Global Information Assurance Certification Paper

Firewall log analysis using ACID

SANS Security Essentials (GSEC) Practical Assignment Version 1.4b Option 2

Anthony Shearer
5th December 2003

1. Introduction

Regular monitoring of firewall logs can reveal a wealth of information about threats to an organisation. In the event of a compromise the logs play a critical role in evaluating the extent of the attack, and as evidence against the attacker. However, for organisations with multiple firewalls the effectively monitoring and responding to the high volumes of alerts can be a time consuming process

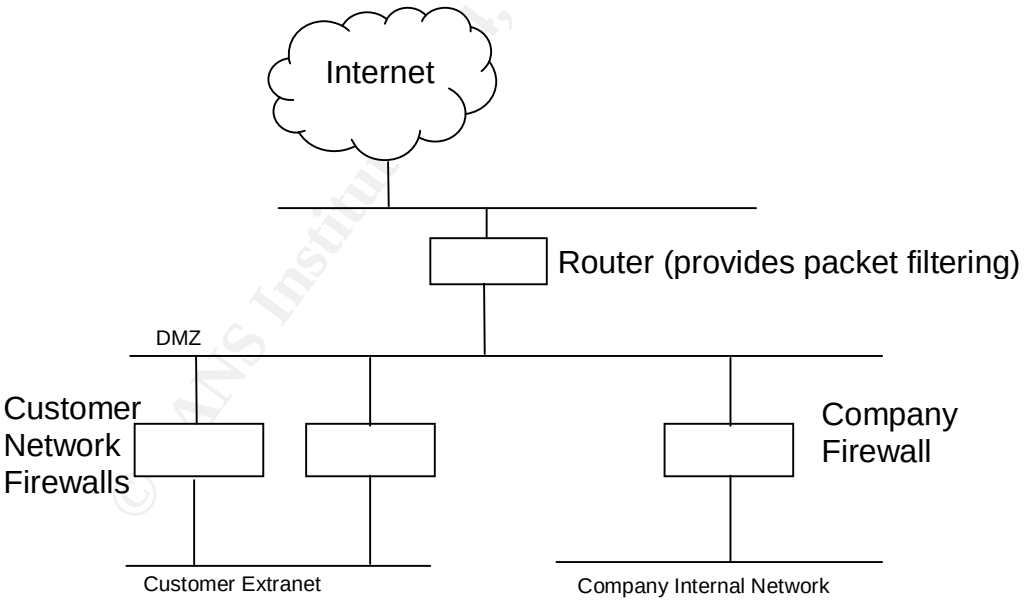
This paper describes how an organisation's security was improved by building a system to allow in-depth analysis of firewall logs using open source tools to automate as much of the monitoring process as possible.

2. Before

2.1. Description of the Problem

One of the services the company I work for offers is secure Internet connectivity to businesses. The customer is provided with the means to connect to our customer extranet, and various Internet services are provided by the way of Proxy firewalls connected to our DMZ.

The company also has a firewall to connect its internal network to the DMZ. A Cisco based packet filter protects the DMZ. The diagram below shows a simplified version of the infrastructure:



In addition, we have provided several customers with firewall solutions that we monitor on their behalf. These firewalls are all based on IPFW running on BSDi Unix, but since the announcement that BSDi is to be discontinued there are plans in place to use Linux/IPTables based firewalls in the near future.

The problem is finding a way to effectively monitor filter violations on all of these firewalls and still maintain enough detail in the logs to be useful in the event of a security incident.

2.2. Why is it important to monitor firewall violations?

System administrators often do not recognise the value of monitoring firewall logs, taking the stance of 'if the firewall has blocked the traffic, it is no longer a threat so why should I spend time analysing it?'

There are numerous reasons to monitor these logs. The information enables the effectiveness of the firewall rule set to be measured, and if necessary adjusted to meet any new vectors of attack or introduce other stronger security measures.

The logs can be used to document attempted attacks and build up a picture of how attackers are trying to gain access to the company's systems. Where enough information is available, the documentation can be presented to the ISP who provides connectivity for the source of the attacks so appropriate action can be taken.

Before an attack even occurs, the firewall logs will usually show any reconnaissance taking place (e.g, port scans). It is quite common for an attacker to check the port that a backdoor Trojan uses before they try to install it on the host. Proper analysis can alert the administrator to the threat.

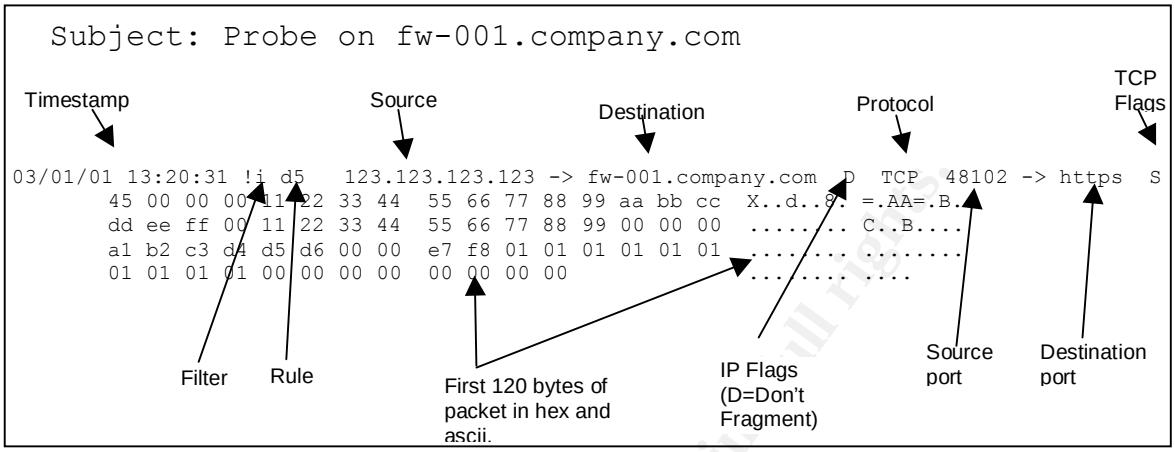
In the event of a security breach the logs can be used to investigate the actions taken by the attacker, and as evidence in any legal proceedings that occur as a result.

From a systems administration point of view, the firewall logs can pinpoint machines that are improperly configured or have unauthorised software installed. An example would be if a workstation started to send HTTP requests directly to the Internet, when usually it uses a Proxy server. Many Trojans that try to send information to the Internet can be identified in the same way.

It could be argued that this information can be provided by a network based IDS system. However, most IDS systems are limited in the alerts they can generate by the extent of their signature database, whereas a well-configured firewall will block and log *all* traffic that hasn't been explicitly allowed. This means that on many occasions a firewall will block and log activity of a Trojan that would appear to an IDS to be 'normal' activity.

2.3. The Current System

Currently each firewall maintains a local log of filter violations. It also e-mails alerts of unusual probes to the administrators. This shows an example alert from a BSDI firewall:



Although this system allows the firewalls to be monitored, it has a number of shortcomings:

- No alerts are currently received from the Router when ACL violations occur.
- The number of alerts being received by the administrators is often more than can be effectively monitored. Tuning the logging settings on the firewall rules has reduced this, but as the number of firewalls grows, so does the number of alerts.
- There is no easy way to monitor trends in firewall violations over time. This would be useful to spot attackers doing reconnaissance and can also provide early warning of vulnerabilities in software by looking for increased activity on destination ports.

3. During

3.1. How the problem was approached

The new system still needed to produce e-mail alerts, but these alerts needed to be sent to the correct person (rather than all the administrators). Another requirement was to cut down on the number of alert mails generated wherever possible (eg, by aggregating duplicate alerts in a short timeframe).

The main requirement was to enable in-depth analysis of the logs from all the firewalls. This is a challenge because of the three different log formats (IPFW, IPTables and Cisco). The solution needed to be straightforward to use and ideally web-based for ease of access.

The first step to analysing the logs is to get all the firewall logs in one place. This could be achieved by having all the firewalls log to a central host using syslog. This was an ideal choice because the firewalls already used syslog to record logs locally, so it would just be a case of modifying the syslog configuration on each firewall. It also keeps the amount of processing required on the firewall to the minimum.

A requirement for this server is strong security to protect the log files once they are on the server. In the event of a firewall being compromised, a copy of the logs is preserved on the central server.

After much thought and investigation into existing tools available to analyse the logs I discovered a tool called 'logsnorter'[1]. This is a perl script that can parse log files from ipchains, iptables, Cisco, and ipfw based firewalls. The script records the information to a database with the same format as the Snort IDS uses. This opened up the possibility of using ACID (the Analysis Console for Intrusion Databases) to analyse and report on the data.

Preliminary testing of the logsnorter script revealed that the script did not parse BSDi ipfw logs, and had no facility to store the hex dump of the start of the packet into the database (see section 2.3 for the format). A search using Google to find a tool with this ability turned up nothing, so the only solution would be to write a script to parse these logs separately.

One facility ACID does not provide is the ability to generate automated e-mail alerts of events that require the attention of an administrator. There are numerous log-monitoring tools available that could be used for this purpose. Once such tool is Swatch (Simple Watcher of Log Files), which allows e-mails alerts to be sent if a line in the log matches a pre-defined pattern.

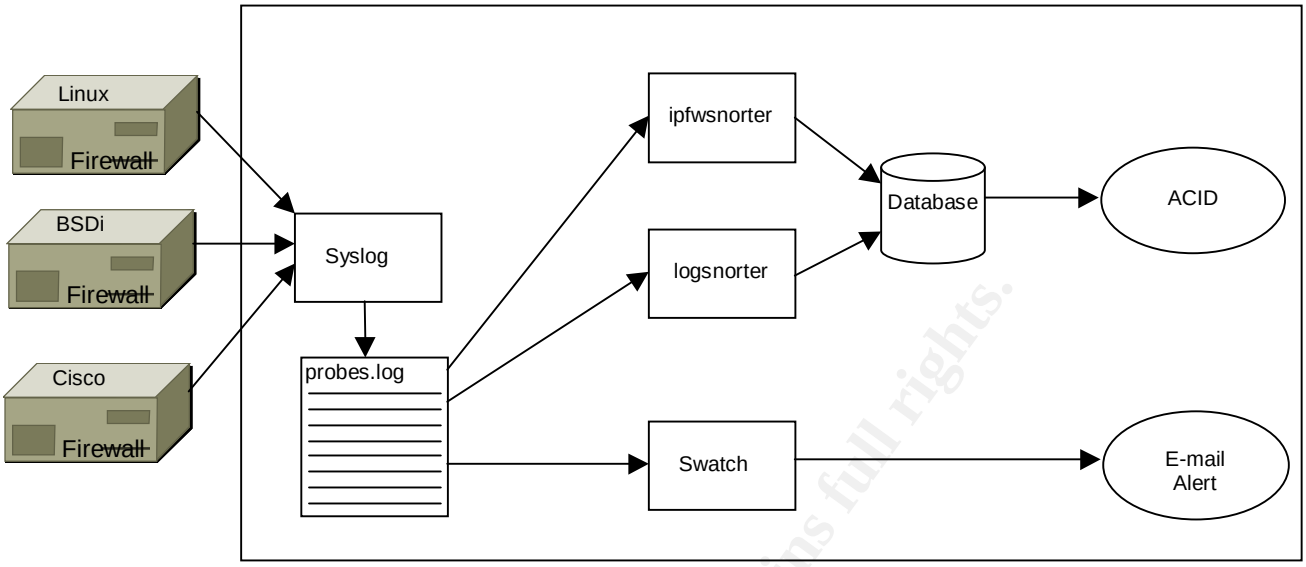
3.2. The final design

A central log server will be built, based upon Linux 9 and secured according to the SANS Institute's Securing Linux guidelines. All firewalls will be configured to log violations to the central server using syslog.

Once on the server, the logs are to be parsed using logsnorter into a MySQL database. The database schema will be identical to that of the Snort IDS database module. An additional script needs creating to parse the logs from the BSDi firewalls ('ipfwsnorter').

[1] Available from http://www.snort.org/dl/contrib/other_logs/

Analysis of the data will be preformed by ACID, while alerts that require more immediate attention will be generated by Swatch.



3.3. Building and securing the Log Server

The first step is to install Linux on the server and reduce the number of services to the minimum needed. There are numerous guides available on how to install and secure Linux. I chose to follow the SANS Securing Linux Step-by-Step guide.

When installing packages, the minimal installation was selected, plus the following:

- Apache Web Server
- MySQL Database Server
- Perl
- Perl DBI module
- PHP
- GD (to enable the graph facilities of ACID)

When removing unnecessary network services, the server should be able to perform its job with only the following network services:

- Apache on port 80/tcp
- SSHD on port 22/tcp
- Syslogd on port 514/udp

Mysqld also needs to be running, but all connections can be made using a local Unix socket. To disable networking in MySQL, add the following line to the /etc/my.cnf file under the `[mysqld]` section:

```
[mysqld]
skip-networking
```

Once only the necessary services are running on the server it is advisable to install a packet filter to ensure only the firewalls are allowed to send syslog packets to the server. This reduces the risk of an attacker 'polluting' the log files by sending syslog packets from a different host. It can also be used to ensure the web server and SSH access is possible only for designated administrator workstations.

Tests should then be made to audit the effectiveness of the above security measures. The server can then be connected to the network.

At this stage it is beneficial to use the 'up2date' utility to update the system. This will also usually result in updates to some of the software packages used in the later stages such as Apache and MySQL.

3.4. Configuring a Central Syslog Host

Syslogd needs to be configured to listen for logs from the network, and to record all firewall logs to a separate log file. I am assuming all logs will be coming to us with a facility of local1. This is done by making the following alterations to /etc/syslog:

- Add the following lines:
```
# Log firewall probes from remote servers
local1.*                                /var/log/probes
```
- Make it so no firewall logs are logged to /var/log messages by adding 'local1.none' to the list of facilities that log to this file:
```
*.info;mail.none;authpriv.none;cron.none;local1.none        /var/log/messages
```

In some circumstances it may be necessary to have firewall logs use other facilities in addition to local1. For example, Cisco routers send syslog messages to local7 by default, but this can usually be changed (See http://www.siliconvalleyccie.com/cisco-hn/syslog-cisco.htm#_Toc42865892).

By default syslogd receives logs using a local Unix socket. The '-r' command line option causes syslogd to also listen on port 514/udp. To make this change, edit the /etc/sysconfig/syslog file, and make sure the syslogd options are defined as:

```
SYSLOGD_OPTIONS="-r -m 0"
```

To make all these changes take effect, restart syslogd using:

```
service syslog restart
```

Now is a good time to start collecting logs from the firewalls so there is some data waiting to analyse in the later stages. Alterations were made to the syslogd configuration on the firewalls to make it so logs are sent to the server using a facility of local1.

3.5. Database Setup

The next step is to configure the database that will be used to store the logs for analysis. Before MySQL can be used it needs to be initialised using the mysql_install_db script, and a root password needs to be set:

```
mysql_install_db
mysqladmin -u root  password 'new-password'
```

To get the database schema it is necessary to download (but not install) the Snort IDS from http://www.snort.org. The Snort distribution contains a file with the SQL commands necessary to install the database tables:

```
cd /usr/local/src
wget http://www.snort.org/dl/snort-2.0.5.tar.gz
tar -zxvf snort-2.0.5.tar.gz
cd snort-2.0.5/contrib.
mysqladmin -p create probes
mysql -p probes < create_mysql
```

To check that the database has been created run the command:

```
echo "show tables;" | mysql -p probes
```

You should then be presented with a list of tables similar to the following:

```
Tables_in_probes
data
detail
encoding
event
icmphdr
iphdr
opt
reference
reference_system
schema
sensor
sig_class
sig_reference
signature
tcphdr
udphdr
```

3.6. Log Parsing Tools

The parsing tools continually watch the probes.log file, parse any lines that are recognised as firewall probes and insert the information into the database. For the reasons explained in section 3.1 it is necessary to have two log parsing tools running – one for BSDi logs and the other for Cisco and IPTables logs (logsnorter).

For the log parser to be able to store information a user must be setup for the probes database. Since the parsers have a well-defined job, a very restricted set of permissions can be granted for this user:

```
mysql -p probes
grant insert on probes.data to logsnorter@localhost identified by 'password';
grant select on probes.detail to logsnorter@localhost;
grant select on probes.encoding to logsnorter@localhost;
grant select,insert on probes.event to logsnorter@localhost;
grant insert on probes.icmphdr to logsnorter@localhost;
grant insert on probes.iphdr to logsnorter@localhost;
grant insert on probes.opt to logsnorter@localhost;
grant insert on probes.reference to logsnorter@localhost;
```

```
grant select,insert on probes.reference_system to logsnorter@localhost;
grant select on probes.schema to logsnorter@localhost;
grant select,insert on probes.sensor to logsnorter@localhost;
grant select,insert on probes.sig_class to logsnorter@localhost;
grant select,insert on probes.sig_reference to logsnorter@localhost;
grant select,insert on probes.signature to logsnorter@localhost;
grant insert on probes.tcphdr to logsnorter@localhost;
grant insert on probes.udphdr to logsnorter@localhost;
exit
```

It is also a very good idea to run the log parsers using unprivileged accounts. In the event of an attacker successfully finding and exploiting a vulnerability in the scripts the attacker will be restricted to the minimal privileges of the 'logsnorter' account:

```
useradd logsnorter
passwd logsnorter
```

To prevent this user from logging onto the server, the /etc/passwd file should be updated so the user's shell is '/sbin/nologin'. The user also needs to be granted read-only permission to the /var/log/probes logfile:

```
chown logsnorter.root /var/log/probes
chmod 420 /var/log/probes
```

These permissions allow only members of the 'root' group to write to the log, and read-only access to the 'logsnorter' user.

3.6.1. Installing Logsnorter

Logsnorter consists of a single perl script that constantly monitors the log file. The process to install it is as follows:

```
mkdir /usr/local/logsnorter
cd /usr/local/logsnorter
wget http://www.snort.org/dl/contrib/other_logs/logsnorter-0.2.tar.gz
tar  -zxvf logsnorter-0.2.tar.gz
mv logsnorter-0.2 logsnorter
chmod 711 logsnorter
```

Logsnorter reads its configuration from /etc/logsnorter.conf. The following configuration should be created to allow logsnorter to connect to the database:

```
$db_server='localhost';
$db_usercode='logsnorter';
$db_database='probes';
$db_password='password';
```

To start the logsnorter run the following command. It should also be added to the end of the /etc/rc.local file to ensure that log monitoring begins when the system boots:

```
sudo -b -u logsnorter /usr/local/logsnorter/logsnorter -t -T /var/log/probes
```

With logsnorter successfully installed, any logs received from IPTables or Cisco based firewalls should start filling up the database.

3.6.2. Monitoring BSDI firewall logs

To parse logs from BSDI ipfw based firewalls (which is the majority in our organisation) a perl script was written called 'ipfwsnorter'. Although this script handles only one format of firewall log, it is in many ways more complex than logsnorter due to it's ability to read probes that span multiple lines in the log file (to accommodate the packet dump – see section 2.3).

The script was saved into /usr/local/logsnorter, and the permissions were set to be the same as for the logsnorter script above. To start monitoring, the following command must be run:

```
sudo -b -u logsnorter tail -0f /var/log/probes |/usr/local/logsnorter/ipfwsnorter
```

Again, this should be appended to the /etc/rc.local file to ensure the script is launched when the server boots.

3.6.3. Swatch Configuration

E-mail alerts of certain events will be generated by Swatch (Simple Watcher of Log files). This will constantly monitor probes.log for certain patterns, as defined in the swatch.conf file. When a pattern is found, the line of the log file is e-mailed to the appropriate administrator.

After downloading the latest release from http://swatch.sourceforge.net/ the following steps should be taken to install and configure Swatch:

```
tar –zxvf swatch-3.0.8.tar.gz
cd swatch-3.0.8
perl Makefile.PL
make
make test
make install
```

The swatch configuration should be specified in /etc/swatch.conf. Lines that should generate alerts are specified using *watchfor /[perl regular expression]/*. It is also possible to make exceptions by using *ignore /[perl regular expression]/*. For example, to send all alerts involving the 10.10.10.0/24 network to admin@company.com, the following configuration can be used:

```
watchfor /10.10.10.[0-9]+/
        mail address=admin@company.com, subject="Firewall Violation"
```

Once a configuration is in place, Swatch can be started by running the command:

```
sudo -b -u logsnorter /usr/bin/swatch --config-file=/etc/swatch.conf --tail-
file=/var/log/probes
```

This should be added to the /etc/rc.local file to ensure Swatch is started next time the server is rebooted.

3.7. Log Rotation

Log rotation needs to be configured to keep the logs for a length of time as defined by the company's security policy. In our case, the requirement is for the logs to be kept for at least 1 year.

Log rotation is configured by editing /etc/logrotate.conf. The following section was added to the file:

```
/var/log/probes {
    prerotate
        /usr/bin/killall logsnorter
        /usr/bin/killall ipfwsnorter
    endscript
    rotate 12
    monthly
    compress
    create 420 logsnorter root
    postrotate
        sudo -b -u logsnorter /usr/local/logsnorter/logsnorter -t -T /var/log/probes
        sudo -b -u logsnorter tail -0f /var/log/probes |
/usr/local/logsnorter/ipfwsnorter
    endscript
}
```

The prerotate and postrotate commands temporarily stop the log parsers while the log is rotated.

3.8. ACID Installation

The process for installing ACID is well documented at http://www.andrew.cmu.edu/~rdanyliw/snort/acid_config.html. However, the steps used to install can be modified in this case since much of the required software has been installed with the Red Hat distribution.

ACID requires different access permissions to the database than those already granted to the logsnorter user. For the initial setup, ACID requires permission to create tables. An 'acid' database user was created with full rights to the database:

```
mysql -p probes
grant all privileges on probes.* to acid@localhost identified by 'password'
exit
```

Once setup is complete these permissions can be tied down to the minimum required for ACID to operate.

3.8.1. Installing Prerequisites

There are several software packages that ACID relies on for its database access and graphing facilities. ADODB provides database extraction classes for PHP. Installation involves the following:

```
cd /var/www/html
wget http://phplens.com/lens/dl/adodb404.tgz
tar -zxvf adodb404.tgz
rm adodb404.tgz
```

JPGraph and PHPLot are used by ACID's graphing facilities. Installation involves:

```
wget http://members.chello.se/jpgraph/jpgdownloads/jpgraph-1.14.tar.gz
tar -zxvf jpgraph-1.14.tar.gz
rm jpgraph-1.14.tar.gz

wget http://ftp1.sourceforge.net/phplot/phplot-4.4.6.tar.gz
tar -zxvf phplot-4.4.6.tar.gz
rm phplot-4.4.6.tar.gz
```

3.8.2. Installing and Configuring ACID

Installation of ACID itself is simply a case of extracting the necessary files in to the web server's document root folder:

```
wget http://www.andrew.cmu.edu/~rdanyliw/snort/acid-0.9.6b23.tar.gz
tar -zxvf acid-0.9.6b23.tar.gz
rm acid-0.9.6b23.tar.gz
```

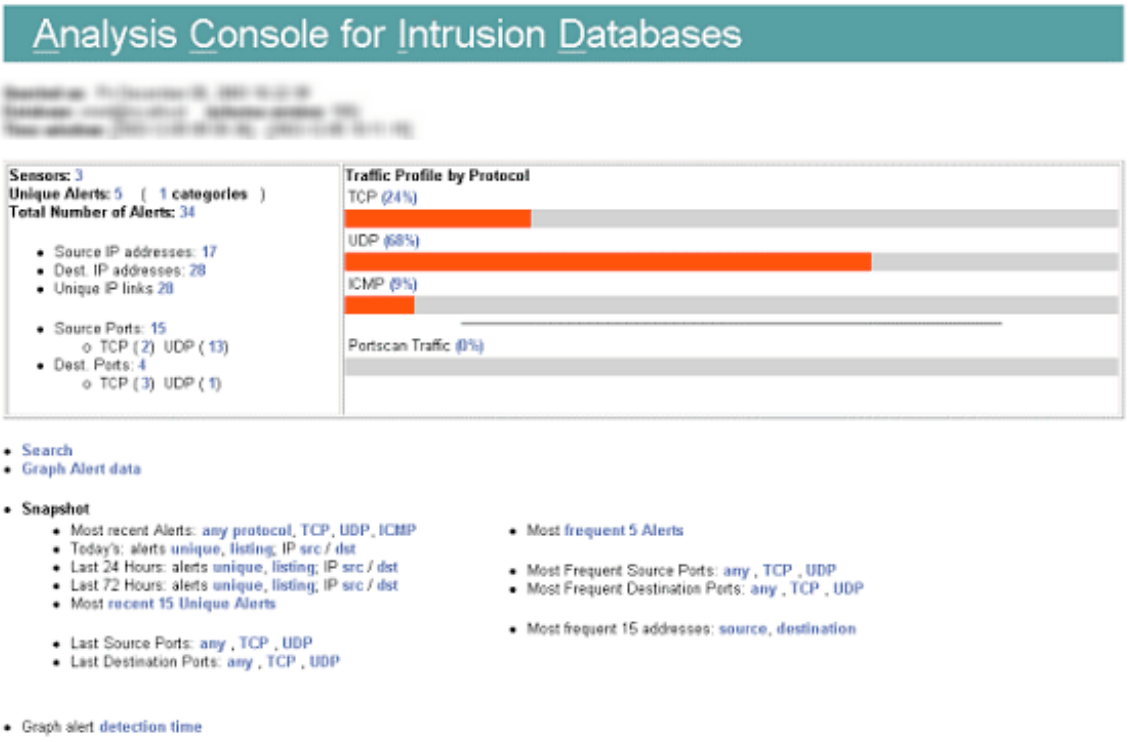
There are several configuration variables that must be set before ACID can be used. These are modified in the acid/acid_conf.php file, where the following changes should be made:

```
$DBlib_path = "/var/www/html/adodb";
$DBtype = "mysql";

$alert_dbname   = "probes";
$alert_host     = "localhost";
$alert_user     = "acid";
$alert_password = "password";

$ChartLib_path = "/var/www/html/jpgraph-1.13/src";
$chart_file_format = "png";

$event_cache_auto_update = 0;
```

ACID maintains a cache of alerts in a format that allows faster analysis than would be possible with the standard Snort schema. The default is for this alert cache to be updated each time a page is accessed from a web browser - this can lead to sluggish performance. The last change in the configuration above ($event_cache_auto_update) disables the automatic updates of this cache. Instead this can be done off-line as a cron job every 5 minutes by adding the following to /etc/crontab:

```
0,5,10,15,20,25,30,35,40,45,50,55 * * * * logsnorter GET
http://localhost/acid/acid_maintenance.php?submit=Update+Alert+Cache > /dev/null

0,5,10,15,20,25,30,35,40,45,50,55 * * * * logsnorter GET
http://localhost/acid/acid_maintenance.php?submit=Update+IP+Cache > /dev/null
```

The final setup is completed by visiting http://[server's_ip]/acid/ in a web browser. On screen instructions should be provided on how to complete the setup. Once this is complete, the main ACID screen should load:



The final step in configuring ACID is to set appropriate database permissions. The minimal permissions required for ACID to function can be found at http://www.andrew.cmu.edu/~rdanyliw/snort/acid_config.html, and can be put into effect by running the following commands:

```
mysql -p probes
revoke all on probes.*;
grant select,insert,delete on probes.acid_ag to logsnorter@localhost;
grant select,insert,delete on probes.acid_ag_alert to logsnorter@localhost;
grant select,insert,update,delete on probes.acid_event to logsnorter@localhost;
grant select,insert,update,delete on probes.acid_ip_cache to logsnorter@localhost;
grant select,insert,delete on probes.data to logsnorter@localhost;
grant select on probes.detail to logsnorter@localhost;
grant select on probes.encoding to logsnorter@localhost;
grant select,insert,delete on probes.event to logsnorter@localhost;
grant select,insert,delete on probes.icmphdr to logsnorter@localhost;
grant select,insert,delete on probes.iphdr to logsnorter@localhost;
grant select,insert,delete on probes.opt to logsnorter@localhost;
grant select,insert,update,delete on probes.reference to logsnorter@localhost;
grant select,insert,update,delete on probes.reference_system to logsnorter@localhost;
grant select on probes.schema to logsnorter@localhost;
grant select,insert on probes.sensor to logsnorter@localhost;
grant select,insert,update,delete on probes.sig_class to logsnorter@localhost;
grant select,insert,update,delete on probes.sig_reference to logsnorter@localhost;
grant select,insert on probes.signature to logsnorter@localhost;
grant select,insert,delete on probes.tcphdr to logsnorter@localhost;
grant select,insert,delete on probes.udphdr to logsnorter@localhost;
exit
```

4. After - Evaluation of the solution

After an evaluation period where firewall logs have been monitored using the new and old systems. The new system has been found to reduce the workload involved in monitoring the logs, while increasing the amount of information available about activity at each firewall.

By tuning the Swatch configuration over a number of days it was possible to make it so e-mail alerts were only sent in exceptional cases where immediate investigation was needed.

All other alerts were still accessible by regular monitoring of ACID. Reports such as the 'Unique alerts in the last x hours' give a simple overview of overnight activity, and can be reviewed each morning, and investigated in more detail where necessary. This is a significant improvement over the old process of having to scan through numerous e-mails full of isolated alerts each morning where it was easy to overlook a problem due to the lack of any facility to group alerts and see the overall picture.

It is now also much easier to monitor the effectiveness of the boundary router's ACLs. This used to involve the arduous task of reading the raw logs , but now these probes can be viewed and analysed in the same detail as the other firewalls.

This increased effectiveness of log monitoring has improved the security of the organisation in a number of ways:

- The central repository of probes allows the methods used to scan/attack the organisation's perimeter to be monitored in much more detail, and in combination rather than treating each firewall as a separate entity.

- Trends in certain types of attacks can be easily identified using ACID's graphing abilities. Even in the short time the system has been in use the system was used to spot a massive increase in attempted connections to port 27347/tcp, which was verified by the Internet Storm Centre (http://isc.incidents.org/port_details.html?port=27347). At the time there was little information available about the cause of this increase, but with the early warning it was possible to run a nmap scan of the network to ensure that none of our machines were vulnerable to this attack.

- Since our firewalls perform egress filtering, it is now easy to spot many configuration problems and/or unauthorised software installed on internal workstations by doing an ACID search for any firewall violations with an internal source address. In combination with a network based IDS this increases the chances of detection in the event of an internal host being compromised.

There are a number of improvements that could be made to the system to increase both performance and security. If the hardware was available, I would choose to split this task over 2 servers. One would be a dedicated syslog host, which would also parse the logs. The information would then be inserted into a database on the other server, which would be used exclusively as an analysis console. The main benefit of this design is the increased security of the log server – ideally a central log host should only listen for syslog messages. Running a web server and a database server increases the chance that an attacker could exploit a vulnerability in this software and compromise the log files.

Another way the system could be improved would be to integrate the information gathered from IDS sensors. The current database schema would need no modifications for Snort to insert information using the database output module. This would then allow a unified view of all IDS/firewall information from one console.

5. References

1. McCormic, Bob; Hartung, Greg; Conger, Zacharay. "Universal Firewall Reporting Tool." SysAdmin Magazine August 2003 (2003): 46 - 53.

2. Danyliw, Roman. "ACID: Installation and Configuration." 9 Oct 2002. URL: http://www.andrew.cmu.edu/~rdanyliw/snort/acid_config.html (5 Dec 2003).

3. Danyliw, Roman. "ACID: Database (v100-103) ER Diagram." 19 Jun 2001. URL: http://www.andrew.cmu.edu/~rdanyliw/snort/acid_config.html (5 Dec 2003).

4. "MySQL Reference manual." URL: http://www.mysql.com/doc/en/index.html (5 Dec 2003)

5. "SWATCH: The Simple WATCHer of Logfiles." URL: http://swatch.sourceforge.net (5 Dec 2003).

6. Harrison, Peter. "Syslog Configuration and Cisco Devices." Cisco Companion PDF Topics – Appendex I. URL: http://www.siliconvalleyccie.com/cisco-hn/syslog-cisco.htm#_Toc42865892

7. Ranch, David; Brotzman, Lee. Securing LINUX Step by Step. SANS Institute, 30 July 1999.

8. Taylor, Laura. "Read your firewall logs!" ZDNet. 5 July 2001. URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2782699,00.html (5 Dec 2003)

9. LURHQ Threat Intelligence Group. "A Firewall Log Analysis Primer." LURHQ. URL: http://www.lurhq.com/firewallprimer.html (5 Dec 2003)