



Micro Focus Security ArcSight Connectors

SmartConnector for Oracle Solaris Basic Security Module

Configuration Guide

June, 2018

Configuration Guide

SmartConnector for Oracle Solaris Basic Security Module

June, 2018

Copyright © 2005 – 2017; 2018 Micro Focus and its affiliates and licensors.

Warranty

The only warranties for products and services of Micro Focus and its affiliates and licensors ("Micro Focus") are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus 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. Except as specifically indicated otherwise, a valid license from Micro Focus is 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.

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated. Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation. UNIX® is a registered trademark of The Open Group.

Revision History

Date	Description
10/17/2017	Added encryption parameters to Global Parameters.
11/30/2016	Updated installation procedure for setting preferred IP address mode.
03/31/2014	Added support for sudo with Solaris SPARC and x86. Removed support for Solaris version 8 and 9.
02/14/2014	Added GA support for Solaris 11 for SPARC.
11/15/2013	Added GA support for Solaris 11 x86.
09/30/2013	Changed Device Vendor mapping to 'Oracle'.
05/15/2012	Added new installation procedure. Added caveat to "BSM Caveats."
11/15/2011	Added Device Custom String 2 mapping for Solaris BSM 10.
09/30/2011	Added mapping for File Name to BSM 10 mappings.

SmartConnector for Oracle Solaris Basic Security Module

This guide provides information for installing the SmartConnector for Oracle Solaris Basic Security Module on a Solaris platform and configuring the device for audit log event collection. Event collection from Solaris SPARC versions 10, 11, and x86 version 11 is supported.



Solaris versions 8 and 9 are no longer supported for SmartConnector installation and have been removed from connector configuration selections. To continue running these versions with the SmartConnector, do not upgrade the connector. To upgrade, you must be using Solaris version 10 or later.

Product Overview

The Oracle Solaris Basic Security Module (BSM) provides a security auditing subsystem. The auditing mechanism lets administrators detect potential security breaches. It performs kernel auditing and provides a device allocation mechanism for the Solaris operating system, which enable Solaris to meet C2 level criteria.



C2 is a security rating originally defined in the Trusted Computer System Evaluation Criteria (TCSEC), published by the United States National Computer Security Center (NCSC), commonly referred to as the Orange Book.

The BSM audit trail is written to binary files on the local system (or NFS mount). Audit records are initiated from two distinct places in Solaris-privileged user land programs (such as login) and the Solaris kernel. All security-sensitive kernel system calls generate an audit record when BSM auditing is enabled.



Reading or executing privileged audit files requires administrator access.

BSM is not enabled by default under Solaris. The administrator is required to run the `bsmconv` script to set up the initial auditing environment for the system. See "Basic Configuration" later in this guide.

BSM Auditing

For complete information about BSM auditing, see the *Oracle Solaris Basic Security Module Guide*. Additional helpful information includes "Solaris BSM Auditing" by Hal Pomeranz of Deer Run Associates (<http://www.deer-run.com/~hal/sysadmin/SolarisBSMAuditing.html>).

Auditing can reveal suspicious or abnormal patterns of system usage and provide the means to trace suspect actions back to a specific user.

Successful auditing depends upon two other security features: identification and authentication. At login, after the user supplies a user name and password, a unique audit ID is associated with the user's process. The audit ID is inherited by every process started during the login session. Even if a user changes identity, all actions performed are tracked with the same audit ID.

After audit data is collected, audit reduction and interpretation tools allow the examination of interesting parts of the audit trail. For example, you can look at audit records for individual users or groups, look at

all records for a certain type of event on a specific day, or select records that were generated at a certain time of day.

Audit Events

System actions that are auditable are defined as audit events in the `/etc/security/audit_event` file. Each auditable event is defined in the file by a symbolic name, an event number, a set of pre-selection classes, and a short description. Most events are attributable to an individual user. However, some events are non-attributable because they occur at the kernel-interrupt level or before a user is identified and authenticated. Non-attributable events are auditable as well.

Each audit event is also defined as belonging to an audit class or classes. By assigning events into classes, an administrator can more easily deal with large numbers of events. When naming a class, you simultaneously address all of the events in that class. The mapping of audit events to classes is configurable and the classes themselves are configurable. These configuration changes can be made in the `audit_event` file.

Whether an auditable event is recorded in the audit trail depends upon whether the administrator pre-selects a class for auditing that includes the specific event. Out of 32 possible audit classes, 18 are defined. The 18 classes include the two global classes `all` and `no`.

Audit Records

Each audit record describes the occurrence of a single audited event and includes such information as who performed the action, which files were affected, what action was attempted, and where and when the action occurred.

Audit records are collected in a trail and can be converted to human-readable format by using `praudit` (see the `praudit(1M)` man page).

Audit Flags

Audit flags indicate classes of events to audit. Machine-wide defaults for auditing are specified for all users on each machine by flags in the `audit_control` file. The system administrator can modify what is audited for individual users by putting audit flags in a user's entry in the `audit_user` file. The audit flags also are used as arguments to `auditconfig` (see the `auditconfig(1M)` man page).

The Audit Trail

The audit trail is created by the audit daemon (see the `auditd(1M)` man page). The audit daemon starts on each machine when the machine is brought up. After `auditd` starts at boot time, it collects the audit trail data and writes the audit records into audit files.

The audit daemon runs as `root`. All files it creates are owned by `root`. Even when `auditd` has no classes to audit, it continuously operates, looking for a place to put audit records. The `auditd` operations continue even if the rest of the machine's activities are suspended due to the kernel's audit buffers becoming full. The audit operations can continue because `auditd` is not audited.

Only one audit daemon can run at a time. An attempt to start a second one results in an error message and the new one exits. If there is a problem with the audit daemon, try using `audit -t` to terminate `auditd` gracefully, then restart it manually.

The `audit_warn` script is run by `auditd` whenever the daemon switches audit directories or encounters difficulty (such as a lack of storage). As distributed, the `audit_warn` script sends mail to an `audit_warn` alias and sends a message to the console. Your site should customize `audit_warn` to suit your needs.

When `auditd` starts on each machine, it creates the file `/etc/security/audit_data`.

To keep audit files at a manageable size, a cron job can be set up that periodically switches audit files (see the `cron(1M)` man page). Intervals range from once per hour to twice per day, depending upon the amount of audit data being collected. The data then can be filtered to remove unnecessary information and then compressed.

The auditreduce Command

Use `auditreduce` to merge together audit records from one or more input audit files or to perform a post selection of audit records. See the `auditreduce(1M)` man page. To merge the entire audit trail, the system administrator enters the command on the machine on which all the audit file systems for the distributed system are mounted.

When multiple machines running BSM are administered as part of a distributed system, each machine performs auditable events, and each machine writes audit records to its own machine-specific audit file. Using `auditreduce`, you can read the logical combination of all audit files in the system as a single audit trail without regard to how the records were generated or where they are stored.

`auditreduce` selects records from one or more audit files and merges them into a single, chronologically ordered output file. The merging and selecting functions of `auditreduce` are logically independent. `auditreduce` selects messages from the input files as the records are read, before the files are merged and written to disk.

Without options, `auditreduce` merges the entire audit trail (which consists of all of the audit files in all of the sub-directories in the audit root directory `/etc/security`) and sends all the audit records to standard output. The location of this directory will be required during SmartConnector installation.

The SmartConnector accesses the directory you specify and invokes the `praudit` command to generate the ASCII output the connector uses to map event data.

Overview of Audit Setup

The following steps are included here to provide an overview of what is required to set up audit directories and specify which audit classes will be audited.

- 1 Format and partition the disks to create the dedicated audit partition or partitions. A rule of thumb is to assign 100 MB of space for each machine that will be on the distributed system; however, the disk space requirements at your site will be based upon how much auditing you perform and may be far greater than this figure per machine.
- 2 Assign the audit file systems to the dedicated partitions. Each disk full machine should have a backup audit directory on the local machine in case its NFS-mounted audit file system or file systems are not available.
- 3 While each machine is in single-user mode, run `tunefs -m 0` on each dedicated audit partition to reduce reserved file system space to 0%.

A reserved space percentage (called the `minfree` limit) is specified for audit partitions in the `audit_control` file. The default is 20%, and this percentage is tunable. Because this value is set by each site in the `audit_control` file, you should remove the automatically reserved file system space that is set aside by default for all file systems.

- 4 Set the required permissions on each of the audit directories on the audit server and make a subdirectory in each audit directory called **files**. Use `chown` and `chmod` to assign the required permissions to each audit directory and to each files subdirectory.
- 5 If using audit servers, export the audit directories using the `dfstab(4)` file.
- 6 Create the `audit_control` file entries for all the audit directories in the `audit_control` file on each machine, specifying the `files` subdirectory.
- 7 On each audit client, create the entries for the audit file systems in the `vfstab(4)` files.
- 8 On each audit client, create the mount point directories and use `chmod` and `chown` to set the correct permissions.

The following table summarizes the commands to use to configure auditing.

Utility	Task
<code>allocate(1M)</code>	Allocate a device
<code>audit(1M)</code>	Control the audit daemon
<code>audit_startup(1M)</code>	Initialize the audit subsystem
<code>audit_warn(1M)</code>	Run the audit daemon warning script
<code>auditconfig(1M)</code>	Configure auditing
<code>auditd(1M)</code>	Control audit trail files
<code>auditreduce(1M)</code>	Merge and select audit records from audit trail files
<code>auditstat(1M)</code>	Display kernel audit statistics
<code>bsmconv(1M)</code>	Enable a Solaris system to use the Basic Security Module
<code>bsmunconv(1M)</code>	Disable the Basic Security Module and return to Solaris
<code>deallocate(1M)</code>	Deallocate a device
<code>auditor(2)</code>	Manipulate auditing
<code>auditsvc(2)</code>	Write audit log to specified file descriptor
<code>sudo(1M)</code>	Generate audit log files

Basic Configuration Steps

- 1 Enable BSM and ensure `auditd` is started at boot time.
 - A Run `/etc/security/bsmconv` as `root` to enable auditing. Auditing is not enabled by default. See "Enabling BSM" for more detailed information.
 - B Set up the `/etc/security/audit_control` file to indicate the type of auditing to be performed. See "audit_control" for more information.
 - C Reboot the system so the `c2audit` module is properly loaded and the internal audit settings are configured.

- 2 Set up the classes of events for which you want to generate audit records and where those records are to go. These are defined in `/etc/security/audit_control`. See "Audit Class and Audit Event" for more information.

For example, to record the login events for all users, add the class `lo` to the flags: line of `/etc/security/audit_control`. The `dir:` line specifies the directory into which audit records are to be written. This is the directory name you should enter for the **praudit Output File** parameter during SmartConnector installation.



The default path for `praudit` is `/usr/sbin`. If you use another path for `praudit`, be sure to add the location to the system `PATH` variable.

```
dir: /var/audit
flags:  lo
minfree:  20
naflags:  lo
```

Including `lo` on the flags: line logs events regardless of whether it was a success or failure; to log only failures, put a hyphen (-) in front of the class name.

Enable BSM Auditing in Solaris 10



Enabling BSM on a server automatically enables the SM features on all of that server's clients.

- 1 After becoming `root`, bring the system into the single-user mode:

```
# /etc/telinit 1
```

- 2 In single-user mode, change directories to the `/etc/security` directory and execute the `bsmconv` script located there. The script sets up a standard Solaris machine to run BSM after a reboot.

```
# cd /etc/security
# ./bsmconv
```

- 3 After the script finishes, halt the system with the `telinit` command. Then reboot the system to bring it up as a multi-user BSM system.

```
# /etc/telinit 6
```



Whenever you need to restart the BSM service, restart through a server reboot.

Enable BSM Auditing in Solaris 11

Auditing is enabled by default on Solaris 11, but only user login/logout events are monitored by default. For monitoring both the OS File change events and OS USER logins/logout events, you can execute the following command with root privilege:

```
# /usr/sbin/auditconfig -setflags fw,fd,fc,fm,fr,lo
```



The `bsmconv` command has been removed on Solaris 11. Use the following command to enable the auditing feature, if needed: `audit -s`

Set Up Classes and Events

The `bsmconv` script creates a number of files in the `/etc/security` directory, including:

- The `audit_startup` script is invoked at boot time and sets a number of different audit policies for the system.
- The `audit_control` file is the primary configuration file for BSM.
- The `audit_class` and `audit_event` files can be used when more fine-grained control of the audit configuration is required.

The following sections describe the `audit_startup` and `audit_control` files, audit classes and events, and custom audit classes you may access when setting up auditing.

Audit Startup

The existence of a file with the path name `/etc/security/audit_startup` causes the audit daemon to be run automatically when the system enters multi-user mode. A default `audit_startup` script that automatically configures the event to class mappings and sets the audit policies is set up during the BSM package installation.

The `audit_startup` script is a series of `auditconfig` commands for initializing the system auditing policy:

```
#!/bin/sh
/usr/sbin/auditconfig -conf
/usr/sbin/auditconfig -aconf
/usr/sbin/auditconfig -setpolicy none
/usr/sbin/auditconfig -setpolicy +cnt
/usr/sbin/auditconfig -setpolicy +argv,arge
```

The first two lines pull configuration information out of the `audit_control` file and set up the basic events the system will audit. The remaining lines set other special auditing policy options:

`-setpolicy none`

Blanks the audit policy for the system to start with a clean slate

`setpolicy +cnt`

Tells the system to continue running even if the auditing partition on the machine fills up (high security sites are required to have the machine shut down if auditing becomes impossible)

`-setpolicy -cnt` and `-setpolicy +argv,arge`

Means to track the full command line and all environment settings for any command executed on the system. Note that the `-setpolicy +argv,arge` line is not part of the default BSM configuration set up by the `bsmconv` script.

Audit Control

The `audit_control` file appears simple:

```
dir:/var/audit
minfree:20
flags:lo,ad,pc,fm,fw,-fc,-fd,-fr
naflags:lo,ad,ex
```

`dir`

is the directory into which audit logs will be written on the system (this directory should only be accessible by the superuser). (Note that this is the directory name required during SmartConnector installation.) There is no built-in facility for writing audit logs to some other system, although some sites have attempted writing to an NFS-mounted directory from some central file server (note that this configuration requires the client system to have root write privileges into the NFS volume, which has some significant security implications).

`minfree`

Specifies the amount of free space, as a percentage, that must exist in the auditing partition; otherwise the system starts complaining. So, with `minfree` set at 20, once the audit partition goes above 80% full, the auditing subsystem starts sending the administrator warning messages.

`flags` and `naflags`

Define to which audit events the system actually is going to pay attention (these are the lines at which the `auditconfig -conf` and `auditconfig -aconf` commands in `audit_startup` are looking). The two letter codes are groups (audit classes) of related events (system calls) defined through the `audit_class` and `audit_events` files.

The `flags` line defines the audit vector for normal user sessions on the machine. The `naflags` line catches all events that are not associated with a particular user's session. Usually, these events are the result of system processes and do not occur often.

Audit Class and Audit Event

In tuning BSM auditing, you should strike a balance between getting the events you need to reconstruct what has been happening on the system while filtering out uninteresting events that add "noise" to the audit trail and consume huge amounts of disk space.

A recommended minimum set of classes is `lo`, `ad`, and `na`. These include login/out events (`lo`), admin events (`ad`), such as file system mounts and creation of users, and non-attributable (`na`) events.

General Events - `lo` (login) and `ad` (administrative)

The `lo` (login) class covers all forms of system logins as well as use of the `su` command.

The `ad` (administrative) class covers a wide variety of administrative actions, including rebooting the system, adding and deleting users, changing auditing and logging parameters, mounting and dismounting both local and remote file systems, changing quotas, loading kernel modules, and even setting the system clock.

Process Events - `ex` (execution) and `pc` (process control)

The `ex` and `pc` classes deal with process execution on the system.

The two events in the `ex` class (`exec()` and `execve()` system calls) are used to execute programs on the system. These events also are contained in the `pc` class, so if your audit vector includes `pc`, you need not worry about `ex`.

The `pc` class also tracks everything that the process might do during its lifetime, such as changing directories, calling `setuid()` and `setgid()` to change its privilege level, making `chroot()` calls, creating child processes with `fork()` and `vfork()`, and so on. The `pc` class also tracks administrative interaction with processes on the system, such as `kill` and `nice`.

`pc` tracks various system calls that usually are not interesting. For example, you probably do not need to know every time your mail server forks a new child process to deal with an incoming connection. What you really want to know is when new processes get started on the system, typically with a `fork()` followed by an `exec()`. So you really want to track just the `exec()`s. To track the important events from the `pc` class but ignore the uninteresting ones requires creating a new custom class that includes just the events you want (see "Custom Audit Classes").

File Attribute Modification - fm Class

The `fm` class tracks changes to file attributes such as ownership (`chown`) and permissions (`chmod`), and even extended file ACL settings. However, `fm` also tracks file locking and updating timestamps on files. These latter events are too frequent on normal UNIX systems to be useful. To track the important events from the `fm` class but ignore the uninteresting ones requires creating a new custom class that includes just the events you want (see "Custom Audit Classes").

Other File Actions - fc (create), fr (read), fw (write), fd (delete)

Oracle recommends avoiding these audit classes in order to reduce the size of the audit trail. However, the DoD guidelines require tracking at least failure for these classes (the specific recommendation is `fw`, `-fc`, `-fd`, `-fr`). These classes really can generate an enormous number of audit events and consume huge amounts of disk space. The default recommendation is to not turn on any auditing of these classes.

Custom Audit Classes

Audit class names are defined in the `audit_class` file. The following are the audit class entries for the classes discussed thus far:

```
0x00000001:fr:file read
0x00000002:fw:file write
0x00000008:fm:file attribute modify
0x00000010:fc:file create
0x00000020:fd:file delete
0x00000080:pc:process
0x00000800:ad:administrative
0x00001000:lo:login or logout
0x40000000:ex:exec
0xffffffff:all:all classes
```

The first field of each line is a unique bit mask used to represent the audit class in the internals of the auditing subsystem. The second field is the class code used in the `flags` and `naflags` lines in `audit_control`, and the third field is a brief descriptive name for the use of the system administrator.

When creating a custom class, pick a bit mask and a two-letter code that are not currently in use by any other class. In the default `audit_class` file installed by `bsmconv`, bit masks from 0x00010000 through 0x08000000 are not used. The following example uses `cc` (custom class) with a bit mask of 0x08000000:

```
0x08000000:cc:CIS custom class
```

Audit Log File Rotation

Audit logs are written to binary files in your audit directory. The file naming convention used is `<start>.<end>.<hostname>`, where `<start>` and `<end>` are time/date stamps in the format `YYYYMMDDhhmmss` and `<hostname>` is the fully-qualified hostname of the local machine. The current audit log that is actively being written is named `<start>.not_terminated.<hostname>` to distinguish it from the other audit logs in the directory.

The command `audit -n` signals the system audit daemon to close its current audit log file and start a new one. Unless told otherwise, the audit daemon will simply continue writing to the current audit log and it will grow without bound until it reaches the file size limit for the machine or fills the partition. To force audit logs to be restarted at the top of every hour:

```
0 8 8 8 8 /usr/sbin/audit -n
```

Once the new audit log has been started, the old log can be compressed or moved off of the local system for archival.

BSM Caveats

- If the connector's event log file tailing process executes more slowly than the BSM's event log generation process, and if a newly rotated log file is detected while the current log file is still being tailed, the current tail process is terminated without having a chance to complete reading the current log file or the subsequent log files. The connector starts tailing the newly rotated log file, leading to an event loss.
- Enabling BSM automatically disables the `<Stop>-A` keyboard sequence on the machine. This occurs to be able to monitor shutdown and reboot events and associate them with a particular user. Disabling `<Stop>-A` means somebody has to log in, become `root`, and halt the machine, all of which are auditable events.
- Enabling BSM disables auto-mounting of CD-ROMs and floppies using `vold`. Again, there is an audit trail issue if a system process spontaneously mounts and dismounts file systems.
- There are known interoperability problems between OpenSSH (particularly with PrivSep enabled) and BSM. The most noticeable issue is that OpenSSH sessions will not appear in the audit logs at all. A patch[4] is available to fix this and some other issues.

Install the SmartConnector

The following sections provide instructions for installing and configuring your selected SmartConnector.

Prepare to Install Connector

Before you install any SmartConnectors, make sure that the ArcSight products with which the connectors will communicate have already been installed correctly (such as ArcSight ESM or ArcSight Logger).

For complete product information, read the *Administrator's Guide* as well as the *Installation and Configuration* guide for your ArcSight product before installing a new SmartConnector. If you are adding a connector to the ArcSight Management Center, see the *ArcSight Management Center Administrator's Guide* for instructions, and start the installation procedure at "Set Global Parameters (optional)" or "Select Connector and Add Parameter Information."

Before installing the SmartConnector, be sure the following are available:

- Local access to the machine where the SmartConnector is to be installed
- Administrator passwords

Install Core Software

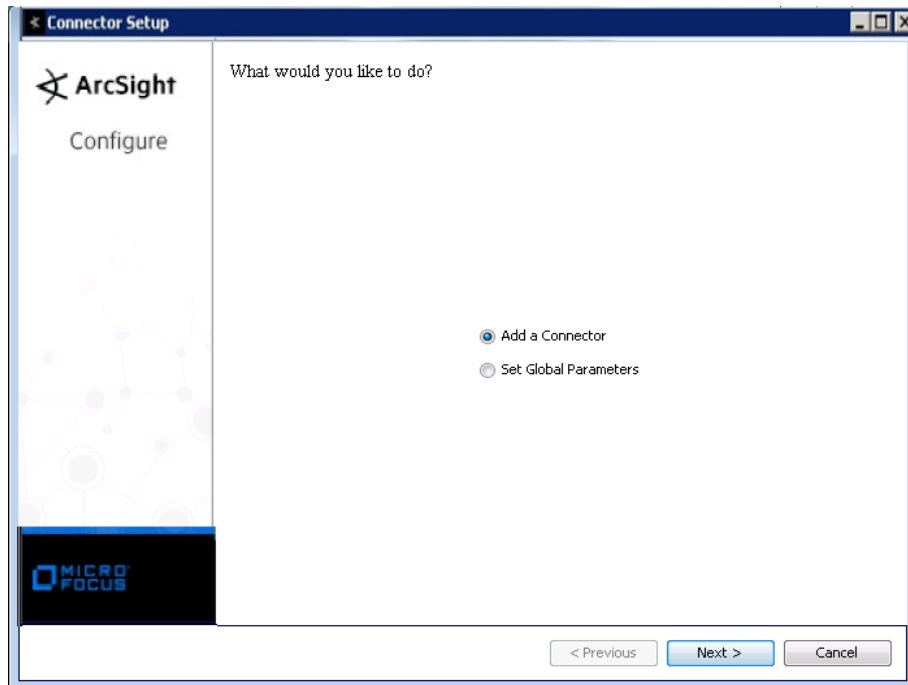
Unless specified otherwise at the beginning of this guide, this SmartConnector can be installed on all ArcSight supported platforms; for the complete list, see the *SmartConnector Product and Platform Support* document, available from the Micro Focus SSO and Protect 724 sites.

- 1 Download the SmartConnector executable for your operating system from the Micro Focus SSO site.
- 2 Start the SmartConnector installation and configuration wizard by running the executable.

Follow the wizard through the following folder selection tasks and installation of the core connector software:

Introduction
Choose Install Folder
Choose Shortcut Folder
Pre-Installation Summary
Installing...

- 3 When the installation of SmartConnector core component software is finished, the following window is displayed:



Set Global Parameters (optional)

If you choose to perform any of the operations shown in the following table, do so before adding your connector. You can set the following parameters:

Parameter	Setting
FIPS mode	Select 'Enabled' to enable FIPS compliant mode. To enable FIPS Suite B Mode, see the SmartConnector User Guide under "Modifying Connector Parameters" for instructions. Initially, this value is set to 'Disabled'.
Remote Management	Select 'Enabled' to enable remote management from ArcSight Management Center. When queried by the remote management device, the values you specify here for enabling remote management and the port number will be used. Initially, this value is set to 'Disabled'.
Remote Management Listener Port	The remote management device will listen to the port specified in this field. The default port number is 9001.
Preferred IP Version	When both IPv4 and IPv6 IP addresses are available for the local host (the machine on which the connector is installed), you can choose which version is preferred. Otherwise, you will see only one selection. The initial setting is IPv4.

The following parameters should be configured only if you are using Micro Focus SecureData solutions to provide encryption. See the *Micro Focus SecureData Architecture Guide* for more information.

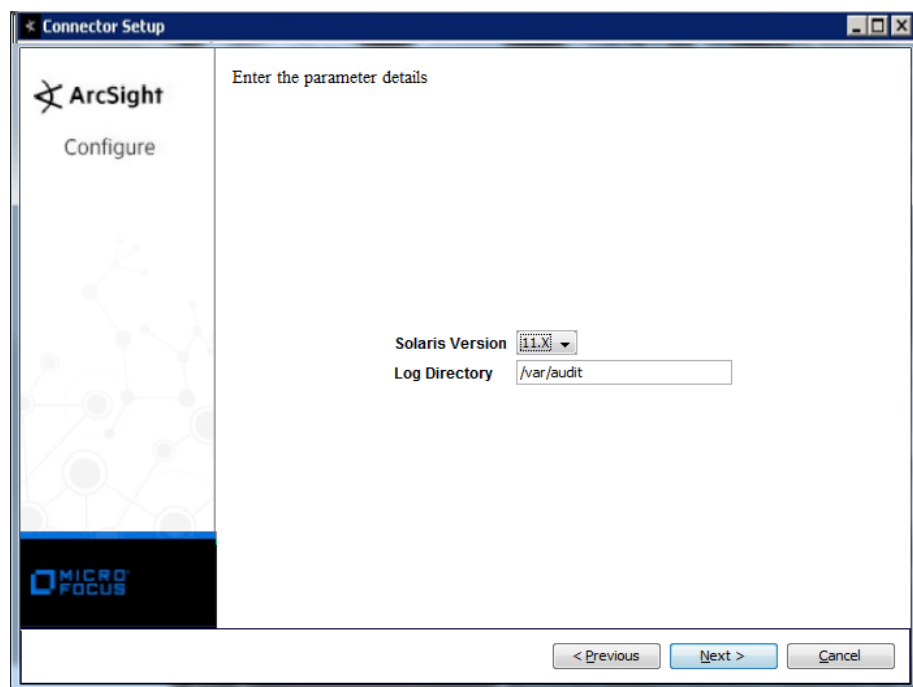
Parameter	Setting
Format Preserving Encryption	Data leaving the connector machine to a specified destination can be encrypted by selecting 'Enabled' to encrypt the fields identified in 'Event Fields to Encrypt' before forwarding events. If encryption is enabled, it cannot be disabled. Changing any of the encryption parameters again will require a fresh installation of the connector.
Format Preserving Policy URL	Enter the URL where the Micro Focus SecureData Server is installed.
Proxy Server (https)	Enter the proxy host for https connection if any proxy is enabled for this machine.
Proxy Port	Enter the proxy port for https connection if any proxy is enabled for this machine.

Parameter	Setting
Format Preserving Identity	The Micro Focus SecureData client software allows client applications to protect and access data based on key names. This key name is referred to as the identity. Enter the user identity configured for Micro Focus SecureData.
Format Preserving Secret	Enter the secret configured for Micro Focus SecureData to use for encryption.
Event Fields to Encrypt	Recommended fields for encryption are listed; delete any fields you do not want encrypted and add any string or numeric fields you want encrypted. Encrypting more fields can affect performance, with 20 fields being the maximum recommended. Also, because encryption changes the value, rules or categorization could also be affected. Once encryption is enabled, the list of event fields cannot be edited.

After making your selections, click **Next**. A summary screen is displayed. Review the summary of your selections and click **Next**. Click **Continue** to return to proceed with "Add a Connector" window. Continue the installation procedure with "Select Connector and Add Parameter Information."

Select Connector and Add Parameter Information

- 1 Select **Add a Connector** and click **Next**. If applicable, you can enable FIPS mode and enable remote management later in the wizard after SmartConnector configuration.
- 2 Select **Oracle Solaris Basic Security Module** and click **Next**.
- 3 Enter the required SmartConnector parameters to configure the SmartConnector, then click **Next**.



Parameter	Description
Solaris Version	Select your Solaris version: 10.x or 11.x

Parameter	Description
Log Directory	Enter the absolute path to the directory containing the log files. The default value is /var/audit.

Select a Destination

- 1 The next window asks for the destination type; select a destination and click **Next**. For information about the destinations listed, see the *ArcSight SmartConnector User Guide*.
- 2 Enter values for the destination. For the ArcSight Manager destination, the values you enter for **User** and **Password** should be the same ArcSight user name and password you created during the ArcSight Manager installation. Click **Next**.
- 3 Enter a name for the SmartConnector and provide other information identifying the connector's use in your environment. Click **Next**. The connector starts the registration process.
- 4 If you have selected ArcSight Manager as the destination, the certificate import window for the ArcSight Manager is displayed. Select **Import the certificate to the connector from destination** and click **Next**. (If you select **Do not import the certificate to connector from destination**, the connector installation will end.) The certificate is imported and the **Add connector Summary** window is displayed.

Complete Installation and Configuration

- 1 Review the **Add Connector Summary** and click **Next**. If the summary is incorrect, click **Previous** to make changes.
- 2 The wizard now prompts you to choose whether you want to run the SmartConnector as a stand-alone process or as a service. If you choose to run the connector as a stand-alone process, select **Leave as a standalone application**, click **Next**, and continue with step 5.
- 3 If you chose to run the connector as a service, with **Install as a service** selected, click **Next**. The wizard prompts you to define service parameters. Enter values for **Service Internal Name** and **Service Display Name** and select **Yes** or **No** for **Start the service automatically**. The **Install Service Summary** window is displayed when you click **Next**.
- 4 Click **Next** on the summary window.
- 5 To complete the installation, choose **Exit** and Click **Next**.

For instructions about upgrading the connector or modifying parameters, see the *SmartConnector User Guide*.

Run the SmartConnector

SmartConnectors can be installed and run in stand-alone mode, on Windows platforms as a Windows service, or on UNIX platforms as a UNIX daemon, depending upon the platform supported. On Windows platforms, SmartConnectors also can be run using shortcuts and optional Start menu entries.

If the connector is installed in stand-alone mode, it must be started manually and is not automatically active when a host is restarted. If installed as a service or daemon, the connector runs automatically

when the host is restarted. For information about connectors running as services or daemons, see the *ArcSight SmartConnector User Guide*.

To run all SmartConnectors installed in stand-alone mode on a particular host, open a command window, go to `$ARCSIGHT_HOME\current\bin` and run: `arcsight connectors`

To view the SmartConnector log, read the file `$ARCSIGHT_HOME\current\logs\agent.log`; to stop all SmartConnectors, enter `Ctrl+C` in the command window.

Device Event Mapping to ArcSight Fields

The following section lists the mappings of ArcSight data fields to the device's specific event definitions. See the *ArcSight Console User's Guide* for more information about the ArcSight data fields.

Oracle Solaris 10 and 11 BSM Common Mappings to ArcSight ESM Fields

ArcSight ESM Field	Device-Specific Field
Destination Host Name	Host
Destination User Name	subject-audit-uid
Device Action	return-errval
Device Custom Number 1	subject-sid
Device Custom Number 1 Label	'Session ID'
Device Custom String 2	exec_args
Device Custom String 2 Label	'exec_args'
Device Custom String 3	subject-tid-host
Device Custom String 3 Label	'Terminal Host'
Device Custom String 4	One of (return-retval, return-errval-reason)
Device Custom String 4 Label	'Reason or Error Code'
Device Custom String 5	subject-rgid or subject-gid
Device Custom String 5 Label	'Source User Group'
Device Custom String 6	subject-rgid
Device Custom String 6 Label	'Destination User Group'
Device Event Class ID	Event
Device Host Name	Host
Device Process Name	'auditid'
Device Product	'BSM'
Device Receipt Time	DateTime
Device Vendor	'Oracle'
Device Version	_DEVICE_VERSION
External ID	subject-pid
File Name	Path
Message	text
Name	Event

Event Type AUE_su

ArcSight ESM Field	Device-Specific Field
--------------------	-----------------------

ArcSight ESM Field	Device-Specific Field
Destination User Name	Text
Device Custom String 4	Text
Device Custom String 5	subject-rgid
Device Custom String 6	NA
Source Host Name	subject-tid-host
Source User Name	subject-ruid

Event Type AUE_rexecd

ArcSight ESM Field	Device-Specific Field
Source Host Name	Text

Event Type AUE_passwd

ArcSight ESM Field	Device-Specific Field
Destination User Name	One of (subject-audit-uid,text)
Device Custom String 6	NA
Source Host Name	host
Source User Name	subject-audit-uid

Event Type AUE_rexd

ArcSight ESM Field	Device-Specific Field
Source Host Name	text

Event Type AUE_ftp_access

ArcSight ESM Field	Device-Specific Field
Device Custom String 4	text
Source Host Name	subject-tid-host

Event Type AUE_login-ssh

ArcSight ESM Field	Device-Specific Field
Source Host Name	subject-tid-host

Event Type AUE_role_login

ArcSight ESM Field	Device-Specific Field
Destination User Name	subject-ruid
Device Custom String 5	subject-gid
Device Custom String 6	subject-rgid
Source Host Name	subject-tid-host

ArcSight ESM Field	Device-Specific Field
Source User Name	subject-audit-uid

Event Type AUE_newgrp_login

ArcSight ESM Field	Device-Specific Field
Destination User Name	subject-ruid
Device Custom String 5	subject-rgid
Device Custom String 6	text
Source Host Name	host
Source User Name	subject-ruid

Event Type AUE_zlogin

ArcSight ESM Field	Device-Specific Field
Device Custom String 1	Zone

Event Type AUE_sudo

ArcSight ESM Field	Device-Specific Field
Device Custom String 5	subject-gid
Source User Name	subject-audit-uid