SNARE

System iNtrusion Analysis & Reporting Environment

Guide to Snare for Solaris v4.0

INTERSECT ALLIANCE

About this guide

This guide introduces you to the functionality of the Snare Agent for the Solaris operating system (Solaris version 10 and version 11). Snare for Solaris interacts with the event, auditing subsystem for the Solaris operating system, and provides 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. Please note that this guide relates to the purchased Enterprise version of the Snare for Solaris agent. Many of the options described in this document are not in the OpenSource agent.

Other guides that may be useful to read include:

- Snare Overview
- The Snare Toolset - A White Paper

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, Linux, MS SQL Server and Mac OSX, 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 organization's business goals.

'Snare for Solaris' allows event logs from the Solaris BSM Basic Security Module (BSM) to be collected from the operating system, and forwarded to a remote audit event collection facility after appropriate filtering. Snare for Solaris will also allow a security administrator to fully remote control the application through a standard web browser if so desired.

Other Snare agents are also available including Snare for Linux, OSX, MSSQL, Epilog and Windows. The agents are capable of sending data to a wide variety of target collection systems, including our very own **'Snare Server'**. See *Chapter 5 Snare Server* for further details.

Welcome to **'Snare'** - **System iNtrusion Analysis & Reporting Environment**.

2 Overview of Snare for Solaris

Snare operates through the actions of the ***audit_snare*** plugin which is loaded by the *auditd* daemon. The *auditd* daemon interfaces with the Solaris kernel and passes kernel audit information to the *audit_snare* plugin.

The *audit_snare* plugin reads, filter and send event logs from the kernel audit BSM 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 BSM subsystem without the requirement of any System Administrator intervention. The logs are then filtered according to a set of 'objectives' chosen by the administrator, and then passed over the network, using the UDP, TCP or TCP (with SSL encryption) protocol, to a remote server. The *audit_snare* plugin is able to be remotely controlled using a standard web browser, or via a custom tool that acts as a web client.

The *audit_snare* plugin reads event log data directly from the audit deamon. *audit_snare* converts the Solaris event log from binary format to text format. Unfortunately, the Solaris BSM audit subsystem has a number of identified 'bugs', which are further detailed in – Known BSM Bugs, this prevents events for some processes from being generated. The events sent by the *audit_snare* plugin are in a text-format, TAB separated series of tokens, which are described in detail by the BSM documentation. This format, is also discussed in Configuration File Description - Event Output Format. The net result is that a ***raw*** event, as processed by the *audit_snare plugin* 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
```

All of the fields in the above record are sent to the remote server, whether this is a Snare Server, or a custom tool. The Snare Server is then used to interpret some or all of the tokens in the log file to determine the value of the event to the security or system administrators.

3 Installing and running Snare

3.1 Snare installation

What you need...

- An appropriate Solaris Distribution
- Enterprise customers may download the SnareSolaris package from the Snare Secure Area at https://www.intersectalliance.com.
- Solaris version 10 only: A Solaris installation does not normally activate the utilities necessary to activate the auditing subsystem. As such, it must be separately activated on the Solaris host, before the Snare agent will work in collecting and filtering events. The auditing subsystem may be activated using the '/etc/security/bsmconv' script.

How to... Install Snare for Solaris package.

1. Logon as root user, i.e. at the command prompt enter the command `/bin/su` and enter the root password when prompted. Issue the command, as root as per your distribution:
   `>pkgadd -d SnareSolaris-supp-4.0.0-i386-S11.pkg`
2. This will install Snare for Solaris and restart the audit daemon (auditd).

How to... Remove Snare for Solaris package (if required).

1. Query the database to ensure Snare is installed
   `>pkginfo -l SnareSolaris`
2. Remove the Snare for Solaris package
   `>pkgrm SnareSolaris`

3.2 Running Snare

To view the Snare Remote Control Interface enter the URL http://localhost:6161 or http://hostname:6161 where "hostname" is the DNS name or IP address of the target machine.

After installation the auditd daemon will be running. This daemon must be running if the events are to be passed to a remote host.

How to... Restart the auditd daemon either:

1. By issuing the command:
   `> svcadm restart system/auditd`
2. Via the Remote Control Interface:
   From the menu on the right hand side select *Apply the Latest Audit Configuration* to restart the daemon.

3.3 Audit configuration

The Snare configuration is stored as */etc/security/snare.conf*. This file contains all the details required by Snare to configure the audit subsystem to successfully execute.

The configuration of */etc/security/snare.conf* can be changed either:

- directly

  Care should be taken if manually editing the snare.conf configuration file to ensure that it conforms to the required format for the audit daemon. Also, any use of the Remote Control Interface 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*Configuration File Description*. Failure to specify a correct configuration file will prevent Snare from running or may result in selected events not being able to be read.

- or by modifying the objectives via the Remote Control Interface (recommended)

  The Remote Control Interface is the most effective and simplest way to configure *snare.conf* and operates completely in memory, with no reliance on any external files. The Remote Control Interface can be access locally via the URL http://localhost:6161 or remotely via http://hostname:6161 where "hostname" is the DNS name or IP address of the target machine.

How to... Remote Audit Monitoring

The Remote Control Interface can be turned off by editing the default */etc/security/snare.conf* file. You can either edit the *snare.conf* file directly, commenting (using #) the `allow=1` line under the `[Remote]` section, or by setting this value to `0`. Save the file.

To ensure any changes to the snare.conf are applied, the agent must be restarted to active the new configuration. This restart process is shown as follows (execute as the root user):

```
>ps -ef |grep auditd
```

It should return something like:

```
root 17608 17595   0 13:50:56 pts/1       0:00 grep auditd
root 17606     1  33 13:47:52 ?           2:48 /usr/sbin/auditd
```

To restart:

```
>svcadm restart system/auditd
```

To check that the processes have restarted ensure the processes have new Ids:

```
>ps -ef |grep auditd
root 17633     1  32 14:12:40 ?           3:14 /usr/sbin/auditd
root 17637 17595   0 14:16:23 pts/1       0:00 grep auditd
```

Note: For administrators, the system log files will be updated whenever settings are applied to the snare.conf, for example, /var/log/messages. This information may assist you when you require it. Any errors in the configuration file will also be logged.

4 The Remote Control Interface

The Remote Control Interface is accessible by entering http://localhost:6161 in the web browser as shown inFigure 1. The Remote Control Interface is turned on by default, and also password protected for security reasons. The default username and password are:

Username: snare
Password: snare

NOTE: The default password is not encrypted in the configuration file. Ensure you change the default password immediately after installation as the new password will be encrypted for security purposes. Update the password via the Remote Control Configuration page.

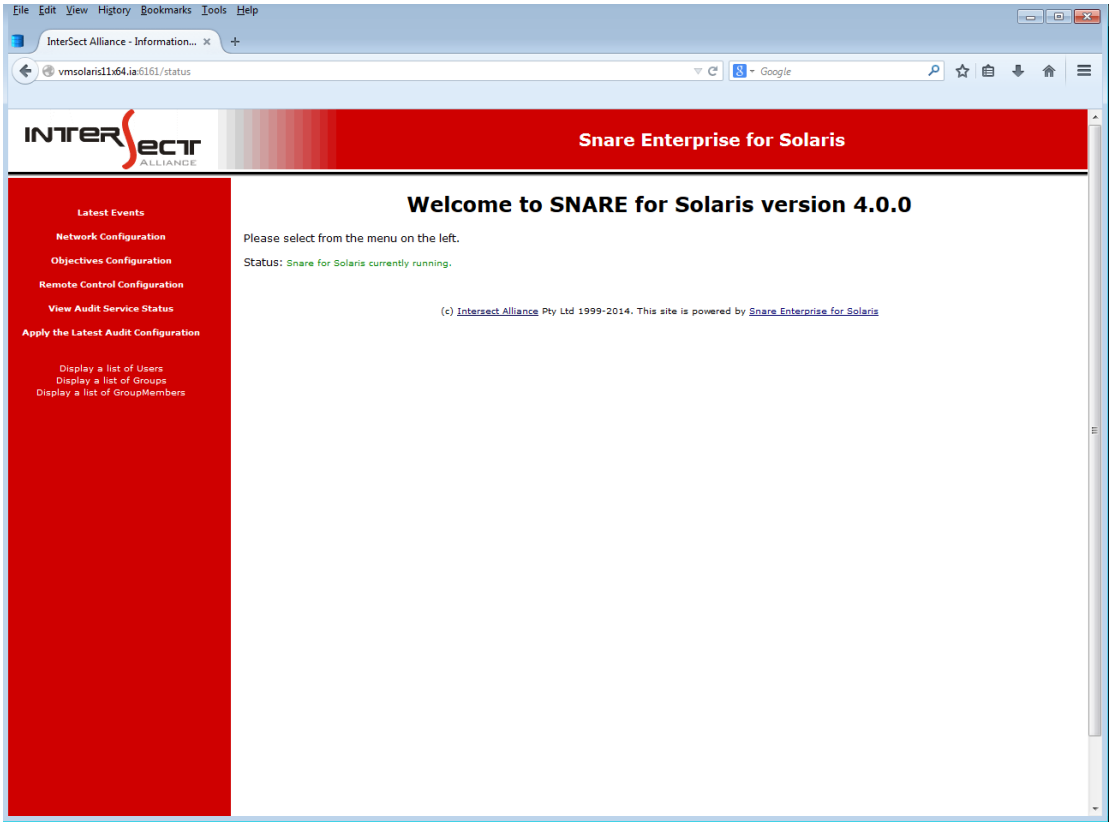


Figure 1: The Remote Control Interface-View Status

The Remote Control Interface provides a number of capabilities including:

- Network Configuration
- Remote Control Configuration
- Objectives Configuration
- Viewing Recent Events
- Displaying User and Group metadata.

4.1 Network Configuration

To set the network configuration parameters, select the 'Network Configuration' link.

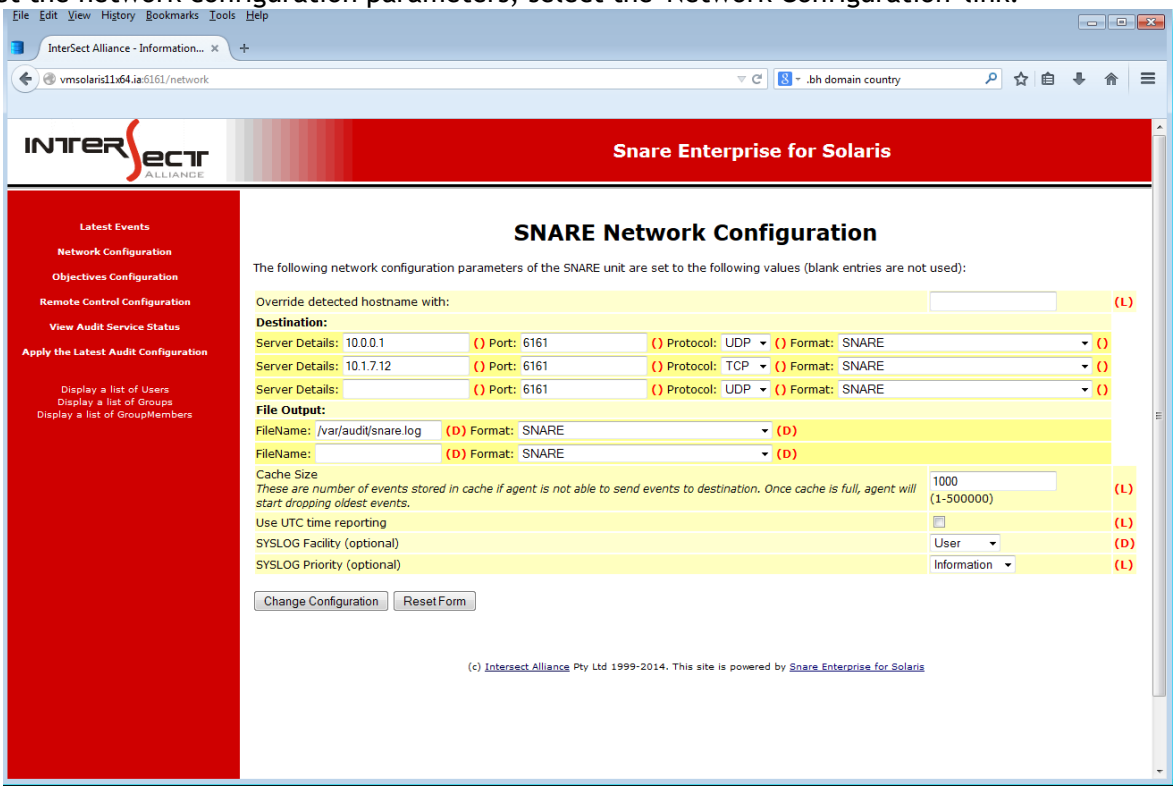


Figure 2: Configure the network settings

The configuration parameters available are as follows, as displayed inFigure 2:

- ***Override detected hostname with:*** Can be used to override the name that is given to the host. Unless a different name is required to be sent in the processed event log record, leave this field blank. The default is to use the fully qualified name for the machine.
- ***Destination:*** Snare can send audit events to one or more network destinations. Snare can send data either to a Snare-compatible server, or a SYSLOG compatible destination. Please be aware that most SYSLOG servers are incompatible with the extremely high volumes of data Snare is capable of generating.

  Server Details: Enter a DNS name, or IP address for each planned destination.

  Port: Select the port you would like Snare to use when sending events.

  Protocol: Select the protocol you would like Snare to use when sending events, UDP,TCP,SSL. Using TCP or SSL will guarantee message delivery. Using SSL will use an encrypted connection to the server.

  Format: Select this option if the requirement is that the event records need to be in a specific format. This feature will allow the event log record to be formatted so it is accepted by a Syslog or Snare server. Note: A warning message will be given if the format selected does not match the normal format for a port.

Click Change Configuration to add another destination. Likewise, to remove a destination, then delete the entry in the *Server Details* and click Change Configuration.

- ***File Output:*** Log events to a file. Please note the log files are to be managed, since Snare does not rotate the files. See Note.

  FileName: Log the output to disk as well as the network to a path with appropriate permissions.

  Format: Select the format for the file.

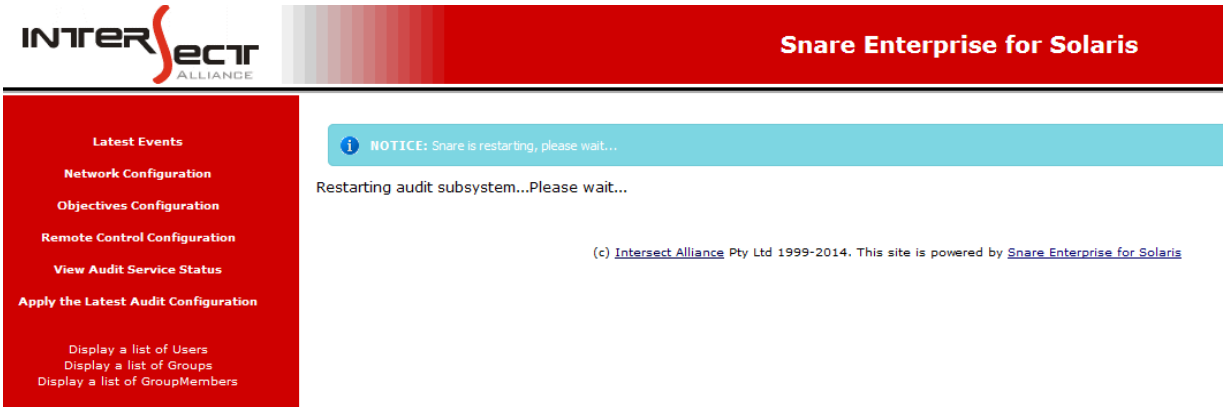
  Click Change Configuration to add another entry for file output.

  Note on Log Rotation: Depending on the Snare configuration the log file may be small or large. In any case it is normal housekeeping practice that loges either be rotated or archived. Depending on the site requirements, a rotation scheme that keeps old copies of the last 7 days may be sufficient. In this case, it may be sufficient to include a CRON 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. This may be scheduled using CRON or undertaken manually. In either case, it's important to note that the audit daemon should restart so that it opens the new log file for writing the events.

- ***Cache size:*** Allow Snare to store messages that could not be sent. Combined with the TCP or SSL this option will allow the agent to cache messages if there is a network failure or the Snare Server is otherwise unavailable. Any cached message is kept until it is sent or the size of the cache exceeds the specified allotment, in which case the oldest message is removed. If the agent is restarted cached messages are first saved to disk and reloaded upon restart.
- ***Use UTC time reporting:*** Enables UTC (Coordinated Universal Time) timestamp format for events instead of local machine time zone format.
- ***SYSLOG Facility (optional):*** If you are sending your data to a SYSLOG server, specifies the subsystem that produced the message. The list displays default facility levels.
- ***SYSLOG Priority (optional):*** If you are sending your data to a SYSLOG server, the agent can be configured to use a static or dynamic priority level.

To save and set changes to these settings, and to ensure the audit daemon has received the new configuration, perform the following:

1. Click on Change Configuration to save any changes.
2. Click on the **Apply the Latest Audit Configuration** menu item. There will be a quick notice that Snare is restarting as displayed below.



4.2 Remote Control Configuration

The Snare for Solaris agent can be controlled remotely by administrators, if required. Remote control is enabled by default. The remote control page is displayed in Figure 3.

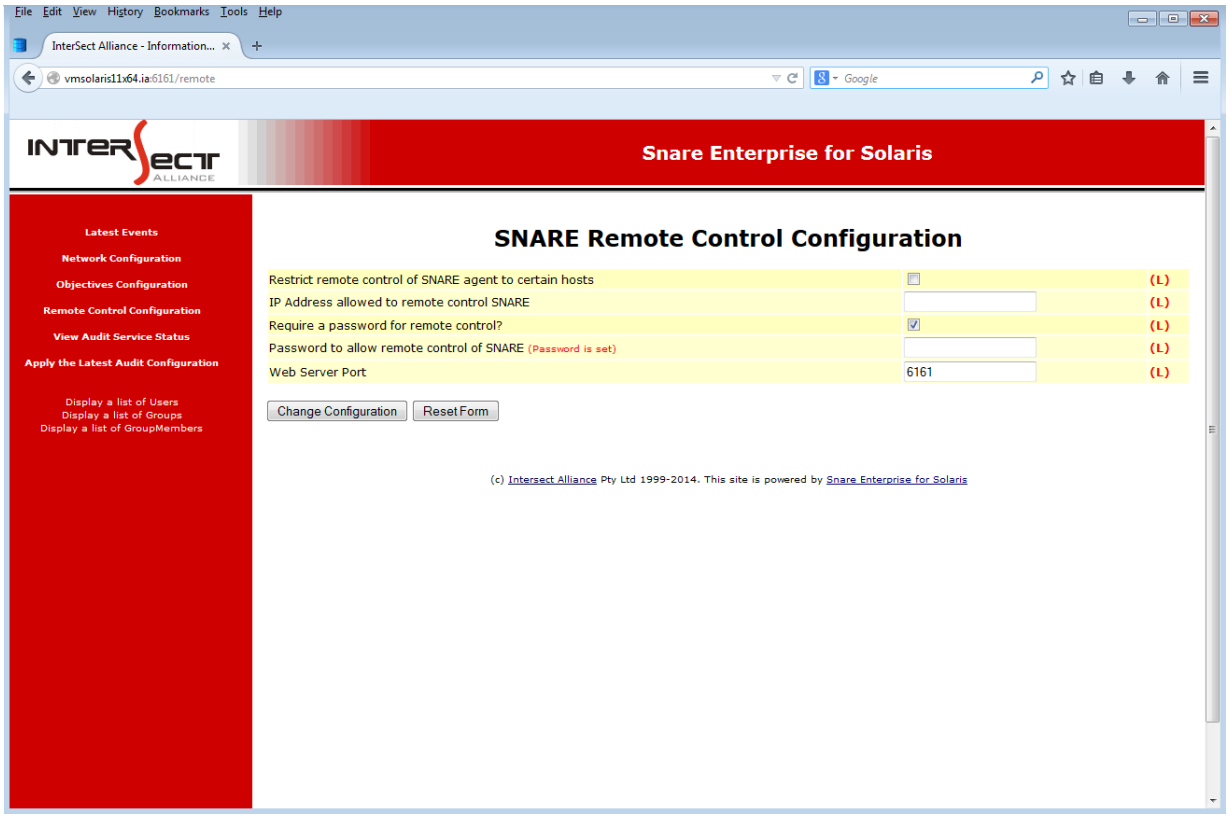


Figure 3: Configure the Remote Control

The parameters which may be set for remote control operation include:

- ***Restrict remote control of SNARE agent to certain hosts:*** By default Snare allows any IP address to connect to the remote control interface. Enabling this option restricts connections to the remote control interface to the IP given in the following option.
- ***IP Address allowed to remote control SNARE:*** Remote control actions may be limited to a given host. This host, entered as an IP address will only allow remote connections to be effected from the stated IP address. Application-level firewall capabilities are also available, which block users from accessing the Remote Control Interface from any IP address other than the one specified.
- ***Require a password for remote control?:*** Indicate whether a password will be set so that only authorised individuals may access the remote control functions. Highly recommended.
- ***Password to allow remote control of SNARE:*** If above checkbox is checked, password must be set. A password of appropriate strength should be set for the remote control facility. It is recommended you use a strong complex password of at least 12 characters. The password is not encrypted when being transmitted via the http session, but is encrypted when stored in the *snare.conf* file.
- ***Web Server Port:*** An optional port that the Remote Control Interface listens on, can be specified. Users of the Snare Server should generally leave this as 6161, in order to take advantage of the Snare Server's user and group audit capabilities. If the web server port conflicts with an established web server then this may be changed. However care should be taken to note the new server port, as it will need to be placed in the URL to access the Snare agent.

To save and set changes to these settings, and to ensure the audit daemon has received the new configuration, perform the following:

1. Click on Change Configuration to save any changes.
2. Click on the **Apply the Latest Audit Configuration** menu item.

To revert your changes to the last saved configuration click Reset Form.

4.3 Objectives configuration

Snare's ability to filter events is accomplished via the auditing 'objectives' capability. The term 'objective' is used within Snare Agents to describe an auditing goal. It is generally made up of events that Snare should watch for, a filter term containing a 'token' and a criticality level. A number of default objectives are available with the agent to aid you with PCI compliance. See Figure 4.

The objective configuration page supplied as part of the web based remote control is intended as a way to enable users to commence audit functions reasonably quickly.

For power users, a far more powerful and functional way is to manually control the */etc/security/snare.conf* file. This is described in more detail in Configuration File Description, and is intended for users who have a very detailed knowledge of Solaris administration and security. It is NOT recommended for novice users.

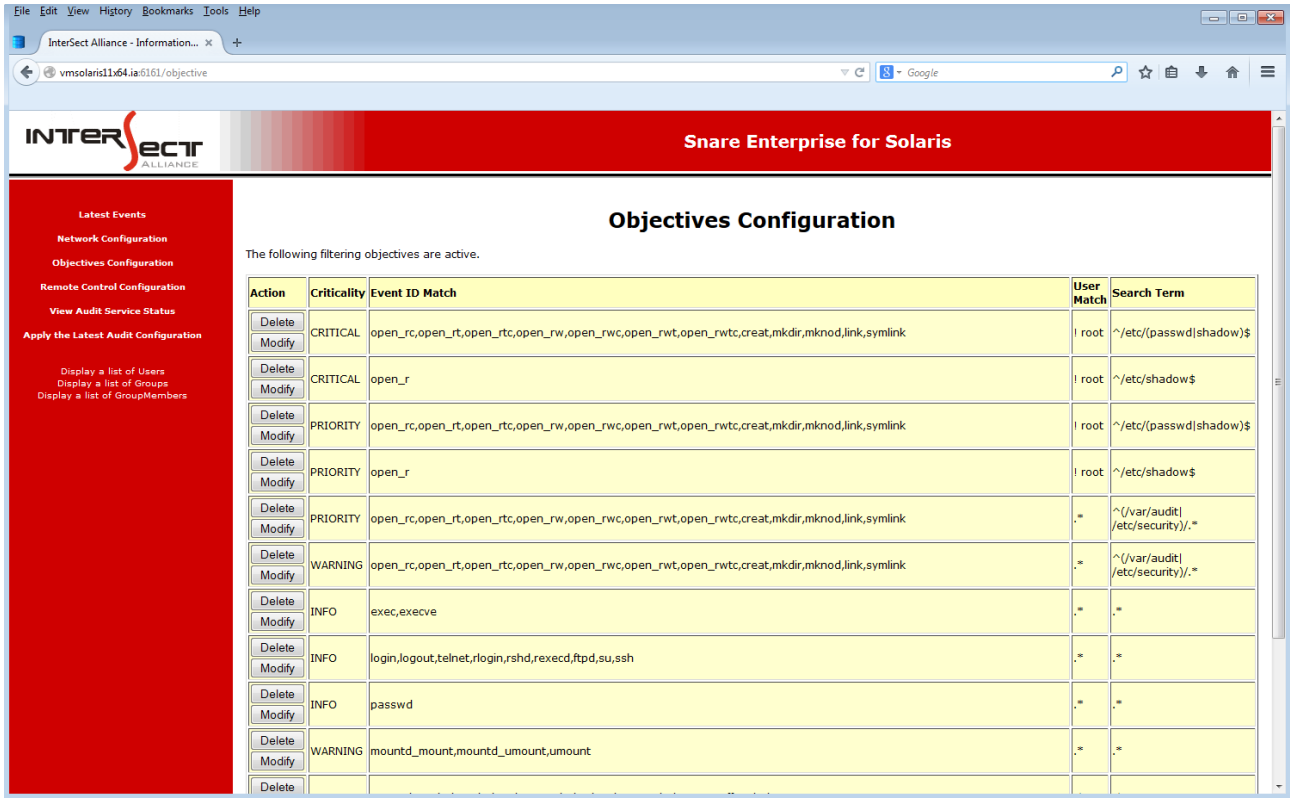


Figure 4: Display the Set objectives

Event Objectives

From the Objectives Configuration page, select 'Add' to insert an objective or 'Modify' to edit an existing objective. You may also select 'Delete' to remove any existing objectives. Generally, the order of objectives is not important, though it is good practice to have any objectives with exclusions at the top of the list.

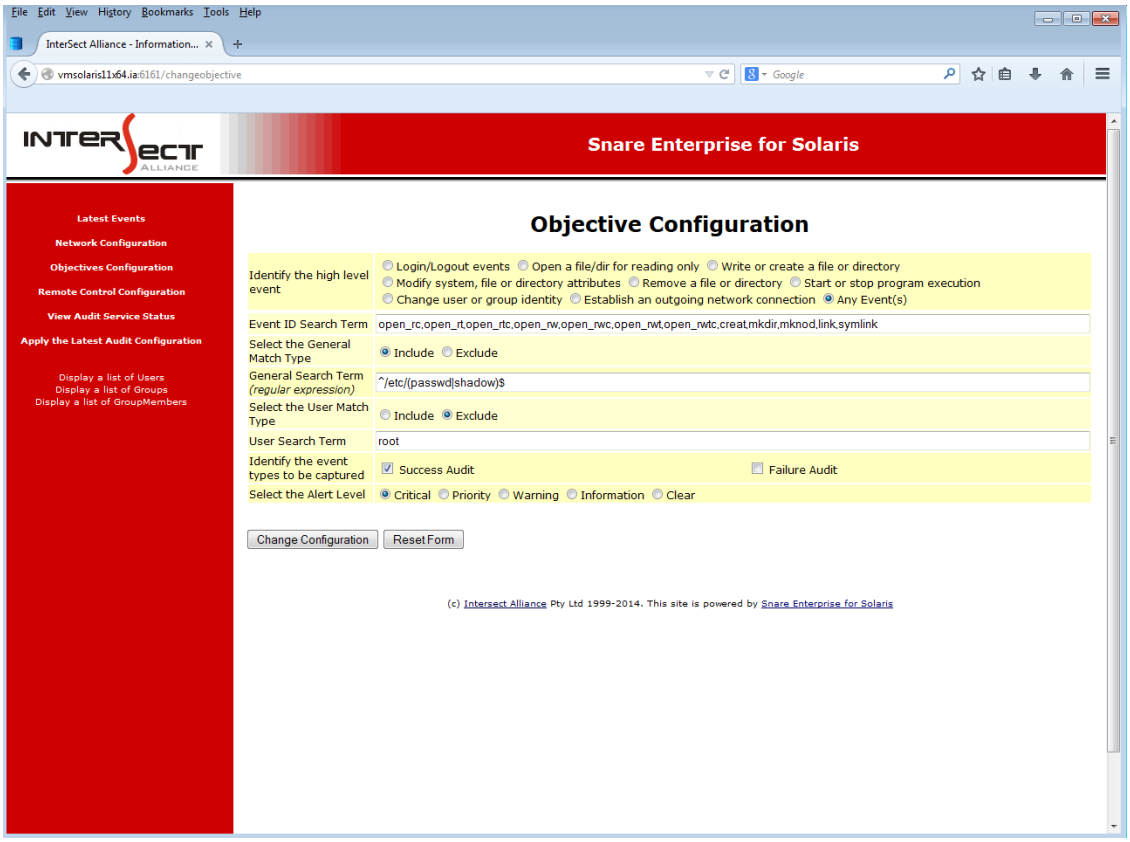


Figure 5: Adding/Modifying an Objective

The following parameters may be set as displayed inFigure 5:

- ***Identify the high level event:*** 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. The following groups are provided to service the most common security tasks required by a system or security administrator and are grouped as follows:
    - Login/Logout events
    - Open a file/dir for reading only
    - Write or create a file or directory
    - Modify system, file or directory attributes
    - Remove a file or directory
    - Start or stop program execution

- Change user or group identity
- Establish an outgoing network connection
- Any Event(s): any event that can be generated by the audit subsystem can be specified (comma separated).

Tip: Turning on file-related events can produce a very high volume of audit events on some systems, and therefore result in a considerable amount of CPU time being used by Snare and the audit subsystem.

The following filters may then be applied to incoming audit events:

- ***Event ID Search Term:*** If 'Any Event(s)' is selected as the high level event, then add a comma separated list of audit events to search for.
- ***Select the General Match Type:*** The search criteria that Snare should use to include or exclude the general search term event.
- ***General Search Term:*** A filter term containing a 'token' which appears within the events of interest, and the search criteria that Snare should use to include or exclude the event. Regular expressions may be used which are an advanced form of specifying filters. For example, a search term of: `/etc/.*` would match any event which mentions any file in `/etc`.

  For example, `'.*[Pp]ass(word|wd).*'` would match the following:

  `/etc/passwd`

  `/tmp/PasswordFile`

  but would not match

  `/etc/PASSWD/`

  `/home/red/PaSsWoRd.txt`

  Regular expression exclusions for example, `^/etc/[^(passwd)]` would match the following:

  `/etc/shadow`

  `/etc/group`

  `/etc/PASSWD`

  but would not match

  `/etc/passwd`

  To search for socketcalls enter the regular expression of `socket` to monitor connects, accepts etc.

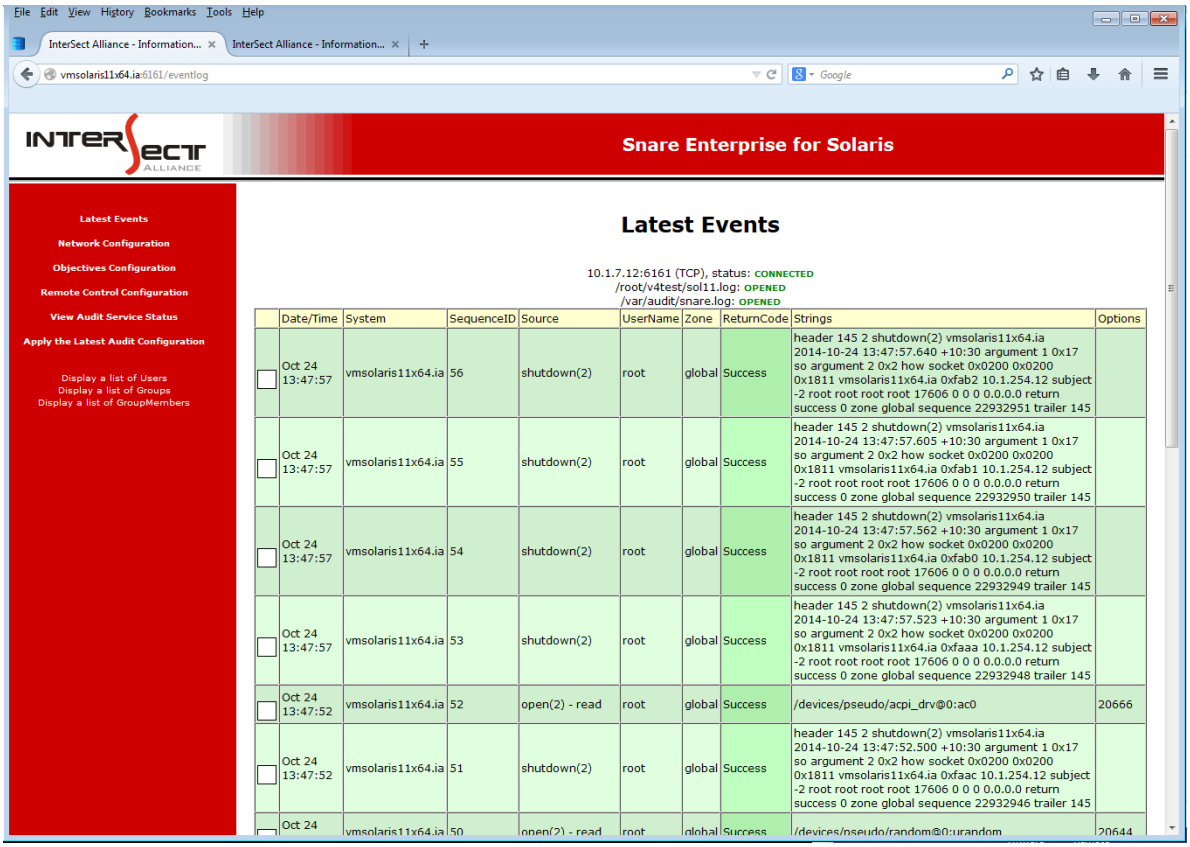
- ***Select the User Match Type:*** The search criteria that Snare should use to include or exclude the user search term event.
- ***User Search Term:*** A comma separated filter term containing users the objective should match. For example using: `root,snare` would cause the objective to match if users root or snare caused the event. Additionally the value `.*` may be used to match any user. If no users are entered, all users are assumed to be audited. If the user exclusion function is selected, then Snare will only report users that DO NOT match the supplied list of users.
- ***Identify the event types to be captured:*** Select the audit event types to capture, and includes Success Audit and Failure Audit.
- ***Select the Alert Level:*** The 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.

To save and set changes to these settings, and to ensure the audit daemon has received the new configuration, perform the following:

1. Click on Change Configuration to save any changes.
2. Click on the **Apply the Latest Audit Configuration** menu item.

4.4 Display of Latest Events / Destination Status

A small rotating cache of audit events is kept by the Snare for Solaris web server. Clicking on the **Latest Events** menu item will display twenty of the most recent events as displayed in Figure 7.



	Date/Time	System	SequenceID	Source	UserName	Zone	ReturnCode	Strings	Options
☐	Oct 24 13:47:57	vmsolaris11x64.ia	56	shutdown(2)	root	global	Success	header 145 2 shutdown(2) vmsolaris11x64.ia 2014-10-24 13:47:57.640 +10:30 argument 1 0x17 so argument 2 0x2 how socket 0x0200 0x0200 0x1811 vmsolaris11x64.ia 0xfab2 10.1.254.12 subject -2 root root root root 17606 0 0 0 0.0.0.0 return success 0 zone global sequence 22932951 trailer 145	
☐	Oct 24 13:47:57	vmsolaris11x64.ia	55	shutdown(2)	root	global	Success	header 145 2 shutdown(2) vmsolaris11x64.ia 2014-10-24 13:47:57.605 +10:30 argument 1 0x17 so argument 2 0x2 how socket 0x0200 0x0200 0x1811 vmsolaris11x64.ia 0xfab1 10.1.254.12 subject -2 root root root root 17606 0 0 0 0.0.0.0 return success 0 zone global sequence 22932950 trailer 145	
☐	Oct 24 13:47:57	vmsolaris11x64.ia	54	shutdown(2)	root	global	Success	header 145 2 shutdown(2) vmsolaris11x64.ia 2014-10-24 13:47:57.562 +10:30 argument 1 0x17 so argument 2 0x2 how socket 0x0200 0x0200 0x1811 vmsolaris11x64.ia 0xfab0 10.1.254.12 subject -2 root root root root 17606 0 0 0 0.0.0.0 return success 0 zone global sequence 22932949 trailer 145	
☐	Oct 24 13:47:57	vmsolaris11x64.ia	53	shutdown(2)	root	global	Success	header 145 2 shutdown(2) vmsolaris11x64.ia 2014-10-24 13:47:57.523 +10:30 argument 1 0x17 so argument 2 0x2 how socket 0x0200 0x0200 0x1811 vmsolaris11x64.ia 0xfaaa 10.1.254.12 subject -2 root root root root 17606 0 0 0 0.0.0.0 return success 0 zone global sequence 22932948 trailer 145	
☐	Oct 24 13:47:52	vmsolaris11x64.ia	52	open(2) - read	root	global	Success	/devices/pseudo/acpi_drv@0:ac0	20666
☐	Oct 24 13:47:52	vmsolaris11x64.ia	51	shutdown(2)	root	global	Success	header 145 2 shutdown(2) vmsolaris11x64.ia 2014-10-24 13:47:52.500 +10:30 argument 1 0x17 so argument 2 0x2 how socket 0x0200 0x0200 0x1811 vmsolaris11x64.ia 0xfaac 10.1.254.12 subject -2 root root root root 17606 0 0 0 0.0.0.0 return success 0 zone global sequence 22932946 trailer 145	
☐	Oct 24	vmsolaris11x64.ia	50	open(2) - read	root	global	Success	/devices/pseudo/random@0:urandom	20644

Figure 7: Display the latest events

Additionally this page shows the status for each Destination that was configured for logging. An example of this destination status is:

10.1.1.30:6161 (TCP), status: **CONNECTED**

This information can be used to help debug potential logging issues. The status can be explained as follows:

- **Host/Port:** e.g.: 10.1.1.30:6161
  The host ip/name and port that logs will be sent too.
- **Log destination Type:** e.g.: TCP
  The protocol of the remote connection. Possible values are TCP, UDP, SSL or File
- **The current State of the connection:** e.g.: CONNECTED
  This field indicates what snare is currently doing with the connection. You will see many different states including:
  - INITIAL - The remote log location is about to begin setup

- RESOLVING - DNS resolution for a hostname is occurring
- RESOLVE_DELAY(x) - DNS resolution failed, a retry will occur in X seconds
- CONNECTING - Snare is trying to connect to the destination
- CONNECT_FAILED - The connection to the destination failed
- CONNECT_DELAY(x) - Connecting to the remote end failed, it will be retried again in X seconds
- CONNECTED - Snare has an active connection to the destination
- SENDING - Snare is currently sending logs to the destination
- DISCONNECTED - The destination has disconnected the snare agent.. a reconnection will occur automatically.
- HANDSHAKE - A SSL/TLS Handshake is in progress
- HANDSHAKE_FAILED - The SSL/TLS Handshake failed
- OPENING - Opening a a file destination is in progress
- WRITING - Writing is occurring to a file
- WRITE_FAILED - A write to file failed
- CLOSED - A file has been closed

4.5 Retrieving User and Group Information

Snare has the ability to retrieve users, groups and group membership from accounts local to the host that is running the agent. In those cases where the primary authentication source is defined as a NIS+ domain or an LDAP directory, then the users, groups and group membership retrieved will be those defined in the remote domain.

Accessed from the menu items, selecting any of the items will then display the relevant details. For exampleFigure 8 displays the output of selecting 'Display a list of Users'. The output from these commands is designed with no HTML markup to assist automated services, such as the Snare Server, to interrogate the users, groups and groups membership.

The output shows a number of tab delimited entries per line. The entries may be interpreted as follows:

Username; ***UID***; ***Description***; ***Password Expired*** (token PASSWD_EXPIRED); Account Inactive (token ACCOUNT_INACTIVE); Account Expired (ACCOUNT_EXPIRED); Account Disabled (token ACCOUNT_DISABLED); Account Locked (token ACCOUNT_LOCKED); No Password (token NO_PASSWD).

The first three fields of ***Username***, ***UID and Description*** will be displayed and tab delimited. The remaining tokens are only visible if they exist on a particular account.

```
HOST: vmsolaris11x64.ia
root    0       Super-User
daemon  1               ACCOUNT_DISABLED
bin     2               ACCOUNT_DISABLED
sys     3               ACCOUNT_DISABLED
adm     4       Admin   ACCOUNT_DISABLED
lp      71      Line Printer Admin      ACCOUNT_DISABLED
uucp    5       uucp Admin      ACCOUNT_DISABLED
nuucp   9       uucp Admin      ACCOUNT_DISABLED
dladm   15      Datalink Admin  ACCOUNT_LOCKED
netadm  16      Network Admin   ACCOUNT_LOCKED
netcfg  17      Network Configuration Admin     ACCOUNT_LOCKED
smmsp   25      SendMail Message Submission Program     ACCOUNT_DISABLED
gdm     50      GDM Reserved UID        ACCOUNT_LOCKED
zfssnap 51      ZFS Automatic Snapshots Reserved UID    ACCOUNT_DISABLED
upnp    52      UPnP Server Reserved UID        ACCOUNT_DISABLED
xvm     60      xVM User        ACCOUNT_LOCKED
mysql   70      MySQL Reserved UID      ACCOUNT_DISABLED
openldap        75      OpenLDAP User   ACCOUNT_LOCKED
webservd        80      WebServer Reserved UID  ACCOUNT_LOCKED
postgres        90      PostgreSQL Reserved UID ACCOUNT_DISABLED
svctag  95      Service Tag UID ACCOUNT_LOCKED
unknown 96      Unknown Remote UID      ACCOUNT_LOCKED
nobody  60001   NFS Anonymous Access User       ACCOUNT_LOCKED
noaccess        60002   No Access User  ACCOUNT_LOCKED
nobody4 65534   SunOS 4.x NFS Anonymous Access User     ACCOUNT_LOCKED
aiuser  61      AI User ACCOUNT_LOCKED
ftp     21      FTPD Reserved UID       ACCOUNT_LOCKED
dhcpserv        18      DHCP Configuration Admin        ACCOUNT_LOCKED
pkg5srv 97      pkg(5) server UID       ACCOUNT_LOCKED
snare   100     snare
```

Figure 8: Output of Display a list of Users

In the case of Groups and Group Membership, the attributes displayed are ***Groupname***, ***GID and Group Members***. The remaining tokens are only visible if they exist on a particular account.

5 Snare Server

The Snare Server is a log collection, analysis, reporting, forensics, and storage appliance that helps your meet departmental, organisational, industry, and national security requirements and regulations. It integrates closely with the industry standard Snare agents, to provide a cohesive, end-to-end solution for your log-related security requirements.

The Snare Server, as shown below collects events and logs from a variety of operating systems, applications and appliances including, but not limited to: Windows (NT through 2012), Solaris, AIX, OSX, Irix, Solaris, Tru64, ACF2, RACF, CISCO Routers, CISCO PIX Firewall, CyberGuard Firewall, Checkpoint Firewall1, Gauntlet Firewall, Netgear Firewall, IPTables Firewall, Microsoft ISA Server, Microsoft IIS Server, Lotus Notes, Microsoft Proxy Server, Apache, Squid, Snort Network Intrusion Detection Sensors, IBM SOCKS Server, and Generic Syslog Data of any variety.

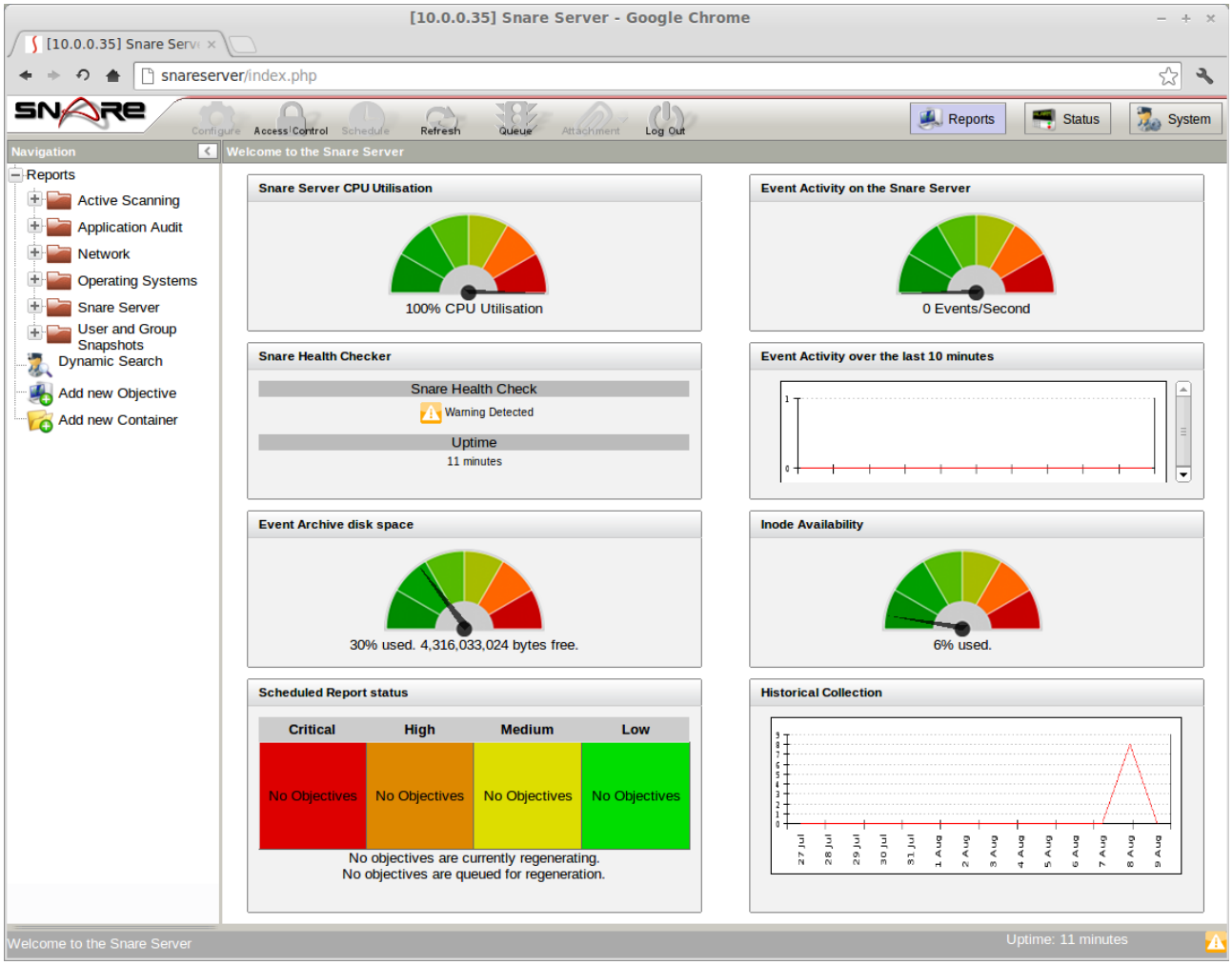


Figure 9:Welcome to the Snare Server (version 6.0)

Some of the key features of the Snare Server include:

- Ability to collect any arbitrary log data, either via UDP or TCP
- Secure, encrypted channel for log data using TLS/SSL
- Proven technology that works seamlessly with the Snare agents
- Snare reflector technology that allows for all collected events to be sent, in real time, to a standby/backup Snare Server, or a third party collection system
- Ability to continuously collect large numbers of events. Snare Server collection rates exceed 60,000 events per minute using a low end, workstation class, Intel based PC on a 100Mbps network.
- Ability to drill down from top level reports. This reduces the amount of data “clutter” and allows a system administrator to fine tune the reporting objectives.
- Ability to 'clone' existing objectives in order to significantly tailor the reporting criteria. These reports, along with all Snare Server objectives, may be scheduled and emailed to designated staff.
- The Snare Server uses extensive discriminators for each objective, allowing system administrators to finely tune reporting based on inclusion or exclusion of a wide variety of parameters.
- Very simple download and installation
- Flexibility when dealing with unique customer requirements
- A strategic focus on low end hardware means that Snare can achieve outstanding results with minimal hardware cost outlay
- Snare gives you useful data, out of the box, with default objectives tuned for common organisational needs
- Ability to manage Enterprise Agents
- All future Snare Server versions and upgrades included as part of an annual maintenance fee.

The Snare Server is an appliance solution that comes packaged with a hardened, minimal version of the Solaris operating system to provide baseline computing functionality, which means you do not need to purchase additional operating system licenses, database licenses, or install additional applications in order to get up and running. Like your android phone, or your home router, any operating-system level management and maintenance is either automated, or is available within the web-based interface.

For further information on the Snare Server refer to the *Snare Server User Guide* on the Intersect Alliance website.

6 About InterSect Alliance

Intersect Alliance, part of the Prophecy International Holdings Group, is a team of leading information technology security specialists. 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 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.

Intersect Alliance welcomes and values your support, comments, and contributions.

For more information on the Enterprise Agents, Snare Server and other Snare products and licensing options, please contact us as follows:

The Americas +1 (800) 834 1060 Toll Free | +1 (303) 771 2666 Denver

Asia Pacific +61 8 8213 1200 Adelaide Australia

Europe and the UK +44 (797) 090 5011

Email intersect@intersectalliance.com

Visit www.intersectalliance.com

Appendix A - Configuration File Description

The purpose of this section is to discuss the parameter settings of the Snare configuration file located at */etc/security/snare.conf*, and this location may not be changed. If the configuration file does not exist, the audit daemon will not actively audit events until a correctly formatted configuration file is present.

Snare can be configured in several different ways, namely:

a. Via the Remote Control Interface (*recommended*), or
b. By manually editing the configuration file (*advanced users only*).

The format of the **audit configuration file** is discussed below. Any line beginning with "#" will be treated as a comment line and ignored. Any number of tabs or spaces can be used. Major tokens such as `[Config]` must be surrounded by the square brackets.

`[Config]`	This section allows you to specify settings relating to the operation of the Snare agent.
`set_audit=[1|0]`	This value determines if Snare should set the auditing configuration for the local machine.
`clientname=override`	The hostname of the client. If no hostname is set, the value of "hostname --fqdn" will be used
`cache_size=(0 - 100000)`	This value determines the size of the event cache,ie; the number of events, that Snare should keep if it cannot reach at least one of the hosts. The value must be between 0 and 100000. This feature only appears in Enterprise Agents only.
`use_utc=[1|0]`	Enable UTC (Universal Coordinated Time). This feature only appears in Enterprise Agents only.
`syslog_facility=facility`	The SYSLOG facility used when sending to a SYSLOG server.
`syslog_priority=priority`	The SYSLOG priority used when sending to a SYSLOG server.
`AgentLog=0`	To be added to the Remote Control Interface as a setting in the future release of version 5.0 of the Snare for Solaris agent.
`HeartBeat=0`	To be added to the Remote Control Interface as a setting in the future release of version 5.0 of the Snare for Solaris agent.

[Remote]	This section allows you to specify settings relating to the Remote Control Interface used to control Snare.
allow=[1|0]	Turn the Remote Control Interface on or off.
listen_port=6161	Set a port that the Snare for Solaris agent should listen on.
accesskey_enabled=[1|0]	Password is required to be set
accesskey=md5password	Md5 checksum of the password used to protect the embedded web server
restrict_ip_enabled=0	Restrict the Remote Control Interface to an IP.
restrict_ip=1.2.3.4	IP address of a system that is used to remotely control the agent. All requests from other systems will be dropped.

[Output]	By default, if no output section exists within the configuration file, the audit daemon will not send any data to anywhere. Otherwise, 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, or to a remote network destination
network=hostname:port:protocol:format	Data will be sent to the remote host, and network port specified here. Audit data can be sent to a remote system using the UDP or TCP protocol. SSL may also be used to indicate an encrypted TCP connection. Format may be either SNARE or SYSLOG. E.g networkOutput0=10.1.1.30:6161:TCP:SNARE
file=/fully/qualified/file/name	The audit daemon will send data to the fully qualified filename. The *directory* must exist. The *file* will be created if it doesn't exist. E.g file=/var/log/filewatch.log

`[Objectives]`	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". Any integer less than 0 will cause the line to be rejected. 2. The event - this must either correspond to a valid syscall event, or a series of events separated by commas, and may be surrounded with round brackets (). Note that the embedded web server will convert the generic "groups" in the Audit Configuration window to the required events. For example, the abstracted group 'Administrative Events', will result in the event entry: 'event=(*reboot,settimeofday,clock_settime, setdomainname,sethostname*)' being written. 3. Return - either Success, Failure or * to indicate both Success and Failure 4. User - The users(s) to watch. This can be a single user, a list of users separated with commas or * to indicate all users 5. match - An optional string to match. This can be either a string literal, a regular expression or .* to indicate all events Note that whitespace will be trimmed from the start and end of items.
`criticality=1 event=execve return=Success user=maria match=/sbin`	Report at criticality level 1, whenever the user 'maria', attempts to execute a binary within /sbin, criticality=`0` for Clear (ordinary security level), `1` for Information, `2` for Warning, `3` for Priority, `4` for Critical.

Shown below is an example */etc/security/snare.conf* file. It is an example file only, and should NOT be used for operational purposes. It has been included to demonstrate the key concepts of formulating a snare.conf file, as discussed above.

Example snare.conf file

```
#This is a comment line with no leading spaces
[Config]
use_criticality=1
set_audit=1
encrypt_msg=0
AgentLog=0
HeartBeat=0
clientname=
cache_size=1000
use_utc=0
syslog_facility=1
syslog_priority=6

[Output]
networkOutput0=10.1.7.12:6161:TCP:SNARE
fileOutput0=/var/audit/snare.log
fileOutput1=/root/v4test/sol11.log

[Remote]
accesskey_enabled=1
restrict_ip_enabled=0
allow=1
listen_port=6161
accesskey=snare

[Objectives]
criticality=4                match=^/etc/(passwd|shadow)$                        user!=root
event=open_rc,open_rt,open_rtc,open_rw,open_rwc,open_rwt,open_rwtc,creat,mkdir,mknod,li
nk,symlink       return=success
criticality=4    match=^/etc/shadow$      user!=root       event=open_r     return=success
criticality=3                match=^/etc/(passwd|shadow)$                        user!=root
event=open_rc,open_rt,open_rtc,open_rw,open_rwc,open_rwt,open_rwtc,creat,mkdir,mknod,li
nk,symlink       return=failure
criticality=3    match=^/etc/shadow$      user!=root       event=open_r     return=failure
criticality=3              match=^(/var/audit|/etc/security)/.*                  user=.*
event=open_rc,open_rt,open_rtc,open_rw,open_rwc,open_rwt,open_rwtc,creat,mkdir,mknod,li
nk,symlink       return=success
criticality=2              match=^(/var/audit|/etc/security)/.*                  user=.*
event=open_rc,open_rt,open_rtc,open_rw,open_rwc,open_rwt,open_rwtc,creat,mkdir,mknod,li
nk,symlink       return=failure
criticality=1    match=.*        user=.* event=exec,execve       return=*
criticality=1                match=.*                                            user=.*
event=login,logout,telnet,rlogin,rshd,rexecd,ftpd,su,ssh        return=*
criticality=1    match=.*        user=.* event=passwd    return=*
criticality=2     match=.*              user=.*  event=mountd_mount,mountd_umount,umount
return=*
criticality=2                match=.*                                            user=.*
event=systemboot,halt_solaris,reboot_solaris,shutdown_solaris,poweroff_solaris
return=*
criticality=2                match=.*                                            user=.*
event=audit,auditon_stermid,auditon_setstat,auditon_setstat,auditon_setumask,auditon_se
tsmask,auditon_setcond,auditon_setclass  return=*
criticality=0    match=.*        user=.* event=open_r,readlink    return=*
criticality=0                match=.*                                            user=.*
event=open_rc,open_rt,open_rtc,open_w,open_wc,open_wt,open_wtc,open_rw,open_rwc,open_rw
t,open_rwtc,creat,mkdir,mknod,xmknod,link,symlink,rmdir,unlink,rename,truncate,ftruncat
e return=*
criticality=0       match=.*                 user=.*   event=connect,shutdown,setsockopt
return=*
```

Appendix B - Event Output Format

The ***audit_snare plugin*** reads data from the Solaris kernel, via auditd. It converts the binary audit data into text format, 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 C - Known BSM 'bugs'

The Solaris BSM subsystem has a number of 'bugs' (or 'features') which have been identified and documented in the course of the Snare for Solaris development process. These are detailed below:

- The 'auditon' system call ONLY turns on system-related audit events (ie. events with an ID less than 512).(Solaris 10)
- Forked processes do NOT inherit audit settings. Each process is re-applied with the contents of /etc/security/audit* at startup.(Solaris 10)
- auditconfig -conf makes a range of decisions w.r.t the processes that need to have auditing applied, but it's decision making process is opaque, and some processes (including nscd, thankfully), are missed out. This leads to some reasonably important processes (such as the mount daemon, and openssh) being ignored by the Solaris BSM subsystem.(Solaris 10 and Solaris 11)

Appendix D - Configuring Solaris Plugins

Solaris 10 and 11 both use plugins.

With ***Solaris 10***, when we install the audit_control file we indicate to use the Snare plugin by default:

```
#
#ident  @(#)audit_control.txt  1.4     2005/11/24 LJP
#
# audit_control file for snare
#
dir:/var/audit
flags:ia
minfree:20
naflags:ia
plugin:name=/usr/lib/security/audit_snare.so
```

If logging to local disk is also required, either Snare can be used to log to file in Syslog or Snare format, or the standard Solaris audit logs can be logged by using the binfile plugin, like so:

```
#
#ident  @(#)audit_control.txt  1.4     2005/11/24 LJP
#
# audit_control file for snare
#
dir:/var/audit
flags:ia
minfree:20
naflags:ia
plugin:name=/usr/lib/security/audit_snare.so
plugin: name=audit_binfile.so;\
p_minfree=20;\
pdir=/var/audit/
```

This particular setup will put the standard Solaris audit logs into the /var/audit directory, limiting to 20MB free.

Solaris 11 uses the Service Manager to handle the plugins rather than the audit_control file. The auditconfig command can be used to configure this:

```
#auditconfig -plugin
Plugin: audit_binfile (active)
        Attributes: p_dir=/var/audit;p_fsize=0;p_minfree=1

Plugin: audit_snare (active)
        Attributes:
        Queue size: 1
```

Further details for configuring Solaris 11 plugins can be found in the Solaris 11 documentation.

Appendix E - Filter on special or specific event

The following filter, filter on special, event-specific fields, can be applied to incoming audit events only available via the configuration file.

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.