SNARE

System iNtrusion Analysis & Reporting Environment

Guide to
SNARE for Solaris

INTERSECT ALLIANCE

Documentation History

Version No.	Date	Edits	By whom
1.0	16 November 2003	First draft for the Guide to SNARE for Solaris documentation, in the updated format.	Leigh Purdie

About this booklet

This booklet introduces you to the functionality of the SNARE Agent for the Solaris operating system. Snare for Solaris wraps the default Solaris C2 auditing subsystem, and facilitates objective-based filtering, and remote audit event delivery. SNARE for Solaris will also allow a security administrator to fully remote control the application through a standard web browser if so desired. SNARE has been designed in such a way as to allow the remote control functions to be easily effected manually, or by an automated process.

Other booklets that may be useful to read include:

- “A Snapshot of the System iNtrusion Analysis Reporting Environment software” booklet.
- The “Installation guide to the SNARE Server” booklet.
- The “System Administrator's guide to the SNARE Server” booklet.
- The “Monthly Task Checklist form” and accompanying information.
- The “Transmission protocol choice for event log collection” document.

Table of contents:

1 INTRODUCTION

The team at InterSect Alliance have experience with auditing and intrusion detection on a wide range of platforms - Solaris, Windows NT, Windows 2000, Novell Netware, AIX, even MVS (ACF2/RACF); and within a wide range of IT security in businesses such as - National Security and Defence Agencies, Financial Service firms, Government Departments and Service Providers.

This background gives us a unique insight into how to effectively deploy host and network intrusion detection systems that support and enhance an organisation's business goals.

The development of "SNARE for Solaris" will now allow for BSM C2 event logs to be collected from the Solaris 8 operating system, to be forwarded to a remote audit event collection facility. SNARE for Solaris will also allow a security administrator to fully remote control the application through a standard web browser if so desired. SNARE has been designed in such a way as to allow the remote control functions to be easily effected manually, or by an automated process.

In the spirit of the release of the SNARE for Linux audit event module, InterSect Alliance are proud to release SNARE for Solaris as an open source intiative. Other event audit modules for Linux and Windows have been release under the terms of the GNU Public License. The overall project is called **'SNARE' - System iNtrusion Analysis & Reporting Environment**.

InterSect Alliance welcomes and values your support, comments, and contributions. Our contact details are available from our contact page at www.intersectalliance.com.

2 OVERVIEW OF SNARE FOR SOLARIS

SNARE operates through the actions of three complementary applications:

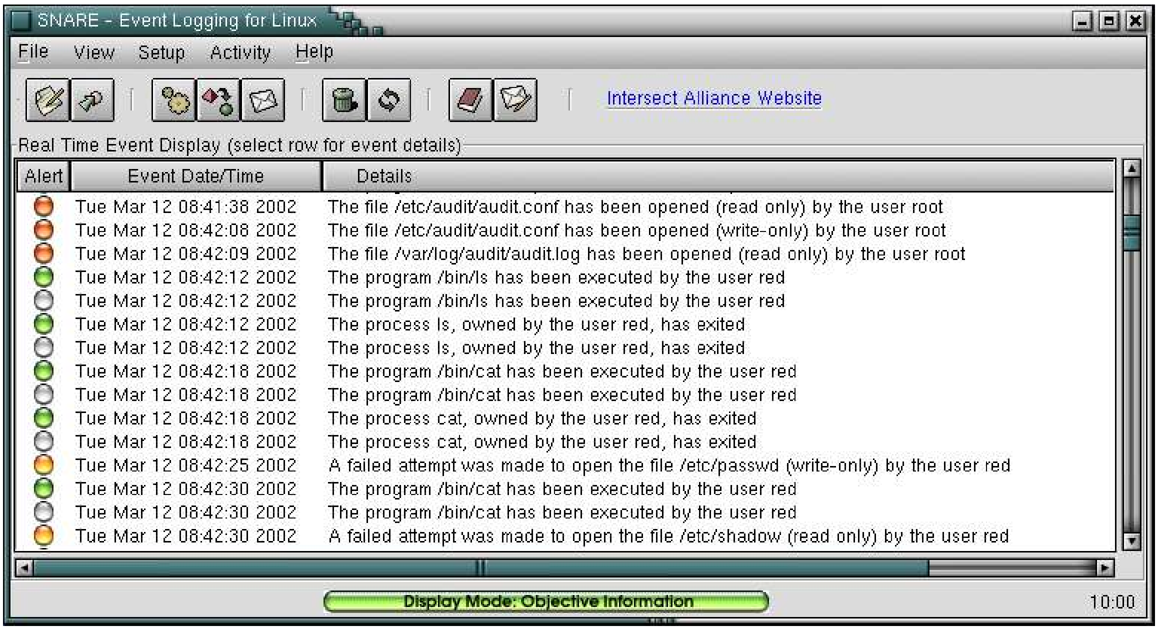
- The “Snare Core” daemon that interfaces with the Solaris kernel
- The graphical configuration and reporting tool (snare)

The SNARE Core service interfaces with the Solaris BSM C2 audit sub-system to read, filter and send event logs from the C2 event logging sub-system (known in Solaris as the Basic Security Module - BSM) to a remote host. The events to be sent will depend on the objectives chosen, and not on the configuration of the BSM files. SNARE will automatically take control of the C2 subsystem without the requirement of any System Administrator intervention. The logs are then filtered according to a set of "objectives" chosen the by the administrator, and then passed over the network, using the UDP protocol, to a remote server. The SNARE Core daemon is able to be remotely controlled using a standard web browser, or via a custom designed tool.

The SNARE Core daemon reads event log data directly from the Solaris kernel. SNARE Core converts the Solaris event log from binary format to text format. Unfortunately, the Solaris C2 audit subsystem has a number of identified "bugs", which are further detailed in Annex C – Known Solaris BSM Bugs. For this, and other reasons, the SNARE Core daemon incorporates a "wrapper" program which will watch for these failures, and restart the service if a key Solaris component fails. It does however mean that the event in question will be lost, an unavoidable consequence of the bugs. The text format event is a TAB separated series of tokens, which are described in detail by the C2 documentation. This format, is also discussed in the section on the Snare Core output format – Annex A. The net result is that a ***raw*** event, as processed by the Snare Core daemon may appear as follows:

```
phoenix SolarisBSM     1     header,146,2,execve(2),,Mon Dec 9
22:23:42 2002, + 140001416 msec   path,/usr/bin/grep
attribute,100555,root,bin,136,379861,0 exec_args,2,grep,snare
      subject,red,root,other,root,other,12228,12212,8236 131095
10.0.1.1   return,success,0 sequence,65941
```

Since the above event log contains a great deal of information for the average user, and in a format which doesn't lend itself to interpretation, SNARE also incorporates a graphical front end tool. The graphical tool allows for easy configuration of all the event logging parameters, and display of event display. A screenshot of the main window, which includes the main display is shown below.


SNARE - Event Logging for Linux
File
View
Setup
Activity
Help
Intersect Alliance Website
Real Time Event Display (select row for event details)
Display Mode: Objective Information
10:00

Alert	Event Date/Time	Details
🔴	Tue Mar 12 08:41:38 2002	The file /etc/audit/audit.conf has been opened (read only) by the user root
🔴	Tue Mar 12 08:42:08 2002	The file /etc/audit/audit.conf has been opened (write-only) by the user root
🔴	Tue Mar 12 08:42:09 2002	The file /var/log/audit/audit.log has been opened (read only) by the user root
🟢	Tue Mar 12 08:42:12 2002	The program /bin/ls has been executed by the user red
⚪	Tue Mar 12 08:42:12 2002	The program /bin/ls has been executed by the user red
🟢	Tue Mar 12 08:42:12 2002	The process ls, owned by the user red, has exited
⚪	Tue Mar 12 08:42:12 2002	The process ls, owned by the user red, has exited
🟢	Tue Mar 12 08:42:18 2002	The program /bin/cat has been executed by the user red
⚪	Tue Mar 12 08:42:18 2002	The program /bin/cat has been executed by the user red
🟢	Tue Mar 12 08:42:18 2002	The process cat, owned by the user red, has exited
⚪	Tue Mar 12 08:42:18 2002	The process cat, owned by the user red, has exited
🟠	Tue Mar 12 08:42:25 2002	A failed attempt was made to open the file /etc/passwd (write-only) by the user red
🟢	Tue Mar 12 08:42:30 2002	The program /bin/cat has been executed by the user red
⚪	Tue Mar 12 08:42:30 2002	The program /bin/cat has been executed by the user red
🟠	Tue Mar 12 08:42:30 2002	A failed attempt was made to open the file /etc/shadow (read only) by the user red

Graphic 2.1 Main window

3 INSTALLING AND RUNNING SNARE

3.1 SNARE INSTALLATION

SNARE is available in tarball format, and has been designed with an installation script to allow for easy installation and configuration of all critical components. The "snarecore-2.1.tar.gz" file includes the following critical components.

WHAT YOU NEED...

•Snarecore
The SNARE Core daemon is contained in the "snarecore" binary. This binary contains all the programs to read the event log records, filter the events according to the "objectives", provide a web based remote control interface, handle known Solaris bugs, configure the BSM C2 subsystem based on required objectives, and provide all the necessary logic to allow the binary to act as a daemon.

•Snare
This binary only contains the program to provide for the SNARE front end (GUI) functions, as shown in Figure 1. However, the graphical user interface will not be of any use, unless the SNARE Core service has also been installed. Note that the graphical environment "Gnome" is required to execute the snare GUI. If the system detects that the "/opt/gnome" directory does not exist, then the GUI will not be installed.

•install.sh/uninstall.sh
These two scripts undertake the installation and uninstall functions required to ensure SNARE for Solaris works as required. The scripts prompt the user on the steps required to be undertaken and the choices to be made (discussed in detail below). They are designed to be executed interactively.

•configuration files
A myriad of configuration files are required to correctly run the BSM C2 subsystem. These configuration files have been tailored to meet the SNARE requirements, and include the files: audit_event, audit_control, snare.conf, etc. These configuration are copied to the /etc/security directory during the installation process.

How to... Install the Snare package for Solaris

1. Download the "snarecore-2.1.tar.gz" file from the Intersect Alliance website.
2. Ensure that the BSM subsystem has been installed, by typing "bsmconv" in a root prompt.
3. As "root", type "tar -zxvf snarecore-2.1.tar.gz" at the root prompt, noting that the file name may be different if a version other than 2.0 is downloaded. A "snarecore-2.1/" directory will be created. Enter this directory by typing "cd snarecore-2.1/"
4. In order to commence the installation, type in "./install.sh". A series of prompts will then be displayed, requesting that various parameters be set. Read these settings carefully, using this manual as reference. Most of the references are discussed later in this guide, so it pays to read this guide first, before installing the software
5. Once the installation process has completed the SNARE Core daemon should be started by issuing the command "/etc/init.d/snare restart".

3.2 RUNNING SNARE

Upon installation of SNARE for Solaris, the GUI binary will be installed in the /opt/gnome/bin directory. Note that if the Gnome desktop has not been installed, then the GUI will not have been installed as part of the installation process.

The SNARE Core daemon must be running, if the events are to be passed to a remote host. The Snare Core daemon may be stopped, started or restarted, by issuing the command: "/etc/init.d/snare stop", "/etc/init.d/snare start" or "/etc/init.d/snare restart", respectively.

How to... Run the GUI:

1. For the gnome 1.4 version of the Snare GUI, ensure you have the gnome 1.4 libraries installed. You can download these from www.sunfreeware.com.
2. Execute the command "/opt/gnome/bin/snare"

How to... Remote Audit monitoring

If the audit daemon is run on a system that does not have X-Windows installed, then you may still configure the audit subsystem, and view logs remotely, using the SNARE GUI. On your audited system, set your X-Windows DISPLAY variable to a system that has X-Windows running, and start the SNARE application:

```
myguisystem$ telnet auditedsystem.mycompany.com
..
auditedsystem$ /bin/su -
[password]
auditedsystem# export DISPLAY=myguisystem.mycompany.com:80
auditedsystem# snare &
```

4 SETTING THE AUDIT CONFIGURATION

4.1 AUDIT CONFIGURATION

The audit configuration is stored as */etc/security/snare.conf*. This file contains all the details required by the audit daemon to successfully execute. Failure to have a correct configuration file available in this location will not "crash" the daemon, but may result in selected events not being able to be read.

> Tip: Manual editing of the snare.conf configuration file is possible, but care should be taken to ensure that it conforms to the required format for the audit daemon. Also, any use of the graphical SNARE tool to modify security objectives or selected events, may result in manual configuration file changes being overwritten. Details on the configuration file format can be viewed in Annex C – SNARE Configuration File.

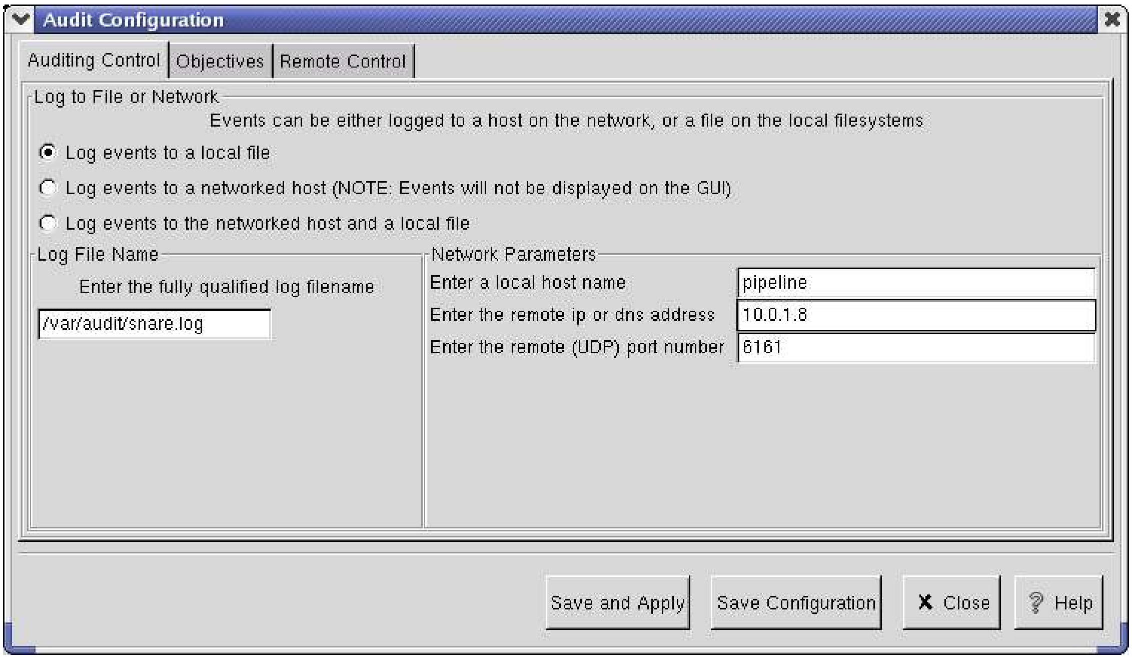
The most effective and simplest way to configure the SNARE audit daemon is to use the graphical front end tool. The audit configuration window can be selected from the *Setup -> Audit Configuration* menu, or directly from the associated toolbar button.

4.2 AUDITING CONTROL

The initial audit configuration parameters to consider are:

- The hostname, IP address and UDP port of the remote collection server,
- The requirement to maintain a log file, or send the events to a remote server, or both, and
- The location of the logfile.

These three parameters are shown in the Auditing Configuration window, shown in Figure 2 below. Note that Graphic 4.1 below shows the tabs "Objectives" and "Remote Control". This is discussed later in this documentation. .



Log to File or Network

Events can be either logged to a host on the network, or a file on the local filesystems

● Log events to a local file

○ Log events to a networked host (NOTE: Events will not be displayed on the GUI)

○ Log events to the networked host and a local file

Log File Name

Enter the fully qualified log filename: /var/audit/snare.log

Network Parameters

Enter a local host name: pipeline

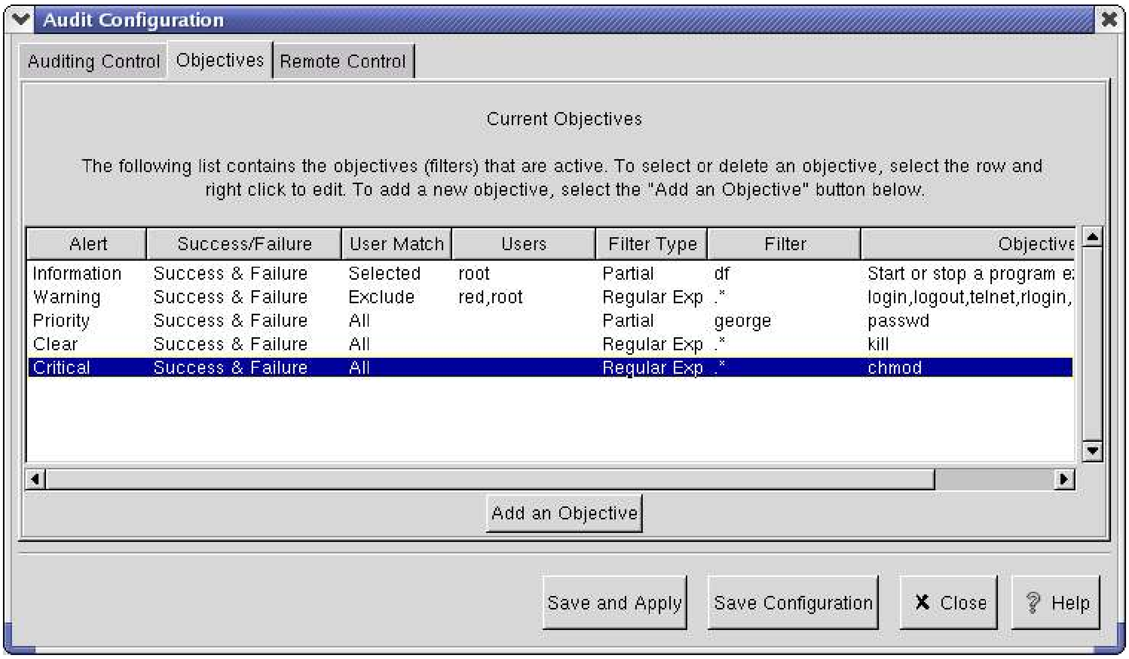
Enter the remote ip or dns address: 10.0.1.8

Enter the remote (UDP) port number: 6161

Graphic 4.1 Audit Configuration Window

The hostname field can be used to override the name that is given to the Solaris host's name, which is set when Solaris is first installed. Unless a different name is required to be sent in the processed event log record, leave this field blank, and the SNARE Core service will use the default host name set during installation. Note that executing the command hostname on a command prompt will display the current host name allocated to the host. The "port" shown in Figure 2 is the remote server's port that will be used to collect the events. If, for example, the Intersect Alliance collection server is used, then this is the default port at which will be used. Since SNARE for Solaris replaces a number of the BSM components, it is required that a facility be provided to log events to a file. This is shown in Graphic 4.1, although the capability does exist to pass all events to a remote hosts without having to save them to disk.

Part of the major function of the SNARE audit subsystem is to filter events. This is accomplished via the advanced auditing "objectives" capability. Any number of objectives may be specified, and are display within the "Objectives" tab, as shown in Graphic 4.2 below.



Current Objectives

The following list contains the objectives (filters) that are active. To select or delete an objective, select the row and right click to edit. To add a new objective, select the "Add an Objective" button below.

Alert	Success/Failure	User Match	Users	Filter Type	Filter	Objective
Information	Success & Failure	Selected	root	Partial	df	Start or stop a program e
Warning	Success & Failure	Exclude	red,root	Regular Exp	.*	login,logout,telnet,rlogin,
Priority	Success & Failure	All		Partial	george	passwd
Clear	Success & Failure	All		Regular Exp	.*	kill
Critical	Success & Failure	All		Regular Exp	.*	chmod

Graphic 4.2 Objectives Window

Each of the objectives provides a high level of control over which events are selected and reported. Events are selected from a group of high level requirements, and further refined using selected filters. These high level requirements can from a pre-defined list of system calls grouped into a common objective, or can be from individually selected system calls. The "high level" groups are as follows:

- Read, write or create a file or directory.
- Modify system, file or directory attributes.
- Start or stop a program execution.
- Change user or group identity.
- Open a file for reading only.
- User logon or logoff.
- Establish an outgoing network connection.
- Any event(s).

Note that the groups above are provided to service the most common security objectives, that are likely to be encountered. If other event types are required, then the "Any event(s)" objective will allow fully tailored objectives to be set. From each of these groups, a criticality level can be applied. These criticality levels are critical, priority, warning, information and clear. These security levels are provided to enable the SNARE user to map audit events to their most pressing business security objectives, and to quickly identify the criticality of an event, via the coloured "buttons" on the SNARE graphical user interface, as shown in Figure 1.

The following filters can be applied to incoming audit events:

1. Filter on the event-specific matchable item

   Each event contains a particular piece of information that represents the core data that needs to be communicated. For the "Open a file for reading only" group, for example, this would be the name of the file and/or directory opened or created. For the "Start or stop a program execution" group, this would be the name of the program in question. The event match allows a "partial", "exact" or "regular expression" match term to check against the event-specific matchable item. A "partial " match will look for the sequence of characters specified somewhere within the event-specific matchable item. For example, if the partial match of "pass" is specified for "Read, write or create a file or directory", then the following example filenames would all match the term:

   - /etc/passwd
   - /usr/lib/passfilt.so
   - /home/red/khyber_pass.txt

   An "exact" match will match the specified term exactly. For example, if the term is /etc/passwd, then the file /etc/passwd would match, but the file /etc/passwd.backup will NOT match.

   A "regular expression " match matches the supplied extended regular expression against the event-specific matchable item. Regular expressions are an advanced form of specifying filters, and should only be used by those with regular expression experienced. For example, the term ".*[Pp]ass(word|wd).*" would match the following:

   - /etc/passwd
   - /tmp/PasswordFile

   but would not match

   - /etc/PASSWD/
   - /home/red/PaSsWoRd .txt

2. Filter on user.

   Any number of users can be selected, and should be entered with commas separating each user. If no users are entered, ALL users are assumed to be audited. Alternatively, specific users may be EXCLUDED from any individual objective, leading to objectives such as "tell me whenever any user except 'root' or 'red' generate an event" . If the user exclusion function is selected, SNARE will only report users that DO NOT match the supplied list of users.

3. Filter on return value

   An audit event will either return a success or failure return code. SNARE allows a user to filter on the return value.

4. Filter on special, event-specific fields

Some events, including open() and socketcall(), allow additional filters to be specified, to provide more flexible search criteria.

- open()

  The open event provides the additional capability to filter on open-flags. The flags are specified in regular expression format, and can include (in the following order):

  O_WRONLY
  O_RDWR
  O_RDONLY
  O_CREAT
  O_EXCL
  O_NOCTTY
  O_TRUNC
  O_APPEND
  O_NONBLOCK
  O_SYNC
  O_NOFOLLOW
  O_DIRECTORY
  O_LARGEFILE

  For Example, the following flags can be logically 'or'ed together (in regular expression form) to specify an objective that translates to "Let me know whenever a file is opened in WRITE mode.

  - open(O_WRONLY|O_RDWR|O_CREAT|O_TRUNC|O_APPEND)

  Whereas the following three examples all specify "Read or Write":

  - open(.*)
  - open(O_.*)
  - open(O_WRONLY|O_RDWR|O_RDONLY|O_CREAT|O_EXCL|O_NOCTTY|O_TRUNC|O_APPEND|O_NONBLOCK|O_SYNC|O_NOFOLLOW|O_DIRECTORY|O_LARGEFILE)

  More information on the flags associated with the open() system call are available from the Solaris manual pages (see 'man open').

- socketcall()

  The Solaris kernel uses the 'socketcall' system call to serve the requirements of the 'connect', 'accept' and related system calls. In order to monitor 'connect' and 'accept' calls only, the system call 'type' can be included within the objective.

  For example, socketcall(ACCEPT) only monitors accept() calls. socketcall(CONNECT) only monitors connect() calls.

  socketcall(.*) or socketcall(CONNECT|ACCEPT) monitor both accept and connect.

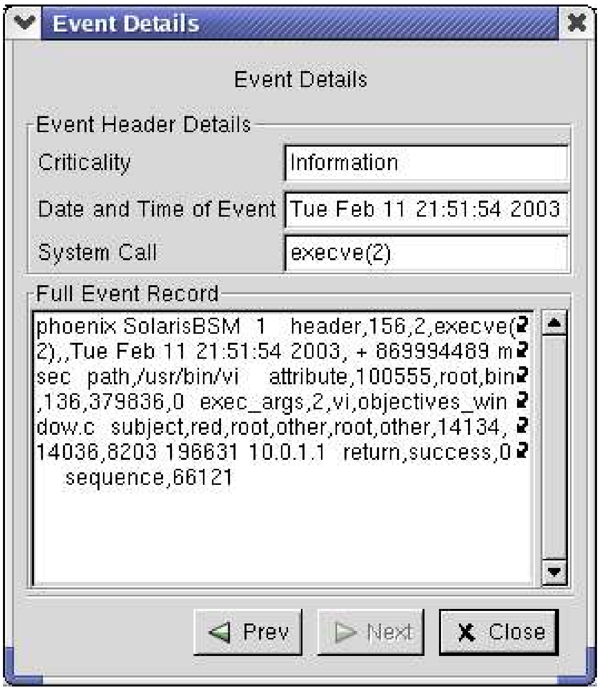
Once the above settings have been finalised, clicking "Save" will save the configuration to the snare configuration file. However, to ensure the SNARE Core daemon has received the new configuration, the SNARE Core daemon MUST be restarted via the "Save and Apply" or via "Apply and Restart Audit" menu item or the corresponding toolbar button.

5 AUDIT EVENT VIEWER FUNCTIONS

The main SNARE window contains the events that have been filtered. Events collected, which meet the filtering requirements as per the Audit Configuration , will be displayed in the main window. Note that the display is essentially a display from the event log file. Once the SNARE front end is started the display will be clear. Selecting the "reload" toolbar button or equivalent menu item will display all events on the log file. Note that if the "log events to a networked host" feature is selected, then events will not be displayed to the main SNARE GUI. A key feature of the Snare Core Service is that events are not stored locally on the host (except for events stored natively in the event log file), but rather sent out over the network to a remote host.

A summary version of the event record is displayed on the main window. For more details on a specific event, the relevant row from the main window can be selected using the mouse. A pop-up window will then display more comprehensive details related to the event. The event details window is shown in Figure 4. The details which are displayed depend entirely on the event that is being displayed. Once an event window is displayed, other events may be displayed by selecting the "Up" or "Down" key.

The fields shown in the event window relate to the parameters of the system call that was audited.



Event Details

Event Header Details

Criticality: Information

Date and Time of Event: Tue Feb 11 21:51:54 2003

System Call: execve(2)

Full Event Record

```
phoenix SolarisBSM  1   header,156,2,execve(
2),,Tue Feb 11 21:51:54 2003, + 869994489 m
sec  path,/usr/bin/vi    attribute,100555,root,bin
,136,379836,0  exec_args,2,vi,objectives_win
dow.c  subject,red,root,other,root,other,14134,
14036,8203 196631 10.0.1.1  return,success,0
   sequence,66121
```

Graphic 5.1 Event Details window

The main window event list may be cleared from the menu by selecting the item Activity->Clear all Current Events, or from the corresponding button from the toolbar. Note that clearing the main event viewer, DOES NOT clear the event log file. Clearing the log file can only be done outside of the SNARE GUI.

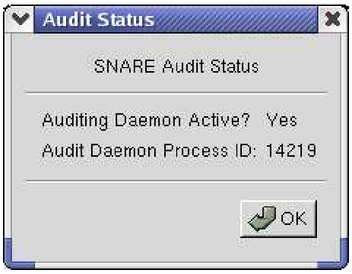
In order to view all previous events contained in the log file, the "Reload" menu item can be selected from *Activity->Reload Log File*, or from the corresponding button from the toolbar. Note that displaying the entire log file may take some time, depending on the size of the file. Alternatively, the main window event list may be cleared from the menu by selecting the item Activity->Clear all Current Events, or from the corresponding button from the toolbar. Note that clearing the main event viewer, DOES NOT clear the log file.

6 REMOTE CONTROL AND MANAGEMENT

The SNARE Core service is a separate standalone component of the SNARE system, as described in *2 Overview of SNARE for on page 5*. However, the SNARE graphical front end can be used to control a number of aspects of its operation. Primarily, the audit configuration can be developed and set using the graphical tool, as described in the previous sections. However, two other functions are available to manage the SNARE Core service.

The audit daemon can be restarted directly from the menu item *Activity->Apply and Restart Audit*. This will instruct the audit daemon to re-read the configuration file, clear the buffers and restart. This function is useful when changes to the audit configuration have simply been saved to the configuration file, without being "applied". The user can therefore select when to activate a new configuration by selecting this menu item, or its corresponding button on the toolbar.

The audit daemon status can be viewed by selecting the View->Audit Status menu item, or its corresponding toolbar button. TThis will display whether the SNARE Core service is active, along with a number of statistical parameters which are displayed on the screen. The audit status window is shown in Graphic 6.1 below.

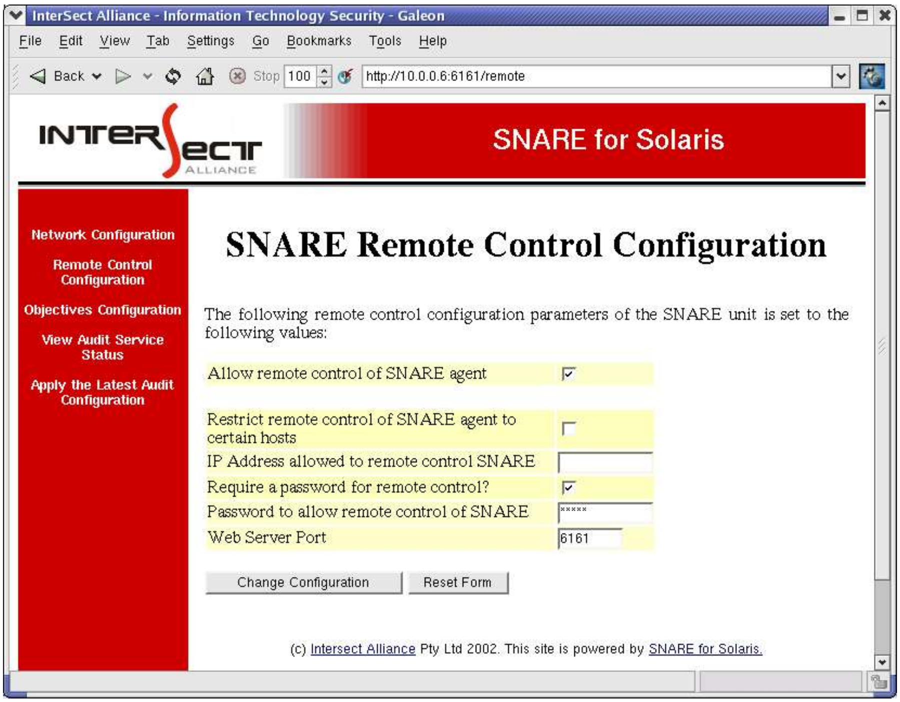


Auditing Daemon Active? Yes
Audit Daemon Process ID: 14219

Graphic 6.1 Audit Status window

6.1 REMOTE CONTROL

A significant function of the SNARE Core service is its ability to be remote controlled. This facility has been incorporated to allow all the functions normally available through the front end SNARE tool, to also be available through a standard web browser. The SNARE Core service employs a custom designed web server to allow configuration through a browser, or via an automated custom designed tool. Graphic 6.2 below shows a web browser connecting to a SNARE agent:


InterSect Alliance - Information Technology Security - Galeon
File Edit View Tab Settings Go Bookmarks Tools Help
Back Stop 100 http://10.0.0.6:6161/remote
INTERSECT ALLIANCE
SNARE for Solaris
Network Configuration
Remote Control Configuration
Objectives Configuration
View Audit Service Status
Apply the Latest Audit Configuration
Change Configuration
Reset Form

SNARE Remote Control Configuration

The following remote control configuration parameters of the SNARE unit is set to the following values:

Allow remote control of SNARE agent	☑
Restrict remote control of SNARE agent to certain hosts	☐
IP Address allowed to remote control SNARE	
Require a password for remote control?	☑
Password to allow remote control of SNARE	*****
Web Server Port	6161

(c) Intersect Alliance Pty Ltd 2002. This site is powered by SNARE for Solaris.

Graphic 6.2 Remote Control window

The functions available through the web browser are identical to those available through the SNARE front end. In either case, the actual remote control parameters may be controlled in a similar fashion as the audit configuration. The parameters which may be set for remote control operation are shown in Graphic 6.2 and discussed in detail below:

- **Allow Remote Control**. Selecting this check box will allow the SNARE agent to be remote controlled by a remote host. This host may be independent from the central audit collection server. If the remote control feature is unselected, it may only be turned on by using the SNARE front end tool on the hosted PC which the SNARE agent has been installed.
- **Allow Remote Control from IP Address**. Remote control actions may be limited to a given host. This host, entered as an IP address in this checkbox, will only allow remote connections to be effected from the stated IP address. Note that access control based on source IP address is prone to spoofing, and should be considered as a security measure used in conjunction with other countermeasures.
- **Set Password**. A password may be set so that only authorised individuals may access the remote control functions. If accessing the remote control functions through a browser or custom designed tool, note that the userid is "snare", and the password is whatever has been set through this setting. Note that this password is not encrypted.

Web Server Port. Normally, a web server operates on port 80. If this is the case, then a user need only type the address into the browser to access the site. If however, a web server is operating on port (say) 8085, then the user needs to type **http://mysite.com:8085** to reach the web server. The default SNARE Core web server port may be changed using this setting, if it conflicts with an established web server. However, care should be taken to note the new server port, as it will need to be placed in the URL needed to access the SNARE agent.

6.2 LOG ROTATION

Depending on the SNARE configuration, the log file may be small or large. In any case, it is normal houskeeping practice that logs either be rotated or archived. Depending on the site requirements, a rotation scheme that keeps old copies of the last (say) 7 days may be sufficient. In this case, it may be sufficient to simply include a CRON or ANACRON job, and use a program such as logrotate to ensure the current log file does not grow to an unmanageable size. Alternatively, you may wish to archive all log files to backup media such as tape or CD-ROM. This may be scheduled using CRON or undertaken manually. In either case, it is important to note that the audit daemon should restarted, so that it opens the new log file for writing the events.

6.3 REMOTE DISTRIBUTION

SNARE provides the facility to send events to a remote host, using UDP. In conjunction with the audit log archive script provided within the SNARE distribution, this will facilitate the remote storage of audit logs for later analysis.

The settings for these options are available from the Setup menu on the main window, or their corresponding toolbar. Note that if events are being sent exclusively to a remote machine via the network, they will not be displayed within the GUI. In order to display events in the GUI, as well as sending the data to a remote system, set the "Auditing Control" options to "Log events to the networked host and a local file.

A simple collection and archive script for receiving logs from a remote SNARE node, is available from the SNARE project page. The collection / archive script receives data from one or more SNARE clients, and saves the data off to a file per-date, per-system in /var/log/audit. Files will be given names based on the date, and the audit source, in the format YYYYMMDD-host.name.SolarisBSM (eg: /var/log/audit/20020321-test.intersectalliance.com.SolarisBSM).

7 AUDIT COLLECTION AND ANALYSIS

The team at Intersect Alliance have produced software that enable remote control, collection, analysis and of output from all SNARE agents, including Windows, Linux and Solaris, as well as applications such as web servers. The details of this software is available from the Intersect Alliance web site (www.intersectalliance.com), and is custom designed to suit a customer's requirements.

8 ABOUT INTERSECT ALLIANCE

InterSect Alliance is a team of leading information technology security specialists in both the "technical" and "policy" areas. In particular, Intersect Alliance are noted leaders in key aspects of IT Security, including host intrusion detection. Our solutions have and continue to be used in the most sensitive areas of Government and business sectors. Intersect Alliance consult and contract to number of agencies in Australia and in Asia Pacific, for both the business and Government sectors.

The Intersect Alliance business strategy includes demonstrating our commitment and expertise in IT security by releasing Open Source products such as SNARE. Intersect Alliance intend to continue releasing tools that enable users, administrators and clients worldwide to achieve a greater level of productivity and effectiveness in the area of IT Security, by simplifying, abstracting and/or solving complex security problems.

Visit the Intersect Alliance website for more information at www.intersectalliance.com.

APPENDIX A - EVENT OUTPUT FORMAT

The SNARE Core daemon reads data from the Solaris kernel, via published APIs. It converts the binary audit data into text format using the "praudit" utility, and separates information out into a series of token/data groups. Three different field separators are used in order to facilitate follow-on processing - TABS separate 'tokens', COMMAS separate data within each token, and SPACES separate elements within data.

A 'Token' is a group of related data, comprising a 'header', and a series of comma separated fields which make up data that relates to the header.

Examples of tokens:

- process,1628,gcc
- return,0
- path,/etc/audit/audit.conf
- arguments,ls -al

Groups of tab separated tokens make up an audit event, which may look something like this, depending on whether the audit daemon has been set to 'objective' or 'event' reporting mode (see the configuration section for more information):

```
phoenix SolarisBSM   1   header,146,2,execve(2),,Mon Dec 9 22:23:42
2002, + 140001416 msec    path,/usr/bin/grep
attribute,100555,root,bin,136,379861,0 exec_args,2,grep,snare
subject,red,root,other,root,other,12228,12212,8236 131095 10.0.1.1
return,success,0 sequence,65941
```

A simple example PERL script for extracting data from a raw SNARE log is as follows:

```
#!/usr/bin/perl
# Usage: cat /var/log/audit/audit.log | ./extract.pl
# Creates an associative array containing the elements of the event record, and prints the data.

while($input=<STDIN>) {
    chomp($input);
    %Record=();
    @tokens=split(/\t/,$input);    # Split the line into TAB delimited tokens.
    foreach $token (@tokens) {
        @elements=split(/,/,$token);    # Pick out the elements within each token.
        $header=$elements[0];
        if($header eq "objective") {
            $Record{$header}{"criticality"} = $elements[1];
            $Record{$header}{"datetime"} = $elements[2];
            $Record{$header}{"description"} = $elements[3];
        } elsif ($header eq "event") {
            $Record{$header}{"eventid"} = $elements[1];
            $Record{$header}{"datetime"} = $elements[2];
        } elsif ($header eq "user") {
            $Record{$header}{"uid"} = $elements[1];     # User ID
            $Record{$header}{"gid"} = $elements[2];     # Group ID
            $Record{$header}{"euid"} = $elements[3];    # Effective User ID
            $Record{$header}{"egid"} = $elements[4];    # Effective Group ID
       } elsif ($header eq "process") {
            $Record{$header}{"pid"} = $elements[1];     # Process ID
            $Record{$header}{"name"} = $elements[2];    # Process Name (max 16 chars)
        } elsif ($header eq "path") {
            $Record{$header}{"path"} = $elements[1];
        } elsif ($header eq "destpath") {
            $Record{$header}{"destpath"} = $elements[1];    # Destination path
        } elsif ($header eq "arguments") {
            $Record{$header}{"args"} = $elements[1];
        } elsif ($header eq "attributes") {
            $Record{$header}{"attrib"} = $elements[1];
        } elsif ($header eq "return") {
            $Record{$header}{"code"} = $elements[1];
        } elsif ($header eq "target") {
            $Record{$header}{"user"} = $elements[1];
        } elsif ($header eq "owner") {
            $Record{$header}{"user"} = $elements[1];
            $Record{$header}{"group"} = $elements[2];
        } elsif ($header eq "socket") {
            $Record{$header}{"sourceip"} = $elements[1];
            $Record{$header}{"destip"} = $elements[2];
            $Record{$header}{"sourceport"} = $elements[3];
            $Record{$header}{"destport"} = $elements[4];
        } elsif ($header eq "sequence") {
            $Record{$header}{"number"} = $elements[1];
        }
    }
    # We now have the data in an associative array.
    # Roll through the array, and print the data in token groups.
    foreach $header (keys(%Record)) {
        print "Header: $header\n";
        foreach $element (keys(%{$Record{$header}})) {
            print "$element = " . $Record{$header}{$element} . "\n";
        }
    }
    # In addition, if the event is execve, the effective user ID
    # is 'root', but the real user ID is NOT, then display an alert.
      if($Record{"event"}{"eventid"} eq "execve" && $Record{"user"}{"euid"} eq "root" && $Record
{"user"}{"uid"} ne "root") {
        print "Danger: SetUID program " . $Record{"arguments"}{"args"} . " has been run by the user "
. $Record{"user"}{"uid"} . " .\n";
     }

    print "\n";
    }

    print "----- Done -----\n";
```

APPENDIX B - SNARE CONFIGURATION FILE

Details on the audit configuration are discussed in the Audit Configuration section. The purpose of this section is to discuss the makeup of the configuration file. The SNARE configuration file is located at ***/etc/security/snare.conf*** , and this location may not be changed. If the configuration file does not exist, the audit daemon will execute, but will not actively audit events until a correctly formatted configuration file is present, or unless specific instructions are passed to the audit module at load time.

SNARE can be configured in several different ways, namely:

- Via the graphical tool (*Recommended*), or
- Via the web server,
- By setting module options at load time, or
- By manually editing the configuration file

The format of the audit configuration file is discussed below.

`[HostID]`	This item stores the hostname, if it is different from the assigned Solaris hostname.
`name=<hostname>`	This is the name of the host.
`[Output]`	By default, if no output section exists within the configuration file, the audit daemon will send audit data out to standard out (STDOUT). Note that audit events will be sent to all valid destinations specified in the Output section. As such, events can be sent to one or all of a file, standard output and to a remote network destination (Only one file, and one network destination is supported however).
`network=hostname:port`	Audit data can be sent to a remote system using the UDP protocol. Data will be sent to the remote host, and network port specified here.
`network=stdout`	If stdout is specifically defined within the Output section, the audit daemon will send data to standard out.
`file=stdout`	If stdout is specifically defined within the Output section, the audit daemon will send data to standard out.
`file=/fully/qualified/file/name`	The audit daemon will send data to the fully qualified filename specified within the [Output] section. Note that if the audit daemon is not running as root, the file must be writable by the user under which the audit daemon is running.

<table>
<tr><td><code>[Objectives]</code></td><td>This section describes the format of the objectives. Objectives are composed of:
1. Criticality - an integer between 0 and 4 that indicates the severity of the event. 0 is "clear", 4 is "critical.
2. The event ID - this must either correspond to a valid auditable event, or be set to "*" for any event. Note that the graphical tool will convert the generic "groups" in the Audit Configuration window to the required events. For example, the abstracted group "Remove a file or directory", will result in the event entry "event=rmdir,unlink" being written, with the events comma delimited. Note also that additional filter flags may be specified, as discussed in section 4 above.
3. The return code defines whether to report event (system call) if it is a success, failure or both ("*")
4. The user list is listed is used to audit events for selected users, and is in extended regular expression format.
5. The match term is the filter expression, and is again defined in extended regular expression format.
Note that whitespace will be trimmed from the start and end of items, but will be assumed to be valid when bracketed by other characters.</td></tr>
<tr><td><code>criticality=1 event=open_r return=* user=^(red|george)$ match=^/etc/shadow$</code></td><td>Report at criticality level 1, whenever the users "red" or "george", open the file /etc/shadow in READ ONLY mode.</td></tr>
<tr><td><code>[Remote]</code></td><td>This subkey stores all the remote control parameters.</td></tr>
<tr><td><code>allow=1</code></td><td>"Allow" is an integer, and set to either 0 or 1 to allow remote control. 1= allow remote, 0=do not allow.</td></tr>
<tr><td><code>listen_port=6161</code></td><td>This value is the web server port. A missing "listen_port" will default the web server to port 80.</td></tr>
<tr><td><code>restrict_ip=10.0.0.1</code></td><td>This is an IP address, that will be used so that this address will be the only host that is allowed to connect to the web server. If this item does not exist, then the web server will not restrict by IP address.</td></tr>
<tr><td><code>accesskey=snare</code></td><td>This value is the password that is used to log into the SNARE web server. If this item does not exist, then a password will not be requested when connecting to the web server.</td></tr>
</table>

APPENDIX C - KNOWN BSM "BUGS"

The Solaris BSM subsystem has a number of "bugs" (or "fetures") which have been identified and documented in the course of the SNARE for Solaris development process. There are detailed below:

- praudit dies when a network related system call is processed. SNARE detects praudit failure, and restarts the process.
- Event log formatting relating to time is inconsistent - some events include a comma between seconds and milliseconds, others do not.
- The 'auditon' system call ONLY turns on system-related audit events (ie: events with an ID less than 512).
- Forked processes do NOT inherit audit settings. Each process is re-applied with the contents of /etc/security/audit* at startup.
- The Solaris +cnt flag doesn't seem to work correctly. Audit must be explicitly turned off, or buffer overflows can occur.
- A auditsvc call with a disk-space size of 0 will cause a kernel panic without appropriate solaris 6/7 patches.
- Using a pipe as the output file for auditsvc will cause a kernel panic without appropriate solaris 6/7 patches.
- Taking some of the new solaris 9 extra audit events out from the audit_event file causes the audit subsystem to fail.
- Solaris 6 requires at least one more patch over and above those discussed in the Snare installation script – we have yet to find out which it is, but any system that has applied the Sun recommended patches post 2001 should be fine.