Global Information Assurance Certification Paper

Windows Event Log – Centralized Logging Using MySQL and NTSyslog

Robert Miller
2004 - 02- 04
GSEC Practical Assignment v. 1.4b – Option 2

Abstract

In this case study the importance and impact of a centralized logging system for Windows hosts will be reviewed. The criteria for a logging system will be outlined and the flaws of the Windows logging environment and its impact to an organization will be reviewed. The solution to overcome the flaws of the existing logging system will be chosen based upon the provided criteria. Then the system architecture to be implemented will be assembled and analyzed. The steps to implement the chosen solution will be documented and explained. Finally the impact of the implemented centralized logging system will be reviewed.

Criteria for Logging Solution

The logging solution to be implemented would have to meet the following criteria:

1. 30-Day Log Retention – The solution must be able to retain a complete 30-days worth of log entries.
2. Log Integrity – The solution implemented must be trustworthy.
3. Zero Cost – Cost for the solution must be zero.
4. Security – The solution must be security conscious.
5. Fault Tolerance – The system must not have single point of failure.
6. Minimal System Resource Impact – The solution will be implemented on existing production hosts. Impact on the existing systems resources must extremely small.
7. High Data Availability – The solution selected must allow review of logs even if system is quarantined or down.
8. Audit of Multiple Hosts – The solution must benefit administrators by decreasing the administration effort of auditing multiple systems.
9. Automated Log Reports – The solution must be able to produce and deliver reports containing pertinent entries with complete automation.

Current Logging System

Windows Event Log - Overview

Windows Event Logs are extremely useful data for system administrators. These logs are the most common used when troubleshooting and auditing events in a Windows environment. Most software products written for the Windows operating system platform use the Event Log service for logging. However, despite its importance and widespread use, the Event Log framework has severe design flaws that lead to a lack of functionality, flexibility and provide little data integrity.

Event Log and Criteria items #1 and #2 - 30-Day Log Retention and Log Integrity

The solution must be able to guarantee criteria one and two, 30-day log retention and log integrity, respectively. The Event Log system has several flaws that allow for loss of log entries. With these flaws it is impossible to meet criteria one and two. Many of these flaws lie in the configuration options for the Event Log service. All the available configuration options listed below, along with the potential drawbacks:

- Overwrite events as needed.
  - Logs are overwritten if threshold size is ever reached.

- Overwrite events older than X number of days.
  - If log size threshold is met, and all entries within the log were created in the last X number of days, logging is halted. If this occurs all subsequent entries will be discarded in order to avoid overwriting the previous X number of days entries.

- Do not overwrite events.
  - If the log size threshold is met, all subsequent entries will be discarded.

All of the above listed drawbacks rely on the threshold size value to be met. A simple solution to avoid these drawbacks would be to increase the threshold size value. However it is quite an administrative undertaking to gauge what threshold size would be appropriate for each individual log on each host throughout an entire enterprise. You could gauge the amount of logging that is currently happening and adjust threshold levels accordingly, however this is not an ideal solution. If any system had software added/removed or the security auditing requirements were changed, then an increase/decrease in logging could occur. Also hosts logging mechanisms could be triggered more or less due to changes in the network environment. These situations would bring into question the size threshold value and it would have to be reviewed and adjusted accordingly for each log and host every time they were encountered.

With the decreased cost of storage space, many machines now have more than ample available data storage. One might think log size could be set to a larger amount and this would eliminate the risk of lost log entries. However there are limitations on the theoretical size the Windows Event Logs can grow to. The Event Log service runs under the services.exe process. Due to Windows architecture limitations, no one process may use over one gigabyte of memory. Since all event files are loaded into memory when the Event Log service starts, the combined size of all Event Log files could never exceed the one-gigabyte limitation. If the memory pool reached this limitation, logging would cease. These drawbacks make it impossible for the Event Log service to meet the criteria of

30-day log retention and data integrity. These issues illustrated by Microsoft are in chapter six of "*Threats and Countermeasure Guide*" entitled "*Event Log*". (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/topics/hardsys/tcg/tcgch06.asp)

Event Log and Criteria Item #3 – Zero Cost

The Event Log service is included with the cost of a Windows operating system software license. Therefore no added cost would be necessary to use it for a solution. The Event Log system has no cost and therefore satisfies criteria item three.

Event Log and Criteria Item #4 – Security

The security of the logging solution is extremely important. The Event Log service runs under the services.exe process as the local system user with several other services. Even though the Event Log service shares a memory pool with other services, all other services also run as the system user. If any of the services running under services.exe were to be exploited then the attacker would have system level access. This architecture is very sound. System level access is required to stop the service or alter the contents of its data in memory. All Data on any host that has been compromised at a system user level is suspect, this simply cannot be avoided.

The Event Log files are stored with permissions allowing only administrators and the local system user the ability to access them. Again an attacker would have to obtain system/administrator level access to alter contents of the logs. These are possible threats, but since all data on a host is suspect if this level of unauthorized access was gained, it cannot be avoided. There have not been known exploits for the Event Log service, however there have however been exploits for the common method of viewing Event Log entries. The Event Viewer MMC snap-in for Windows 2000 systems is the tool provided for viewing the event files. The unchecked buffer would allow an attacker to execute code of their choice if it was injected into an Event Log entry in a malformed manner. But again, the risk for this too is low. An attacker would have to be able to insert malformed entries into the log file and then wait for the Event Viewer MMC snap-in to view that log. (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-013.asp)The Event Log framework does provide a secure environment for logging.

Event Log and Criteria Item #5 – Fault Tolerance

The logging solution must provide a fault tolerant logging environment. The storage location for Event Logs provides only a single point of physical data storage. Since all Event Logs reside on the hosts locally, and the Event Log service has no method for remote transport of log entries between hosts, the Event Log system is not fault tolerant.

Event Log and Criteria Item #6 – Minimal System Resource Impact

The Event Log service uses memory-mapped files. This means when the Event Log service is running, each log is loaded into memory in its entirety. If the threshold size value increased, which would be necessary to guarantee 30-day log retention, then available system resources would be decreased. Since the solution must not impact system resources, the Event Log service does not meet the criteria.

Event Log and Criteria Item #7 – High Data Availability

The ability to be able to react quickly and well informed after an incident has taken place is key. If a host compromised, it could be powered off to avoid further damage and stop the attack. If the incident caused a high risk of malicious code being executed at boot, administrators would have to wait for the disks to be removed and mounted in another host to review log entries. Alternatively, a host could have a hardware or software issue severe enough to stop it from booting. If this occurs then logs cannot be analyzed quickly to help determine what lead to the failure. The Event Log service does not provide high data availability due to single physical storage location.

Event Log and Criteria Item #8 – Audit Multiple Hosts

The Windows Event Log framework does not lend its self to auditing multiple hosts. In any Active Directory setup, there should be at least two or more domain controllers. If the Active Directory architecture has multiple sites, you would have two or more domain controllers for each site. Now if the given scenario requires review of Active Directory events then logs from all the hosts that are domain controllers in your organization would have to be analyzed. This example just covers reviewing Active Directory events, such as domain user logon attempts. If there was a need to track a users access to all member servers in an Active Directory environment, it could mean analyzing logs from several hundred hosts. Since no tools are provided for viewing log entries from multiple hosts simultaneously and with ease, this requirement criteria is not met by Event Log.

Event Log and Criteria Item #9 – Automated Log Reports

Reports generated from Event Log data are very useful documents to system administrators. The provided Event Log tools do provide means of creating reports through the export functionality of the Event Viewer MMC snap-in. However this cannot be automated solely through the Event Viewer. The types of output formats are limited to tabular and comma delimited. The ability to filter the output of these reports to only include pertinent entries would further increase their value. Unfortunately the filtering options provided by the Event

Viewer are extremely weak. You cannot make exceptions to the filtering criteria. For example, if you wanted to see all log entries except ones created by the administrative user, you could not. The provided filtering options also do not support pattern matching which is a very useful tool for extracting pertinent entries. Event Log and its tools do not provide a means for automated reports containing only pertinent data to be generated.

Windows Event Log - Conclusion

In conclusion, the Event Log service cannot be solely relied upon for a solution. It would not provide 30-day retention, integrity, fault tolerance, minimal system resource impact, high data availability, ease of multiple host auditing, or automated reporting of pertinent events. It does provide a zero cost and secure solution, however this does not meet all the required criteria. Additional entities must be added to the logging system to create a solution that meets or exceed the criteria.

Logging Solution

Solution Architecture

After review of the drawbacks of the existing logging system, the need for improvements is evident. An architecture that would help overcome the outlined flaws must be assembled. The best architecture to defeat some of the flaws would be a centralized client/server system with real time log forwarding. The use of multiple servers would also be necessary. This type of architecture will help meet the listed items of criteria and decrease overall risk.

Solution Hardware

The selection of hardware is highly limited by criteria item two, zero cost. With no money available for hardware, existing unused hardware had to be reviewed to see if it would meet the standards for this solution. Many desktops that had been cycled out of use to storage were available. The best systems available had the following specs:

- Processor - PIII 800 Mega-Hertz
- Memory – 128 Mega-Bytes
  - Other systems cycled to storage used identical memory. Memory from those hosts was used, this increased system memory to 384MB.
- Hard Drive – 40 Giga-Bytes
- Network Card – 100 Mega-Bit/Second

Systems with these specs would have to be used for the hardware requirements of the centralized logging system. Two machines would be used since the centralized system called for multiple servers to provide fault tolerance.

Solution Software Components

Operating System

Operating system choices were also limited to zero dollar cost, again this was provided by criteria item two. The most popular choices left were Linux, BSD, and other variant operating systems. Out of these choices Mandrake Linux (http://www.mandrakelinux.com/en/9.2/features/) was chosen for the operating system platform. The ease of administration that comes from the MandrakeUpdate utility (http://www.linux-mandrake.com/en/demos/Spotlight/SoftwareMgr/pages/rpmdrake14.php3) that has been integrated into Mandrake's Software Manager makes patching systems quite easy. Also the MandrakeUpdate utility is run during installation time to ensure that a host is never brought onto a network with known vulnerable or unpatched versions of software. Also the MSec utility (http://www.mandrakeuser.org/docs/mdoc/ref/prog-msec.html) eases administration effort by making it easy to adjusting security settings and generate automated email security reports. Since current employee knowledge of the OS was high, it was the best choice.

Logging Services

The type of logging service to be run on the server must be selected. The default method of logging for Mandrake Linux is Sysklogd (http://www.infodrom.org/projects/sysklogd/). Sysklogd is comprised of two individual daemons, Syslogd and Klogd. The Syslogd daemon supports logging from remote hosts and is based on the BSD Syslog daemon. Sysklogd adheres to the BSD syslog protocol defined in RFC 3164 (http://www.faqs.org/rfcs/rfc3164.html). With the widespread use of this type of logging protocol and its preexisting nature in the selected platform operating system, it was chosen as the logging daemon to be used for the server.

The logging service selection for the client would remain to be the Windows Event Log service. If a real-time remote transport method for log entries is used, then the burden of meeting criteria is shifted to the centralized system instead of the client.

Remote Transport and Conversion

A method for transporting and converting logs from clients to the centralized server must be selected. There is a lack of common services for transportation of files between hosts in default installs of Microsoft Windows and bare bones installs of Mandrake Linux. Installation of new services or client software would be necessary. Windows hosts could export the logs and upload

them to the centralized location by means of FTP, SMB, SSH or other protocols. However if these means of transport were selected, log file copies would be incremented and manual intervention would be required to complete this tasks. If to many log entries occurred between the incremented file copies, then log entries could be overwritten or not recorded. A near or real-time conversion and transport method from client to server must be used. The software that meets these needs is application called NTSyslog (http://ntsyslog.sourceforge.net/). This software would take all Event Log entries, convert them to syslog format, and transport them to the centralized location over the network using the User Datagram Protocol (UDP).

Data Storage

With all logs in a central location in a flat text file, a better method of data storage was necessary. A database would yield advantages over flat text data storage. Indexed columns for faster searching along with the power of the SQL query language for data extraction. MySQL (http://www.mysql.com/products/mysql/index.html) has these capabilities along with a replication feature and a free Windows GUI administrative utility called MySQL Control Center (http://www.mysql.com/products/mysqlcc/index.html). MySQL would be used as the method of data storage due to these advantages.

Using a database for data storage would require a mechanism for parsing flat text files and importing them into the database. Perl (http://www.perl.org/about.html) was the perfect language choice for creating such a mechanism. Already being installed by default under Mandrake and having preexisting modules for accessing databases made it the ideal solution for this task. Since Perl has support for regular expressions, pattern matching could be used to parse the flat text file. Then the DBI Perl module (http://dbi.perl.org/about/) would be used as the means of accessing the MySQL database when importing records to the database.

Data Extraction and Delivery Methods

The Perl DBI module provides an ample way to access information in the MySQL database from a Perl script, but this provides no method of remote data delivery. The ability to deliver data to remote hosts would increase the systems value to system administrators. The ability to deliver the data would have to be on an automated request basis to fulfill criteria requirement item nine. Additionally the ability to extract data remotely on a per request basis would be a useful tool for system administrators.

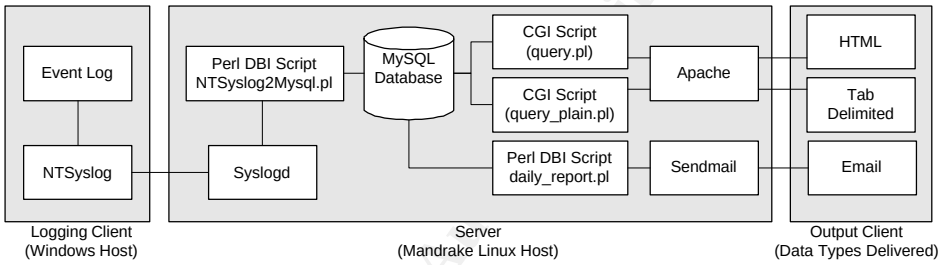
A Perl script could be written to extract data from the database use the DBI module, then the data could be piped to the Sendmail client utility and delivered to a recipients mailbox by means of SMTP. If this script were executed at timed intervals by use of Cron (http://www.mandrakeuser.org/docs/admin/acron2.html), it would

provide complete automation of data extraction and delivery. This would fulfill the criteria for automated reporting.

The means to extract data remotely on a per request basis would prove a useful tool. If a simple web front end were created to the database, then this would be possible. Apache would be used to provide the HTTP services. Using Apache's CGI functionality a script, written in Perl using the DBI module for database access, can be executed remotely via HTTP that would extract data and return it to the HTTP client. A variety of data outputs, such as HTML or delimited formats, could be provided.

Solution Architecture and Component Review

Below is a graphical representation of the solution to be implemented. This shows the data flow of an Event Log entry from creation to output format:



Now a review of the provided criteria must be completed to ensure that this solution will provide necessary and desired benefits.

Criteria Item #1 – 30-Day Log Retention

A centralized system would make it easier to guarantee 30-day log retention. Since all logs are centralized, the question of "What size thresholds for each log per each host is correct?" has been removed. Now the question is "How much storage space is necessary for all logs?", this question is easier to answer and monitor for changes. With all logs in the centralized location, system administrators only have to ensure ample space is remaining on the host acting as the syslog server.

Criteria Item #2 – Log Integrity

The use of a centralized logging system also increases log integrity. Most issues with the Event Log service decreasing log integrity stems again from log size threshold selection for each log per host. This risk will be reduced since the dedicated centralized system will have more data storage resources.

Since log data will travel across a network to the centralized logging location, there must be a way to guarantee that log integrity will not be lost during

transmission. The logs are transmitted across the network using UDP. UDP is a layer four protocol that is part of the layer three IP protocol suite. These protocols use the functionality of the layer two Ethernet protocol for guaranteed delivery and integrity. Ethernet breaks the packets from UDP/IP into frames and uses CRC's to detect data errors during frame transmission. The contents of the Ethernet frame is fingerprinted and tagged with the CRC before delivery. Upon reception of the Ethernet frame packet, the contents are fingerprinted again. Then the CRC's are then compared, if the CRC's do not match, the frame is discarded and a retransmit request is sent. Ethernet also automatically retransmits lost frames by waiting for responses that frames were transmitted correctly. If this response is not received, the frame is then retransmitted. This is illustrated in Gorry Fairhurst's "*Cyclic Redundancy Check*" (http://www.erg.abdn.ac.uk/users/gorry/course/dl-pages/crc.html).

Since the log entries will now be transported over the network, they run the risk of being tampered with while in transit or being spoofed entirely. If an attacker was able to perform a "man in the middle" attack (http://www.watchguard.com/infocenter/editorial/135324.asp) they could alter the contents of the UDP packet containing the log entry while in transit. If an attacker had the ability to send packets to the syslog server, they could spoof the source of the packet along with the entire log entry (http://www.insecure.org/sploits/aix.generic.syslogd.problem.html). Since these vulnerabilities do exist this solution does not provide total log integrity. However, since this system better guarantees that all entries are recorded, administrators have the ability to review and discern which are real entries and which are spoofed, rather than have no entries to review.

Criteria Item #3 – Zero Cost

Though several additional software components have been selected, they are all no cost solutions. The software components, Apache, Event Log, Mandrake Linux, MySQL, NTSyslog, Perl, Perl DBI, Sysklogd, and Sendmail are all available for free. Hardware costs will also remain at zero since the logging solution will use hardware that had been cycled to storage.

Criteria Item #4 – Security

The outlined logging solution can be implemented securely, provided configurations and permissions are done correctly. Here are a list of the services that must be added and what must be done to secure them.

- Apache
  - On Mandrake Linux, Apache already runs as an unprivileged user on a default install. Clients will be able to remotely execute scripts and access log data through Apache, a password should be added to protect the use of system resources. The passwords would go across the network in plain text by default, HTTP with SSL

encryption, known as HTTPS, should be used to ensure the password cannot be sniffed from the network.

- MySQL
  - By default no password is set for the root@localhost account, one will be added. In addition low privilege users for the MySQL database will be created and used when data mush be accessed or added. The user readwrite@localhost, will have only read and write access to all tables in the syslog database. The readonly@localhost user will have only read access to all tables in the syslog database.
- NTSyslog
  - This service is not remotely accessible and runs with system user level access. A user would have to have system level access on the local host in order to stop the service or access its memory pool.
- Sysklogd
  - This service is remotely accessible and runs as the root user. This service is very mature however and has had only had one exploit since its inception according to CERT (http://search.cert.org/query.html?col=certadv&col=vulnotes&qt=sysklogd&charset=iso-8859-1). The threat exists, but no known vulnerability. Since Risk = Threat x Vulnerability, the security risk still remains low.
- Sendmail
  - Sendmail will not be run as a service, it is only installed so the client tools and libraries are available as a means of sending emails.

Criteria Item #5 – Fault Tolerance

The use of a centralized logging system would increase fault tolerance. The most recent log entries would be located on the client and server simultaneously. The outlined solution uses the NTSyslog clients ability to log to multiple syslog hosts simultaneously and multiple syslog servers to provide fault tolerance.

Criteria Item #6 – Minimal System Resource Impact

If a centralized logging solution was implemented it would decrease use of system resources on client systems. Since all logs would be moved to the central location, the amount of storage space necessary on the clients would be decreased. Also due Event Log's use of memory-mapped files, system memory use would be decreased.

Criteria Item #7 – High Data Availability

Data availability is heightened substantially by using a centralized logging system with real-time event transport. If a host were not available for any reason,

the log data would still be accessible on the centralized log server. Since all logs would be transported in real-time to the multiple log servers, data availability is very high even if a centralized server is down.

Criteria Item #8 – Audit of Multiple Hosts

A centralized solution allows logs from multiple hosts will be in one physical data storage location. This solution would decrease the amount of effort necessary for auditing multiple systems simultaneously.

Criteria Item #9 – Automated Log Reports

The solution outlined would provide automated generation and delivery of log reports. Using a Perl script data can be extracted and filtered as needed, then transported using the Sendmail client to administrators mailboxes.

Solution Architecture and Component Review - Conclusion

The outlined solution would meet all criteria listed, and provide useful tools to system administrators. This solution provides 30-day retention, integrity, zero cost, security, fault tolerance, minimal system resource impact, high data availability, multi-host auditing and automated reporting. In addition it has multiple formats for extracted data and provides a method to extract information remotely on a per request basis.

Server Setup

Installation

The Mandrake Linux host was setup the following way during installation:

- Language – Select your language (English American)
- License Agreement – Accept the license agreement to continue.
- Select Mouse Type – Select your mouse type.
- Install Type – Select option to do fresh install and delete all existing partitions.
- Security Level – Select security level.
    - Level High - is the highest level of security that keeps the machine at a useable level.
    - Set security administrator login/email to the location you want security email alerts from Mandrake to be sent to.
- Disk Partitioning – Erase all existing disk partitions
    - This host had a 40GB hard drive, the /var slice was resized to 30GB to give ample space for the MySQL database Syslogd logs which reside on this slice by default.

- Package Group Selection – All options unchecked, select “Individual package selection”
    - Mandrake will prompt you because you have selected no packages, it will give you three options, select “With X” and “With basic documentation”.
- Individual Package Selection – Select the following packages and their necessary dependencies when prompted.
    - Server -> Web/FTP
        - apache2
        - apache2-mod_ssl
    - Server -> Database
        - MySQL
        - perl-Mysql
    - Server -> Network Computer Server
        - openssh-server
- Individual Package Selection – Unselect the following packages and dependencies.
    - Server -> Mail
        - Postfix
- Click the “recycle” button at the bottom of the screen to see a full list of packages available in alphabetical order, select the following packages and their dependencies:
    - bind-utils
    - perl-Date-Calc
    - sendmail
    - wget
- You will be prompted that MySQL and openssh-server will be activated by default, select yes and click next.
- Set your root password
- Add an administrative user and set their password (required)
    - For this user, check all options on the screen to give that user access necessary to administer the system.
- Install the bootloader on the MBR of the hard drive.
- Review the summary screen to ensure correct configuration options.
- Install applicable updates. Select an appropriate mirror. The applicable updates will be checked automatically if you system has an older package installed.
- Reboot and login as administrative user
    - Execute below command from shell and supply root password
        - su -
- Upgrade Kernel
    - A local vulnerability for Mandrake 9.2 in the kernel was discovered. This update can’t be applied through the update manager. You must download the update from an update mirror and apply it manually. Please visit the below URL for instructions on how to upgrade the kernel:

(http://www.mandrakesecure.net/en/kernelupdate.php)

- Please note that the secured kernel was used in installation for this case study. To keep inline with the setup, the secure version of the kernel should be downloaded and installed.

Post Install Configuration

- Syslog
  - Edit the /etc/sysconfig/syslog file to enable remote logging support for syslog.
    - Change line 6 from
      SYSLOGD_OPTIONS="-m 0"
      to
      SYSLOGD_OPTIONS="-r -m 0"
- TCP Wrappers
  - Edit the /etc/hosts.allow file to enable remote hosts to connect to services that use TCP Wrappers (such as OpenSSH).
    - Add the below line to the end of the file
      ALL : ALL : ALLOW
      - This is over simplified and allows any host to connect to services using TCP Wrappers. Please review and configure the TCP Wrappers (http://www.mandrakeuser.org/docs/mdoc/server/network-security.html) for your scenario.
- Sendmail
  - Edit the /etc/mail/submit.cf file to use the mail exchange for your network.
    - Change line number 118 from
      D{MTAHost}[localhost]
      to
      D{MTAHost}[YourMailExchangeHostName]
- Runlevel
  - Edit /etc/inittab to change default runlevel
    - Change line number 18 from
      id:5:initdefault:
      to
      id:3:initdefault:
- Services on Startup
  - Execute the following commands to remove services on startup

```
rm –f /etc/rc3.d/S11portmap
rm –f /etc/rc3.d/S17alsa
rm –f /etc/rc3.d/S18sound
rm –f /etc/rc3.d/S20xfs
rm –f /etc/rc3.d/S25netfs
rm –f /etc/rc3.d/S56rawdevices
```

  - Execute the following commands to add services to start on boot

```
ln –s /etc/init.d/mysql /etc/rc3.d/S40mysql
```

- Reboot the system by executing the “reboot” command from the shell

- Login as administrative user and “su –“ and enter the root password
- MySQL
    - Execute the following commands to setup database, table, and users

```
mysql
set password for root@localhost = password('yourpassword');
create database syslog;
use syslog;
\! wget -O /tmp/mysql_create_syslog www.variate.net/sans_gsec_practical/mysql_create_syslog.txt
source /tmp/mysql_create_syslog;
\! rm –f /tmp/mysql_create_syslog
\! wget –O /tmp/mysql_create_tmp_host www.variate.net/sans_gsec_practical/mysql_create_tmp_host.txt
source /tmp/mysql_create_tmp_host
\! rm –f /tmp/mysql_create_tmp_host
grant select on syslog.* to readonly@localhost;
grant select, insert on syslog.* to readwrite@localhost identified by 'yourpassword';
flush privileges;
quit
```

Below are the contents of the mysql_create_syslog.txt file:

```
CREATE TABLE `syslog` (
  `Date` date NOT NULL default '0000-00-00',
  `Time` time NOT NULL default '00:00:00',
  `Host` varchar(50) NOT NULL default '',
  `Notifier` varchar(20) NOT NULL default '',
  `ErrorType` varchar(20) NOT NULL default '',
  `EventID` mediumint(9) NOT NULL default '0',
  `User` varchar(35) NOT NULL default '',
  `ErrorText` text NOT NULL,
  KEY `Time` (`Time`),
  KEY `Date` (`Date`),
  KEY `ErrorType` (`ErrorType`),
  KEY `EventID` (`EventID`),
  KEY `Host` (`Host`),
  KEY `Notifier` (`Notifier`),
  KEY `User` (`User`)
) TYPE=MyISAM ROW_FORMAT=DYNAMIC;
```

Below are the contents of the mysql_create_tmp_host.txt file:

```
CREATE TABLE `tmp_host` (
  `Host` varchar(20) NOT NULL default ''
) TYPE=MyISAM;
```

- Cron Jobs
    - These Cron jobs will provided needed automation, execute the below commands from the shell.

```
mkdir /root/scripts
wget –O /root/scripts/NTSyslog2MySQL.pl www.variate.net/sans_gsec_practical/NTSyslog2MySQL.pl
wget –O /root/scripts/report_general_events.pl www.variate.net/sans_gsec_practical/report_general_events.pl
wget –O /root/scripts/report_security_events.pl www.variate.net/sans_gsec_practical/r eport_security_events.pl
wget –O /root/scripts/update_tmp_host.pl www.variate.net/sans_gsec_practical/update_tmp_host.pl
mkdir /etc/cron.syslog
echo –e '#!/bin/bash\n/usr/bin/perl /root/scripts/NTSyslog2MySQL.pl\n' > /etc/cron.syslog/convert_syslog_mysql
echo –e '#!/bin/bash\n/usr/bin/perl /root/scripts/ report_general_events.pl\n' > /etc/cron.syslog/report_general_events.pl
echo –e '#!/bin/bash\n/usr/bin/perl /root/scripts/report_security_events.pl\n' > /etc/cron.syslog/report_security_events
echo –e '#!/bin/bash\n/usr/bin/perl /root/scripts/update_tmp_host.pl\n' > /etc/cron.hourly/update_tmp_host
echo –e '02 2 * * * root nice –n 19 run-parts /etc/cron.syslog' >> /etc/crontab
chmod –R 0750 /root/scripts
chmod –R 0750 /etc/cron.syslog
chmod –R 0750 /etc/cron.daily
/etc/init.d/crond restart
```

    - You must also edit the /root/scripts/NTSyslog2MySQL.pl script to suit the configuration of the host you will run it on. The value for

$SysLogHost on line 10 should be set to the hostname of the localhost. On line 13, you should replace 'yourpassword' with the password for the readwrite@localhost MySQL user.
    - The variables on lines 21-23 should be configured for report_general_events.pl and report_security_events.pl.
    - Lines 15 and 26 of update_tmp_host.pl should be updated to include the password for the root@localhost MySQL user.
    - The full text of these scripts are provided at the end of this case study.

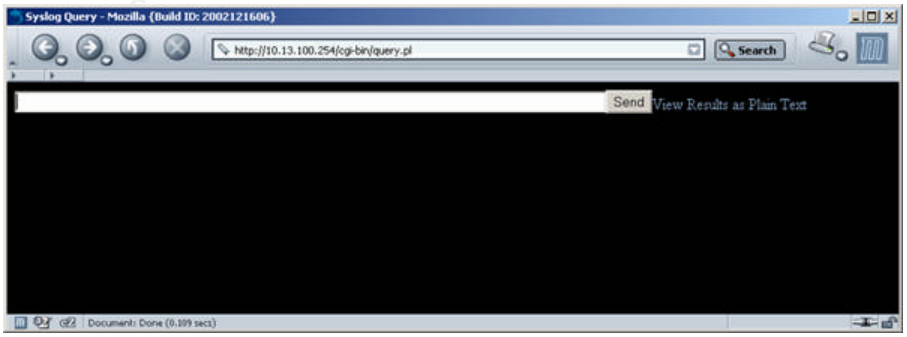
- Apache2_mod-ssl
    - Execute the below commands from the shell

```
wget -O /var/www/cgi-bin/query.pl www.variate.net/sans_gsec_practical/query.pl
wget -O /var/www/cgi-bin/query_plain.pl www.variate.net/sans_gsec_practical/query_plain.pl
chmod –R 0755 /var/www/cgi-bin/
echo –e '<VirtualHost *:80>\nServerName *FullyQualifiedHostName*\nRedirect /
https://*FullyQualifiedHostName*/cgi-bin/query.pl\n</VirtualHost>\n' >> /etc/httpd/conf/vhosts/Vhosts.conf
```

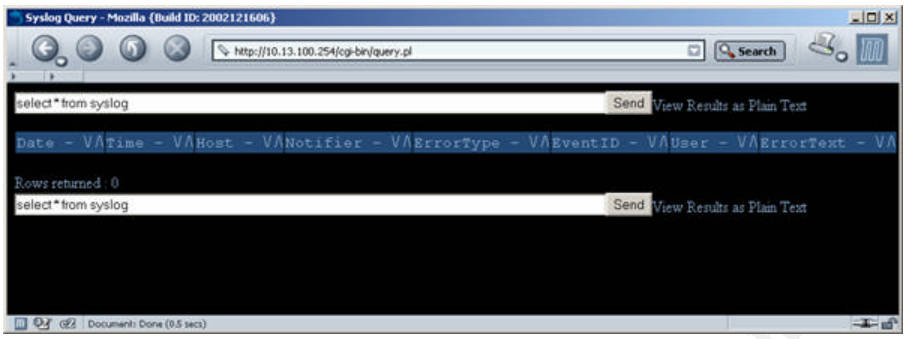
    - Line 10 of /var/www/cgi-bin/query.pl must have the value of $HostFQDN set to the fully qualified domain name (or IP address) of the machine on which it resides.
    - Full text for these scripts is included at the end of this case study.
    - Setup password authentication
        - Execute "htpasswd –c /var/www/cgi-bin/.htpasswd admin" from the shell, the provide a password.
        - Create a file called /var/www/cgi-bin/.htaccess with the following contents

```
AuthType Basic
AuthName "Password Required"
AuthUserFile /var/www/cgi-bin/.htpasswd
Require valid-user
```

To test that the CGI script is accessible open a web browser and visit http://*hostname* and the screen should prompt you about a security certificate, accept it. Then you will be prompted for a username and password, enter the admin username and password you just created, then the screen should look like this:



Enter a SQL query into the field, to see all records currently in the database enter "select * from syslog" and click send. An empty set return (zero rows) should be seen in the web browser.



Client Setup

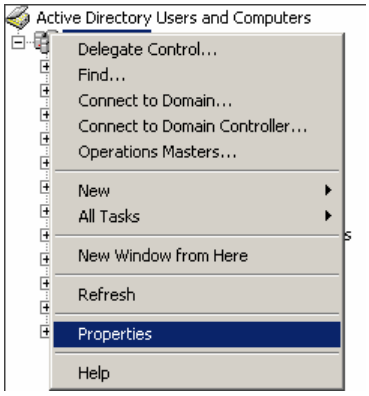
Installation

Logon to the Windows host and complete the following steps to install the necessary client software.

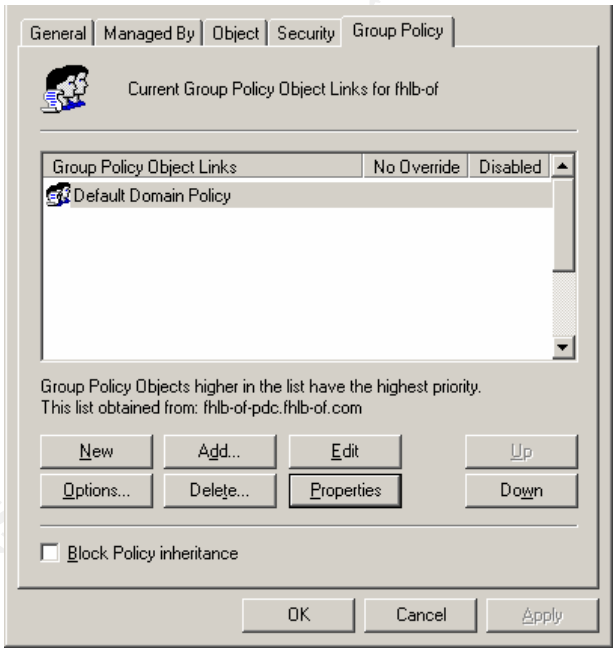
- NTSyslog – Install mechanism that transports eventlog entries to centralized server by means of the syslog protocol.
  - Download the software from one of the mirrors listed
    http://prdownloads.sourceforge.net/ntsyslog/ntsyslog-1.13.zip?download
  - Uncompress the saved archive and copy ntsyslog.exe and ntsyslogctrl.exe to your system root (usually C:\winnt)
  - Open a command window (Start -> Run -> cmd) and execute the below command
    ntsyslog –install

Configuration

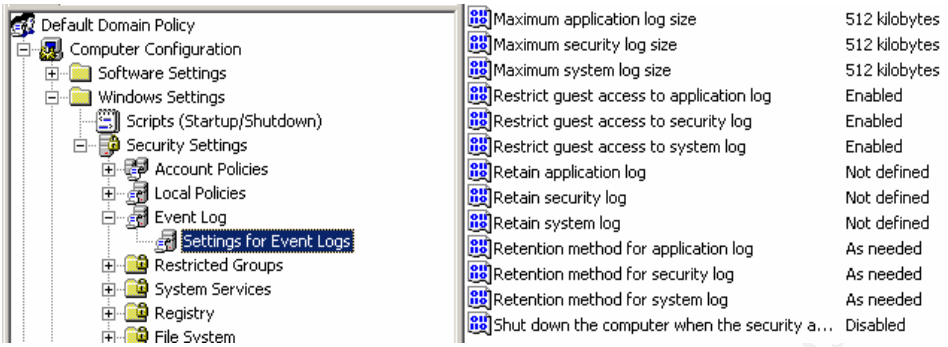
- EventLog / Security Auditing – Configure Multiple Hosts
  - You can configure the EventLog settings across multiple hosts in an Active Directory environment quite easily by using system policies. To access the domain policies, open Active Directory Users and Computers (provided with the Admin Pack installation from the Windows 2000 Server CD), right click the root domain object, and select “Properties”.



- Click the “Domain Policy” tab, then from the list of policies select “Default Domain Policy” and click the “Edit” button
    - This can be done to configure the behavior of all hosts that are a member of the Active Directories EventLog settings, alternative you can create a policy that is only applied to the hosts you wish to use for the centralized logging.

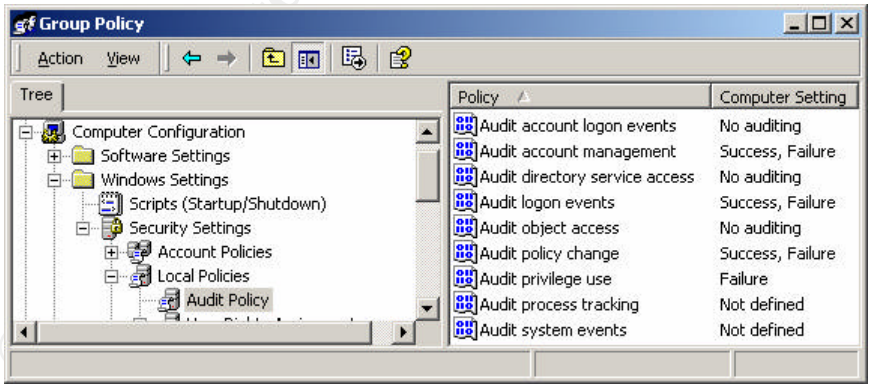


- Drill down to the following location Computer Configuration -> Windows Settings -> Security Settings -> Event Log -> Settings for Event Logs. Then configure the various options to suit your organizations needs.



- As a baseline, log retention method for all logs should be to overwrite “As needed” and the amount of days to retain logs for should not be defined. Even with this policy implemented, it is necessary to check your hosts Event Log configuration to ensure the policy updated correctly. Also this policy only allows you to set retention methods for application, security, and system Event Logs. There are several other types on a default domain controller alone, such as DNS, File Replication service etc. Therefore, all hosts that are to be logged should be double checked using the steps outlined in the next section “EventLog – Configure Standalone/Single Server”

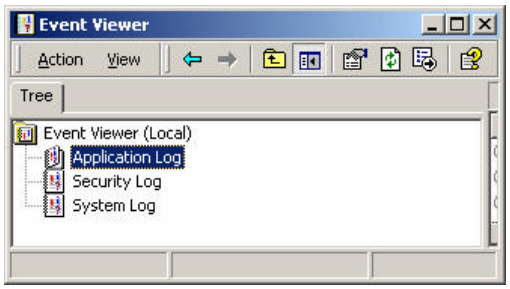
- By default, Windows 2000 does not audit security events. To configure it to audit security settings drill down to the following location in the current policy. Computer Configuration -> Windows Settings -> Local Policies -> Audit Policy. Then configure with the below settings.



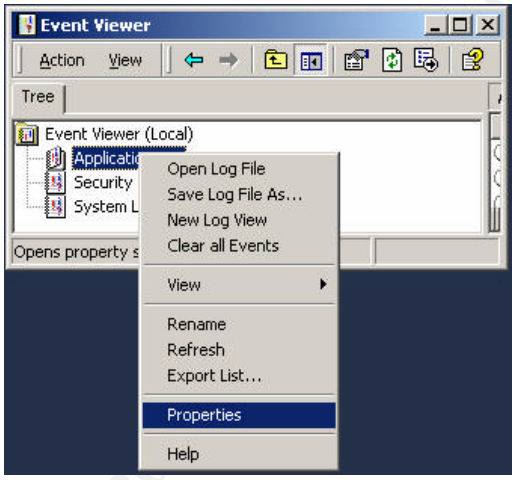
- The above configuration provides a good baseline, however the requirements for your auditing may differ. Please review the article at the link below to determine what configuration suits the criteria for your environment. (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnexnt00/html/ewn0054.asp)

- EventLog – Configure Standalone/Single Server

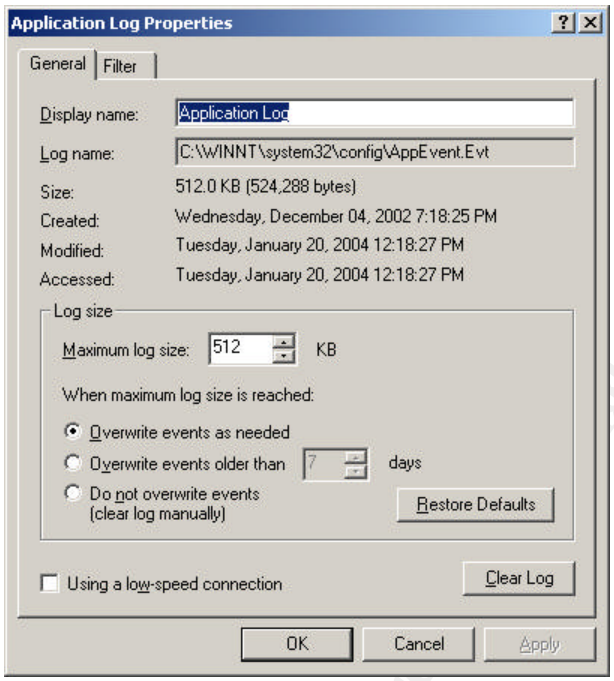
- Click Start -> Run and enter “eventvwr” and click OK to bring up this screen



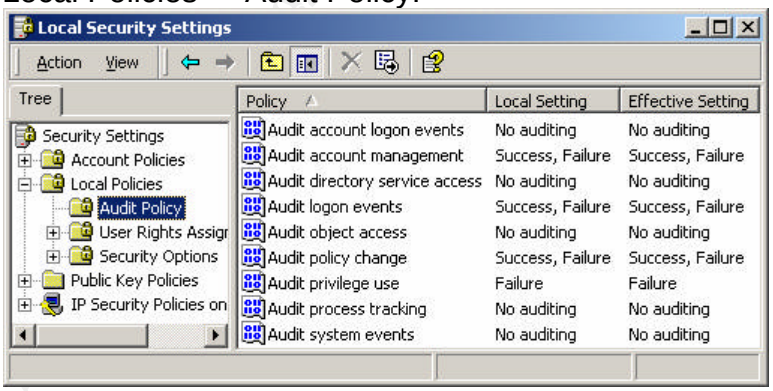
- Right click on the first EventLog listed and select properties



- Change the radio button selection from “Overwrite events older than 7 days” to “Overwrite events as needed” and click “OK”

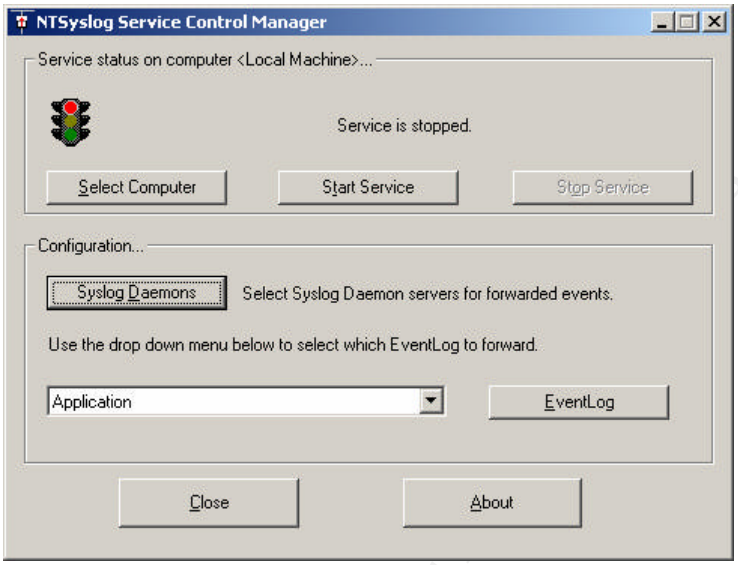


- Security Auditing – Configure Standalone/Individual Server
  - Drill down to the following location to configure the local security policy. Start -> Settings -> Control Panel -> Administrative Tools -> Local Security Policy. Once the Local Security Settings snap-in has opened, drill down to the following location. Security Settings -> Local Policies -> Audit Policy.

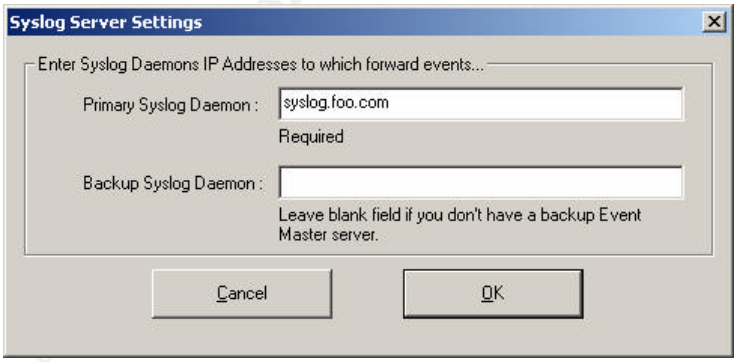


    - As you configure the options the “Effective Setting” column will not update automatically. Close and re-open the Local Security Settings snap-in after configuring to refresh the “Effective Setting” column
    - Again, the above configuration is a good baseline but may not meet the requirements for all given situations. Documentation of audit settings are available from Microsoft. (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnexnt00/html/ewn0054.asp)

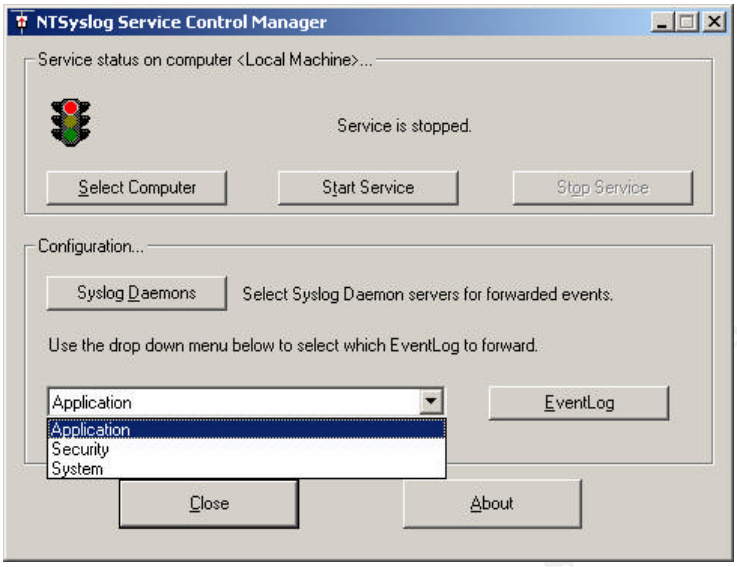
- NTSyslog – Configure Services
  - Open the NTSyslog Control utility (%SystemRoot%\NTSyslogCtrl.exe)



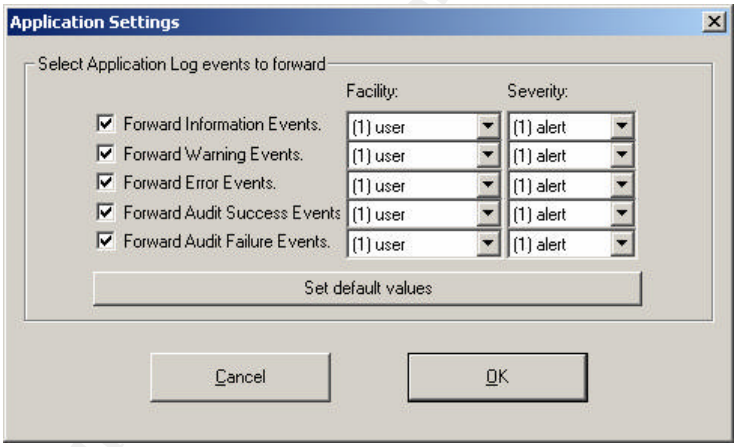
  - Click the “Syslog Daemons” button to bring up this window, enter the fully qualified domain name of your Syslogd host(s), click OK.



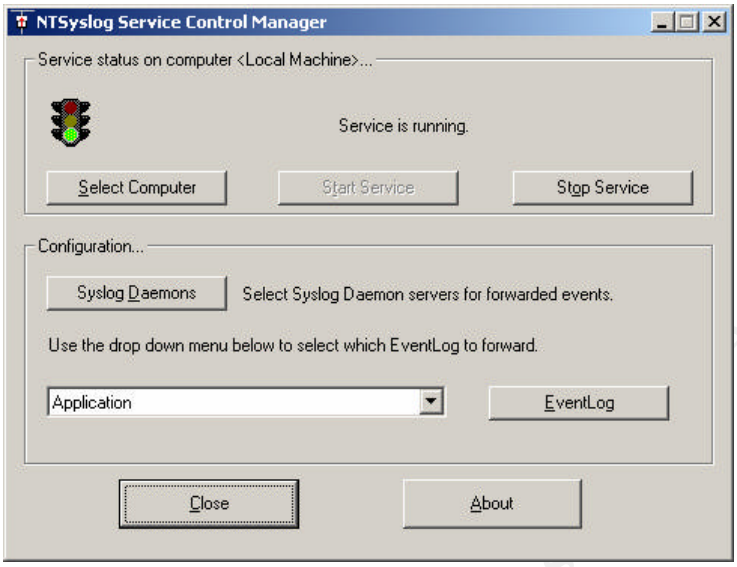
  - Selecting the drop down menu will show the event logs on the host. Select one from the list and click the EventLog button on the right.



- Make sure all options are checked so that all types of events are forwarded to the syslog server, click ok. Repeat this step for all event logs that are on the host.



- Click “Start Service” to begin forwarding your logs



Appendix

The appendix contains the full Perl code for all scripts refrenced in this case study in alphabetical order.

NTSyslog2MySQL.pl

```
#!/usr/bin/perl -T
#
# This script adds the previous days NTSyslog to a MySQL database
use strict;
use Date::Calc qw(:all);
use DBI;
use FileHandle;
#
# Configure and open database/files
my $SysLogHost = "syslog";                              # Set this to your machines hostname
my $SysLog = "/var/log/syslog";
open (SYSLOG,"<$SysLog") or die "Could not open $SysLog : $! \n";
my $DBH = DBI->connect('DBI:mysql:syslog', 'readwrite', 'yourpassword', { RaiseError => 1, AutoCommit => 1});
#
# Compute yesterdays date
my ($Year, $Month, $Day) = Today();
($Year, $Month, $Day) = Add_Delta_Days($Year, $Month, $Day, -1);
my $Month_Text = Month_to_Text($Month);
my $Day_Text = $Day;
while (length($Month_Text) > 3) {
        chop($Month_Text);
}
while (length($Day_Text) < 2) {
        $Day_Text = " ".$Day_Text;
}
my $Date      = "$Year-$Month-$Day";
my $Date_Text = "$Month_Text $Day_Text";
while (my $Line = <SYSLOG>) {
        my ($Time,$HostName,$Notifier,$EventType,$EventID,$User,$ErrorText) = FormatLog($Line,$Date_Text);
        InsertRecord($Date,$Time,$HostName,$Notifier,$EventType,$EventID,$User,$ErrorText,$DBH);
}
close(SYSLOG);
$DBH->disconnect();
exit(0);
```

```
sub FormatLog {
	my $Line = $_[0];
	my $Date_Text = $_[1];
	if ($Line !~ /^$Date_Text/) {
		next;
	}
	if ($Line =~ /$Date_Text...:..:...$SysLogHost/) {
		next;
	}
	if ($Line =~ /last.message.repeated/) {
		next;
	}
	#
	# Global RegEx replacements
	#
	$Line =~ s/\'/\'\'/g;
	$Line =~ s/\\/\\\\$1/g;
	$Line =~ s/NT.AUTHORITY/NT-AUTHORITY/g;
	$Line =~ s/\$/\\\$/g;
	#
	# Parse line and return values
	#
	$Line =~ m/(\w+?)\s+(\d+?)\s(\d\d:\d\d:\d\d)\s(.+?)\s(.+\])\s(\d+?)\s(.+?)\s+?(.+)/;
	#
	# Convert variables to human readable and writeable values
	#
	my $Month	= $1;
	my $Day		= $2;
	my $Time	= $3;
	my $HostName	= $4;
	my $Notifier_EventType = $5;
	my $EventID	= $6;
	my $User	= $7;
	my $ErrorText	= $8;
	#
	my @split = split(/['[']/, "$Notifier_EventType");
	chop $split[1];
	my $Notifier	= $split[0];
	my $EventType	= $split[1];
	if ( $User !~ /\\/ ) {
		$ErrorText = "$User " . $ErrorText;
		$User = '';
	}
	$User =~ s/NT-AUTHORITY/NT AUTHORITY/g;
	$ErrorText =~ s/^ //g;
	return($Time,$HostName,$Notifier,$EventType,$EventID,$User,$ErrorText);
}

sub InsertRecord {
	my $SQL = qq{INSERT INTO syslog (Date, Time, Host, Notifier, ErrorType, EventID, User, ErrorText) VALUES
('$_[0]','$_[1]','$_[2]','$_[3]','$_[4]','$_[5]','$_[6]','$_[7]')};
	my $STH = $_[8]->prepare($SQL);
	$STH->execute();
	$STH->finish();
}
```

query.pl

```
#!/usr/bin/perl -wT
#
#
use strict;
use CGI;
use DBI;
use URI::Escape;
#
# config
my $HostFQDN = "hostname.domainname.topleveldomainname";   # example = "syslog.variate.net" or IP address can be
used
```

```
#
# CGI
my($query) = CGI->new();
my($SQL) = $query->param("query");
print   "Connection: close\n",
            "Content-type: text/html\n\n",
            "<html>\n",
            "<head>\n",
            "<title>Syslog Query</title>\n",
            "</head>\n",
            "<STYLE>\n",
            "           <!--\n",
            "           a:link{ text-decoration : none; color : #93B1D3}\n",
            "           a:hover{ text-decoration : none; color : #417AB0}\n",
            "           a:active{ text-decoration : none; color : #417AB0}\n",
            "           a:visited{ text-decoration : none;}\n",
            "           body {\n",
            "                       scrollbar-arrow-color: #244A75;\n",
            "                       scrollbar-base-color: #001E4B;\n",
            "                       scrollbar-darkshadow-color: #001E4B;\n",
            "                       scrollbar-track-color: #000000;\n",
            "                       scrollbar-face-color: #001E4B;\n",
            "                       scrollbar-border-color: #244A75;\n",
            "                       scrollbar-shadow-color: #244A75;\n",
            "                       scrollbar-highlight-color: #244A75;\n",
            "                       scrollbar-3d-light-color: #244A75;\n",
            "           }\n",
            "           -->\n",
            "</STYLE>\n",
            "<body bgcolor=\"#000000\" text=\"#93B1D3\" link=\"#93B1D3\" vlink=\"#93B1D3\">\n";
my $SQL_Print = $SQL;
$SQL =~ s/\\/\\\\/g;
$SQL = qq {$SQL};
my $SQL_URI = uri_escape($SQL_Print);
my $URL = "http://$HostFQDN/cgi-bin/query.pl?query=";
if ($SQL eq '') {
print   "<form name \"sql_query_html\" action=/cgi-bin/query.pl method=post>\n",
      "<table border=0 cellpadding=0 cellspacing=0>\n",
      " <tr width=\"100%\">\n",
       "  <td><input type=text size=100 name=query value=\"$SQL_Print\"></td><td align=right><input type=submit
name=send value=Send><a href=\"http://$HostFQDN/cgi-bin/query_plain.pl?query=$SQL_URI\" target=\"_blank\">View
Results as Plain Text</a></td>\n",
      " </tr>\n",
      "</table>\n",
      "</form>\n",
            "</body>\n",
            "</html>\n";
}
else {
my $DBH = DBI->connect('DBI:mysql:syslog', 'readonly', '', { RaiseError => 1, AutoCommit => 0}) || die "Database
connection not made: $DBI::errstr\n";
print   "<form name \"sql_query_html\" action=/cgi-bin/query.pl method=post>\n",
            "<table border=0 cellpadding=0 cellspacing=0>\n",
            " <tr width=\"100%\">\n",
            "  <td><input type=text size=100 name=query value=\"$SQL_Print\"></td><td align=right><input type=submit
name=send value=Send><a href=\"http://$HostFQDN/cgi-bin/query_plain.pl?query=$SQL_URI\" target=\"_blank\">View
Results as Plain Text</a></td>\n",
            " </tr>\n",
            "</table>\n",
            "</form>\n",
            "<font face=\"Courier\" color =#003370>\n",
            "<table border=0 cellpadding=1 cellspacing=1>\n",
            "<td nowrap bgcolor=#244A75 cellpadding=1 cellspacing=1 align=left>Date - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20date%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20date%20asc\">\/\\</font></td>",
      "<td nowrap bgcolor=#244A75 align=left> Time - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20time%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20time%20asc\">\/\\</font></td>",
```

```
	"<td nowrap bgcolor=#244A75 align=left> Host - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20host%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20host%20asc\">/\\</font></td>",
	"<td nowrap bgcolor=#244A75 align=left> Notifier - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20notifier%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20notifier%20asc\">/\\</font></td>",
	"<td nowrap bgcolor=#244A75 align=left> ErrorType - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20errortype%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20errortype%20asc\">/\\</font></td>",
	"<td nowrap bgcolor=#244A75 align=right> EventID - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20eventid%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20eventid%20asc\">/\\</font></td>",
	"<td nowrap bgcolor=#244A75 align=left> User - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20user%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20user%20asc\">/\\</font></td>",
	"<td nowrap bgcolor=#244A75 align=left> ErrorText - <font face=\"arial\"><a
href=\"$URL$SQL_URI%20order%20by%20errortext%20desc\">\/</a> <a
href=\"$URL$SQL_URI%20order%20by%20errortext%20asc\">/\\</font></td>",
	"</tr>\n";

my $STH = $DBH->prepare($SQL);
$STH->execute();

my($Date,$Time,$Host,$Notifier,$ErrorType,$EventID,$User,$ErrorText);
$STH->bind_columns(undef,\$Date,\$Time,\$Host,\$Notifier,\$ErrorType,\$EventID,\$User,\$ErrorText);

my $Counter = 0;

while($STH->fetch() ) {
		$Counter++;
		$ErrorText =~ s/  /<br>\n/g;
		#
		# HTML Output
		#
		#	"Date      : $Date<br>\n",
		#	"Time      : $Time<br>\n",
		#	"Hostname  : $Host<br>\n",
		#	"Notifier  : $Notifier<br>\n",
		#	"ErrorType : $ErrorType<br>\n",
		#	"EventID   : $EventID<br>\n",
		#	"User      : $User<br>\n",
		#	"ErrorText :<br>\n","$ErrorText<br>\n",
		#	"</font>\n",
		#	"-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-<br>\n",
		#	"-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-<br>\n";
		print	"<tr>",
			"<td nowrap bgcolor=#001E4B align=left valign=top>$Date </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $Time </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $Host </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $Notifier </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $ErrorType </td>",
			"<td nowrap bgcolor=#001E4B align=right valign=top> $EventID </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $User </td>",
			"<td nowrap bgcolor=#001E4B align=left valign=top> $ErrorText</td>",
			"</tr>\n";
}
print	"</table></font><br>\n",
		"Rows returned  : $Counter<br>\n",
		"<form name \"sql_query_html\" action=/cgi-bin/query.pl method=post>\n",
		"<table border=0 cellpadding=0 cellspacing=0>\n",
		" <tr>\n",
	"  <td><input type=text size=100 name=query value=\"$SQL_Print\"></td><td align=right><input type=submit
name=send value=Send><a href=\"http://$HostFQDN/cgi-bin/query_plain.pl?query=$SQL_URI\" target=\"_blank\">View
Results as Plain Text</a></td>\n",
		" </tr>\n",
		"</table>\n",
		"</form>\n",
		"</html>\n";

$STH->finish();
```

```
$DBH->disconnect();
}
```

query_plain.pl

```
#!/usr/bin/perl
#
#
use strict;
use CGI;
use DBI;
#
# CGI
my($query) = CGI->new();
my($SQL) = $query->param("query");
print   "Connection: close\n",
        "Content-type: text/plain\n\n",
        "Date\tTime\tHost\tNotifier\tErrorType\tEventID\tUser\tErrorText\n";
my $DBH = DBI->connect('DBI:mysql:syslog', 'readonly', '', { RaiseError => 1, AutoCommit => 0}) || die "Database
connection not made: $DBI::errstr\n";
$SQL =~ s/\\/\\\\/g;
$SQL = qq {$SQL};
my $STH = $DBH->prepare($SQL);
$STH->execute();

my $Counter = 0;

my($Date,$Time,$Host,$Notifier,$ErrorType,$EventID,$User,$ErrorText);
$STH->bind_columns(undef,\$Date,\$Time,\$Host,\$Notifier,\$ErrorType,\$EventID,\$User,\$ErrorText);
while($STH->fetch() ) {
        $Counter++;
        print "$Date\t$Time\t$Host\t$Notifier\t$ErrorType\t$EventID\t$User\t$ErrorText\n";
}


$STH->finish();
$DBH->disconnect();
```

report_general_events.pl

```
#!/usr/bin/perl -w
#
#
#use strict;
use Date::Calc qw(:all);
use DBI;
use MIME::Lite;
#
#
#
my ($Year,$Month,$Day) = Today();
($Year,$Month,$Day )=Add_Delta_Days($Year,$Month,$Day,-1);
if(length($Month) < 2) {
        $Month = "0" . $Month;
}
if(length($Day) < 2) {
        $Day = "0" . $Day;
}
#
# Mail Conf
my $mailserver    = 'yourmailexchange.yourdomain.com';
my $from_address  = 'SyslogReporter@YourDomain.com';
my $to_address    = 'YourEmailAddress@YourDomain.com';
my $subject       = "[SyslogReport] - $Year-$Month-$Day - General Events";
my $mime_type     = 'text/plain';
my $message       = "";
my $filename_path = "";
my $filename      = "";
#
```

```
# Database Conf
my $DBH = DBI->connect('DBI:mysql:syslog', 'readonly', '', { RaiseError => 1, AutoCommit => 1});
my $SQL = "select distinct host from tmp_host";
#
# Program
my @Hosts=ExecuteSQLHosts($DBH,$SQL);
$message = $message . "Syslog Report for $Year-$Month-$Day\n";
$message = $message . "Hosts :\n-------------------------------------\n";
foreach (@Hosts) {
        $message = $message . "$_\n";
}
$SQL = "select * from syslog where date = '$Year-$Month-$Day' and notifier != 'security' and errortext != 'Virus definitions
are current.' order by host asc, notifier asc, time asc";
my @Result=ExecuteSQLSyslog($DBH,$SQL);
foreach(@Result){
        $message = $message . "$_\n";
}

$DBH->disconnect();
MailMIME();
#
# End
exit(0);

sub MailMIME {
        my $mime_msg = MIME::Lite->new(
                From    => $from_address,
                To      => $to_address,
                Subject => $subject,
                Type    => $mime_type,
                Data    => $message
                )
                or die "Error creating MIME body: $!\n";
        #
        #
        #
        #
        if ($filename_path && $filename ne "") {
                        $mime_msg->attach(
                        Type     => 'text/plain',
                        Path     => $filename_path,
                        Filename => $filename
                        )
                        or die "Error attaching file: $!\n";
        }
        $mime_msg->send;
}

sub ExecuteSQLHosts {
        my $DBH = $_[0];
        my $SQL = $_[1];
        my $Host;
        $SQL = qq{$SQL};
    my $STH = $DBH->prepare($SQL);
        $STH->execute();
        $STH->bind_columns(undef,\$Host);
        while($STH->fetch()){
                push @Hosts, $Host;
        }
        $STH->finish();
        return(@Hosts);
}

sub ExecuteSQLSyslog {
    my $DBH = $_[0];
        my $SQL = $_[1];
        $SQL = qq{$SQL};
        my $STH = $DBH->prepare($SQL);
        my @Result;
        my $Previous_Host = "";
```

```
	my $Previous_Notifier = "";
	$STH->execute();
	$STH->bind_columns(undef,\$Date,\$Time,\$Host,\$Notifier,\$ErrorType,\$EventID,\$User,\$ErrorText);
	while($STH->fetch() ) {
		#
		# RegEx to cut down on logs reported
		#
		# Printing Successful
		if ($Notifier eq "print" && $ErrorType eq "info" && $EventID eq "10") {
			next;
		}
		#
		# Printing Deleted Jobs
	if ($Notifier eq "print" && $ErrorType eq "info" && $EventID eq "13") {
	      next;
	}
		#
		# MAD Monitoring
		if ($Notifier eq "msexchangesa" && $ErrorType eq "info" && $EventID eq "9095") {
			next;
		}
		#
		# MAD Monitoring
		if ($Notifier eq "msexchangesa" && $ErrorType eq "info" && $EventID eq "9096") {
			next;
		}
		#
		# Security Policy Update Succesfully
		if ($Notifier eq "scecli" && $ErrorType eq "info" && $EventID eq "1704") {
			next;
		}
		#
		# NTDS Online Defragmentation Started
		if ($Notifier eq "ntds isam" && $ErrorType eq "info" && $EventID eq "700") {
			next;
		}
		#
		# NTDS Online Defragmentation Completed
		if ($Notifier eq "ntds isam" && $ErrorType eq "info" && $EventID eq "701") {
			next;
		}
		#
		# NTDS Starting Backup
		if ($Notifier eq "ntds isam" && $ErrorType eq "info" && $EventID eq "200") {
			next;
		}
		#
		# NTDS Completed Backup
		if ($Notifier eq "ntds isam" && $ErrorType eq "info" && $EventID eq "202") {
			next;
		}

		if ($Previous_Host ne $Host) {
			push(@Syslog,"\n\n$Host : \n--------------------------------");
			$Previous_Host = $Host;
		}
		if ($Previous_Notifier ne $Notifier) {
			push(@Syslog,"\n$Notifier alerts :\n$Date $Time $Host $Notifier $ErrorType $EventID
$User $ErrorText");
			$Previous_Notifier = $Notifier;
		}
		else {
			push(@Syslog,"$Date $Time $Host $Notifier $ErrorType $EventID $User $ErrorText");
		}
		next;
	}
	$STH->finish();
	return(@Syslog);
}
```

report_security_events.pl

```
#!/usr/bin/perl -w
#
#
#use strict;
use Date::Calc qw(:all);
use DBI;
use MIME::Lite;
#
#
#
my ($Year,$Month,$Day) = Today();
($Year,$Month,$Day )=Add_Delta_Days($Year,$Month,$Day,-1);
if(length($Month) < 2) {
        $Month = "0" . $Month;
}
if(length($Day) < 2) {
        $Day = "0" . $Day;
}
#
# Mail Conf
my $mailserver    = 'yourmailexchange.yourdomain.com';
my $from_address  = 'SyslogReporter@YourDomain.com';
my $to_address    = 'YourEmailAddress@YourDomain.com';
my $subject       = "[SyslogReport] - $Year-$Month-$Day - Security Events";
my $mime_type     = 'text/plain';
my $message       = "";
my $filename_path = "";
my $filename      = "";
#
# Database Conf
my $DBH = DBI->connect('DBI:mysql:syslog', 'readonly', '', { RaiseError => 1, AutoCommit => 1});
my $SQL = "select distinct host from tmp_host";
#
# Program
my @Hosts=ExecuteSQLHosts($DBH,$SQL);
$message = $message . "Syslog Report for $Year-$Month-$Day\n";
$message = $message . "Hosts :\n-------------------------------------\n";
foreach (@Hosts) {
        $message = $message . "$_\n";
}
$SQL = "select * from syslog where date = '$Year-$Month-$Day' and notifier = 'security' and EventID  != '538' and EventID
!= '540' and EventID != '672' and EventID != '673' and EventID != '674' and EventID != '675' and EventID != '677' and
EventID != '680' order by host asc, EventID asc, time asc";
my @Result=ExecuteSQLSyslog($DBH,$SQL);
foreach(@Result){
        $message = $message . "$_\n";
}

$DBH->disconnect();
MailMIME();
#
# End
exit(0);

sub MailMIME {
        my $mime_msg = MIME::Lite->new(
                From    => $from_address,
                To      => $to_address,
                Subject => $subject,
                Type    => $mime_type,
                Data    => $message
                )
                or die "Error creating MIME body: $!\n";
        #
        #
        #
        #
```

```
        if ($filename_path && $filename ne "") {
                $mime_msg->attach(
                Type     => 'text/plain',
                Path     => $filename_path,
                Filename => $filename
                )
                or die "Error attaching file: $!\n";
        }
        $mime_msg->send;
}

sub ExecuteSQLHosts {
        my $DBH = $_[0];
        my $SQL = $_[1];
        my $Host;
        $SQL = qq{$SQL};
    my $STH = $DBH->prepare($SQL);
        $STH->execute();
        $STH->bind_columns(undef,\$Host);
        while($STH->fetch()){
                push @Hosts, $Host;
        }
        $STH->finish();
        return(@Hosts);
}

sub ExecuteSQLSyslog {
    my $DBH = $_[0];
        my $SQL = $_[1];
        $SQL = qq{$SQL};
        my $STH = $DBH->prepare($SQL);
        my @Result;
        my $Previous_Host = "";
        my $Previous_EventID = "";
        $STH->execute();
        $STH->bind_columns(undef,\$Date,\$Time,\$Host,\$Notifier,\$ErrorType,\$EventID,\$User,\$ErrorText);
        while($STH->fetch() ) {
                #
                # RegEx to cut down on logs reported
                #
                if ($Previous_Host ne $Host) {
                        push(@Syslog,"\n\n$Host : \n---------------------------------");
                        $Previous_Host = $Host;
                }
                if ($Previous_EventID ne $EventID) {
                        push(@Syslog,"\n$EventID alerts :\n$Date $Time $Host $Notifier $ErrorType $EventID
$User $ErrorText");
                        $Previous_EventID = $EventID;
                }
                else {
                        push(@Syslog,"$Date $Time $Host $Notifier $ErrorType $EventID $User $ErrorText");
                }
                next;
        }
        $STH->finish();
        return(@Syslog);
}
```

update_tmp_host.pl

```
#!/usr/bin/perl -w
#
# Updates the tmp_host tables host column in the syslog database with unique entries
# from the syslog tables host column, should run hour
use strict;
use DBI;
#
#
#
```

```
my($Host);
my(@Hosts);
#
#
#
my $DBH = DBI->connect('DBI:mysql:syslog','root','rootuserspassword',{RaiseError=>1,AutoCommit=>1});
my $SQL = qq{select distinct host from syslog};
my $STH = $DBH->prepare($SQL);
$STH->execute();
$STH->bind_columns(undef,\$Host);
while($STH->fetch() ) {
        push @Hosts, $Host;
}
$STH->finish();
$DBH->disconnect();

$DBH = DBI->connect('DBI:mysql:syslog','root','rootuserspassword',{RaiseError=>1,AutoCommit=>1});
#
$SQL = qq{drop table tmp_host};
$STH = $DBH->prepare($SQL);
$STH->execute();
$STH->finish();
#
$SQL = qq{CREATE TABLE `tmp_host` (`Host` varchar(20) NOT NULL default '') TYPE=MyISAM};
$STH = $DBH->prepare($SQL);
$STH->execute();
$STH->finish();
#
foreach (@Hosts) {
        $SQL = qq{insert into tmp_host (host) values('$_')};
        $STH = $DBH->prepare($SQL);
        $STH->execute();
        $STH->finish();
}
$STH->finish();
$DBH->disconnect();
#
#
exit(0);
```

References

Microsoft *"Threats and Countermeasure Guide – Chapter 6 - Event Log".* - April 23rd, 2003
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/topics/hardsys/tcg/tcgch06.asp February 02, 2004

Microsoft "*Microsoft Security Bulletin MS01-013 - Windows 2000 Event Viewer Contains Unchecked Buffer*" - June 23, 2003
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-013.asp February 02, 2004

Defrance, Fred *"A Case for Centralized Logging"* December 7, 2001
http://ebuzzsaw.com/whitePapers/Case_for_Centralize_Logging.htm February 02, 2004

MandrakeSoft *"Mandrake Linux"* - 2004
http://www.mandrakelinux.com/en/9.2/features/ February 02, 2004

MandrakeSoft *"Software Management - MandrakeUpdate"* - 2004
http://www.linux-mandrake.com/en/demos/Spotlight/SoftwareMgr/pages/rpmdrake14.php3 February 02, 2004

MandrakeSoft "*Mandrake Linux 8.2: Reference Manual - Chapter 12. msec – Mandrake Security Tools"* 2002
http://www.mandrakeuser.org/docs/mdoc/ref/prog-msec.html February 02, 2004

Schulze, Martin *"Sysklogd"* January 04, 2001
http://www.infodrom.org/projects/sysklogd/ February 02, 2004

Lonvick, C. *"RFC3164 - The BSD syslog Protocol"* – August 2001
http://www.faqs.org/rfcs/rfc3164.html February 02, 2004

SaberNet.net *"NTSyslog"* – 2002
http://ntsyslog.sourceforge.net/ February 02, 2004

MySQL AB "*MySQL"* – 2004
http://www.mysql.com/products/mysql/index.html February 02, 2004

MySQL AB *"MySQL Control Center"* – 2004
http://www.mysql.com/products/mysqlcc/index.html February 02, 2004

Perl.org *"The Perl Directory: About Perl"* – 2004
http://www.perl.org/about.html February 02, 2004

Bunce, Tim *"Perl DBI"* – 2004
http://dbi.perl.org/about/ February 02, 2004

MandrakeSoft *"Scheduling II – cron"* – 2002
http://www.mandrakeuser.org/docs/admin/acron2.html February 02, 2004

Fairhurst, Gorry *"Cyclic Redundancy Check"* - January 09, 2001
http://www.erg.abdn.ac.uk/users/gorry/course/dl-pages/crc.html February 02, 2004

Nachreiner, Corey *"Anatomy of an ARP Poisoning Attack"* – 2003
http://www.watchguard.com/infocenter/editorial/135324.asp February 02, 2004

Volobuev, Yuri "syslogd fun" – August 27, 1997
http://www.insecure.org/sploits/aix.generic.syslogd.problem.html February 02, 2004

MandrakeSoft *"Mandrake Linux Kernel Updates"* – July 28, 2003
http://www.mandrakesecure.net/en/kernelupdate.php *February 02, 2004*

MandrakeSoft *"Mandrake Linux 8.2: Server Reference Manual - 10.8.2. System services and tcp_wrappers"* – 2002
http://www.mandrakeuser.org/docs/mdoc/server/network-security.html February 02, 2004

ElementK *"Using Audit Policies to Secure Your Windows 2000 Network"* – 2000
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnexnt00/html/ewn0054.asp February 02, 2004