Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

Practical Implementation of Syslog in Mixed Windows Environments for Secure Centralized Audit Loggin

Secure system logging is a cornerstone of a well-designed layered network security policy. Collection and timely analysis of system auditing, event and security logs is critical to ensuring that network security personnel can effectively audit the network and its components for evidence of many types of security events. One of the frustrations for systems administrators working in a Windows or mixed Windows and Unix-based operating system environment is the paucity of centralized logging tools. All Unix based operating...

Copyright SANS Institute

Author Retains Full Rights

AD

Author: Frederick C Garbrecht
GSEC Practical Assignment Version 1.4 Option 2

Practical Implementation of Syslog in Mixed Windows Environments for Secure Centralized Audit Logging

Abstract
Secure system logging is a cornerstone of a well-designed layered network security policy. Collection and timely analysis of system auditing, event and security logs is critical to ensuring that network security personnel can effectively audit the network and its components for evidence of many types of security events. One of the frustrations for systems administrators working in a Windows or mixed Windows and Unix-based operating system environment is the paucity of centralized logging tools. All Unix based operating systems have implementations of the Syslog protocol, which facilitates the centralized remote collection of system messages from network devices, workstations and servers. Windows operating systems in contrast record operating system and process auditing data to the system event logs via the Windows Event log service. The Event log service is by design a distributed system, and there are no native Windows tools available to facilitate centralization of logging functions. In addition, the failure to conform to any external logging format standard makes it impossible to interoperate with the logging functions of other operating systems or network devices. The Windows Event viewer application offers only basic functionality and is inadequate for monitoring the audit log files of any medium to large size network. In this paper, I survey some of the options available to access the Windows Event log and demonstrate how to implement a versatile centralized remote logging solution using a commercially available Win32 implementation of the Syslog protocol.

Description of the Syslog Protocol

Syslog is a protocol designed for the efficient delivery of standardized system messages across networks. The protocol is described in detail in RFCs 3164 and 3195. Syslog originated in the Unix family of operating systems and has found its most extensive use there, but other implementations have been coded, including ones for Windows operating systems. The way syslog works is that syslog aware processes (the operating system and application programs) generate and send messages to a local syslog daemon. The syslog daemon receives the message (according to the specification, any udp message received on port 514 must be interpreted as a syslog message) and then stores it locally in the host file system or forwards it to a syslog daemon elsewhere in the network using transport over udp port 514. The Syslog message consists of a line of text containing a PRI code, a HEADER section, and MESSAGE. The PRI (priority) string encodes the syslog facility and severity strings as a single decimal number enclosed by angle brackets. Facility strings allow categorization of events by source type; by convention, the facility codes have been assigned to specific operating system or application functions. For example, facility code 0 represents OS kernel messages, and facility

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

4 represents security and authorization messages. Facility codes 16 through 23 are assigned to the 'local user' facilities numbered 0 through 7 (local0, local1, etc.), and can be defined to suit the developer. There are 8 hierarchical levels of severity codes (0 through 7) defined as Emergency, Alert, Critical, Error, Warning, Notice, Informational, and Debug (the smaller the severity code, the greater the severity). The PRI string is obtained by multiplying the numeric FACILITY code by 8 and adding the SEVERITY code; thus each possible combination of FACILITY code and SECURITY code produces a unique PRI code. The HEADER portion of the syslog message consists of a TIMESTAMP and a HOSTNAME field. The HOSTNAME contains the hostname, Ipv4 or Ipv6 address of the originator. Finally, the MESSAGE portion contains a TAG field and a CONTENT field. The TAG simply identifies the originating process or program, and the CONTENT contains the actual message sent by the originating process.

Since Syslog uses udp transport, there is no guarantee that the message actually arrives at the addressee, nor is there notification of non-delivery. In addition, syslog packets may arrive late or out of order, so that careful reading of timestamps may be required when reviewing audited events. Also worth noting is that there are no security mechanism defined *per se* in the Syslog specification. An internet draft has been written that attempts to resolve some of these shortcomings by defining the 'Syslog-sign' protocol, "an enhancement of syslog [RFC3164] that adds origin authentication, message integrity, replay resistance, message sequencing, and detection of missing messages to syslog." (Kelsey & Callas)

Syslog has been widely implemented by many hardware device vendors; consequently syslog is the *de facto* standard for collecting messages from the majority of devices found on modern networks including servers, workstations, printers, routers, switches, etc.

Windows Event Logging

Windows operating systems (Windows NT, 2000, XP) and applications produce audit data that are written to the Windows event log in a binary format (in contrast to Unix logs and Syslog messages which are written in plain text). To examine event log data, one must use the Windows Event viewer application. The Event viewer application has limited capabilities, and the user is restricted to sequential viewing of individual event log entries. Although data can be alphabetically sorted on any field, and can be exported in delimited text format, no search capabilities exist and no advanced queries are possible within Event Viewer. Because even a well functioning network of moderate size will generate a large number of events (assuming that appropriate security audit settings have been chosen), sifting through them in this manner is inefficient. A second limitation of the Windows Eventlog service is its distributed design; each individual event log resides locally in the registry of the host workstation; centralization of logging functions is not provided as an option. The administrator may view the event logs of remote networked computers, but each remote machine's logs must be examined individually.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

In addition, the Windows event log is incapable of responding to messages originating from network devices such as router and switches. Incorporating auditing log data from network devices into a unified database of network event messages is therefore not possible using native Windows tools.

In consequence of the inherent limitations of Windows event logging, either external solutions to central log management must be found, or (very likely) log management and analysis is performed poorly, if at all. Efficient and timely event log auditing is essential for effective systems administration, and it is absolutely critical for the detection and analysis of network security breaches.

Solving the Logging Problem in Windows Networks

Ideally, a logging system operating in pure Windows or mixed operating system environments would overcome the limitations of the Windows event log and solve the following problems:

- Logs should not be written in proprietary format; analysis capabilities must be flexible, using tools appropriate to the requirements of the system administrator.
- Network auditing events are collected centrally, including servers, workstations, network printers, routers and switches, and other attached network devices, regardless of device type or operating system. Central collection eases the administrative burden by automatically dumping or streaming network events into a single database (if desired). Centralization allows critical security events to be collected into a protected internal network location that is less likely to be susceptible to log file manipulation by an attacker. Centralization also facilitates time coordination of log entries.
- The logging system can be distributed to multiple collection points if desired. Log data can be routed to distributed databases according to specific message criteria, or data can be routed to multiple identical routing servers for redundancy so that there is no single point of failure within the logging system.
- The logging system itself can generate events in response to received messages; these might include the ability to notify an administrator by generating a 'net send' or by sending SMS messages, email, or pager notifications.

System administrators and programmers have crafted numerous solutions to these problems, ranging from creative batch files and perl scripts, to limited standalone command line applications to full blown Win32 application suites. Some of these solutions are summarized in Table 1. The data in this summary is not authoritative; the Table is not intended to be exhaustive, but reflects my own experience in trying different solutions to the logging problem.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Table 1 Syslog Daemons and Tools for Manipulating Windows Event Logs

Tools for dumping event log data to file		
Program Name	Source	Comments
Dumpel.exe	Windows NT Resource kit	Command line utility to dump an Eventlog (local or remote systems) into a tab-delimited text file; limited to dumping entire file; Commercial
Dumpevt.exe	Somarsoft	Command line utility to dump local or remote event logs to delimited text files suitable for import to database; versatile in scripting applications; Commercial
ELDump.exe	Jesper Lauritsen	Command line utility to dump local or remote event logs, many command line options; Free
Win32::EventLog	Jesse Dougherty	Perl module allows processing of Windows event logs from perl scripts
Tools for dumping event log data to syslog server		
BacklogNT.exe	Intersect Alliance	NT service converts Event log events in real-time to syslog messages, simple GUI configuration; Free, GNU Public License
NTSyslog	Jason Rhoads	NT service converts event log entries and sends to syslog host; GNU GPL, Free
Event Reporter	Adiscon	NT service converts event log events in near real-time to remote or local syslog servers; very configurable with many options via GUI interface; Commercial
Windows Administration Suites		
Hyena	System Tools Software	Remote viewing of Event logs, allows sorting and filtering based on keyword
Windows syslog daemons		
Kiwi syslog daemon	Kiwi Enterprises	Full featured, flexible syslog daemon Free and commercial versions
Winsyslog	Adiscon	Full featured, flexible syslog daemon; Commercial
Syslog Daemon	Tri Action Software	Syslog daemon, Commercial
SL4NT	Franz Krainer	Syslog daemon, Commercial

One common approach to centralization of logging functions in Windows networks is to use the Windows scheduler service (or batch files or scripts) and the NT resource kit utility dumpel.exe to automate collection of Eventlog entries. This solution suffers from the limited functionality of dumpel.exe, most importantly its inability to dump portions of Eventlogs based upon event number or timestamp rather than the full log. Thus one is forced to recycle each Eventlog after running the job; otherwise the central log file fills with redundant entries and grows unmanageable very quickly.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

The utilities Dumpevt.exe and ELDump.exe solve this problem by allowing the retrieval of Eventlog entries according to user specified criteria. Both utilities are versatile and can be incorporated into quite workable scripting solutions for Windows Eventlog centralization. For perl programmers, a perl module (Win32::Eventlog) can be used that makes the Windows Eventlog APIs accessible.

An additional solution is the inclusion of general auditing and logging functions within complete systems administration and monitoring packages. For example, Hyena (System Tools Software) allows access to remote Event logs in Windows networks. I have found this suite to be useful for its general system administration tools, and I use its Event log tools to examine workstation Event logs which have not otherwise been incorporated into our centralized logging system. It does not allow viewing of log files on devices or non-Windows computer however, and it does not physically centralize the collection of log file data.

Syslog implementations in Windows networks operate somewhat differently than in non-Windows networks. Whereas processes running under Unix generally understand that event messages should be sent to the local syslog daemon, processes and applications running under Windows have not been programmed to comply with the syslog specification; rather they are accustomed to sending their messages to the default Windows logging service. Therefore, a separate application or functionality is required that performs a translation between Windows native Event log binary format and syslog protocol message format. This functionality may be programmed into a separate application altogether or incorporated into the Win32 syslog daemon application. In either case, the application obtains the raw Event log data by intercepting messages going to the Event log or by polling the Event log on a periodic basis. Messages intercepted or retrieved by polling are forwarded to a syslog daemon running locally or remotely either on a Unix or Windows host.

Making Windows Talk Syslog Using Winsyslog

Before: Network Security Posture Before Implementing Centralized Logging
Prior to the logging implementation described in this paper, I relied on the logging and auditing settings available by default in any Windows NT/2000 network. Like many others, I suspect, our network had initially been constructed in an ad hoc fashion absent any guiding framework of a security policy. The first step therefore was to perform security auditing to ascertain and document the existing vulnerabilities, and to begin to define an appropriate security policy that was workable within the context of our organizational structure, needs and capabilities. This is a continuously evolving process and mostly beyond the scope of this paper; however, it was clear early in the audit process that auditing and logging represented an important point of vulnerability. The absence of a security policy was most evident in the lack of consistent settings for implementation of Windows auditing. In most cases the Windows operating system *default auditing settings* were found to be in place. In our Windows 2000 workstations, that meant that

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

nothing was written to the Security event log at all. The obvious vulnerability here is that the administrator has no audit trail or log information from which to work in the event that an intrusion or other security incident has occurred. Similar findings were made when our networking devices were examined; for example, the border router had been configured without any access control lists whatsoever, nor had logging been implemented.

The network which I wished to 'unify' by means of a centralized logging solution is a fairly typical small corporate firewalled network consisting of 3 segments: an external (internet) segment, a DMZ which houses our externally accessible servers (mail, web, ftp) and an internal protected segment containing user workstations and internal servers. The network is separated from the external world by a router, and traffic is controlled through and between the network segments using a software firewall. In addition, a Linux based server running Snort (an intrusion detection system) is located in the internal segment. Each of these devices generates (or has the potential to generate) audit logs documenting important events regarding its operation. Each can generate important records of security policy violations, and it is important that the administrator have ready access to the logs for analysis

To summarize the auditing related problems identified within my network prior to implementation of a general security solution:

- No formal security policy
- No consistent policy for security auditing settings
- No security logging enabled on many workstations, servers and networking devices
- Examination of logs cumbersome and time consuming using available tools (Event viewer)
- Log formats among operating systems and network devices inconsistent
- No ability to correlate security logs between different operating systems and devices

The vulnerabilities and risks associated with these problems are very real. Clearly the absence of a security policy and inconsistency in security auditing settings increase the likelihood of adverse security events because an attacker is less likely to encounter obstacles to intrusion into the network. Intrusion attempts may not even be recognized if the appropriate auditing events are not being recorded. And if the event logging system is so cumbersome that the administrator is averse to regularly examining the event logs, security events are not likely to be recognized until they become disastrously obvious. Even if the security events are recorded, the analysis and correlation of events is quite difficult or even impossible when the logging system is in multiple log formats and distributed all over the network.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

During: Evaluating Possible Solutions and Implementing Centralized Syslog
After completing the security analysis of our network and determining that the distributed audit logging system represented significant risks to our organization, I began to investigate possible solutions. First, it was necessary to define a consistent policy for security auditing settings so that security audit logging would be enabled. The audit settings that I use throughout our network are shown in Table 2, and are largely derived from the recommendations of Nordberg.

Table 2 Windows 2000 Security Audit Settings

Audit Setting	*Recommended Setting*
Logon and Logoff / Audit logon events	Success and failure
File and Object Access / Audit object access	Success and failure
Use of User Rights	None
User and Group Management	Success and failure
Security Policy Changes	Success and failure
System event audit	Success and failure
Process Tracking	None

Although auditing of 'Use of User Rights' and 'Process Tracking' may be useful, they tend to generate huge amounts of log data, and it is recommended that they not be routinely audited for that reason.

The primary goal as it pertained to a distributed logging system however, was to facilitate routing of all important auditing and logging activity from our network devices (switches and router), DMZ servers, intrusion detection system, network printer, firewall and internal servers to an internal logging host. In this way, logs could be collected in real time on a protected logging host, and all alerting, notification and database processing tasks could be implemented in a single safe location. I evaluated and experimented with most of the software solutions outlined in Table 1 with varying degrees of success. The 'non-syslog' solutions (e.g. ELDump, DumpEvt) actually worked well in resolving some of the inherent limitations of Event viewer, but still are limited to gathering log data from Windows machines; other devices and computers running Unix are left out. Taking a cue from the Unix world and implementing the syslog protocol under Windows seemed however to make the most sense because it had the potential to unify logging amongst almost every device in our network.

I implemented a commercial application suite produced by Adiscon IT Solutions GmbH. Adiscon's product provides an integrated syslog solution running under Win32. This suite consists of the following components:

- Syslog daemon (Winsyslog)
- Event log to syslog message format translation application (Event Reporter)
- Analysis application (Monilog).

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Event Reporter retrieves data from Windows event logs by polling (at a user definable interval), performs the translation into Syslog message format, and sends the record to a syslog (Winsyslog) server, where further forwarding or writing to the local disk subsystem occurs. Monilog may then be used to analyze the collected data. Winsyslog receives syslog messages from any source. It can therefore be used to collect and centralize messages for an entire network, including Windows and Unix hosts, printer, backup devices, routers, etc. As long as a server or device can reach the network and can send messages conforming to the syslog specification, Winsyslog should be able to receive them.

Software Installation

The first step prior to installation is to plan the logging system. This should be a straightforward process but could present some challenges in more diverse or geographically dispersed auditing environments. It may not be practical to centrally monitor auditing data from all internal user workstations because of the volume of data arriving at the logging console and the cost of licensing many software installations. Event Reporter may be configured to extract and send only the specific auditing data of most interest, so the volume of data sent centrally may be reduced considerably. For example, the administrator may wish to have only centralized records of failed remote workstation login attempts and changes in local security policy. I decided to forgo local workstation logging however mainly for reasons of licensing cost. I set up a single dedicated internal workstation running the syslog daemon application to receive syslog messages from the critical servers and devices in the network only. The installation of the component applications of the Winsyslog application suite is trivial; each uses standard windows installation programs. The core application is Winsyslog, the syslog daemon application, which should be installed on the central logging host. Additional instances of Winsyslog may be desirable to provide redundant backup logging or as relay hosts across network segments. Event Reporter is installed on each Windows machine from which event log reporting is desired, and I placed a copy on each of our internal and DMZ servers, and on the firewall. Monilog may be installed wherever it is convenient to establish an analysis workstation.

Software Configuration and Capabilities

The firewall must be configured to pass udp port 514 traffic so that legitimate syslog traffic originating from the DMZ may pass into the protected network (where the syslog daemon is located). The source and destination syslog hosts should be explicitly defined in the firewall rulebase to minimize the opportunity for spoofed non-syslog traffic masquerading as legitimate syslog traffic. In addition, as there is no reason for syslog traffic to travel on the internet, your border router(s) should be configured to prohibit udp/514 traffic from passing in or out.

Winsyslog is a typical syslog daemon in that it can be a final destination syslog process, or it can forward to syslog relay hosts or to a final remote syslog destination. Winsyslog conforms to the syslog protocol specification in using udp transport listening on port 514 but can be configured to listen on other user-defined

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

ports. In addition, the application can use a proprietary non-syslog tcp transport protocol, which offers greater reliability, albeit at the expense of greater overhead. Complex rules can be created to perform a multitude of functions ranging from saving forwarded messages to database files (via ODBC connector) or to flat files in delimited text. Messages can be discriminated based upon originating IP address, content and other variables and discarded, forwarded or saved, and used to trigger a number of events including firing of email or pager notifications. Figure 1 shows the configuration application for Winsyslog. As can be seen in the figure, I have implemented separate rules which handle message forwarding, logging messages to text file, logging to an ODBC database file, and notification of several important events, such as mail server relay events, logon failures, and remote access service (RAS) usage.

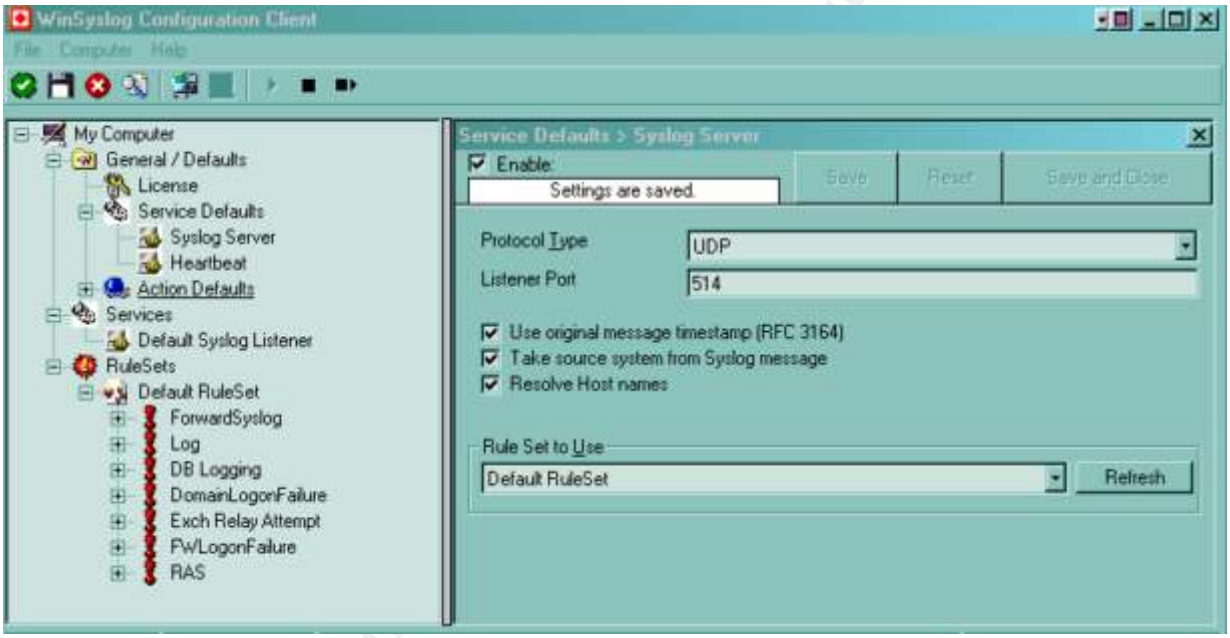

Fig. 1 Winsyslog Main Configuration

An additional useful feature is real-time display of syslog messages coming into the central logging server, as shown in Figure 2. The message arrival times are actually slightly delayed and are dependent on the polling interval selected the Administrator when configuring Event Reporter at each computer.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

	Time	Facility	Priority	RealSource	Message
33	7/3/2002 1:39:03	LOCAL4	ERRO-	192.168.2.130	Jul 3 13:39:03 192.168.2.130 -realsource="192.168.2.130" EvntLog: [ERR]Wed Jul
32	7/3/2002 1:39:03	LOCAL4	WARNING	192.168.2.130	Jul 3 13:39:03 192.168.2.130 -realsource="192.168.2.130" EvntLog: [WRN]Wed Ju
31	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
30	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
29	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
28	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
27	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
26	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
25	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
24	7/3/2002 1:38:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:38:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
23	7/3/2002 1:37:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:37:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
22	7/3/2002 1:37:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:37:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
21	7/3/2002 1:37:31	LOCAL1	NOTICE	192.168.2.1	Jul 3 13:37:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [AUD]Wed Jul 03
20	7/3/2002 1:36:43	LOCAL0	INFO	63.174.65.1	Jul 3 13:36:43 63.174.65.1 RealSource "63.174.65.1" 38465: Jul 3 14:34:13 EST:
19	7/3/2002 1:36:43	LOCAL0	INFO	63.174.65.1	Jul 3 13:36:43 63.174.65.1 RealSource "63.174.65.1" 38465: Jul 3 14:34:13 EST:
18	7/3/2002 1:36:31	LOCAL1	WARNING	192.168.2.1	Jul 3 13:36:31 192.168.2.11 RealSource "192.168.2.1" EvntLog: [WRN]Wed Jul 03
17	7/3/2002 1:35:43	LOCAL0	INFO	63.174.65.1	Jul 3 13:35:43 63.174.65.1 RealSource "63.174.65.1" 38464: Jul 3 14:33:13 EST:
16	7/3/2002 1:34:43	AUTH	ALERT	192.168.2.178	Jul 3 13:34:43 192.168.2.178 -realsource="192.168.2.178" error:: [1:1480:2] ICMP PING
15	7/3/2002 1:34:43	AUTH	ALERT	192.168.2.178	Jul 3 13:34:43 192.168.2.178 RealSource="192.168.2.178" error:: [1:1480:2] ICMP PING
14	7/3/2002 1:34:33	AUTH	ALERT	192.168.2.178	Jul 3 13:34:33 192.168.2.178 RealSource="192.168.2.178" error:: [1:1480:2] ICMP PING
13	7/3/2002 1:34:33	AUTH	ALERT	192.168.2.178	Jul 3 13:34:33 192.168.2.178 RealSource="192.168.2.178" error:: [1:1480:2] ICMP PING

Fig. 2 Real-time Winsyslog message display

The default polling interval is set to 5 minutes and works well in our network.

The Event Reporter main configuration screen is shown in Figure 3. The receiving syslog server is specified here by IP address, and the provision exists for forwarding message copies to an additional backup server as well if desired. A number of additional configuration options are demonstrated in Figure 4. The system, security and application event logs can be configured for syslog messaging individually in machines running NT; furthermore, directory, DNS, and file replication event logs may be configured as well in Windows 2000 machines. Specific categories of auditing events (success, error, warning, etc.) are configured here; additionally, very fine control of syslog messaging is possible using the 'enable filter rules' capability.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

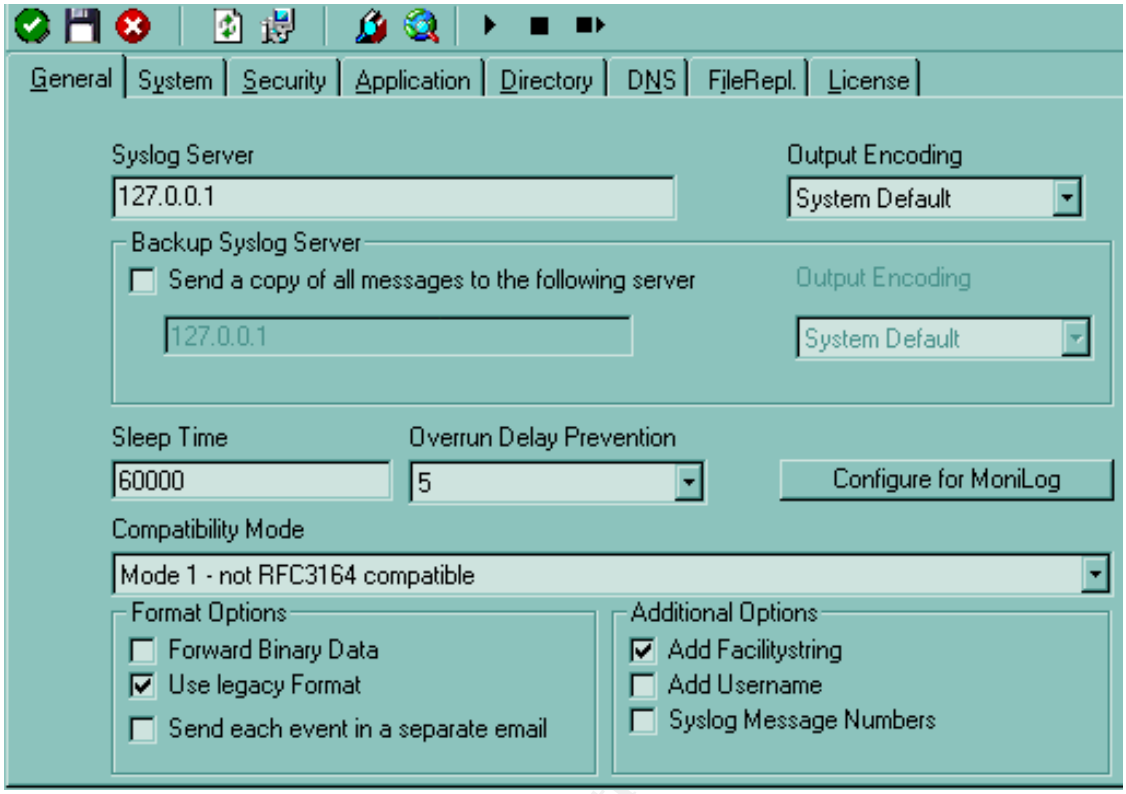

Figure 3 Event Reporter Configuration

Specific auditing events can also be sent via email, either from Event Reporter rules or configuration within Winsyslog. The specific configuration details of course will vary with the requirements of your network. In most cases, I have stayed with the default settings as much as possible. The applications' configuration options however lend a great deal of flexibility to the implementation of centralized logging.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

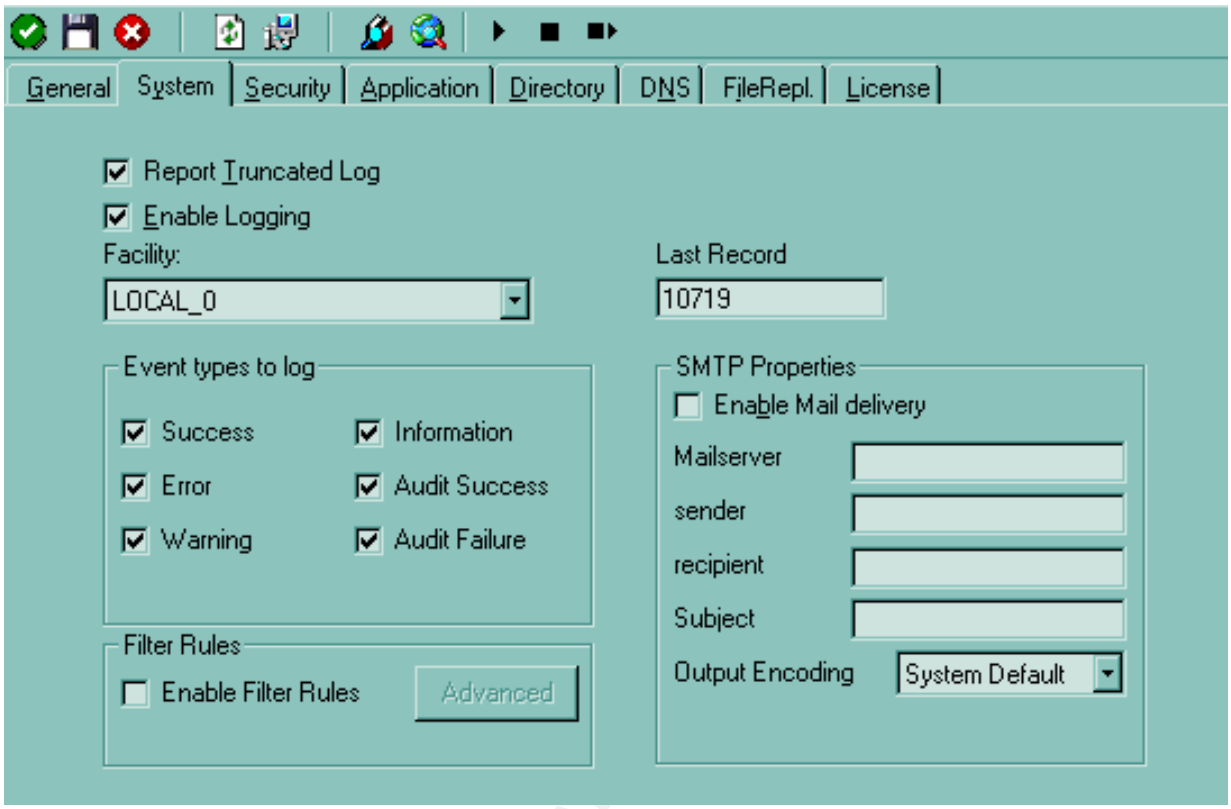

Fig. 4 Event Reporter Configuration

The final component of the software application picture is Monilog, which provides a painless way of rapidly reviewing logged events. The Monilog main screen is shown in Figure 5. Monilog produces html format reports from text formatted log files generated by Winsyslog. Reports can be generated on a recurring schedule or on an ad hoc basis, and the report contents can be configured by specific reporting server and time frame, as well as by filtering on message ID number or on keywords in the message body; thus it is quite easy to find specific auditing events of interest. An example of the html output from Monilog is shown in Figure 6. One disadvantage of Monilog is that it ignores syslog messages received from sources other than Event Reporter enabled servers. Therefore it does not report events from Linux servers, routers or other network devices. To overcome this shortcoming, I routinely supplement use of Monilog with analysis of the central ODBC logging database maintained by Winsyslog with specially crafted queries to identify specific auditing events of interest on our network.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

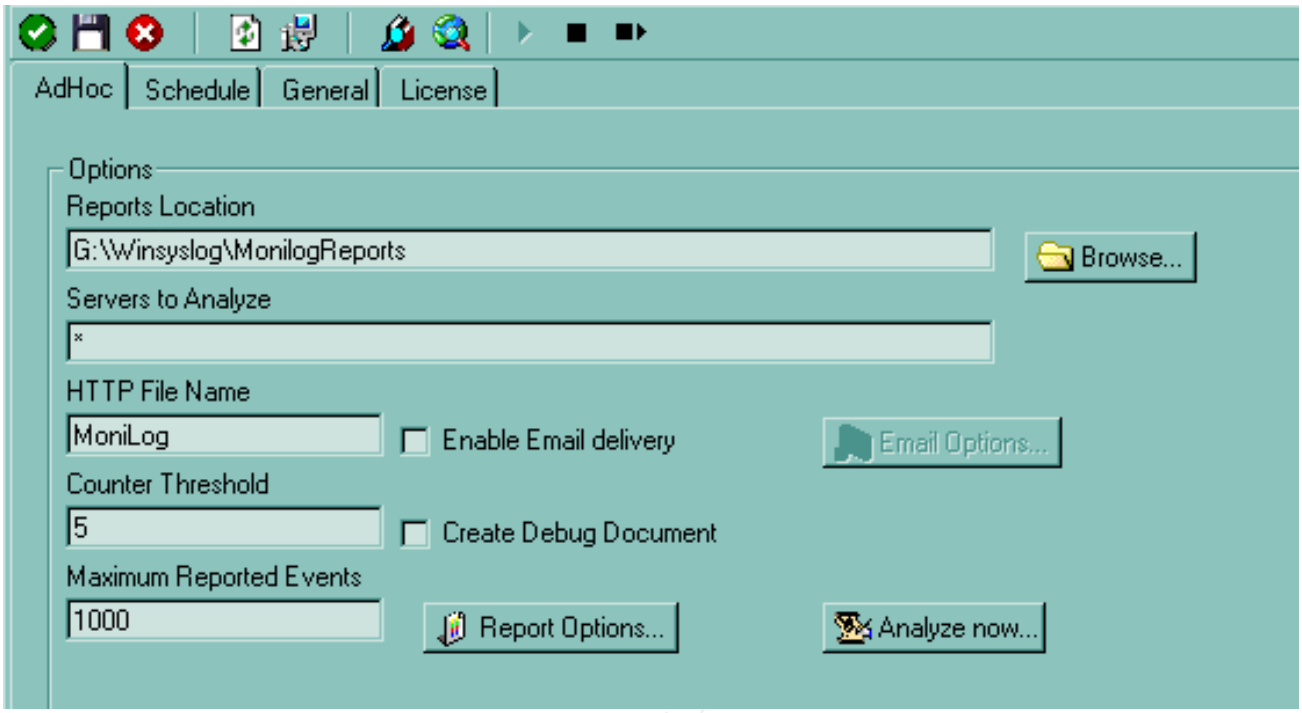

Fig. 5 Monilog Main Configuration Screen

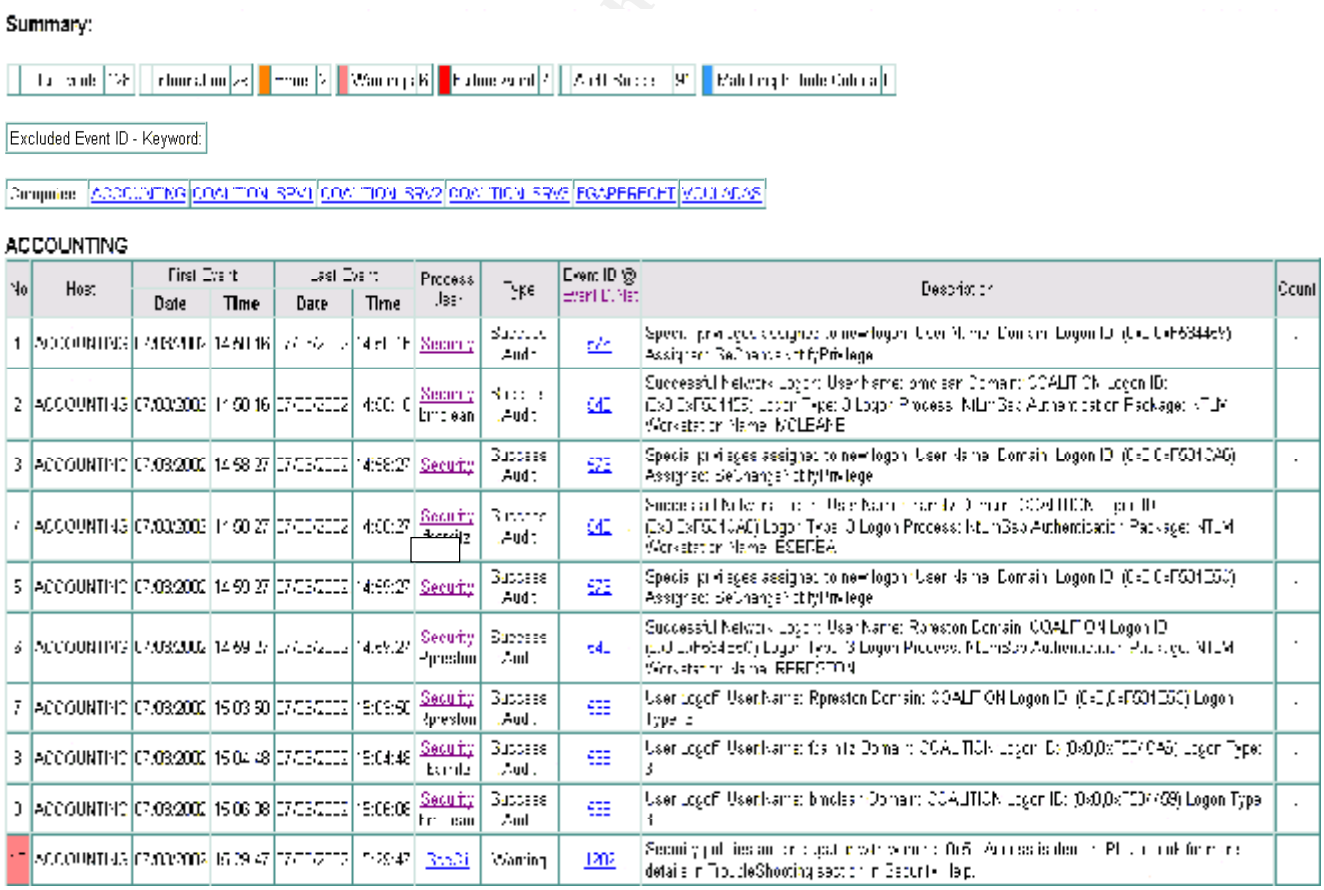

Fig. 6 Monilog HTML Output Example

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Configuring Unix hosts to transmit Syslog information
Hosts running Unix, Linux, BSD, etc can also be configured to originate and/or receive syslog messages in a mixed OS network. Assuming that Unix-based hosts running in the network will be logging to a central Windows server, the following configuration steps need to be performed:

1. Tell the OS to listen on the standard syslog port by adding the statement "**syslog 514/udp**" to /etc/services.
2. Tell syslog the type of messages to send, and the network location of the remote syslog daemon by adding the statement **"*.* @hosthame**" to /etc/syslog.conf

The wildcard designation *. * in the preceding statement represents the message type and directs syslogd to forward ALL log messages to the host specified by @hostname. Fine control of message forwarding can be obtained by replacing the wildcard designation with one or more specific facility name and priority strings, in the format 'facility'.'priority'. For example, to send kernel messages of alert priority (and above) only, use the syslog.conf statement **kern.alert @hostname**. The specific details may vary by Unix dialect or Linux distribution, so the syslogd man page should be consulted before making use of this information.

Configuring Network Devices to transmit Syslog Information
Cisco routers can be configured to transmit logging information using Syslog. Cisco IOS utilizes 8 levels of logging messages (level 0 through level 7) corresponding directly to the Severity codes defined in Syslog. Adding Syslog is as simple as adding logging hostname and logging trap commands; a sample IOS configuration to implement Syslog logging might appear as follows:

```
Cisco#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Cisco(config)#logging 192.168.1.153
Cisco(config)#logging trap debugging
Cisco(config)#^Z
```

The **logging 'host ip'** command directs the router to send Syslog formatted messages to the Syslog server located at 192.168.1.153. The **logging trap debugging** command specifies the minimum level of logging to be sent to the Syslog daemon. As 'debugging' is the lowest level of severity (level 7), all log messages will be sent. To send only the most critical messages, one might choose instead to use the command **logging trap emergencies** or **logging trap alerts**. These commands enable only logging of IOS internal messages.

It is recommended that you also implement logging on your IOS access lists as well. A very simple example follows:

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
Cisco#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Cisco(config)#ip access-list extended byebye
Cisco(config-ext-nacl)#permit tcp any host 192.168.2.25 eq smtp
Cisco(config-ext-nacl)#deny ip any any log
Cisco(config)#^Z
```

This bit of IOS code defines an extended IP access list named 'byebye' that allows all tcp traffic coming into host 192.168.2.25 on port 25 (smtp) to pass. The 'deny' statement then blocks any traffic that does not meet the preceding criteria, and sends a log message to the Syslog host. Utilizing IOS access list logging is an important tool to keep tabs on what kind of traffic is hitting your router, what is getting through and what is being blocked. I have found that this information is invaluable in fine-tuning my access lists. Additional detailed information about configuring logging (and many other things) in Cisco routers can be found in Leinwand and Pinsky.

Additional devices may be configurable as well; consult the relevant manufacturers documentation for instructions, as the specific implementations and configuration details vary widely.

***After**: Network Security Posture After Implementing Centralized Logging*

Our network now has a relatively complete centralized logging host running Winsyslog located within the internal protected segment of the network. The program is configured to collect syslog messages to a local drive, as well as to an access database residing on an external 120 gigabyte attached hard drive via Winsyslog's ODBC capabilities. Database and log file backups are created on CD-ROM periodically. Centralization of the logging function serves to protect critical audit data from attackers by removing it almost immediately from the machines on which it is generated (and may themselves be the subject of attack). Each Windows server running in the DMZ or internal network segment runs Event Reporter, and syslog messages from these machines are directed at 5-minute intervals to the central logging server. The firewall also runs Event Reporter; the operating system Event logs are therefore sent to the logging host, although the firewall logs themselves have not been incorporated into the system. The border router as well is configured to log to the central logging host, and IOS router events as well as violations of the router ACL's are recorded in the central log. Finally, the logs produced by the intrusion detection system are incorporated as well into the central logs. A monitor attached to the logging host shows a real-time display of syslog data arriving from the network (Figure 2), so the administrator can see time-correlated data arriving from multiple sources and may be alerted to ongoing security events. Furthermore, Winsyslog is configured to send SMS messages (short text messages) to the administrators cell phone upon receipt of specific security events such as repeated logon failures or high-risk events detected by the intrusion detection

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

system. I have configured Monilog to produce tailored reports on each reporting device, which are run automatically each morning and are examined by the administrator as a first task of the day. In addition, a variety of access queries have been written and are available for daily summary or weekly reviews of security related audit events. In this way, centralization of logging and the automatic generation of alerts (via SMS messaging) and daily reports has largely solved the problem of timely access and review by systems personnel of security related audit logs.

Risks and Vulnerabilities: Room for Improvement
The process of implementing a centralized syslog logging system in our network has had an important impact on the security posture of the network and has ameliorated to large degree the security vulnerabilities and risks related to system auditing and logging that were found to exist in our initial security audit. The central points are defined as follows:

- A security policy was drafted which defines system wide goals for security event auditing and collection of audit data
- Centralized logging using the syslog protocol allows the collection of audit data from almost all event generating devices present within the network, thus greatly simplifying the analysis and correlation of security events
- Centralized logging allows the collection of data in a hardened, protected logging host where it is far more difficult for an attacker to alter or destroy logging data
- Automated centralization frees up a great deal of time for the system administrator to actually examine audit files rather than running around collecting them, and automated reporting makes routine examination of audit records on a daily basis feasible; impending problems and ongoing intrusions are far less likely to be overlooked under these conditions

There are several areas in the implementation of the centralized logging server that could benefit from some improvements to enhance security or ease of use. These include:

- Despite inclusion of virtually every important source of auditing data within the network, our Hewlett-Packard Procurve switch cannot be configured to send its audit data by syslog protocol. Since many devices (including HP switches) can send status messages using SNMP traps, one possible solution to this problem would be to incorporate SNMP traps into the centralized logging system. This is not a transparent process, but Kiwi syslog daemon is able to listen for SNMP traps in addition to its primary function as an NT syslog daemon. I am currently experimenting with this software in an effort to incorporate switch generated SNMP traps into our centralized logging solution.
- The central logging host could serve as a central point of failure. A better solution might be to run a pair of logging hosts, each receiving the same syslog messages concurrently as syslog endpoints; alternatively one host

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

could receive and log messages and then forward the message along to the endpoint syslog host. The former configuration would have the advantage that messages that might be lost in one part of the network and fail to reach one of the syslog hosts might be able to reach the remaining host via another network pathway.

- Centralized logs could be subject to manipulation. The logging host could write directly to a non-rewritable medium, such as a CD-ROM drive. In this way, an intruder would be unable to alter log files to cover their tracks.
- Logging data is neither authenticated nor encrypted in transit from its source to the final syslog daemon. It is possible that an attacker could intercept and alter syslog data in transit, or inject false syslog data into the data stream. In security critical environments, syslog data encryption could be achieved by implementation of ssh or open-ssh, which is implemented in Cisco IOS, and versions of which are available for both Windows and Linux.
- Time synchronization between servers, devices and logging hosts has not been implemented yet. Time synchronization is obviously a critical element in the correlation of disparate event logs and is required for forensic investigations of network intrusion events.

Summary

Setting up a centralized secure logging system in a mixed operating system environment is well worth the considerable effort it may take. There are several viable options available. Although none are perfect in terms of breadth or ease of implementation, most function well enough to solve at least some of the problems associated with the Windows distributed logging system. I have successfully incorporated Adiscon's suite of syslog products in our corporate network to serve as the central element of an integrated logging system. Besides offering simplification of the administrative overhead of collecting and monitoring network audit data, centralized logging offers significant benefits in terms of security risk reduction.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

References

"The BSD Syslog Protocol", RFC 3164
http://www.ietf.org/rfc/rfc3164.txt

"Reliable Delivery for Syslog", RFC 3195
http://www.ietf.org/rfc/rfc3195.txt

Murray, James, Windows NT Event Logging, Sebastopol, O'Reilly & Associates, 1998

Kelsey, John and Callas, Jon, "Syslog-Sign Protocol", Network Working Group
http://www.ietf.org/internet-drafts/draft-ietf-syslog-sign-06.txt

Leinwand, Pinsky, "Cisco Router Configuration, Indianapolis, Cisco Press, 2001

Nordberg, Stefan, Securing Windows NT/2000 Servers for the Internet, Sebastopol, O'Reilly & Associates, 2001

Software References

Kiwi Syslog Demon, Kiwi Enterprises
http://www.kiwisyslog.com/

Winsyslog, Adiscon IT Solutions GmbH
http://www.adiscon.com/

DumpEvt, Somarsoft
http://www.somarsoft.com/

ELDump, Jesper Lauritsen
http://www.ibt.ku.dk/jesper/ELDump/default.html

BackLogNT, Intersect Alliance
http://www.intersectalliance.com/projects/BackLogNT/index.html

Event Reporter, Adiscon IT Solutions GmbH
http://www.eventreporter.com/en/

NTSyslog
http://ntsyslog.sourceforge.net/
http://sabernet.home.attbi.com/software/ntsyslog.html;

Syslog Daemon, Tri Action Software
http://www.triaction.nl/Products/SyslogDaemon.asp

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

SL4NT, Franz Krainer, netal
http://www.netal.com/

Win32::EventLog, Jesse Dougherty
http://search.cpan.org/search?dist=libwin32

Hyena, System Tools Software
http://www.systemtools.com

© SANS Institute 2002, Author retains full rights.

Last Updated: August 28th, 2015

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

FOR578 Cyber Threat Intelligence	**Tysons Corner, VAUS**	**Aug 31, 2015 - Sep 04, 2015**	**Live Event**
SANS Milan 2015	**Milan, IT**	**Sep 07, 2015 - Sep 12, 2015**	**Live Event**
SANS Crystal City 2015	**Crystal City, VAUS**	**Sep 08, 2015 - Sep 13, 2015**	**Live Event**
SANS Network Security 2015	**Las Vegas, NVUS**	**Sep 12, 2015 - Sep 21, 2015**	**Live Event**
SANS Seoul 2015	**Seoul, KR**	**Sep 14, 2015 - Sep 19, 2015**	**Live Event**
SANS Tallinn 2015	**Tallinn, EE**	**Sep 21, 2015 - Sep 26, 2015**	**Live Event**
SANS Perth 2015	**Perth, AU**	**Sep 21, 2015 - Sep 26, 2015**	**Live Event**
SANS Baltimore 2015	**Baltimore, MDUS**	**Sep 21, 2015 - Sep 26, 2015**	**Live Event**
Data Breach Investigation Summit & Training	**Dallas, TXUS**	**Sep 21, 2015 - Sep 26, 2015**	**Live Event**
SANS ICS Amsterdam 2015	**Amsterdam, NL**	**Sep 22, 2015 - Sep 28, 2015**	**Live Event**
SANS Bangalore 2015	**Bangalore, IN**	**Sep 28, 2015 - Oct 17, 2015**	**Live Event**
SANS DFIR Prague 2015	**Prague, CZ**	**Oct 05, 2015 - Oct 17, 2015**	**Live Event**
SANS Seattle 2015	**Seattle, WAUS**	**Oct 05, 2015 - Oct 10, 2015**	**Live Event**
SOS: SANS October Singapore 2015	**Singapore, SG**	**Oct 12, 2015 - Oct 24, 2015**	**Live Event**
SANS Tysons Corner 2015	**Tysons Corner, VAUS**	**Oct 12, 2015 - Oct 17, 2015**	**Live Event**
SANS Gulf Region 2015	**Dubai, AE**	**Oct 17, 2015 - Oct 29, 2015**	**Live Event**
SANS Cyber Defense San Diego 2015	**San Diego, CAUS**	**Oct 19, 2015 - Oct 24, 2015**	**Live Event**
SANS Tokyo Autumn 2015	**Tokyo, JP**	**Oct 19, 2015 - Oct 31, 2015**	**Live Event**
SANS South Florida 2015	**Fort Lauderdale, FLUS**	**Nov 09, 2015 - Nov 14, 2015**	**Live Event**
SANS Sydney 2015	**Sydney, AU**	**Nov 09, 2015 - Nov 21, 2015**	**Live Event**
SANS London 2015	**London, GB**	**Nov 14, 2015 - Nov 23, 2015**	**Live Event**
Pen Test Hackfest Summit & Training	**Alexandria, VAUS**	**Nov 16, 2015 - Nov 23, 2015**	**Live Event**
SANS Hyderabad 2015	**Hyderabad, IN**	**Nov 24, 2015 - Dec 04, 2015**	**Live Event**
SANS Chicago 2015	**OnlineILUS**	**Aug 30, 2015 - Sep 04, 2015**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**