edit the content of the file. The default values have many commented lines, which you can activate by removing the leading # character.
4. Once the configuration is finished, press **OK** to save the changes.

Chapter Inventory Agent (sections Configuration and Command-line) explains in details how to control getting the settings by the Inventory Agent.

Editing configuration with WYSIWYG editor

You can toggle between raw editing (default, great for experienced users and for quick copy and paste between different dialogs and config files) and a more structured form which guides you through the options exposed by the Inventory Agent.



The screenshot shows the 'CONFIGURATION EDITOR' dialog box with the 'PROPERTIES VIEW' tab selected. It contains two sections: 'Download' and 'Upload'.

Download Section:

| Property | Value | Active |
|------------|--|-------------------------------------|
| Source: | http://MWS0060.raynet.corp:8099/rviaconfig/default.cfg | <input checked="" type="checkbox"/> |
| User: | CustomUser | <input checked="" type="checkbox"/> |
| Password: | CustomPassword | <input type="checkbox"/> |
| Proxy URL: | | <input type="checkbox"/> |
| Curl args: | | <input type="checkbox"/> |
| Max delay: | | <input type="checkbox"/> |

Upload Section:

| Property | Value | Active |
|--------------------|-------|--------------------------|
| Results directory: | | <input type="checkbox"/> |

At the bottom right, there are 'OK' and 'Cancel' buttons.

Tagging

Tagging allows users to brand all incoming or outgoing NDI files with specific meta information, which can be used later on to identify the origin of each scan. This advanced feature is useful in case of multi-tier architecture of several RayVentory Scan Engine instances or many Distribution Servers / Upload locations.

This **Settings** screen contains three fields to be configured by the user:

Organization

This is the name of the organization performing the scan. You can enter any value that uniquely identifies your company, branch, or division.

Location

The location name. You can enter any value that uniquely identifies your physical, geographical, or infrastructure related location.

Contact

The information about the Point-Of-Contact for that particular scan. This can be a name, e-mail address, telephone number or any other value that uniquely identify the person responsible for the scan results.

The information provided by the user are merged with a few dynamically generated values. All NDI files receive the following information:

- Automatically created values:
 - The date and time of the scan (property `RVP-Time`).
 - The name of the server that initiated the scan (property `RVP-Server`).
 - The version of RayVentory Scan Engine instance that initiated the scan (property `RVP-Server`).
- Values specified by the user (only non-empty values are being written):
 - The name of the organization (property `Organization`).
 - The name of the location (property `Location`).
 - The name of the contact (property `Contact`).

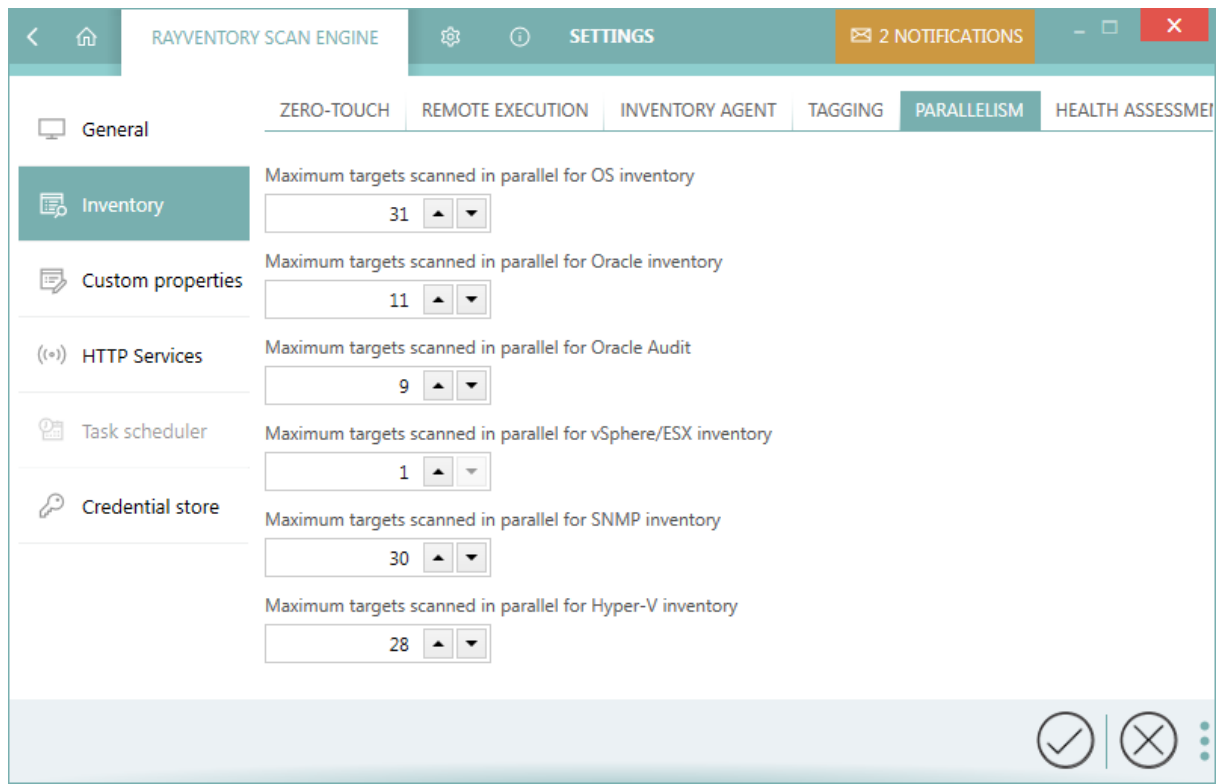
All values are saved in a special entry (Hardware), having `MGS_Identity` as its class name and `MGMT_API` as its evidence.

Conflict Handling

In case of processing of files that have already been branded by any other RayVentory Scan Engine, the original branding is kept intact. This way, the results always contain fine grained information about the low-level entity that initiated the scan.

Parallelism

Parallelism enables the user to set the maximum count of targets processed during an inventory run for each inventory type.

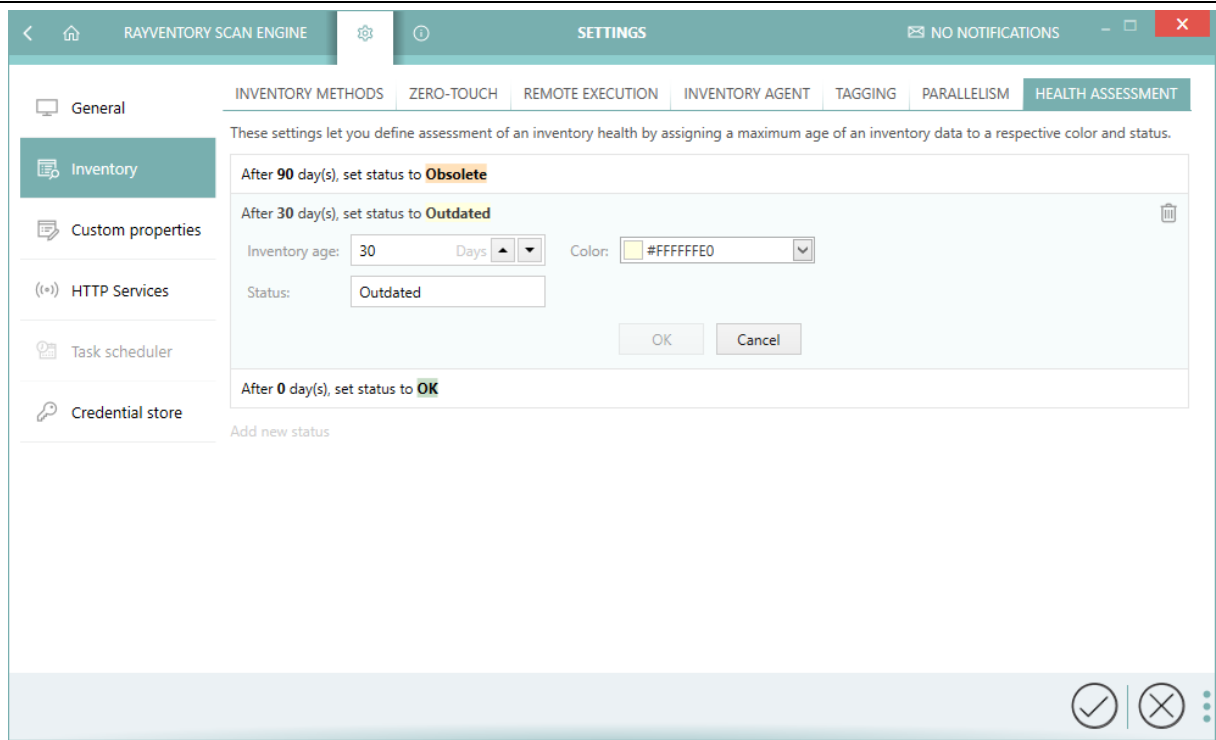


Health Assessment

This tab allows you to edit custom rules for assessing the status of an inventory.

The default rules are:

- A device that has been scanned within last 30 days is considered to be up-to-date (status OK).
- Otherwise, if a device has been scanned not later than 90 days ago, it is considered to be outdated (status Outdated).
- Finally, if the last time a device has been scanned is more than 90 days ago, then the result is considered to be obsolete (status Obsolete).



The list is read from the top to the bottom, and once the criteria are fulfilled the processing stops, and the result is determined from the defined name and color.

You can customize, remove or add custom statuses and rules.

In order to adjust an existing status...

1. Click on the status
2. Enter the adjusted details (the lower bound of days, name and the color).
3. Accept the changes by pressing OK.

In order to add a new status

1. Make sure no status is currently being edited.
2. Press **Add new status** link
3. Enter the required details (the lower bound of days, name and color).
4. Accept the changes by pressing OK.

In order to remove an existing status...

1. Click on the status
2. Click on the **Trash** icon on the right side

**Note:**

It is not possible to manually sort the list. Once the lower-bound of days is changed, the list will be sorted automatically to reflect the processing order of how RayVentory Scan Engine determines the status for each device.

Custom properties

This screen manages the custom properties, which are visible on device-base in the Devices screen.

For more details about how to work with custom properties, refer to the following chapter: [Defining custom attributes](#)

HTTP Services

This tab contains two sub-tabs that control the incoming (**Server** tab) and outgoing (**Upload Location** tab) uploads.

Server

These settings control how the built-in HTTP Upload Server is working.

Specify how scanned devices communicate with this machine to send their inventory files.

Port + protocol


Port number (default: 591)

IP address

- ☒ Any IP address
☐ Localhost only (127.0.0.1)
☐ Custom IP address

SSL Certificate

Full path to a certificate file (*.cer) to use safe HTTPS connections. Leave this field empty to use unencrypted traffic (HTTP).

Authentication

- ☒ Do not use authentication
☐ Use basic authentication

Note: Changing this setting will be effective from the next start of the application/service.

Port Number (Default: 8099)

This is the port number that the HTTP server is listening on. You can set it on any value not used by any other process. Typical values are 80 for HTTP connections and 443 for HTTPS, but to avoid any conflicts the default that RayVentory Scan Engine uses is set to 8099.

IP address

This is a setting which controls the addresses on which the built-in HTTP server listens. The default option (before version 12.2 the only available one) is to listen on any IP address, but this can be changed to only listen on `localhost` (127.0.0.1) or on a specific IP address.

SSL Certificate

In order to set up an encrypted traffic between the target devices and RayVentory Scan Engine, provide a full path to the `.cer` file containing your SSL certificate. The certificate authority has to be trusted on clients connecting to the server. Once a certificate is selected, HTTPS will be the default protocol for incoming connections. In order to revert back to unencrypted traffic, press the **Clear** button to make the path empty.

Authentication

Configuration of authentication options is not relevant for daily tasks started from RayVentory Scan Engine. These settings should be reviewed and applied when using custom scans triggered from local copies of `ndtrack.exe` combined with upload options.

The communication between target devices and RayVentory Scan Engine can optionally require

authentication. RayVentory Scan Engine supports two modes:

- No authentication
- Basic authentication

Selecting **Use basic authentication** requires that the target devices send user name and encoded password over the wire. If **No authentication** is chosen, then any device can upload its data to the local HTTP Upload server.



Note:

Basic authentication does send the credential in an unencrypted, encoded form. HTTPS connections should be used to secure the connection.



Be aware:

Depending on whether you installed the HTTP Upload Service or started a portable version of RayVentory Scan Engine, changing the HTTP settings requires a restart of the Service (installed instance) or of the main application (portable).

Upload Location

RayVentory Scan Engine is able to push software and hardware inventory results to a parent instance. The following parent instances are supported:

- RayVentory Scan Engine
- RayManageSoft Reporting Location
- RayVentory Reporting Location

RayVentory Scan Engine also supports the following upload locations:

- Local directory
- UNC folder path
- FTP address

These settings enables the user to control which parent instance is in use and how to upload data to it.

Setting up these values is not relevant if you do not intend to upload the data to any parent server (for example if your RayVentory Scan Engine instance is already self-contained root).

URL to Upload Inventory Files

The full URL to the parent upload location (see supported types section). This value should include the protocol and the port number. RayVentory Scan Engine uses this path for uploads of inventory files (.ndi). If you do not know this value, ask your administrator.

URL to Upload Legacy Discovery Data

The full URL to the parent upload location (see supported types section). This value should include the protocol and the port number. RayVentory Scan Engine uses this path for uploads of legacy discovery files (.disco). If

you do not know this value, ask your administrator.

Upload Legacy Discovery Data during Manually Triggered Inventory Upload

If this checkbox is checked, legacy `.disco` files will be uploaded to parent URL, as specified in the previous text box. If you uncheck this checkbox, the files will not be automatically uploaded.

Delay in seconds between discovery and inventory upload

The delay after uploading discovery data and before uploading inventory data. The delay is supposed to ensure that if the upload target was a RayVentory Server with direct import enabled for discovery and inventory then the server has enough time to process the discovery data before inventory import, in order to avoid missing inventory status updates for the network devices based views and reports.

Upload rules

This setting controls which files are accepted when uploaded to the built-in HTTP server. For supported NDI files, built-in routing is always applied to ensure they are correctly parsed. For any custom types, providing the name here will ensure that the file is accepted and eventually routed to the parent upload location.

Ignore Server Certificate Errors

If this checkbox is checked and HTTPS is used as the communication protocol, any SSL-related errors will be ignored.



Note:

The connection will be not secure if SSL-errors are ignored.

Certificate for Authentication against the Upload Endpoint

A custom certificate (`.cer`) which is used for encrypted connections via HTTPS protocol. If no value is provided, the standard Windows Certificate Store will be used.

Use Credentials from the Credential Store for Authentication to the Upload Endpoint

In case the parent location requires basic authentication, you can select the required credentials from the list. Credentials can be defined in the **Credential Store** screen.

Proxy Settings

Optional proxy settings used for communication with the parent upload location.

Task Scheduler

This screen enables the user to configure which credentials are used by the built-in Scheduling Service.

Scheduler credentials

Windows credentials used by the Scheduling Service.

You can leave the default option **None** in order to force RayVentory Scan Engine to use the current user identity.

If you want to execute the tasks impersonating another user, first ensure that his credentials (type Windows) are configured in the Credential Store and then select them from the drop-down list in this view.

Credential Store

This view is merely a shortcut to the **Credential Store Manager**, which is available in the **Devices + Services** screen.

Credentials store

These settings are available in a separate screen.

Manage credentials and assignments...

Press **Manage credentials and assignments...** to go to the configuration grid.

Scheduling

RayVentory Scan Engine offers a schedule for automation of operations.

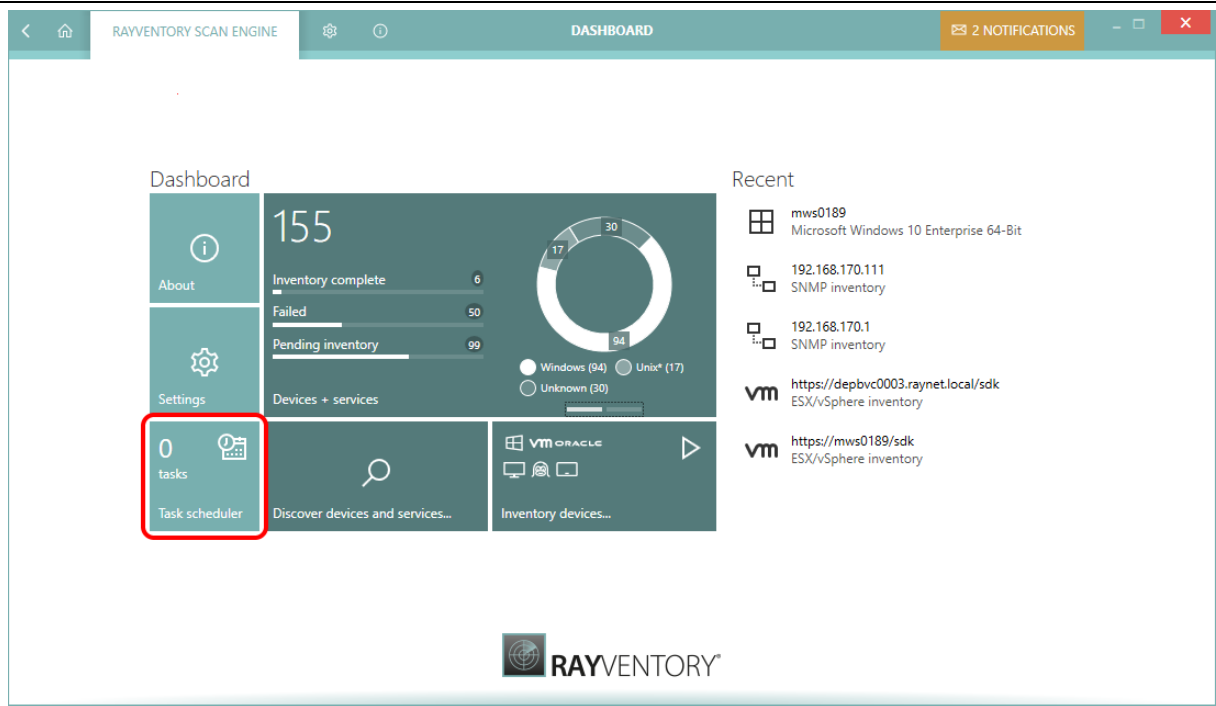
The following tasks and operations can be scheduled:

- Remote OS Inventory
- Oracle Inventory
- vSphere / ESX Inventory
- Discovery (with automatic inventory option)
- Inventory Upload

During the setup of RayVentory Scan Engine, the scheduler is installed as a service. It executes scheduled tasks in the background even if the RayVentory Scan Engine application is not running.

During the start-up of the RayVentory Scan Engine application, if no running scheduling service is found and if the scheduling service cannot be installed with the permissions of the currently logged in user, a warning is shown in the Notification Center and an instance of the scheduler is run within the RayVentory Scan Engine application itself. This instance of the scheduler will be stopped if the RayVentory Scan Engine application is closed. Therefore, the execution of the schedule will also be stopped when exiting RayVentory Scan Engine.

In the Schedule screen it is possible to create and edit the schedule. The Schedule screen can be opened by clicking on the Schedule tile which is located on the dashboard. The tasks which are listed in the schedule can also be triggered manually.



Triggers

Each scheduled task is triggered by one of four triggers:

- **Daily:** Once on a certain day at a certain tie for all or certain days of the week.
- **Daily recurring:** Multiple times a day, delayed by a certain number of minutes between the end of and the next run, for all or certain weekdays, from a certain time of day to a certain time of day.
- **Monthly:** Once a month on a certain day, on a certain time, on all or certain weekdays.
- **Monthly recurring:** Multiple times on a certain day of month, delayed by a certain number of minutes, between the end of a previous run and the next run, for all or certain weekdays from a certain time of day to a certain time of day.

For all triggers a date from when to begin and a date when the trigger will expire can be configured. It is also possible to set a limit of the number of times that a task is run. These settings are optional.

SCHEDULED TASK DIALOG

GENERAL

OPERATIONS

Name:

Description:

Trigger:

☒ Daily
 ☐ Daily recurring
 ☐ Monthly
 ☐ Monthly recurring

Schedule:

Days:

Mon, Tue, Wed, Thr, Fri, Sat, Sun

▼

Time:

00:00

▼

Limitations:

☐ Limited Executions:

1

▲ ▼

☐ Valid from

19.11.2018 12:45

▼

☐ Expires on

19.11.2019 12:45

▼

Other:

☒ Run asap after missing a trigger

OK

Close

Apply

Scheduled Operations

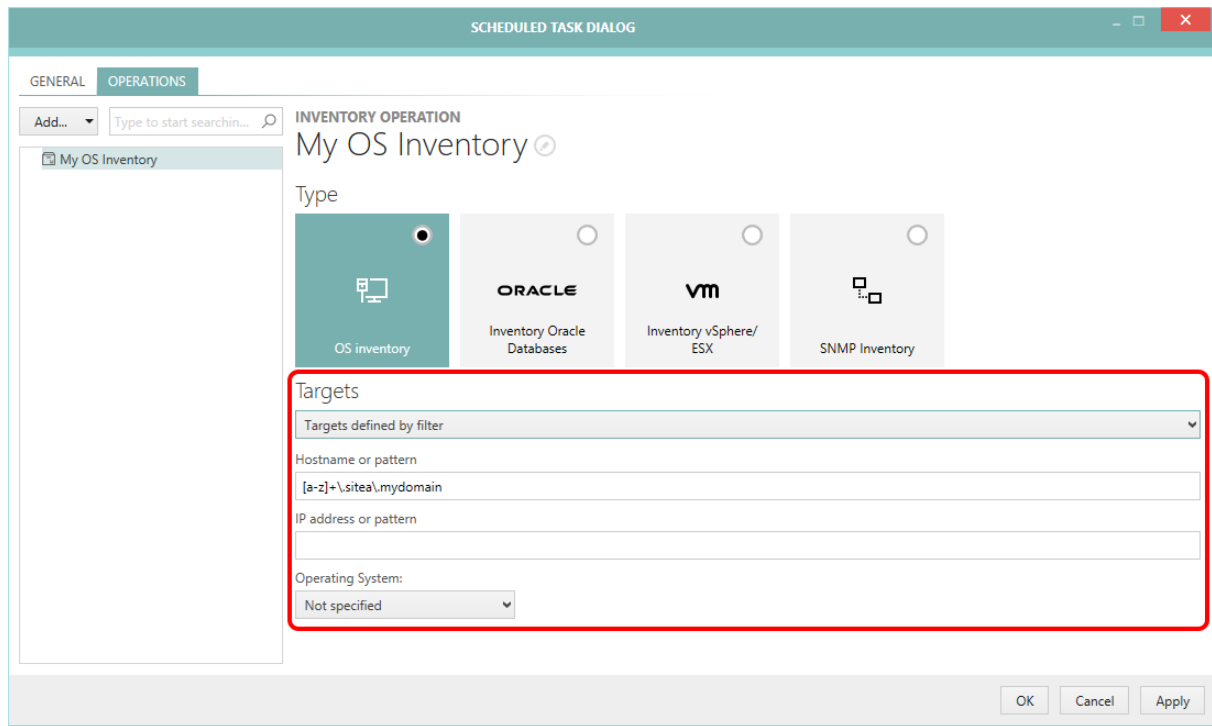
For each action or operation it is necessary to set a caption. A scheduled task consists of at least one action or operation.

Remote OS Inventory

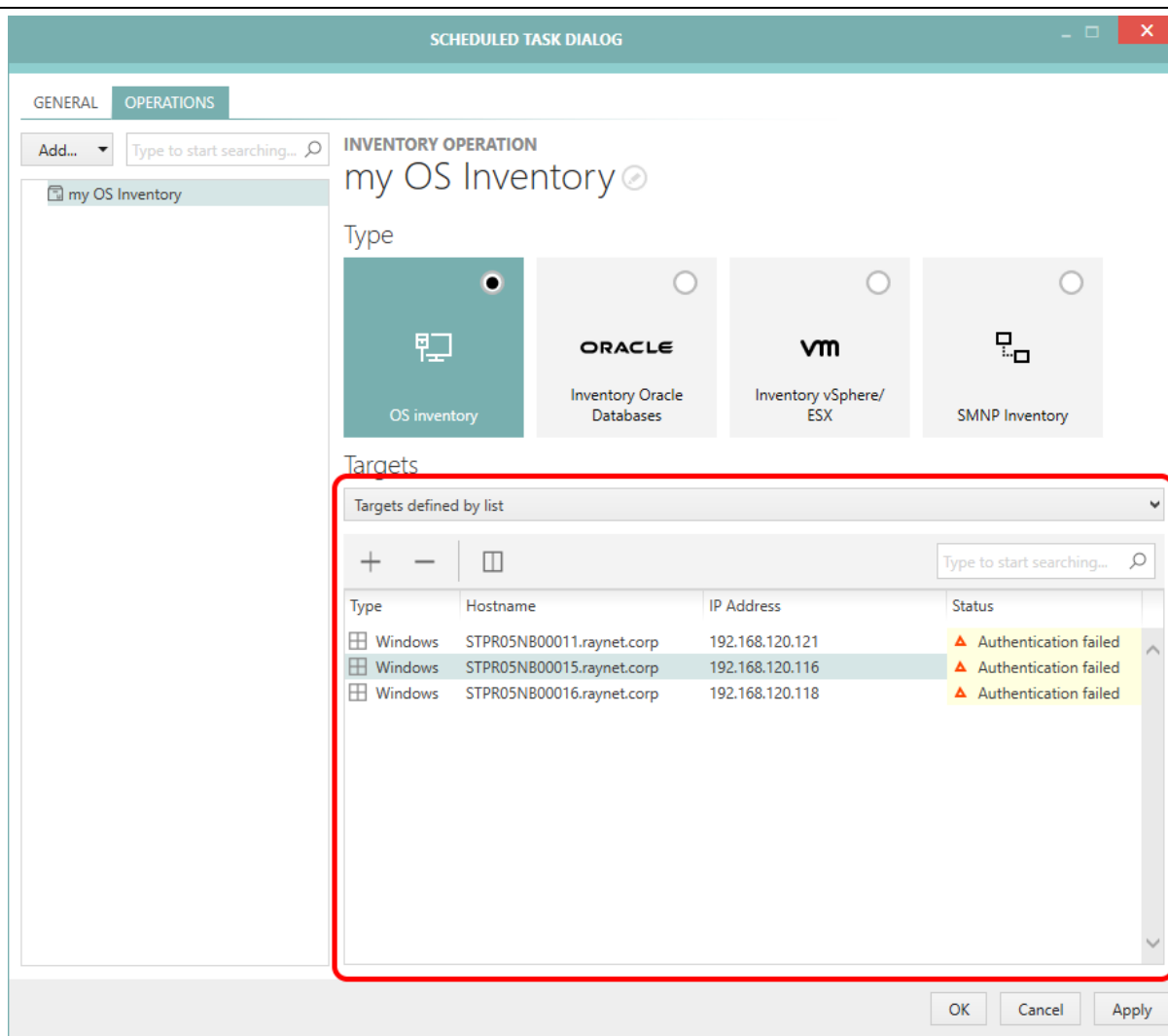
A remote OS inventory is performed by the OS Inventory action. The target host for this operation is defined by either a filter (**Targets defined by filter** option) or a list of hosts (**Targets defined by list** option).

The **Targets defined by filter** option has an optional filter for hostname and address which is labeled **Hostname or pattern** and which may contain a concrete hostname, address, or regular expression. Furthermore, the filter allows for the filtering by the type of the platform (**All**,

Windows, or **Unix/Linux**). This filter is applied to the **OS connections** list and used to find the set of hosts that is targeted by the inventory.



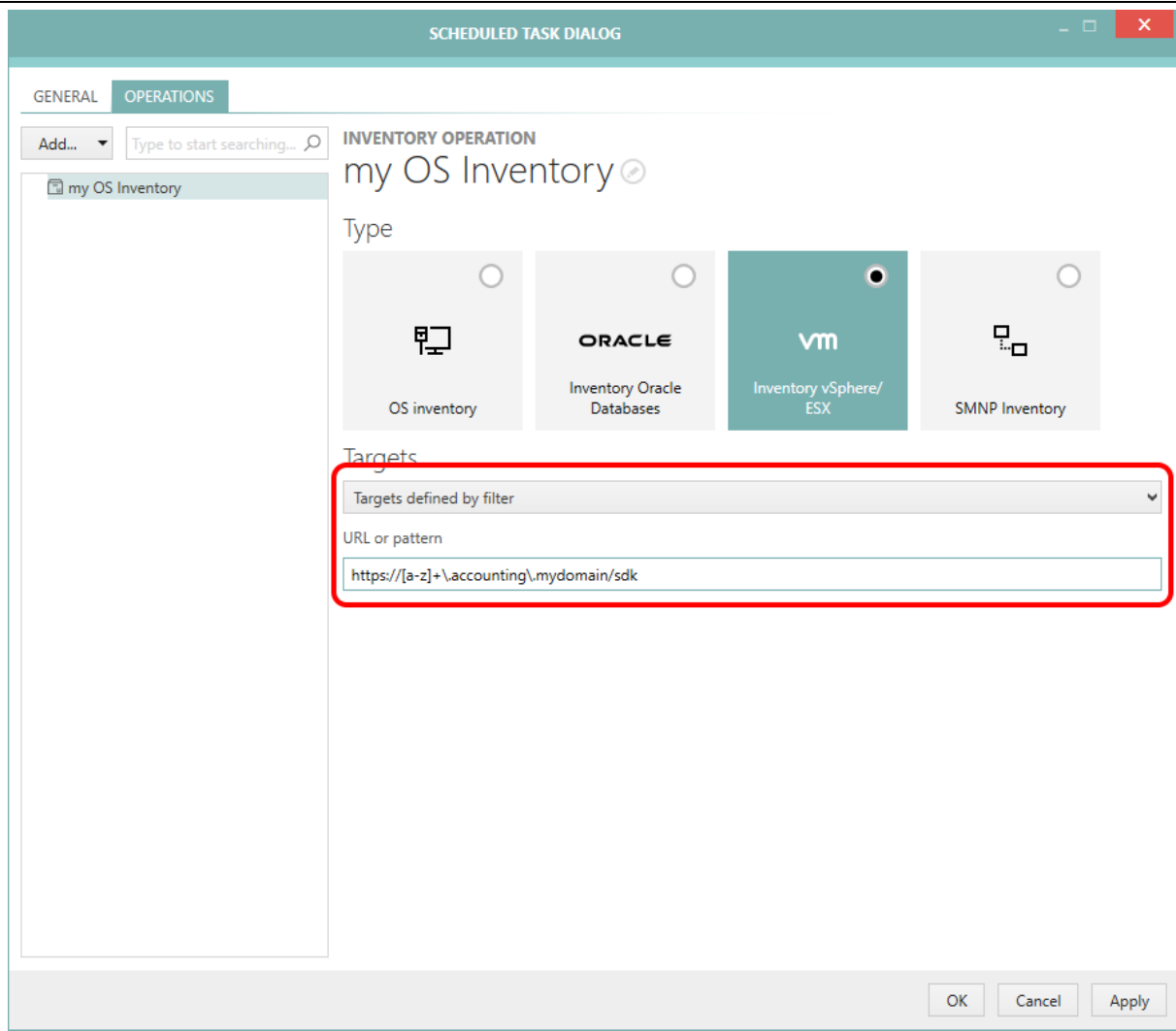
The **Targets defined by list** option allows to pick one or multiple hosts from the **OS connections** list.



VMware Inventory

A vSphere / ESX Inventory is performed by the vSphere Inventory action. The target hosts for this operation are either defined by a filter (**Targets defined by filter** option) or by a list of service endpoints (**Targets defined by list** option).

The **Targets defined by filter** option has an optional filter for the SDK service which is labeled **URL or pattern** and which can contain a specific URL or a regular expression. The filter is applied to the list of vSphere connections to find the set of hosts that is targeted by the inventory.

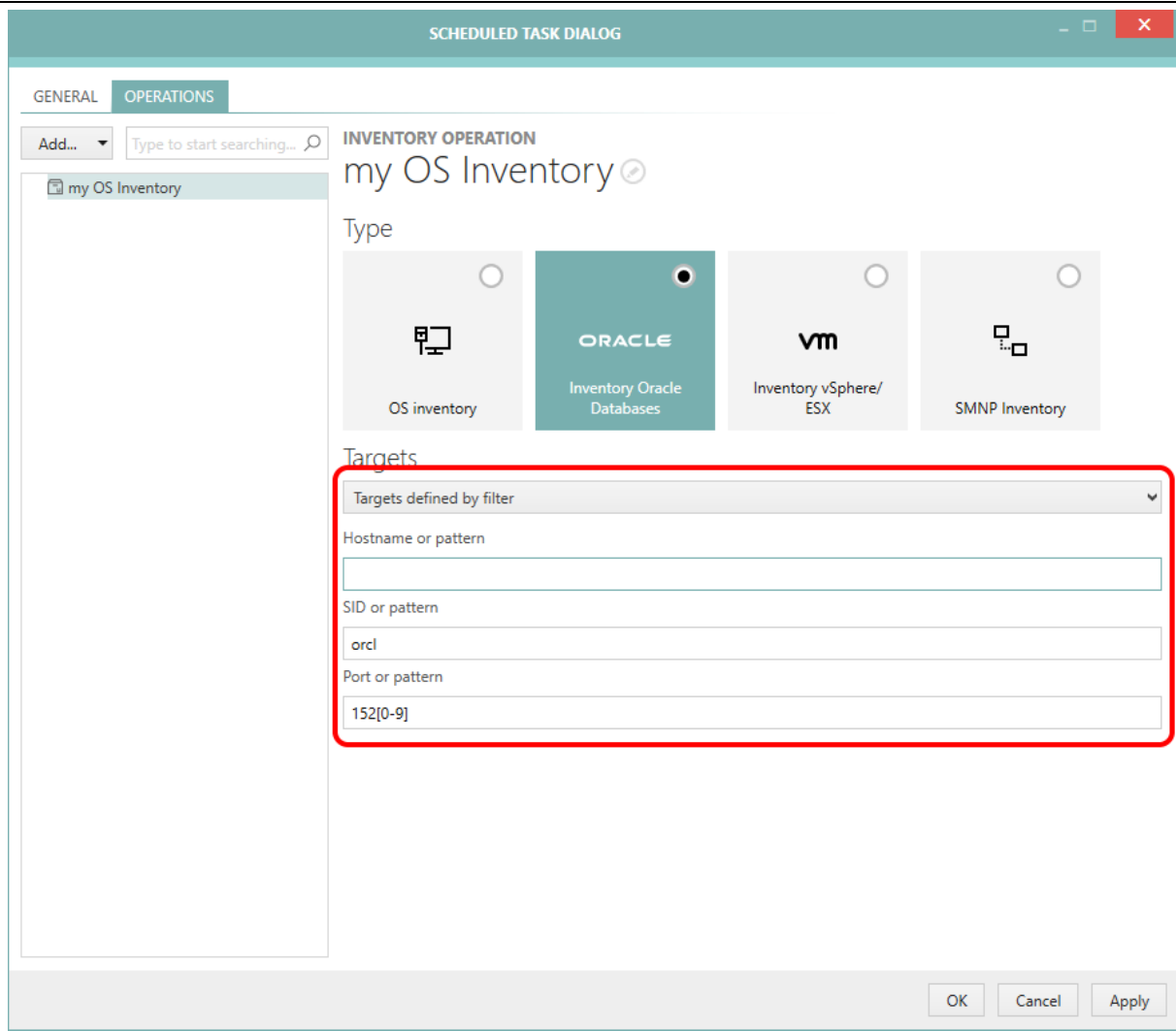


The **Targets defined by list** option can be used to pick either one or multiple hosts from the **vSphere connections** list.

Oracle Inventory

The Oracle Inventory action performs an Oracle inventory. The target databases for this operation by either a filter (**Targets defined by filter** option) or a list of databases (**Targets defined by list** option).

The **Targets defined by filter** option has an optional filter for hostname / address which is labelled **Hostname or pattern** and which can contain a hostname / address or a regular expression. Furthermore, there is an option to filter the target SID / service name for either a specific SID / service name or a regular expression, as well as for the target port or a regular expression. This filter is applied to the list of Oracle connections to find the set of databases that are targeted by the inventory.

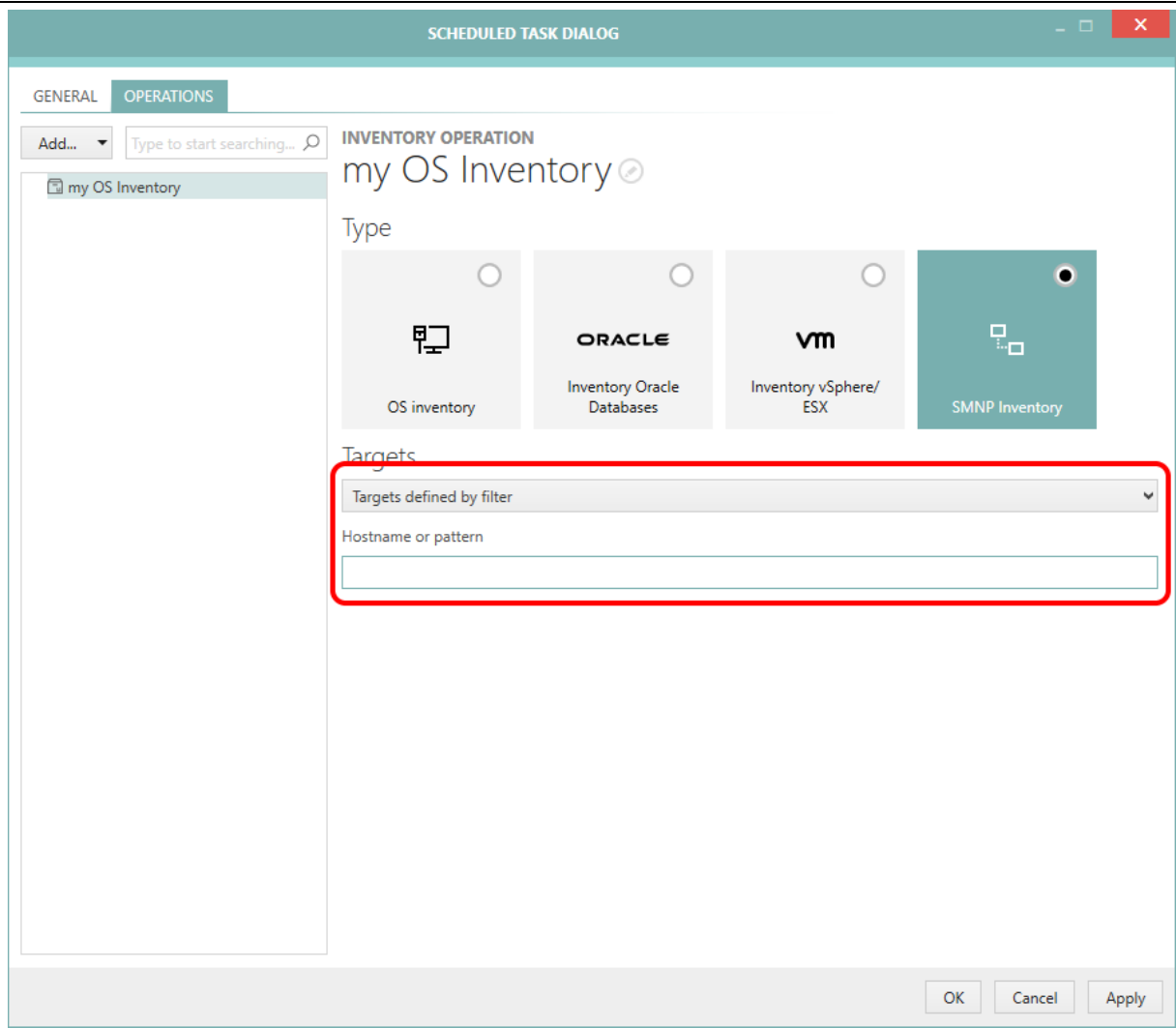


The **Targets defined by list** option allows to pick one or multiple databases from the **Oracle connections** list.

SNMP Inventory

An SNMP Inventory is performed by the SNMP action. The target hosts for this operation are either defined by a filter (**Targets defined by filter** option) or by a list of service endpoints (**Targets defined by list** option).

The **Targets defined by filter** option has an optional filter for the connection which is labeled **Hostname or pattern** and which can contain a specific URL or a regular expression. The filter is applied to the list of SNMP connections to find the set of hosts that is targeted by the inventory.



Discovery

The discovery action can be used to run the **Active Directory** import, the network scan, the service discovery / probing, and the automatic inventory in the same way as the discovery in the discovery screen, but the presentation differs. For further details on the available options and features, refer to the Discovery chapter.

SCHEDULED TASK DIALOG

GENERAL
OPERATIONS

Add...

New Discovery operation

DISCOVERY OPERATION
New Discovery operation

LDAP
PING SWEEP
DISCOVERY

Domain
Browse...

CN=Computers,DC=raynet,DC=corp

A semicolon separated list of domains to import computers from. You can also specify organizational units (OU) for example OU=yourOU1,OU=yourOU2,DC=yourdomain,DC=local

Filter
Preview...

Hostname filter

A regular expression that is matched against the full qualified host name of each computer found. Leave this empty to import unfiltered list of computers.

Fallback settings

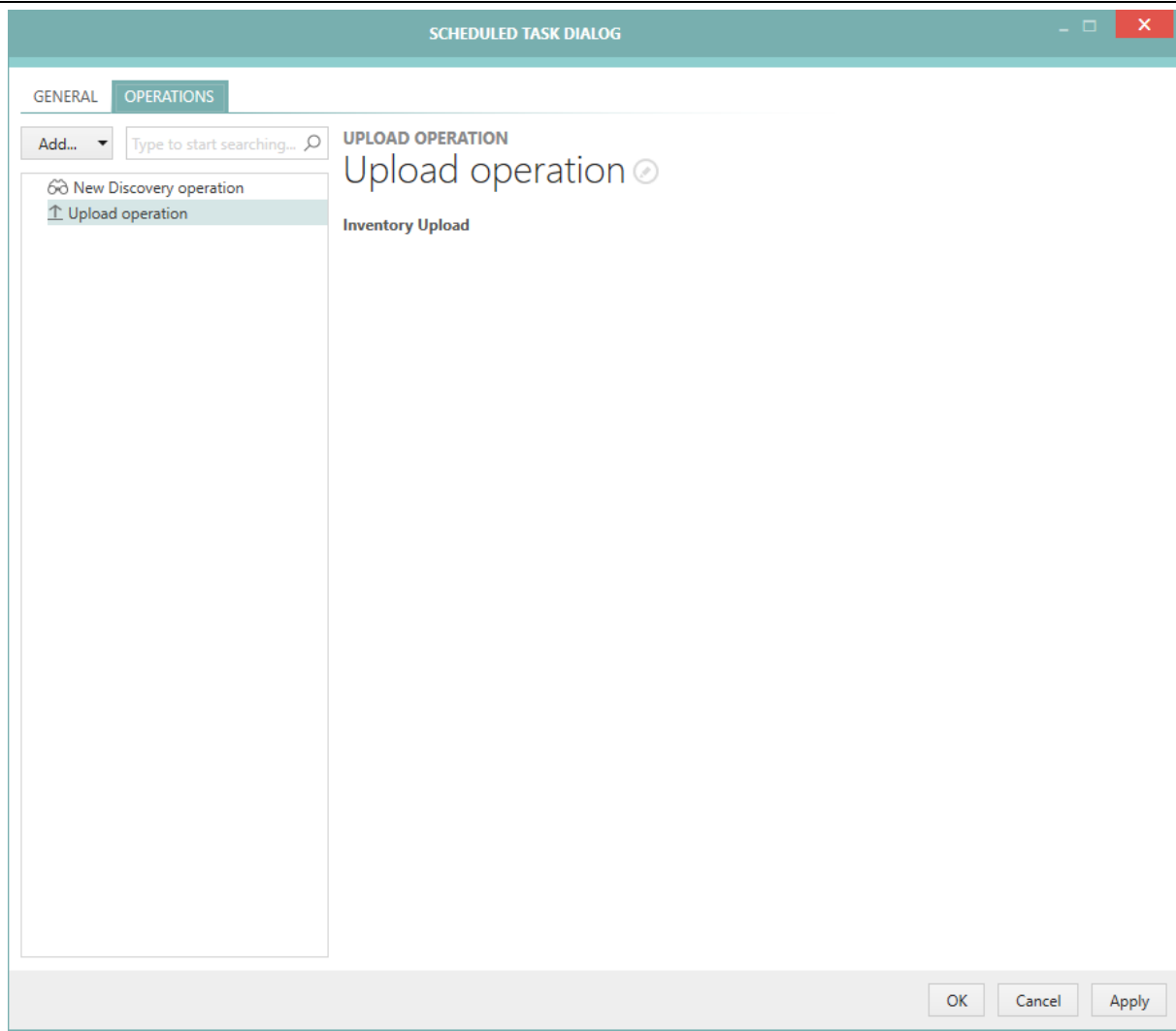
☐ Include devices that are disabled

☐ If a device is unreachable, treat it as Windows-based machine

OK
Cancel
Apply

Upload

There are no parameters for the **Upload** operation available in the schedule user interface. The upload operation will trigger an inventory upload identical to the operation that is triggered when the Upload tile in the **Notification Center** is being used.



Chaining Operations

It is possible to execute one or multiple actions / operations with a task. The operations will be executed one after another using the same order in which they appear in the action list starting at the top. The position of an operation can be shifted using the arrow buttons located in the action bar on top of the list. A checkbox is available for each operation in order to ignore errors that occur during the execution of the previous operation. If the checkbox is not activated, the whole task will be canceled if an error occurs during any of the previous operations. No further operations of this task will be executed until the task is triggered once more.

Composite Operations

The composite operation / action encompasses one or more actions / operations in one step. The actions / operations in a composite action / operation will be executed in parallel. When all operations of the composite operation are finished, the next action in the list will be started.

The schedule editor does not allow for the nesting of composite operations within composite operations.

SCHEDULED TASK DIALOG

GENERAL
OPERATIONS

Add...

Type to start searching...


New Composite operation

Sub-operation of New Composi...


Sub-operation of New Composi...

INVENTORY OPERATION
Sub-operation of New Composite operation ✎


Type




OS inventory



Inventory Oracle Databases



Inventory vSphere/ ESX



SNMP Inventory

Targets

Targets defined by filter
▼

Hostname or pattern

Operating System:

Not specified
▼

OK

Cancel

Apply

User Guide12.2

143

Inventory Agent

The **RayVentory Inventory Agent (RVIA)** is designed to continuously deliver hard- and software inventory and usage data from computer systems running Windows, Linux or Unix.

Inventory Agent has the following features:

- Inventory of computer hardware and software (Operating System inventory)
- Discovery and inventory of Oracle databases
- Upload of discovery/inventory results
- Download of configuration data
- Secure transfer of credentials for upload/download
- Scheduling of inventory and discovery operations, automatic download of configuration, uploading of results and execution of custom commands

Additionally, Windows devices running the Inventory Agent can use the following:

- Application usage metering (runs continuously as a service)
- SaaS discovery (for (web-)browser-based applications)

Installation

To install the agent, utilize Windows Installer (`msiexec`) and the available configurations from the MSI package.

The following MSI properties are most commonly used:

| Property | Description |
|-------------------------|---|
| INSTALLDIR | The installation folder Default value: C:\Program Files (x86)\RayVentory\ |
| CONFIGDOWNLOADSOURCE | The URL from which the configuration will be downloaded. This is a required property if installed in a non-interactive mode. |
| RESULTUPLOADDESTINATION | The URL for the upload location to which the NDI files will be uploaded. |
| SCHEDULEGETCONFIG | The default schedule for getting the configuration Default value: 25 0 * * * Default value represents downloading a new configuration file every day at 00:25 |
| SAASDISCODAYSBACK | Configures the SaaS discovery |

| | |
|-------------------|---|
| | Default: 30 days of browser history |
| USAGEAGENTDISABLE | Disabled usage agent (one of the following: <code>true</code> or <code>false</code>) Default value: <code>true</code> |

For a detailed description for each available config command, please, refer to the corresponding chapters.

Example

```
msiexec /V /IA:msi INSTALLDIR="C:\RayVentory Agent" SCHEDULEGETCONFIG="25 0 * * *"
CONFIGDOWNLOADSOURCE=http://192.168.123.123:951/rviaconfig/special.cfg
RESULTUPLOADDESTINATION=http://192.168.123.123:951/Inventories
```

This command installs the agent and configures it to download the configuration from the defined location every day at 00:25.



Be aware:

Changes in the configuration file may be later overridden by incoming configuration files pulled from the download location.

Configuration

After installing the agent, it operates using a configuration file called `rvia.cfg`, which can be found below the `ProgramData` directory. This file is created during the first run of the Inventory Agent and takes over the settings from `template.cfg`, which can be found in the Inventory Agent installation directory.

Configuration files can also be distributed remotely by the RayVentory Scan Engine (see `configDownloadSource` setting).

Default configuration

The following snippet shows the default configuration served by RayVentory Scan Engine:

```
configDownloadSource=http://localhost:8099/rviaconfig/default.cfg
configDownloadUser=
configDownloadPassword=
configDownloadProxyURL=
configDownloadCurlArgs=
configDownloadMaxDelay=
resultUploadDestination=
resultDirectory=
resultUploadUser=
resultUploadPassword=
resultUploadProxyURL=
resultUploadCurlArgs=
resultUploadMaxDelay=
oracleUser=
oraclePass=
```

```

oracleSysDb=
saasDiscoDaysBack=30
#usageWhitelist=\\winword\\.exe
#usageWhitelist=\\powerpnt\\.exe
#usageWhitelist=\\teams\\.exe
#usageWhitelist=\\outlook\\.exe
#usageDisabled=false;
#usageLogFileSize=4000000;
#usageEnableSessionLogging=false;
#usageSessionBackupPeriod=600;
#usageStartupDelay=300;
#usageMinRunTime=10;
#usageProcessUpdatePeriod=60
logLevel=info
logFileSizeLimit=
encryptionKey=
#schedule:command:echo Hello world! > /tmp/message.txt:0 0 * * *
schedule:getconfig::25 0 * * *
#schedule:saas::26 0 * * *
#schedule:inventory:-o IncludeDirectory=/opt:30 0 * * *
#schedule:oracle:-o user=smith -o pass=tiger -o asSysDBA=true:33 0 * * *
#schedule:upload::45 0 * * *
#schedule:command:curl.exe --output ""C:\\Program Files (x86)\\RayVentory
\\InventoryAgent\\whitelisted.xml"" -f http://rayventory\\:591/rviaconfig/
whitelisted.xml:45 0 * * *

```

Removing/Adding a hash (#) will enable or disable the entries.

Usage

The Agent either operates based on the schedules called from the configuration or when being called manually on the command line.

Command-line

To start the agent, execute the following command:

```
rvia <command>
```

where <command> is the name of one of the supported commands:

| Command | Description |
|-----------------------------|--|
| help | Shows help and available commands |
| inventory [<options>] | Runs OS inventory. This command may be followed by optional command line parameters for NDTRACK. |
| oracle [<options>] | Runs the Oracle database discovery and inventory. This command may be followed by optional command line parameters for ORATRACK. |
| saas [<number- of-days>] | Runs the SaaS discovery. The command may be followed by the number of days to look back in historic usage data. |

| | |
|---|--|
| <code>getConfig</code> [<source-host>] | Downloads the configuration |
| <code>schedule</code> | Applies the current schedule |
| <code>upload</code> [<destination-host>] | Uploads the results |
| <code>encrypt</code> [<key>] | Encrypts all usernames and password marked as blank. |

Example command:

```
rvia inventory
```

Example command with additional command line options:

```
rvia inventory -o MachineName=testbox
```

Usage agent

If enabled the Usage Agent will meter every application (OS related services will not be included) that is being used on the system.

To enable the usage agent set `usageDisabled` / `USAGEAGENTDISABLE` to `false`.

The usage agent can be configured with the following configurations which are only available for usage in a config file:

| Options | Description |
|---|---|
| <code>usageLogFileSize</code> | Max size (in bytes) of the usage log written by the agent. Default value: 4000000 |
| <code>usageEnableSession Logging</code> | If set to <code>true</code> a usage log is written which contains all metering information. This should be used for debugging purposes only. Default value: <code>false</code> |
| <code>usageSessionBackup Period</code> | Time span (in seconds) specifying how often the information is dumped from memory into a cache file. Default value: 3600 |
| <code>usageStartupDelay</code> | Time span (in seconds) specifying when usage agent should start operating after booting Default value: 300 |

| | |
|--------------------------|---|
| usageMinRunTime | Time span (in seconds) specifying how long application runs until the usage gets picked up by the agent. Default value: 60 |
| usageProcessUpdatePeriod | Time span (in seconds) specifying how often running processes should be pulled. Default value: 60 |

White listing

To restrict the metering to a list of applications it is possible to add a whitelist to the Agent. To white-list certain processes, add lines for the setting `usageWhitelist` to the configuration file.

On successful download, these settings will be automatically applied by creating from them a file of the name `whitelisted.xml` in the Inventory Agent installation directory.

This way your process white-lists can be distributed along the other configuration settings.



Be aware:

The values for white-listed processes must represent valid Regular Expressions. This is why in the following examples special characters "." (dot) and "\" (backslash) are escaped with another backslash.

```
usageWhitelist=\\winword\\.exe
usageWhitelist=\\powerpnt\\.exe
usageWhitelist=\\teams\\.exe
usageWhitelist=\\outlook\\.exe
```

You can also manually create a file containing required definitions for white-listing. The file should have an XML syntax, for example:

```
<Whitelist>
  <Process Path="\\winword\\.exe" Regex="true"/>
  <Process Path="\\powerpnt\\.exe" Regex="true"/>
  <Process Path="\\teams\\.exe" Regex="true"/>
  <Process Path="\\outlook\\.exe" Regex="true"/>
</Whitelist>
```

Save it in the root directory where the agent was installed, under the name `whitelisted.xml`.

Alternatively, you may create such a file on your RayVentory Scan Engine host, in the directory for Inventory Agent configuration files.

In this case, the white-list could also be downloaded using Inventory Agent scheduled arbitrary command line feature, by adding a command to download into the configuration file.

Example:

```
schedule:command:curl.exe --output ""C:\Program Files (x86)\RayVentory
\InventoryAgent\whitelisted
```

Scheduling

The following operations can be automated by setting up scheduling:

- OS inventory
- Oracle Database Discovery and Inventory
- Upload of discovery/inventory results
- Download of configuration data
- SaaS Discovery
- Horizon connection metering
- Arbitrary commands

Inventory Agent for Windows uses the Windows Task Scheduler to execute its scheduled tasks. On UNIX/Linux, cron is used instead.

A scheduled task is defined in the configuration file by a single line, which needs to follow a specific set of rules:

```
<command >:<options>:<schedule-pattern>
```

Valid values for command are:

| Command | Description |
|-----------|--|
| command | Executes an arbitrary command |
| oracle | Runs Oracle database inventory |
| horizon | Gather Horizon connection information. |
| saas | Runs SaaS discovery |
| getconfig | Download a configuration file |
| upload | Uploads the results |

The triggering-pattern is inspired by cron. On Windows, this pattern is interpreted to translate it to the Windows Task Scheduler triggering scheme. Therefore, only a small subset of cron triggering patterns, including the keywords logon and logoff, can be used for Windows.

Overview of possible schedule patterns:

```
<Minute> <Hour> <DayOfMonth> <Month> <DayOfWeek>
```

| Character | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|------------------|---|
| (1) <Minute> | Minute. Either a range (0-59) or * for unspecified value. |
| (2) <Hour> | Hour. Either a range (0-23) or * for unspecified value. |
| (3) <DayOfMonth> | The day of the month. Either a range (1-31) or * for unspecified value. |
| (4) <Month> | The month. Either a range (1-12) or * for unspecified value. |
| (5) <DayOfWeek> | The day of the week. Either a range (0-7) or * for unspecified value. Use 0 for Sunday, 1 for Monday, 2 for Tuesday etc. The value of 7 means Sunday,. |

Examples:

```
schedule:getconfig::25 0 * * *
```

Download a new configuration file every day at 00:25.

```
schedule:saas::26 0 * * *
```

Run a SaaS discovery every day at 00:26.

```
schedule:inventory:-o IncludeDirectory=/opt:30 0 * * *
```

Run an inventory with an additional command line option for NDTRACK, every day at 00:30.

```
schedule:oracle:-o user=smith -o pass=tiger -o asSysDBA=true:33 0 * * *
```

Run an Oracle discovery and inventory with an additional command line option for ORATRACK, every day at 00:33.

```
schedule:upload::45 0 * * *
```

Upload the discovery, metering and inventory results, every day at 00:45.

Logging

- **logLevel**
This setting sets the detail level for the Inventory Agent log file. The default log level is `info`. Valid values are (from less to more verbose):
 - `off`
 - `fatal`
 - `error`
 - `warn`
 - `info`
 - `debug`
 - `trace`
 - `all`
- **logFileSizeLimit**
This setting limits the size of the Inventory Agent log file. By default, the size is not limited. The log file is reset/deleted whenever Inventory Agent finds the file to exceed the maximum size before writing a new log entry.

Commands

encrypt

This command will encrypt all credentials which are not marked as encrypted by the `$$` prefix.

The encryption scheme used is AES128 CBC and the encrypted credentials are encoded as base64. Inventory Agent will use a default value for the encryption key unless you specify an encryption key in the configuration file by setting `encryptionKey` or provide an encryption key with the encrypt command.

**Note:**

This command is not available for scheduling.

getconfig

This command will attempt to download a configuration file from a defined host.

Inventory Agent will attempt to download from one of the source URLs in the configuration file unless you include an URL after the command. If an URL was specified, then Inventory Agent will not try any other URLs on error.

After a successful download, Inventory Agent will attempt to apply its schedule to the systems scheduler (delete previously defined scheduled tasks and create new scheduled tasks). Further, usage metering related registry settings will also be applied and the file `whitelisted.xml` for the usage metering service is written if needed.

You can force Inventory Agent to try to apply the schedule in the current configuration file (see command *schedule*). You need to add a source URL by adding a line for the setting `configDownloadSource` to the configuration file.

Example of downloading a configuration file from a location specified in the configuration file (from command line):

```
rvia getconfig
```

Example of downloading a configuration file from a specific URL:

```
rvia getconfig http://192.168.123.123:951/rviaconfig/special.cfg
```

Example of two download sources in the configuration file:

```
configDownloadSource=http://192.168.123.123:951/rviaconfig/special.cfg  
configDownloadSource=http://192.168.123.123:951/rviaconfig/default.cfg
```

Bear in mind that RayVentory Scan Engine will always host a basic configuration file (which sets basic auth credentials and upload- and download locations, according to the RVSE configuration) at

`http://<yourhostname>:<configured-port>/rviaconfig/default.cfg`

or

`https://<yourhostname>:<configured-port>/rviaconfig/default.cfg`

The configuration can be managed from the Settings screen.

Mind that you can use the settings `configDownloadUser` and `configDownloadPassword` to specify credentials for basic authentication during download. See command *encrypt* for details.

You can specify a proxy (including credentials) by the setting `resultUploadProxyURL` for the upload.

Example:

```
configDownloadProxyURL=https://proxyUser:proxyUserPassword@172.16.1.1:8080/
```

To pass additional arguments to curl for download, use the setting `configDownloadCurlArgs`.



Note:

For large numbers of Inventory Agent instances, you should use `configDownloadMaxDelay` to delay individual downloads to a maximum of the specified number of seconds. That will spread the network load over the time domain.

horizon

This enhances the usage agent to also gather information about Horizon connections. If enabled, by default the horizon feature is scheduled to run during logon and logoff.

oracle

The **oracle** command discovers local Oracle databases and gathers license relevant inventory data from these databases. This command accepts further optional command line arguments for ORATRACK, the tool that RVIA uses to discovery and inventory Oracle databases.

Example command for command line:

```
rvia oracle
```

Example for command line with additional command line options:

```
rvia oracle -o user=sys -o pass=5ecr3T -o asSysDb=TRUE
```

Bear in mind that you do not have to pass credentials this way; RVIA has got settings to accept encrypted credentials in its configuration file. The settings are `oracleUser`, `oraclePass` and `oracleSysDb`.

saas

This command runs the SaaS Discovery for browser-based applications. By default, the SaaS Discovery will consider the browser data regarding the past 30 days. You can override that by the setting `saasDiscoDaysBack` or provide a number on the command line.

schedule

This command will simply try to apply the schedule from Inventory Agent configuration file to the system scheduler. After editing a configuration file in-place, you can use this command to apply that schedule.



Note:

This command is only available from a CLI interface.

settings

This command will try to apply the registry settings and usage metering whitelist from the Inventory Agent configuration file to the system scheduler resp. registry and white-list file. After editing a configuration file in-place, you can use this command to apply these settings.



Note:

This command is not available for scheduling.

upload

The upload command will attempt to upload all files from Inventory Agent results directory to one of the result destinations.

Add a destination URL by adding a line for the setting `resultUploadDestination` to the

configuration file. You can use the settings `resultUploadUser` and `resultUploadPassword` to specify credentials for basic authentication during upload. See command *encrypt* for more details.

You can specify a proxy (including credentials) by the setting `resultUploadProxyURL` for the upload.

Example:

```
resultUploadProxyURL=https://proxyUser:proxyUserPassword@172.16.1.1:8080/
```

To pass additional arguments to curl for upload, use the setting `resultUploadCurlArgs`.



Note:

For large numbers of Inventory Agent instances, you should use `configDownloadMaxDelay` to delay individual downloads to a maximum of the specified number of seconds. That will spread the network load over the time domain.

Advanced Topics

This chapter covers some advanced topics and refers to technical details for IT professionals and administrators that help to understand how RayVentry Scan Engine works and how to use most of its functionalities.

Receiving Uploads from Remote Scans

RayVentry Scan Engine exposes a lightweight upload HTTP server, that accepts incoming inventory files (.ndi) and legacy discovery files (.disco). It uses a zero-touch configuration which by default starts a HTTP server listening on port 8099 without authentication.

The server can be started in one of the following ways:

- If Windows Service for HTTP uploads has been installed, the service is automatically started by Windows.
- Otherwise or if the service is stopped, RayVentry Scan Engine starts a local private HTTP server upon launch and closes it upon closing the main application.

To control firewall access rules, the following executable can be added to firewall rules:

```
[INSTALLDIR]\RayVentryScanEngine.HttpUploadServer.exe
```

The default port, authentication, and certificate settings can be changed in the RayVentry Scan Engine options. You can find out more information about the server configuration in the following chapter: Settings > HTTP Services > Server.

Once configured, the HTTP upload service is automatically used for remote-execution methods that collect the results from target devices via HTTP(S) upload (see more on that in chapter Inventory Methods Overview).

You can also use the HTTP upload service to send results from custom implementations of NDTRACK / ORATRACK scans, by using command line parameters to redirect the output to a specified upload location. The configuration requires a full URL to the local upload server. You can specify it on your own, knowing the configured port and full machine name or let RayVentry Scan Engine help you. Simply press the button **NOTIFICATION CENTER** in the upper-right corner to reveal a pop-up menu. In this flyout, a separate entry is dedicated to the status of the HTTP service, with the possibility to copy its full URL address.

Uploading Results to Parent Servers

The upload transports the collected inventory files to the HTTP / HTTPS endpoint configured by the setting Upload URL in the UPLOAD section of the settings. Depending on the implementation, it may be any of the following:

- Another RayVentory Scan Engine instance
- RayManageSoft Reporting Location
- RayVentory Reporting Location

RayVentory Scan Engine also supports the following upload locations:

- Local directory
- UNC folder path
- FTP address

An upload URL for the default configuration would look like `http://myRayVentoryServerHostname/ManageSoftRL/inventories` (when reusing the RayVentory / RayManagesoft Reporting Location) or `http://myRayVentoryScanEngineHostname:port/` (when using HTTP Upload Server provided by the parent RayVentory Scan Engine instance).

An upload can be triggered by clicking on the **Upload** tile in the Notification Center. The button is only visible if the upload location is configured and any of the following is true:

- There are some pending uploads.
- The option to upload legacy discovery files is active AND the upload location for legacy discovery files is set.

Configuring Parent Server URL and Authentication

The upload feature supports authentication by providing a username / password pair that is taken from the Windows type credentials from the credential store or the authentication by a client certificate. You can learn more about parent upload settings in the following chapter: Upload Location.

Providing Custom HTTP Upload Handlers

The parent upload server can be configured as a custom Microsoft IIS web service that is using anonymous authentication and allows HTTP PUT for compressed (`.ndi.gz`) and uncompressed NDI files (`.ndi`).

Inventory Methods Overview

This chapter contains detailed descriptions of every supported inventory method together with their impact on the security, the architecture, and general recommendations.

The availability of methods for given jobs may be limited by the following factors:

- Job type (Windows / UNIX / Oracle / vSphere / SNMP etc.)
- Device type (Windows / UNIX)
- Declared device capabilities

- RayVentry Scan Engine settings (HTTP server, Zero-Touch and Remote-Execution settings)
- Current selection of active Inventory methods
- Currently installed software (for example Java runtime)
- Ports opened on the target device
- Other factors

The following guide will help you understand these dependencies and find out which methods are suitable in your environment.

The methods are divided into the following groups:

- Windows Inventory
- Linux / UNIX Inventory
- Oracle Audit
- Oracle Inventory
- SNMP Inventory
- ESX / vSphere Inventory

Windows Inventory

These inventory methods gather basic hard- and software inventory data on the target device from the Windows OS. The inventory methods operate on targets of the type **Device**.

Zero-Touch Inventory by WMI / WINAPI on Windows

Description

The Zero-Touch OS / platform inventory for Windows hosts using WMI queries. Using WINAPI based Windows-Registry queries as a fallback for earlier Windows versions that do not provide the StdReg-WMI-Provider.

Usage and Recommendation

This is the least invasive inventory method for Windows.. The only requirements are sufficient privileges to run all required WMI queries to RayVentry Scan Engine. This inventory method can be customized to gather additional data, available via WMI or from the Windows registry.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Zero-touch
 - WMI
- **Prerequisites:**
 - None

Remote-Execution Inventory by Service Manager / SMB Local Files on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by a temporary local copy of NDTRACK pushed via SMB, executed by a temporary service, and inventory copied via SMB.

RayVentry Scan Engine mounts the target's built-in share `ADMIN$`, copies the NDTRACK files to a temporary subdirectory of `ADMIN$\Temp\`, and starts the NDTRACK via a temporary service. Eventually, the resulting inventory file is copied to the RayVentry Scan Engine host. Later, the temporary service and the temporary directory are deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch inventory on Windows. This inventory method uses NDTRACK.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Access to the File System
 - Remote execution
 - Windows Service Manager
- **Prerequisites:**
 - None

Remote-Execution Inventory by WMI / SMB Local Files on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by a temporary local copy of NDTRACK pushed via SMB, executed via WMI, and inventory copied via SMB.

RayVentry Scan Engine mounts the target's built-in share `ADMIN$`, copies the NDTRACK files to a temporary subdirectory of `ADMIN$\Temp\`, and starts the NDTRACK by via a temporary service. Eventually, the resulting inventory file is copied to the RayVentry Scan Engine host. Later, the temporary directory is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch inventory on Windows. This inventory method uses NDTRACK.

Technical details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Access to the File System
 - Remote execution
 - WMI
- **Prerequisites:**
 - None

Remote-Execution Inventory by Service Manager Upload HTTP(S) on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by NDTRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed by a temporary service, and inventory uploaded via HTTP(S).

RayVentory Scan Engine starts a temporary service that references the service executable, located on the RayVentory Scan Engine utilities share, which starts the NDTRACK, located on the same share. NDTRACK will upload its results to the RayVentory Scan Engine HTTP Service. Later, the temporary service is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch inventory on Windows. This inventory method uses NDTRACK.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Remote execution
 - Windows Service Manager
- **Prerequisites:**
 - Configured and reachable RayVentory Scan Engine HTTP Service

Remote-Execution by Service Manager Upload SMB on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by NDTRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed by a temporary service, and

inventory uploaded via SMB to the UNC path for RayVentory Scan Engine inventories.

RayVentory Scan Engine starts a temporary service that references the service executable located on the RayVentory Scan Engine utilities share which starts the NDTRACK, located on the same share. NDTRACK will upload its results to the RayVentory Scan Engine inventories share. Later, the temporary service is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the database are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Remote execution
 - Upload to the SMB shares
 - Windows Service Manager
- **Prerequisites:**
 - SMB share to receive inventory files
 - The setting **Save inventory results from target devices on the following UNC share** must be configured in the **Remote execution** section of the **Settings** screen

Remote-Execution by WMI Upload HTTP(S) on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by NDTRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed via WMI, and inventory uploaded via HTTP(S).

RayVentory Scan Engine starts the NDTRACK, located on the RayVentory Scan Engine utilities share via WMI (`Win32_Process`). NDTRACK will upload its results to the RVP HTTP Service.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Remote execution
 - WMI
- **Prerequisites:**
 - The setting **Execute RayVentory Scan Engine Utilities from the following UNC shared**

path on target device must be configured in the **Remote execution** section of the **Settings** screen

- The scan utilities must be present on the shared location (for example installed via the **Install Scan Utilities** button in the **Remote execution** section of the **Settings** screen)
- A configured and reachable RayVentory Scan Engine HTTP Service

Remote-Execution by WMI Upload SMB on Windows

Description

Remote-Execution OS / platform inventory for Windows hosts by NDTRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed via WMI, and inventory uploaded via SMB to the UNC path for RayVentory Scan Engine inventories.

RayVentory Scan Engine starts the NDTRACK, located on the RayVentory Scan Engine utilities share, via WMI (`Win32_Process`). NDTRACK will upload its results to the RayVentory Scan Engine inventories share.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Windows
- **Required capabilities:**
 - Remote execution
 - Upload to the SMB shares
 - WMI
- **Prerequisites:**
 - The SMB share to receive inventory files
 - The setting **Execute RayVentory Scan Engine Utilities from the following UNC shared path on target device** must be configured in the **Remote execution** section of the **Settings** screen
 - The scan utilities must be present on the shared location (for example installed via the **Install Scan Utilities** button in the **Remote execution** section of the **Settings** screen)
 - The setting **Save inventory results from target devices on the following UNC share** must be configured in the **Remote execution** section of the **Settings** screen

Linux / UNIX Inventory

These inventory methods gather basic hard- and software inventory data on the target device from the Linux / UNIX-like Operating Systems. The inventory methods operate on targets of the type **Device**.

Zero-Touch Inventory by WMI / WINAPI on Linux / UNIX

Description

Zero-Touch OS / platform inventory for Linux / UNIX hosts by RIU, using SSH remote commands to query platform specific standard configuration and diagnosis tools and utilities.

Usage and Recommendation

This is the least invasive inventory method for Linux / UNIX-like Operating Systems. The only requirements are sufficient privileges to run all the needed commands and utilities or use privilege elevation by `sudo`, `pbrun` or `priv` for RayVentory Scan Engine.

Technical Details

- **Used credential type:**
 - SSH
- **Required capabilities:**
 - Zero-touch
- **Prerequisites:**
 - None

Remote-Execution Inventory by SSH / SCP Local Files on Linux / Unix

Description

Remote-Execution OS / platform inventory for Linux / Unix by a temporary local copy of NDTRACK pushed via SCP, executed via SSH, and inventory copied via SCP.

RayVentory Scan Engine copies the NDTRACK files via SCP to a temporary subdirectory of the home directory for the user associated with the username of the SSH credentials. Then it runs NDTRACK by issuing a command via SSH. Eventually, the resulting inventory file is copied to the RayVentory Scan Engine host. Later, the temporary directory is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch inventory on Linux / UNIX-like Operating Systems. This inventory method uses NDTRACK.

Technical Details

- **Used credential type:**
 - SSH
- **Required capabilities:**
 - Access to the File System
 - Remote Execution
 - SSH

- **Prerequisites:**

- None

Oracle Audit

This inventory method gathers files produced by the **Review Lite Script** (provided by Oracle to their customers) or similar scripts on Oracle database instances.

This method operates on targets of the type **Oracle**.

Zero-Touch Audit by SQLPLUS

Description

Zero-Touch Oracle audit (executing the 'review lite script') by an SQLPLUS remote session.

This is a tool for running the 'Review Lite Script' that customers receive via a local installation of SQL*plus.

Usage and Recommendation

Use this to help you run the 'Review Lite Script' that you received from the DB vendor on many DB installations at once.

Technical Details

- **Used credential type:**

- Oracle DB

- **Required capabilities:**

- Zero-touch

- **Prerequisites:**

- Path to SQL Plus executable has to be configured in the Oracle section of the **Settings** screen

Oracle Inventory

These inventory methods gather data on enabled options and usage on Oracle database instances.

These methods operate on targets of the type Oracle.

Zero-Touch Inventory by ORATRACK

Description

Zero-Touch Oracle remote inventory by ORATRACK.

Usage and Recommendation

Use this to generate an inventory of your database instances. This is the least invasive method to create such an inventory. The only requirements are credentials with sufficient privileges on the

target database for RayVentory Scan Engine.

Technical Details

- **Used credential type:**
 - Oracle DB
- **Required capabilities:**
 - Zero-touch
- **Prerequisites:**
 - The setting **Java executable path** has to be configured in the Oracle section of the **Settings** screen
 - Java has to be installed on the host machine
 - (Optional) The setting **Oracle Database Feature Usage Statistics script path** must be configured in the **Oracle** section of the **Settings** screen to run the DBFUS script together with the inventory

Remote-Execution by Service Manager / SMB Local Files on Windows

Description

Zero-Touch Oracle remote inventory by ORATRACK. Remote-Execution Oracle inventory for Windows hosts by a temporary local copy of ORATRACK pushed via SMB, executed by a temporary service, and inventory copied via SMB. RayVentory Scan Engine mounts the target's built-in share `ADMIN$`, copies at least one version of the ORATRACK Java executable and the encrypted queries file to a temporary subdirectory of `ADMIN$\Temp\`, and starts ORATRACK by one of the local Java runtimes via a temporary service. Eventually, the resulting inventory file is copied to the RayVentory Scan Engine host. Later, the temporary service and the temporary directory are deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed. This is the most invasive ORATRACK inventory method as ORATRACK is (temporarily) copied to the target and a temporary service is installed.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Access to the File System
 - Remote execution
 - Windows Service Manager
- **Prerequisites:**
 - None

Remote-Execution by SSH / SCP Local Files on Linux / Unix

Description

Remote-Execution Oracle inventory for Linux / Unix hosted by a temporary, local copy of ORATRACK, pushed via SCP, executed via SSH, and inventory copied via SCP. RayVentory Scan Engine copies at least one version of ORATRACK and the encrypted queries file via SCP to a temporary subdirectory of the home directory for the user associated with the username of the SSH credentials. Then it runs ORATRACK via a local Java runtime by issuing a command via SSH. Eventually, the resulting inventory file is copied to the RayVentory Scan Engine host. Later, the temporary directory is deleted.

Usage and Recommendation

Use this as an alternative for the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Oracle DB SSH
- **Required capabilities:**
 - Access to File System
 - Remote execution
 - SSH
- **Prerequisites:**
 - None

Remote-Execution by WMI / SMB Local Files on Windows

Description

Remote-Execution Oracle inventory for Windows hosts by a temporary, local copy of ORATRACK pushed via SMB, executed via WMI, and inventory copied via SMB. RayVentory Scan Engine mounts the target's built-in share `ADMIN$`, copies at least one version of the ORATRACK Java executable and the encrypted queries file to a temporary subdirectory of `ADMIN$\Temp\`, and starts ORATRACK by one of the local Java runtimes via a temporary service. Eventually, the resulting inventory file is copied to the RayVentory Scan Engine host. Later, the temporary directory is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Access to the File System
 - Remote execution
 - WMI
- **Prerequisites:**
 - None

Remote-Execution by Service Manager with HTTP(S) Upload on Windows

Description

Remote-Execution Oracle inventory for Windows hosts by ORATRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed by a temporary service, and inventory uploaded via HTTP(S). RayVentory Scan Engine starts a temporary service that references the service executable located on the RayVentory Scan Engine utilities share which starts ORATRACK by a Java runtime expected on the target located on the same share. ORATRACK will upload its results to the RayVentory Scan Engine HTTP Service. Later, the temporary service is deleted

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Remote execution
 - Windows Service Manager
- **Prerequisites:**
 - Configured and reachable RayVentory Scan Engine HTTP Service

Remote-Execution by Service Manager Upload SMB on Windows

Description

Remote-Execution Oracle inventory for Windows hosts by ORATRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed by a temporary service and inventory uploaded via SMB to the UNC path for RayVentory Scan Engine inventories.

RayVentory Scan Engine starts a temporary service that references the service executable located on the RayVentory Scan Engine utilities share which starts ORATRACK, located on the same share by a Java runtime expected on the target. ORATRACK will upload its results to the

RayVentory Scan Engine inventories share. Later, the temporary service is deleted.

Usage and Recommendation

Use this as an alternative to the zero-touch ORATRACK inventory. You may need this if remote connections to the DB are blocked by the host firewall and local connections are allowed.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Remote execution
 - Upload to the SMB shares
 - Windows Service Manager
- **Prerequisites:**
 - SMB share to receive inventory files
 - The setting **Save inventory results from target devices on the following UNC share** must be configured in the **Remote Execution** section of the **Settings** screen

Remote-Execution by WMI Upload HTTP(S) on Windows

Description

Remote-Execution Oracle inventory for Windows hosts by ORATRACK, loaded via SMB from the UNC path to the RVP utilities, and executed via WMI and inventory uploaded via HTTP(S).

RayVentory Scan Engine starts ORATRACK, located on the RayVentory Scan Engine utilities share by a Java runtime expected on the target via WMI (`Win32_Process`). ORATRACK will upload its results to the RayVentory Scan Engine HTTP Service.

Usage and Recommendation

This is the second least invasive method together with **Remote-Execution by WMI upload SMB on Windows** to execute an Oracle inventory by ORATRACK. No files are copied to the target, no temporary service is installed. RayVentory Scan Engine will just create a process that starts a local Java runtime to run ORATRACK.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Remote execution
 - WMI
- **Prerequisites:**
 - The setting **Execute RayVentory Scan Engine Utilities from the following UNC shared path on target device** must be configured in the **Remote Execution** section of the **Settings** screen

- The scan utilities must be present on the shared location (for example installed via the **Install Scan Utilities** button in the Remote Execution section of the **Settings** screen)

Remote-Execution by WMI Upload SMB on Windows

Description

Remote-Execution Oracle inventory for Windows hosts by ORATRACK, loaded via SMB from the UNC path to the RayVentory Scan Engine utilities, executed via WMI, and inventory uploaded via SMB to the UNC path for RayVentory Scan Engine inventories.

RayVentory Scan Engine starts ORATRACK, located on the RayVentory Scan Engine utilities share by a Java runtime expected on the target via WMI (`win32_Process`). ORATRACK will upload its results to the RayVentory Scan Engine inventories share.

Usage and Recommendation

This is the second least invasive method together with **Remote-Execution by WMI upload HTTP(S) on Windows** to execute an Oracle inventory by ORATRACK. No files are copied to the target, no temporary service is installed. RayVentory Scan Engine will just create a process that starts a local Java runtime to run ORATRACK.

Technical Details

- **Used credential type:**
 - Oracle DB Windows
- **Required capabilities:**
 - Remote execution
 - Upload to the SMB shares
 - WMI
- **Prerequisites:**
 - SMB share to receive inventory files
 - The setting **Execute RayVentory Scan Engine Utilities from the following UNC shared path on target device** must be configured in the **Remote Execution** section of the **Settings** screen
 - The scan utilities must be present on the shared location (for example installed via the **Install Scan Utilities** button in the **Remote Execution** section of the **Settings** screen)
 - The setting **Save inventory results from target devices on the following UNC share** must be configured in the **Remote Execution** section of the **Settings** screen

SNMP Inventory

This inventory method gathers basic data on network devices that support the SNMP protocol. This is intended for devices like printers, UPSs, and network equipment that do not expose Operating Systems or a standardized interfaces besides SNMP. This method operates on targets of the type **SNMP**.

Zero-Touch by SNMP

Description

Zero-Touch SNMP based inventory by the SNMP Tracker for SNMP enabled devices, supporting SNMP versions 1, 2/2c, and 3. This is intended for network devices like printers and UPSs.

Usage and Recommendation

Use this to gather basic configuration and inventory data from SNMP enabled network devices. We currently support retrieving data on model, manufacturer, and serial number from a range of different printers, switches, routers, and UPSs. This inventory method can be customized to gather vendor or device specific data.

Technical Details

- **Used credential type:**
 - SNMP
- **Required capabilities:**
 - Zero-touch
- **Prerequisites:**
 - None

ESX / vSphere Inventory

This inventory method gathers data on virtual guests and host configurations from a VMware vSphere instance or single VMware ESX hosts. RayVentory Scan Engine has experimental support for VMware Workstation instances as well, but their data is not taken into account in all of the related reports.

This method operates on targets of type ESX / vSphere.

Zero-Touch Inventory by VMware API via HTTP(S)

Description

Zero-Touch VSphere / ESX inventory using the VMware Management API via HTTP(S).

Technical details

- **Used credential type:**
 - ESX / vSphere
- **Required capabilities:**
 - Zero-touch
- **Prerequisites:**
 - None

Application of Credentials

RayVentory Scan Engine differentiates between five different credential types:

- Windows
- SSH
- Oracle DB
- SNMP
- vSphere

Windows and SSH Credentials

The types Windows and SSH are used for OS / Device inventory and during Discovery for discovering / probing Oracle databases. The remote execution based inventory methods for Oracle use these credentials too, in order to find evidence for Oracle instances in the host file system and services and for authentication to the host system.

Oracle DB Credentials

The type Oracle DB is used for authentication against Oracle databases for creating the Oracle inventory.

SNMP Credentials

The type SNMP is used for selecting an SNMP community or authentication by SNMP v3.

vSphere Credentials

The type vSphere is used for authentication against the VMware management API for the creation of inventories of vSphere / ESX virtualization hosts.

In general, the application of credentials to a target system / service happens in the order they appear in the credentials list. If certain credentials have been set for a device / service then these are tried first, before other credentials are tried. On a successful inventory, RayVentory Scan Engine remembers which credentials have been used for a device or service and tries these first the next time an inventory is run.

Credentials may be restricted to be applied to certain devices. To restrict credentials to a specific hostname enter that hostname or IP address into the field **Target name pattern**. This field supports Regular Expressions.



WARNING

Whilst the idea of creating a list of credentials to try on each host may sound handy (especially if you are not certain which credentials actually belong to a certain device / service), keep in mind that such brute-forcing of credentials may trigger alarms in your security and network monitoring solution. In case of an Oracle database you may even lock a user as by default, as three failed authentication attempts in a row will lock the user / login that has been used. It is highly recommended to use as few generic credentials as possible, by either selecting concrete credentials for devices or setting up filters to limit their application.

Custom Windows Scans

Zero-Touch Inventory Scan for Windows can be customized by providing a specially prepared .xml containing instructions how to query the target device via WMI, as well as file system and the Windows registry.

To Create a Custom Scan Definition...

1. Go to the installation directory of RayVentory Scan Engine.
2. Start the executable file `RemoteWmiInventory.exe` with a single argument `example`.
3. Save the created file as scan template (by default it will be saved in your current working directory under the name `example.xml`).
4. Customize the file to include custom scanned content.



Be aware:

The generated content does not reflect the default settings of Windows Inventory scans.

After the file is created and customized, you can point to it by configuring the path under **Settings > Inventory > Windows > Custom configuration**.



WARNING

This is a critical piece of the Windows Inventory functionality which can be easily broken by incomplete or incorrect configurations. We recommend to have it configured by one of our RayVentory Scan Engine consultants. Failing to correctly customize the Windows scan may result in broken scans or a huge load on the target systems and the networks.

Sample Configuration File

The following shows a typical content of a custom scan definition:

```
<?xml version="1.0" encoding="utf-8"?>
<QueryFile xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance">
  <Queries Mandatory="true" IsSoftware="false" WmiClass="Win32_ComputerSystem"
Namespace="\root\cimv2" Name="Name">
    <Fields WmiPropertyName="Manufacturer" />
    <Fields WmiPropertyName="Model" />
    <Fields WmiPropertyName="Domain" />
    <Fields WmiPropertyName="DomainRole" />
    <Fields WmiPropertyName="NumberOfProcessors" />
    <Fields WmiPropertyName="NumberOfLogicalProcessors" />
    <Fields WmiPropertyName="TotalPhysicalMemory" />
```

```
<Fields WmiPropertyName="Status" />
<Fields WmiPropertyName="UserName" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_ComputerSystemProduct"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="IdentifyingNumber" />
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="UUID" />
  <Fields WmiPropertyName="Vendor" />
  <Fields WmiPropertyName="Version" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_OperatingSystem"
Namespace="\root\cimv2" Name="Caption">
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="Manufacturer" />
  <Fields WmiPropertyName="Version" />
  <Fields WmiPropertyName="ServicePackMajorVersion" />
  <Fields WmiPropertyName="ServicePackMinorVersion" />
  <Fields WmiPropertyName="SerialNumber" />
  <Fields WmiPropertyName="InstallDate" />
  <Fields WmiPropertyName="LastBootUpTime" />
  <Fields WmiPropertyName="OSLanguage" />
  <Fields WmiPropertyName="FreePhysicalMemory" />
  <Fields WmiPropertyName="FreeVirtualMemory" />
  <Fields WmiPropertyName="CountryCode" />
  <Fields WmiPropertyName="WindowsDirectory" />
  <Fields WmiPropertyName="SystemDirectory" />
  <Fields WmiPropertyName="Caption" />
  <Fields WmiPropertyName="CSDVersion" />
  <Fields WmiPropertyName="Status" />
  <Fields WmiPropertyName="CSName" />
  <Fields WmiPropertyName="OSType" />
  <Fields WmiPropertyName="OSArchitecture" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_BIOS" Namespace="\root
\cimv2" Name="Name">
  <Fields WmiPropertyName="Manufacturer" />
  <Fields WmiPropertyName="Version" />
  <Fields WmiPropertyName="ReleaseDate" />
  <Fields WmiPropertyName="SerialNumber" />
  <Fields WmiPropertyName="BiosCharacteristics" />
  <Fields WmiPropertyName="Status" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_Processor"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="Manufacturer" />
  <Fields WmiPropertyName="Version" />
  <Fields WmiPropertyName="ProcessorId" />
  <Fields WmiPropertyName="CurrentClockSpeed" />
  <Fields WmiPropertyName="CurrentVoltage" />
  <Fields WmiPropertyName="L2CacheSize" />
  <Fields WmiPropertyName="Status" />
  <Fields WmiPropertyName="MaxClockSpeed" />
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="ProcessorType" />
  <Fields WmiPropertyName="NumberOfLogicalProcessors" />
  <Fields WmiPropertyName="NumberOfCores" />
  <Fields WmiPropertyName="DeviceID" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_DiskDrive"
Namespace="\root\cimv2" Name="Name">
```

```
<Fields WmiPropertyName="Description" />
<Fields WmiPropertyName="InterfaceType" />
<Fields WmiPropertyName="Manufacturer" />
<Fields WmiPropertyName="Model" />
<Fields WmiPropertyName="Partitions" />
<Fields WmiPropertyName="Size" />
<Fields WmiPropertyName="Status" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_LogicalDisk"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="VolumeName" />
  <Fields WmiPropertyName="FileSystem" />
  <Fields WmiPropertyName="FreeSpace" />
  <Fields WmiPropertyName="Size" />
  <Fields WmiPropertyName="VolumeSerialNumber" />
  <Fields WmiPropertyName="DriveType" />
  <Fields WmiPropertyName="MediaType" />
  <Fields WmiPropertyName="Status" />
  <Fields WmiPropertyName="ProviderName" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_CDROMDrive"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="Manufacturer" />
  <Fields WmiPropertyName="Drive" />
  <Fields WmiPropertyName="Status" />
  <Fields WmiPropertyName="Capabilities" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_NetworkAdapter"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Manufacturer" />
  <Fields WmiPropertyName="MACAddress" />
  <Fields WmiPropertyName="MaxSpeed" />
  <Fields WmiPropertyName="Speed" />
  <Fields WmiPropertyName="Status" />
</Queries>
<Queries Mandatory="false" IsSoftware="false"
WmiClass="Win32_NetworkAdapterConfiguration" Namespace="\root\cimv2" Name="Caption">
  <Fields WmiPropertyName="Caption" />
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="Index" />
  <Fields WmiPropertyName="MACAddress" />
  <Fields WmiPropertyName="IPEnabled" />
  <Fields WmiPropertyName="DHCPEnabled" />
  <Fields WmiPropertyName="IPAddress" />
  <Fields WmiPropertyName="DHCP Server" />
  <Fields WmiPropertyName="DNSHostName" />
  <Fields WmiPropertyName="DNSDomain" />
  <Fields WmiPropertyName="DNSServerSearchOrder" />
  <Fields WmiPropertyName="DefaultIPGateway" />
  <Fields WmiPropertyName="IPSubnet" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_PhysicalMemory"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Capacity" />
  <Fields WmiPropertyName="MemoryType" />
  <Fields WmiPropertyName="PositionInRow" />
  <Fields WmiPropertyName="Speed" />
  <Fields WmiPropertyName="Status" />
</Queries>
```

```
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_SoundDevice"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="Manufacturer" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_VideoController"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="VideoProcessor" />
  <Fields WmiPropertyName="DriverVersion" />
  <Fields WmiPropertyName="DriverDate" />
  <Fields WmiPropertyName="InstalledDisplayDrivers" />
  <Fields WmiPropertyName="AdapterRAM" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_VideoConfiguration"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="AdapterRAM" />
  <Fields WmiPropertyName="AdapterType" />
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="HorizontalResolution" />
  <Fields WmiPropertyName="MonitorManufacturer" />
  <Fields WmiPropertyName="MonitorType" />
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="VerticalResolution" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_SystemEnclosure"
Namespace="\root\cimv2" Name="Name" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="SoftwareLicensingProduct"
Namespace="\root\cimv2" Name="Name">
  <Fields WmiPropertyName="ApplicationID" />
  <Fields WmiPropertyName="Description" />
  <Fields WmiPropertyName="EvaluationEndDate" />
  <Fields WmiPropertyName="GracePeriodRemaining" />
  <Fields WmiPropertyName="LicenseStatus" />
  <Fields WmiPropertyName="MachineURL" />
  <Fields WmiPropertyName="Name" />
  <Fields WmiPropertyName="OfflineInstallationId" />
  <Fields WmiPropertyName="PartialProductKey" />
  <Fields WmiPropertyName="ProcessorURL" />
  <Fields WmiPropertyName="ProductKeyID" />
  <Fields WmiPropertyName="ProductKeyURL" />
  <Fields WmiPropertyName="UseLicenseURL" />
</Queries>
<Queries Mandatory="false" IsSoftware="false" WmiClass="SoftwareLicensingService"
Namespace="\root\cimv2" Name="KeyManagementServiceProductKeyID">
  <Fields WmiPropertyName="ClientMachineID" />
  <Fields WmiPropertyName="IsKeyManagementServiceMachine" />
  <Fields WmiPropertyName="KeyManagementServiceCurrentCount" />
  <Fields WmiPropertyName="KeyManagementServiceMachine" />
  <Fields WmiPropertyName="KeyManagementServiceProductKeyID" />
  <Fields WmiPropertyName="PolicyCacheRefreshRequired" />
  <Fields WmiPropertyName="RequiredClientCount" />
  <Fields WmiPropertyName="Version" />
  <Fields WmiPropertyName="VLAActivationInterval" />
  <Fields WmiPropertyName="VLRenewalInterval" />
</Queries>
<Queries Mandatory="false" IsSoftware="true" WmiClass="Win32_Product"
Namespace="\root\cimv2" Name="Name" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="MGS_MSSQL2000"
Namespace="\root\cimv2" Name="Name" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="MGS_MSSQL20058"
Namespace="\root\cimv2" Name="Name" />
```

```
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ComputerSystem"
Namespace="\root\virtualization" Name="Name" Evidence="virtualization" />
<Queries Mandatory="false" IsSoftware="false"
WmiClass="Msvm_VirtualSystemSettingData" Namespace="\root\virtualization" Name="Name"
Evidence="virtualization" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_MemorySettingData"
Namespace="\root\virtualization" Name="Name" Evidence="virtualization" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ProcessorSettingData"
Namespace="\root\virtualization" Name="Name" Evidence="virtualization" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_KvpExchangeComponent"
Namespace="\root\virtualization" Name="Name" Evidence="virtualization" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ComputerSystem"
Namespace="\root\virtualization\v1" Name="Name" Evidence="virtualization\v1" />
<Queries Mandatory="false" IsSoftware="false"
WmiClass="Msvm_VirtualSystemSettingData" Namespace="\root\virtualization\v1"
Name="Name" Evidence="virtualization\v1" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_MemorySettingData"
Namespace="\root\virtualization\v1" Name="Name" Evidence="virtualization\v1" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ProcessorSettingData"
Namespace="\root\virtualization\v1" Name="Name" Evidence="virtualization\v1" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_KvpExchangeComponent"
Namespace="\root\virtualization\v1" Name="Name" Evidence="virtualization\v1" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ComputerSystem"
Namespace="\root\virtualization\v2" Name="Name" Evidence="virtualization\v2" />
<Queries Mandatory="false" IsSoftware="false"
WmiClass="Msvm_VirtualSystemSettingData" Namespace="\root\virtualization\v2"
Name="Name" Evidence="virtualization\v2" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_MemorySettingData"
Namespace="\root\virtualization\v2" Name="Name" Evidence="virtualization\v2" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_ProcessorSettingData"
Namespace="\root\virtualization\v2" Name="Name" Evidence="virtualization\v2" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Msvm_KvpExchangeComponent"
Namespace="\root\virtualization\v2" Name="Name" Evidence="virtualization\v2" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_serverFeature"
Namespace="\root\cimv2" Name="Name" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="MSCluster_Cluster"
Namespace="\root\MSCluster" Name="Name" />
<Queries Mandatory="false" IsSoftware="false" WmiClass="Win32_Service"
Namespace="\root\cimv2" Name="Name" />
<Keys Namespace="\root\cimv2" Hive="HKEY_LOCAL_MACHINE" Path="SOFTWARE\Microsoft
\Windows\CurrentVersion\Uninstall" View="64" />
<Keys Namespace="\root\cimv2" Hive="HKEY_LOCAL_MACHINE" Path="SOFTWARE\Wow6432Node
\Microsoft\Windows\CurrentVersion\Uninstall" View="32" />
<Keys Namespace="\root\cimv2" Hive="HKEY_LOCAL_MACHINE" Path="SOFTWARE\Microsoft
\Virtual Machine\Guest\Parameters" View="32">
  <Values>
    <RegistryValue Name="PhysicalHostNameFullyQualified" Type="string" />
    <RegistryValue Name="VirtualMachineId" Type="string" />
    <RegistryValue Name="VirtualMachineName" Type="string" />
  </Values>
</Keys>
<Keys Namespace="\root\cimv2" Hive="HKEY_LOCAL_MACHINE" Path="SOFTWARE\Microsoft
\Virtual Machine\Guest\Parameters" View="64">
  <Values>
    <RegistryValue Name="PhysicalHostNameFullyQualified" Type="string" />
    <RegistryValue Name="VirtualMachineId" Type="string" />
    <RegistryValue Name="VirtualMachineName" Type="string" />
  </Values>
</Keys>
<Files GetContent="true" SearchSubdirs="false" Path="temp\*" DriveLetter="c"
FileName="*.bat" />
</QueryFile>
```

Using Remote Execution Plugins

RayVentry Scan Engine supports advanced Remote-Execution scanning of Windows devices with the help of custom plugins. The following checklist summarizes the actions required to set up custom plugins:

1. Ensure that a valid UNC path is configured in the **Settings > Inventory > Remote Execution** screen. Press **Install scan utilities** to install the tools for the first time.
2. Put the custom plugins into the `/plugins` subfolder, which should be located in the root UNC path configured in the first step.
3. Make sure your devices are scanned with Remote-Execution scanning methods (by default, Zero-Touch methods take precedence). This can be achieved in one of the following ways:
 - a. Disable Zero-Touch support for particular devices that you want to reach with plugin-based Remote-Execution scan.
 - b. Disable Zero-Touch globally (see Inventory Methods chapter for more information on that).
4. Scan the affected Windows devices or rescan already existing devices.
5. Additional information and logs documenting the execution of a plugin can be found on the target devices in the `Tracker.log` file. The level of details can be changed in the RayVentry Scan Engine settings.

Commandline Tools

Certain components of RayVentory Scan Engine can be operated using the command line. These components are:

- **RIW:** Remote Inventory for Windows
- **RIU:** Remote Inventory for Unix
- **NDTRACK:** OS Inventory Tracker
- **ORATRACK:** Oracle Tracker
- **VMTRACK:** RayVentory VM Console
- **SNMPTRACKER:** SNMP Tracker

A RayVentory consultant may use the binaries of the RayVentory Scan Engine components to create scripts and scheduled tasks for a RayVentory implementation. With the exception of **NDTRACK**, all these command line tools have got a built-in help that checks command line arguments and shows a list of command line arguments with their descriptions. If illegal arguments or the argument help, -help, --help, or /? are being entered, then the programs show their respective usage example and argument list.

The command line tools do not make use of the settings and credentials from RayVentory Scan Engine.

Remote Inventory for Windows (RIW)

Use the program to run a remote OS inventory on Windows targets. See the online help of the program for arguments that allow for customization. The program may use credentials passed to it via the command line or the current session login for authentication against the target hosts.

Remote Inventory for Windows (`RemoteWmiInventory.exe`) is a command line oriented tool for generating inventories from remote machines running Windows with WMI or at least SMB and DCOM capability.

RIW uses Windows login, authentication, and impersonation to connect to the target machines in order to perform WMI queries, WMI method invocation, and registry scans. Further it uses SMB for file scans.

The inventory is customizable via a file (called control file). It allows you to perform WMI queries, WMI method invocation, registry, and file system scans.

The command line arguments are passed to RIW as name-value pairs. Flags are set by their name.

Usage:

```
RemoteWmiInventory.exe [<argument-name>=<value>|<flag-name>]...
```

Table of Arguments and Flags

| Name | Type | Description |
|------------------------|---------|---|
| batch | TEXT | A list of hosts with optional credentials. Format: <host> [<user> <password>] <whitespace> #<line-comment> [[<CR>]<LF>...] |
| classesFile | TEXT | WMI, registry and file queries as XML file. |
| conFailureLog | TEXT | Enables output of the failure log to the file specified. |
| conSuccessLog | TEXT | Enables output of the success log to the file specified. |
| DeviceID | TEXT | RMS / RV network device ID for inventory binding. |
| forceW32registryAccess | FLAG | Force access to the registry by W32 API instead of WMI. |
| help | FLAG | Shows a usage hint and lists the command line options. Ignores all other arguments except <code>pause</code> and <code>quits</code> . |
| host | TEXT | Host to scan. |
| job | TEXT | Job tag / ID (this is supposed to be used by Raynet's inventory / discovery agents, only). |
| outputpath | TEXT | Path for output Excludes: upload |
| pass | TEXT | Password |
| pause | FLAG | Pauses before exit and waits for a single key press. |
| scanTimeout | INTEGER | Defaults to 1200: Timeout in seconds for scanning a single target. Set this to 0 in order to disable the timeout. |
| talkingFileNames | FLAG | Use certain bits from the inventory for the inventory file name instead of a generic number. |
| testDelay | INTEGER | Delay between connection tests (3 connection attempts) in milliseconds. |
| testTimeout | INTEGER | Connection test timeout in milliseconds, set to 0 to disable the connection test. |
| upload | TEXT | Address for upload Excludes: <code>outputpath</code> . |
| user | TEXT | Username |
| verbose | FLAG | Log the remote commands. |
| workers | INTEGER | Defaults to 10: Maximum worker threads running in parallel. |

Privileges

The user that operates RIW needs privileges to run WMI queries. Some queries and require elevated privileges as a member of the local or domain administrator group has got. Also, file scans require such permissions to access the administrative shares which allow access to the file systems of the target machines.



Tip:

In case of Windows XP, the remote Win32 API calls are the only option to get inventory relevant information from the target machines registry.

Ports in Use

- 135 for running WMI queries or using Win32 API calls as an alternative method for gathering inventory relevant data from the registry.
- 445 for running software inventory relevant file scans
- 80(443) for HTTP(S) upload to the data sink or 445 for SMB upload to the data sink

Remote Inventory for Unix / Linux (RIU)

Use the program to run a remote OS inventory on Linux and unix-like target platforms. See the online help of the program for arguments that allow customization. The program may use credentials passed to it via the command line for authentication against the target hosts. This program comes with a built-in credential store, a target database, and job management.

Remote Inventory for Unix / Linux (`RemoteInventory4Unix.exe`) is a command line oriented tool for generating inventories from remote machines running different unixoid platforms including different flavors of Linux.

The RIU tool can be operated in two modes:

- **Batch mode**

Allows you to manually create job files, target hosts, or automatically create job files from the content of target files while specifying simple credential store files with optional encryption. During startup RIU looks for job files, checks their status, and continues them until all job items (target machines) reached one of the end-states (Success, AuthenticationFailure).

- **Single mode**

Allows you to specify a single host as a target machine

RIU generates inventories by connection to a target machine via SSH and performing different built-in shell commands and system utilities to determine the platform and architecture characteristics. According to those characteristics, it gathers hard- and software inventory relevant data by running further commands and by parsing and evaluating their results to store them in an inventory file for each target machine. Then the resulting NDI file is uploaded to an inventory data sink like RayVentory Scan Engine, RayVentory Server, and RayManageSoft or it is stored in the local file system.

The command line arguments are passed to RIU as name-value pairs. Flags are set by their name.

Usage

RemoteInventory4Unix.exe [<argument-name>=<value>|<flag-name>]...

Table of Arguments and Flags

| Name | Type | Description |
|-------------------|------|--|
| addhosts | TEXT | Add hosts from a .csv file to the targets. Each line of the file must be in the format: <code>hostname[,port][,hostkey]</code> Mind that port and hostkey are optional. |
| authKey | TEXT | Fallback authentication key in the OpenSSH format. This is used if no authentication key is given for a target. |
| authKey.pass | TEXT | Passphrase for the fallback authentication key. |
| authKeyFile | TEXT | Fallback authentication key file in the OpenSSH format. |
| conFailureAuthLog | TEXT | Enables output of the authentication failure log to the file specified. |
| conFailureLog | TEXT | Enables output of the connection failure log to the file specified. |
| config | TEXT | Path to the configuration file. The optional configuration file is an XML file root element <code>Config</code> and sets the values <code>ConsoleRefreshInterval</code> , <code>UploadAddress</code> , and <code>Workers</code> which equal the command line arguments <code>refresh</code> , <code>upload</code> , and <code>workers</code> . |
| conSuccessLog | TEXT | Enables output of the connection success log to the file specified. |
| credentials.file | TEXT | Defaults to <code>credentials.xml</code> : Credentials store file. |
| credentials.key | TEXT | Encryption key for the passwords in the credentials store file. |
| decryptLog | TEXT | Decrypting the partially encrpyted log file, given, and exit. |
| decryptLog.key | TEXT | The key for decrypting the file given by the argument <code>decryptLog</code> . |
| DeviceID | TEXT | RMS / RV network device ID for inventory binding. |
| forceFileName | TEXT | Forces the use of supplied file name for the inventory file. |
| help | FLAG | Shows a usage hint and lists the command line options. Ignores all other arguments except <code>pause</code> and <code>quits</code> . |
| host | TEXT | Host to scan. |
| host.add | FLAG | Adds to the store and exit. Requires: host |
| host.key | TEXT | Expected host key fingerprint for host authentication. Requires: host |
| job | TEXT | Job tag / ID (this is supposed to be used by Raynet's inventory / |

| | | |
|--------------------|---------|--|
| | | discovery agents, only). |
| jobs | TEXT | Path to directory for job files. |
| mssqlts.connection | TEXT | Connection string to a MSSQL DB as source for targets Requires: mssqlts.query. |
| mssqlts.query | TEXT | Query string for a MSSQL DB as source for targets. Requires: mssqlts.connection |
| outputpath | TEXT | Path for output. Excludes: upload |
| pass | TEXT | Fallback password. |
| pass.key | TEXT | Encryption key for fallback password. |
| pause | FLAG | Pauses before exit and waits for a single key press. |
| port | INTEGER | defaults to 22: SSH port on the target host. |
| recordHostKeys | FLAG | Add the host-keys that were reported by the hosts for hosts which do not have got a host-key, yet. |
| refresh | INTEGER | Defaults to 5000: Interval for refresh of console and saving of job state in milliseconds. |
| rerunAllJobs | FLAG | Retry all jobs no matter what their status is. Excludes: retryAllJobs, single, retryAuthFailure. |
| retryAllJobs | FLAG | Retry all jobs no matter what their status is. Excludes: single |
| retryAuthFailure | FLAG | Retry targets in jobs which failed due to authentication errors. Excludes: single. |
| scanTimeout | INTEGER | Timeout in seconds for scanning a single target. Set this this to 0 in order to disable the timeout. |
| single | FLAG | Run for a single host given on command line, ignoring all job files. |
| successfulHosts | TEXT | During job creation, set the targets in the specified list to status Done. Excludes: single |
| talkingFileNames | FLAG | Use certain bits from the inventory for the inventory file name instead of a generic number. |
| targetsfile | TEXT | Path to a file with targets. |
| upload | TEXT | Address for upload. Excludes: outputpath |
| user | TEXT | Fallback username. |

| | | |
|-----------------------------|---------|---|
| <code>user.add</code> | FLAG | Add the fallback user with password or authentication key and authentication key passphrase to the credentials store and exit. Requires: <code>user</code> |
| <code>user.elevpass</code> | TEXT | The password that is passed to <code>sudo</code> . Requires: <code>user</code> |
| <code>user.sudo</code> | FLAG | Indicates that the <code>sudo</code> command is enabled for the fallback user. Requires: <code>user</code> |
| <code>user.super</code> | FLAG | Indicates that the fallback user is a super user Requires: <code>user</code> |
| <code>user.target</code> | TEXT | Regular expression to match the targets (by hostname) that the fallback credentials apply to. Requires: <code>user</code> |
| <code>useTargetCache</code> | FLAG | Loads targets not from the given sources but from a cache when present or creates the cache. |
| <code>verbose</code> | FLAG | Log the remote commands. |
| <code>workers</code> | INTEGER | Maximum number of workers to spawn. |

Privileges

What privileges are needed depends on what data is needed.

You will need a user with root privileges or that is allowed to `sudo`, `prbrun`, or `priv` to run programs with elevated privileges with or without a keyboard interactive password query.

For example: Without such privileges you will not be able to read inventory and dependency mapping relevant data as for example the BIOS serial number of a VM (to determine which virtual guest system reflects the target machine) or a system enclosure serial number.

Ports in Use

- 22 for connecting to the target machine.
- 80(443) for HTTP(S) upload to the data sink or 445 for SMB upload to the data sink.

VMware Inventory

RV-VM-Console (`RayVentory.VM.Console.exe`) is a command line oriented tool for generating inventories from vSphere / ESX hosts using VMware's restful management API. You may specify a single target on the command line. If no target has been specified on the command line then it will target all vSphere / ESX hosts that RVP knows.

If the the target is a vSphere cluster then VM-Console will create an NDI file for each ESX host in the cluster.

The command line arguments are passed to RIW as name-value pairs. Flags are set by their name.

Usage

RayVentory.VM.Console.exe [-o <argument-name>=<value>]...

Table of Arguments and Flags

| Name | Type | Description |
|--------------------|---------|---|
| auth | BOOLEAN | Its entered connection data will be saved. |
| basepath | TEXT | Path to store configuration files and results unless <code>vsphereoutput</code> was specified. |
| DeviceID | TEXT | RMS / RV network device ID for inventory binding. |
| encryptionKey | TEXT | Key for encryption and decryption of credentials. |
| help | FLAG | RMS / RV network device ID for inventory binding. |
| ignoreSSL | BOOLEAN | Ignore SSL / TLS connection warnings. |
| job | TEXT | Job tag / ID (this is supposed to be used by Raynet's inventory / discovery agents, only). |
| password | TEXT | Password of the user for authentication. |
| upload | TEXT | URL for uploading the result. |
| url | TEXT | The targets seb service URL. Usually it has the form of <code>https://myTargetHost/sdk</code> . |
| user | TEXT | Name of the user for authentication. |
| vsphereconnections | TEXT | Filename for the vSphere connections file. |
| vsphereoutput | TEXT | Path for the output of the vSphere inventory. |

Privileges

The RV VM Console needs a user with read privileges on the whole object tree of the vSphere infrastructure or ESX host. Depending on the permissions set, the user may need special permissions to extract sensitive details like the serial number / license key.

Ports in Use

- 80(443) for communication with VMware's management API via HTTP(S) and for HTTP(S) upload to the data sink.

ORATRACK

ORATRACK (`oratrack.jar`) is a command line oriented tool for generating inventories for Oracle database instances. You may target a single database by command line arguments, target many databases by a connections file, or let it discover and inventory local and remote databases on its own.

The command line arguments are passed to ORATRACK as name-value pairs. Flags are set by their name.

Usage

```
java -jar oratrack.jar [-o <argument-name>=<value>]...
```

Table of Arguments and Flags

| Name | Type | Description |
|--------------------------------|---------|---|
| <code>asSysDb</code> | BOOLEAN | <p>Connect as SYSDBA.</p> <p><code>true</code> The connection is established with SYS role.</p> <p><code>false</code> The connection is established without giving the SYS role.</p> |
| <code>auth</code> | BOOLEAN | <p>Specifies whether the set of targets collected is written to <code>OracleConnections.xml</code>. This argument defaults to <code>true</code>.</p> <p><code>true</code> Write to <code>OracleConnections.xml</code>.</p> <p><code>false</code> Do not write.</p> |
| <code>authPath</code> | TEXT | <p>Specify the targets filename. By default a the current working directory is searched for a file called <code>OracleConnections.xml</code>. This file will be created when it is not present and argument <code>auth</code> is set to <code>true</code>.</p> |
| <code>auto</code> | OPTION | <p>Turns on discovery of local configuration files in order to discover targets known to the local machine.</p> <p><code>local</code> Ignore remote targets.</p> <p><code>all</code> Use all targets found.</p> |
| <code>checkCertificates</code> | BOOLEAN | <p>Enable / disable checking the host certificate when uploading using HTTPS. Enabled by default.</p> |
| <code>csv</code> | BOOLEAN | <p>Set output mode.</p> <p><code>true</code> Write results as CSV files.</p> <p><code>false</code> Write results as NDI file.</p> |
| <code>dbhost</code> | TEXT | <p>Specifies a target address or a pattern to match target addresses, not specified by command line.</p> |

| Name | Type | Description |
|------------------|---------|---|
| | | <p><hostname> A concrete hostname.</p> <p>localhost Special hostname, indicates using the local machine.</p> <p>127.0.0.1 Special IP address, indicates using the local machine.</p> <p><ip-address> A concrete IP address.</p> <p><pattern> Use targets with host addresses that match this expression, only.</p> |
| debug | BOOLEAN | <p>Controls the log level.</p> <p>true Log debug messages.</p> <p>false Suppress debug messages.</p> |
| deviceId | INTEGER | Sets the device ID. This is supposed to be used by Raynet's inventory / discovery agents, only. |
| discoverySource | OPTION | <p>This argument is used in conjunction with argument <code>auto</code> and controls where to look for the home directory of a database.</p> <p>The following values for the argument <code>discoverySource</code> may be may be combined in a comma seperated list. For example: <code>discoverySource=oratab,files</code></p> <p>environment Scan the environment variables.</p> <p>registry Scan the windows registry or <code>oratab</code> file.</p> <p>oratab Look for and process the <code>oratab</code> file.</p> <p>files Scan scan the filesystem for configuration files.</p> |
| domain | TEXT | Gives a value to put in the inventory file as domain. |
| encryption | OPTION | Specifies the encryption algorithm(s) to use as comma separated list of algorithms enclosed in brackets when specifying a target by command line. Mind that different versions of ORATRACK support different sets of encryption algorithms. For example: RC4_40, AES128, AES192, 3DES112 etc. |
| encryption.level | OPTION | Specifies the encryption level, controlling the negotiation between client and server together with the parameter <code>encryption</code> . Valid values are <code>REQUIRED</code> , <code>REQUESTED</code> , <code>REJECTED</code> , or <code>ACCEPTED</code> . |
| encryptionKey | TEXT | Sets the encryption key that is used for encrypting and decrypting all credentials. |
| dfusScript | TEXT | Sets the optional Oracle DFUS script filename for |

| Name | Type | Description |
|--------------------------|---------|--|
| | | running it by sqlplus. |
| | | If you set this argument and the file is not present, then this will cause the query to fail with an error. |
| | | If you do not set this argument then oratrack looks for a file called <code>oracle.sql</code> and runs it. |
| failedconnectionbehavior | OPTION | <p>Sets the condition for reporting a failed connection by exit code.</p> <p>ignore Ignore failed connections.</p> <p>all Returns an error if all connections failed.</p> <p>any Returns an error if any connection fails.</p> |
| help | FLAG | Show the usage hint and command line argument list. |
| ignorenames | BOOLEAN | Ignore all service names or SIDs matching the semi-colon-separated regular expressions listed, here, during discovery. Default is <code>CLRExtProc;EXTPROC;PLSExtProc</code> . |
| integrity | OPTION | Specifies the integrity algorithm(s) to use as comma separated list of algorithms enclosed in brackets when specifying a target by command line. Currently supported: SHA1 or MD5. |
| integrity.level | OPTION | Specifies the integrity level controlling the negotiation between client and server together with the parameter integrity. Valid values are <code>REQUIRED</code> , <code>REQUESTED</code> , <code>REJECTED</code> , or <code>ACCEPTED</code> . |
| job | TEXT | Sets the job tag / ID. This is supposed to be used by Raynet's inventory / discovery agents only. |
| listenerControl | BOOLEAN | Enable or disable querying the listener control status during discovery. Enabled by default. |
| mode | OPTION | Sets the program mode. This argument defaults to query. |
| | | <p>encryptquery Read <code>query.xml</code> from the current working directory and write an encrypted copy named <code>query.xml.enc</code> to the current working directory.</p> <p>test Tries to connect to a target specified by command line and returns the result as exit code.</p> <p>query Run the queries on the targets.</p> |
| logfile | TEXT | Sets the filename for the log file. |

| Name | Type | Description |
|------------------|---------|--|
| orahome | TEXT | This argument is used in conjunction with the argument <code>auto</code> . When given <code>oratrack</code> will not search for the Oracle instance home directories but scan for configuration files in this directory only. This argument also overrides the argument <code>discoverySource</code> . |
| oratab | TEXT | Specifies an oratab filename and overrides the discovery of that file. This argument conflicts with the argument <code>tnsnames</code> . |
| pass | TEXT | Specifies the password for connecting to a database. |
| path | TEXT | Sets the output directory. |
| plainQueries | TEXT | <p>Format: <code><filename>[,<query name>][;...]</code> Specify additional text files with queries and an optional name for the result, where the last result is written to the output file.</p> <p><code><filename></code> Filename for the text file. <code><query name></code> Optional name for the result (default is <code>PlainQuery</code>)</p> |
| port | INTEGER | Specifies a target port number or a pattern to match target port numbers, which were not specified by command line. |
| queryPath | TEXT | Specify the queries filename. By default the current working directory is searched for a file called <code>query.xml.enc</code> with fallback to <code>query.xml</code> |
| reportQueryError | BOOLEAN | <p>Enabled to create an error report instead of a regular inventory on a failed query that is considered an error. Disabled by default.</p> <p><code>true</code> Enable error report. <code>false</code> Disable error report.</p> |
| reportStatus | BOOLEAN | <p>Controls the connection status reporting.</p> <p><code>true</code> On a failure / error creates an NDI file that reports this failure / error. <code>false</code> Does not output an NDI for a failed connection.</p> |
| reportStandbyDB | BOOLEAN | <p>Controls the standby database reporting enabled by default.</p> <p><code>true</code> Looking for indicators for a standby database for</p> |

| Name | Type | Description |
|------------------------|---------|--|
| | | the target database. <code>false</code> Do not look for indicators for standby databases. |
| <code>silent</code> | BOOLEAN | Controls the output to the standard output channel. <code>true</code> Suppress logging to the standard output channel. <code>false</code> Allow logging to the standard output channel. |
| <code>sname</code> | TEXT | Format: <code><service name> <SID> <pattern></code> Specifies a target service name or SID or a pattern to match target services, not specified by command line. <code><service name></code> A concrete service name. <code><SID></code> A concrete SID. <code><pattern></code> Use targets with service names that match this expression only. |
| <code>sqlnet</code> | TEXT | Specifies a sqlnet configuration filename and overrides discovery of that file. |
| <code>tnsnames</code> | TEXT | Specifies a <code>tnsnames</code> filename and overrides discovery of that file. This argument conflicts with argument <code>oratab</code> . |
| <code>tnslister</code> | TEXT | Specifies a <code>tnslister</code> configuration filename and overrides the discovery of that file. Discovery or explicit specification of this file is implied when argument <code>auto</code> is set to <code>local</code> . |
| <code>upload</code> | TEXT | Specifies the upload location's URL for the output files. This contradicts argument <code>csv</code> with value <code>true</code> and argument <code>path</code> . |
| <code>user</code> | TEXT | Specifies the username for connecting to a database. <code><username></code> The username <code>?</code> Indicates that the username is read from standard input during execution |


Tip:

You cannot use the arguments `path` and `upload` together. The arguments `tnsnames`, `tnslister`, `sqlnet`, and `oratab` are allowed in conjunction with argument `auto`, only. The argument `csv=true` will only produce output for query files with queries with attribute `LMS="true"`.

Examples

1. Query single target by RVP

This is how the RayVentory Scan Engine UserUI will call oratrack in order to scan a single target.

```
-o dbhost=<host address> -o sname=<service name> -o port=<port number> -o  
authPath=<RVP UserUI path to OracleConnections.xml> -o user=<username> -o  
pass=<password> -o path=<RVP UserUI path to Oracle query results folder>
```

2. Test single target by RVP

This is how the RayVentory Scan Engine UserUI will call oratrack in order to test connection to a single target.

```
-o dbhost=<host address> -o sname=<service name> -o port=<port number> -o  
authPath=<RVP UserUI path to OracleConnections.xml> -o user=<username> -o  
pass=<password> -o mode=test
```

3. Query all target from a targets file given by RVP

This is how the RayVentory Scan Engine UserUI will call oratrack in order to scan all targets from RVP's targets file.

```
-o authPath=<RVP UserUI path to OracleConnections.xml> -o path=<RVP UserUI path to  
Oracle query results folder>
```

4. Scan all local targets found in the local configuration files using OS Authentication

Oratrack is installed on every machine that hosts Oracle databases and setup to be frequently run by a scheduler.

The user that runs oratrack is setup for OS Authentication with the targets. Instead of the argument path you can provide the argument upload

```
-o path=<output path> -o auto=local -o auth=false
```

5. Scan all targets found in the local configuration files using OS Authentication

This is an alternative to use case 4.

Oratrack is installed on a machine that knows several Oracle DBs and setups to be frequently run by a scheduler.

The user that runs oratrack is setup for OS Authentication with the targets.

```
-o path=<output path> -o auto=all -o auth=false
```

6. Scan all targets specified in a given tnsnames file with encryption and integrity options specified in a given sqlnet configuration file.

This is an alternative to use case 4.

Oratrack is installed on a machine that holds a tnsnames file and a sqlnet configuration file.

The user that runs oratrack is setup for OS Authentication with the targets.

```
-o path=<output path> -o auto=all -o auth=false -o tnsnames=<path to tnsnames file> -o  
sqlnet=<path to sqlnet configuration file>
```

Privileges

What privileges are needed depends on how ORATRACK is operated. For running queries against a database instance, the user that is passed to ORATRACK needs permissions to read certain

system tables and views. What exactly is needed may change. The Raynet consultants can explain what permissions are needed and give you scripts for creating a user with the needed permissions. You may always use a privileged user like sys.

To make use of ORATRACK's database discovery capabilities, you need to run it as a user with permission to read certain configuration files like `oratab`, `tnsnames.ora`, `listener.ora`, and `sqlnet.ora`. Further, to run `lsnrctl` and read the Oracle databases' home directories.

Ports in Use

- 1521 (default) for communication with an Oracle database via a TNS listener.

NDTRACK

Use the program to run a local OS inventory on Windows platforms. The correct scope for the NDTRACK inventory needs to be specified. Please refer to the RayManageSofti documentation *Ref_PreferencesForMD.pdf* for command line arguments.

NDTRACK (`ndtrack.exe`) is a command line oriented tool for generating inventories from a Windows or unixoid platform including different flavors of Linux. It is locally executed on the target host and produces an NDI file that may be uploaded to an inventory data sink such as RVP, RV Server, or RMS AS.

The command line arguments are passed to NDTRACK as name-value pairs. Flags are set by their name.

Usage:

```
ndtrack.exe [-o <argument-name>=<value>]...
```

or

```
./ndtrack.sh [-o <argument-name>=<value>]...
```

Table of Arguments and Flags

| Name | Description |
|----------------------------|---|
| BenchmarkCPU | |
| CheckCertificateRevocation | Determines whether it checks the certificate revocation list when accepting web server signatures from an HTTPS server. |
| Compress | Determines whether inventory files are compressed for upload. |
| CompressedExtension | File extension for the compressed inventory file. The default is <code>.gz</code> . |

| Name | Description |
|----------------------------------|--|
| Configuration | |
| DateTime | Datetime for the inventory file. |
| DateTimeFormat | Datetime format string for the inventory file. |
| DETECTCmdLine | |
| DetectMultiCore | |
| DetectSerial | |
| DeviceID | RMS / RV network device ID for inventory binding. |
| Difference | Determines whether differential inventories are performed on the managed device. |
| EmbedFileContentDirectory | What directory to include for <code>files</code> can. |
| EmbedFileContentExtension | What file extension to include for <code>files</code> can. |
| EmbedFileContentMaxSize | Defines the maximum file size for <code>files</code> can. |
| EnableWinOldAppPolicySetting | |
| ExcludePermissionsMask | Permissions for files to be excluded. |
| ExcludeDirectory | Folders to exclude from inventory. |
| ExcludeEmbedFileContentDirectory | What directory to exclude from <code>files</code> can. |
| ExcludeExtension | What file extension to exclude from <code>files</code> can. |
| ExcludeFile | What filenames to exclude from <code>files</code> can. |
| ExcludeFileSystemType | What file system type to exclude from <code>files</code> can. |
| ExcludeMD5 | Files matching this MD5 checksum are excluded from the inventory. |
| ExpectHardware | |
| ExpectSoftware | |
| ExpectTrackFilesInUserInventory | |
| Full | |
| FullQualifiedDomainName | |
| GenerateMD5 | Generate an MD5 for every file being tracked. |
| Generation | |
| GenerationMax | The number of differential inventories performed between |

| Name | Description |
|-------------------------|---|
| | full inventories. |
| Hardware | Determines whether to track hardware (in the machine context). |
| HyperV | |
| IncludeDirectory | Folders to include in the inventory. |
| IncludeExecutables | |
| IncludeExtension | File extensions to include in the inventory. |
| IncludeFile | Files to include in the inventory. |
| IncludeFileSystemType | |
| IncludeMachineInventory | Specifies whether or not to conduct a computer inventory of hardware and all user packages. |
| IncludeMD5 | Files matching this MD5 checksum are included in inventory. |
| IncludeNetworkDrives | |
| IncludePermissionsMask | Octal mask for file permissions on non-Windows clients that limits which files are scanned for inventory. |
| IncludeRegistryKey | Registry keys or values to include in the inventory. |
| IncludeUserInventory | Specifies whether or not to conduct a user inventory. |
| IncrementalDiff | When differential inventories are performed this determines what differences RayManageSoft will collect. |
| Inventory | |
| InventoryDirectory | Directory for saving the inventory file. |
| InventoryExtension | File extension for saving the inventory file. |
| InventoryFile | File name of a local copy of the inventory file. |
| InventoryScriptsDir | (Applies to Windows targets, only): If <code>RunInventoryScripts</code> is <code>True</code> , this specifies the location of scripts to be run after the inventory scanning completes. |
| InventoryType | |
| job | Job tag / ID (this is supposed to be used by Raynet's inventory / discovery agents only). |
| LibsmbiosCmdLine | |

| Name | Description |
|---------------------------|---|
| LogMsgCatPath | |
| LowProfile | |
| MachineId | See DeviceID. |
| MachineInventory | |
| MachineInventoryDirectory | Location for computer inventories. |
| MachineName | Machinename for binding inventory data. |
| MachineZeroTouchDirectory | |
| ManageSoftPackages | Determines the installed software packages. |
| MinInventoryInterval | Specifies the minimum interval (in hours) between the collections of inventories. |
| MSI | Determines whether Microsoft Installer (MSI) packages are included in the inventory. |
| MSSQL | |
| NetworkSense | Determines whether network checks are bypassed. |
| PackageDatabaseTypes | |
| PkgCacheDirectory | |
| PkgLiveDirectory | |
| PlatformSpecificPackages | |
| PNP | |
| ProgramFiles | |
| ProgressDepth | |
| Recurse | |
| ReportingLocation | URL of the RVS / RMS AS / DS reporting location for uploading files. |
| RunInventoryScripts | (Used in conjunction with InventoryScriptsDir): Specifies whether or not to run inventory scripts after gathering inventory data. |
| Security | Determines whether or not to perform security compliance checking. |
| SecurityAnalysis | |
| SecurityAnalysisFile | |

| Name | Description |
|---------------------------|---|
| SecurityAnalysisRule | |
| ShowIcon | (Applies to Windows targets only): Controls whether RayManageSoft displays an icon in the system tray. This preference can be set separately for the installation agent and inventory agent. |
| SkipInventory | |
| Software | |
| SMBIOSCmdLine | Command line to run the <code>smbios</code> utility. |
| SwidTagExtension | File extension for files that are to be handled as Software-ID tags. |
| SysDirectory | |
| Title | |
| TLMAgentIniPath | |
| TLMStandaloneIniPath | |
| TrackFilesInUserInventory | |
| TrackProductKey | |
| Upload | |
| UploadFile | |
| UploadLocation | The URL of the reporting location to upload to (for example <code>http://myrvserver/ManageSoftRL/</code>). The file will be uploaded to <code>http://myrvserver/ManageSoftRL/inventories</code> . |
| UploadRule | |
| UploadRuleValue | |
| UploadUser | |
| UploadPassword | |
| UploadProxy | |
| UploadRetries | |
| UploadType | Determines whether the upload agent uploads machine generated files or user generated files. For example, machine inventory, user inventory, all user installation logs, or current user installation logs. This is a functionality of the upload library that NDTRACK itself does not use but is |

| Name | Description |
|------------------------|---|
| | relevant if it is used as part of the RMS / RV managed device agent. |
| UploadDirectory | Directory to stage files for the upload. |
| UserUploadDirectory | Directory to stage user related files for the upload. |
| UserHardware | Determines whether to track hardware (in the user context). |
| UserId | User ID for binding inventory data. |
| UserInteractionLevel | Determines the level of user interaction. |
| UserInventory | |
| UserInventoryDirectory | Location for user inventories on the managed device. |
| UserName | Username for binding inventory data. |
| UserZeroTouchDirectory | |
| Version | |
| VersionInfo | Determines whether the file version header information is included in inventory or not. |
| WinDirectory | |
| WMI | (Applies to Windows targets only): Get hardware info by WMI. |
| WMIConfigFile | (Used in conjunction with WMI): File that defines (additional?) WMI queries. |

The files `ndtrack.sh` and `ndtrack.ini` must reside in the current directory or working directory. Otherwise, NDTRACK will not scan the hardware.



Tip:

- The optional http-upload does a HTTP-PUT.
- Instead of HTTP you may use FTP. Just change the protocol prefix from `http` to `ftp`.
- If you do not upload to a RayVentory or RMS server then you must make sure that the upload endpoint's URL ends on `/inventories`. For example: The URL is `http://myhost/Uploads/inventories` and command line argument is `-o UploadLocation=http://myhost/Uploads/`.

Examples (UNIX / Linux)

Create an inventory (Machine or 'everything' scope) and upload it to an RMS server. A copy will stay in the default inventory location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/
```

Create an inventory (Machine or 'everything' scope) and upload it to an RMS server by CURL. A copy will stay in the default inventory location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o  
UploadCurl=True
```

Create an inventory (Machine or 'everything' scope) and upload it to an RMS server by CURL, providing an username and a password for the upload. A copy will stay in the default inventory location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o  
UploadCurl=True -o UploadUser=myuser -o UploadPassword=mypassword
```

Create an inventory (Machine or 'everything' scope) and upload it to an RMS server by CURL using an username / password from the uploader credential store with a custom credential store key. A copy will stay in the default inventory location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o  
UploadCurl=True -o UploadCredFile=/home/someuser/mycredentials.cred -o  
UploadCredFileKey=myCredentialStoreKey
```

Create an inventory (Machine or 'everything' scope) and store it in the directory `/var/tmp/stuff` (generating an uncompressed `.ndi` file and a compressed `.ndi.gz` file):

```
./ndtrack.sh -t Machine -o Upload=False -o MachineInventoryDirectory=/var/tmp/stuff -o  
InventoryDirectory=/var/tmp/stuff -o UploadDirectory=/var/tmp/stuff
```

Create an inventory and store it together with a compressed copy to the default location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o Upload=False
```

Create an inventory, upload it, and specify a CA-bundle or CA-cert file. Use this if HTTPS fails because openssl is not able to get the issuer cert.

```
./ndtrack.sh -t Machine -o UploadLocation=https://myRVserver/ManageSoftRL/ -o  
SSLCACertificateFile='absolutePathToCACert.pem'
```

Create an inventory, upload it, and disable the server / peer-certificate verification (issuer-cert

and CRL check). Use this if HTTPS fails because server / peer-cert verification fails.

```
./ndtrack.sh -t Machine -o UploadLocation=https://myRVserver/ManageSoftRL/ -o
CheckCertificateRevocation=false -o CheckServerCertificate=false
```

Create an inventory and store it to `/home/user/inventory/linuxInventory.ndi` without compression:

```
./ndtrack.sh -o Upload=False -o InventoryDirectory=/home/user/inventory/ -o
InventoryFile=linuxInventory.ndi -o Compress=False
```

Create an inventory (Machine or 'everything' scope) and upload it to an RMS server by CURL using a certificate (the client-certificate and client-key must be PEM format) for client authentication. A copy will stay in the default inventory location `/var/tmp/managesoft/tracker/inventories/`:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o
UploadCurl=True -o UploadCurlCommand="curl --cert <client-certificate>:<optional
passphrase> --key <client-key>"
```

Or with a certificate to check the server certificate:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o
UploadCurl=True -o UploadCurlCommand="curl --cert <client-certificate>:<optional
passphrase> --key <client-key> --cacert <certificate-authority-certificate>"
```

Or with a client certificate that combines public and private keys, specifying a certificate format:

```
./ndtrack.sh -t Machine -o UploadLocation=http://myRVserver/ManageSoftRL/ -o
UploadCurl=True -o UploadCurlCommand="curl --cert-type <DER, PEM or ENG> --cert
<client-certificate>:<optional passphrase>"
```

Or:

```
./ndtrack.sh -t Machine -o UploadLocation=https://myDS/ManageSoftRL/ -o
UploadCurl=True -o UploadCurlCommand="curl --cert /home/myUser/Desktop/Certificates/
myServerAuthPrivateKey.pem -v"
```

Or:

```
./ndtrack.sh -t Machine -o UploadLocation=https://myDS/ManageSoftRL/ -o
UploadCurl=True -o UploadCurlCommand="curl --cert /home/<User>/<path>/<clientCert>.pem
-v"
```

Examples (Windows)

Create an inventory and store it to `%temp%\inventory\WindowsInventory.ndi`:

```
ndtrack.exe -o Upload=False -o InventoryDirectory=%temp%\inventory -o
InventoryFile=WindowsInventory.ndi -o Compress=False
```

Ports in Use

- 80 (443) for uploads of the inventory to an inventory data sink via HTTP(S).
- 445 for uploads of the inventory to an inventory data sink via SMB.
- 20/21 for uploads of the inventory to an inventory data sink via FTP.

SNMP Tracker

The SNMPTracker (`SNMPTracker.exe`) is a command line oriented tool for generating inventories by querying a target device using SNMP. This is intended for SNMP enabled network devices like printers, switches, routers, and UPSs. SNMPTracker is highly customizable and supports a wide range of different devices.

The command line arguments are passed to SNMPTracker as name-value pairs.

Usage

```
SNMPTracker.exe [-o <argument-name>=<value>]...
```

Table of Arguments

| Name | Type | Description |
|------------|---------|--|
| auth | OPTION | The authentication method. Valid values are MD5 and SHA1. Implies authphrase. |
| authphrase | TEXT | The authentication passphrase for SNMP v3. |
| community | TEXT | The community name for SNMP. Default is: public |
| config | TEXT | The configuration file. Alternatively, you can pass in one or more filenames after the options. |
| control | TEXT | The path or name of the control file, which is an XML file of a certain format. |
| deviceID | INTEGER | Sets the device ID (this is supposed to be used by Raynet's inventory / discovery agents only). |
| encsys | OPTION | Encryption system for SNMP v3. Valid values are AES and DES. Default is: DES |
| host | TEXT | Target host, alternative to scanRule. |
| job | TEXT | Sets the job tag / ID (this is supposed to be used by Raynet's inventory / discovery agents only). |
| maxDelay | INTEGER | Maximum seconds of delay for the upload. The value 0 disables delayed upload. Default is: 0 |

| Name | Type | Description |
|------------|---------|--|
| mibpath | TEXT | The path to the MIB files to compile. Default is: <code>mibs</code> |
| mibpattern | TEXT | The filename pattern for the MIB files to compile. Default is: <code>*.txt</code> |
| minDelay | INTEGER | Minimum seconds of delay for the upload. Default is: <code>0</code> |
| output | TEXT | Output directory for inventory files. |
| outfile | TEXT | Output file name if the option <code>host</code> is used. |
| port | SHORT | Port number for communication with the target device. Default is: <code>161</code> |
| privacy | OPTION | Privacy method for SNMP v3. Implies the argument <code>privphrase</code> . |
| privphrase | TEXT | Privacy phrase for SNMP v3. |
| protocol | OPTION | Protocol version. Default is: <code>2</code> |
| scanRule | TEXT | Supply a list of lists where the inner lists list targets hosts. If the current machine is <code><host></code> then target <code><targets></code> . Example: <code>myMachine1:target1,target2;myMachine2:target2</code> |
| security | OPTION | The security level for SNMP v3. Valid values are: <code>NOAUTHNOPRIV</code> , <code>AUTHNOPRIV</code> , and <code>AUTHPRIV</code> . |
| threads | INTEGER | Maximum number of threads. Default is: <code>1</code> |
| timeout | INTEGER | Timeout for SNMP responses. Default is: <code>2000</code> |
| upload | TEXT | URI for inventory upload. |
| useDns | BOOLEAN | Query the DNS for a hostname for an IP address. Default is: <code>True</code> |
| useProcess | BOOLEAN | Creates processes instead of threads for parallel queries. Default is: <code>False</code> |
| user | TEXT | The user for SNMP v3 authentication. |

Ports in Use

- `80 (443)` for upload of the inventory to an inventory data sink via HTTP(S).
- `445` for upload of the inventory to an inventory data sink via SMB.
- `161` for communication with the target devices via SNMP.

PowerShell Automation

RayVentory Scan Engine exposes its core functions via PowerShell commandlets bundled in a module. To automate the operations via PowerShell import the following module to your session:

```
[INSTALLDIR]\Libs\Raynet.RayVentory.ScanEngine.Automation.psd1
```

The following commands are available:

- Get-DeviceConnections
- Get-OracleConnections
- Get-SnmpConnections
- Get-VsphereConnections
- Import-DeviceList
- Import-LdapDevices
- Import-NetworkDevices
- Send-Inventory
- Start-DeviceInventory
- Start-OracleInventory
- Start-SnmpInventory
- Start-VsphereInventory

This user describes the usage of the `Get-DeviceConnections` and the `Send-Inventory` commandlets. For more information about other commands, execute the following command in the PowerShell session (after importing the module first):

```
Get-Help <command_name> -Full
```

Get-DeviceConnections

Returns all devices or a filtered collection of devices as shown in the **Devices + Services** screen. Returns a list of `OsConnection` objects.

Syntax:

```
Get-DeviceConnections [-DnsPatternName <wildcard>] [-IpAddressPattern <wildcard>] [-StatusPattern <wildcard>] [-OsType <Windows|Unix|Unspecified>] [-IsSingle]
```

Parameters:

All parameters are optional. Specifying filter parameters is additive (all specified filters must match before a device is returned).

-DnsNamePattern <wildcard>

The wildcard pattern (supports * for zero or more characters and ? for exactly a single character) for the DNS names of the returned devices.

-IpAddressPattern <wildcard>

The wildcard pattern (supports * for zero or more characters and ? for exactly a single character)

for the IP addresses of the returned devices.

-StatusPattern <wildcard>

The wildcard pattern (supports * for zero or more characters and ? for exactly a single character) for the states of the returned devices.

-OsType <Windows|Unix|Unspecified>

The type of the hosts to return.

-IsSingle

If specified, only the first record is returned.

Returns

List of `OsConnection` objects that match the given criteria.

Full Parameter Reference:

| Parameter name | Type | Is required? | Specific position? | Accepts pipeline input? | Set name | Aliases | Is dynamic? |
|------------------|------------|--------------|--------------------|-------------------------|----------|---------|-------------|
| DnsNamePattern | String | No | No | No | (all) | None | No |
| IpAddressPattern | String | No | No | No | (all) | None | No |
| IsSingle | Switch | No | No | No | (all) | None | No |
| OsType | <HostType> | No | No | No | (all) | None | No |
| StatusPattern | String | No | No | No | (all) | None | No |

Example:

To get all connections where the status contains the word "authentication":

```
Get-DeviceConnections -StatusPattern "*authen*"
```

To get all connections where the IP address starts with 192.168:

```
Get-DeviceConnections -IpAddressPattern "192.168.*"
```

To get all UNIX devices where the last inventory succeeded:

```
Get-DeviceConnections -OsType Unix -StatusPattern "OK"
```

To get all devices, show them as a table, and group them by the discovery source:

```
Get-DeviceConnections | Format-Table -GroupBy CreatedBy
```

Send-Inventory

Uploads the inventory files to the parent upload location.

Syntax:

```
Send-Inventory -UploadLocation <string> [-Credentials <PSCredential>]
```

Parameters:

Upload location is required, the credentials are optional:

-UploadLocation <string>

The full upload location of the parent upload server.

-Credentials <PSCredential>

The optional credentials to authenticate against the upload location server.

Full Parameter Reference:

| Parameter name | Type | Is required? | Specific position? | Accepts pipeline input? | Set name | Aliases | Is dynamic? |
|----------------|--------|--------------|--------------------|-------------------------|----------|---------|-------------|
| UploadLocation | String | Yes | 0 | No | (all) | None | No |
| Credentials | String | No | 1 | No | (all) | None | No |

Example:

To upload inventory files to the specified parent:

```
Send-Inventory "http://parent:80/Inventories"
```

OsConnection

The `OsConnection` objects have the following structure (sample data provided for a reference):

| Property | Value |
|-----------------------|---|
| Uuid | : be785fbd-ef84-4f9f-87da-dd073edc9717 |
| CreatedBy | : Discovery AD-import |
| CreatedDate | : 16.10.2018 09:09:44 |
| Credentials | : |
| CurrentInventoryFiles | : {} |
| Host | : MWS0231.raynet.corp |
| Hostname | : MWS0231.raynet.corp |
| Id | : hostname='MWS0231.raynet.corp', ipaddress='192.168.120.165' |
| IPAddress | : 192.168.120.165 |
| LastInventoryAttempt | : 16.10.2018 09:10:53 |
| LastInventoryDate | : |

```
SshPort           : 22
Status            : Authentication failed: No credentials allowed us to
run ndtrack via mgsreservice.exe.
TargetType        : Windows
UseWinSessionCreds : False
DisabledInventoryMethods : 0
LastSuccessfulInventoryMethod :
LastFailedInventoryMethods :
LastFailedInventoryMethodsDetail :
UserName          :
Password          :
AsSysDb           :
```

RayVentory Scan Engine is part of the RaySuite

More information online
www.raynet.de



Raynet GmbH

Technologiepark 20
33100 Paderborn, Germany
T +49 5251 54009-0
F +49 5251 54009-29
info@raynet.de

www.raynet.de