Global Information Assurance Certification Paper

Secure Central Logging in a Heterogeneous Environment

Phil Lyons

9.28.2004

GSEC Practical, Version 1.4b Option 1

Style Definition: RTF_Num 4 1: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 2: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 3: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 4: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 5: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 6: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 7: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 8: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 9: Font: (Default) StarSymbol, 9 pt

Style Definition: RTF_Num 4 10: Font: (Default) StarSymbol, 9 pt

Abstract

The purpose of this paper is to discuss central logging in a secure fashion. The logging this paper refers to are the event logs which are generated by operating systems including Windows NT, 2000, 2003, XP, and syslogs sent by *nix systems such as Linux. This paper specifically uses Red Hat Linux, though much of the work can be extended to other *nix systems as well. Our solutions will also be able to accept legacy, enencrypted syslog data which may be sent by software and devices such as Cisco routers, and snort Intrusion Detection software. Secure in this context refers to protection of the data in transit from the syslog agents to the central syslog server.

By the end of the paper, I will have described multiple means of setting up secure logging to a standard syslog log sets, for example /var/log/messages, as well as logging to a SQL database. By providing multiple ways of accomplishing the same task the reader should gain sufficient knowlege to implement a secure logging solution in their own Linux/Windows environment. Also, as one mixes and matches solution components, a better sense for how the different pieces function is achieved.

The problem

If we use logging to monitor our security infrastructure, but give away security information via cleartext logging, then we may be providing more security with one hand, while taking it away with the other hand. Please review the tcpdump snippit below.

```
tcpdump: listening on eth0
15:06:31.178566 10.5.2.5.514 > 10.5.2.230.514: [udp sum ok] udp 48 (DF) (ttl
64, id 0, len 76)
0x0000   4500 004c 0000 4000 4011 21ad 0a05 0205        E..L..@.@.!.....
0x0010   0a05 02e6 0202 0202 0038 58b3 3c36 3e6b        .........8X.<6>k
0x0020   6572 6e65 6c3a 2064 6576 6963 6520 6574        ernel:.device.et
0x0030   6830 2065 6e74 6572 6564 2070 726f 6d69        h0.entered.promi
0x0040   7363 756f 7573 206d 6f64 650a                  scuous.mode.
15:06:31.180007 10.5.2.5.514 > 10.5.2.230.514: [udp sum ok] udp 48 (DF) (ttl
64, id 0, len 76)
0x0000   4500 004c 0000 4000 4011 21ad 0a05 0205        E..L..@.@.!.....
0x0010   0a05 02e6 0202 0202 0038 58b3 3c36 3e6b        .........8X.<6>k
0x0020   6572 6e65 6c3a 2064 6576 6963 6520 6574        ernel:.device.et
0x0030   6830 2065 6e74 6572 6564 2070 726f 6d69        h0.entered.promi
0x0040   7363 756f 7573 206d 6f64 650a                  scuous.mode.
15:06:31.189921 10.5.2.5.514 > 10.5.2.230.514: [udp sum ok] udp 48 (DF) (ttl
64, id 0, len 76)
0x0000   4500 004c 0000 4000 4011 21ad 0a05 0205        E..L..@.@.!.....
0x0010   0a05 02e6 0202 0202 0038 58b3 3c36 3e6b        .........8X.<6>k
0x0020   6572 6e65 6c3a 2064 6576 6963 6520 6574        ernel:.device.et
0x0030   6830 2065 6e74 6572 6564 2070 726f 6d69        h0.entered.promi
0x0040   7363 756f 7573 206d 6f64 650a                  scuous.mode.
15:06:50.439760 10.5.2.5.514 > 10.5.2.230.514: [udp sum ok] udp 73 (DF) (ttl
64, id 0, len 101)
0x0000   4500 0065 0000 4000 4011 2194 0a05 0205        E..e..@.@.!.....
0x0010   0a05 02e6 0202 0202 0051 3b1d 3c33 383e        .........Q;.<38>
0x0020   7375 2870 616d 5f75 6e69 7829 5b37 3133        su(pam_unix)[713
0x0030   305d 3a20 7365 7373 696f 6e20 6f70 656e        0]:.session.open
0x0040   6564 2066 6f72 2075 7365 7220 6c79 6f6e        ed.for.user.lyon
0x0050   7320 6279 2070 6c79 6f6e 7328 7569 643d        s.by.plyons(uid=
0x0060   3530 3029 0a                                   500).
```

From the above output of a tcpdump view of a few syslog packets, we can glean the fact that the host sending the logs has eth0, that eth0 can go into promiscuous mode, and that there is a valid user name 'lyons' with a uid of 500 on host 10.5.2.5. We can also know that a central log server is at 10.5.2.254. These are all good items of information that don't need to be displayed.

Exposure to snooping of syslog traffic could occur during several scenarios. Possibilities include a corporate network spread across physically dispersed sites which use a central syslog server, and a "Network Administration Services company" remotely managing systems via Internet[1].

Several excellent papers at SANS describe centralized logging. This paper will extend the work of earlier SANS papers to cover setup of secure logging from Windows & *nix hosts to central syslog repositories.

Project Requirements

Requirements are simple for this project:

- No cost software solution (G.P.L. or freeware)
- SQL Database backend storage of log data.
- Keeping the existing syslog data files in place.

The reason for the no cost solution is partly my limited budget, and partly due to the success I have had in implementing free solutions as pilots and proof-of-concepts for potential commercial solutions. Once the value has been proven, many times a customer will opt to either pay for a commercial implementation, or sometimes the free solution will be extended to cover an enterprise. In any case, the project "got in the door" due to the low startup costs.

My reasoning in keeping the existing syslog message file structure intact is that in my experience, fewer changes to the existing environment result in less political resistance to the project, and less time is spent supporting the project.

"Nice to have" features of the solution include the ability to accept log data on UDP port 514, in addition to TCP ports used to tunnel data via stunnel.

Operating systems used for the testing and setup in this paper include:

- Red Hat 7.3
- Red Hat 9.0
- Fedora Core 1
- Fedora Core 2
- Windows NT
- Windows XP Pro
- Windows 2000
- Windows 2003

[1] URL: http://www.kiwisyslog.com/info_secure_tunnel.htm

As a side note, I find VMware to be indispensible in simulating a heterogenous network environment on a limited hardware budget. The VMware service enables running multiple operating systems on a single host. In my case, a Dell laptop was used. More information is available at http://www.vmware.com/.

Getting started
We'll look at two different architectures for our log servers: hosting on a Linux server, and hosting on a Windows server. We'll log to those servers from Linux, Windows, and a printer and router. We won't be able to encrypt data from such devices as printers, but as these are usually in the network, they will be included to show complete interoperability.

Linux Server Configuration
Before beginning work on your syslog host, follow best practices for a clean install. Critical tasks include time synchronization and host lockdown.

Note: When building the Linux server, you may want to hold off on installing the web server from new media, as the php script langauge project recommends build 1.3 of apache, and if you install a later OS such as Fedora 2, it will install 2.0. I had problems running 2.0 apache with the php scripts required by php-syslog-ng. These components will be discussed later in this paper.

Time Synchronization
If there is an incident, you will want to make sure that your servers are centrally synchronized. I schedule time updates through crontab. There are other methods for time synchronization available[2].

Example crontab session to setup time synchronization:

```
# crontab -e

0 8 * * *       /usr/sbin/ntpdate ntp-1.cso.uiuc.edu   > /dev/null 2>&1
0 14 * * *      /usr/sbin/ntpdate ntp-1.cso.uiuc.edu   > /dev/null 2>&1
```

In the above crontab I have ntpdate synchronize at 8:00 am and 2:00 pm. The server I am contacting is ntp-1.cso.uiuc.edu. Choose the best time server for your site.

Additional steps I would recommend include installation of a file integrity checker such as tripwire[3]. As this is to be a secure implementation I would also install a firewall on the syslog host to lock down critical ports and services. For this task, I use iptables[4] for my Linux hosts.

Software Selection

[2] http://www.eecis.udel.edu/~mills/ntp/html/index.html
[3] http://sourceforge.net/projects/tripwire/
4 http://www.iptables.org/

For the core logging environment, the software that I have selected is as follows:

A good quality SQL database which we can search for events as required. MySQL fits this role, as well as the project requirements very well. In addition, I have seen MySQL gathering momentum in the corporate marketplace at the various customer sites I visit on a regular basis.

The logging server which meets the requirements as stated in this paper's abstract is syslog-ng. syslog-ng is a freely available central syslog server which can write data to a SQL database and maintain the existing syslog data file structure. In addition, the syslog-ng server can listen on both UDP and TCP ports. This feature is helpful should you decide to use a TCP-oriented service like stunnel[5] to wrap your syslog data.

The Apache web server[6]. Apache's function is to host the syslog reports and display those reports upon request to the security administrator. The recommended release of Apache as of 8.13.2004 is 1.3. The Apache version 2.0 is not recommended for this project, as per the php scripting language project[7].

Syslog database queries through the web server are produced by php-syslog-ng.

Installation
There are several components to install, so a methodolgy will be employed where we install from the bottom-up.

1.MySQL Database[8]
2. syslog-ng[9]
3.apache[10]
4.PHP[11,12]
5.php-syslog-ng[13]

We will install the components in the same order as given for the downloads. It is considered best practice to check a download's md5 value. If you don't have such a utility available, there are several available for download.[14]

The MySQL install

5 http://www.stunnel.org/
6 www.apache.org/
7 http://us4.php.net/manual/en/faq.installation.php#faq.installation.apache2
8 http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.21-win.zip/from/pick#mirrors
9 http://www.balabit.com/products/syslog_ng/
10 http://httpd.apache.org/download.cgi
11 http://www.php.net/
12 http://www.rpmfind.net/
13 http://sourceforge.net/projects/php-syslog-ng/
14 http://www.fourmilab.ch/md5/

To install MySQL, download the rpms to a directory, then use the rpm command to install.

```
[plyons@plyons3 MySQL]$ ls -lrt
total 13632
-rw-rw-r--  1 plyons plyons 2722900 Sep 18 11:57 MySQL-client-4.0.18-
0.i386.rpm
-rw-rw-r--  1 plyons plyons  946826 Sep 18 11:57 MySQL-devel-4.0.18-
0.i386.rpm
-rw-rw-r--  1 plyons plyons 9986797 Sep 18 11:57 MySQL-server-4.0.18-
0.i386.rpm
-rw-rw-r--  1 plyons plyons  263206 Sep 18 11:57 MySQL-shared-4.0.18-
0.i386.rpm
[plyons@plyons3 MySQL]$ pwd
/download/MySQL
[plyons@plyons3 MySQL]$ su -
Password:
[root@plyons3 root]# cd /download/MySQL
[root@plyons3 MySQL]# rpm -Uvh *rpm
warning: MySQL-client-4.0.18-0.i386.rpm: V3 DSA signature: NOKEY, key ID
5072e1f5
Preparing...                ########################################### [
100%]
   1:MySQL-shared           ########################################### [
25%]
/sbin/ldconfig: File /usr/lib/libpng12.so.0.1.2.2.#prelink#.wRtJ3K is too
small, not checked.
   2:MySQL-client           ########################################### [
50%]
   3:MySQL-devel            ########################################### [
75%]
   4:MySQL-server           ###########################################
[100%]
Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables
040918 12:05:00  /usr/sbin/mysqld: Shutdown Complete


PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:
/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h plyons3.lyonsnet.npv password 'new-password'
See the manual for more instructions.

Please report any problems with the /usr/bin/mysqlbug script!

The latest information about MySQL is available on the web at
http://www.mysql.com
Support MySQL by buying support/licenses at https://order.mysql.com

[root@plyons3 MySQL]# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 4.0.18-standard
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+----------+
| Database |
+----------+
| mysql    |
| test     |
+----------+
2 rows in set (0.01 sec)

mysql>
```

With MySQL running, let's move on to the core syslog server. In this instance, it will be syslog-ng. syslog-ng's features are[15]:

- compatible with a wide variety of platforms,
- can be used in heavily firewalled environments,
- allows you to manage audit trails
- drop in replacement for syslog
- Distributed under GPL

Download syslog-ng[16] and the supporting library libol[17], then install. Downloads for this product are available via http. After download, you should see something similar to this output:

```
[plyons@plyons3 syslog-ng]$ ls -l
total 736
-rw-rw-r--  1 plyons plyons 344270 Sep 18 22:31 libol-0.3.14.tar.gz
-rw-rw-r--  1 plyons plyons 396947 Sep 18 22:32 syslog-ng-1.6.5.tar.gz
[plyons@plyons3 syslog-ng]$
```

You will need the tar xzvf, ./configure, make and make install process to install syslog-ng. Starting with libol, then building syslog-ng works fine.

We next need to configure syslog-ng to replace syslog. In addition, we will configure syslog-ng to log identically to the original syslog program. For example, we will have syslog-ng generate output to /var/log/messages as the standard syslog daemon would natively. To get started let's discover how are current syslog daemon is configured to run. Chkconfig is a utility available to Red Hat system administrators which can display which runlevels a service is active. In our case, we'll be checking on syslog.

```
[root@plyons3 root]# chkconfig --list  syslog
syslog          0:off   1:off   2:on    3:on    4:on    5:on    6:off
[root@plyons3 root]#
```

The above output shows that syslog currently will be active when the current host is at

15 http://www.balabit.com/products/syslog_ng/
16 http://www.balabit.com/downloads/syslog-ng/1.6/src/
17 http://www.balabit.com/downloads/syslog-ng/libol/0.3/

runlevels 2 (multi-user) 3 (network services started), 4 (custom runlevel), and 5 (X Windows instantiated). Since we will be configuring a different log server, we'll need to disable our existing syslog as follows:

```
[root@plyons3 root]# chkconfig --level 2345 syslog off
[root@plyons3 root]# chkconfig --list  syslog
syslog          0:off   1:off   2:off   3:off   4:off   5:off   6:off
[root@plyons3 root]#
```

To configure syslog-ng for startup at runlevels 2-5, we need a startup script. Go to the build directory structure /syslog-ng-1.6.5/contrib. Get the contributed script init.d.RedHat-7.3 This script worked for several releases of Red Hat Linux for me including Red Hat 9.0, Fedora Core 1 and Fedora Core 2.

Copy this file to the /etc/init.d directory, then rename it to syslog-ng. chmod it to be executable, then chkconfig the file to start at runlevels 2-5.

```
[root@plyons3 init.d]# chmod 755 syslog-ng
[root@plyons3 init.d]# chkconfig --add syslog-ng
[root@plyons3 init.d]# chkconfig --level 2345 syslog-ng on
```

The listed chkconfig steps have disabled the original syslog, and enabled our new syslog-ng to run at startup.

syslog-ng uses a configuration file named syslog-ng.conf to determine its behavior. By default, this is located at /usr/local/etc/syslog-ng. Fortunately, in the contrib directory of the extracted tar files there is a configuration file `syslog-ng.conf.RedHat` which will enable syslog-ng to act almost identically to the original syslog.

Manually create the syslog-ng directory in the /usr/local/etc directory. Copy the configuration file to the expected location:

```
[root@plyons3 contrib]# cp syslog-ng.conf.RedHat /usr/local/etc/syslog-
ng/syslog-ng.conf
```

Now it's a simple matter to configure MySQL logging, logging to the console, and standard syslog behavior with the syslog-ng configuration file. "...a syslog-ng.conf file consists of options{}, source{}, destination{}, filter{}, and log{} statements. Each of these statements may contain additional settings, usually delimited by semicolons.

...Statements are terminated by semicolons; whitespace is ignored and may therefore be used to enhance readibility."[18]

To enable logging to the console remove the comment (#) from the following line in syslog-ng.conf :

```
#log { source(s_sys); filter(f_filter1); destination(d_cons); };
```

To send data as per the behavior or syslog, make sure the syslog-ng.conf file has an

18 "O'Reilly - Building Secure Servers with Linux" pg. 336

entry similar to this:

```
source s_sys { pipe ("/proc/kmsg" log_prefix("kernel: ")); unix-stream
("/dev/log"); internal(); };
```

Add a simple log configuration:

```
log { source(s_sys); destination(d_mesg); };
```

Let's next add the ability to listen to incoming syslog messages from the network on UDP 514, the standard syslog port. Perhaps we will want to listen to a device such as a printer which can only send unencrypted syslog messages. If we deem this to be not too much risk, our ability to listen on the standard port will be helpful. The alternative is to make sure that all logging clients have the ability to send log data to non standard ports, possibly using TCP instead of UDP as well. Then set syslog-ng to listen to the non standard port/protocol pair.

For simple UDP support, add udp(); to our list of sources:

```
source s_sys { pipe ("/proc/kmsg" log_prefix("kernel: ")); udp(); unix-stream
("/dev/log"); internal(); };
```

By restarting syslog-ng, we will now see that port UDP 514 is in the listening state.

```
[root@plyons3 syslog-ng]# netstat -na |grep 514
udp        0      0 0.0.0.0:514             0.0.0.0:*
[root@plyons3 syslog-ng]#
```

Next, let's configure MySQL to be able to accept messages from syslog-ng. The first order is to create the database and table required by syslog-ng. Reference the document included in the contrib directory of the syslog-ng download named `syslog-ng.conf.doc`. You will see the following database section:

```
#  Writing to a MySQL database:
#
#  Assumes a table/database structure of:
#
#            CREATE DATABASE syslog;
#            USE syslog;
#
#            CREATE TABLE logs ( host varchar(32) default NULL,
#                                facility varchar(10) default NULL,
#                                priority varchar(10) default NULL,
#                                level varchar(10) default NULL,
#                                tag varchar(10) default NULL,
#                                date date default NULL,
#                                time time default NULL,
#                                program varchar(15) default NULL,
#                                msg text, seq int(10) unsigned NOT NULL
auto_increment,
#                                PRIMARY KEY (seq),
#                                KEY host (host),
#                                KEY seq (seq),
#                                KEY program (program),
#                                KEY time (time),
#                                KEY date (date),
#                                KEY priority (priority),
#                                KEY facility (facility))
#                                TYPE=MyISAM;
```

Run the above create scripts as per the documentation (removing the '# ' characters of course). E.g., create a text file with the above content. Then redirect the content into the mysql client.

```
[root@plyons3 db]# mysql -u root -p <makeDB.sql
Enter password:
```

Testing the results shows:

```
[root@plyons3 db]# mysql syslog -u root -p
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.0.18-standard

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show tables;
+------------------+
| Tables_in_syslog |
+------------------+
| logs             |
+------------------+
1 row in set (0.00 sec)

mysql> desc logs;
+----------+------------------+------+-----+---------+----------------+
| Field    | Type             | Null | Key | Default | Extra          |
+----------+------------------+------+-----+---------+----------------+
| host     | varchar(32)      | YES  | MUL | NULL    |                |
| facility | varchar(10)      | YES  | MUL | NULL    |                |
| priority | varchar(10)      | YES  | MUL | NULL    |                |
| level    | varchar(10)      | YES  |     | NULL    |                |
| tag      | varchar(10)      | YES  |     | NULL    |                |
| date     | date             | YES  | MUL | NULL    |                |
| time     | time             | YES  | MUL | NULL    |                |
| program  | varchar(15)      | YES  | MUL | NULL    |                |
| msg      | text             | YES  |     | NULL    |                |
| seq      | int(10) unsigned |      | PRI | NULL    | auto_increment |
+----------+------------------+------+-----+---------+----------------+
10 rows in set (0.01 sec)

mysql>
```

This is fun! Let's get some data from syslog-ng to MySQL. For better security, add a new user to your syslog database on MySQL. This user will be used to insert records from syslog-ng. MySQL syntax is outside the scope of this document, but a good discussion of adding a user may be found by either the mysql.org site or a google search[19].

For my sample implementation I created a user 'syslogu', which you will see throughout the examples.

19 http://groups.google.com/groups?q=g:thl3681460468d&dq=&hl=en&lr=&ie=UTF-8&selm=9se05d%24s3n%241%40host.talk.ru

To have syslog-ng forward data to MySQL, modify the syslog-ng.conf file by adding a pipe configuration section. This snippet was taken from the `syslog-ng.conf.doc` file included in the contrib section of the extracted syslog-ng build directory structure.

```
destination d_mysql {
    pipe("/tmp/mysql.pipe" template("INSERT INTO logs (host, facility,
priority, level, tag, date, time, program, msg)
VALUES('$HOST','$FACILITY','$PRIORITY','$LEVEL','$TAG','$YEAR-$MONTH-
$DAY','$HOUR:$MIN:$SEC','$PROGRAM','$MSG');\n") template-escape(yes)); };
```

Note that the 'template-escape' variable setting will configure syslog-ng to escape characters which might otherwise cause a SQL syntax error, such as an apostrophe. Consult the `syslog-ng.conf.doc` for more details.

The last missing piece required to move data from syslog-ng to MySQL is the actual pipe itself, named /tmp/mysql.pipe in the example above. There is a script page at syslog-ng web site which details how to create such a fifo pipe file[20]. As the superuser 'root', simply execute

```
mkfifo /tmp/mysql.pipe
```

After the pipe is created, copy the text from the scripts page which will pipe the output from our syslog server to the pipe. Make sure this is run at system startup, preferably begining at runlevel 2. The directive which pipes output looks like[21]:

```
mysql -u root --password=passwd syslog < /tmp/mysql.pipe
```

Towards the bottom of the configuration file after defining keywords such as s_sys and d_mesg in our example, tell syslog-ng to actually send data to our waiting fifo pipe. Do this by modifying the 'log' directive we had created earlier.

```
log {   source(s_sys);
        destination(d_mesg);
        destination(d_mysql);
};
```

Restarting syslog-ng will enable our changes to take effect.

Note that there are alternates to the pipe solution. For example, a program may be used instead of the pipe[22]. This sacrifices flexibility for stability but it is a viable option. Please consult the `syslog-ng.conf.doc` file for a discussion of the pros and cons of pipe vs. file vs. programmatic data insertion.

If all goes according to plan, you can generate a syslog message by running 'su' or a similar log-producing message to have a syslog message created, captured by syslog-ng, forwarded to MySQL, and written to your database. Test by running a SQL query

20 http://vermeer.org/display_doc.php?doc_id=1#scripts

21 http://vermeer.org/display_doc.php?doc_id=1

22 http://www.frasunek.com/sources/security/sqlsyslogd/

such as the following:

```
[root@plyons3 syslog-ng]# mysql -u syslogu syslog -p
Enter password:
---> cut for brevity
mysql> select count(*) from logs;
+----------+
| count(*) |
+----------+
|        5 |
+----------+
1 row in set (0.01 sec)

mysql> select * from logs;
+---------+----------+----------+--------+------+------------+----------+-----
---------+-------------------------------------------------------------------
--+-----+
| host    | facility | priority | level  | tag  | date       | time     |
program      | msg
| seq |
+---------+----------+----------+--------+------+------------+----------+-----
---------+-------------------------------------------------------------------
--+-----+
| plyons3 | syslog   | notice   | notice | 2d   | 2004-09-24 | 11:46:37 |
syslog-ng    | syslog-ng[2494]: syslog-ng version 1.6.5 starting
|   1 |
| plyons3 | local7   | notice   | notice | bd   | 2004-09-24 | 11:46:36 |
syslog-ng    | syslog-ng: syslog-ng startup succeeded
|   2 |
| plyons3 | authpriv | info     | info   | 56   | 2004-09-24 | 11:46:53 | su
| su: pam_succeed_if: requirement "uid < 100" not met by user "plyons" |   3 |
| plyons3 | auth     | info     | info   | 26   | 2004-09-24 | 11:46:53 |
su(pam_unix) | su(pam_unix)[2500]: session opened for user plyons by
plyons(uid=0)  |   4 |
| plyons3 | auth     | info     | info   | 26   | 2004-09-24 | 11:46:56 |
su(pam_unix) | su(pam_unix)[2500]: session closed for user plyons
|   5 |
+---------+----------+----------+--------+------+------------+----------+-----
---------+-------------------------------------------------------------------
--+-----+
5 rows in set (0.08 sec)
```

At this point our core infrastructure is in place. We have a functional syslog service running on our Linux host. Logs are being forwarded to /var/log/messages, and to a SQL database. In addition we are flexible: by simple modifications to the syslog-ng-conf file, we can listen on any number of configurable TCP or UDP ports.

A Web Interface

For those desiring more robust viewing capabilities, addition of the PHP and syslog-ng-php will provide such enhancements. I will not cover syslog-ng-php in-depth as this project is more concerned with safeguarding logs in transit from clients to server, however additional components required include:

- The apache web server
    - Usually installed during a Red Hat Linux server install, but if not, may be download from http:apache 1.3 is installed, not version 2 as mentioned earlier.

- php[23] the web scripting langauge module. May also be download from http://rpmfind.net
- php-syslog-ng[24] The actual frontend for viewing syslog-ng messages logged to MySQL in realtime[25] run on the apache server against our MySQL database.

Please note that during the extract of the php-syslog-ng, an `INSTALL` file is expanded which has some helpful hints for caching of queries. Make sure to edit the db_fns.php as per the instructions with the database to contact, username and password.

Data Privacy in Transit
We now can support remote syslog clients sending syslog messages. We can also display local syslog messages. How is it possible that we can support the remote syslog clients? Recall the entry in the `/usr/local/etc/syslog-ng/syslog.conf` file:

```
source s_sys { pipe ("/proc/kmsg" log_prefix("kernel: ")); udp(); unix-stream
("/dev/log"); internal(); };
```

The second source, upd(), sets up UDP port 514 to listen for syslog messages. However, all messages will be cleartext. This is not our objective. And in fact, we have two puzzles to solve.

1. Encrypting traffic from the client to the central log server
2. Transmitting the hostname of the system that sent us the log entry.

Issue number 2 arises as we will be building a tunnel from the log client to the log server. Those familiar with tunnelling through secure shell will recognize this issue. Your connects on a secure shell tunnel look as though you are connecting to the local host. For example, to use secure shell (ssh) to a remote web server, you would enter a URL such as :

```
http://localhost:9080
```

If you had previously port-forwarded the local port TCP 9080 to a remote machine, then you would have built a tunnel through ssh, and securely connected to the remote web server. The problem is most likely becoming apparent. In our case, syslog-ng, or syslog for that matter, will be logging all of our tunnelled connects as coming from localhost. We will look at this momentarily.

Our tunnel
Our first client-server tunnel configuration will be that of a Linux host to our Linux server. We could do this numerous ways.

Scenario #1
Keep syslog running on our clients. In fact, were it not for our requirement of logging to a database, we could simply keep syslog running on the central log server as well. To

23 http://www.php.net/
24 http://sourceforge.net/projects/php-syslog-ng/
25 http://sourceforge.net/projects/php-syslog-ng/

keep using syslog as a client which forwards log traffic, we need to encrypt UDP traffic. This can be accomplished using a UDP-encrytping service such as zebedee[26].

Scenario #2
Replace syslog on the Linux client machines with syslog-ng. Forward traffic via TCP or UDP. Enrypt with stunnel, ssh tunnels, or zebedee. Another option may be IPSec. IPSec is not explored in this paper, but would be a good area for further research.

As you can see, there are actually several ways of protecting our data in transit. Once we have spent some time configuring our environment, it will be apparent that with little additional effort we can mix and match solutions to meet our security requirements.

Scenario #1 details
Let's start by making sure we can move traffic from our Linux client machines to the central syslog-ng log server. In the client's /etc/sysconfig directory, make sure that the '`/etc/sysconfig/syslog`' configuration file looks as follows:

```
# Options to syslogd
# -m 0 disables 'MARK' messages.
# -r enables logging from remote machines
# -x disables DNS lookups on messages recieved with -r
# See syslogd(8) for more details
SYSLOGD_OPTIONS="-m 0"
```

Espcially note the lack of the '-r' option in the SYSLOGD_OPTIONS.

This will keep the local syslog service from opening our local port UDP 514 for listening on startup. Thus, UDP port 514 will be available to us to port-foward to our remote logging host. With port 514 available, we can open up UDP 514 with another process such as zebedee, to forward our UDP traffic.

Our `/etc/syslog.conf` file is edited to include entries such as

```
*.info;mail.none;authpriv.none;cron.none                @localhost
```

This entry tells the syslog process to forward syslog messages to the localhost's UDP port 514. For this to work however, we need to construct the secure tunnel. Download the zebedee server from http://www.winton.org.uk/zebedee/download.html While not as robust as ssh or stunnel, Zebedee provides the benefits of being a simple secure tool, easy to setup, and able to forward UDP traffic as well as TCP.

Make sure to download the supporting libraries as well. Copy all to a directory, then extract at the same level in the directory structure. This will look something like:

```
[root@plyons3 tunnel]# tar xzvf zeb*gz

[root@plyons3 tunnel]# tar xzvf blowfish-0.9.5a.tar.gz; tar xzvf bzip2-
1.0.1.tar.gz; tar xzvf zlib-1.1.4.tar.gz
```

26 http://www.winton.org.uk/zebedee

If you decide to perform an optimized build ('make optimize') for the blowfish library, expect this process to take a long time. If this is a test implementation, you may want to consider just a plain 'make' to get things going, then do an optimized build when you can spare the CPU cycles.

To instantiate a tunnel between client and server, we will need both ends to be in place. Zebedee needs to run both on the client and on the server. Zebedee has a configuration file which we need to modify to properly setup our secure tunnels. In addition, we would like to ensure that the tunnel comes up as the machine is restarted. This is accomplished by building a script to start the tunnel, then using chkconfig to start the tunnel at the appropriate run levels.

Note that "The tunnel between Zebedee clients and servers still uses a TCP/IP connection even in UDP-mode."[27] The listening TCP port is dependent upon which mode the tunnel is set to forward. In our case, we will be listening for UDP traffic. When zebedee is listening for UDP traffic, the listening port will be TCP 11230.

Let's take a look at the server.zbd configuration file.

```
#
# Zebedee server configuration file

verbosity 2     # Slightly more than basic messages

message "ZEBEDEE CONFIGURATION FILE -- UDP SYSLOG TUNNEL"

detached true   # You will probably want this 'true' for normal
                # use but I want to make sure that you see the
                # preceding message if you haven't edited this.

server true     # Yes, it's a server!
udpmode true

compression zlib:9      # Allow maximum zlib compression
keylength 256           # Allow keys up to 256 bits
keylifetime 36000       # Shared keys last 10 hours
maxbufsize 16383        # Allow maximum possible buffer size
logfile SYSLOG

keygenlevel 2   # Generate maximum strength private keys

checksumlevel 3     # Allow maximum strength checksums
minchecksumlevel 0  # Allow no checksums if client requests
target localhost:514/udp

[plyons@plyons3 zebedee]$
```

Key points with regard to the server.zbd file displayed above. First, create a directory in the /usr/local/etc/ directory named zebedee

27 http://www.winton.org.uk/zebedee/manual.html

Copy this server.zbd file as modified above from the install package to this directory. In your options section of your zebedee startup script, point to this file. For example review this snippet from the zebedee script from /etc/init.d directory:

```
ZEBEDEE_OPTS='-s -U -f /usr/local/etc/zebedee/server.zbd'
```

Now on startup the zebedee tunnel can find its configuration information.

As per the last line in the configuration, the tunnel will forward traffic to UDP port 514 on the localhost. The syslog-ng server is listening on this port.

Important
The ramification of zebedee opening up a port other than UDP 514 to build the tunnel is probably obvious, but I will cover it anyway. We could disable remote access to our log server's UDP port 514 and use any port we desired for log transmission if we have no requirements to support legacy applications and/or clients such as HP printers, older routers, etc., which can only send data to UDP port 514.

When we are successful on the syslog-ng server host, we will see our netstat looking like the line below, once we have started zebedee.

```
netstat -nl |grep 11230
tcp        0      0 0.0.0.0:11230           0.0.0.0:*               LISTEN
```

Linux methodology is to use a startup script, which is kept in the `/etc/init.d` directory. As per the methodology we used for starting syslog-ng, the startup script will be configured to start via chkconfig to run at the proper runlevels, 2-5. Please reference appendix B for the startup script. After creating the startup script in the `/etc/init.d`, making it executable (`chmod 755 zebedee`), modifying the server.zbd file, and then copying it to `/usr/local/etc/zebedee` directory, we can start the zebedee secure tunnel.

```
[root@plyons init.d]# service zebedee start
Starting zebedee:                                          [  OK  ]
```

By way of review, our tasks have included

- Download zebedee
- Build zebedee, copy binary to /usr/local/sbin directory
- Edit server.zbd file, copy to /usr/local/etc directory
- Create zebedee init script in /etc/init.d (appendix B)
- Make zebedee script executable (`chmod 755 zebedee`)
- chkconfig zebedee startup script by –add and –level commands

```
# chkconfig –add zebedee
# chkconfig –level 345 zebedee
```

Now, on to the client.

zebedee client configuration

The same source code for zebedee can be used to build the client. In addition, I was able to use the same source code to build for both the Red Hat 7.3 and Red Hat Fedora Core 2 platforms with no problems.

We will follow the exact same instructions on the client host as we did to build the server.

1) Download zebedee
2) Build zebedee, copy binary to /usr/local/sbin directory
3) Edit server.zbd file, copy to /usr/local/etc directory
4) Create zebedee init script in /etc/init.d (appendix B)
5) chkconfig zebedee startup script by –add and –level commands

```
# chkconfig –add zebedee
# chkconfig –level 345 zebedee
```

Our only differences will be in the configuration file. Let's complete that configuration now.

Recall we modified a file called server.zbd. We'll do a similar configuration, except for the client end of the tunnel. The client configuration file is presented below. You may name the file as you desire. My selection was client.zbd.

```
#
# Zebedee config file to start up a tunnelled syslog service
#

verbosity 1     # Basic messages only
server false    # It's a client
detached true   # Detach from terminal
udpmode true

message "Starting zebedee client secure tunnel"
compression zlib:6      # Request normal Zlib compression
listenip 127.0.0.1
serverhost 10.5.2.38
tunnel 514:10.5.2.38:514
```

Important to note about this configuration file is 'udpmode true'. This will present fewer problems if this mode matches our server UDP mode. Also consider the last line of the configuration file. tunnel 514:10.5.2.38:514 tells zebedee to listen on the local port 514, connect to a server on 10.5.2.38, and send data to remote port 514. Remember that on the remote UDP port 514, we have our zebedee process ready to accept data.

We are ready to test. I execute the 'su' command on the client machine. To make sure we aren't transmitting data in cleartext, I run tcpdump as follows:

```
# tcpdump -nnvvXi eth0  not port 23
```

This will show verbose output without name resolution, in hex and ASCII (X), on interface eth0, excluding telnet traffic. (I am using telnet on the test networks – not the most secure of protocols, but this is an isolated VMware environment).

Output shows:

```
15:04:10.594660 IP (tos 0x0, ttl  64, id 17914, offset 0, flags [DF], proto 6,
length: 60) 10.5.2.100.1105 > 10.5.2.38.11230: S [tcp sum ok]
1210395460:1210395460(0) win 5840 <mss 1460,sackOK,timestamp 9334462
0,nop,wscale 0>
        0x0000:  4500 003c 45fa 4000 4006 dc2e 0a05 0264  E..<E.@.@......d
        0x0010:  0a05 0226 0451 2bde 4825 2b44 0000 0000  ...&.Q+.H%+D....
        0x0020:  a002 16d0 05bf 0000 0204 05b4 0402 080a  ................
        0x0030:  008e 6ebe 0000 0000 0103 0300            ..n.........
15:04:10.595586 IP (tos 0x0, ttl  64, id 0, offset 0, flags [DF], proto 6,
length: 60) 10.5.2.38.11230 > 10.5.2.100.1105: S [tcp sum ok]
4205480125:4205480125(0) ack 1210395461 win 5792 <mss 1460,sackOK,timestamp
112410179 9334462,nop,wscale 0>
        0x0000:  4500 003c 0000 4000 4006 2229 0a05 0226  E..<..@.@.")...&
        0x0010:  0a05 0264 2bde 0451 faaa 88bd 4825 2b45  ...d+..Q....H%+E
        0x0020:  a012 16a0 3d7f 0000 0204 05b4 0402 080a  ....=...........
        0x0030:  06b3 3e43 008e 6ebe 0103 0300            ..>C..n.....
15:04:10.596897 IP (tos 0x0, ttl  64, id 17915, offset 0, flags [DF], proto 6,
length: 52) 10.5.2.100.1105 > 10.5.2.38.11230: . [tcp sum ok] 1:1(0) ack 1 win
5840 <nop,nop,timestamp 9334463 112410179>
        0x0000:  4500 0034 45fb 4000 4006 dc35 0a05 0264  E..4E.@.@..5...d
        0x0010:  0a05 0226 0451 2bde 4825 2b45 faaa 88be  ...&.Q+.H%+E....
        0x0020:  8010 16d0 6c13 0000 0101 080a 008e 6ebf  ....l.........n.
        0x0030:  06b3 3e43                                ..>C
15:04:10.602780 IP (tos 0x0, ttl  64, id 17916, offset 0, flags [DF], proto 6,
length: 54) 10.5.2.100.1105 > 10.5.2.38.11230: P [tcp sum ok] 1:3(2) ack 1 win
5840 <nop,nop,timestamp 9334463 112410179>
        0x0000:  4500 0036 45fc 4000 4006 dc32 0a05 0264  E..6E.@.@..2...d
        0x0010:  0a05 0226 0451 2bde 4825 2b45 faaa 88be  ...&.Q+.H%+E....
        0x0020:  8018 16d0 6a07 0000 0101 080a 008e 6ebf  ....j.........n.
        0x0030:  06b3 3e43 0202                           ..>C..
15:04:10.603545 IP (tos 0x0, ttl  64, id 46842, offset 0, flags [DF], proto 6,
length: 52) 10.5.2.38.11230 > 10.5.2.100.1105: . [tcp sum ok] 1:1(0) ack 3 win
5792 <nop,nop,timestamp 112410186 9334463>
        0x0000:  4500 0034 b6fa 4000 4006 6b36 0a05 0226  E..4..@.@.k6...&
        0x0010:  0a05 0264 2bde 0451 faaa 88be 4825 2b47  ...d+..Q....H%+G
        0x0020:  8010 16a0 6c3a 0000 0101 080a 06b3 3e4a  ....l:.........>J
        0x0030:  008e 6ebf                                ..n.
15:04:10.645631 IP (tos 0x0, ttl  64, id 46843, offset 0, flags [DF], proto 6,
length: 54) 10.5.2.38.11230 > 10.5.2.100.1105: P [tcp sum ok] 1:3(2) ack 3 win
5792 <nop,nop,timestamp 112410229 9334463>
        0x0000:  4500 0036 b6fb 4000 4006 6b33 0a05 0226  E..6..@.@.k3...&
        0x0010:  0a05 0264 2bde 0451 faaa 88be 4825 2b47  ...d+..Q....H%+G
        0x0020:  8018 16a0 6a03 0000 0101 080a 06b3 3e75  ....j.........>u
        0x0030:  008e 6ebf 0202                           ..n...
```

So far, so good. All data is encrypted and no transmissions over UDP 514. (Tcpdump produced more output, the extra data was cut for brevity). However, a check of the MySQL database reveals the following records:

```
+-----------+----------+--------------+-------------------------------------------------------------
-----------------------+
| host      | facility | program      | msg
|
+-----------+----------+--------------+-------------------------------------------------------------
-----------------------+
| plyons3   | user     | zebedee      | zebedee: zebedee(4272/60368):   compression level 0x6,
key length 128          |
| 127.0.0.1 | local7   | zebedee      | zebedee: zebedee(10983/1024):  Starting zebedee client
secure tunnel
| 127.0.0.1 | local7   | zebedee      | zebedee: zebedee(10983/1024):  Listening on local port
514                     |
| 127.0.0.1 | local7   | zebedee      | zebedee: zebedee startup succeeded
|
| plyons3   | user     | zebedee      | zebedee: zebedee(4272/60368):   read 367 bytes (443
expanded) in 6 messages    |
```

The good news is data has been securely transmitted to our log server. The bad news is we don't really know where the data came from. Everything is coming from '127.0.0.1'. This is the second "puzzle" we needed to solve.

Possible solutions include either running zebedee in "transparent mode" or swapping out syslog with syslog-ng. Zebedee's transparent mode will allow UDP traffic to show the source IP address. Syslog-ng allows the syslog-ng client program to send the hostname in the syslog message.

Scenario #2 Details
Let's move on to scenario #2 introduced earlier on page 14. As a brief reminder, scenario #2 was described as "Replace syslog on the Linux client machines with syslog-ng. Forward traffic via TCP or UDP. Enrypt with stunnel, ssh tunnels, or zebedee. " We will again use zebedee for our tunnel, but swap out syslog for syslog-ng as the logging client.

This may sound more complex than it really is. syslog-ng as a client is a much simpler configuration than actually setting up syslog-ng as the server logging to a backend SQL database.

Our steps:

1. download and install syslog-ng on the client machine.
2. chkconfig syslog scripts to the 'off' state.
3. configure syslog-ng to forward the hostname.
4. chkconfig syslog-ng clients to similar runlevels as used by the legacy syslog install (Detailed earlier in this paper when configuring the syslog-ng log server).
5. stop syslog, start syslog-ng

As a side note, I was able to compile and run syslog-ng on both Red Hat 7.3 and Red Hat Fedora Core 2.

Please note that the above steps are identical to those described in this paper when configuring the syslog-ng log server. The only difference is in the `/usr/local/etc/syslog-ng/syslog-ng.conf` file on the machine running syslog-ng as a client:

```
options { sync (0);
          time_reopen (10);
          log_fifo_size (1000);
          long_hostnames (off);
          use_dns (no);
          use_fqdn (no);
          create_dirs (no);
          keep_hostname (yes);
        };

# Add Sources - perhaps add  tcp(port(4800) keep-alive(yes)
max_connections(100)); };

source s_sys { pipe ("/proc/kmsg" log_prefix("kernel: ")); unix-stream
("/dev/log"); internal(); };
```

```
# Add destinations
destination d_mesg { file("/var/log/messages"); };

# Our Server
destination net_server {
        udp("127.0.0.1" port(514));
};

# Log it.
log {   source(s_sys);
        destination(net_server);
        destination(d_mesg);
};
```

Stop syslog, start syslog-ng and test again. A quick SQL check of the database will show different output.

This command sequence will look like:

```
Service syslog stop
Service syslog-ng start
```

Logon to MySQL, then execute:

```
mysql> select count(*) from logs;
+----------+
| count(*) |
+----------+
|      644 |
+----------+
1 row in set (0.03 sec)

mysql> select host, facility, program, msg from logs where seq > 640;
mysql> Sep 25 16:53:24 plyons su(pam_unix)[20078]: session opened for user lisa by plyons(uid=0)
Sep 25 16:53:25 plyons su(pam_unix)[20078]: session closed for user lisa
select host, facility, program, msg from logs where seq > 640;
+---------+----------+--------------+-----------------------------------------------------------------
--------------------+
| host    | facility | program      | msg
|
+---------+----------+--------------+-----------------------------------------------------------------
--------------------+
| plyons3 | user     | zebedee      | zebedee: zebedee(4272/74352):  tunnel established to target
127.0.0.1, port 514 |
| plyons3 | user     | zebedee      | zebedee: zebedee(4272/74352):   compression level 0x6, key
length 128          |
| plyons  | syslog   | syslog-ng    | syslog-ng[20072]: syslog-ng version 1.6.5 starting
|
| plyons  | local7   | syslog-ng    | syslog-ng: syslog-ng startup succeeded
|
| plyons  | auth     | su(pam_unix) | su(pam_unix)[20078]: session opened for user lisa by
plyons(uid=0)             |
| plyons  | auth     | su(pam unix) | su(pam unix)[20078]: session closed for user lisa
|
+---------+----------+--------------+-----------------------------------------------------------------
--------------------+
6 rows in set (0.01 sec)

mysql>
```

Success! The host field in the logs table now has a hostname 'plyons' listed as the source of the syslog event. Our Linux environment is complete. However, we have Windows clients we must connect as well.

Windows Client Configuration

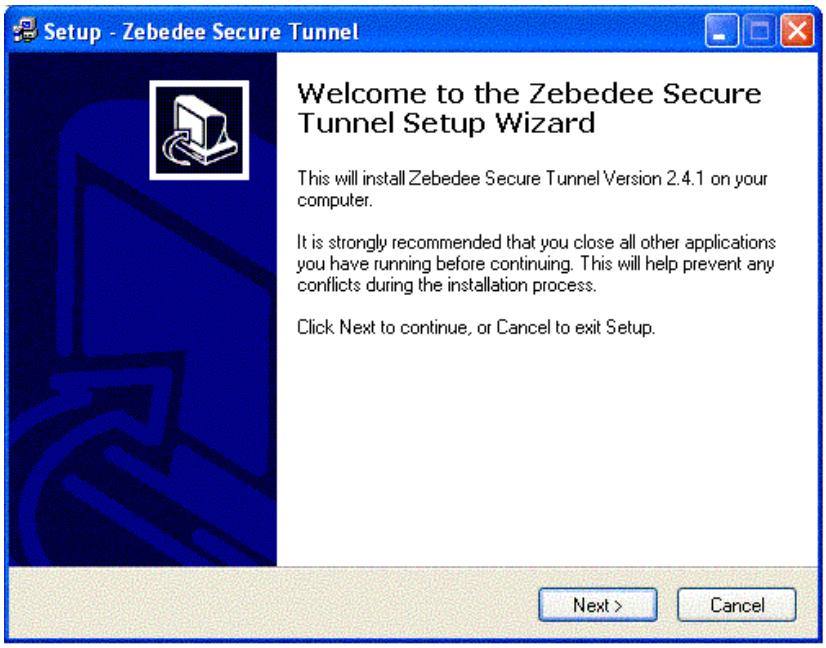
For the Windows client, I will again use the zebedee program to protect our data. One of the design goals of zebedee is to

* Provide full client and server functionality under both UNIX/Linux and Windows.

Zebedee is a good fit for our Windows installs, and maintains similiarities between our *nix and Windows platforms. This is a win for support requirements.

As with our Linux configuration, we will set up a UDP tunnel.

This tunnel will be connecting to the zebedee server process running on the central log server setup previously. The steps to install include downloading the executable and configuration. The Zebedee Secure Tunnel install is wizard-driven in the Windows environment.



In the zebedee install directory, edit the server.zbd file. For example, go to the 'Program Files\Zebedee' directory and edit server.zbd. For our purposes, we won't edit values for key generation. We will log to the "SYSLOG facility". Obviously Windows doesn't have a syslog facility per se, but as referred to by the server.zbd file this is the Windows event log. Comment out the “ipmode both” as we want only to tunnel UDP traffic. If you would like to test outside of the syslog architecture, I found a simple test environment is to encrypt a tunnel to a tftp server. This lets you discover whether your problem is syslog or communications related.

Once you are satisfied with the functionality of the Snare/zebedee combination, you will want to make use of zebedee's ability to run as a Windows service[28]:

```
Windows Service:
     zebedee [-n name] -S [install[=file] | remove | run]
```

After finishing the zebedee tunnel setup wizard, configure the .zbd file along the same lines as used by our Linux zebedee client tunnel configuration. Here is the client.zbd file I generated:

```
----->cut
verbosity 2 # Slightly more than basic messages

message "RUNNING FOR UDP CLIENT MODE"

detached true     # You will probably want this 'true' for normal
# server true     # Yes, it's a server!
server false      # no, actually it's a client.
ipmode udp  # Operate in UDP mode
compression zlib:9      # Allow maximum zlib compression
keylength 256           # Allow keys up to 256 bits
keylifetime 36000 # Shared keys last 10 hours
maxbufsize 16383  # Allow maximum possible buffer size
logfile SYSLOG
keygenlevel 2     # Generate maximum strength private keys
redirect none
serverhost 10.5.2.38
tunnel 514/udp:10.5.2.38:514/udp
----->cut
```

My syslog server is at IP address 10.5.2.38. UDP port 514 is our syslog port.

We will have similar issues to solve on the Windows platform that we solved on the Linux platform. Namely, our data is being tunnelled through an endpoint on the localhost. Data tunnelled in this fashion will report a hostname of locahost. This of course makes it difficult to see where data is originating from.

We next need a Windows event lot to syslog client to use for our our solution. There are other papers and resources available to help make this decision[29]. I chose Snare for its ease of installation and ability to forward hostnames in the syslog messages. Snare is distributed per the terms of the GNU General Public License.

Snare only transmits data over UDP, but we can handle this with zebedee's ability to secure and forward UDP traffic.

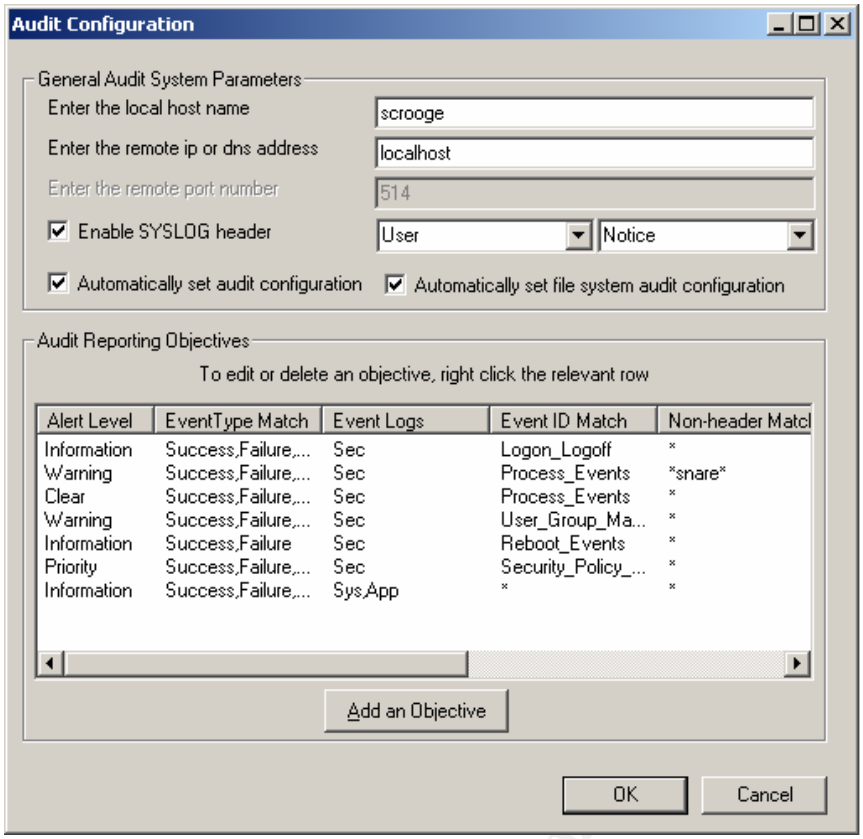
Snare also provides a wizard driven install. Configuration is simple and is almostly completely ready for use 'out-of-the-box'. Our main change will be to setup the 'remote ip or dns address'. Enter 'localhost' into this text box.

---

28 http://www.winton.org.uk/zebedee/manual.html

29 Central Logging with Open Source Software in a Unix/Windows Environment

After entering localhost as the 'remote ip or dns address', click ‘OK’, then choose the 'Activity' dropdown, then 'Apply and Restart' Audit.



We are ready to test our logging configuration from the Windows machine. Initiate a MySQL connection to our log server. Run the following select, paying particular attention to the last few entries:

```
mysql> select host, msg from logs;
| plyons3   | zebedee: zebedee(4272/60368):  compression level 0x9, key length 256

| scrooge   | MSWinEventLog      1       Application     2189    Sat Sep 25 18:06:27 2004        0
zebedee Unknown User    N/A     Information     SCROOGE Devices
zebedee(1132/1124):  Listening on local port 514  2 |
| scrooge   | MSWinEventLog      1       Application     2190    Sat Sep 25 18:06:35 2004
108     SNARE   Unknown User    N/A     Information     SCROOGE None            The service was
stopped.          3                       |
+----------+-------------------------------------------------------------------------------
-------------------------------------------------------------------------------+
24 rows in set (0.02 sec)

mysql>
```

It is a bit difficult to pick up the output from the SQL query. However, you can make out the first field in the SQL output correctly lists the host generating the syslog/eventlog message as scrooge, not localhost. We have now completed configuration of the

Windows syslog client.

We should be getting comfortable with the process of swapping components of the secure logging infrastructure. Let's change a bit more at this point, moving from a Linux based central logging server to a Windows-based log server. Our requirements will remain the same: i.e., free software components, database logging, and secure transmission of log data.

The Windows-based Central Log Server

In some environments, it may be necessary to install secure logging in a pure Microsoft Windows environment. This section discusses the configuration and setup steps involved. As with the rest of the central logging architectures discussed in this paper, the backend will consist of a MySQL database. At this point in time, the freeware Kiwi syslog server will not log to a SQL database. Free database logging is a requirement for this document. The commercial Kiwi products will provide both a secure tunnel[30], and the ability to log to an ODBC capable database[31]. Both are very nice features.

To review the overall Windows/syslog2ODBC architecture, you may consult appendix C.

We will use the Snare syslog client described earlier in this paper to get the syslog data from the logging client to the syslog server, tunnelling the log data in the same fashion as the previous Linux-based logging solutions. The central log service will wait on UDP port 514 for syslog messages, and forward those messages to the configured ODBC data source. I will use MySQL for the backend database. Mysql does not need to run on the same machine, but it can. As noted earlier, our UDP port 514 may be locked down to all external access, only available from the local tunnel endpoint if so desired.

The Windows syslog server configuration requires the following components:

- A backend SQL database for log storage.
- A syslog service which stores logs to the database. I have selected syslog2ODBC[32]
- The web server which hosts the web pages which display the log entries is apache 1.3[33].
- Web queries are supported by PHP[34]
- php-syslog-ng[35] provides the front end query logic.

Download

Start the process of software installation by first downloading all software as described above.

30 URL: http://www.kiwisyslog.com/info_secure_tunnel.htm
31 URL: http://www.kiwisyslog.com/products.htm#syslog
32 The download can be found at http://sourceforge.net/projects/syslog2odbc/
33 URL: http://www.apache.org/
34 URL: http://www.php.net/
35 URL: http://www.vermeer.org/display_project.php?project=php-syslog-ng

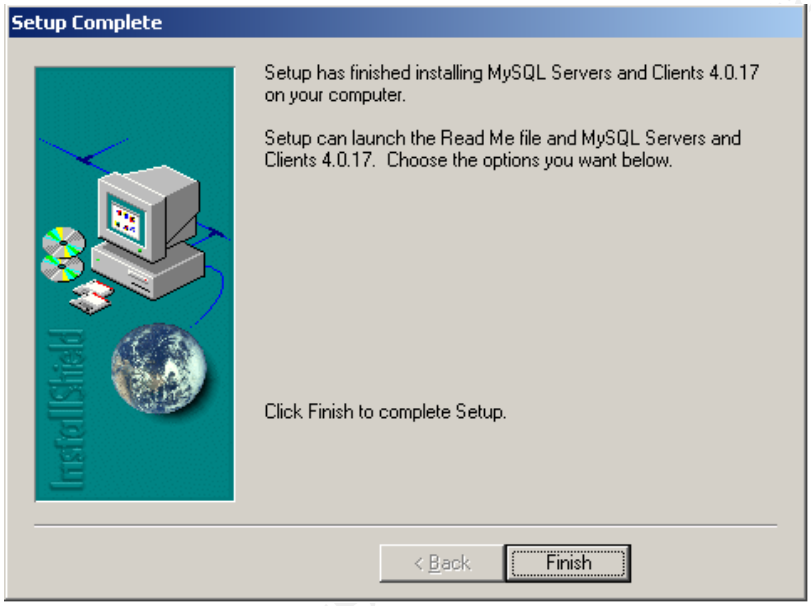
Installation
There are several components to install, so a methodology will be employed where we install from the bottom-up.

1. MySQL Database
2. Download the server[36]
3. Download the ODBC driver[37]
4. syslog2ODBC
5. apache
6. PHP
7. php-syslog-ng

Items 5-7 are strictly to provide web-based reporting.

Install the components in the same order as given for the downloads.

MySQL
The database is a wizard-driven install. The complete project, except for the ODBC drivers, is installed via the wizard.



Just accept the defaults during the MySQL install. To convert the MySQL program to a Windows service, be sure that you first stop the current server by using the following command:

36 http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.21-win.zip/from/pick#mirrors
37 http://dev.mysql.com/get/Downloads/MyODBC3/MyODBC-standard-3.51.9-win.msi/from/pick

```
shell>  -u root shutdown
```

Then execute:

```
c:\mysql\bin\mysqld –install
```

When you have finished the wizard-driven installation process, you will have installed the MySQL server and the client (named mysql) to access the database. Initially, there will be no password for the 'root' user to the database. This should be changed as soon as possible. Next we need to setup our Windows log server.

Log Server Selection
There are fewer central log server selections available for the Windows platform than I was able to find for the *nix platforms. When the requirement for database logging was added, I selected syslog2ODBC[38] as the best suitable, free solution. In fact, it may be the only solution available to meet this project's requiements. One very nice feature of the syslog2ODBC server is the ability to send logs to a valid ODBC data source. This means that we could further 'remote' the database from the central log server, even hosting the log database on yet another machine. Of course, this would present additional security challenges as well.

Database Schema Decision Required
One puzzle to be solved before implementation of this particular logging architecture is to select a database schema. Both the syslog2ODBC and syslog-ng projects provide a database schema. However, the syslog-ng project enjoys a wider following, at least as reported on the sourceforge.net project pages. For this reason, I will opt to use the schema from the syslog-ng project. Using a different schema with syslog2ODBC means that we will need to modify the .ini file which is included as part of the syslog2ODBC install package. I will include a section of the modified syslog2ODBC.ini file with the modified SQL INSERT statements.

Install the SysLog2ODBC Log Server
The Readme.txt file included with the SysLog2ODBC-0.4.zip gives installation details. I will only cover differences which enable syslog2ODBC to work with the syslog-ng schema and add an operating system note.

The SysLog2ODBC Readme.txt states that operating system support is limited to Windows 2000 and Windows 2003. I was able to configure and run syslog2ODBC on an Windows XP Professional Operating System.

Install the syslog2ODBC program as per the readme.txt file. I created a directory under "program files" called syslog2ODBC, and unzipped the install files to that directory.

The install is very easy to accomplish. Again from the ReadMe.txt file:

---

38 http://sourceforge.net/projects/syslog2odbc/

```
SysLog2ODBC  runs 2 different threads (the syslog server and the
ODBC  writer),  using a FIFO queue to store messages that are to
be written in the db.

Installation:
=============

You can install the service executing:

   SysLog2ODBC -i

The service will be started automatically by the Service Control
Manager  during  system  startup.  You  can  start  and stop the
service manually from the Services Control Panel.
```

The syslog2ODBC.ini file includes all configuration examples. The full configuration file includes :

```
[ODBC]
ConnectionString=DSN=SysLog2ODBC
SQLStatement=INSERT INTO SysLogData( Msg, SenderIP, SenderPort_S,
SenderPort_I, Priority, Severity, SeverityDesc, Facility, FacilityDesc,
RawMsg, ReceivedAt ) VALUES ( ?, ?,
 ?, ?, ?, ?, ?, ?, ?, ?, NOW() )
Param1=MSG
Param2=SENDERDEVICE_IP
Param3=SENDERDEVICE_PORT_S
Param4=SENDERDEVICE_PORT_I
Param5=PRIORITY
Param6=SEVERITY
Param7=SEVERITYDESC
Param8=FACILITY
Param9=FACILITYDESC
Param10=RAWMSG
WaitOnError=
MaxRetryCount=
```

I have modified the relevant code to read:

```
[ODBC]
ConnectionString=DSN=SysLog2ODBC;UID=syslog;PWD=syslog

SQLStatement=INSERT INTO logs( host, facility, priority, level, tag, date,
time, program, msg, seq ) VALUES ( ?, ?, ?, ?, ?, NOW(), CURTIME(), ?, ?, ?)

Param1=SENDERDEVICE_IP
Param2=FACILITY
Param3=SEVERITYDESC
Param4=SEVERITYDESC
Param5=SEVERITY
Param6=FACILITYDESC
Param7=MSG
Param8=RAWMSG
Param9=RAWMSG
Param10=RAWMSG
WaitOnError=
MaxRetryCount=
```

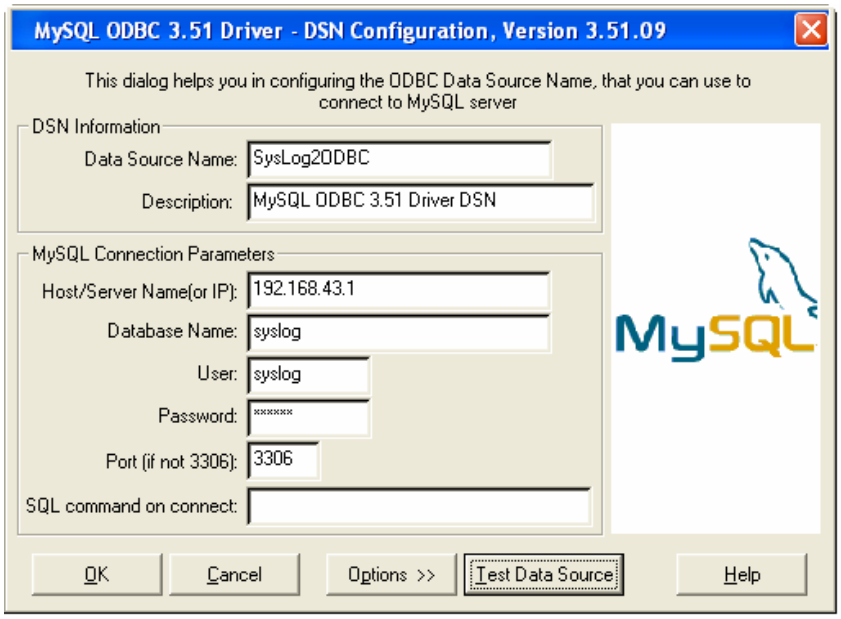
Note that the table name was changed to 'logs' to match the table name given for the syslog-ng implementation. Column names were also modified, as were "Param"

names.

When you have completed the syslog2ODBC install, move on to build the schema as per the instructions for syslog-ng (detailed earlier in this document on page 9). Remember that this was accomplished by redirecting the database build script into the MySQL client. For example:

```
mysql -u root -p <makeDB.sql
```

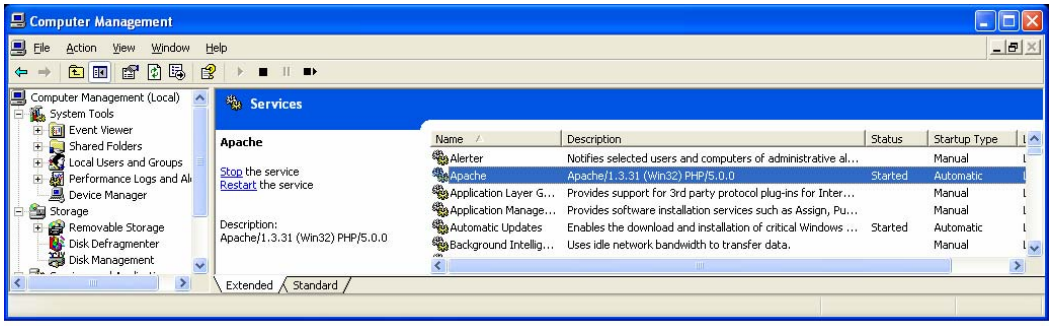
As one might suspect from the name of the project syslog2ODBC, an ODBC data source is a requirement of the syslog2ODBC service. The MySQL ODBC client works perfectly for this solution. The MySQL ODBC client is another wizard-based install.



The ODBC driver is available as a download from MySQL.com[39].

If you will be adding a web interface to your log server's database repository, then follow the log server installation by installing the apache web server. The apache web server is a wizard-based install. Be sure to use version 1.3 of the apache web server. When complete, your apache we server will show the proper version and be listed along with your other Windows services.

39 http://dev.mysql.com/downloads/connector/odbc/3.51.html



After apache has been installed, finish your web interface by installing php and php-syslog.

To encrypt our data in transit we will install zebedee on both syslog client hosts and the syslog server. Follow the instructions on page 22 of this paper to install the Windows zebedee tunnel client on both the Windows client and server. Use the configuration file on pages 22 & 23 for the client.

To finish the tunnel, configure the server's configuration file server.zbd as follows:

```
verbosity 2 # Slightly more than basic messages
detached true     # You will probably want this 'true' for normal
server true # Yes, it's a server!
ipmode udp  # UDP mode

compression zlib:9      # Allow maximum zlib compression
keylength 256           # Allow keys up to 256 bits
keylifetime 36000 # Shared keys last 10 hours
maxbufsize 16383  # Allow maximum possible buffer size


logfile SYSLOG

keygenlevel 2     # Generate maximum strength private keys

target localhost:514/udp
```

If we are successful, our netstat will reflect that zebedee is running in a state in which it listens for UDP connections:

```
TCP    0.0.0.0:11230          0.0.0.0:0              LISTENING
```

We can also install zebedee as a service on the Windows platform by following this install syntax:

```
Service: zebedee [options] -S [install[=config-file] | remove | run]
```

This completes the configuration of the secure Windows logging architecture for our Windows client and server.

Conclusion

We have identified a security risk in transmitting our log data in clear text. After the identification of the problem we have described solutions to that problem: encrypting our log data while in transit. Most of if not all of the environments we protect have key differences in operating systems, network infrastructure, and the software we may be allowed to install. Limitations on software may be in place via corporate policy – spoken or unspoken.

This paper thus approached the solution by describing various solutions with a focus on understanding how to replace components as necessary, and how the components work together to provide a solutions architecture. By the end of the paper we were able to lean on our understanding of processes learned earlier which described a Linux and Windows architecture, and apply that understanding to a strictly Windows environment.

Bibliography

[1.] Kiwi Enterprises Web Site "Initial Setup and Configuration of Kiwi Secure Tunnel"
URL: http://www.kiwisyslog.com/info_secure_tunnel.htm (15 Sep. 2004).

[2.] The Network Time Protocol (NTP) Distribution Web Site
URL: http://www.eecis.udel.edu/~mills/ntp/html/index.html (26 Sep. 2004).

[3.] The Tripwire open source web site
URL: http://sourceforge.net/projects/tripwire/ (26 Sep. 2004).

[4.] The iptables web site
URL: http://www.iptables.org/ (26 Sep. 2004).

[5.] The stunnel web site
URL: http://www.stunnel.org/ (17 Sep. 2004).

[6.] The apache web site home page
URL: http://www.apache.org/ (2 Sep. 2004).

[7.] "Why shouldn't I use Apache 2 in a production environment?", The PHP web site Installation page.
URL:http://us4.php.net/manual/en/faq.installation.php#faq.installation.apache2 (27 Sep. 2004).

[8.] MySQL web site Windows database download page
URL: http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.21-win.zip/from/pick#mirrors (19 Apr. 2004).

[9.] The syslog-ng web site
URL: http://www.balabit.com/products/syslog_ng/ (5 Aug. 2004).

[10.] The apache web site download page
URL: http://httpd.apache.org/download.cgi (2 Sep. 2004).

[11.] The php project home page
URL: http://www.php.net/ (24 Aug. 2004).

[12.] The rpmfind.net main search page
URL: http://www.rpmfind.net/ (24 Aug. 2004).

[13.] The php-syslog-ng project home page
URL: http://sourceforge.net/projects/php-syslog-ng/ (24 Aug. 2004).

[14.] MD5 Message Digest Web page including manual and downloads
URL: http://www.fourmilab.ch/md5/ (5 Aug. 2004).

Bibliography

[15.] The syslog-ng web site
URL: http://www.balabit.com/products/syslog_ng/ (5 Aug. 2004).

[16.] The syslog-ng web site binary download page
URL: http://www.balabit.com/downloads/syslog-ng/1.6/src/ (5 Aug. 2004).

[17.] The syslog-ng web site libraries download page
URL: http://www.balabit.com/downloads/syslog-ng/libol/0.3/ (5 Aug. 2004).

[18.] Bauer, Michael D. Building Secure Servers with Linux O'Reilly, 2002. 336

[19.] Excellent Google post displaying MySQL add user syntax
URL: http://groups.google.com/groups?q=g:thl3681460468d&dq=&hl=en&lr=&ie=UTF-8&selm=9se05d%24s3n%241%40host.talk.ru (8 Aug. 2001).

[20.] php-syslog-ng web site page displaying syslog to MySQL scripts
URL: http://vermeer.org/display_doc.php?doc_id=1#scripts (24 Aug. 2004).

[21.] Ibid.

[22.] sqlsyslogd web site syslogd to MySQL wrapper download page
URL: http://www.frasunek.com/sources/security/sqlsyslogd/ (26 Apr. 2004)

[23.] The php project home page
URL: http://www.php.net/ (24 Aug. 2004).

[24.] The php-syslog-ng project home page
URL: http://sourceforge.net/projects/php-syslog-ng/ (24 Aug. 2004).

[25.] Ibid.

[26.] zebedee web site home page for Secure IP Tunnel
URL: http://www.winton.org.uk/zebedee (3 Sep. 2004)

[27.] zebedee web site manual page
URL: http://www.winton.org.uk/zebedee/manual.html (3 Sep. 2004)

[28.] Ibid.

[29.] Malmberg, Joe Central Logging with Open Source Software in a Unix/Windows Environment, GSEC Practical, February 11, 2004

[30.] Kiwi Enterprises Web Site "Initial Setup and Configuration of Kiwi Secure Tunnel"
URL: http://www.kiwisyslog.com/info_secure_tunnel.htm (15 Sep. 2004).

Bibliography

[31.] Kiwi Enterprises Web Site Product Information for the syslog daemon
URL: http://www.kiwisyslog.com/products.htm#syslog (15 Sep. 2004).

[32.] syslog2ODBC web site
URL: http://sourceforge.net/projects/syslog2odbc/ (4 Nov. 2003)

[33.] The apache project home page
URL: http://www.apache.org/ (2 Sep. 2004).

[34.] The php project home page
URL: http://www.php.net/ (24 Aug. 2004).

[35.] The php-syslog-ng project home page
URL: http://sourceforge.net/projects/php-syslog-ng/ (24 Aug. 2004).

[36.] MySQL web site Windows database download page
URL: http://dev.mysql.com/get/Downloads/MySQL-4.0/mysql-4.0.21-win.zip/from/pick#mirrors (19 Apr. 2004).

[37.] MySQL web site ODBC driver download page
URL: http://dev.mysql.com/get/Downloads/MyODBC3/MyODBC-standard-3.51.9-win.msi/from/pick (13 Jan 2004).

[38.] The syslog2ODBC web site
URL: http://sourceforge.net/projects/syslog2odbc/ (4 Nov. 2003)

[39.] MySQL web site ODBC driver download page
URL: http://dev.mysql.com/downloads/connector/odbc/3.51.html (13 Jan 2004).

Appendix A

Screen Blanking on a Linux syslog server

In a secure area, it may be desireable to display the syslog output on the console. On a Linux system this is a problem as a Linux host will blank a text-based terminal after several minutes. To avoid this, you may execute the following command:

```
echo -n -e "\033[9;0]"
```

Appendix B

Zebedee start script

Tested on Red Hat 7.3, Fedora Core 2

This is script is taken from various Red Hat /etc/init.d/ scripts and I have modified it for zebedee.

```
#!/bin/sh
################################################################################
# Program: zebedee init script for Red Hat
# Date:    9.25.2004
# Purpose: zebedee is secure tunnel software.
# chkconfig: 345 90 25
# description: This script takes care of starting and stopping zebedee
################################################################################

INIT_PROG=zebedee

#
# Source function library.
#
. /etc/rc.d/init.d/functions

# path to zebedee
ZEBEDEE_PATH=":/usr/local/sbin"

PATH=$PATH$ZEBEDEE_PATH
export PATH
ZEBEDEE_OPTS='-s -U -f /usr/local/etc/zebedee/server.zbd'
RETVAL=0
umask 077

# begin script logic
start() {
        echo -n "Starting $INIT_PROG: "
        daemon $INIT_PROG $ZEBEDEE_OPTS
        RETVAL=$?
        echo

        [ $RETVAL -eq 0 ] && touch "/var/lock/subsys/${INIT_PROG}"
        return $RETVAL
}

stop() {
        echo -n "Stopping $INIT_PROG: "
        killproc $INIT_PROG
        RETVAL=$?
        echo

        [ $RETVAL -eq 0 ] && rm -f "/var/lock/subsys/${INIT_PROG}"
        return $RETVAL

}
```

```
rhstatus() {
        status $INIT_PROG
}

restart() {
        stop
        start
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        rhstatus
        ;;
  restart|reload)
        restart
        ;;
  condrestart)
        [ -f /var/lock/subsys/${INIT_PROG}] && restart || :
        ;;
  *)
        echo $"Usage: $0 {start|stop|status|restart|reload}"
        exit 1
esac

exit $?
```

Appendix C

Diagram of a Secure Logging Architecture for a Heterogeneous OS Environment