Global Information Assurance Certification Paper

Audit Log Consolidation in a Windows Environment

Bennett Schneider
GSEC Certification
Option 1
Version 1.4b
August 26, 2003

Abstract

Logging in Windows has some severe shortcomings. It is limited to the Windows event log and its interface, the Event Viewer. Here we present its inadequacies and solutions to them. This paper serves as an introduction to logging options available in either a mixed Windows environment or a pure Windows environment. It also focuses on the many alternatives that users have for converting the Event Log into plaintext for easy reading and log parsing. It also covers many tools that allow for easy conversion of event log information into syslog format. Furthermore, it provides an overview of syslog server implementations that run under Windows and their benefits over other log consolidation implementations. In addition, it also provides an overview of some of the log analysis options that are open to people, using either freeware or commercial software.

When an incident occurs on a system, the first tool to turn to is the system log. It provides valuable insight into an intrusion or even a simple malfunction and can be one of the most important tools at a system administrator's disposal. System logs are very simple in a Unix environment. Many tools come with the operating system to make the job much simpler. They provide much of the functionality for a complete centralized logging architecture. In the Windows universe, life is not so simple. The only tool that Windows provides is the event log, which has several important limitations.

The Windows Event Log

The event log can be viewed through an application known as the Event Viewer. The Event Viewer is located in the Administrative Tools. Information in the event log is categorized into three different folders on a normal workstation. The Application folder stores any messages from user applications, like errors or informational messages. The System folder stores messages relevant to system services and components, such as networking or hardware issues. The Security folder stores the information that is most useful to the auditor. It stores all information about logons, logoffs, accesses to security relevant information and more. Servers may have several other folders related to the various services that have been configured to run. In addition, the user may define custom log folders.

Before the Security folder stores any logs at all, it must first be set up. There are two ways to accomplish this. If a domain controller is being run, it can all be configured through group policy. If this is not the case, it can be done through the Local Security Policy area of Administrative Tools. Under Local Policies is located the Audit Policy. This is where the contents of the Event Log Security folder are configured. For any of the listed items, Success, Failure or both can be logged. Below is a table of recommended settings for Windows 2000 taken from the NSA's guide to securing Windows 2000.

Audit Category	Recommended Setting
Audit account logon events	Success, Failure
Audit account management	Success, Failure
Audit directory service access	No auditing
Audit logon events	Success, Failure
Audit object access	Failure
Audit policy change	Success, Failure
Audit privilege use	Failure
Audit process tracking	No auditing
Audit system events	Success, Failure

[1]

Extracting the Event Log

As stated before, the Event Viewer does have several limiting problems. It only allows very restricted viewing of the data. It incorporates no search capability and only allows sorting based on the fields it displays. This makes it very tedious to search through the logs manually. In addition, it stores all of its information in a binary format. This proprietary format limits its accessibility. It restricts administrators to tools which handle only its encoded format and renders all plaintext based tools useless. Fortunately, it does allow the event log to be output in a tab delimited format and even allows this to be done on remote computers. The downside is that this task cannot be automated. Since it is done through the Event Viewer's graphical user interface, it cannot be done through the use of a script.

While posing a small problem, this problem can be bypassed. One utility available is the Win32::EventLog Perl module. It can be easily integrated into a perl script that will allow the extraction of the Windows event logs and even incorporate any filtering or processing that is desired to limit the size of the log files.

An alternative to writing a Perl script is to use the tool Dumpel from the Windows Resource Kit. It allows command line access to the event log. Another method of doing this would be to use a utility from Somarsoft called DumpEvt This free utility allows you to extract the Windows Event logs into a plain text format. It is a much better alternative to the Dumpel utility because DumpEvt allows one to extract only those logs that have been added since the last time it was run. This is its default behavior and does not require any additional command line parameters to do this.

Dumpel, DumpEvt, or a script using Win32::EventLog can all be queued to run at a certain time daily using the Scheduled Task utility. Dumpel and DumpEvt also allow extraction of the event logs of remote machines. For Dumpel, this is done through the `-s` *computer* option. For DumpEvt, this is done through the `/computer=`*computer* option. This solves another of the event log's limitations. Without a centralized log repository, a clever hacker will go

[1] J. Haney, p. 30

undetected. It is quite easy for a hacker to sanitize the logs of the compromised machine. This could allow the break in to go unnoticed for quite a while. If the logs are being sent to a remote machine, the hacker must now compromise another machine in order to cover up his tracks. This makes his job much harder. The extra time that must be spent in the system makes it that much more likely that his activities will be noticed and that the attacker will be contained.

Using Dumpel or DumpEvt to store to a remote server does solve many of the problems. They are still somewhat flawed, unfortunately. Since they are only run daily, they do create a window of about a day in which someone could break in to a system and cover their tracks. If they were able to do so in that time frame, all record of their intrusion would be lost. The solution is to send our logs in real time. Any batch job will either provide a window or become so taxing on a system that it will be unable to do anything else. To see the second problem, one has to consider the network as a whole. If there are Unix machines, how will those logs be consolidated if we are using Windows exclusive tools? What if we have hardware devices, like routers, and we want to consolidate those logs as well? It would be quite difficult to have to develop a separate logging system for every environment. One the other hand, how should we decide to send our logs over the network? What standard should we adopt? Fortunately, we do not have to decide. A network logging standard has existed in the Unix world for many years. Its usage is universal in all flavors of Unix and it is utilized by many hardware devices as well. This standard is known as syslog and it is quite easy to understand.

The Syslog Format

The syslog format is a simple means of sending logs over the network and was defined in RFC 3164. It runs over UDP on port 514. UDP was originally chosen for its ease of implementation. Although UDP does not guarantee delivery, in practice very few, if any, logs are lost. In addition, the logs could arrive out of order, so careful attention must be paid to the timestamps. The possibility does exist for a denial of service attack on the log server. A malicious user could flood the log server with useless messages, blocking out relevant messages. These messages could also be faked, as the syslog protocol does not define any kind of authentication procedures. A flood attack would almost certainly be noticed due to the large amounts of network traffic that would be generated, but it might disguise exactly what the hacker was attempting to do. Despite the unlikelihood of this attack, it is a weakness of the system and because of this it should be understood by any implementer.

The format of the messages is very simple and straightforward. They consist of a priority, a timestamp, the originator's hostname, and a message. The message portion is a simple plaintext description of the log. The priority is a number made up of two factors. The first is the facility. This is a number between 0 and 23 which designates where the message came from. In a Unix environment, 0-15 are reserved for specific operating system uses. The numbers 16-23 are specifically reserved for custom user purposes and are

defined as local0 through local7. Since we will be sending in a Windows environment, we may use either since the reserved numbers will not be used by the operating system.

The other portion of the priority is the severity. It is a number from 0 to 7 and indicates how the message will impact the system. They range from debug level (7) to emergency level (0). To calculate the priority, multiply the facility by 8 and add the severity. This number is then used by the syslog server to decide where to route the incoming logs. It is important to note that in the syslog.conf file in Unix based systems, the severity will match it's own level, and all levels below it. For example, if an emergency level message is received, it will do the steps for all levels of that facility.

Outputting the Event Log in Syslog Format

We now need a means of converting the Windows event log into syslog format. Several options exist for this purpose, each with their own benefits and limitations.

Event Reporter is a one of the tools that allows us to convert our event log messages into syslog format. It is not freeware but it does have one feature that others lack. It gives users the ability to send out email messages for particularly important messages that occur. This is a nice feature, but is not absolutely necessary for the client to do this. If this is an important consideration, this can be done in the server portion of the syslog setup. Commercial software for the workstation can get quite expensive, since it must be licensed for every machine that you wish to extract logs from. Fortunately, there are freeware alternatives that provide plenty of features.

One of these is called SNARE for Windows. It consists of two executables, SNARECore and SNARE. SNARECore is all that is required to send syslog messages. It allows for a web-based configuration interface, which is best to disable. This is an unnecessary port to open on a machine and it is better to be safe than sorry. SNARE can be configured to output plaintext messages but its default behaviour is to add the syslog header and output these syslog messages on port 514. Syslog is the format that is desired so it is important to verify that this header is being added. Without it, many syslog servers will discard the messages.

Another simpler freeware alternative is known as NTSyslog. It runs as a Windows service, as all of these do. A Windows service is the equivalent of a Unix daemon and is a program which runs in the background. NTSyslog comes with the source code, and can be compiled using Visual C++. Configuration is done either through a separate executable called NTSyslogCtrl that provides a graphical interface or through modifying several registry entries. The registry folder that it uses is located at [HKEY Local Machine\SOFTWARE\SaberNet]. It allows for two syslog servers to be specified for redundancy. SNARE does not provide this, but it is also not an absolutely essential feature. It can also specify exactly which entries to send messages for and which to ignore. This feature helps cut down on the sending of unnecessary log messages, which take up space and network traffic. It also allows for different messages to be sent on

different priorities, which helps in categorizing the messages once they get to the syslog server. SNARE does not provide users with this freedom. Configuration through the registry is easy and a VB script can easily be configured to write these registry values for deployment of NTSyslog through group policy, which eases installation for a large number of workstations.

Outputting our logs in syslog format is very useful if we have a Unix machine handy. We would already have our syslog server and life would be good. If we have an exclusively Windows environment, life gets a little trickier. Fortunately, a number of solutions exist to fill this void and provide us with a syslog server that runs under Windows.

Syslog Servers for Windows

One possible syslog server is the Kiwi Syslog Daemon. It comes in both a commercial and freeware version. The freeware version has enough functionality to suffice as a barebones syslog server, but the commercial version provides many nice features. In addition to syslog messages, it can also receive SNMP traps, which are sent by some hardware devices. It will display graphs of syslog activity and send email messages when certain messages show up. It also handles all log archival automatically. These features are quite useful, but other freeware alternatives do exist.

One of these is provided by 3com and is called WSyslogD. It is very basic but provides some nice features. It can be set to separate the logs based on IP address, but not based on facility. This is its major drawback, and it might keep some from using it. It also does incorporate log rotation, which is important to keep log files at a manageable size.

Another freeware version is SL4NT. It comes in both commercial and freeware versions. The freeware is very basic, but does provide all of the features expected of a syslog server daemon. The commercial version integrates many features such as email notification, routing based on source IP address and logging to an ODBC database.

It is even possible to run the Unix syslog server in Windows using Cygwin. It requires the base cygwin package and several additional packages. These include the cygrunsrv (Admin), gcc, make,(Development) and inetutils (Net) packages. For inetutils, the source is necessary since a compiled version of syslogd is no longer included with the binaries. The rest only require the binaries. Gcc, make, and their related packages are required to build the syslogd binaries. Cygrunsrv is a program that allows any cygwin executable to become a Windows service. A service is the Windows equivalent to a Unix daemon. The only requirement cygrunsrv makes of a program is that it must not behave like a normal Unix daemon. Normal Unix daemons exit after forking a child process. This behavior will not yield the results that we want. Syslogd does not come with a command line switch to prevent forking, so a slight source code modification is required.

Upon opening the /usr/src/inetutils-xxxx/syslogd/syslog.c file, the line that reads:

```
if (!Debug)
```

```
            (void)daemon(0, 0);
      else
      {
```

must be changed to:

```
      if (!Debug)
      {}
//          (void)daemon(0, 0);
      else
      {
```

After making this minor change, change to the /usr/src/inetutils-xxxx/ directory. Execute the ./configure script to configure the various inet make files. Now type make in the /usr/src/inetutils-xxxx/ directory. This will compile some necessary libaries, but will exit with an error. This is OK. Make can now be run in the /usr/src/inetutils-xxxx/syslogd/ directory to get the modified syslogd executable that we want.

A syslog.conf file must now be created. This should be in the same format as the Unix syslog.conf with one exception. When sending to files, the directories must be specified as /cygdrive/c/my.log if you were sending them to c:\my.log. The reason for this is that syslogd expects files to start with a leading forward slash. Copy both the syslog.conf and syslogd.exe to the c:\syslog directory. All log files that you specified in the syslog.conf must be created prior to starting syslogd, just as you would have to in Unix.

Installing the service is quite easy now. Execute the following command through cygwin:

```
cygrunsrv -I SyslogDaemon -p c:/syslog/syslogd -a "-f
c:/syslog/syslog.conf -p c:/syslog/log" -1
c:/syslog/daemon.log -2 c:/syslog/daemon.log
```

The –I option specifies that this is an installation. The –p specifies the working path. The –a specifies what arguments are to be passed to the application when it starts. The –f option being passed to syslogd specifies where the configuration file is located and the –p specifies the location of the log device. The –1 is used to redirect stdout to a file and the –2 is used to redirect stderr. In this case, both are being sent to the c:\syslog\daemon.log file. The forward slashes are necessary for cygwin to interpret the file paths correctly. The service will need to be started this time, but by default it is installed in auto mode. This means that it will start whenever the system loads. To load it now without rebooting, execute another switch of cygrunsrv:

```
cygrunsrv -S SyslogDaemon
```

The –S specifies start. To uninstall the program, simply run the following line at the commandline:

```
cygrunsrv -R SyslogDaemon
```

and the SyslogDaemon program will be stopped and removed from the list of services.

One additional step that is nice is to set up log rotation. This can be accomplished through a scheduled task that runs nightly. The batch file must move the old task to another location (possibly add the date as a suffix to the file name) and also stop and restart the server. These two can be done using the

commands `net stop SyslogDaemon` and `net start SyslogDaemon` respectively.

This setup has the benefit of running a completely familiar syslog server. It maintains the full flexibility of Unix syslog and does not force people to learn a new standard. The syslog.conf file is something that most people familiar with Unix or Linux will not have a problem configuring. Furthermore, cygrunsrv using a similar setup can be used to run many other traditionally Unix only applications. One example that is very useful from a security standpoint and has no Windows equivalent is OpenSSH. This package allows one to log in to a machine remotely like telnet, but supports a variety of encryption technologies to make sure that the session is not intercepted.

Examining the Logs

Now that the logs have been routed to a central location, they should be regularly reviewed. While this can be done manually, this task becomes nearly impossible for all but the smallest of networks. The log files for a single workstation can easily grow to hundreds of lines in one day while the log files of a busy domain computer can go into the megabytes. For people with lots of extra time and a penchant for the boring and repetitive, this might be heaven. For everyone else, it is a perfect opportunity for automation.

The registered version of the Kiwi Syslog Daemon provides a very nice solution to this problem. It allows the user to implement filters based on priority, time of day, IP address, hostname, and message contents. When examining the message contents, it allows for full regular expression support. These filters can all be combined to provide a very powerful screen that will catch the majority of all intrusion attempts. One example might be to combine a time filter made up of after work hours with message filter for administrative account logins. This situation is one of many that are a good indication that something inappropriate is happening on a network.

The filters work in conjunction with a number of actions that can be defined. These include running an external executable, executing a script, emailing an alert, logging to a database, and many others. Many pagers and cell phones support receiving email, so the email notification can be used to notify these devices as well for twenty-four hour warning.

As an added feature, Kiwi Syslog Daemon provides some support for graphing the log activity over the course of one hour or up to one day. The log graphs are nice, but do not afford much functionality over the regular expression support.

For those that want a freeware solution, one of the best log monitoring tools for Unix can be made to work under Windows. Swatch was originally designed to monitor Unix syslog logs and is available under the GNU General Public License. Since we are using the exact same syslog format, Swatch will work perfectly. Swatch is incredibly versatile as well and can be configured to look in a log file for any regular expression. Swatch is written using perl, but depends on some Unix utilities for it to work. Fortunately, it can be made to work under Windows, but not as a service.

Cygwin requires one more package beyond those used for syslogd. The perl package, which can be found underneath the Interpreters tree, is necessary. Once perl is installed, Swatch also requires some additional perl modules in order to function properly. In particular, it requires the Bit::Vector, Date::Calc, File::Tail and Date::Parse modules. All of these can be downloaded from http://www.cpan.org. After downloading, gunziping, and untaring the files, run the following series of commands in each module's directory to ensure that they are properly installed:

```
perl Makefile.PL
make
make test
make install
make realclean
```

Swatch is now able to run, but it must be configured. Swatch has the ability to follow a file as it grows and look for certain regular expression. When it reaches one, it will perform the task that the user has designated for it. In Unix, Swatch supports mail notification, echoing to the command line, ringing the system bell, executing commands, and sending text to users. While running in Windows, executing a command and echoing are the only usable options.

One of the most useful of the commands to have Swatch or the Kiwi Syslog Daemon run is the net send command. It allows the user to pop up alert windows on either the local computer or a remote computer. The default installation of Windows includes the Messenger service, which is the program which listens for the net send messages. The proper syntax is `net send` *<computer>* *<message>* where *<computer>* is either the IP address or computer name and *<message>* is what will appear in the popup.

To get the Swatch daemons running, open cygwin. After entering the swatch directory, run the following command:

`./swatch -c` config file `-t` log file `--use-cpan-file-tail -daemon --pid-file` pid file `> /dev/null`

This will set up the log parsing daemon for one log file, throwing away all output from swatch and the commands that are executed by it. The pid file is not required, but it does make log rotation much easier. Starting these log monitors can be automated using a script which is loaded with in a user's Startup folder. The main problem with this solution is that the script will only work as long as there is a user logged in. Forcing a user to be logged in constantly is quite a drawback, even though the screen lock provides basic security.

In addition, every time the logs are rotated, the daemons must be restarted. The number stored in each swatch instance's pid file must be passed to the kill command in a cygwin. After that, the same call that started swatch maybe be run again.

The configuration file for swatch is relatively easy to create. It consists of either the word "watchfor" or "ignore" and then a regular expression. On the next line following the "watchfor" statement is placed the action. For Windows, this

will always be “echo” by itself or “exec” followed by the command and the options that need to be passed to it. To include the message that caused the alert in the exec command, use a $0. Any other number will include that field of the message. If “echo” is used with the –t option, it is important to redirect the output to a legitimate file instead of /dev/null.

An alternative is to run swatch using the –f option instead of the –t. –f tells it to run through a single pass of a given logfile. For this option, use of the “echo” command is probably most appropriate. This output can be redirected to a file so that a manual review can be performed on the filtered logs to make sure that no malicious activity was underway. This option is much simpler than the –t option, but it does allow a window of time for a compromise to occur. The command to start this would be:

`./swatch -c` config file `-f` log file `>` results

Important messages in Windows

Now that the log review tools are in place, the administrator must carefully consider which regular expressions are important enough to include in the log monitoring tool. This is a very important decision to make and it can mean the success or failure of the logging solution. If the regular expressions match too many legitimate messages, the person in charge will be inundated with logs to hand review and will be sure to miss a malicious one among all of the false positives. On the other hand, if the regular expressions are not general enough, they will miss the important logs that would clue someone in on a compromise.

Below is a listing of events from the Windows logs that are normally associated with malicious behaviour. They are taken from Counterpane Securities, one of the leaders in network security.

Event ID	Event Descriptor
529	Logon Failure: Reason: Unknown user name or bad password
530	Logon Failure: Reason: Account logon time restriction violation
531	Logon Failure: Reason: Account currently disabled
532	Logon Failure: Reason: The specified user account has expired
533	Logon Failure: Reason: User not allowed to logon at this computer
534	Logon Failure: Reason: The user has not been granted the requested logon type at this machine
535	Logon Failure: Reason: The specified account's password has expired
537	Logon Failure: Reason: An unexpected error occurred during logon
624	User Account Created
627	Change Password Attempt
630	User Account Deleted
632	Security Enabled Global Group Member Added
636	Security Enabled Local Group Member Added
642	User Account Changed
643	Domain Policy Changed
608	User Right Assigned
609	User Right Removed
610	New Trusted Domain
611	Removing Trusted Domain
612	Audit Policy Changed
614*	IPSec policy agent disabled
615*	IPSEC Policy Changed
616*	IPSec policy agent agent encountered a potentially serious failure.
617*	Kerberos Policy Changed
618*	Encrypted Data Recovery Policy Changed
620*	Trusted Domain Information Modified
512	Windows NT is starting up.
513	Windows NT is shutting down.
516	Internal resources allocated for the queuing of audit messages have been exhausted, leading to the loss of some audits.
517	The audit log was cleared.

[2]

The table entries marked with an asterisk are only necessary if IPSec is being run. In addition, many of these descriptions only apply to Windows NT. While the descriptions may have changed, the Event IDs still correspond to the same messages, so it is safe to use the Event Id as the discerning pattern in the regular expressions. The majority of these should be relatively rare events so they should not place an extreme burden on a person who is assigned to manually review them.

With these tools in place, almost any incident on any of the machines can be easily tracked down. In addition, many incidents will be discovered through their use. This shows that a consolidated logging setup can be accomplished in Windows for relatively cheap or even free and that such a setup has a variety of benefits.

[2] Counterpane Securities

Counterpane: Log Analysis: Windows Logging. 5 December 2001
URL: http://www.counterpane.com/log-windows.html (26 August 2003)

Haney, J. Guide to Securing Microsoft Windows 2000 Group Policy: Security Configuration Tool Set. Version 1.2. 3 December 2002
URL: http://nsa2.www.conxion.com/win2k/guides/w2k-3.pdf (26 August 2003)

Dougherty, Jesse. Win32::EventLog – Process Win32 Event Logs from Perl
URL:
http://aspn.activestate.com/ASPN/docs/ActivePerl/site/lib/Win32/EventLog.html
(26 August 2003)

Somarsoft Utilities .
URL: http://www.systemtools.com/somarsoft/ (26 August 2003)

Dumpel.exe:Dump Event Log.
URL:
http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/dumpel-o.asp (26 August 2003)

Lonvick, C. The BSD syslog Protocol . August 2001
URL: http://www.ietf.org/rfc/rfc3164.txt (26 August 2003)

Event Reporter: The NT Event Monitor
URL: http://www.eventreporter.com/en/ (26 August 2003)

NTSyslog
URL: http://ntsyslog.sourceforge.net/ (26 August 2003)

Syslog Daemon for Windows, Free Syslog Server, Firewall logging, Kiwi Syslog Daemon
URL: http://www.kiwisyslog.com/products.htm (26 August 2003)

3Com Software Library - Utilities for 32 bit Windows
URL: http://support.3com.com/software/utilities_for_windows_32_bit.htm (26 August 2003)

Krainer, Franz. SL4NT 0.3. 25 July 2003.
URL: http://www.netal.com/sl4nt03.htm (26 August 2003)

SWATCH: The Simple WATCHer of Logfiles
URL: http://swatch.sourceforge.net/ (26 August 2003)

Cygwin Information and Installation
URL: http://www.cygwin.com/ (26 August 2003)