

HP Server Automation

for the HP-UX, IBM AIX, Red Hat Enterprise Linux, Solaris, SUSE Linux Enterprise Server, VMware, and Windows® operating systems

Software Version: 7.80

Technical Note: Software Discovery

Document Release Date: August 2009

Software Release Date: August 2009



Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

© Copyright 2009 Hewlett-Packard Development Company, L.P.

Trademark Notices

Intel® Itanium® is a trademarks of Intel Corporation in the U.S. and other countries.

Java™ is a US trademark of Sun Microsystems, Inc.

Microsoft®, Windows®, Windows® XP are U.S. registered trademarks of Microsoft Corporation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

To check for recent documentation updates or to verify that you are using the most recent edition of a document, go to:

<http://h20230.www2.hp.com/selfsolve/manuals>

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to:

<http://h20229.www2.hp.com/passport-registration.html>

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

HP Support

Visit the HP Software Support Online web site at:

www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

<http://h20229.www2.hp.com/passport-registration.html>

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

Software Discovery	7
Software Discovery Prerequisites	7
Generating Reports with SAR.	8
Supported Platforms and Configurations.	8
Downloading the Software Discovery Package.	8
Contents of the Software Discovery Package	9
Deploying Software Discovery Using Inventory Snapshots	10
Viewing Discovered Software on a Managed Server.	10
Inventory Snapshot Job Error Messages.	11
Software Discovery Permissions	12
How Software Discovery Works	12
Application Discovery	12
Application Signatures	13
Configuration and Customization	18
Configuration Attributes	18
Configuration Options	20
Syntax Errors	21
Discovery Filtering Process	21
How Filters Affect Scanning	21
Sample Configuration	21
Concurrency and Multi-User Considerations	22
Software Discovery Usage Examples	23
Example 1	23
Example 2	23
Example 3	24
Extending Software Discovery	24
Writing Custom Software Discovery Code	25

Software Discovery

The HP Server Automation (SA) Software Discovery feature provides a signature-based software discovery mechanism for Windows and UNIX managed servers to help you manage applications and software that are not already managed by SA. Specifically, the Software Discovery feature:

- Discovers unlicensed software, unregistered software, custom-built software and software that was not installed using one of the standard packaging technologies for Windows or UNIX.
- Creates an inventory of software programs that have not been installed as an OS-registered application.
- Provides system administrators the ability to create snapshots of all the discovered software on a server and then periodically audit against the snapshot over time. This can help you upgrade a server, remove unwanted software and modify a server to conform to your organization's security policies.
- Gives auditors a convenient method of capturing the current state of a server's software and discovering unsupported or unlicensed software installed on a server. By running the audit on a regular schedule, you can monitor software changes over time.

Software Discovery is a read-only server module that provides a rich inventory of software on a managed server.

Software Discovery Prerequisites

To deploy the Software Discovery feature, you must meet the following requirements:

- SA 7.50 or later, installed and configured.
- SA patch 7.50.01 or later, installed on the SA 7.50 core.
- A BSA Essentials Network account. To request a BSA Essentials account, see <http://www.hp.com/go/bsanetwork>.
- The HP BSA connector (previously called the Live Network connector or LNC) installed and configured on your core server. See the *HP Live Network connector Installation and Configuration Guide* which is available from the BSA Essentials web site, <http://www.hp.com/go/bsanetwork>.
- If you want to use SAR (Service Automation Reporter) to create reports about discovered software, you must first subscribe to the SAR software_discovery_reports stream. See [Generating Reports with SAR](#) on page 8.



Software Discovery is not supported on IA64 Linux.

Generating Reports with SAR

If you want to use SAR (Service Automation Reporter) to create reports about discovered software, you must first subscribe to the SAR `software_discovery_reports` stream. If you do not download this stream and you use SAR, some of the discovered software data will not appear in your reports generated from SAR. If you do not plan to use SAR, you do not need to download this stream. Use the following command to display the SAR streams:

```
live-network-connector list-streams --product=sar
```

Set the `software_discovery_reports` line to 1 in the `live-network-connector.conf` file:

```
# SAR Software Discovery Reports stream
software_discovery_reports=1
```

For more information, see the *HP Live Network connector Installation and Configuration Guide* available on the BSA Essentials Network web site, <http://www.hp.com/go/bsanetwork>.

For information about SAR, see *HP Service Automation Reporter User Guide*.

Supported Platforms and Configurations

Software Discovery is supported on all UNIX platforms and Windows versions which the SA Agent supports for managed servers.

For more information on managed servers supported platforms, see the *SA Release Notes* or the *SA Planning and Installation Guide*.

Software discovery is ISO-8859-1 compliant.

Downloading the Software Discovery Package

The Software Discovery package is available from the BSA Essentials Network. To get the Software Discovery package, you must use the BSA connector (previously called the Live Network connector or LNC).



For more information on BSA Essentials and to request a BSA Essentials account, visit the BSA Essentials web site at <http://www.hp.com/go/bsanetwork>.

The BSA connector downloads the Software Discovery package in the form of a zip file with the following name:

```
OPSWsmo_discovered_software-<version>.zip
```

This zip file is placed in the following directory on your SA core server:

```
/var/opt/opsware/ogfs/mnt/root/var/opt/opsware/sm
```

This directory can also be accessed from the SA Global Shell at the following directory:

```
/var/opt/opsware/sm
```

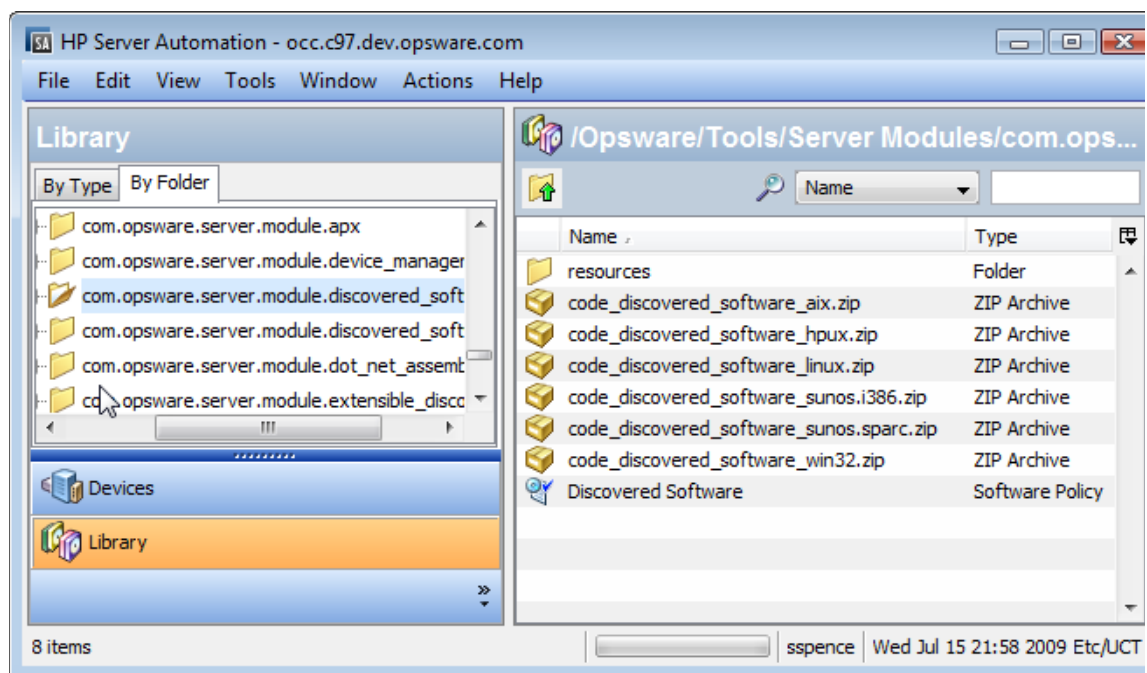

Contents of the Software Discovery Package

After you install the Software Discovery package, you can see the contents in the SA Client as follows.

- 1 In the SA Client, select Library in the navigation pane.
- 2 In the Library, select the By Folder tab.
- 3 Navigate to
Library/Opware/Tools/Server Modules/com.opsware.server.module.discovered_software.
This displays the following contents of the Discovered Software package as shown in [Figure 1](#).
 - A set of zip file packages, one for each supported managed server platform.
 - A software policy named Discovered Software that contains all the zip file packages.
 - A subdirectory named resources.

The Discovered Software policy is automatically remediated (installed) when you initiate a snapshot with the “Perform Inventory” option selected, as described in [Deploying Software Discovery Using Inventory Snapshots](#) on page 10.

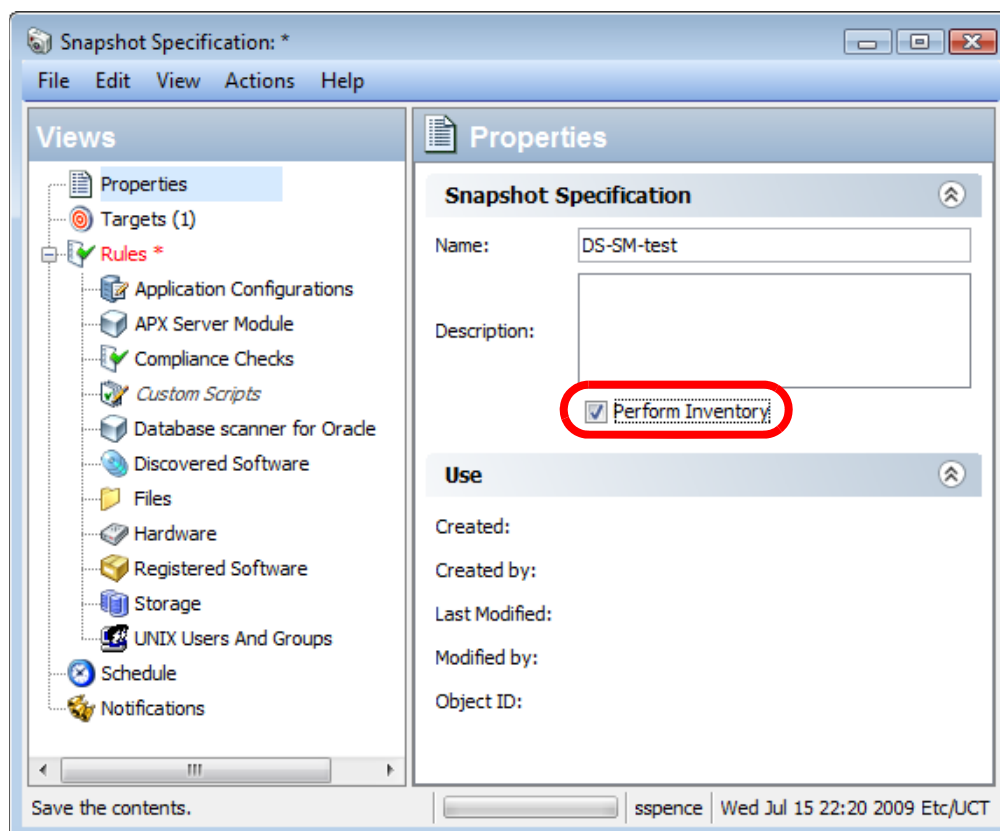
Figure 1 SA Client Library Showing Location of Software Discovery Packages



Deploying Software Discovery Using Inventory Snapshots

To deploy the Software Discovery feature to discover software installed on a server, you must run a snapshot with the “Perform Inventory” option selected, as shown in [Figure 2](#). For details, see “Creating a Snapshot Specification” and “Running a Snapshot Specification” in the *SA User Guide: Application Automation*.

Figure 2 Snapshot with Perform Inventory Option Selected to Enable Software Discovery



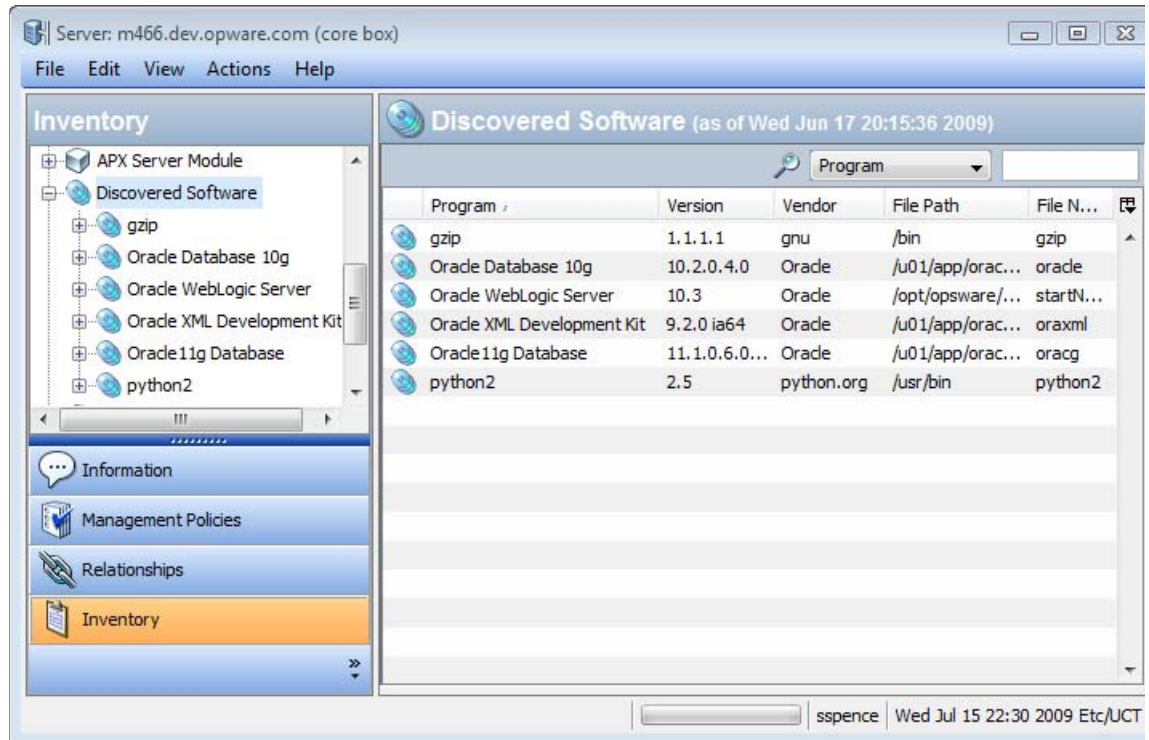
Each time a snapshot is run on a server with Perform Inventory selected, SA verifies that the Software Discovery server module is installed on the managed server. If the Software Discovery server module is not on the target server, a remediate job is launched automatically by the core that installs the contents of the Software Discovery Software Policy. The policy remediation occurs automatically when the snapshot is run.

This installs the Software Discovery server module on the server, including the signature zip package, which in turn enables you to view Software Discovery on a managed server’s Device Explorer, and use the Snapshot’s software inventory in an Audit.

Viewing Discovered Software on a Managed Server

The SA Client displays the software that has been discovered on the server in the Device Explorer in the Inventory view, as shown in [Figure 3](#).

Figure 3 Device Explorer Showing Software Discovered



Unlike other server objects in the managed server inventory that fetch data in real time from a server, the Discovered Software server object is configured to only show data from the latest inventory Snapshot (a Snapshot that has the Perform Inventory check box selected when the snapshot specification is created). If there are no inventory snapshots available, the server browser will show an error message suggesting that an inventory Snapshot should be made.

The Device Explorer displays a time stamp at the top of the Device Explorer window indicating the last time the inventory Snapshot was run on the server. For example, in [Figure 3](#), the Contents pane (right side) of the window shows the label: “(as of Thu May 29th 23:45:29 2008)”. This indicates the last time the inventory Snapshot was run.

For more information, see “Creating a Snapshot Specification” and “Running a Snapshot Specification” in the *SA User Guide: Application Automation*.

Inventory Snapshots in Audits

Once you have performed an inventory Snapshot, you can add it as the source of an audit, so whenever the audit runs, it will compare the original state of the snapshot with the current state of the server, and you can determine if the installed software has changed since you last ran the Snapshot.

For more information on Snapshots and Audits, see “Audit and Remediation” in the *SA User Guide: Application Automation*.

Inventory Snapshot Job Error Messages

The following error codes and messages are specific to Software Discovery when running an inventory Snapshot:

- * 1-TADNSW unexpectedly quit!

- * 2-Invalid JSON format
- * 3-Another instance of DS-SM is running.
- * 4-Database merge failed
- * 6-TADNSW run failure
- * 8-TADNSW Error: <error>

Software Discovery Permissions

The Software Discovery feature can be run on a server by any SA user with read access to the server and belongs to a user group that has the client feature permission “Allow Execute Server Modules: Yes”.

If a user wants to add or modify custom entries for Software Discovery, then the user will require the “Manage Server Modules: Read & Write” permission. Permissions are granted in the SAS Web Client.

For more information on SA permissions, see the *SA Administration Guide*.

How Software Discovery Works

Software Discovery provides a signature-based software discovery mechanism for SA Windows and UNIX managed servers. It consists of discovery logic, module metadata, and a signature database. This signature database contains application signatures from HP's Asset Management Tool, Discovery Dependency Mapping Inventory (DDMI), which is the Software Application Index (SAI) used by DDMI to discover software.



This feature is not designed to allow you to install or remediate software on a managed server. For information on using software policies to install and remediate software onto managed servers, consult the *SA User Guide: Server Automation*.

Application Discovery

Software Discovery uses DDMI SAI content for discovering applications on SA managed servers. The SAI signatures are used to generate the SCT (Signature Component Table) for use with Software Discovery. The SCT contains all the signatures imported from the DDMI SAI which are to be used by Software Discovery.

SCT application signatures are grouped together by product IDs and compared to the results retrieved by a file system scan. For every file found on the file system, a hash is computed using the DDMI hash algorithm. The signatures collected from the file scan are referred as 'raw' signatures. Application signatures represented in the DDMI SAI and corresponding SCT contain the same hash values computed from the same algorithm. The 'raw' signatures are then compared to the DDMI SAI signatures stored in SCT and a rating is calculated off the best possible match. The highest rated match is then reported as the application back to the user who initiated the scan.

All components are compared to the corresponding properties obtained from raw signatures. When a match occurs in any one of the components, the rating is incremented and used for gauging the best possible match among various product versions. When the best match is found, the product is reported using the display components:

```
DISPLAY PRODUCT: <display product-name>
DISPLAY VENDOR: <display vendor-name>
DISPLAY VERSION: <display version-number>
```

There are a few differences to note between the DDMI and Software Discovery results that are accepted around specific boundary conditions. The first difference in comparison is that Software Discovery will report all instances of software discovered on a server. For example, when evaluating multiple installs of the Java JRE. DDMI will only report the first install of a series of products of the same version. On the other hand, Software Discovery will report all instances of the installed product.

The second difference occurs when no exact matching signature is available in the SAI. In this situation, DDMI and the Software Discovery feature will attempt to find the next best match based off the ratings calculated. DDMI will evaluate all signatures, along with all 'associated' entries, to generate a high rating during guess estimates and find more accurate results. The Software Discovery feature will sometimes reach the same results in the guess. However, the guess does occasionally differ due to remaining 'associated' entries not being imported into SCT from the SAI.

Please note again that due to sizing constraints, not all SAI associated entries are imported into SCT. Only overlapping 'associated' entries between concurrent applications versions will be imported as well. As a result, SA will not pinpoint or discover suites or editions of applications.

Application Signatures

All application signatures are generated using the DDMI SAI Master.xml (WIN32) and unix.xml (UNIX). All application signatures are stored in a database for processing and identification during the scan process. SCT is used for the software discovery process in the database.

During a core install, these application signature packages must be uploaded to the core and attached to the Software Discovery Software Policy to which the Software Discovery package is attached. During upload, each package is associated with all UNIX or all Windows platforms. These packages are automatically installed onto a managed server when the server is remediated with respect to this policy. These signatures are used by the underlying scanning software to determine what software is discovered on a server.

Application Signatures from BSA Essentials Network

The Software Discovery feature comes with a database consisting of application signatures derived from the DDMI SAI content. When the Software Discovery package is imported into the SA core, all applications represented by those signatures are discovered. This signature database is updated periodically (usually monthly). These updates are provided through BSA Essentials Network and must be imported into the SA core.

The Software Discovery signature update package is a zip file, which has a file name similar to:

```
OPSWsmo_discovered_software_sig_prod-<version>.zip
```

After the BSA connector downloads the zip file, it is stored in the following directory on the SA core:

```
/var/opt/opsware/ogfs/mnt/root/var/opt/opsware/sm/data
```

This location can also be accessed from the Global Shell with the following command:

```
/var/opt/opsware/sm/data
```

The zip file can be uploaded to the core with the following dstool command:

```
/opt/opsware/smtool/dstool --user=<username> --pass=<password>  
OPSWsmo_discovered_software_sig_prod-<version>.zip
```

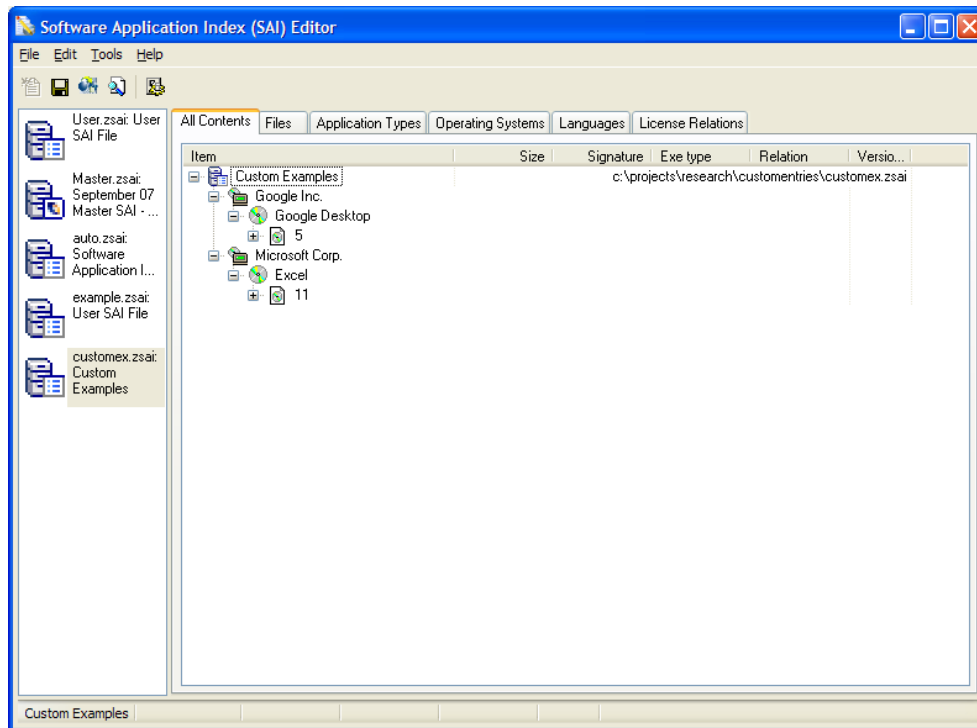
The --help option gives additional details on how to use dstool:

```
/opt/opsware/smtool/dstool --help
```

Custom Signatures

You can add custom application signatures using the DDMI SAI Editor and corresponding platform scanners as shown in [Figure 4](#).

Figure 4 Software Application Index (SAI) Editor



The SAI editor provides a convenient way to add new publishers and applications. Its scanners retrieve raw application signatures related to the specific applications you are interested in reporting as applications to SA.

It is recommended that you run the scanner prior to working with the SAI editor. The entire file system or specific directories can be targeted with the scanner by using the following command line options:

```
scanwin32-x86-2.50.000.7199.exe -fast -p:C:\projects\research\edscan\  
-paths:"C:\Program Files"
```

(scanwin32-x86-2.50.000.7199.exe is the latest version of the executable and may change without notice.)

In the SAI Editor, you can add the publisher, application, release, and version details for the specific applications that need to be categorized in SAI as shown in [Figure 5](#) through [Figure 8](#).

Figure 5 Publisher Properties

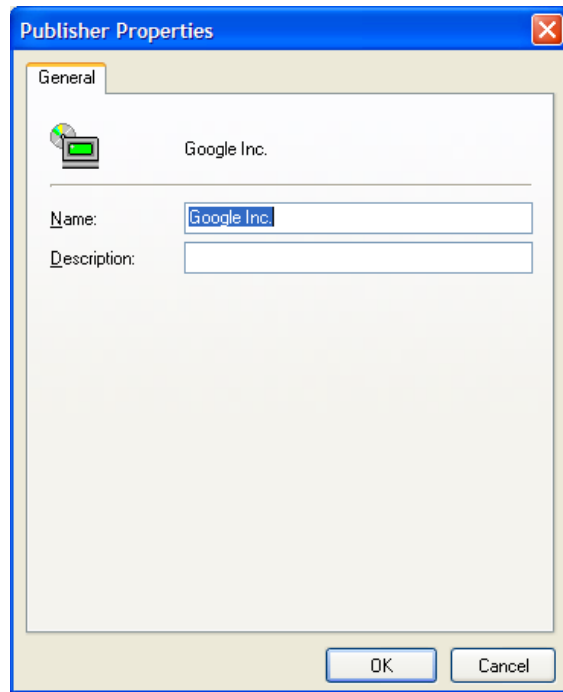


Figure 6 Application Properties

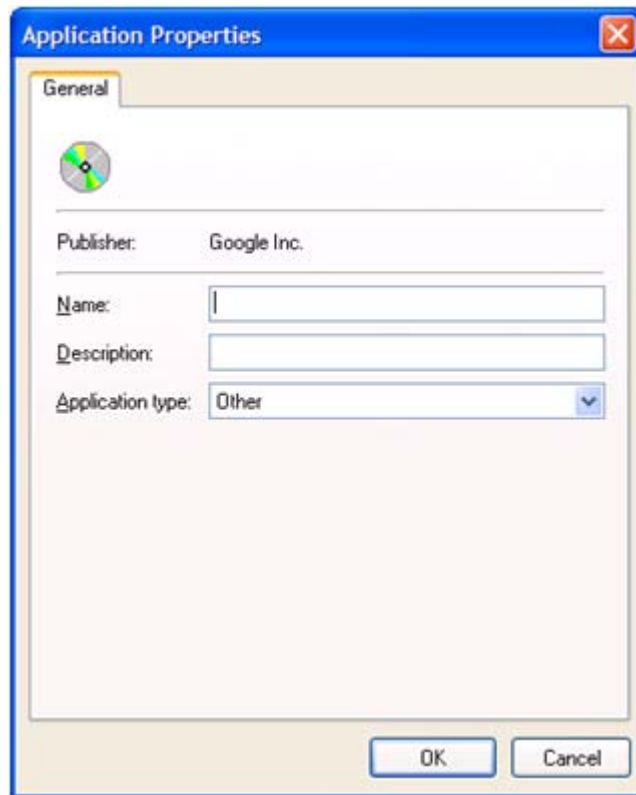
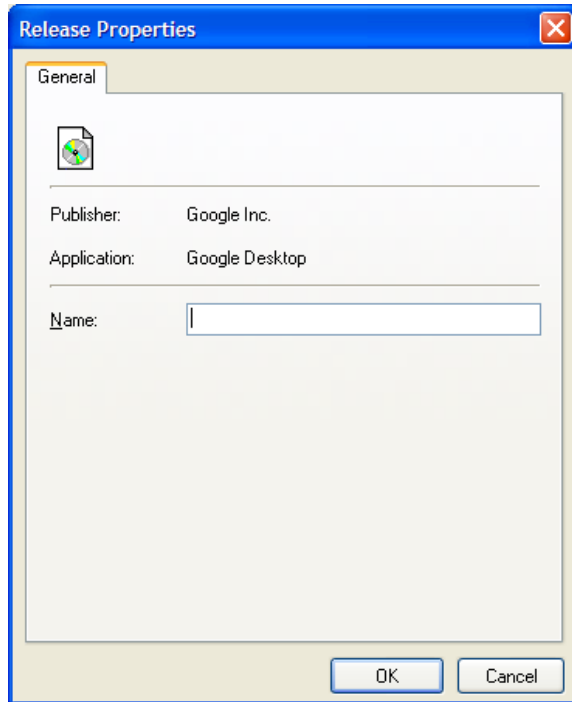
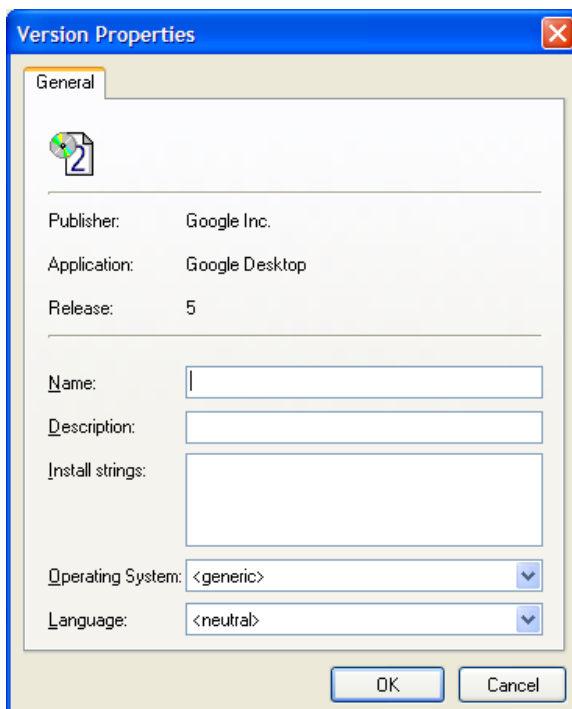


Figure 7 Release Properties



The 'Release Properties' dialog box has a blue title bar with the text 'Release Properties' and a close button. It features a 'General' tab. Below the tab is a CD icon. The form contains the following fields: 'Publisher:' with the value 'Google Inc.', 'Application:' with the value 'Google Desktop', and 'Name:' with an empty text box. At the bottom are 'OK' and 'Cancel' buttons.

Figure 8 Version Properties



The 'Version Properties' dialog box has a blue title bar with the text 'Version Properties' and a close button. It features a 'General' tab. Below the tab is a CD icon with the number '2'. The form contains the following fields: 'Publisher:' with the value 'Google Inc.', 'Application:' with the value 'Google Desktop', and 'Release:' with the value '5'. Below these are 'Name:', 'Description:', and 'Install strings:' text boxes. At the bottom are 'Operating System:' and 'Language:' dropdown menus, both currently set to '<generic>' and '<neutral>' respectively. At the bottom are 'OK' and 'Cancel' buttons.

At this stage, you can open the scan results in the SAI Editor and begin adding corresponding signatures to the versions previously added in the SAI Editor as shown in [Figure 9](#) and [Figure 10](#).

Figure 9 Recognition Verification

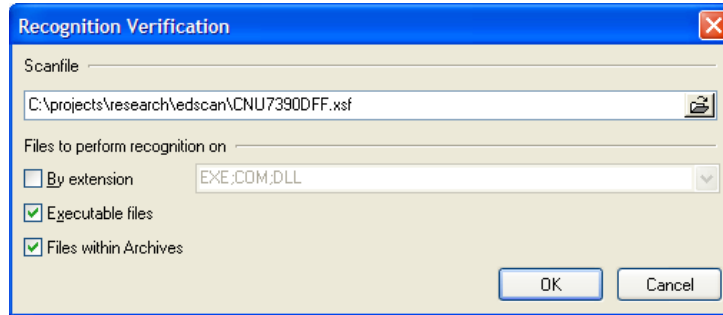
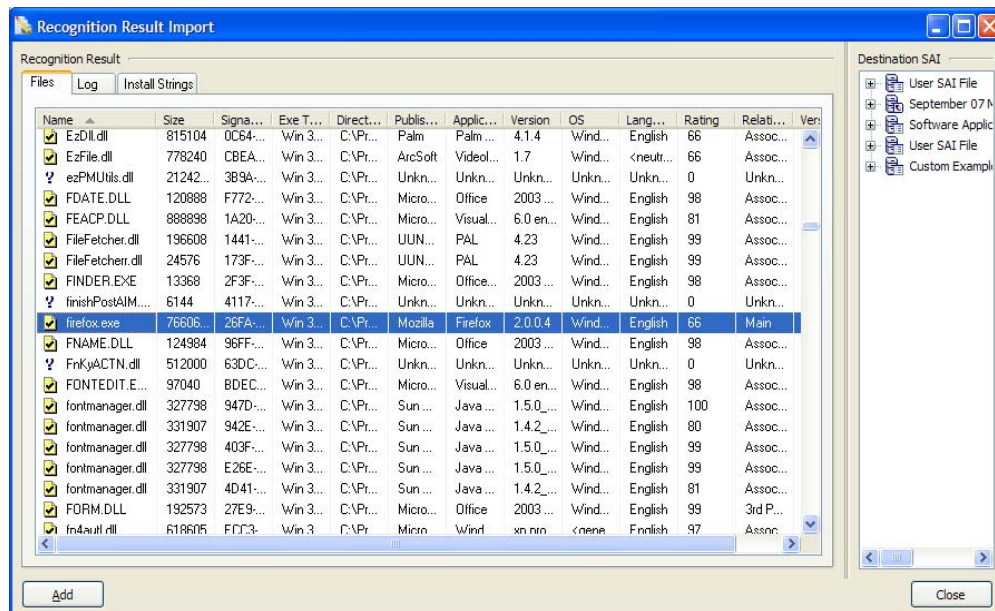
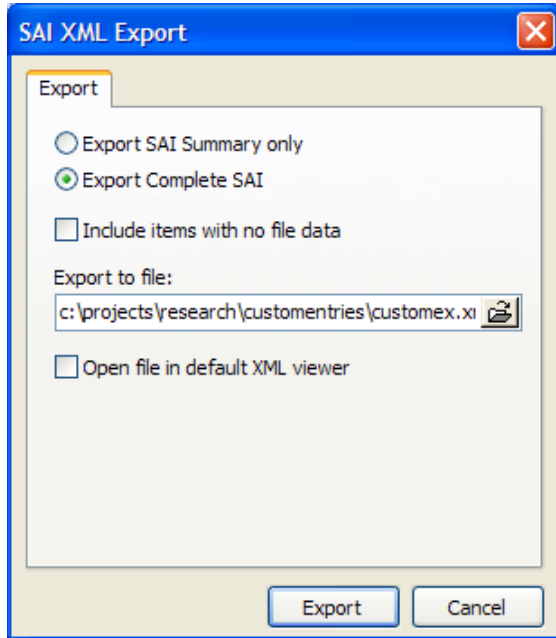


Figure 10 Recognition Result Import



Once the content is modified, you can export the content to the SAI XML database (User XML) as shown in [Figure 11](#).

Figure 11 SAI XML Export



The User XML database is then copied using the following command to the core where it is used by dstool:

```
/opt/opsware/smttool/dstool --username=<username> --password=<password>  
customex.xml
```

dstool generates a new (User) SCT database which is uploaded as a custom dependency to Software Discovery. During the next inventory snapshot, Software Discovery checks for the custom database and uses it along with the production database to categorize new applications.

If at any stage a mistake is made or you choose to remove all custom entries, invoke the dstool with the following command line options:

```
/opt/opsware/smttool/dstool --remove --username=<username>  
--password=<password>
```

Use the following --help option for additional details about the usage of dstool:

```
/opt/opsware/smttool/dstool --help
```

Please refer to the DDMI SAI Editor and scanner documentation for more details.

Configuration and Customization

UNIX and Windows Software Discovery support configuration attributes which drive how they run and the profile of resource usage on a managed server.

Configuration Attributes

The name of the custom attribute is HPSW_DS_SM_CONFIG.

The value of this custom attribute is in INI file format in Windows. Windows filters are grouped under the FILTERS_WINDOWS section while UNIX filters are grouped under the FILTERS_UNIX section. The METHODS section specify the scanning settings and the CONFIGURATION section contains other configuration settings.

The following is the syntax of a custom attribute:

```
[FILTERS_WINDOWS]
FILTER#={INCLUDE | EXCLUDE | EXIST}, { FILENAME | FILEPATH | FILESYSTYPE |
VOLUME | PRODUCT | VENDOR | VERSION}, criteria

[FILTERS_UNIX]
FILTER#={INCLUDE | EXCLUDE | EXIST}, { FILENAME | FILEPATH | FILESYSTYPE |
VOLUME | PRODUCT | VENDOR | VERSION}, criteria

[METHODS]
DELTA_SCAN= {TRUE | FALSE}

[CONFIGURATION]
LOG_MODE= {LOW | MEDIUM | HIGH | DEBUG}
INCLUDE_SYMBOLIC_LINKED_FILES={TRUE | FALSE}
INCLUDE_SYMBOLIC_LINKED_DIRS={TRUE | FALSE}
```

Configuration Options

Table 1 describes the configuration options used within the HPSW_DS_SM_CONFIG custom attribute.

Table 1 Configuration Options

Attribute	Platform	Description
'FILTER#'	All	Create a list of filters for the Software Discovery Gauntlet for Cataloged and/or Uncataloged software. Filters are separated into two categories under INI format according to platform, [FILTERS_WINDOWS] and [FILTERS_UNIX]. Filter arguments are provided below their corresponding platform in the format of FILTER#=Action,Type,Criteria. Action can be one of 'Include', 'Exclude', or 'Exist'. Supported types are 'VOLUME', 'FILEPATH', 'FILENAME', 'PRODUCT', 'VENDOR', 'VERSION', 'FILESYSTYPE'. Criteria must correspond to the type.
'INCLUDE_SYMBOLIC_LINKED_FILES'	UNIX	True/False Default is False. Enable if you wish to include symbolic linked files in the scan.
'INCLUDE_SYMBOLIC_LINKED_DIRS'	UNIX	True/False Default is False. Enable if you wish for tadnsw to traverse into symbolic linked directories.
'DELTA_SCAN'	All	Enabled/Disabled or True/False. Default is False. Enable/Disable the delta scanning module.
'LOG_MODE'	All	Value is one of "LOW", "MEDIUM", "HIGH" or "DEBUG", default is "LOW". Sets the amount of information written to the log file. "LOW" is used to provide general runtime progress. "MEDIUM" will provide additional details regarding progress along with location if performing filescan. If "DEBUG" or "HIGH" is selected, the module will write enough information to the file so developers can diagnose obscure problems.

Syntax Errors

Refer to [Inventory Snapshot Job Error Messages](#) on page 11 to see all the Software Discovery error messages and codes. Prior to using the syntax, the values in the custom attributes in [Table 1](#) are validated for proper syntax. If the configuration values violate the syntax, Software Discovery returns errors. Such messages are reported in place of <error> referenced in the list in [Inventory Snapshot Job Error Messages](#) on page 11. Validation only applies to the INCLUDE/EXCLUDE action assignments. If other invalid configuration options are specified, the default values override

Discovery Filtering Process

The use of Software Discovery filters benefits users by preventing unwanted data from consuming valuable network and database resources. The filtering scheme supports three actions, which are Include, Exist and Exclude.

The Software Discovery feature first checks the Include list to see what files should be examined. If a file is included, it progresses to the next step called Exist, which checks to make sure that selected fields contain certain data. Last, the discovery module compares the scanned data with all required fields against the Exclude list to see if the file should be excluded. If a file passes all configured filters, it is reported, otherwise the file is discarded and the discovery module continues scanning. All criteria fields accept inputs of <criteria>, prefix, postfix, as well as wild cards. If a wild card is specified at the end of the path string, the exact path is matched and sub-directories are not affected.

How Filters Affect Scanning

Include and exclude filters can affect the Software Discovery in a variety of ways. In general, the rules listed below apply.

Directory Scanning:

- If a directory is excluded, exclude all files and subdirectories.
- If a directory is included, include all files and subdirectories.
- If a directory does not match the include set, the status stays neutral and only the selected directories are scanned.
- If no directories are included or excluded, the status is set to include.

File Scanning:

- If a file is excluded, that file is not reported.
- If a file is included, that file is reported.
- If no files are included or excluded, the default status is set to include.

Sample Configuration

The following is a sample of a custom attribute in INI format.

```
[FILTERS_WINDOWS]
FILTER0=INCLUDE, VOLUME, "C: "
FILTER1=EXCLUDE, FILEPATH, "<WINDIR>\INSTALL*"
FILTER2=EXCLUDE, FILEPATH, "<WINSYSDIR>\DLLCACHE\"
```

```

FILTER3=EXCLUDE, FILEPATH, "<WINSYSDIR>\DRIVERS\"
FILTER4=EXCLUDE, FILENAME, "SETUP*"
FILTER5=EXCLUDE, FILEPATH, "\RECYCLE*"
FILTER6=EXCLUDE, FILEPATH, "\SYSTEM VOLUME INFORMATION*"
FILTER7=EXCLUDE, FILEPATH, "**\TEMPORARY INTERNET FILES\*"
FILTER8=EXCLUDE, FILEPATH, "**\LOCAL SETTINGS\TEMP\*"
FILTER9=EXCLUDE, FILEPATH, "**\LOCAL SETTINGS\HISTORY\*"
FILTER10=EXCLUDE, FILEPATH, "**DLLCACHE*"
FILTER11=EXCLUDE, FILEPATH, "**$NTUNINST*"
FILTER12=EXCLUDE, FILEPATH, "**$NTSERVICEPACKUNINSTALL$*"
FILTER13=EXCLUDE, FILEPATH, "\I386"
FILTER14=EXCLUDE, FILEPATH, "**SP4\*"
FILTER15=EXCLUDE, FILEPATH, "**$HF_*"
FILTER16=EXCLUDE, FILEPATH, "**SERVICEPACKFILES*"
FILTER17=EXCLUDE, FILEPATH, "<WINSYSDIR>\WinSxS\"
FILTER18=INCLUDE, FILEPATH, "\DOCUMENTS AND SETTINGS*"
FILTER19=INCLUDE, FILEPATH, "\PROGRAM FILES*"
FILTER20=EXIST, PRODUCT
FILTER21=EXIST, VERSION

[FILTERS_UNIX]
FILTER0=EXCLUDE, FILENAME, "**.dll"
FILTER1=EXCLUDE, FILENAME, "**.com"
FILTER2=EXCLUDE, FILENAME, "**.cmd"
FILTER3=EXCLUDE, FILENAME, "**.html"
FILTER4=INCLUDE, FILEPATH, "/etc*"
FILTER5=INCLUDE, FILEPATH, "/opt*"
FILTER6=INCLUDE, FILEPATH, "/bin*"
FILTER7=INCLUDE, FILEPATH, "/usr/bin*"
FILTER8=INCLUDE, FILEPATH, "/usr/lib*"
FILTER9=INCLUDE, FILEPATH, "/usr/games*"
FILTER10=INCLUDE, FILEPATH, "/usr/sbin*"
FILTER11=EXCLUDE, FILEPATH, "/cygdrive*"
FILTER12=EXCLUDE, FILEPATH, "/lost+found*"
FILTER13=EXCLUDE, FILEPATH, "/proc*"
FILTER14=EXCLUDE, FILEPATH, "/tmp*"

[METHODS]
DELTA_SCAN=TRUE

[CONFIGURATION]
LOG_MODE=LOW
INCLUDE_SYMBOLIC_LINKED_FILES=FALSE
INCLUDE_SYMBOLIC_LINKED_DIRS=FALSE

```

Concurrency and Multi-User Considerations

Software Discovery is configured to look at a Snapshot before attempting to run the inventory Snapshot. If an inventory Snapshot is available, the results are immediately displayed instead of running a new inventory Snapshot. However, if two users initially run an inventory

Snapshot, one of the instances will be accepted and run an inventory Snapshot. The other user will see a message when attempting to view the Discovered Software server module in the Device Explorer alerting them that an inventory Snapshot is already in progress.

If Software Discovery is used during an ad-hoc or scheduled Snapshot job, the data from the Discovered Software server module is persisted as a Snapshot package (snapshot.zip) on the SA core. These zip packages are used, for example, by the SAS Web Client when it needs to perform an Audit operation.

Software Discovery Usage Examples

The Software Discovery discovery mechanism can be useful in case of inherited servers and other usage examples. Some examples are listed below:

Example 1

You are an IT Administrator at a company that has 500 managed servers in your data center. You want to know how many servers have 1.3 Java JRE installed. You create a new Audit and select a source server that you know has Java JRE installed. You perform the following steps:

- 1 From the Discovered Software tab under Rules select and expand Software Node.
- 2 Selects JRE installed on the source server and add an additional regex property check for version='1\.3.*'.
- 3 Form the Targets tab select all the active servers in the lab.
- 4 Save and run the audit to check all the managed servers in the lab for 1.3 versions of Java JRE.

The results come back and you see there are still some servers that are not compliant and proceed to update the remaining servers.

Example 2

For the next task, you want to check all 64-bit machines and see what 32-bit applications were installed. You create a new policy and add your own configuration for Software Discovery using the HPSW_DS_SM_CONFIG custom attribute. In that attribute, you assign the following options:

```
[FILTERS_WINDOWS]
FILTER0=INCLUDE, VOLUME, "C:"
FILTER1=INCLUDE, FILEPATH, "\\PROGRAM FILES (X86)*"
```

```
[METHODS]
DELTA_SCAN=TRUE
```

```
[CONFIGURATION]
LOG_MODE=LOW
```

You attach all the 64-bit Windows 2003 Servers to the new policy and remediate the machines. Next, you create a new snapshot including all 64-bit Windows 2003 Servers and select the wildcard under the Discovered Software tab. You save the snapshot and proceed to

run the snapshot against all the 64-bit Windows 2003 Servers. Results come back and now you have a list of all the 32-bit applications deployed across all the 64-bit Windows 2003 Servers.

Example 3

You are auditing the lab to make sure all Windows servers have the appropriate virus detection software installed. You create a new Audit and select a source server you know has the correct software installed. You navigate to the Discovered Software tab and select Norton Antivirus under the Software node.

You include all the Windows machines in the lab as target machines and then save the Audit. The Audit is then run and the results come back with half the machines in the lab non-compliant. You proceed to install Norton Antivirus on the remainder noncompliant servers.

Extending Software Discovery

While Software Discovery can discover a very large number and a wide variety of applications, you can extend this feature by adding your own Python scripts to perform custom software discovery. To extend Software Discovery, perform the following steps.

- 1 Write your custom software discovery code in Python as described in [Writing Custom Software Discovery Code](#) on page 25.
- 2 Package your Python file into a zip file.
- 3 In the SA Client, select Library in the navigation panel.
- 4 Select the By Folder tab.
- 5 Create the following folder. You will import your zip file and create a software policy in this folder. Make sure you have adequate permission to create this folder. See the *SA Administration Guide* for details on permissions.

```
Library/Opware/Tools/Server Modules/  
com.opware.server.module.discovered_software.ext/
```

To create this folder, you need write permission on the parent folder. For more information, see “Folder Permissions” in the *SA Administration Guide*.

- 6 Import your zip file to this folder. Right click and select **Import Software...** or select the **Actions ► Import Software...** menu item. For details, see “Importing Packages” in the *SA Policy Setter’s Guide*.
- 7 Open the zip package and set the Default Install Path to one of the following.

— For UNIX servers set the default install path to:

```
/opt/opware/sm/com.opware.server.module.discovered_software.ext
```

— For Windows servers set the default install path to:

```
%PROGRAMFILES%\Opware\sm\com.opware.server.module.discovered_software.ext
```

For details, see “Viewing Package Properties” and “Editing Package Properties” in the *SA Policy Setter’s Guide*.

- 8 In the same folder (from [step 5](#) above), create a software policy named `com.opsware.server.module.discovered_software.ext`. Right click and select **New Software Policy...** or select the **Actions ► New Software Policy...** menu item. For details, see “Creating Software Policies” in the *SA Policy Setter’s Guide*.
- 9 Add your imported zip file to the software policy. Open the software policy, select the Policy Items view, and select the “+” button. For details, see “Adding Software Resources to a Software Policy” in the *SA Policy Setter’s Guide*.
- 10 On the core server, run the `smtool` command with the `--upgrade` and `--force_upgrade` options. Specify the user name and password of an SA user with the “Manage Server Module: Read & Write” permission. Specify the Software Discovery zip package that you downloaded from BSA Essentials as described in [Downloading the Software Discovery Package](#) on page 8. For example:

```
smtool --username=<user name> --password=<password> --upgrade  
--force-upgrade OPSWsmo_discovered_software-<version>.zip
```

This command adds your custom software discovery package to the Discovered Software policy. `OPSWsmo_discovered_software-<version>.zip` is the Software Discovery package downloaded from the BSA Essentials Network as described in [Downloading the Software Discovery Package](#) on page 8.

For more information on the `smtool`, run `smtool --help`. The `smtool` command is located on the SA core server in the directory `/opt/opsware/smtool`. For more information on SA permissions, see [Software Discovery Permissions](#) on page 12 and the *SA Administration Guide*.

- 11 Run your custom software discovery code as described in [Deploying Software Discovery Using Inventory Snapshots](#) on page 10.

Writing Custom Software Discovery Code

This section describes how to write custom Python code to discover software that is not already discoverable by the SA Software Discovery feature. For instructions on how to integrate your custom code into SA, see [Extending Software Discovery](#) on page 24.

- The Python script must contain a `discover()` function that takes a list argument named `objects`, and returns a tuple of (`objects`, `merge_flag`) where `objects` is a list and `merge_flag` is an integer.
- The `objects` argument contains all the software objects Software Discovery has found. The returned list contains either the list of objects passed into the `discover()` function merged with the software objects discovered by your extension, or only the software objects discovered by your extension, depending on the integer value returned in the second parameter. See below for an example.

- The integer value, `merge_flag`, returned by the `discover()` function must be 0, 1 or 2 as described in the following table.

Table 2 Integer Value Returned by the `discover()` Function

Value Returned	Meaning
0	Your extension has merged the software objects passed in to the <code>discover()</code> function with the software objects it has found and returns the union of these two sets.
1	Your extension returns only the software objects it has found. The Software Discovery server module will merge the returned objects with its list of software objects. If there is a conflict, the software objects discovered by your extension will take precedence.
2	Your extension returns only the software objects it has found. The Software Discovery server module will merge the returned objects with its list of software objects. If there is a conflict, the software objects discovered by the Software Discovery server module will take precedence.

- The file name of your Python script must be in the following format:

`ds_ext_<unique name>.py`.

The `<unique name>` must be unique in all extensions in both the HP Software Discovery directory and the software discovery extensions directory. As a best practice, use your company name in the `<unique name>` to avoid conflicts. For example, the following could be a script from the “abc company”: `ds_ext_abcco_app.py`.

- The `discover()` function will be called after the Software Discovery server module runs and before it returns the result to the caller.
- Below is a sample implementation of the `discover()` function that returns a list containing one simulated software object:

```
def discover(objects):
    obj = {
        'opswType': 'software',
        'FILENAME': '/usr/local/bin/',
        'FILEPATH': 'my_util',
        'primaryKey': '/usr/local/bin/my_util',
        'VERSION': '1.3.0.1',
        'DISPLAY_VERSION': '1.3',
        'PRODUCT': 'My Util',
        'DISPLAY_PRODUCT': 'My Util',
        'VENDOR': 'abc co',
        'DISPLAY_VENDOR': 'abc co',
        'VOLUME': '',
        'FILESIZE': 32656,
        'FILEVER': '1.3.0.1',
        'FILEDATE': '2008-12-02 01:03:38'
    }
    return ([obj], 1)
```

- After writing your software discovery code, integrate it into SA as described in [Extending Software Discovery](#) on page 24.

