Global Information Assurance Certification Paper

Logging and Monitoring to Detect Network Intrusions and Compliance Violations in the Environment

GIAC (GCIA) Gold Certification

Author: Sunil Gupta, sgupta911@gmail.com
Advisor: Dr. Kees Leune

Accepted: July 4, 2012

Abstract

Log Management and Intrusion Detection solutions have been evolving for years. Yet, it remains a challenge for organizations of all sizes to meet the operational, audit and security needs using these solutions. This paper presents a solution to bridge logging, log based intrusion detection and network based intrusion detection using well known free open source tools available on the Security Onion Linux Distribution. It walks through the logging, monitoring and alerting approach necessary for security, compliance and quality of service. In the process it provides for cost effective, customizable and scalable solution alternative to vendor based Security Information & Event Management (SIEM) solutions.

1. Introduction

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. An intrusion detection system (IDS) is software that automates the intrusion detection process. Network-Based IDS (NIDS) monitors network traffic for particular network segments or devices and analyzes the network and application protocol activity to identify suspicious activity (Scarfone & Mell, 2007).

Security Log Analysis Systems are also known as Log-based Intrusion Detection Systems (LIDS). Log Analysis For Intrusion Detection is the process or techniques used to detect attacks on a specific environment using logs as the primary source of information. LIDS is also used to detect computer misuse, policy violations and other forms of inappropriate activities (Cid, 2007).

The main thesis of this paper is that NIDS and LIDS are necessary for effectively monitoring the security posture of an organization. Both techniques, network-based detection and log-based detection, complement each other in the identification and reporting of security incidents.

1.1. Outline

This paper describes how to build a system that combines Network Based Intrusion Detection with Log Based Intrusion Detection to create a comprehensive security monitoring platform. Chapter 2 provides an overview of essential terminology in the field of Security Information Event Monitoring and Log Management. Chapter 3 builds on the terminology by proposing a technical architecture and by providing configuration guidance. Chapter 4 discusses Log Analysis and Correlation and the paper concludes by discussing Alerting and Reporting in Chapter 5.

This paper describes a fictional scenario in which an intrusion is detected using NIDS and LIDS alerts on the monitor console. The example demonstrates the value of this approach by following an intruder performing a network scan, connect to a system and control gain, followed by privilege escalation on the target system.

Sunil Gupta, sgupta911@gmail.com

1.2. Problem Addressed

In an organization, there are many possible signs of incidents which may go unnoticed each day. These events can be studied mainly by analyzing network behavior or by reviewing computer security event logs. In order to avoid or minimize the losses from an incident outcome, the events need to be analyzed as close to real-time as possible. Logging and intrusion detection systems have the potential to produce very large amount of data, and all that data must be managed, filtered and analyzed. Having a single approach and a unified platform helps with this very difficult and challenging task to monitor and report in near-real time.

Automation is needed to perform an initial analysis of the data and to alert on select events of interest for human review. Event correlation software and centralized logging can be of great value in automating the analysis process. However, the effectiveness of the process depends on the quality of the data and the data rules that goes into it.

2. Log Management and SIEM Overview

The NIST Guide to Computer Security Log Management (Kent & Souppaya, 2006) states that information regarding an incident may be recorded in several places, such as firewalls, routers, network IDS, host IDS, and application logs. Organizations should deploy one or more centralized logging servers and configure logging devices throughout the organization to send duplicates of their log entries to the centralized logging servers. A log management infrastructure consists of the hardware, software, networks and media used to generate, transmit, store, analyze, and dispose of log data. This section describes the typical architecture and functions of a log management.

2.1. Log Management Architecture

The NIST Guide to Computer Security Log Management (Kent & Souppaya, 2006) explains that a log management infrastructure typically comprises of three tiers: log generation, log analysis and storage, and log monitoring.

The *log generation tier* involves hosts making their logs available to log servers in the second tier. This is performed in two different ways. The exact method depends on the log type, and, on the host and network controls. In one way hosts run some services to send their log data over the network to log collection servers. Alternatively, hosts allow the log servers to pull the log data from them. The logs are often transferred to the log receivers either in a real-time or near-real-time manner, or in occasional batches based on a schedule.

The *log analysis and storage tier* is composed of one or more log servers receiving log data from the hosts. These log receivers are also called collectors or aggregators. To facilitate log analysis, automated methods of converting logs from multiple formats to a single standard format needs to be implemented. Syslog format of logging is often used for this purpose.

The *log monitoring tier* contains consoles that are used for monitoring and reviewing of log data and the results of automated analysis. Report generation, management dashboards and log baselines may also be done using consoles as part of this tier.

The scope of log management infrastructure can be dictated by many factors, including the organization's internal structure, system types (e.g., a separate infrastructure for enterprise security systems), log types (e.g., a separate infrastructure for application audit logs), and facility locations.

2.2. Log Management Functions

Log management infrastructures typically perform several functions that assist in the storage, analysis, and disposal of log data. These functions are normally performed in such a way that they do not alter the original logs (Kent & Souppaya, 2006).

General functions of log management infrastructure include log parsing, event filtering and event aggregation. On the storage side, log management has to provide for log rotation, log archival, log compression, log reduction, log conversion, log normalization and log file integrity. Event correlation, log viewing and log reporting are some of the analysis functions of a log management infrastructure.

Kent & Souppaya (2006) also explain that a log management infrastructure usually encompasses most or all of the functions described in this section. The placement of some of the log functions among the three tiers of the log management infrastructure depends primarily on the type of log management software used. It is in the best interest of organizations to have appropriate auditing in place that allows for effective and efficient log management.

2.3. SIEM

Security information and event management (SIEM) software provides the log management infrastructure encompassing log analysis, log storage and log monitoring tiers. What sets SIEM products apart from traditional log management software is the ability to perform event correlation, alerting, incident management, reporting and forensic investigation based on event analysis. There are many SIEM solutions commercially available today and these solutions provide different set of these features and additional add-ons.

According to Gartner description of the SIEM (Nicolett & Kavanagh, 2012), SIEM technology aggregates the event data produced by security devices, network devices, systems and applications. The primary data source is log data, but SIEM technology can also process other forms of data. Event data is combined with contextual information about users, data and assets. The data is normalized, so that events from disparate sources can be correlated and analyzed for specific purposes, such as network security event monitoring, user activity monitoring or compliance reporting. The technology provides real-time security monitoring, historical analysis, and other support for incident investigation and compliance reporting.

2.4. Log Management Benefits

Log events are the primary records of system and network activity. In the SANS Log Management Survey, Shank (2010) provides an overview of typical reasons why log management is used in an organization. In the order of importance:

- Detect/Prevent Unauthorized Access and insider Abuse
- Meet Regulatory Requirement
- Forensic Analysis and Correlation
- Ensure Regulatory Compliance

- Track Suspicious Behavior
- IT Troubleshooting and Network Operation
- Monitor User Activity
- Best Practices/Frameworks such as COBIT, ISO, ITIL, etc.
- Deliver Reports to Departments
- Measure Application Performance
- Achieve ROI or Cost Reduction in System Maintenance

3. Proposed Architecture

Organizations should establish logging standards and procedures to ensure that adequate information is collected by logs and security software and that the data is reviewed regularly. The length of time to maintain log data is dependent on several factors, including the organization's data retention policies and the volume of data. As each business has different needs and regulatory requirements, legal counsel should be obtained to determine the appropriate retention schedule for logs.

This paper uses the Security Onion (SO) live CD distro created by Doug Burks for setting up of the logging and monitoring system. Snort is used as the intrusion detection engine from the two different kinds of intrusion detection engines, Snort and Suricata, available on SO. Sguil, Squert and Snorby provide the management console to view and classify sensor alerts. OSSEC's ability for log analysis, integrity checking, rootkit detection, real-time alerting and active response across platforms makes it an excellent choice for host based intrusion detection. This paper utilizes OSSEC as the log collector on the SO monitor to archive logs as well as review log files in near real time, while scrutinizing them for known attack patterns.

3.1. Security Onion

Security Onion (SO) is a Linux distribution for IDS (Intrusion Detection) and NSM (Network Security Monitoring). It is based on Xubuntu 10.04 and contains Snort®, Suricata, Sguil, Snorby, Squert, argus, Xplico, tcpreplay, scapy, hping, and many other security tools. Security Onion makes it phenomenally easy to join lots of pieces in the jigsaw puzzle in set up and integration of these tools (Burks, 2012). Some of the major components of SO used in this document are described here.

3.1.1. Sguil

Sguil (pronounced sgweel) is probably best described as an aggregation system for network security monitoring tools. Sguil's main component is an intuitive GUI that provides access to real-time events, session data, and raw packet captures. Sguil facilitates the practice of Network Security Monitoring and event driven analysis (Sguil, 2012). When an alert that needs more investigation has been identified, the Sguil client provides seamless access to the data that is needed to make a decision as how to handle the situation. Sguil uses a database backend for most of its data, which allows users to perform SQL queries against several different types of security events (nsmwiki.org, 2012).

3.1.2. Squert

Squert is a web application that is used to query and view event data stored in a Sguil database (typically IDS alert data). Squert is a visual tool that attempts to provide additional context to events through the use of metadata, time series representations and weighted and logically grouped result sets (Squert, 2012).

3.1.3. Snort

Snort is an open source network intrusion prevention and detection system (IDS/IPS) developed by Sourcefire. Combining the benefits of signature, protocol, and anomaly-based inspection, it is the most widely deployed IDS/IPS technology worldwide (Snort, 2012).

3.1.4. Snorby

Snorby is a front end web application (scripted in Ruby on Rails) for any application that logs events in the unified2 binary output format. Snorby integrates with intrusion detection systems like Snort, Suricata and Sagan. The basic fundamental concepts behind Snorby are simplicity and power. The project goal is to create a free, open source and highly competitive application for network monitoring for both private and enterprise use (Snorby, 2012).

3.1.5. OSSEC

OSSEC is an Open Source Host-based Intrusion Detection System (HIDS). It performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, real-time alerting and active response. It runs on most operating systems, including Linux, OpenBSD, FreeBSD, Mac OS X, Solaris and Windows (OSSEC, 2012).

3.1.6. ELSA

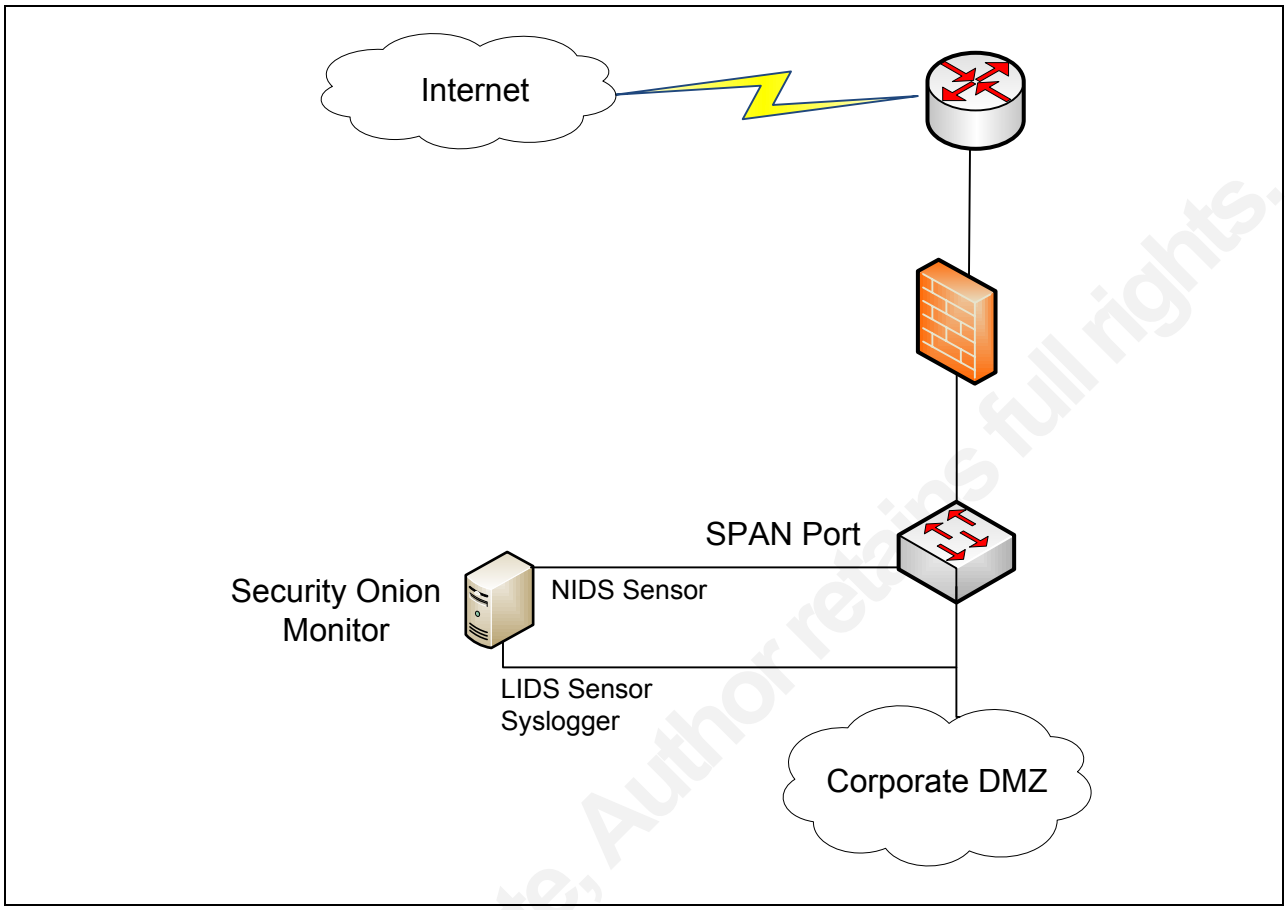
Enterprise Log Search and Archive (ELSA) is a centralized syslog framework built on Syslog-NG, MySQL, and Sphinx full-text search. It provides a fully asynchronous web-based query interface that normalizes logs and makes searching billions of them for arbitrary strings as easy as searching the web (Holste, 2012). ELSA integration into the Security Onion is in QA now and might be integrated by the time this paper is out. ELSA integration into Security Onion will provide a SIEM alternative out of the box.

3.2. Security Monitoring Architecture with Security Onion

The Security Onion Live Distribution enables a monitoring system to be set up and provides a convenient setup shortcut on the screen to configure the NSM infrastructure. Detailed installation steps are shown in Appendix A. A monitoring system should have enough resources such as memory, processor and disk space to handle the network throughput and to meet the log archive needs of the environment.

There are no documented hardware requirements for the Snort/Sguil sensor. Hardware sizing of the monitoring system mainly depends on the network throughput of the monitored links, number of sensors, types of rules, number of rules, preprocessor configuration and output plug-ins. For the network throughput of 200 Mbps or slower, we were able to use server virtual platform with dual core, 2 GHz processor, 4 GB ram for the dual sensor approach. Storage requirements also vary based on the network traffic, amount of logging and archival needs of the environment.

The deployment model presented in this section follows a two sensor deployment strategy on the same monitor. We call them LIDS sensor and NIDS sensor based on their functions. NIDS sensor network interface monitors the network using switch spanning port. LIDS sensor monitors syslog log traffic to the destination syslog collector network interface. This figure below illustrates the configured layout.



Setup involves multiple hosts generating the different kinds of application and system log traffic and sending it to the central log collector. Syslog messages are forwarded by syslog client, for example syslog-ng, Snare, or Adiscon event reporter on the client system using a defined syslog port. Syslog forwarding examples and configurations are shown in Appendix C. The effectiveness of the log analysis process depends on the quality of the log data that goes into it. The Network Time Protocol (NTP) is used to synchronize clocks among all hosts with the reliable source.

OSSEC is installed as host intrusion detection system on the monitoring system. OSSEC's `ossec-logcollector` process is enables as syslog collector on the monitor. Rsyslog or another tool such as syslog-ng can be used as syslog collector on SO, if OSSEC log collector is required to run in secure mode to collect logs for the OSSEC agents. LIDS sensor can utilize OSSEC or Snort based rules based on the log type and rules to detect the events of interest.

Sunil Gupta, sgupta911@gmail.com

Snort rules are customized differently for both LIDS and NIDS sensors to detect system intrusions, compliance anomalies and other events of interest. Sensor generated alerts are stored in the Sguil MySQL database which is viewed using Sguil GUI or Squert web based UI. Sensor alerts are also stored in the Snorby MySQL database which is accessed with Snorby web interface. Sguil, Squert and Snorby data can be accessed remotely from different system. SO becomes a powerful log management and intrusion detection platform with the inclusion of NIDS, LIDS functions and, log repository.

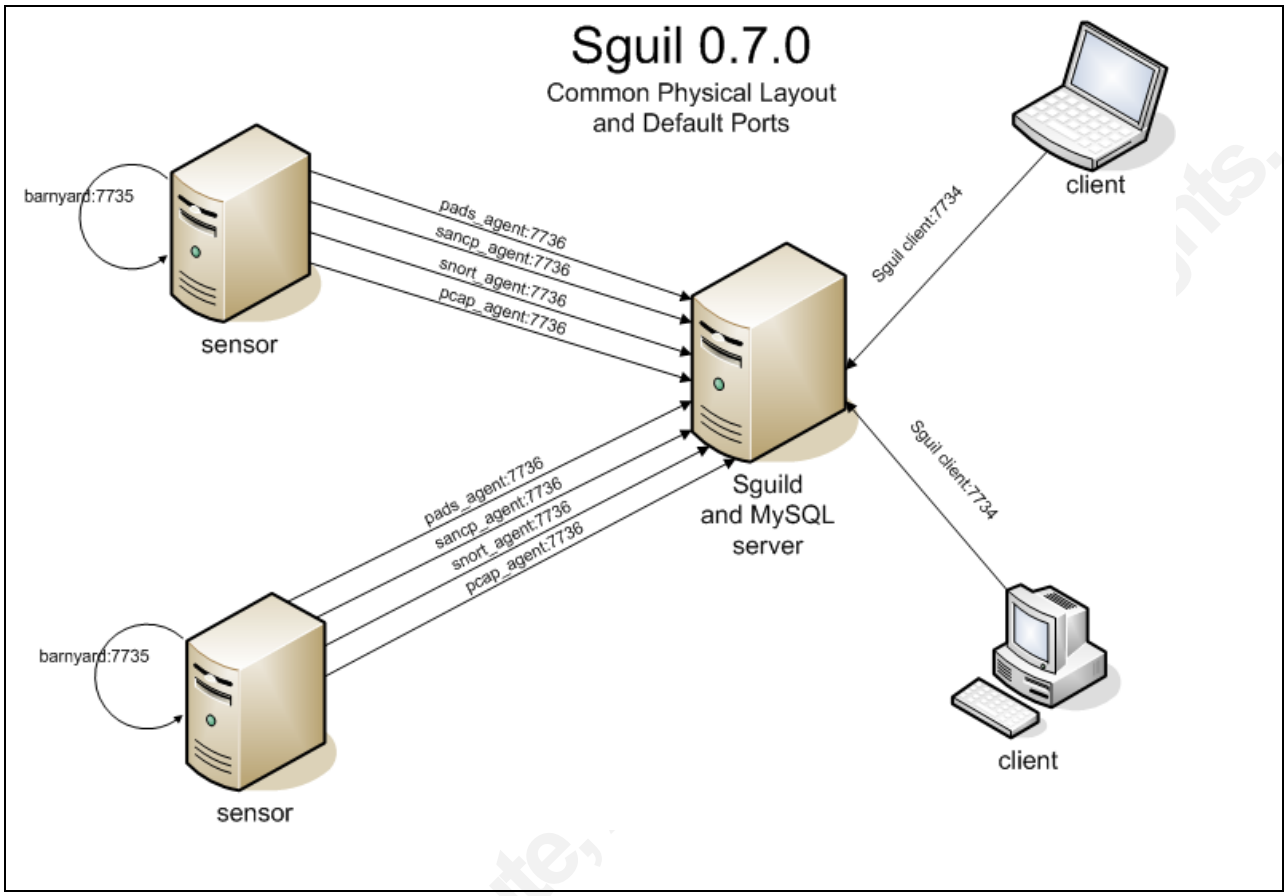
3.3. Distributed sensors for complex environments

Security Onion allows deployment in a master / slave distributed sensor architecture for large environments. Sensor deployment architecture depends on the network design, network throughput and additional custom requirements.

A Sguil system is composed of a single Sguil server and of an arbitrary number of Sguil network sensors. The sensors perform all the security monitoring tasks and feed information back to the server on a regular basis. The server coordinates this information, stores it in a database and communicates with Sguil clients running on administrator desktops. It can also issue requests for specific information from the sensors.

Each sensor monitors a single network link although multiple sensors can be present on one physical machine. They collect several different types of information:

1. Snort monitors the link for security events, and logs them to a file on the local disk.
2. Barnyard , an output spool reader for Snort, takes events from the Snort log file and sends them to the sensor agent, which inserts them into database running on the Sguil server in near real-time.
3. Daemonlogger™, a packet logger, sniffs the network packets and logs the full content of all network packets to the local disk.
4. SANCP, a tool designed to collect statistical information regarding network traffic, records TCP/IP sessions and forwards them to the database on the Sguil server.
5. The Sguil agent also listens for commands from the Sguil server. These commands are typically requests for packet data previously logged by Snort (nsmwiki.org, 2012).



(nsmwiki.org, 2012)

3.4. Configuring Security Onion for Monitoring

Security Onion contains several network security monitoring tools and applications integrated together which helps with the download, compile and installation of these applications. When not in use, tools like Bro, or Suricata can either be disabled or removed. The NSM infrastructure setup is done by clicking on setup icon and following the wizard as shown in appendix A. Additional customization is needed to suit this particular environment as described in this section.

3.4.1. Configuring OSSEC as Log Collector

OSSEC HIDS agent monitors host activities based on the rules defined for anomalous event such as rootkit detection, integrity checking etc. These agents can also forward the logged events or intrusion activities to the OSSEC management system. In this section, OSSEC is configured on the SO monitor as a log collector to receive the logs from other hosts. OSSEC `remote` configuration option makes the OSSEC agent run as a management system that listens for agent traffic on the specified port.

The default location for the OSSEC HIDS installation is in the `/var/ossec` directory. The `logall` global parameter and syslog configuration are defined in the configuration file `/var/ossec/etc/ossec.conf` to listen for incoming syslogs.

```
<global>
<logall>yes</logall>
</global>

<remote>
<connection>syslog</connection>
<allowed-ips>any</allowed-ips>
<protocol>udp</protocol>
<port>514</port>
</remote>
```

Then, syslog traffic needs to be allowed into the log system (e.g. `192.168.200.50`) by the `UFW` host firewall and OSSEC needs to be restarted:

```
$sudo ufw allow proto udp from 192.168.0.0/16 to \
192.168.200.50 port 514
$sudo /var/ossec/bin/ossec-control restart
```

OSSEC log collector can be configured to run in the secure mode to collect logs from the OSSEC agents if desired. OSSEC server `remote` listening options are configured in the `ossec.conf` to run in the secure mode like this:

```
<remote>
<connection>secure</connection>
<allowed-ips>any</allowed-ips>
<protocol>udp</protocol>
<port>1514</port>
</remote>
```

The raw logs are saved to files organized by date in `/var/ossec/logs/archives`. Logs and alerts log files are compressed, signed and rotated daily. The log message headers are in the format of `YYYY Month dd HH:MM:ss agent_name->/path/to/log/file.`

3.4.2. Configuring NIDS Sguil/Snort Sensor

Snort is configured to monitor network traffic in the NIDS mode using switch spanning port in this section. Custom Snort configuration is created for this interface in the file located under respective interface folder at `/etc/nsm/HOSTNAME-INTERFACE1/snort.conf`. Network variables, dynamic loaded libraries and preprocessors are configured to match the custom environment in the Snort configuration file.

Global Snort custom rules and rule classifications are added to `local.rules and classifications.config` respectively located at `/etc/nsm/rules/`. Specific sensor custom rules are added to respective sensor `/etc/nsm/HOSTNAME-INTERFACE1/rules/local.rules` rule configuration. Custom classifications are defined to the config file `/etc/nsm/HOSTNAME-INTERFACE1/classifications.config`. Sensor data is collected into the directory `/nsm/sensor_data/HOSTNAME-NIC1`.

Snort alerts are configured to output into the unified2 binary format. The barnyard2 is configured to parse that Snort output into the Sguil and Snorby database. It is important to have good understanding of underlying Snort IDS for proper tuning and configuration updates.

3.4.3. Configuring LIDS Sguil/Snort Sensor

In this section, a second sensor is configured to monitor log traffic. Snort follows the same directions as in the previous section for different interface used for the syslog collection. OSSEC is configured to collect and archive logs using this interface. OSSEC monitors logs

based on the OSSEC rules configured. OSSEC writes alerts to `/var/ossec/logs/alerts/alerts.log` on the OSSEC rule matching events. The OSSEC Agent for Sguil integrated on SO reads alerts from this alert log file and sends it to the Sguil database.

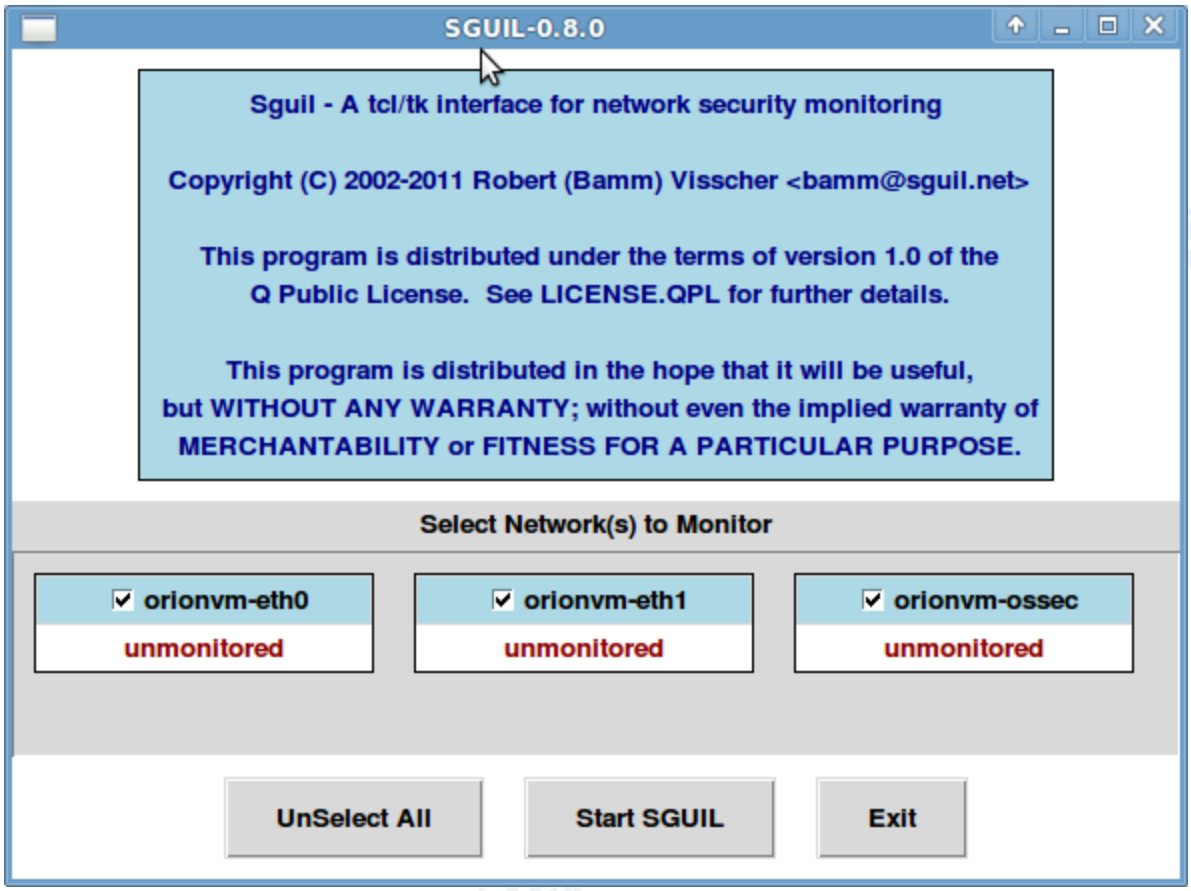
There are well defined OSSEC rules for cross-platform which makes it a good choice for generating alerts from the hosts. This sensor is configured with log based Snort custom rules to alert on the log events not covered by OSSEC. That includes logs such as custom application, appliances and logs received by another logger application such as rsyslog as explained in the earlier section.

3.4.4. Configuring Sguil Server

The Sguil database is created when NSM setup wizard is first run. Sguil configuration file `/etc/sguild/server.conf` allows customization of the Sguil database and other environment settings to suit the custom environment. The `DAYSTOKEEP` variable in the configuration file `/etc/nsm/securityonion.conf` allows setting a retention period for the alerts in the Sguil database. The NSM infrastructure service is started with the `nsm` script provided on Security Onion.

```
user@orionvm:~$ sudo service nsm start
Starting: securityonion
  * starting: sguil server                                                    [  OK  ]
Starting: orionvm-eth0
  * starting: pcap_agent (sguil)                                              [  OK  ]
  * starting: sancp_agent (sguil)                                             [  OK  ]
  * starting: snort_agent (sguil)                                             [  OK  ]
  * starting: snort (alert data)                                              [  OK  ]
  * starting: barnyard2 (spooler, unified2 format)                            [  OK  ]
  * starting: sancp (session data)                                            [  OK  ]
  * starting: pads (asset info)                                               [  OK  ]
  * starting: pads_agent (sguil)                                              [  OK  ]
  * starting: daemonlogger (full packet data)                                 [  OK  ]
  * starting: argus                                                           [  OK  ]
  * starting: httpry                                                          [  OK  ]
  * starting: httpry_agent (sguil)                                            [ OK ]
Starting: orionvm-eth1
  * starting: pcap_agent (sguil)                                              [  OK  ]
  * starting: sancp_agent (sguil)                                             [  OK  ]
  * starting: snort_agent (sguil)                                             [  OK  ]
  * starting: snort (alert data)                                              [  OK  ]
  * starting: barnyard2 (spooler, unified2 format)                            [  OK  ]
  * starting: sancp (session data)                                            [  OK  ]
  * starting: pads (asset info)                                               [  OK  ]
  * starting: pads_agent (sguil)                                              [  OK  ]
  * starting: daemonlogger (full packet data)                                 [  OK  ]
  * starting: argus                                                           [  OK  ]
  * starting: httpry                                                          [  OK  ]
  * starting: httpry_agent (sguil)                                            [ OK ]
Starting: HIDS
  * starting: ossec_agent (sguil)                                             [  OK  ]
```

After all services are started, the Sguil client can be launched. Sguil then allows selecting which networks to monitor (`eth0`, `eth1` and `ossec`). Clicking the `Select All` button shows alerts from all sensors in the Sguil client.



3.5. Rules

Snort and OSSEC have a large number of rule sets available to choose from. Large numbers of anomalies are detected right from the start using these rulesets. These rulesets needs to be tuned to reduce the number of false positives. This section provides two different ways to write the custom rules to distinguish a log based event. LIDS sensor uses OSSEC, Snort, or both kind of rules depending on the log type and other factors such as log encryption, rule complexity. Analysts can opt for the most effective and easier way to write a rule to alert for LIDS sensor based on their expertise. NIDS sensor works with Snort rules to alert on a network event of interest. Writing rules becomes most important and arguably most difficult part of the network security monitoring.

This section is focused more on writing rules to detect significant event using log traffic or the logs collected. Critical security log events cheatsheet shown in Appendix B provides important events that can be monitored using custom Snort or OSSEC rules.

3.5.1. Snort Rule

Snort rules are powerful, flexible and relatively easy to write. All Snort rules follow a very simple format and define what Snort should watch for as it inspects packet header, payload or both. Snort rules are divided into two logical sections, the rule header and the rule body. The optional rule body follows the rule header and is surrounded by parentheses. Snort rules based on content inspection look for raw text, hex data ("!9090!"), or a mix of both. That makes it easy to write a rule to look for the known patterns and detect a log based event.

Here is a rule writing example to alert for a Windows security log `event id 540` associated with anonymous logon.

```
alert udp any any -> $central-log-server 514 (msg:"Windows
Anonymous Network Logon"; content:"Security,540,"; nocase;
content: "anonymous"; nocase; reference:bugtraq,540;
reference:url,http://www.ultimatewindowssecurity.com/securitylog
/encyclopedia/event.aspx?eventid=540; classtype:attempted-user;
priority:2; sid:505401; rev:1;)
```

In the example above, the first part of the rule header before "`(`"describes the rule to alert an event for the UDP traffic flowing from any IP address, any source port to the central log server on port 514. The second part of the rule body looks in the payload for the content "`Security,540,`" unique to the Windows successful network logon and then content "`anonymous`" for anonymous user logon. It assigns an id "`505401`", revision level "`1`", priority "`2`" and a rule message "`Windows Anonymous Network Logon`" to identify the rule.

Similarly, the rule below is another example added to the `local.rules` file to alert on the "`Windows Account Created`" event. It matches on the content "`Security,624,`" for Windows security log `event id 624` for account creation.

```
alert udp any any -> $central-log-server 514 (msg:"Windows
Account Created"; content:"Security,624,"; nocase;
reference:url,http://www.ultimatewindowssecurity.com/securitylog
/encyclopedia/event.aspx?eventid=624;classtype:attempted-admin;
priority:3; sid:506241; rev:1;)
```

3.5.2. OSSEC Rule

OSSEC parses the log using built in or custom decoders and parses out fields defined in the decoder. It then checks for any configured rules to determine if an alert should be generated based on the parsed values. These values can also be passed to active-response commands if enabled. OSSEC decoders and rules both follow XML format rules writing. OSSEC decoders and rules can be written for well formatted new application log. There is a need to write a new decoder first for new log type before writing rules for it. OSSEC decoders and rules are located at `/var/ossec/etc/decoders.xml` and `/var/ossec/rules/*.xml` respectively. New customized rules are added to the rules file `/var/ossec/rules/local_rules.xml`.

Here is an example of the default TELNET decoder shipped with OSSEC in which the decoder extracts source ip address field.

```
<decoder name="telnetd">
  <program_name>^telnetd|^in.telnetd</program_name>
</decoder>

<decoder name="telnetd-ip">
  <parent>telnetd</parent>
  <regex>from (\d+.\d+.\d+.\d+)$</regex>
  <order>srcip</order>
</decoder>
```

Following are some of the rules based on above TELNET decoder named `telnetd`. First a catch all generic rule `5600` is created in the group name `syslog,telnetd` for the logs decoded by this log decoder. This rule is set as level `0` because an alert does not need to be triggered. Subsequently, a sub-rule with id tag `5601` is defined to alert on `refused connect from` pattern match for the event `Connection refused by TCP Wrappers` in the log. Then, there are few more rules looking for an event condition to trigger an alert.

```
<group name="syslog,telnetd">
  <rule id="5600" level="0" noalert="1">
    <match>telnetd</match>
    <description>Grouping for the telnetd rules</description>
  </rule>

  <rule id="5601" level="5">
    <if_sid>5600</if_sid>
    <match>refused connect from </match>
    <description>Connection refused by TCP Wrappers.
</description>
  </rule>

  <rule id="5602" level="3">
    <if_sid>5600</if_sid>
    <match>: connect from </match>
    <description>Remote host established a telnet connection.
</description>
  </rule>

<rule id="5631" level="10" frequency="6" timeframe="120">
    <if_matched_sid>5602</if_matched_sid>
    <same_source_ip />
    <description>Multiple connection attempts from same source
</description>
    <description>(possible scan).</description>
  </rule>
</group>
```

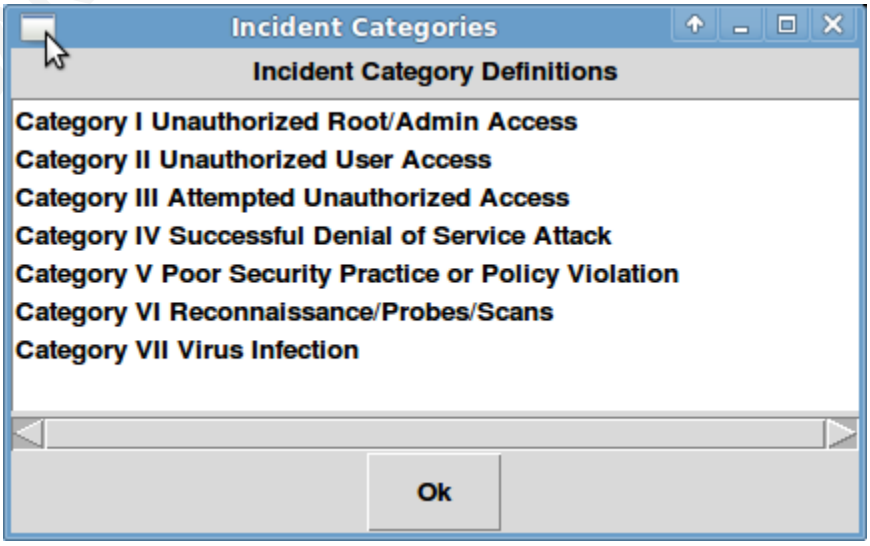
4. Log Analysis & Correlation

Log analysis is an art and is geared towards narrowing down to the events of interest. Analyst needs to focus on recent changes, failures, errors, status changes, access and administration events, and other events unusual for your environment. Hence, it is important to minimize noise by removing routine, repetitive log entries from the view after confirming that they are benign. Analyst needs to correlate activities across different logs to get a comprehensive picture of the situation (Chuvakin & Zeltser, 2012).

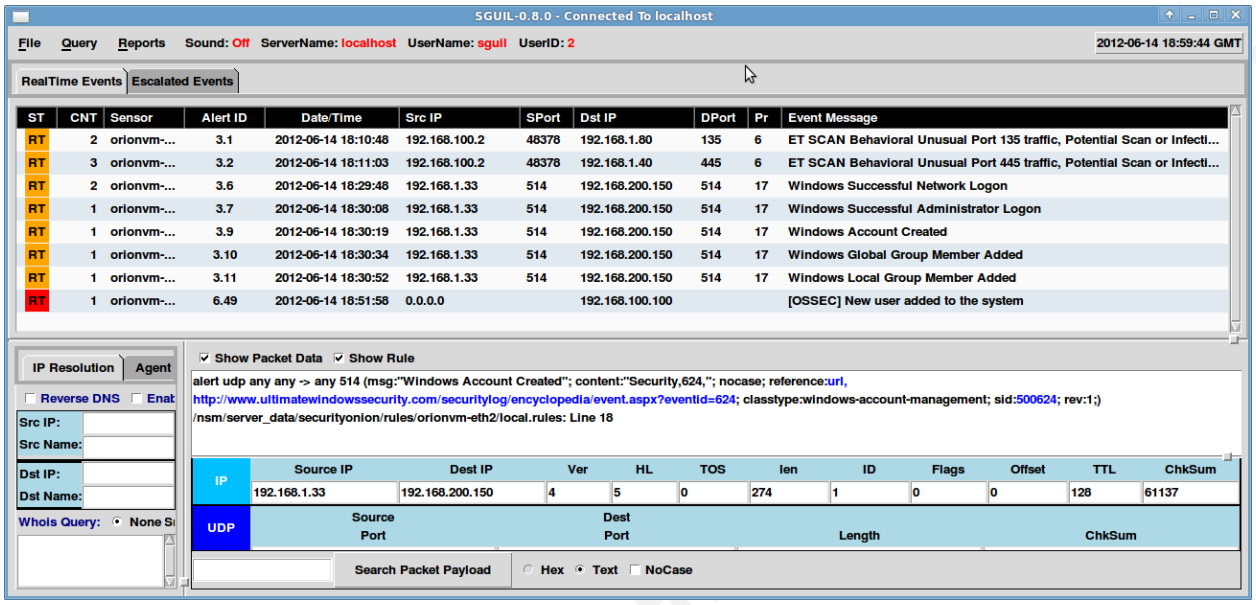
4.1. Event Analysis

Analysis typically begins with Snort or OSSEC alerts displayed on the Sguil console in near real time. Analysts can then categorize the alert based on type of activity or escalate the alert to a more senior analyst for further analysis. Analyst highlight the alert and press the appropriate function key associated with the event classification or right click on the alert and select the appropriate event status. Optional comments (about the action and why the action is taken) to the alerts can be added.

Once alerts are categorized, they disappear from the console (Bianco, 2012). Alerts are still available for reporting or further analysis at a later date from the database. Sguil provides full logging and audit trail of alert activity i.e. who took the action, when action was taken etc. This figure shows the default categories available to classify an alert.



We see the Sguil console in the next screen with some alerts. These alerts are combination of Snort NIDS alerts, custom Snort Windows log alerts and OSSEC alerts.



In the first two alerts, a reconnaissance scan is performed for services on RPC port 135 and SMB port 445 to network `192.168.1.0`. Following the scan, there are Windows Logon events to address `192.168.1.33` from the intruder or compromised machine with address `192.168.100.2`. Next, an account was created on the Windows system and that account was added to a global group to escalate privilege for that account. There is also an OSSEC `user addition` alert from system `192.168.100.100`.

Highlighting the alert shows the alert data and the rule that triggered this event if the respective checkboxes are selected. Wireshark can be used for the further pcap analysis. Event transcripts into the Wireshark are generated by selecting an alert and right click on the `sid.cid` column.

The original Windows administrator log on and account creation events can also be pulled from syslog archive from the location `/var/ossec/logs/archives/` as in the example below:

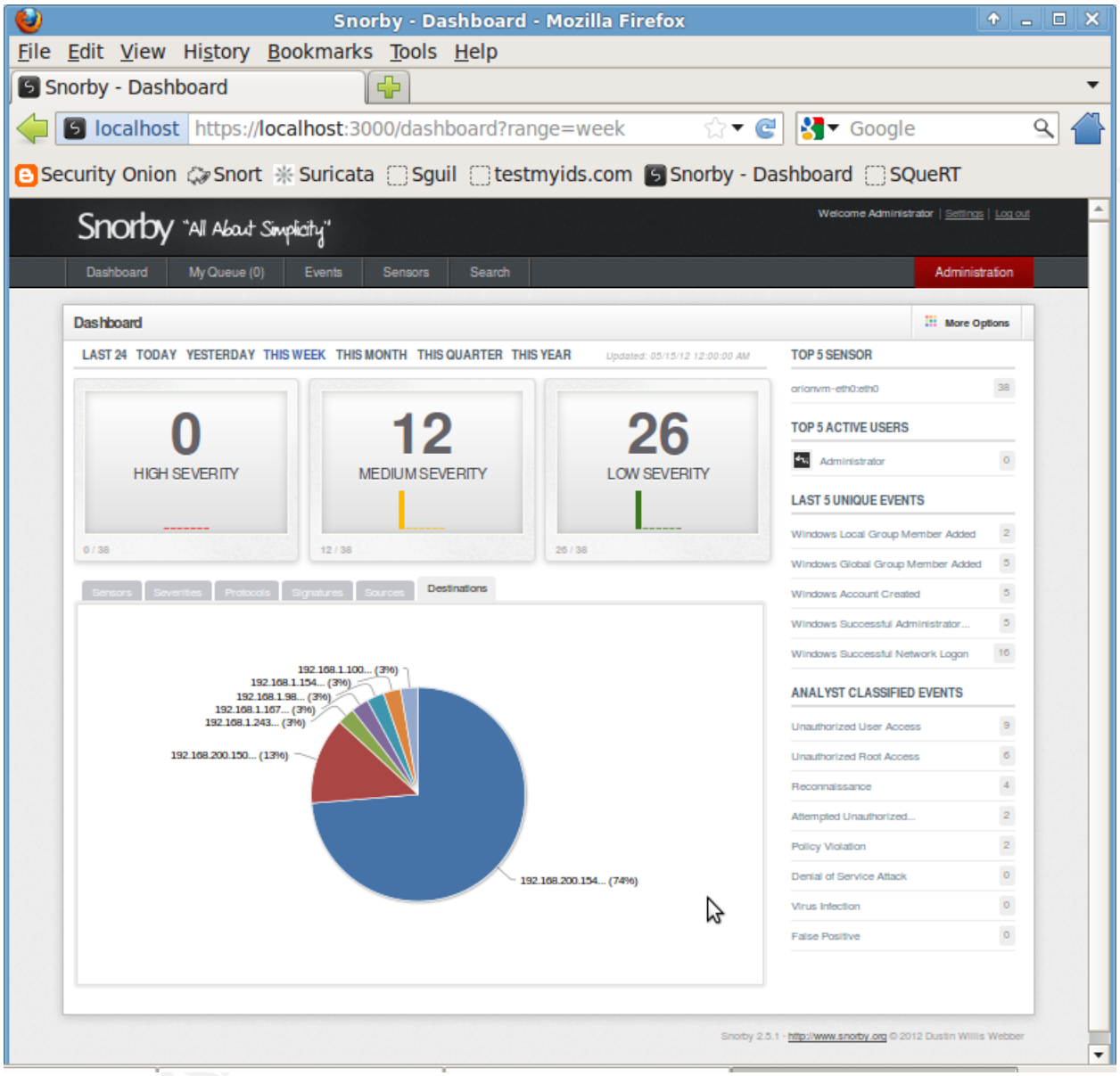
```
2012 Jun 14 18:30:08 LABSRV01->192.168.1.33 Jun 14 14:30:19
LABSRV01 EvntSLog: Security,540,Fri Jun 14 18:30:19 2012,
TESTLAB\\Administrator,3,Administrator;
TESTLAB;(0x0,0x822BE);3;Kerberos;Kerberos;;{ceab435b-da16-6668-
c8ad-f860d17b1253};-;-;-;-;-;192.168.100.2;0;

2012 Jun 14 18:30:19 LABSRV01->192.168.1.33 Jun 14 14:30:30
LABSRV01 EvntSLog: Security,624,Fri Jun 14 18:30:30
2012,TESTLAB\\Administrator,7,ServiceAccount;LABSRV01;
LABSRV01\\ServiceAccount;Administrator;TESTLAB;(0x0,0x50D38);-
;ServiceAccount;%1793;-;%1793;%1793;%1793;%1793;%1793;%1794
```

The OSSEC alert `new user addition` is generated by OSSEC rule. Alert event detail is seen by highlighting the alert and can also be pulled from OSSEC alert location `/var/ossec/logs/alerts/` as in the example below:

```
** Alert 1339699918.65814: mail  - syslog,adduser
2012 Jun 14 18:51:58 (CentOSHost) 192.168.100.100-
>/var/log/secure
Rule: 5902 (level 8) -> 'New user added to the system'
Src IP: (none)
User: (none)
Jun 14 18:41:23 localhost useradd[19265]: new user:
name=ftpuser, UID=510, GID=501, home=/var/ftp, shell=/bin/bash
```

This OSSEC alert event details looks suspicious for the system based on the user details, time and the network it is on. All logs should be pulled from the archive to validate the system compromise.

An analyst can also use the Snorby front-end for the same monitoring and analysis. Snorby has similar alert classifications but is independent of Sguil alert classification. Snorby provides easy to use console which is very configurable and works great for creating queries and detailed analysis.



4.2. Database Query

Sguil database alerts searches are initiated using different templates. For a blank template, the Query->Event Query may be used. Once selected, a query builder pops up to edit the query. Only the `WHERE` statement can be edited. Other templates are available by selecting an event and right clicking on the src/dst ip columns. A blank template for session queries can be found under Query->Session Query and a completely blank query builder can be found under Query->Query Builder.

Sunil Gupta, sgupta911@gmail.com

User query can be saved for future run using Standard Query. In the Standard Query dialog, select File->Add Query to add a new user query. User queries are saved locally and therefore are only available on the machine used to create them. To create a standard query for global use, edit the `sguild.queries` file on your `sguild` server. In order to have `sguild` reload the `sguild.queries` file and update the clients, a `HUP` signal is sent to the `sguild` process (nsmwiki.org, 2012).

4.3. Event Correlation

It becomes easier to correlate events by having multiple sensors feeding different types of events into the same analysis console. Correlating activities across different logs provides a comprehensive picture of the chain of events. Analysts need to develop theories about what occurred and explore logs to confirm or disprove those theories. It is important for the analyst to rely on the time stamps contained in logs, especially when time zone differences are considered. Event correlation becomes more difficult if the devices reporting events have inconsistent clock settings.

The chain of events in the followed example show that an initial service scan, followed by an administrator account log on, account creation and account membership change events were all part of the same organized incident. All these events fall in the time sequence and addresses used.

Events that have the same source IP address and signature/message are correlated under the same event in the appropriate real time pane. Each time an event is correlated the CNT, count field is incremented by one. To view all the correlated events, select the event->right click on the CNT column->View Correlated Events. The shortcut for this info is to middle click on the CNT column of a selected event (nsmwiki.org, 2012).

4.4. Auto Categorization

Sguil can automatically categorize events by editing the `autocat.conf` file at `/etc/nsm/securityonion/` on the Sguil server. These event will have a status automatically assigned to them and will not appear in any analyst's console. Analysts can query for auto categorized events by selecting the `Auto Cats` global standard query (nsmwiki.org, 2012). The format to auto categorize an event in the `autocat.conf` is this:

```
#   <erase time>||<sensorName>||<src_ip>||<src_port>||<dst_ip>|
|<dst_port>||<proto>||<sig msg>||<cat value>
```

5. Log Alerting & Reporting

The sensor alerts on Security Onion are sent to both the Snorby and Sguil MySQL databases on the master server. Therefore, there are two different ways to perform analysis and reporting based on the database source. Alert notifications can be produced in different ways as well. Analyst can decide what works best for their custom environment to suit their alerting requirement.

5.1. Alert Classification and Prioritization

Real-time alerting with Snort is highly customizable. Alerts that need to result in real time notification can be chosen by assigning a priority to each rule, and by rule classifications. Each rule can have an individual priority attached to it, and every rule can be included in a classification of rules that has a priority attached to it. Rules can be prioritized as such that one priority of rule can be sent to one person while a different priority is sent to another. These different rules alerts can also be notified in different manners. One priority of rules can be sent to an email address that notifies via pager while another can simply send an email (Koziol, 2003).

Using the classification and priority option method helps classify rules that are important to a specific environment. The priority levels for rule categories are edited in the `classification.config` file located at `/etc/nsm/HOSTNAME-INTERFACE/`.

The example here shows how a new classification looks like:

```
config classification: successful-windows-default-admin,
Successful Administrator Privilege Gain,1
```

This defined classification is associated with the new `classtype` keyword `successful-windows-default-admin` for that alert rule in the `local.rules` file located at `/etc/nsm/rules` to alert in real time as shown here:

```
alert udp any any -> $central-log-server 514 (msg:"Successful
Administrator Privilege Gain"; content:"Security,528,";nocase;
content:"Administrator "; classtype: successful-windows-default-
admin;)
```

Assigning a priority option to the rule changes the priority for the specified rule. Individual priority assigned to a rule takes precedence over a classtype option with a different priority assigned on the same rule. For example, different priority level of `2` is assigned to the previous *Successful Administrator Privilege Gain* rule with the use of priority keyword like this:

```
alert udp any any -> $central-log-server 514 (msg:"Successful
Administrator Privilege Gain"; content:"Security,528,";nocase;
content:"Administrator "; classtype: successful-windows-default-
admin; priority:2;)
```

5.2. Email Alert with Sguil

Sguil's email alerting configuration is in the file `sguild.email` located at `/etc/nsm/securityonion/` and it contains email related information such as smtp server, from to email ids etc. Alerts can be notified based on the alert classes, alert SIDs and priorities in a space delimited manner configured in the above file. Any particular alert SID(s) can also be disabled to stop sending email about that alert.

Restart the Sguil daemon on the master server to take it into effect.

```
   $sudo nsm_server_ps-restart
```

5.3. Email Alert with OSSEC

The email address and host related information is configured inside the `<global>` section of the OSSEC configuration file at `/var/ossec/etc/ossec.conf`.

```
<global>
    <email_notification>yes</email_notification>
    <email_to>admin@myorg.com</email_to>
    <smtp_server>smtp.myorg.com</smtp_server>
    <email_from>ossec@myorg.com</email_from>
 </global>
```

The `email_alert_level` option set inside the `<alerts>` section of the `ossec.conf` file specifies the minimum alert level to send email notifications. Then restart the OSSEC:

```
$sudo service ossec restart
```

5.4. Email Alert with Snorby

In order to configure the Snorby email alert, Snorby configuration file at `/usr/local/share/snorby/config/snorby_config.yml` should have the domain setting in the production section to be the fully qualified domain name of the Snorby server. Next, email configuration file `/usr/local/share/snorby/config/initializers/mail_config.rb` needs to reflect the settings appropriate for the mail server. Then restart the Snorby `delayed_job` process:

```
$sudo pkill -f delayed_job

$sudo su www-data -c "cd /usr/local/share/snorby; bundle \
exec rake snorby:update RAILS_ENV=production"
```

5.5. Sguil Reporting

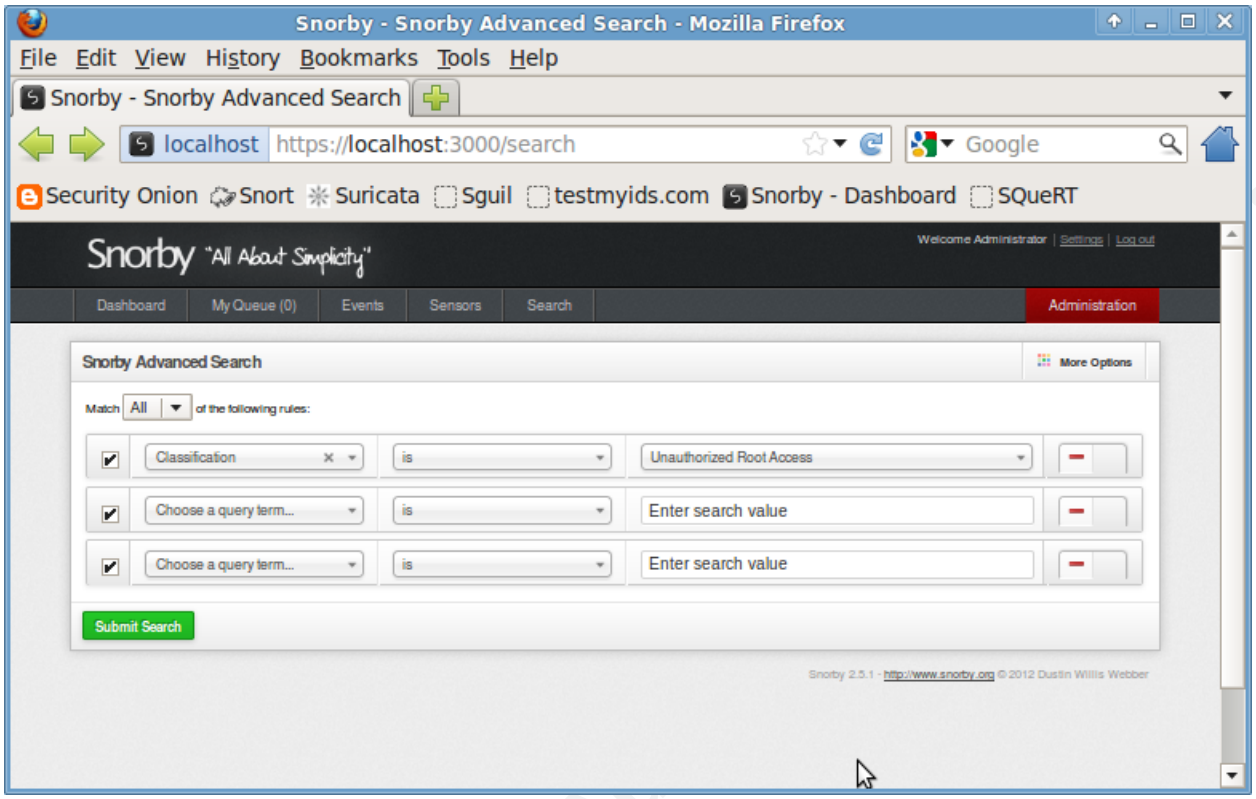
Sguil offers few basic reporting but lacks the mechanism to schedule reports, and reports with charts and graphs. Plain text or email reports are created by selecting the events to report and choosing appropriate report type from the report menu. Summary reports contain the full packet headers while detail reports add the payloads as well. Reports can be sanitized with the IP addresses obfuscated in the packet header. Query output can also be exported to a text file in different formats from the query pane. This will export only the information that is contained in the query pane (i.e. no header details, no payloads, etc) (nsmwiki.org, 2012).

Squert, LAMP based application, provides better interactive reporting based on Sguil alerts data although it is not intended to be a real time console replacement for Sguil. Squert web based application is accessed with the same user id as Sguil as it works with the same database.

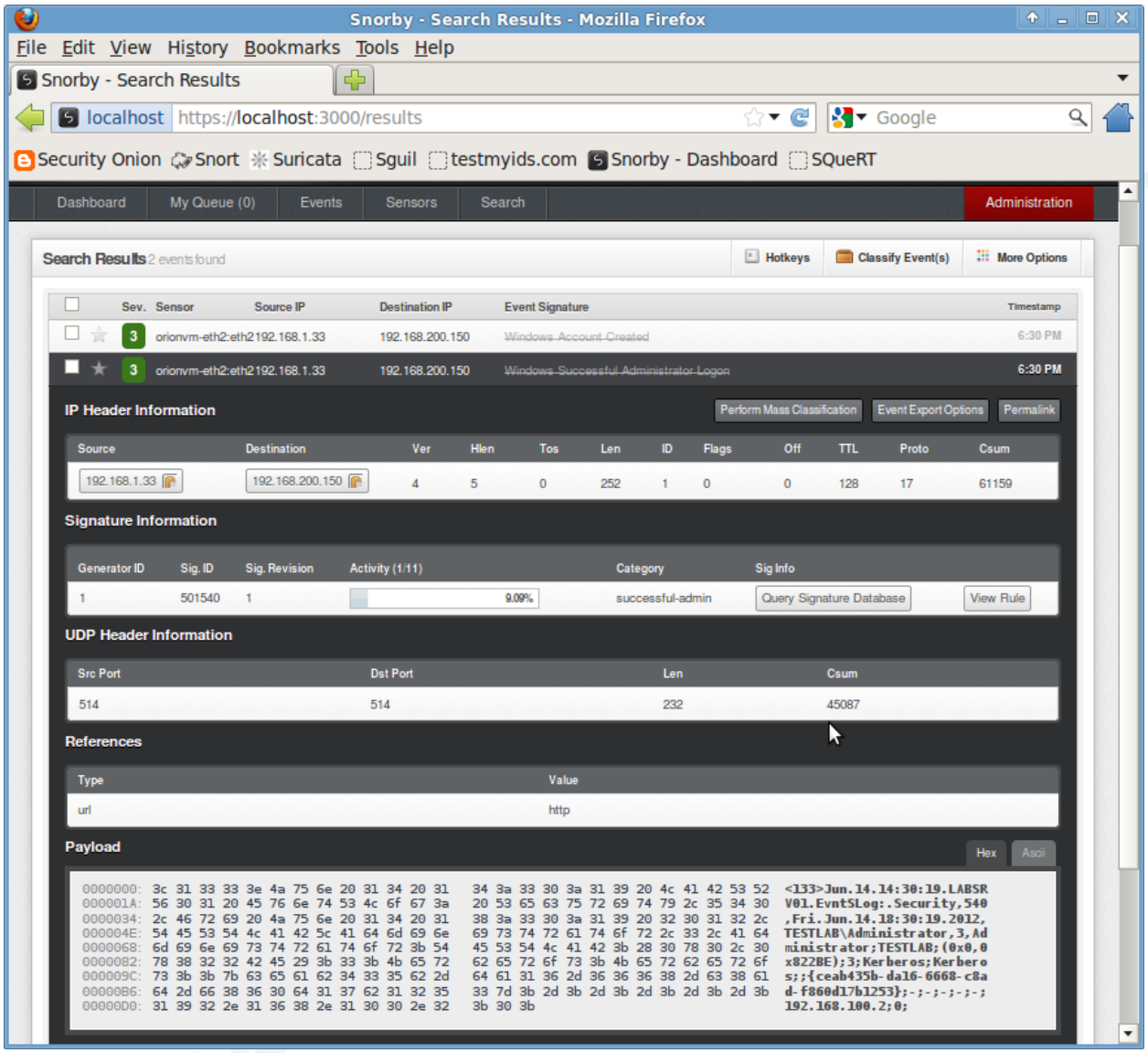
5.6. Snorby Reporting

Snorby brings network security monitoring data to life with a suite of beautiful, relevant and actionable metrics. Snorby is also very configurable. It can add custom severities or classifications, manage email notifications, and even extend functionality with third party products from an intuitive administration menu. It allows sharing data reports like sensor activity comparisons or most active signatures with daily, weekly, monthly, and ad-hoc PDF reports (Snorby.org, 2012).

In the next screen, a query is created on different fields. This query can also be saved for future reports. Selecting individual event in the query output displays the details about the event as shown in the subsequent screen.



Sunil Gupta, sgupta911@gmail.com



Sunil Gupta, sgupta911@gmail.com

6. Conclusion

This paper shows the importance of log managements and network monitoring for the effective security monitoring and compliance of an organization. It provides an open-source solution to a complex and very common challenge of log management and network monitoring. The solution is based on a framework provided by the Security Onion Linux Distribution, which makes it possible to integrate necessary applications on one platform. It tries to provide a cost effective logging, alerting and monitoring solution alternative to the organizations that cannot afford commercially available SIEM (Security Information and Event Management) solutions.

This paper highlights the necessary components in the logging process and how each plays a key role to stay on top of security monitoring. We show, how not only network traffic but log traffic can also be monitored to detect, log and report different activities using same techniques. Our approach works on the log or network traffic using known Snort and OSSEC XML based rules.

7. References

Bianco, David J. (2012). Open Source Network Security Monitoring With Sguil. Retrieved from http://www.vorant.com/files/nsm_with_sguil.pdf

Burks, Doug (2012). Security Onion. Retrieved from http://securityonion.blogspot.com/

Chuvakin, A & Zeltser, L. (2012). Critical Log Review Checklist for Security Incidents. Retrieved from http://zeltser.com/log-management/security-incident-log-review-checklist.html

Cid, Daniel B. (2007). Log Analysis using OSSEC. Retrieved from http://www.ossec.net/ossec-docs/auscert-2007-dcid.pdf

Holste, M. (2012). Enterprise-log-search-and-archive. Retrieved from http://code.google.com/p/enterprise-log-search-and-archive/

Kent, K. & Souppaya, M. (2006). Guide to Computer Security Log Management. National Institute of Standards and Technology (NIST) Publication 800-92. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

Sunil Gupta, sgupta911@gmail.com

Koziol, J. (2003). Intrusion Detection with Snort. (Indianapolis, Ind, Sams, ISBN 978-1578702817)

Nicolett, M. & Kavanagh, K. (2012). Magic Quadrant for Security Information and Event Management. Retrieved from http://www.gartner.com/technology/reprints.do?id=1-1ANUJF3&ct=120525&st=sb

NSM wiki (2012). Sguil Overview and Architecture. Retrieved from http://nsmwiki.org/Sguil, http://nsmwiki.org/Sguil_FAQ

OSSEC (2012). www.ossec.net. Retrieved from http://www.ossec.net

Scarfone, K. & Mell, P. (2007). Guide to Intrusion Detection and Prevention Systems (IDPS). National Institute of Standards and Technology (NIST) Publication 800-94. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

Shank, J. (2010). SANS Sixth Annual Log Management Survey. Retrieved from http://www.sans.org/reading_room/analysts_program/logmgtsurvey-2010.pdf

Snorby (2012). Snorby, All About Simplicity. Retrieved from http://snorby.org/.

Snort (2012). Snort :: Home Page. Retrieved from http://www.snort.org/

8. Acknowledgements

Many thanks to Doug Burks for creating the Security Onion and for the review of this paper.

Many thanks to Dr. Kees Leune for his time and mentoring on this paper writing.

General credit and thanks to SANS training courses and web site for the great source of information.

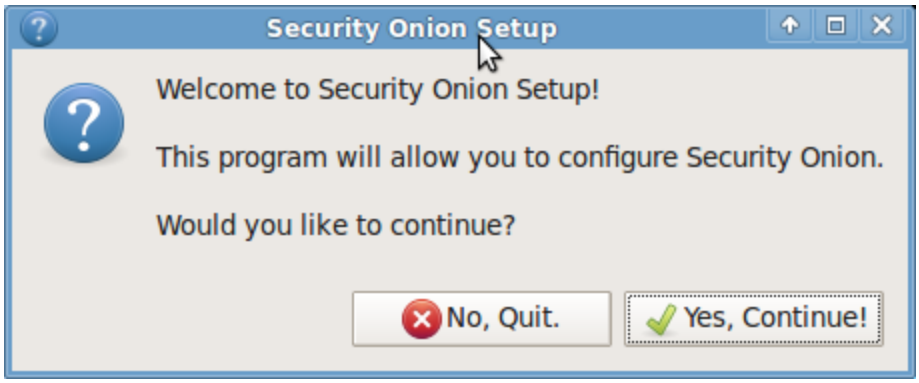
Sunil Gupta, sgupta911@gmail.com

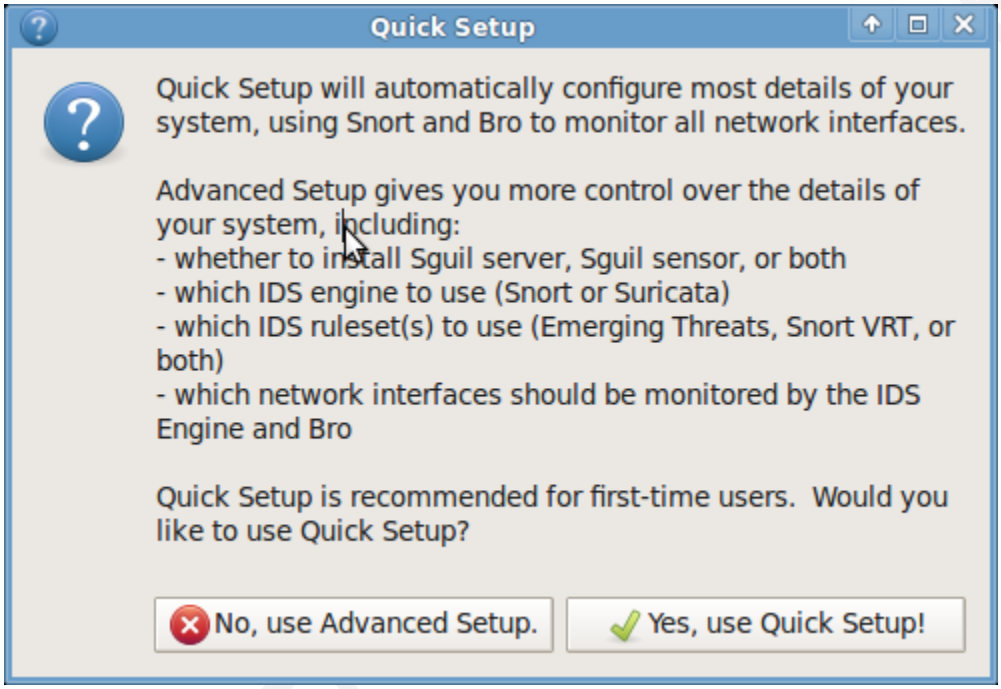
Appendix A - Security Onion Setup

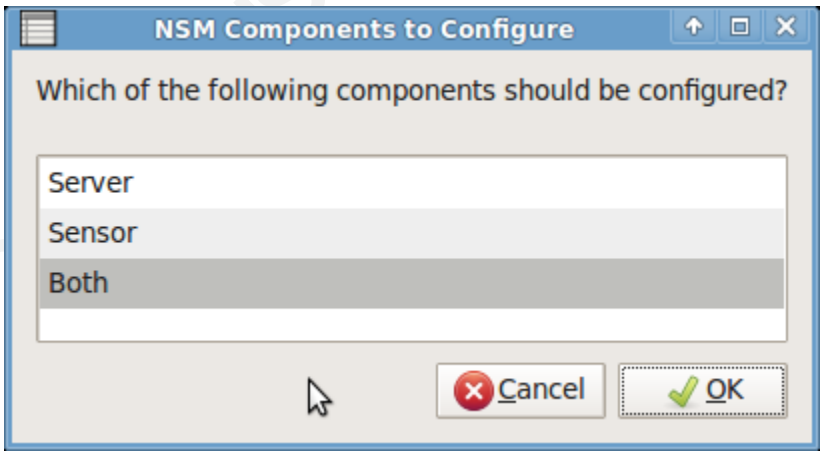
Security Onion download and installation instructions can be found at
http://code.google.com/p/security-onion/wiki/Installation

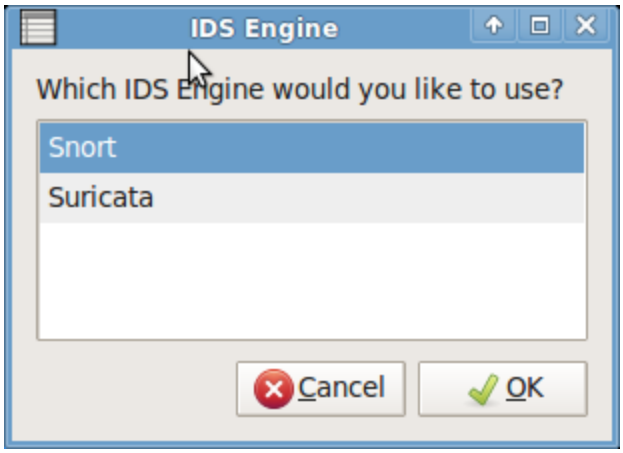
Here is how it looks when the Security Onion is installed to a system. It provides a setup shortcut on the screen to configure the NSM infrastructure. Setup can be followed from there as in the screens below.

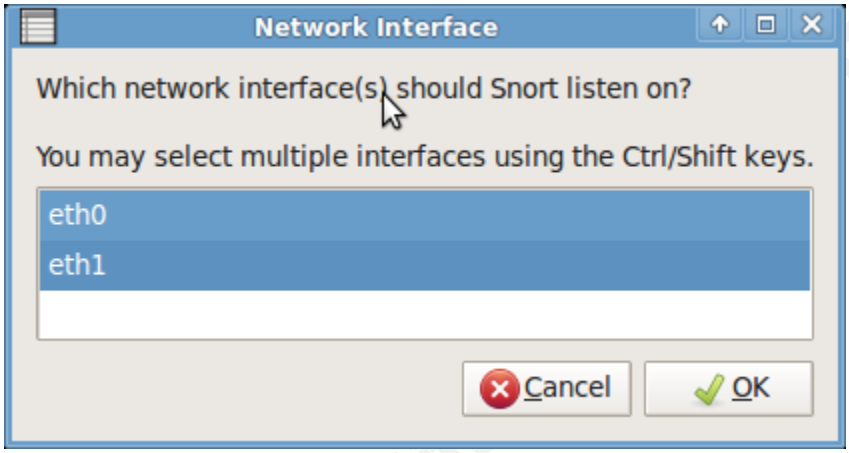
Sunil Gupta, sgupta911@gmail.com

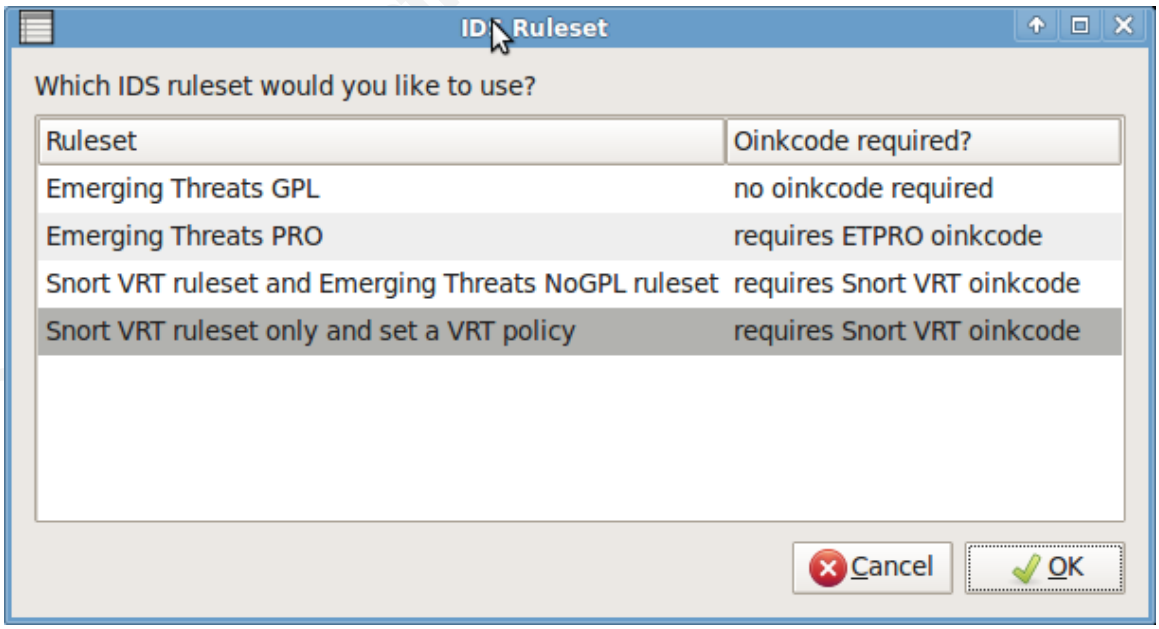


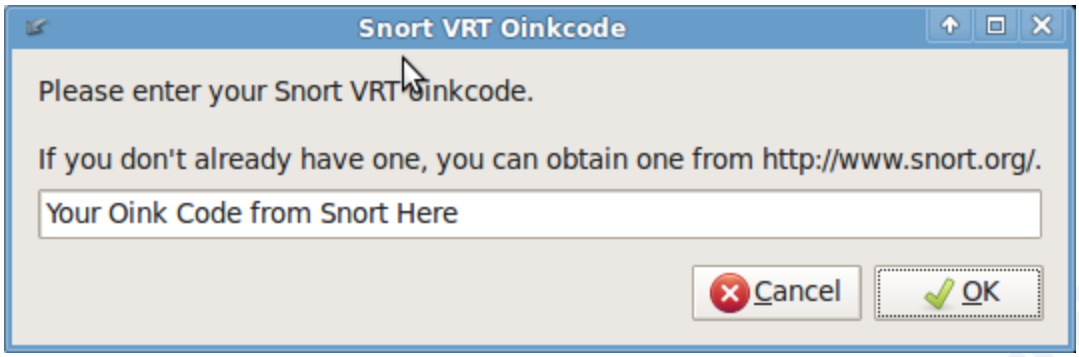


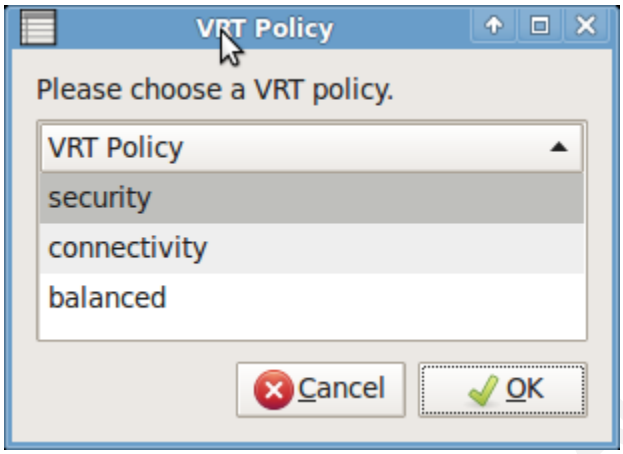


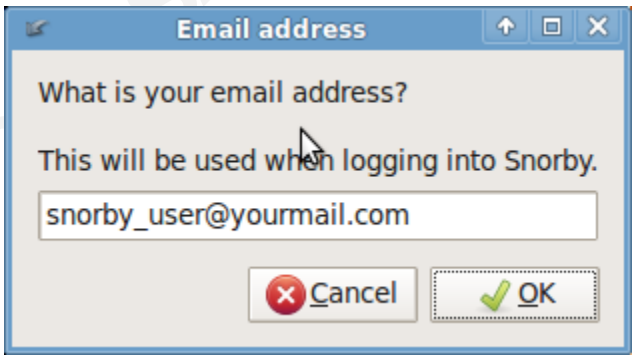


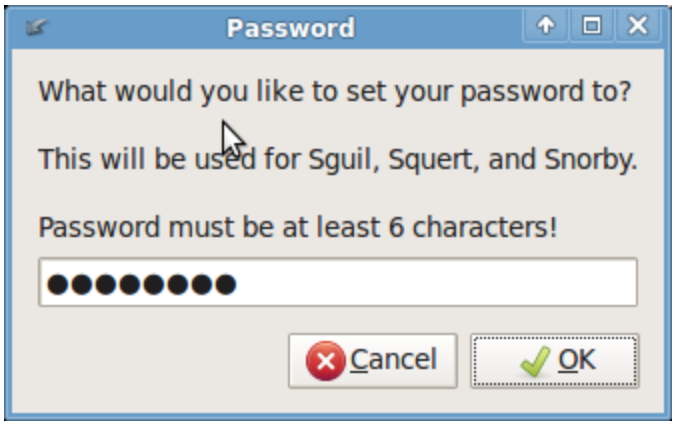


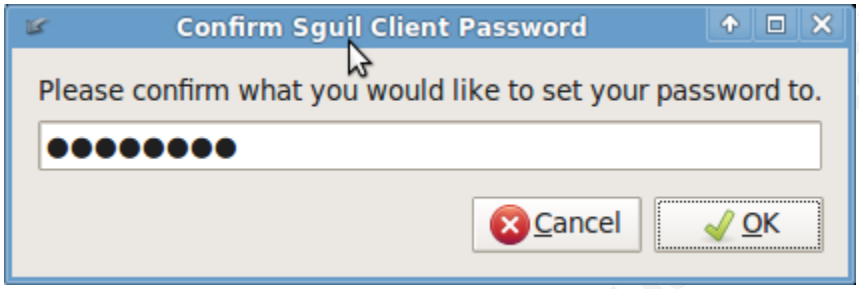


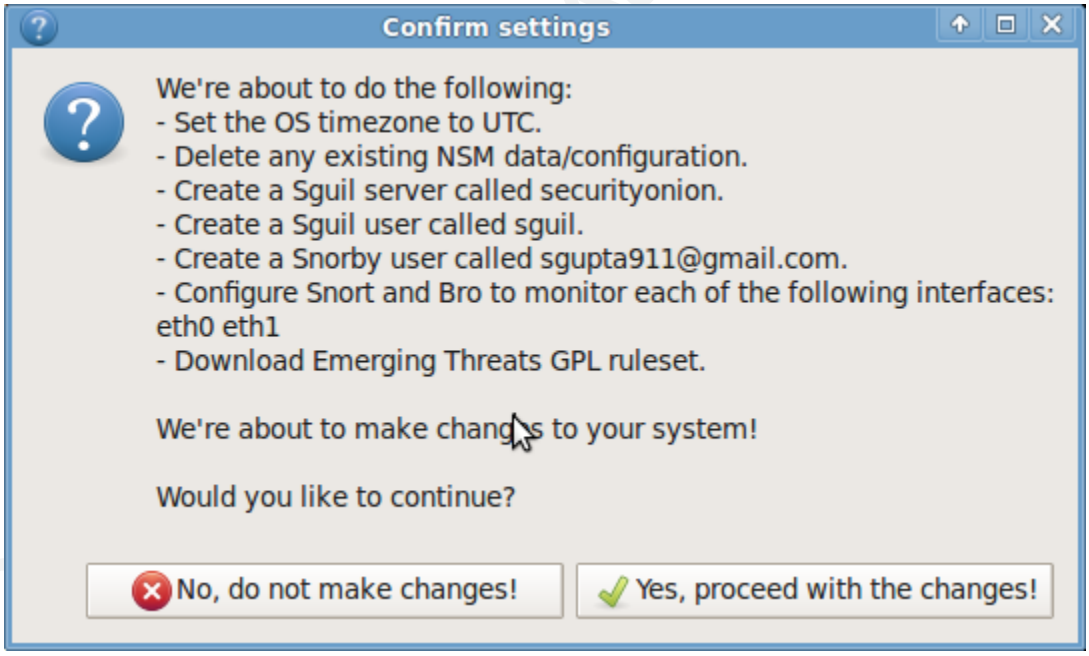


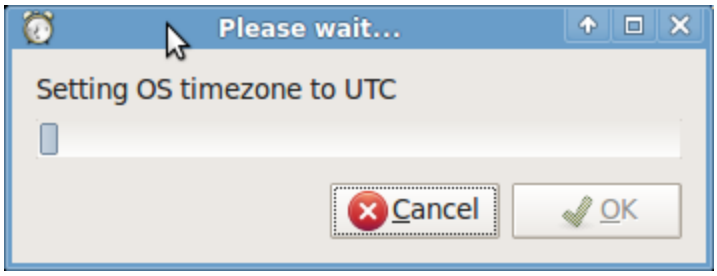


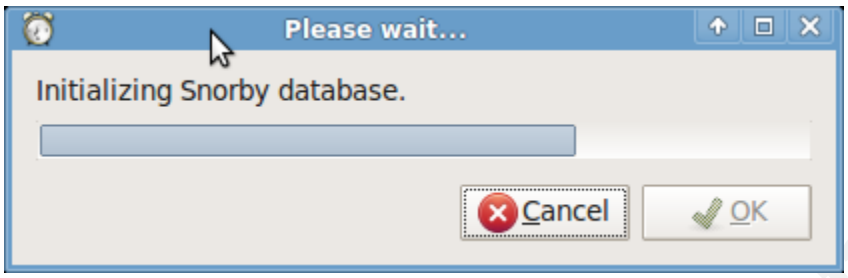


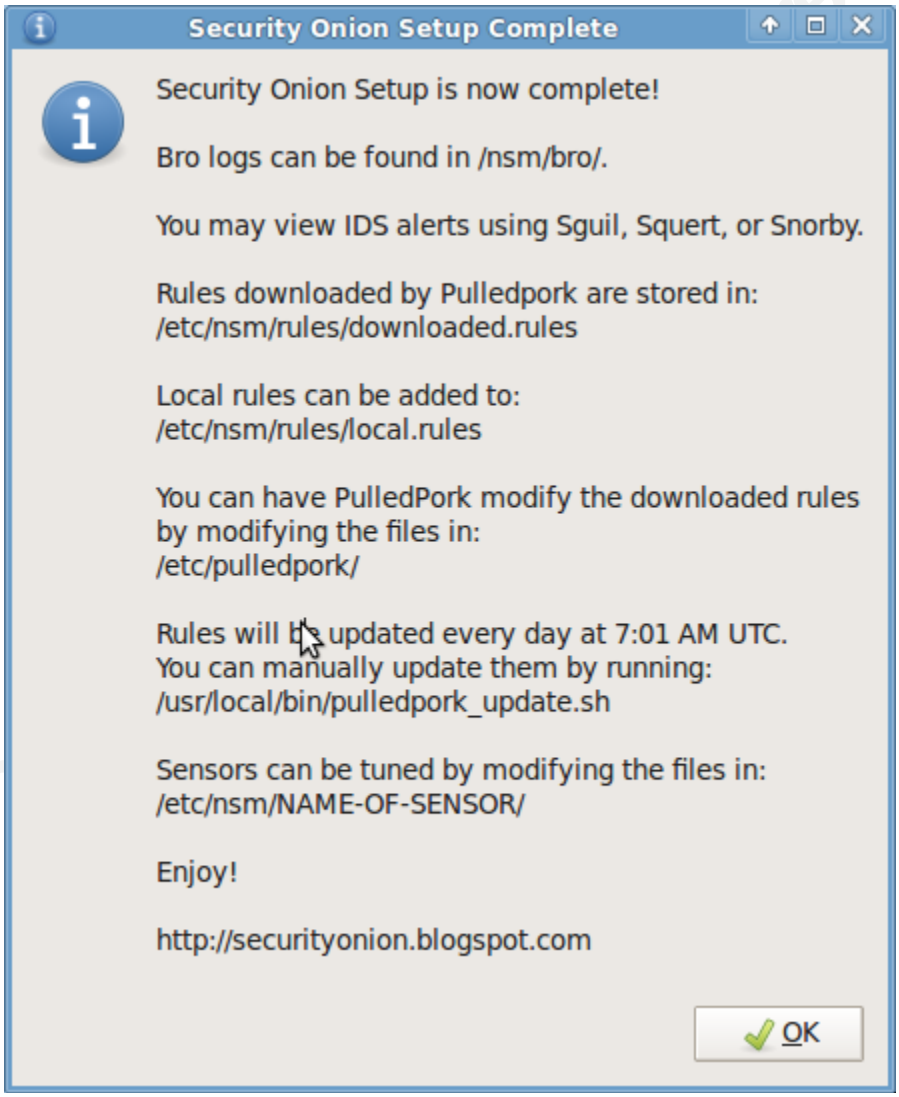






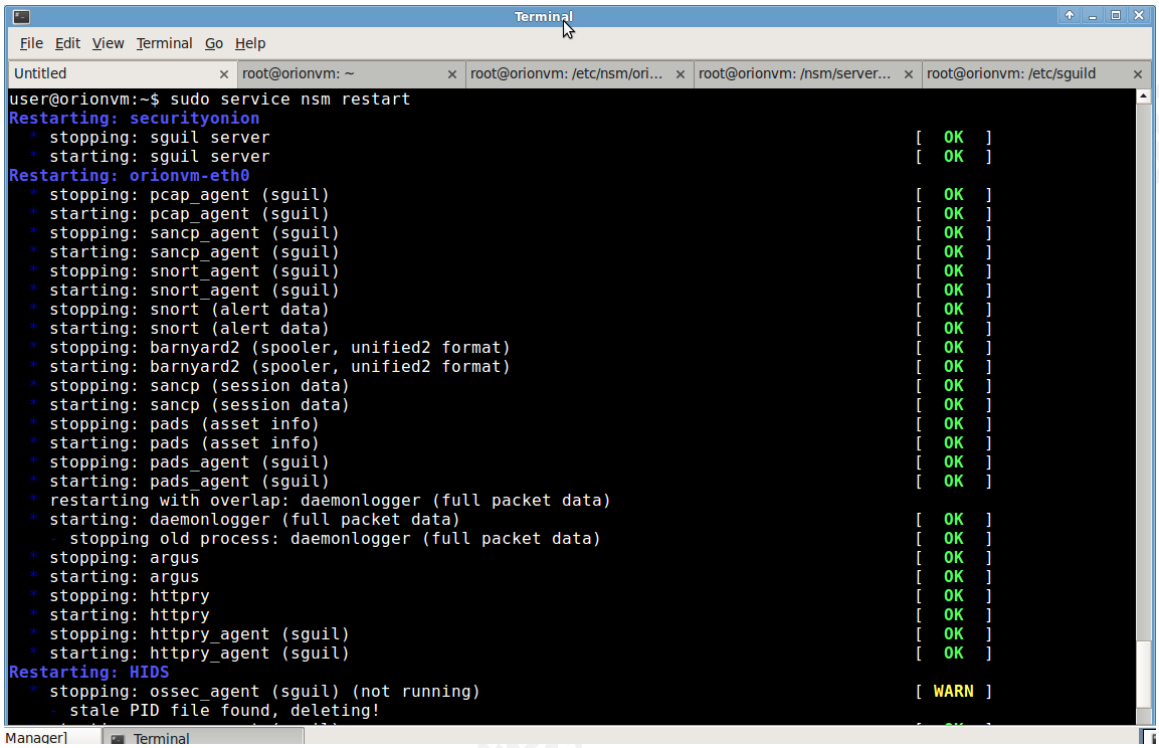






Security Onion provides a wrapper script to restart the entire NSM structure.

```
$ sudo service nsm start|stop|restart
```



Security Onion provides various configuration scripts such as sensor addition, user addition etc. at location `/usr/local/sbin` for the NSM management.

Sunil Gupta, sgupta911@gmail.com

Appendix B - Events to Monitor

This cheat sheet presents a checklist for reviewing critical logs when responding to a security incident. It can also be used for routine log review (Chuvakin & Zeltser, 2012).

What to Look for on Linux

Successful user login	"Accepted password", "Accepted publickey", "session opened"
Failed user login	"authentication failure", "failed password"
User log-off	"session closed"
User account change or deletion	"password changed", "new user", "delete user"
Sudo actions	"sudo: … COMMAND=…" "FAILED su"
Service failure	"failed" or "failure"

What to Look for on Windows

Event IDs are listed below for Windows 2000/XP. For Vista/7 security event ID, add 4096 to the event ID. Most of the events below are in the Security log; many are only logged on the domain controller.

User logon/logoff events	Successful logon 528, 540; failed logon 529-537, 539; logoff 538, 551, etc
User account changes	Created 624; enabled 626; changed 642; disabled 629; deleted 630
Password changes	To self: 628; to others: 627
Service started or stopped	7035, 7036, etc.
Object access denied (if auditing enabled)	560, 567, etc

Sunil Gupta, sgupta911@gmail.com

What to Look for on Network Devices

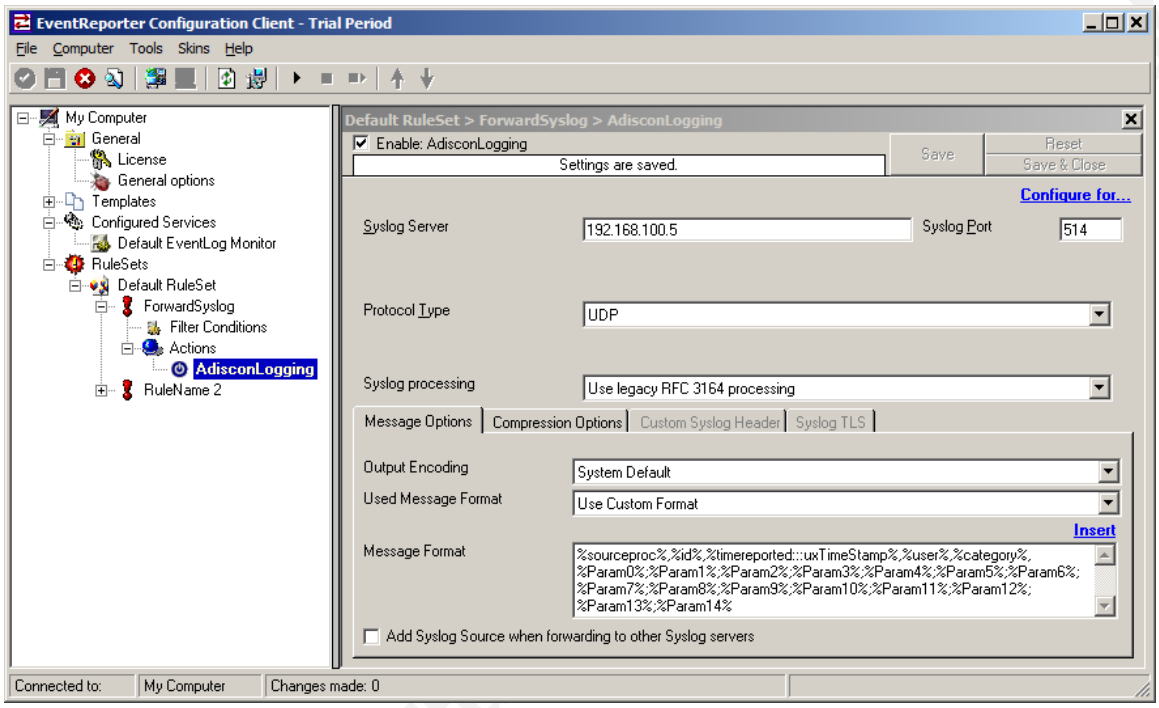
Look at both inbound and outbound activities. Examples below show log excerpts from Cisco ASA logs; other devices have similar functionality.

Traffic allowed on firewall	"Built … connection", "access-list … permitted"
Traffic blocked on firewall	"access-list … denied", "deny inbound", "Deny … by"
Bytes transferred (large files?)	"Teardown TCP connection … duration … bytes …"
Bandwidth and protocol usage	"limit … exceeded", "CPU utilization"
Detected attack activity	"attack from"
User account changes	"user added", "user deleted", "User priv level changed"
Administrator access	"AAA user …", "User … locked out", "login failed"

Appendix C – Client log forwarding

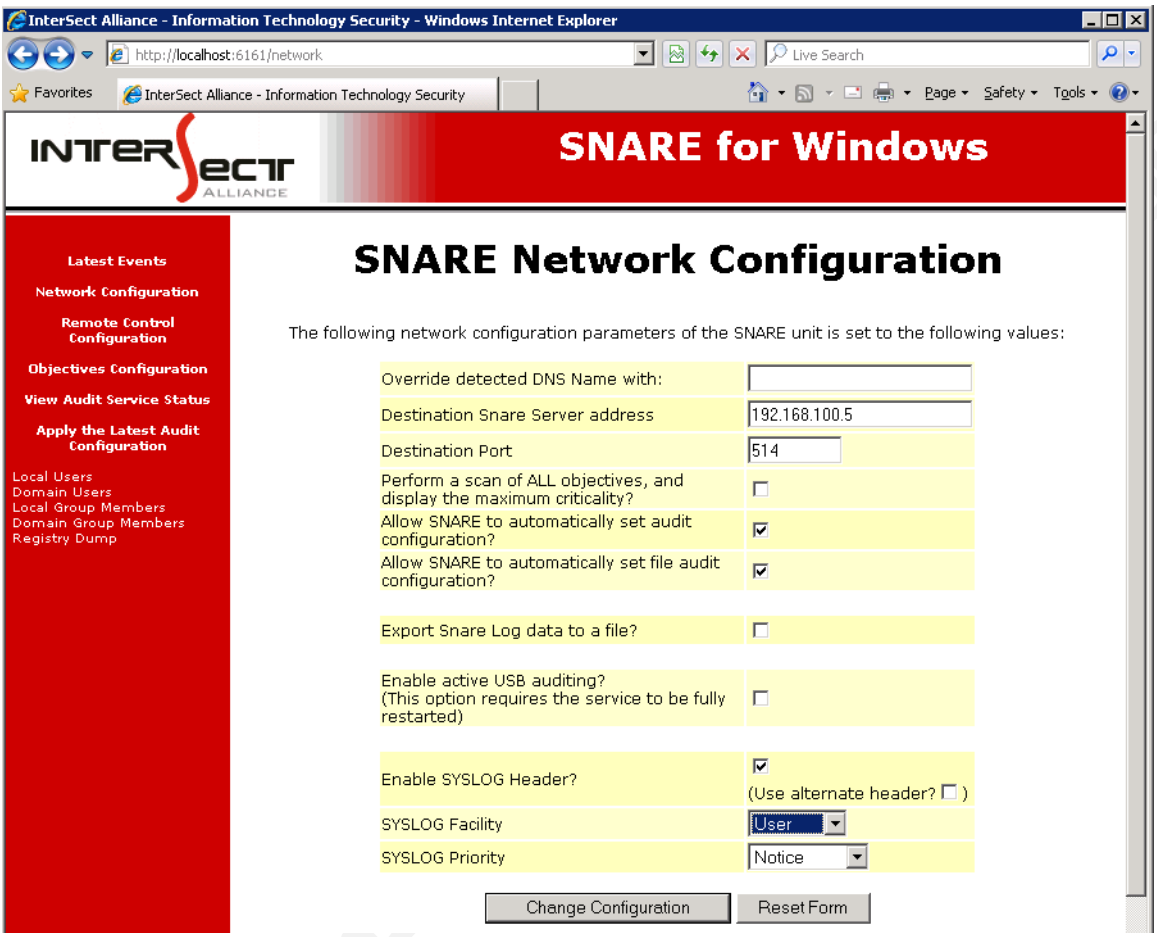
Following are different examples of the syslog forwarding configuration of clients.

Log forwarding by Adiscon EventReporter agent on Windows host to remote syslog server (192.168.100.5)



Sunil Gupta, sgupta911@gmail.com

Log forwarding by SNARE client on Windows host to remote syslog server (192.168.100.5)



Log forwarding by configuring the syslog-ng client on linux/unix hosts to remote syslog server (192.168.100.5)

Update `/etc/syslog-ng/syslog-ng.conf` as below and restart syslog-ng

Add a destination that points to the syslog server.

```
destination central_log_host { udp("192.168.100.5" port(514));};
```

Then, Add a new log entry that uses the destination just added.

```
log { source(src); destination(central_log_host);};
```

Log forwarding by configuring the rsyslog client on linux/unix hosts

Update the `/etc/rsyslog.conf` with entry below and restart rsyslog afterwards.

```
*.*     @192.168.100.5:514      # forwards log to remote syslog
server.
```

Sunil Gupta, sgupta911@gmail.com