Global Information Assurance Certification Paper

Centralized Logging in a Windows Environment
Filomeno Iturzaeta IV
GSEC Version 1.4b

Summary

Centralized logging in a Windows environment is a powerful tool for a system administrator. Building a centralized logging system for the Windows environment is the goal of this paper. Instead of focusing on a product you can buy, you will create a functional centralized logging system for free. This paper will cover the need for a centralized logging system, and a bit of its history. The next topic will be the requirements and options we have to make this system work. Setting up a syslog server and getting the event log messages sent to the syslog server will be covered next. The paper will conclude with a tutorial on how to view the logs, do some basic reporting, and system maintenance.

Introduction

Being able to monitor all server logs from one location in real time is a proactive way to keep your network secure and handle problems as soon as they happen. 'Defense in depth[1]', one of the fundamentals of computer security means that your security is designed in a series of layers. Centralized logging is another step you can take to add a layer to your networks security model. Auditing your network should be a standard practice. Using the centralized logging model helps in the auditing process.

Centralized Logging is not a new concept, but for the Windows Environment it is not that common. Let us look into what kind of logging is done in a Windows Environment. In its most basic form Windows has the Event Log. The Event Logs consist of at least three basic log categories. Those categories are Application, Security, and System logs. The security log contains information such as successful logins, login failures, policy changes, password changes, etc. Being able to audit this information is very important to the administrator.

Windows servers store the event logs locally, which makes it cumbersome for an administrator to check logs on multiple servers. As a result, most administrators neglect to monitor the event logs unless there is a problem. Here are just a few of examples of how a centralized logging system can help you in your ability to secure your network. In the case of a system compromise, being able to piece together the steps the attacker took to get into your network and where they went after they gained access helps greatly in the forensic and recovery process. Having the ability to check all of your servers security logs for log in failures can help you find someone that is trying to crack passwords, and if they succeeded. Checking for unauthorized or abnormal use of user accounts can tip you off of a malicious insider. After an attack, your log files are an invaluable resource in determining what happened. A centralized log server

[1] Defense in Depth, Todd McGuiness See Glossary

attempts to make that more secure by keeping a copy of the Event Logs on a secure server that the attacker does not have access to.

There are products available that you can buy to handle Event Logs, and besides being relatively expensive these products either do not integrate well or it takes too many resources to manage and maintain the product. Not all security conscious people have it in their budget to purchase such a system. This paper will show you how you can create a centralized logging system for free.

History of Centralized Logging

Centralized Logging has been around for a long time in the form of Syslog on the Unix platforms and other variants of Unix. *'In its most simplistic terms, the syslog protocol provides a transport to allow a machine to send event notification messages across IP networks to event message collectors - also known as syslog servers*[1]*'*. Syslog has 3 roles of operation[2]:

1. Device Sends a message to a *relay* or *collector*
2. Relay Relays a message to another *relay* or a *collector*
3. Collector Collects messages from *devices* or *relays*

Many network devices can send syslog messages such as servers, routers, and switches. The syslog collector accepts syslog messages from remote devices on UPD[3] port 514. The message includes the 'Date', 'Time', 'Facility', 'Priority', 'Hostname' and the 'Message'. 'Date' and 'Time' is self explanatory as well as Hostname and Message. Facility is an identifier of what sent the message, that could be, 'Application', 'Daemon (service)', etc. The Priority is the urgency of the message, 'warning', 'error', and 'information' are examples of a priority. Once the message is received, it can be presented in a number of ways. Writing the message to a file or being displayed on a console are just two ways to present a syslog message. By writing the syslog messages to a file, the log can then be maintained and reviewed from one central location. Some of the administration benefits of a centralized syslog collector are: Archiving of all server event logs from one location. Collecting events before they can be modified. Looking at events from a broad range of network sources and checking the logs of a source that is no longer available (in the case of denial of service or a catastrophic failure). Syslog is a proven method of logging network device messages, and is the perfect candidate for a centralized logging solution.

Requirements of a Syslog System

The core of a centralized logging system is the syslog collector. There are many syslog collectors available for the windows platform, each with its own plusses and minuses. It is really up to you to decide which one you are

[1] RFC3164 C. Lonvick Section 1 Paragraph 2 See References
[2] RFC3195 New & Rose Section 2 See References
[3] UDP Protocol, See Glossary

comfortable with. The one I prefer to use is the Kiwi syslog service[1] from Kiwi Enterprises.

The next thing we need to do is find an application that can act as our syslog device. The application should run as a service and be able to convert and send our event log messages to the syslog collector. It really is not important which one you use, but the one you use needs to be able to format the 'message' part of the syslog message in a compatible way for you to report and monitor it without too much trouble. The application I chose is called NTSyslog from SaberNet[2]. NTSyslog is available free under the terms of the GNU General Public License. Whenever a new event message is generated in the application, security, or system log, NTSyslog gets the message and sends it to the syslog collector.

Time synchronization is a crucial part of any log and forensic analysis. In order to decipher and analyze logs, the log times must be synchronized, or none of it will make sense. The time services for Windows 2000 is the W32Time service. Windows NT does not have a native time service, but there are ones available.

You are going to need a server to act as the syslog collector. For a higher level of security you will want to use a dedicated machine. Regardless, the server you use needs to be as secure as possible.

To monitor the syslog logs, you look at the console for a real time view. Most syslog collector programs allow you to filter the syslog messages. This allows you to view the data you want to see.

To create a report and drill down into the syslog logs, we will have to create something ourselves, or use available tools. If we create something ourselves, we have more control on the output and the way the report looks. If we use freely available tools, we will get a report that is usable but not as user friendly. Whichever on we choose, we will still be able to get the data we want from the logs.

Implementation

We have defined the requirements, now to come up with a plan to implement it. Based on our requirements, we should first synchronize the time on all devices that will be communicating with the syslog collector. Then build and secure the server that will have the syslog collector service on it. After that install and configure the syslog daemon service on the server. Next install and configure the syslog device service on the servers you need to collect event log messages from. Then you will need to be able to monitor and report on the data collected. The final task is to learn how to perform maintenance on the system.

Synchronizing the time

Syslog messages are written to the syslog log file and displayed in the order in which they are received. The windows event log message contains the

[1] Kiwi Syslog, Kiwi Enterprises See References
[2] NTSyslog, Sabernet See References

date and time it was created. So the time the event was created and the time the message was receive may be different depending on a number of factors. In my testing, on average, the syslog messages where received around 1-20 seconds after they where created. In order to be able to tell what time the actual event happened, you should stick with the time in the 'message' part of the syslog message. That time stamp was created in the Windows event log. Things like network traffic and the availability of the device can affect when the message is received. Since you should rely on the time on the server that is creating the message, all devices sending syslog messages to the collector must be synchronized.

If you use Windows NT 4, there is no native time synchronizing service built into the operating system. In that case you could use one of the many time synchronization programs available on the Internet. Microsoft recommends that you use the TimeServ[1] service. You can get the TimeServ service from the NT resource kit 3.5 or above. If have a Windows NT system and need help setting that up, look at this great article by Tao Zhou on Windows & .NET Magazine Network website[2].

Windows 2000 has a service called W32Time to handle time synchronization. W32Time is a fully compliant implementation of the Simple Network Time Protocol[3]. The ideal configuration for setting up time synchronization is to have one server act as a SNTP server. The perfect candidate for that is the same server you will use as your syslog collector. Microsoft recommends you use one of the U.S. Naval Observatory SNTP[4] servers to synchronize your SNTP server to.

To configure your server as a SNTP server, have it synchronize with an outside SNTP server. Then you need to point your devices to that server. To check your current SNTP configuration run this command at the windows 2000 command prompt: **C:\> net time /querysntp**. The program will return the SNTP server IP/name if configured, or a message stating that it is not configured to use a SNTP server. Run the **/setsntp** switch to change that setting so it knows of a legitimate SNTP server. That looks like this: **C:\> net time /setsntp:192.5.41.41** in your windows shell. In that example, we are making the w32time service look at 192.5.41.41 to synchronize its time. Then we set the W32Time service on our Windows 2000 servers to use the syslog server as their time source. That command would be: **C:\> net time /setsntp:<your syslog server ip>**. After that is completed, restart the W32Time service on your servers starting with the syslog server. Then check the event logs for error messages to determine if the time is synchronizing correctly.

[1] D. Hogarth TimeServ See References
[2] Tao Zhou http://www.win2000mag.com/Artides/Index.cfm?ArtideID=522
[3] D. Mills RFC1769 See References
[4] U.S. Naval SNTP Servers See References

Secure the Syslog Collector Daemon

The syslog collector server must be as secure as possible. Microsoft has a great tool called the "Microsoft Security Toolkit"[1] that helps Microsoft administrators "harden" servers. The only issue I have had is that it seems to work better when installed on a Windows 2000 server with service pack 2 or lower installed. You should run that tool before you apply service pack 3 and your hotfixes. If you haven't already, make sure that the security event log is turned on. I like to create baseline policy that I can apply to all my servers. Make sure that all event logs are at least 8 megabytes or greater and the logs are overwritten as needed. You should disable any unused services and run HFNETCHK[2] to make sure you are up to date with patches for the services you left running. Keep the server services to a minimum and secure the services that are left running. An anti-virus package is also another good step to making this server secure. Name the server something that would not make it appear to be a log server, make sure it fits in with your naming convention standards. If an attacker can easily see that a particular server is a log server then they will probably attempt to make the log server incapable of receiving logs from the server they are attacking, or possibly they may attack the log server so they can modify or destroy the logs. For more information on securing a Windows 2000 Server, check out the NSA recommendations and the Microsoft Windows 2000 Security Configuration Guide links in the references. Because syslog uses the UDP Protocol, which is connectionless by design there is an inherent flaw. There is no handshake between the two systems when the UDP datagram is sent and received. This allows any UDP datagram to be received by the collector, and if an attacker knew of your syslog collector's location they could send spoofed syslog messages and flood your logs. It is crucial to make sure that this server is as secure as possible.

Installing the Syslog Server

For our purposes we will use the freeware version of the Kiwi Syslog. Download the "Service Edition"[3] of the syslog software. The installer is pretty simple and does not give you many options. The default options can just be left alone. The only thing you may wish to change is the path if your install requires it (defaults to "C:\ProgramFiles\Syslogd").

To install the service, open the Service Manager and go to the Manage Menu->Install the syslogd service. Once that is completed (it takes a second or two) go back to the Manage Menu and Start the syslogd Service. A message should pop up in the Service Manager Display with a message saying the service was started. Now go into the properties page and under the DNS settings, configure it to 'Resolve the IP address of the sending device' and 'Resolve the IP address of the originating device'. Also check the 'Use a local DNS cache'. To send some test messages drill down the menu like this: Rules->Actions->Log to

[1] Microsoft Security Toolkit http://www.microsoft.com/security/kitinfo.asp
[2] Microsoft HFNETCHK, See References
[3] Kiwi Syslog Service, See References

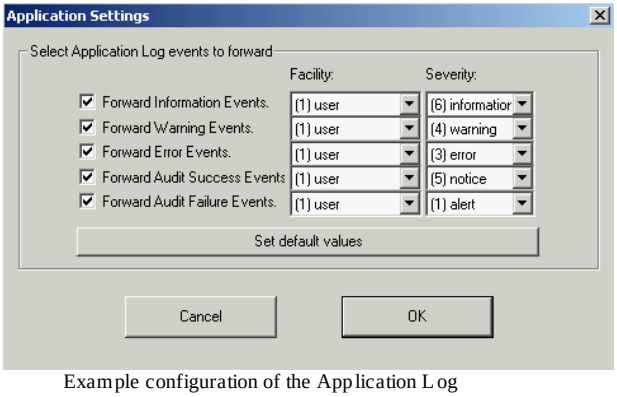
File and click the Test button. Do this to the Display properties to confirm that the Syslog Daemon is working correctly as well.

To archive and maintain the log files we need to create a new archive schedule. Right click on Archive in the properties menu, click 'add new archive schedule'. Configure the Archive Time to 'Daily' and make the destination folder match the source folder. Set the filename to use dated filenames. Keeping the backed up syslog files in the same folder makes our job of reporting from them easier later on, and helps in the backup process.

Send Messages to the Collector

To install the NTSyslog service, extract the NTSyslog.exe and the NTSyslogCtrl.exe files from the zip to your **C:\WINNT\SYSTEM32** directory. Then go to your command prompt in the windows shell and type **C:\>ntsyslog –install**. Now the service is installed. SaberNet recommends that you run the service as a local user and grant that use rights to log in as a service as well as give it the ability to manage and audit the security log. That is the correct way to configure the service, but I found that the software would keep on sending the event log messages over and over. So I left it using the local system account. But that bug may be fixed, or it may even work without any problems on your system. So to make the change, first create a local user account and give it a strong password. To make the policy changes, open up the mmc (Microsoft Management Console) and add the Security Configuration and Analysis snap in. Open your database (or make a new one) and 'Analyze Computer Now'. Then go into the Local policies and then into User Rights Assignments. Now, add your local user account to the 'log on as service' policy and 'manage and audit the security log'. Right click on 'Security configuration and analysis' and configure the computer. Now go into the NTSyslog service properties and set it to log on as the local account.

The next step is to configure the software. The NTSyslogCtrl.exe program is a configuration utility that allows us not only to configure how the messages are sent, but it also allows us to administer remote systems that are running the NTSyslog service. In order to help with the monitoring process, we will want to send our event log messages with the right facility and priority. That way we can organize the way we view the messages later. I use the same settings for all logs, and keep in mind that each log has individual settings, so do not forget to configure



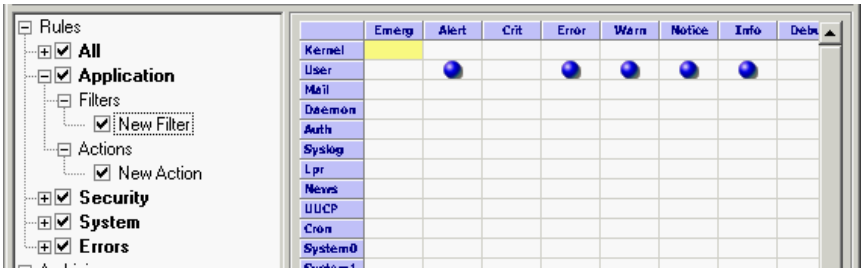
Example configuration of the Application Log

them all. Enable all events, and set the facility as follows, for the Application log use the User facility, for the System log use the System facility, and for the Security log, use the Security/Auth facility. Setting the Severity (or priority) is just as simple. Set the Information events to use the Information *Severity,* set Warning to Warning, set Error to Error and for the Success and Failure audits, set them to Notice and Alert respectively. The final step in configuring NTSyslog is to point it to the syslog collector, click on the 'Syslog Daemon' button and enter the IP of your syslog collector. If you are installing it on the syslog collector, give it a loop back address like 127.0.0.1. Keep a copy of NTSyslogCtrl on the syslog server so you can easily manage the remote syslog devices. Finally start the NTSyslog service and ensure it is running correctly by monitoring the syslog console display for new messages.

Monitoring and Reporting

Monitoring

Now that the syslog services are sending the Event Log messages to our syslog daemon we are ready to start monitoring the logs. You can use the Syslog Service Manager program to monitor the syslog in real time. When you open the program, the real time display of your syslog will appear. The default display is display 01 and it is configured to display all syslog messages. Since we configured the different logs to send messages using different facilities and priorities, it will be simple to configure different syslog message displays. Open the configuration screen by hitting CTRL+P. Now we need to create some rules so that we can send the messages to the appropriate display window. In the following screen shot, we can see what the filter rule is for our application log. The formats for the other logs are similar. Right click on rules and create a New Rule. Then name it the name of one of your logs, for example, 'Application' for the application log. Then right click on filter and add a filter. Since we know that our NTSyslog service is sending different priorities, on the field drop down, choose priorities. Now you will see the grid for the filter configuration. The priorities we are sending are Alert, Error, Warning, Notice and Info. So those are the ones we select. Each of the log types uses a different facility, Application is using User, Security is using Security/Auth, and System is using System (or Daemon like Kiwi Syslog likes to call it). So in our example we want to see the 'Application log', we would select the priorities in the appropriate facility. Confused yet? Check the image below for a good example on the filter for the Application log. Now that we have the filter, we need to send that data to a display. To do that, we need to



create a new action. So right click on Action and add an action. Select Display in the drop down menu and choose a display to send it to (it easiest to send it to 01, 02, etc…). Now just re-do those steps for the remainder of the logs. You could also create custom filters to suit your needs, for such things as a display with only errors or failure audits. You should also rename the displays from 01, 02, etc. to the name of the log you are monitoring. Do that by clicking on the 'Display' word in the configuration screen and modify and update them in the settings window. Configuration is done and now monitoring your syslog logs will easily become second nature.

Syslog stores its log data in a flat text file. One event is included on each line. A typical syslog message will contain the Data, Time, Priority, Hostname and Message. The actual Event Log messages data is all in the message portion. The sample message below is color coded so you can see the actual formatting of the message.

```
Date        Time        Facility.Priority  Hostname    Message
2003-01-17 12:46:49     User.Info          computer    Jan 17 12:46:31 kiwi syslog
daemon[info] 105  The service was started.
```

Reporting

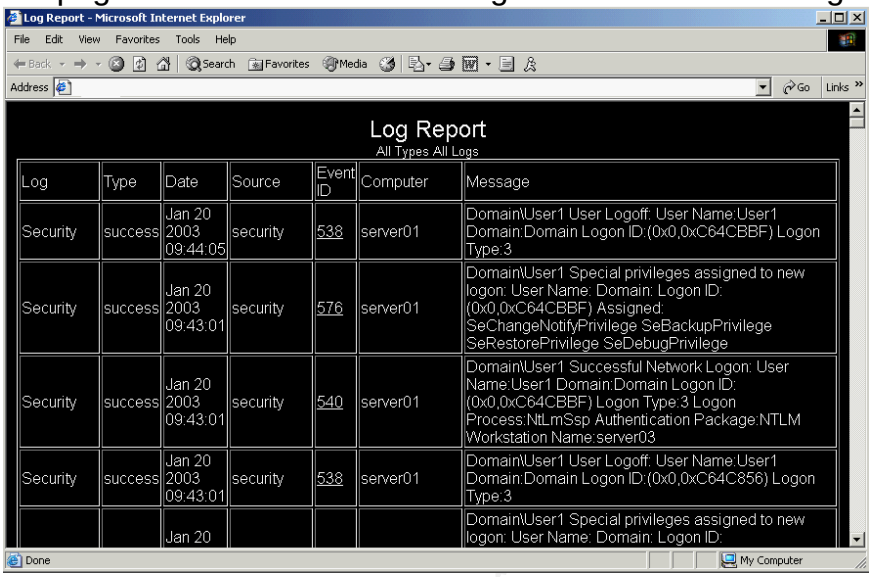
You can see that to create a report you have a lot of data to work with. One great tool you can use is a Unix utility called 'grep'[1]. Grep does not come with windows, but you can find many grep utilities on the Internet. What grep does is parse a file (or files), and display all the lines that match the string you are searching for. For example if you wanted to parse the syslog log files for all events that happened on January 9th, 2003 and send the results to a text file you would run grep like this while you where in the same path as the log files: "**grep -S "2003-01-09" *.txt >report.txt"**. The exact syntax may vary from utility to utility so check the documentation first. Most grep programs follow the same format and require you to pass a filename and a string to search.

Another way to create reports from our syslog data is to write a program to create a custom report for us. Perl is the perfect candidate for that. PERL or 'Practical Extraction and Reporting Language' has all of the functions we need to create a report. I cannot teach you how to program Perl, but what I will do is include a Perl script that I have written to generate reports specifically for this project. While it is not pretty because I am an amateur coder, it is functional. You are free to use and distribute the script and if you know how to program Perl you could customize it or use it as a guide for your own script. On the Windows platform, ActiveState makes a Windows distribution or Perl, called ActivePerl[2]. Install ActivePerl before running the script.

The attached Perl script (Appendix A) will search the syslog log folder and analyze the logs based on the criteria you give it and output the data to a HTML report. There are three things the script accepts when you do a search. The first tells the script what type of message to report on, Information, Warning, Error,

[1] Grep See References
[2] ActiveState – ActivePerl See References

Success Audit, Failure Audit or All message types. The second switch tells the script where to look for those messages, Application log, Security log, System log or All logs. The last thing the script looks for is a string it will search for. The report the script generates looks like the image below. One nice thing the script



does that makes it more functional it links all of the Event Id's to www.eventid.net. The web site provides description of the event and possible solutions to fix it if it is a problem. The only configuration that must be done for the script to work is to set $path variable, you need to change it so it points to the correct path of your log files. The command to run the script would similar to this: **C:\>perl sysrep.pl –a –sec username**. To see the usage just run the script without any arguments. Copy the sysrep.pl script to a Reports folder, if it is in the same location as the log files it may have problems.

Now that we can create some basic reports, there are a few things to focus on. Using grep or the Perl script to search for a particular username or event id number and the program will return all the log entries with that specific string. For a detailed list of Security event log id numbers look at the Microsoft knowledge base article Q174074[1]. Using those tools and bit of imagination you can create reports for almost any auditing purpose.

Maintenance

Maintenance is a small but important aspect of this system. In general there is not much you need to do to maintain it. Having verified good backups of the syslog log files is a must. Set up a retention policy so you only keep a couple of months of data on the server and have the older data archived and stored off site to keep safe. Other than watching the logs, configuring new syslog devices when needed, and doing routine server maintenance there is not much else.

[1] Q174074 See References

Conclusion

Building, monitoring and maintaining a centralized logging system is not an impossible task, even on a budget. Utilizing some of the best free tools available, combined with a little bit of time and motivation produces a fully functional auditing and tracking tool. Being able to run reports on your servers and audit security logs will help you proactively do your job affectively, and help you understand your network that much more. Monitoring your system's logs on a regular basis will make you a more informed and allow you to make good decisions based on what you know about your Windows network.

Glossary

Defense In Depth - "Defense in depth is the concept of protecting a computer network with a series of defensive mechanisms such that if one mechanism fails, another will already be in place to thwart an attack. Because there are so many potential attackers with such a wide variety of attack methods available, there is no single method for successfully protecting a computer network." Todd McGuiness, November 11, 2001

UDP Protocol – "UDP, documented in RFC 768, provides users access to IP-like services. UDP packets are delivered just like IP packets - connection-less datagrams that may be discarded before reaching their targets. UDP is useful when TCP would be too complex, too slow, or just unnecessary." *UDP Protocol Overview* http://www.freesoft.org/CIE/Topics/85.htm

References

The references are listed in the order they appeared.

C. Lonvick "The BSD syslog Protocol RFC3164 August 2001
http://www.ietf.org/rfc/rfc3164.txt

New & Rose "Reliable Delivery for syslog" RFC3195 November 2001
http://www.ietf.org/rfc/rfc3195.txt

'Kiwi Syslog Daemon Service Edition' Kiwi Enterprises
http://www.kiwisyslog.com/software_downloads.htm#syslog

NTSyslog, SaberNet
http://sourceforge.net/projects/ntsyslog

'TimeServ 1.55 Time Service' Microsoft Corporation Douglas W. Hogarth, 1995-1998
http://www.niceties.com/timeserv.htm

'Basic Operation of the Windows Time Service' Microsoft Knowledge Base Article KB224799, 10-10-2002
http://support.microsoft.com/default.aspx?scid=kb;en-us;224799

D. Mills "Simple Network Time Protocol (SNTP)" RFC1769 March 1995
http://www.ietf.org/rfc/rfc1769.txt

U.S. Naval Observatory SNTP Servers:
ntp2.usno.navy.mil 192.5.41.209
tick.usno.navy.mil 192.4.41.40
tock.usno.navy.mil 192.5.41.41

HFNETCHK developed for Microsoft by Shavlik Technologies LLC
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/tools/hfnetchk.asp

NSA Security recommendations for Windows 2000
http://nsa1.www.conxion.com/win2k/download.htm

'Microsoft Windows 2000 Security Configuration Guide' Microsoft Technet, 2003
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/issues/W2kCCSCG/W2kSCGca.asp

'Windows 2000 Default Security Policy Settings' Microsoft Technet, 2003
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/issues/W2kCCSCG/W2kSCGca.asp

'Grep for Windows' tcharron@interlog.com, January 26, 2001
http://www.interlog.com/~tcharron/grep.html

ActiveState – ActivePerl The industry-standard Perl distribution for Linux, Solaris, and Windows
http://www.activestate.com/Products/ActivePerl

'Security Event Descriptions' Microsoft Knowledge Base KB174074, 8-9-2001
http://support.microsoft.com/default.aspx?scid=kb%3Ben-us%3B174074

Fred DeFrance "A Case for Centralized Logging" Dec 7, 2001
http://ebuzzsaw.com/whitePapers/Case_for_Centralize_Logging.htm

Appendix A

```
# SYSlog REPorter
# For use with Kiwi Syslog and Ntsyslog
# as outlined in 'Centralized Logging in a Windows Environment'
# GSEC Practical Assignment
# ver 0.3
#
# Filomeno Iturzaeta
```

```
# rconlives@yahoo.com
#
# The Kiwi syslog should be configured to archive the logs once a day and name the
# log using the date. The archive files and the current logs should be saved in the
# same folder. This script looks in the path you supply for the syslog logs. Then it
# opens all the logs and puts them into one array. The array is then searched for certain
# criteria and then the results are put into a HTML report.
#
# Usage:
#
# perl sysrep.pl -a|i|w|e|s|f -all|app|sec|sys <search_string>
#
# -a searches All messages
# -i searches the Information messages
# -w searches the Warning messages
# -e searches the Error messages
# -s searches for Success Audits (Security log only)
# -f searches for Failure Audits (Security log only)
#
# -all searches All logs
# -app searches the Application log
# -sec searches the Security log
# -sys searches the System log
#

# Point it to the location of the Kiwi Syslog log file folder
$path = 'C:\Program Files\Syslogd\Logs';
# Change this for a custom title
$title = 'Log Report';

# Setting up some variables
$type = $ARGV[0]; chomp $type;
$logtype = $ARGV[1]; chomp $logtype;
$string = "$ARGV[2]"; chomp $string;

if ($ARGV[1] eq '') {
print <<USEAGE;
Usage:

perl sysrep.pl -a|i|w|e|s|f -all|app|sec|sys <search_string>

-a searches All messages
-i searches the Information messages
-w searches the Warning messages
-e searches the Error messages
-s searches for Success Audits (Security log only)
-f searches for Failure Audits (Security log only)

-all searches All logs
-app searches the Application log
-sec searches the Security log
-sys searches the System log
USEAGE

exit;
}

# Get all files in the path
opendir (DIR, $path); @dirlist=readdir(DIR); closedir DIR;
@dirlist = sort {$b cmp $a} @dirlist;
pop @dirlist; pop @dirlist;

# Put all logs into one big array
foreach $file (@dirlist) {
        $log="$path/$file";
        open (SCAN, $log); @scan = <SCAN>; close SCAN;
        @logs = (@scan, @logs);
}

# Get type
```

```
if ($type eq '-i') { $searchtype = 'Info';
} elsif ($type eq '-w') { $searchtype = 'Warning';
} elsif ($type eq '-e') { $searchtype = 'Error';
} elsif ($type eq '-s') { $searchtype = 'Notice';
} elsif ($type eq '-f') { $searchtype = 'Alert';
} else { $typeall = 'All Types'; $searchtype = ''; }

# Get log
if ($logtype eq '-app') {
        $searchlog = 'User';
} elsif ($logtype eq '-sec') {
        $searchlog = 'Auth';
} elsif ($logtype eq '-sys') {
        $searchlog = 'Daemon';
} else {
        $logall = 'All Logs';
        $searchlog = '';
}

foreach $scan(@logs){
        if ($scan =~ /$searchlog/) {
                if ($scan =~ /$searchtype/) {
                        if ($scan =~ /$string/i) {
                                chomp $scan;
                                push (@lines, $scan);
                        }
                }
        }
}
@lines = sort {$b cmp $a} @lines;
open (REPORT, ">report.html");
print REPORT <<HTML;
<html>
<title>$title</title>
<bodyBGCOLOR=000000 ALINK=ffffff VLINK=ffffff LINK=ffffff TEXT=ffffff>
<font face=Arial><head><center><font size=5>$title<br></head><font size=2>
$typeall$searchtype $logall$searchlog $string<br>
 <table border=1> <tr> <td>
   Log </td> <td>
   Type </td> <td>
   Date </td> <td>
   Source </td> <td>
   Event ID </td> <td>
   Computer </td> <td>
   Message </td> </tr>
HTML

foreach $line (@lines) {
        # Below is the regex engine to get the event log message parts and
        # set them up so we can report on them
        @split = split /\t/, $line;
        @getlog= split /\./, $split[1];
        $log = $getlog[0];
        if ($log eq 'User') { $log = 'Application';
        } elsif ($log eq 'Auth') { $log = 'Security';
        } elsif ($log eq 'Daemon') { $log = 'System'; }
        $hostname=$split[2];
        @message = split /\s/, $split[3];
        @getyear= split /-/, $split[0];
        $time="$message[0] $message[1] $getyear[0] $message[2]";
        shift @message; shift @message; shift @message;
        $joined = join(' ', @message);
        @message = split /]/, $joined;
        @getsource = split /\[/, $message[0];
        $type=$getsource[1];
        $source=$getsource[0];
        @getnumber = split /\s/, $message[1];
        $eventid=$getnumber[1];
        shift @getnumber; shift @getnumber;
print REPORT <<HTML;
```

```
 <tr> <td>
   $log </td> <td>
HTML

$color = "ffffff";
if ($type eq warning) { $color = "ffff00";
} elsif ($type eq error) { $color = "ff0000";
} elsif ($type eq failure) { $color = "ff0000";
} elsif ($type eq success) { $color = "ffffff";
} elsif ($type eq info) { $color = "ffffff"; }
print REPORT "<font color=$color>\n";
print REPORT <<HTML;
   $type </td> <td>
   $time </td> <td>
   $source </td> <td>
   <a href=http://www.eventid.net/display.asp?eventid=$eventid&source=$source>$eventid</a> </td> <td>
   $hostname </td> <td>
   @getnumber </td> </tr>
HTML

}
print REPORT <<HTML;
 </table>
</center></body></html>
HTML

close REPORT;
exit;
```