Global Information Assurance Certification Paper

Centralized Logging using Logsentry in a Large UNIX Environment

Saleem Kazmi
September 23, 2002

Abstract

One of the significant issues in information security is how to effectively monitor system logs with the frequency necessary to detect security violations and unusual activity. If you have a small number of systems it is difficult but with thousands of systems in your environment, it is effectively impossible to check logs on each system on a regular basis without some level of automation. Integrity of the logs is another critical issue; one of the first things any attacker does when they take control of a system is to delete any logs that might aid the system administrator in investigating the intrusion. Wouldn't it be nice if, as a system administrator, when you come in the morning, all the logged events you have specified as interesting were waiting in your mailbox? And even if an attacker had deleted the logs on the compromised system you were still are able to investigate the incident? Centralized event logging provides these capabilities. I had an opportunity to implement centralized logging in a large heterogeneous Unix environment and I would like to share my experience with the readers.

In this paper, I will detail the process of implementing centralized logging using Logsentry a free tool from Psionic Technologies. Before I begin doing so, I will explain what centralized logging is and what the benefits of centralized logging are in general in a data center environment. After I have discussed the benefits, I will focus on centralized logging for intrusion detection. All the components of the centralized logging will be discussed in detail, including syslog, Logsentry and the configuration of both. As well, an estimate of the disk space required to keep the logs for a month, specific details of the configurations and other useful information that may save considerable time for any reader who is involved in implementing centralized logging is presented. I have also included samples for /etc/syslog.conf files for different vendors with the required changes as well as a sample of log file

Introduction to Centralized Logging

Centralized logging can be defined as collecting the systems logs for a group of systems in centralized location.[1] Event logging is typically done in every data center in either locally on each individual system or on a central logging server (sometimes both). The three main reasons to have event logging in place are: -

a. Troubleshooting the Problems after they occur
   System logs are frequently essential in troubleshooting system problems. Whenever things break down, whether it is; network connectivity, disk

[1] Note that this centralized location may in fact be multiple systems or locations depending on the level of redundancy required for other considerations

drives going bad, a CPU failure, system board errors, etc…system logs almost always make the task of diagnosing the problem much less difficult. The same holds true for application-level troubleshooting. All this information could be logged to just the local systems, but if your system is down or your are not at a location where you can reach it directly, having a copy of the logs at a different location can make the difference between a short downtime that users never notice and a longer one that is noted in your performance review. When I was a system administrator, it was not uncommon for the Informix DBA to change the kernel parameters of a system and the only way that we would be able to determine the cause of the slow performance or system crash was by reviewing the logs. Centralized logging is a potential solution to address these issues. [1] http://ebuzzsaw.com/whitePapers/Case_for_Centralize_Logging.htm

b. Preventing problems
Very often system downtimes due to hardware failures could have been prevented with the help of event logging. For example, many kinds of disk, CPU or system board errors are reported days, or weeks prior to the system finally failing. I have been involved in many cases where we were able to prevent potential downtime by monitoring these early warnings and changing the defective parts in our regularly scheduled downtime. Centralized logging can provide information to improve uptimes and aid proactive systems administration. [1]

c. Centralized Logging for intrusion detection
Very simply, you cannot detect an attack if you do not know who is logged on to your systems and what kind of activity is happening on your networks. Logs provide crucial details of what is occurring in your environment. They make it possible to determine which systems are being attacked and which systems have been compromised. Logging should occur and the output should be monitored on all systems. At the very least it must be done on all key systems. I would recommend configuring all systems to log information both locally and on a central log server. This design provides both redundancy and an extra layer of security. It will also help you where the integrity of the local logs may be suspect. For instance, when comparing logs on the local machine with the log server, if any differences are found it would indicate suspicious activity. In addition, having all the logs in a single repository makes correlation of logged events much easier. For example, one line in a log file on a single server may not be suspicious, but the same entry on 50 or 100 systems in a data center or across an organization within a minute of each other could indicate a major issue. Logs should be archived according to your company's security policy but if your policies don't specify a timeframe, a good rule of thumb is at least one year. As well, I would strongly recommend that you back up your logs on removable media and store them off site. This is another advantage of centralized logging, if you ever

need any logs to restore from a backed up media, it is easier to get those logs from one or two tapes from a single system rather that going through dozens of tapes from multiple systems. [1]

Last but not least, one of the most important security functions of logging is deterrence. Human psychology dictates that people will behave differently when they know their actions are being watched. Because of this you should seriously consider publicizing the fact that you have set up active logging in your environment. At the same time you may not want to publicize the fact that you are logging centrally to make it less likely that an attacker will attempt to sabotage that activity. The fear of getting caught may prevent some classes of intruders from attempting to hack the system. One effective way is to have a warning message in your "motd" (n message of the day) file o rlogin banner that says something like: "All activity is logged with your username, host name and IP address". [2]
Ross, Seth "UNIX System Security Tools" McGraw-Hill, 1999. 183 –189

Following are some of the advantages of implementing centralized logging. [3]
www.yale.edu/its/security/sysadmin

- Audit trail if the client machine is compromised
- Central place to run logcheck scripts
- Dedicated machine could be tuned for logchecking
- One disk could be dedicated to just logs
- Highly secure with no other services
- Packet filtered/firewalled to allow only approved machines
- Logs could be sent to any email address for daily analysis.
- Convenient backup and restore

Some disadvantages. [3]

- UDP is used which is "best effort" protocol (Messages could be lost)
- There could be some cross platform compatibility issues
- Single point of failure, if you have only one Central log server.

Three main components of Centralized logging using Logsentry

1. Logsentry (old name logchecker) software from [4] http://www.psionic.com/logsentry.html
2. Syslog
3. Syslog configuration file /etc/syslog.conf

Introduction to Logsentry (logchecker)

According to the author of the logchecker in README file, Logcheck is a software package that is designed to automatically run and check system log files for security violations and unusual activity. Logcheck uses a program called

logtail that remembers the last position it read from in a log file and will use this position on subsequent runs to process new information. All source code is available for review and the implementation has been kept simple to avoid problems. This package is a clone of the frequentcheck.sh script from the Trusted information Systems Gauntlet (tm) firewall Package. [4]

Logcheck can help you to manage the large volume of log data generated by hundreds or thousands of systems and weed out normal log information to give you a condensed look at the possible break-in attempts. Instead of having you to go through that enormous amount of log data, it screens your logs to determine which log messages represent possible attacks. It then could be configured to email a list of those messages to the system administrators or wherever you please. [11] http://docsrv.caldera.com:8457/en/OLsag/using-logcheck.html

Logcheck uses grep to look through the filter files and processes the logs. The filter files contain keywords, such as root, su, sendmail etc. There are four files that logcheck consults in order to determine which log events to ignore and to look for. By default they are found in /usr/local/etc. More details about these files are discussed in the later sections of this paper. [10]
www.astro.uiuc.edu/~r-dass/logcheck

The logcheck allows two methods of log file auditing.

1. It reports everything it is asked to look for via keywords
2. It also reports everything, you did not tell it to ignore, via keywords

This ensures that important messages are specifically brought to your attention via the keywords you look for and that important messages you may have overlooked are also reported by only ignoring items you tell it to. The logcheck configuration files come with some of the default keyword entries common to most of the UNIX system logs out there. As you gain experience monitoring system activity, you might be adding more keywords to these files. Most likely, you will be adding to the ignore file, so that some logs you think are not important to pay attention should be ignored. The author encourages checking the appropriate keywords are entered in the configuration files. [5] [11]
www.freeos.com/articles/3540
http://docsrv.caldera.com:8457/en/OLsag/using-logcheck.html

How to download logsentry

This free software could be downloaded from Psionic Technologies website. They offer a TriSentry suit that includes PortSentry, HostSentry and LogSentry. But for this project I was only interested in LogSentry. Following are the steps to download the software.

a To download LogSentry go to www.psionic.com
b Click on LogSentry, it will take you to “Download LogSentry” page.

c. Click on " Download LogSentry package."
d. If you are interested to join optional "Announce" mailing list, enter your email address and pick a password otherwise click to skip this step
e. On the next page again if you are interested in joining optional Sentry mailing list, type your email address and pick a password, otherwise click to skip this step.
f. You will be on the page where you will choose what exactly you want to download. The most recent version which I installed is LogSentry 1.1.1

Introduction to Syslog

Before syslog was introduced, processes were responsible for writing messages to log files on their own. Syslog facility was originally developed at the University of California Berkeley as a part of the sendmail program. Over the years syslog has been adapted for many logging functions and ported to many UNIX systems including on system V and Linux. When syslog appeared it relieved programmers of the need to open and update log files. They replaced the file creation and write commands with calls to syslog, and syslog took care of routing their messages for the proper files. Any program can log events via syslog. Syslog can log to the system console, and write to a file or a device or can send a message to a user. It can log events locally or to another host over a network. In our project, we will be using syslog to forward logs to a centralized logging server. [6]
http://swexpert.com/C4/SE.C4.JAN.98.pdf

The syslog facility is based on two key elements: /etc/syslogd (the daemon) and the /etc/syslog.conf configuration file. By convention most syslog messages are written to the messages file, which lives in either the /var/adm or /var/log directories, though they can be written anywhere. A typical syslog record cites the name of the generating program and a text message. To utilize syslog, syslogd is executed at startup and runs in the background. This daemon listens for log messages from three sources:

- /dev/log. A UNIX domain socket that receives messages generated by processes running on the local machine
- /dev/klog. A device that receives messages from the UNIX kernel
- Port 514 an Internet domain socket that receives syslog messages generated by other machines through UDP. [7]
  Atkins, Darek. "Internet Security Professional Reference" New Riders, 1997. 150 - 151

What is /etc/syslog.conf

When syslogd receives a message from any of these sources, it checks its configuration file /etc/syslog.conf for the appropriate destination of the message. A message can go to multiple destinations or it might be ignored, depending on

the corresponding entries in the configuration file. The syslog.conf file specifies the logging behavior of the syslogd program, which consults the configuration file when it starts up. The file consists of individual entries for different programs or message categories, each on its own line. For each message category, a selector field and an action field are presented. These fields are separated by tabs. The selector field specifies the types of messages (facilities) and priorities and the action field specifies the action to be taken if a message syslogd receives matches the selection criteria. Each selector is composed of a paired facility and priority (level). The facility can be auth, authpriv, cron, daemon, ftp, kern, lpr, mail, news, etc and level or priority can be one of emerg, alert, crit, err, warning, notice, info, and debug. [2]

"Unix System Security Tools" [2] describes the syslog facilities and priorities as follows

Name	**Facility**
Auth	The authorization system: login, su, getty, etc
Authpriv	Same as auth, but logged to a file readable only by selected individuals
Cron	The cron daemon
Daemon	Other system daemons, such as routed
ftp	The file transfer protocol daemons: ftpd, tftpd
kern	Messages generated by the kernel (as opposed to user processes)
lpr	The line printer spooling system: lpr, lpd
mail	the mail system
news	The network news system
syslog	Messages generated internally by syslogd
user	Messages generated by random user processes; default facility if none is specified.
Uucp	The uucp system
Local0…local7	Reserved for local use

Priority	**Meaning**
Emerg	A panic condition (normally broadcast to all the users)
Alert	A condition that should be corrected immediately, such as corrupted database
Crit	Critical conditions such as hard drive errors
Err	errors
Warning	warning messages
Notice	conditions that are not error conditions, might require special handling
Info	Informational messages
Debug	Messages that contain information normally of use only when debugging a program

[2] Ross, Seth "UNIX System Security Tools" McGraw-Hill, 1999. 183 –189

Following is a sample of Linux system's /etc/syslog.conf file, If you notice the entry "auth.info @loghost.company.com" that means all the authorization system messages that is login, su, etc (security related issues) will be loged to locally as well as they will be forwarded to a remote host, in our project a centralized logging server. In this example the centralized log server is loghost.company.com

Linux /etc/syslog.conf file after adding the loghost entry

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                 /dev/console

#logging auth messages to centralized logging servers
auth.info          @loghost.company.com

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none;cron.none          /var/log/messages

# The authpriv file has restricted access.
authpriv.*                          /var/log/secure

# Log all the mail messages in one place.
mail.*                          /var/log/maillog


# Log cron stuff
cron.*                          /var/log/cron

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                                 *
```

The spaces in these entries are not just any white spaces, but are the tab characters. Entries will not work correctly if spaces are used instead. The entry loghost will be replaced by the hostname you would be choosing for the logserver.

What is Next?

Now that we have some understanding of the main components of the Centralized logging. We could start implementing it, to do that I personally took the following steps.

1. **Identifying the Central logging system**.

   Since default configuration of logcheck has been tailored for Linux Red hat system, and you will not have to do any changes to the script, I will recommend using Linux system as the central logging host. Logchchecker is not CPU or memory intensive so any decent system for example a Pentium 4 with 256 MB of memory assuming that you are not doing anything else on that system. But of course you can use any system as a log sever as long as the system is reliable and up most of the time, and not too many people should be aware of the name or IP address of the system. The reason is obvious that to keep the integrity of the logs. I will discuss, how to harden the log server in the later section.

1. Identifying the Clients

Now we need to find out how many systems will be sending logs to the central logging host. There could be thousands of systems with different flavors of Unix including HP. Linux and Solaris. Once we know the number of clients that will be sending logs to the log server, we will be able to get an estimate of the disk space required to keep those logs. For that matter create a list of the clients, and confirm that syslogd is running on all the clients. You can find out if syslogd is running by executing the following command.

Ps –ef|grep syslogd

2. Estimating the Disk Space needed to gather logs on the Central logging Server

I came up with the required disk space numbers as I added more and more clients and then observed on weekly basis to calculate the approximate disk space. Surprisingly you do not need lots of disk space even for 2000 + systems. I started off with 100 systems, monitored them for a week, and then added more systems gradually. Following were the approximate sizes of the weekly messages files. It could vary depending on the number of users and the nature of the machines. All the 2500 systems were workstations

DISK SPACE ESTIMATE FOR UPTO 2500 SYSTEMS

Number of clients monitored	Required Disk space weekly (7 days)	Total space to keep 4 messages files at a time
100	6MB	24MB
1000	60MB	240MB
2500	175MB	700MB

I would suggest for this big of a log size your allocated disk space for /var/log should be around 1GB. Again it depends on the environment, this disk space estimate should be used as a general guideline.

3. Installing Logsentry

Once you have selected the log server with enough disk space, syslogd is running on the system, change the current directory to /usr/local (cd /usr/local) and download the tar file in this directory.

rw-r--r-- 1 root root 204800 Feb 6 2002 logsentry-1.1.1.tar

a. Untar the file with this command
   tar –xvf logsentry-1.1.1.tar

b. you should get the following
   drwxr-xr-x 4 1000 users 1024 Jul 29 08:22 logcheck-1.1.1

c. cd logcheck-1.1.1

d. If your loghost is a Linux system run the following commands as root
   make linux
   make install

All the scripts and configuration files are installed in /usr/local/etc, and logtail.c file will be in ./logcheck/src directory. You can look at those files (copied and pasted from the system), by running the following command.

cd to /usr/local/etc
ls –la

```
total 40
drwxr-xr-x    3 root    root     1024 Feb  6  2002 .
drwxr-xr-x   15 root    root     1024 Feb  6  2002 ..
-rw-------    1 root    root     1037 Feb  6  2002 logcheck.hacking
-rw-------    1 root    root     1172 Feb  6  2002 logcheck.ignore
-rwx------    1 root    root    10633 Feb  6  2002 logcheck.sh
-rw-------    1 root    root      407 Feb  6  2002 logcheck.violations
-rw-------    1 root    root       14 Feb  6  2002 logcheck.violations.ignore
```

Description of the scripts and configurations files

Logcheck.sh: The main script. This file controls all processing and looks at logfiles with simple grep commands. This file is executed on a timed basis from cron and reports findings to the sysadmins or to the mailing address you enter in this file. [5] www.freeos.com/articles/3540

Logtail: A custom executable that remembers the last position of a text file. This program is used by logcheck to parse out information from the last time the log was opened; this prevents reviewing old material twice. All log files will be processed with this program and will have a file named secure.offset put in the same directory. Where secure is the name of the log file checked. This file contains the decimal offset information for logtail to work. If you delete it, logtail will parse the file from the beginning again. Logcheck tracks the size and inode of a log file to enable it to tell when a log file has been rotated. If the inode of the log changes or the file size is smaller than the last run, logtail will restart the counter and parse the entire file. [5]

Logcheck.hacking: This file contains known active hacking attack messages. Matches of these keywords that are found in the log files fall in the category “Active System Attack Alerts”. The keywords found in this file represent two types of attacks, one is from Internet Security scanner and the other is illegal syntax found in address lines of email messages from sendmail program. [11]
http://docsrv.caldera.com:8457/en/OLsag/using-logcheck.html

Logcheck.violations: The file contains keywords of system events that are usually seen as negative, but are not known to be associated with particular attacks. This messages generated by the keywords in this file could be for instance, “failed login attempts”. Which often simply result from real users just typing a user name or password incorrectly. This file has the keywords like, reject, fail, shutdown, denied or debug. So you will get the logs, as system services are shutdown and requests for services are denied. [11]

Logcheck.violations.ignore: This file contains more complete sentences that actually have keywords from logcheck.violations. These keywords are normal and are not cause for concern but could cause a false alarm. An example of this is the word “refused” which is often reported by sendmail if a message cannot be delivered or can be a more serious security violation of a system attaching to illegal ports. DO NOT LEAVE THE FILE EMPTY; some versions of grep will assume that an EMPTY file means a wildcard and will ignore everything. [5] [10]

Logcheck.ignore: This file contains patterns that we should ignore if found in a log file. If you have repeated false alarms or want specific errors ignored, you should put them in here. [5]

4. Configuring the Log server

Preferably ssh should be used to access that system, all RPC daemons and other miscellaneous services should be turned off. The only data allowed to the machine should be UDP/Port 514. Following are the steps to make your log server more secure. These steps are not required to run logchecker, but to make log server as isolated and inaccessible as possible. [8]
http://linuxsecurity.com/feature_stories/remote_logserver-4.html

a. Install SSH on this system
   Install SSH (secured shell) on your log server, so that you could stop using telnet or ftp, instead you will be using ssh to logon to the system, which will encrypt your session. See the following URL for more details about installing SSH. [8]

b. Turn off all INETD services
The services including echo, discard, daytime, chargen, time, ftp, telnet (if you are able to install SSH), login, exec, talk, ntalk etc. should be turned off. The way to accomplish that is to comment out this services by putting "#" sign in the beginning of the line of that service. [8]

c. Disable all RPC services
Anything listed in the rc. directories that start with a capital "S" mean that the service will start at boot time. Disable these services by renaming the executable; the command would be "mv S11portmap s11portmap" This would disable portmapper from starting at boot time. It is recommended to do the same with all unneeded services in this directory. [8]

d. Disable Accounts
Remove all the accounts that are not being used. Since this system will be used as a log server only, we do not need login account for everyone on this system, just keep number of accounts on this system very limited. [8]

The above steps will take care to secure the log server.

e. Run the syslogd with –r option, so that log server will receive messages from the clients you will be configuring later. The option –r will enable the facility to receive messages from the network using an Internet domain socket with the syslog service. The default is to not receive any messages from the network.

```
XXXX 54> ps -ef|grep syslogd
root    19313     1  3 Jul29 ?        01:53:10 syslogd -r
XXXX  12739 11357  0 13:08 pts/0    00:00:00 grep syslogd
```

f. Also change logcheck.sh file so that I could receive alerts in my UNIX mail.

```
# Person to send log activity to.
SYSADMIN=myuserid@abc.company.com
```

5 Configuring the Clients

a. You will have to make changes to the syslog.conf file on each client. For a large environment with 2500 + systems it is not practical to change the file on each system one at a time. I started off with few systems and added 20 systems every day or so, up to 100 systems and made the changes manually but once I passed the testing phase I

used a program to propagate the changes simultaneously to a group of systems. Generally every environment has some script or a utility to make changes simultaneously to a group of systems for each platform. I would recommend you to read the man pages for each UNIX flavor before making any change to syslog.conf file because each platform has slightly different syslog.conf file. I have attached samples of syslog.conf files after making the required changes to the files. You can use those files as a guideline. Following are the few tips while configuring the syslog.conf file

- Use tabs only, no spaces
- Use:set list in vi to display control characters
- File needs to exist before logging will occur.

[3] www.yale.edu/its/security/sysadmin

b. Once you have changed the /etc/syslog.conf on all the systems you need to reload the configuration file by sending it a hang-up signal (kill –HUP pid). The process id (PID) of the current invocation is kept in /etc/syslog.pid.

To restart after configuration changes

```
#ps –ef|grep syslogd
   root   366     1  0  Aug 10  ?        0:10 /usr/sbin/syslogd
   root  24409 24343  0 17:38:50 ttyq6     0:00 grep syslogd

#kill –HUP 366
```

To simplify this effort, you can add the following lines to /etc//init.d/syslog before the line with "*)":

```
'reload')
  if [ -f /etc/syslog.pid ]; then
    syspid=`/usr/bin/cat /etc/syslog.pid`
    [ "$syspid" -gt 0 ] && kill -HUP $syspid
  fi
  ;;
```

Subsequently you only need to issue the command /et/init.d/syslog reload to get syslogd to reload its configuration

Note: I have not added the above program and I have not used it either in my environment. I found it in one of the SAN's papers in the Reading room and thought it might be useful for the readers. The URL to look at the paper is [9] http://rr.sans.org/unix/syslog.php

6 Schedule the logcheck to run via Cron

Once we are done with all the configurations and the required changes, we schedule the logcheck to be run via cron. Following would be the cron entry if you want to run it every hour.

```
#Logcheck runs every hour

    0 **** /usr/local/etc/logcheck.sh
```

This concludes the installation process, and you are ready to get the logs on your central logging system and also to receive those logs via email if you wanted to.

Following are the samples of the /etc/syslog.conf files for different flavors of UNIX after making the changes to the files. If you notice In HP-UX, I have added the line "**auth.info @loghost.company.com**" and in the Solaris file I have added the line "**auth.notice @loghost.company.com**".

Again it is important to keep in mind that we are using tab to separate the two fields.

HP-UX 11 (copied and pasted from an HP system)

```
#@(#) $Revision: 74.1 $
#
# syslogd configuration file.
#
# See syslogd(1M) for information about the format of this file.
#
auth.info              @loghost.company.com
mail.debug             /var/adm/syslog/mail.log
*.info;mail.none       /var/adm/syslog/syslog.log
*.alert                /dev/console
*.alert                root
*.emerg                *
```

Syslog.conf for Solaris 5.6 (copied and pasted from the Solaris system)

```
#ident  "@(#)syslog.conf        1.4     96/10/11 SMI"   /* SunOS 5.0 */
#
# Copyright (c) 1991-1993, by Sun Microsystems, Inc.
#
# syslog configuration file.
#
# This file is processed by m4 so be careful to quote (`') names
# that match m4 reserved words.  Also, within ifdef's, arguments
# containing commas must be quoted.
#
*.err;kern.notice;auth.notice                   /dev/console
*.err;kern.debug;daemon.notice;mail.crit        /var/adm/messages
```

```
*.alert;kern.err;daemon.err          operator
*.alert                              root
auth.notice                          @loghost.company.com

*.emerg                              *

# if a non-loghost machine chooses to have authentication messages
# sent to the loghost machine, un-comment out the following line:
#auth.notice          ifdef(`LOGHOST', /var/log/authlog, @loghost)

mail.debug            ifdef(`LOGHOST', /var/log/syslog, @loghost)
"syslog.conf" [Read only] 35 lines, 1019 characters
```

Following is a **sample of a log file**; I have replaced original hostnames and the userids.

```
Jul 29 14:43:51 xxhost su: 'su root' failed for abuserid on /dev/pts/10
Jul 29 14:44:02 adhost su: 'su root' succeeded for acuserid on /dev/pts/10
Jul 29 14:48:57 anhost login: ROOT LOGIN /dev/pts/83 FROM aad.company.com
Jul 29 14:38:21 xxhost last message repeated 2 times
Jul 29 14:49:49 adchost  last message repeated 1 time
Jul 29 14:51:03 abchost sshd2[399]: User userid's local password accepted.
Jul 29 14:51:03 cfdhost sshd2[399]: Password authentication for user userid
cepted.
Jul 29 14:57:38 hostname su: 'su userid' succeeded for userid on /dev/pts/7
Jul 29 15:23:36 hostname login: yp_all: failed to get server's name
Jul 29 15:26:05  hostname  login: yp_all: failed to get server's name
Jul 29 15:26:05 hostname login: yp_all: failed to get server's name
Jul 29 15:26:31 hostname login: ROOT LOGIN /dev/pts/8 FROM
hostname.company.com
Jul 29 16:03:12 hostname login: yp_all: failed to get server's name
Jul 29 16:58:15 hostname  su: 'su userid' succeeded for userid on /dev/pts/2
Jul 29 18:33:15 hostname su: 'su userid' succeeded for userid on /dev/pts/20
Jul 29 18:34:02 hostname su: 'su userid' succeeded for hostname on /dev/pts/4
Jul 29 18:52:12 hostname su: 'su userid succeeded for hostname on /dev/pts/5
Jul 29 18:58:46 hostname su: 'su userid' succeeded for hostname on /dev/pts/2
Jul 29 19:02:49 hostname  su: 'su userid' succeeded for hostname on /dev/pts/3
```

Conclusion

In this paper we discussed two aspects of logging, centralized event logging and the use of logchecker. The centralized design not only provides ease of log management but it also provides improved integrity of the logs, as they will be stored on a separate machine. The integrity of the logs is very critical to our ability to investigate security violations. Having multiple layers of security is the most effective method to stop intrusion; centralized logging is one of the layers. For reporting purposes, to your management or to vendor technical support, it would be very convenient to have

quick access to the logs at one place. Logchecker is the tool that makes log monitoring even more convenient by bringing important messages to your attention via the key words you look for, and ignoring the unwanted messages. It also could be configured to get those messages via email to your Unix mail as well your outlook mail (by having a .forward file in your home directory}. It is also suggested to publicize that you have an efficient event logging system set up so that intruders will be reluctant to intrude. Finally, these logs have no use unless someone reads them. So make sure that those logs are monitored on a regular basis. [1]

References

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

2. Ross, Seth "UNIX System Security Tools" McGraw-Hill, 1999. 183 –189

3. Coleman, John. "syslog and Abacus" Yale Library Systems. November 30, 1999. www.yale.edu/its/security/sysadmin

4. http://www.psionic.com/logsentry.html

5. Waren, Trevor. "Intrusion Detection Systems, Part 1V: Logcheck" www.freeos.com/articles/3540

6. Lee, Henry "Living with syslog". SunExpert Magazine January 1998: 34-37 http://swexpert.com/C4/SE.C4.JAN.98.pdf

7. Atkins, Darek. "Internet Security Professional Reference" New Riders, 1997. 150 – 151

8. Hines, Erick. "Complete Reference Guide to Creating a Remote Log Server" August 22, 2000 http://linuxsecurity.com/feature_stories/feature_story-64.html

9. Pitts, Donald. "Log Consolidation with syslog" December 23, 2000 http://rr.sans.org/unix/syslog.php

10. Dass, Rami. "logcheck" January 09, 2001 www.astro.uiuc.edu/~r-dass/logcheck

11. "Using Logcheck" Calera Open Linux System Administration Guide http://docsrv.caldera.com:8457/en/OLsag/using-logcheck.html