



EMC[®] Watch4net[®]

Version 6.3u2

Device Discovery CLI Guide

P/N: pending

Rev : 45575

EMC²

Copyright ©2010 - 2014 EMC Corporation. All rights reserved. Published in the USA.

Published April, 2014

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The information in this publication is provided as is. EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose. Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

EMC², EMC, and the EMC logo are registered trademarks or trademarks of EMC Corporation in the United States and other countries. All other trademarks used herein are the property of their respective owners.

For the most up-to-date regulatory document for your product line, go to EMC Online Support (<https://support.emc.com>).

Contents

Preface	iii
Intended Audience	iii
Prerequisites	iii
Where to get help	v
1 Product Overview	1
2 Architectural Overview	3
3 Controlling the Device Discovery Application	5
4 Typical use cases	11
Appendix	12
A SeedFileSourceAdapter Input File Format	13
B SNMP <i>Target Device</i> Definition	17

Preface

The purpose of this guide is to provide information about the *Device Discovery CLI module for APG*. It contains all the necessary information to install and configure the module, as well as usage and typical use cases.

Intended Audience

This guide is intended to be read by an *APG* administrator or systems integrator who will write scripts and schedule them in order to automate the use of the *Device Discovery* application.

Prerequisites

It is assumed that the *APG* application has been properly installed, as well as the *Device Discovery*. The *Device Discovery* must be running and fully configured in order to use the *Device Discovery CLI module for APG*. What is meant by a fully configured instance is as follows:

- At least one Device Source is configured.
- The SNMP Capability Discoverer is fully configured and the required capability discovery action scripts are provided for all capabilities that can be polled by the configured SNMP Collectors handled by the SNMP Collector Proxies.
- At least one SNMP Collector Proxy is configured and the SNMP Collector that is bound to it is running.

The Device Discovery instance must be configured through its GUI. See the APG-Device-Discovery-GUI.pdf document for more information.

Where to get help

EMC support, product, and licensing information can be obtained as follows:

PRODUCT INFORMATION - For documentation, release notes, software updates, or information about EMC products, go to EMC Online Support at:

<https://support.emc.com>

TECHNICAL SUPPORT - Go to EMC Online Support and click Service Center. You will see several options for contacting EMC Technical Support. Note that to open a service request, you must have a valid support agreement. Contact your EMC sales representative for details about obtaining a valid support agreement or with questions about your account.

Product Overview

The *Device Discovery CLI module for APG* is a component used to control the actions performed by a Device Discovery instance already fully configured through its GUI.

Actions to be performed by the Device Discovery instance are triggered by integration scripts. This approach is used because it allows the most flexibility when integrating the Device Discovery into existing systems and process flows. These integration scripts are provided by Watch4Net's professional services, Watch4Net's partners or customers themselves. Possible actions performed by the Device Discovery instance through the command line interface (CLI) include:

- Import Device Candidates from Device Sources.
- Discover the Device Candidates Capabilities.
- Generate SNMP Collector configurations by distributing Devices to them.
- Apply the generated SNMP Collector configuration.
- Enable and disable Device Sources.
- Enable and disable SNMP Collector Proxies to prevent them from accepting devices upon distribution.
- and further management Device management actions.

2

Architectural Overview

This section describes the *Device Discovery* main components and their relationships with the *Device Discovery CLI module for APG*. The scripts calling the Device Discovery CLI commands must be provided by the integrator or customer. These scripts can either be called manually or through scheduled jobs in order to automate the workflow.

The input seedfile, if used, will also be provided by the integrating party. The format of this seed file is described in the APG-Device-Discovery-GUI.pdf document, Appendix C.

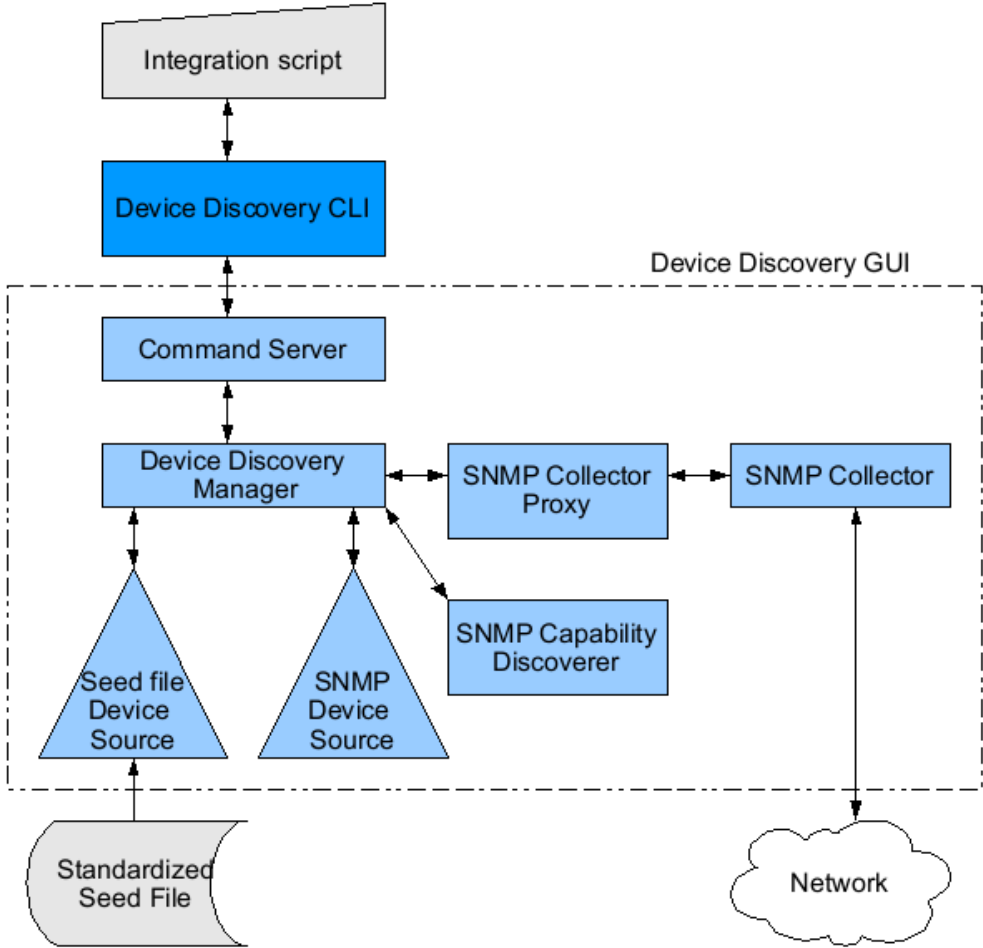


Figure 1: Architecture

3

Controlling the Device Discovery Application

This section provides information about how to use the Device Discovery CLI application.

Setting up the Device Discovery application

The Device Discovery application must be setup to accept commands from the Device Discovery CLI module. This is done by performing the following steps:

1. Edit the `<APG>/Custom/WebApps-Resources/Default/device-discovery/command-server.xml` file. Specify the interface to use when listening to incoming commands. This is done by editing the `<tns:bind-to-interface-lpAddress>` element. It may be left at 0.0.0.0 in order to let the application listen on all interfaces. Setting this value to 127.0.0.1 will cause the application to listen only on the localhost interface.

2. Limiting access to the *Device Discovery* may be done by adding one or more `<tns:acceptCommandsFromIPAddress>` elements in the `command-server.xml` file. By default, the *Device Discovery* application accepts commands from any host. Adding an IP address in this list will white list it, meaning that it becomes the only host which can use the CLI to send commands to the *Device Discovery*. Commands sent from any other host will be refused. As mentioned, one or more IP addresses may be added in this access list.
3. Restart the APG-Tomcat service to apply changes on the running *Device Discovery*. The command server would now be listening on the specified interface. The default port is 2011 (may be customized in the `command-server` file).

Script arguments

A command line script is provided in order to control the *Device Discovery* application. This script is usually located in `<APG>/bin/` and is called `send-command-to-device-discovery`, where its extension is either `.cmd` (on Windows OS) or `.sh` (on UNIX type OS).

The `send-command-to-device-discovery` scripts accept several arguments:

- `-c` The command name. This is mandatory.
- `-t` The device type. Right now, the only valid device type is "snmp". This argument is optional, depending upon the command type.
- `-a` Arguments for the command, separated by commas if several are present.
- `-i` The IP Address of the *Device Discovery* application. It defaults to 127.0.0.1 (localhost) if not provided.
- `-p` The listening port of the *Device Discovery* application. It defaults to 2011 if not provided.
- `-b` The IP Address of the interface to be used when communicating with the *Device Discovery* application. Defaults to kernel auto-select if not provided.

Script commands

The following lists the available CLI commands:

1. `ApplyConfigurations`: Apply the generated configuration to the specified collectors. One or more collectors may be specified with this command. Alternately, the device type (SNMP) may be specified here instead to target all applicable collectors.

Command example:

To apply distribution to all enabled SNMP Collectors:

`send-command-to-device-discovery ... -c ApplyConfigurations -t snmp`

To apply distribution to SNMP Collectors named “abc d” and “MyTest 1”:

`send-command-to-device-discovery ... -c ApplyConfigurations -a “abc d,MyTest 1”`

2. **ClearMessageStack:** Clear the Device Discovery Command Server message stack. This message stack contains the information about requests sent by the Device Discovery Command CLI client.
3. **ClearTargetDeviceContainers:** Clear the specified target device container. Multiple containers can be specified by providing the device type: all container for this type will be cleared in case. Or specified explicitly, by their name, as argument values.
4. **DisableCollectorProxy:** Disable the specified collector proxy (by name, 1 or many). A disabled collector proxy will not accept devices when configurations are generated. This effectively blocks its configuration from being updated.

Command example:

To disable SNMP Collector Proxy for collectors named “abc d” and “My Test 1”:

`send-command-to-device-discovery ... -c DisableCollectorProxy -a “abc d,MyTest 1”`

5. **DiscoverTargetDeviceCapabilities:** Discover the capabilities of the imported Device Candidates only. The Device type (snmp) must be specified. Command example:
`send-command-to-device-discovery ... -c DiscoverTargetDeviceCapabilities -t snmp`
6. **EnableCollectorProxy:** Enable the specifies collector proxy (by name, 1 or many). An enabled proxy will accept devices when configurations are generated, thus updating its configuration.
7. **GenerateConfigurations:** Distributes all Discovered Devices, specified by their device type (SNMP), on enabled Collector Proxies of the same device type. The collector configuration is then generated and written. These configurations may be applied to their collectors or imported into a MIB Browser for further fine tuning. A request name will be returned by this command. This name may be used to track the request completion in the message stack.
8. **GetMessageStack:** Get the Device Discovery Command Server message stack content. Search criteria may be included in the arguments to filter out non-essential information. The complete stack content will be returned if no argument is provided.
9. **ImportTargetDevices:** Import the Device Candidates from the specified sources (specified by name, or device type).

10. Interrupt Capability Discovery: Interrupts the capability discovery process on the specified device type name.
11. InterruptTargetDevicesImport: Interrupts the device Candidate import process on the specified device source.
12. List Capability Discoverers: List the available capability discoverers.
13. ListCollectorProxies: List the available collector proxies, specified by the device type name (SNMP) or all.
14. ListTargetDeviceSources: List the available device sources, specified by the device type name (SNMP) or all.
15. ListTargetDevicesInTargetDeviceContainer: List the Devices or Device Candidates present in the specified TargetDeviceContainer. The target device container name must be provided.
16. ListTargetDeviceContainers: List the available Devices or Device Candidates containers. A container actually represents a state. The target device type must be specified.
17. MoveDevices: Move Devices from a source Target Device Container to a destination Target Device Container, effectively allowing Device state changes.

Command example:

send all Devices from the IMPORTEDsnmp Target Device Container to the DISCOVEREDsnmp Target Device Container:

```
send-command-to-device-discovery ... -c MoveDevices -a IMPORTEDsnmp, DISCOVERED-  
snmp
```

Command example:

send Devices specified in file myDevices.txt from the IMPORTEDsnmp Target Device Container to the DISCOVEREDsnmp Target Device Container:

```
send-command-to-device-discovery ... -c MoveDevices -a IMPORTEDsnmp, DISCOVERED-  
snmp,myDevices.txt
```

The content of the myDevices.txt file is a simple 1 deviceId per line, preceded by an optional include/exclude keyword. Fields are separated by a TAB character.

The "include" keyword simply specifies which Device to move from the source to the destination Target Device Container; other devices found in the source will not be moved when at least one device is "included" in the file.

The “exclude” keyword specifies which Device not to move from the source to the destination Target Device Container; other devices will be moved to the destination if at least one device is “excluded” in the file.

Combining both keywords in this file will result in moving only “included” devices to the destination.

File content example: move all included devices from the source to the destination.

```
include .W10N-FL17-151.4223
include .W10N-TEST-71.5830
include .W10N-TEST-9-213.8132
include .W10N-TEST-135.134
```

File content example: move all devices from the source to the destination, excepted the ones that are excluded.

```
exclude .W10N-FL17-151.4223
exclude .W10N-TEST-71.5830
exclude .W10N-TEST-9-213.8132
exclude .W10N-TEST-135.134
```

Useful Target Device Container names include:

- IMPORTED: This contains approved Device Candidates only.
- IMPORTED_APPROVAL: This contains pending Device Candidates.
- DISCOVERED: This contains approved Devices.
- DISCOVERED_APPROVAL: This contains pending Devices.
- DISCOVERED_FAILED: This contains Devices for which the capability discovery process failed.
- EXCLUDED: This contains the explicitly excluded Devices.

Note: Using the Device Discovery GUI while sending CLI commands may lead to inconsistencies. CLI commands should be used for automatic collector provisioning, while the GUI should be used for manual interventions such as Device Source creation and update, editing *Target Devices* and other non-automated tasks.

4

Typical use cases

A typical use case for the Device Discovery CLI module could follow this example:

1. Start the APG Tomcat service (this starts the Device Discovery application).
2. Update the Device Source seedfile by copying it to its location (via ftp, scp or the APG Centralized Configuration Manager). The seedfiles are usually located in the *<APG>/Custom/WebApps-Resources/Default/device-discovery/seedfiles* directory.
3. Import the Device Candidates by sending the ImportTargetDevices command with the Device Source name as the argument.
4. Trigger the Capability discovery process on the imported Device Candidates by sending the DiscoverTargetDeviceCapabilities command, with snmp as the device type. The discovery batch name will be returned.
5. Check the message stack periodically by sending the GetMessageStack command and the batch name as the search argument. A message stating that the request has been processed will be sent eventually. Capability discovery can be a lengthy process when performed on a large network (1000 devices) but it is possible to calculate the longest scenario (all devices timeout) by using the following formula:

Total discovery time = ((devices) / (discovery threads)) * (average number of SNMP tries * SNMP timeout)

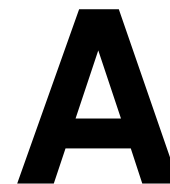
Example:

(1000 devices / 10 threads) * (4 tries * 3 seconds) = 1200 seconds = 20 minutes.

6. Once the Capability Discovery is done, Distribute the Devices to the enabled SNMP Collector by sending the GenerateConfigurations command, using snmp as the device type.
7. Apply the generated configurations to the SNMP Collectors, specified by the snmp device type.

This typical use case is only a best case scenario, where errors are not checked for. Common additional steps often include:

- Checking if there are Devices in the DISCOVERED_FAILED list: this would indicate Devices for which the discovery process did not work correctly.
- Checking if there are Devices in the IMPORTED_APPROVAL or DISCOVERED_APPROVAL list: these devices will not be part of the generated configurations.
- Checking the generated configurations by downloading them and making sure there are some devices in them. The download could be performed via FTP, SCP or the APG CCM module. Parsing the configuration to make sure they contain devices should be done via a script.



SeedFileSourceAdapter Input File Format

The SeedFileSourceAdapter is meant to read a file which contains one *Target Device* definition per line. The *Target Device* attribute fields are separated by a TAB character. The TAB character is invalid as the value itself. An attribute field contains the attribute name, followed by the “=” character, and a value. A *Target Device* definition must contain at least one required field for indexing purposes, while other fields are optional.

A minimal input file, based on the provided SNMP *Target Device* definition (Appendix B), may contain the following information:

```
name=name1 snmpPort=161 devtype=Router ipAddress=192.168.0.1 capability=ipsla
snmpCommunityOrUserV3Name=commString1
name=name5 snmpPort=161 devtype=Server ipAddress=192.168.0.5 snmpPort=161 capability=server
snmpCommunityOrUserV3Name=commString1
name=name2 snmpPort=161 devtype=Router ipAddress=192.168.0.2 collectorGroupName=net1
snmpCommunityOrUserV3Name=commString1
name=name6 snmpPort=161 devtype=Router ipAddress=192.168.0.6 collectorGroupName=net1
collectorGroupName=net2 snmpCommunityOrUserV3Name=commString1
```

```

name=name7 snmpPort=161 devtype=Node ipAddress=192.168.0.7 collectorName=W4nSnmp-1
snmpCommunityOrUserV3Name=commString1
name=name8 snmpPort=161 devtype=Node ipAddress=192.168.0.8 collectorName=W4nSnmp-1
snmpCommunityOrUserV3Name=commString1 pollingPeriod=3600
name=name9 snmpPort=161 devtype=Node ipAddress=192.168.0.9 collectorName=W4nSnmp-1
snmpCommunityOrUserV3Name=commString1 capability=server@1800
name=name10 snmpPort=161 devtype=Node ipAddress=192.168.0.8 collectorName=W4nSnmp-1
snmpCommunityOrUserV3Name=commString1 pollingPeriod=3600 capability=ipsla capability=
server@1800
name=name11 snmpPort=161 devtype=Node ipAddress=192.168.0.9 collectorName=W4nSnmp-1
snmpCommunityOrUserV3Name=commString1 capability=server@1800 capability=ipsla

```

SNMP Target Device attribute field descriptions The following are the attribute fields which can be used in a SNMP seed file. Please note that all fields are optional, except the name and snmpPort fields, which are part of the *Target Device* id.

- *name*: The name of the *Target Device*. This field is mandatory as it is an index field for the *Target Device*.
- *devtype*: The *Target Device*'s type. Common values include: "Host", "Node", "Router", "Switch". Any value may be set in this field. Use "Node" to indicate that the *Target Device* type is unknown.
- *doNotUseGetBulkRequests*: Specifies to use only snmp requests for polling SNMP v2c and v3 agents when set to true. Valid values: "true", "false".
- *ipAddress*: The *Target Device*'s IP Address. This may be an IPv4 or a fully defined IPv6 address. This field is *mandatory* when using the seed file with a SeedFileSourceAdapter.
- *onFailTryWith*: When polling agents, use the specified version specific requests for supplementary retry. Valid values are: "v1" (get request), "v2" (getBulk request), "doNotRetry" (do not perform supplementary retry).
- *transportProtocol*: The transport protocol used to communicate with the SNMP agent residing on the *Target Device*. This defaults to udp when not provided, although tcp may also be used.
- *snmpPort*: The port used to reach the SNMP agent residing on the *Target Device*. This value defaults to 161 when not specified. It is an index field for the *Target Device*.
- *timeout*: This is the SNMP request maximum time (in milliseconds) out when querying the SNMP agent residing on the *Target Device*. The default value is 1000.
- *retries*: This is the number of SNMP request retries to send to the SNMP agent if it does not answer within the allowed time out value. The default value is 2.

- *snmpVersion*: The SNMP version to use when communicating with the SNMP agent residing on the *Target Device*. Allowed values include: v1, v2c and v3. The default value is v2c.
- *snmpCommunityOrUserV3Name*: The SNMP community string (or SNMP v1 and v2c) or user name (SNMP v3). This is required.
- *snmpV3AuthenticationPassword*: The SNMP v3 authentication password. This is required when using SNMP v3.
- *snmpV3AuthenticationProtocol*: The SNMP v3 authentication protocol. Allowed values include: md5, sha. This is optional (noAuthNoPriv if not provided) when using SNMP v3.
- *snmpV3PrivacyPassword*: The SNMP v3 privacy (encryption) password. This is optional when using SNMP v3.
- *snmpV3PrivacyProtocol*: The SNMP v3 privacy (encryption) protocol. Allowed values include: aes, des. This is optional when using SNMP v3.
- *snmpV3ContextName*: The SNMP v3 context name. Default empty context name is used if not provided. Optional.
- *useV3ContextNameInVariable*: Set the SNMP v3 context name in the variable. This will create variables linked to the context name.
- *snmpV3ContextEngineId*: The SNMP v3 context engine id. Optional. Default to empty if not provided.
- *useV3ContextEngineIdInVariable*: Set the SNMP v3 context engine id in the variable. This will create variables linked to the context engine id.
- *delete* The delete attribute. Setting this attribute to true will cause the SeedFileSourceAdapter to delete this *Target Device* if the import priority is greater or equal than the existing *Target Device* and that the source action policy is set to DO_ACTION. It may also cause the *Target Device* to be sent to the pending list, depending on the source action policy.

collectorName and collectorGroupName fields A *Target Device* may contain one or more collectorName values. The collectorName values are used when distributing the *Target Devices* among the collectors. A *Target Device* with a specified collectorName attribute can only be polled by the same-named collector. Having several collectorNames means that the *Target Device* may be polled by anyone of the specified collector. The same behavior applies to the collectorGroupName attribute: a *Target Device* assigned to a collector group may only be polled by a Collector that is part of this collector group.

pollingPeriod A *Target Device* may have its own polling period, which overrides its collector default period. This polling period must be set in seconds. Supported values include positive integers. 0 or negative values are ignored. The pollingPeriod field is used to set the polling period.

capability fields A *Target Device* may have one or more capability fields. A capability field consists of a capability name followed by an optional polling period string. The polling period string contains a polling period separator followed by a polling period which should be a non-negative integer. The polling period separator string can be user specified using the system property *devicediscovery.pollingperiodseparator*. A default value of "@" is used if it is not specified. Invalid or negative values of polling period are reset to zero. The default capability (specified in the *Collector Proxy* configuration) is used when none are provided. The capability is used to determine whether a collector can poll a *Target Device* or not and for distribution purposes.

Note: In the case of the SnmpCollectorProxy, a capability corresponds to a valid snmp-mask defined in the collector's snmp-masks.xml configuration file.

doNotDiscoverCapability fields Some *Target Device* may not support specific capabilities, to the point that even trying to discover them may have adverse effects on the device behavior. Those capabilities which must not be searched for can be defined by adding one (or more) doNotDiscoverCapability value to the device definition. The doNotDiscoverCapability values may also be used when the provisioning system used to generate the seedfile can provide this information.

Note: All attribute fields, except collectorName, collectorGroupName, pollingPeriod, doNotDiscoverCapability and capability fields should be provided.



SNMP *Target Device* Definition

This release of the *Device Discovery* provides the SNMP agent *Target Device* definition as a valid type for import and export. This definition is specified in the `<APG>/Custom/WebApps-Resources/Default/device-discovery/target-device-defs.xml` file. The content of this file is listed below and is self-documented:

```
<?xml version="1.0" encoding="UTF-8"?>
<target-device-defs>
  <target-device-type type="snmp" collectorDomainInterface="NetworkCollectorDomain">
    <possible-properties>
      <property name="name" isIndex="true" valueType="string"/>
      <property name="devtype" value="Node" valueType="string"/>
      <property name="ipAddress" valueType="string"/>
      <property name="transportProtocol" value="udp" valueType="string"/>
      <property name="snmpPort" value="161" isIndex="true" valueType="int+"/>
      <property name="timeout" value="1000" valueType="int+"/>
      <property name="retries" value="2" valueType="int0+"/>
      <property name="snmpVersion" value="v2c" valueType="string"/>
      <property name="snmpCommunityOrUserV3Name" valueType="string0" encrypted="true"/>
      <property name="snmpV3AuthenticationPassword" valueType="string0" encrypted="true"/>
      <property name="snmpV3AuthenticationProtocol" valueType="string0"/>
    </possible-properties>
  </target-device-type>
</target-device-defs>
```

```
<property name="snmpV3PrivacyPassword" valueType="string0" encrypted="true"/>
<property name="snmpV3PrivacyProtocol" valueType="string0"/>
<property name="v1TableRowPolling" value="false" valueType="string"/>
<property name="onFailTryWith" value="v1" valueType="string"/>
<property name="doNotUseGetBulkRequests" value="false" valueType="string"/>
<property name="maxBulkSize" value="50" valueType="int+"/>
<property name="snmpV3ContextName" valueType="string0"/>
<property name="useV3ContextNameInVariable" value="false" valueType="string"/>
<property name="snmpV3ContextEngineId" valueType="string0"/>
<property name="useV3ContextEngineIdInVariable" value="false" valueType="string"/>
<possible-properties>
<target-device-type>
<target-device-defs>
```

Note: The devtype attribute can be set to Node when unknown. In this case, the SNMP Capability Discoverer component will poll the SNMP Agent and match its sysObjectOid with the correct devtype, if found in the oid2devtype file. This oid2devtype can be edited to add missing entries. Simply save the SNMP Capability Discoverer after the oid2devtype has been edited.