

## Malware Quiz #6

Jim Halfpenny <jim.halfpenny@gmail.com>

### *Getting started*

Files downloaded, binaries unpacked from zip archives. I started looking first at the binary files, since knowing a little about them may help to understand the log files. The first thing that came to mind when I saw the filename cmd.gif was the recent Windows WMF exploit. A malicious WMF will still have the payload triggered if it is renamed to another image file type, such as a GIF. I file the command `file` to try and determine what content of the files.

```
[jim@fox binaries]$ file cmd.gif
cmd.gif: exported SGML document text
[jim@fox binaries]$ file shelll
shelll: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for
GNU/Linux 2.2.5, dynamically linked (uses shared libs), not stripped
```

### **1. Are these files packed? If so, which packer?**

So we have two files, one a Linux ELF executable and the other find detects as an SGML file. This is strange since the file extension on the second files suggests it's a GIF image. I ran strings on both of these files and analysed the output. There were a lot of strings for cmd.gif. In fact it was a text file, a PHP script to be precise. Looks like my initial WMF suspicion was unfounded! I checked the files with the virus scanners AVG, F-prot and clamav. Only clamav identified either of these files as malware.

```
/home/jim/malware_quiz/binaries/cmd.gif: PHP.Defash.A FOUND
```

A comment appears at the start of this file:

```
<!--
Defacing Tool 2.0 by r3v3ng4ns
revengans@gmail.com
se for modificar o codigo, por favor, mantenha o nome de seus autores
originais
e por favor, entre em contato comigo...

ae galera, serio, tem mta gente fdp q simplismente usa, nao seja soh
um sucker do script,
n seja um lammer imbecil, n seja o merda dum script kiddie, n seja um
babaca, ajude a melhora-lo tambem!!
-->
```

It looks like it's in Spanish, so I am unable to read it. I can hazard a guess at the meaning of, "merda dum script kiddie," though. To answer the questions, cmd.gif is not packed or compressed.

Moving on to shelll. The name suggests it may give an attacker some manner of shell access to the compromised host. Again I started by looking at the strings output. A number of things stood out straight away such as, "irc.ircnet.net." This is the name of an Internet Relay Chat server and IRC is a popular means of controlling compromised

hosts, particularly botnets consisting of a large number of nodes. This ties in with the suggestion that it has been used as part of a DoS attack. There are also a number of other strings in this file that look IRC status messages and commands that can be issued to the. The abundance of interesting strings such as this in the binary file suggests it has not been packed.

## **2. Without running the applications, is it possible to identify what the malware can and will do?**

I've already begun to look at the capabilities of the suspicious files. cmd.gif is relatively easy to examine since it is a PHP script. The lazy way to determine it's capabilities is to search the web for, "Defacing Tool 2.0 by r3v3ng4ns."

*"Defacing Tool 2.0 by r3v3ng4ns" is a suite of php based scripts that allows the attacker to send commands to the server primarily with the intent to deface websites."*

Because assumption is the mother of all foul-ups, I performed a manual inspection and was satisfied that this is indeed the case; cmd.gif allows a remote attacker to execute commands and gather information via a web server.

shelll appears to be an IRC backdoor. Judging solely by the output from `strings` I would suggest that it connects to a channel on irc.ircnet.net and offers password protected access to a number of functions. The string /usr/dict/words may indicated that it uses this file as a source of IRC nicknames. I have copied a number of strings that appear to be help prompts for this IRC agent. This suggests that the primary aim of this tool is to carry out DoS attacks by sending a flood of SYN packets.

```
NOTICE %s :MOVE <server>
NOTICE %s :TSUNAMI <target> <secs> = Special packeter that wont be
blocked by most firewalls
NOTICE %s :PAN <target> <port> <secs> = An advanced syn flooder
that will kill most network drivers
NOTICE %s :UDP <target> <port> <secs> = A udp flooder
NOTICE %s :UNKNOWN <target> <secs> = Another non-spoof udp flooder
NOTICE %s :NICK <nick> = Changes the nick of the client
NOTICE %s :SERVER <server> = Changes servers
NOTICE %s :GETSPOOFS = Gets the current spoofing
NOTICE %s :SPOOFS <subnet> = Changes spoofing to a subnet
NOTICE %s :DISABLE = Disables all packeting from this client
NOTICE %s :ENABLE = Enables all packeting from this client
NOTICE %s :KILL = Kills the client
NOTICE %s :GET <http address> <save as> = Downloads a file off the
web and saves it onto the hd
NOTICE %s :VERSION = Requests version of client
NOTICE %s :KILLALL = Kills all current packeting
NOTICE %s :HELP = Displays this
NOTICE %s :IRC <command> = Sends this command to the server
NOTICE %s :SH <command> = Executes a command
```

To corroborate these assumptions I can take a look at the data taken from the compromised machine. The output from `netstat` shows a number of connections to the IRC server irc.ircnet.net.

```
tcp          0      0 192.168.0.52:1418      216.115.95.70:6667
ESTABLISHED
tcp          1      0 192.168.0.52:1353      216.115.95.70:6667
CLOSE_WAIT
```

```

tcp          0      0 192.168.0.52:1288      216.115.95.70:6667
CLOSE_WAIT
tcp          1      0 192.168.0.52:1225      216.115.95.70:6667
CLOSE_WAIT
tcp          1      0 192.168.0.52:1161      216.115.95.70:6667
CLOSE_WAIT

```

One thing to note is there is no web server on port 80. For cmd.gif to work I would expect to see a web server listening on this host. There could be a web server listening on a port other than port 80, or the port could be hidden, by a rootkit for instance.

### **3. Now, using any methods available to you, which changes, if any, will the malware make on the system?**

Over an above the execution of arbitrary command and downloading arbitrary files from a web server I could not see that cmd.gif made any changes to the host system.

I executed shelll on a Linux box, running tcpdump at the same time on another terminal. It started successfully and detached from the terminal. I examined what files the process had opened by using lsof.

```
[root@fox ~]# tcpdump -s 0 -w /var/tmp/shelll.dump host 216.115.95.70
```

```

[jim@fox binaries]$ lsof -n -p 6887
COMMAND  PID USER  FD  TYPE DEVICE  SIZE  NODE NAME
shelll   6887  jim   cwd      DIR          3,65          216 13283
/home/jim/malware_quiz/binaries
shelll   6887  jim   rtd      DIR    3,6      584      2 /
shelll   6887  jim   txt      REG          3,65          29662 13276
/home/jim/malware_quiz/binaries/shelll
shelll   6887  jim   mem      REG    0,0              0 [heap] (stat: No
such file ordirectory)
shelll   6887  jim   mem      REG    3,6      63276      232 /lib/libresolv-
2.3.5.so
shelll   6887  jim   mem      REG          3,6      38352      228
/lib/libnss_nisplus-2.3.5.so
shelll   6887  jim   mem      REG    3,6      17848      217 /lib/libnss_dns-
2.3.5.so
shelll   6887  jim   mem      REG    3,6      34328      225 /lib/libnss_nis-
2.3.5.so
shelll   6887  jim   mem      REG    3,6      38376      220 /lib/libnss_files-
2.3.5.so
shelll   6887  jim   mem      REG    3,6 1229976      241 /lib/tls/libc-
2.3.5.so
shelll   6887  jim   mem      REG    3,6      68576      204 /lib/libnsl-
2.3.5.so
shelll   6887  jim   mem      REG    3,6 569518      187 /lib/ld-2.3.5.so
shelll   6887  jim    0u     CHR 136,1              3 /dev/pts/1
shelll   6887  jim    1u     CHR 136,1              3 /dev/pts/1
shelll   6887  jim    2u     CHR 136,1              3 /dev/pts/1
shelll   6887  jim    3u     IPv4 15153              TCP 10.0.0.3:47859-
>216.115.95.70:ircd (CLOSE_WAIT)
shelll   6887  jim    4u     IPv4 15186              TCP 10.0.0.3:47860-
>216.115.95.70:ircd (CLOSE_WAIT)
shelll   6887  jim    5u     IPv4 15188              TCP 10.0.0.3:47861-
>216.115.95.70:ircd (CLOSE_WAIT)
shelll   6887  jim    6u     IPv4 15190              TCP 10.0.0.3:47862-
>216.115.95.70:ircd (CLOSE_WAIT)
shelll   6887  jim    7u     IPv4 15192              TCP 10.0.0.3:47863-
>216.115.95.70:ircd (ESTABLISHED)

```

As expected it connected to irc.ircnet.net (216.115.95.70). Since there is no /usr/dict/words file on my Linux workstation it has picked a random nickname. Please note that I did not create a /usr/dict/words files to test this hypothesis, so this remains speculation on my part. Sheer laziness I know, but I saw no merit in spending any time on this. The response from the IRC server shows that it has adapted to these automated botnet tools by restricting, 'random,'

```
:irc1.us.open-ircnet.net 465 NQSBY :You (*@host-84-9-22-204.bulldogdsl.com) are banned from this server: Random drone-like nicks are prohibited.
```

#### **4. Now, what are the purpose of the malware? Are they related?**

Neither of these tools appears to be related to one another insofar as there is no interdependency on one to the other. Of course the initial compromise could have involved uploading cmd.gif to a vulnerable web server and using it to download other tools, including shelll. The purpose of both tools is to provide the attacker with remote access to the host.

#### **5. Why didnt the 'shell' or the 'cmd' applications show up at the ps aux ?**

There could be a rootkit in place on this server that is masking the presence of attackers tool kit in the process listing. Another explanation is that the tools in question were not running. Consider the PHP backdoor cmd.gif. This will only be running while it is in use. If the attacker was not sending any command over this channel when the process listing was taken then it will not appear in the list. However the netstat output showed connections opened to irc.ircnet.net, suggesting that the tool shelll was running at the time that netstat was run. The conclusion is that there was some mechanism such as a rootkit in place to hide the intruder's tracks.

#### **6. Do you have any clues of how the machine was compromised?**

Only two valid accounts are in the passwd.txt file, 'root' and 'backup'. A login as use 'backup' appears in the logs following a number of failed attempts to access other logins that do not exist on this host. It is most likely that the backup use account has a weak password and the attacker has guessed it. Since each login in this brute force attack appears to have been tried only once I suggest that the attack is looking for accounts where the password is the same as the username. To test this out with the 'backup' login I ran it through John the Ripper.

```
[jim@fox run]$ ./john ~/malware_quiz/data/passwd.txt
Loaded 2 password hashes with 2 different salts (OpenBSD Blowfish
[32/32])
backup          (backup)
admin           (root)
guesses: 1 time: 0:00:04:46 100% (2) c/s: 8.36 trying: admin
```

Almost instantly John has guessed the password. John's, "Single crack," mode will use strings found in the password file, including login names, as it's initial search space and uncovers weak password choices such as this. A few minutes more and we have the root password, which is 'admin'. I have to assume at this point that the intruder has gained root access and may have installed a rootkit to hide their activities.

### **7. About the 'shelll' and cmd.gif file, what useful information could you get?**

I've already managed to find a lot of useful information about both of these tools, and I don't have a lot else to offer. It took a core dump of the running shelll process and examined it using strings but I did not find anything of interest. If the timestamps on the two suspicious files are correct they allow us to construct a timeline of the intrusion.

### **8. Using all your creative mind, please, describe the possible attack scenario... :-)**

The initial intrusion occurred at 11:05:50 on 13<sup>th</sup> February and was the result of a successful attempt to brute force the login 'backup' via the SSH service. The user had the same password as the login name. A password crack was run on /etc/passwd and the weak root password obtained. At this point a rootkit was installed

The files cmd.gif and shelll were copied onto the compromised system to provide remote access to host. The 'shelll' command is an IRC bot that will log into a channel on irc.ircnet.net and wait for commands. The attacker used this channel to send a command to the compromised host to send a SYN flood, most likely as part of a distributed DoS attack.

### **9. Based on this attack, which security measures would you recommend to this linux box owner?**

The point of entry for this attacker was easily guessed login credentials. The owner of this Linux box should be made aware of the need for strong passwords on all accounts. This should not be too difficult since the owner has had a practical demonstration!

The point of entry was SSH and there are measures that can be taken to mitigate the risk of running this service. An access control mechanism based on the SSH client's IP address such as a firewall or TCP wrappers can be used to deny access to all hosts save those that are allowed legitimate access. In addition to this I would suggest disabling password authentication for SSH in favour of using SSH keys since this prevents the risk of brute force login guessing attacks. If this is not an option then disabling root logins over SSH can reduce the risk somewhat.

If the file passwd.txt is a copy of the file /etc/passwd then this system did not have shadow passwords enabled. This would have prevented the intruder from reading the

password hashes as a non-root user and escalating their privileges by cracking the root password.

Finally since I believe there is a rootkit installed on this host I would recommend that then host is rebuilt from clean install media and that data is restored from backup. They do have backups, right? It would be ironic if the host was compromised through the account, 'backup,' only to find that no backups were made of important data.