

2008 Malware Challenge (www.malwarechallenge.info)

Analysis by: Dan Roberts, [droberts29/at/gmail/dot/com](mailto:droberts29@gmail.com)

The challenge: *A system administrator within your organization has come to you because a user's PC was infected with malware. Unfortunately, anti-virus is unable to remove the malware. However, the administrator was able to recover the suspected malware executable [malware.zip - MD5 31d2ec3b312d0fd27940aae5c89e3787]. Your job is to analyze the malware.*

Q1: Describe your malware lab.

In this case, I used a Windows XP system running on VMware Server for easy snapshot and rollback of the test system between trials. Since malware is sometimes VM-aware, I additionally have some sacrificial hardware that I can lay a Ghost image down on. Both the VM host and the hardware host are attached to a SOHO router that's air-gapped from the rest of my network to prevent accidental propagation of malware into my LAN or the Internet.

Downloading malware samples is a risky, since a network-attached computer is involved at some point in the process. Several precautions can reduce the likelihood of infection:

- Make sure the computer is fully patched and has up-to-date AV signatures. Since AV will shoot down malware on-sight, I make an exception for a directory on the target media so the whole AV program doesn't have to be stopped.
- Avoid downloading malware using the OS it was written to.
- Keep an eye on process and network connection listings for unusual behavior.
- Use Wget or Firefox with NoScript. The sites that malware come from are often riddled with malicious browser code that could cause an unintended infection.

Samples can be transported into the lab on removable media like a thumb drive.

Tools I used for this analysis included:

- **PE Explorer** – unpacker, unscrambler, disassembler and resource editor all bundled together. (<http://www.heaventools.com/overview.htm>)
- Several tools from **Sysinternals Suite** including **Strings**, **ProcExp** and **TcpView**. (<http://technet.microsoft.com/en-us/sysinternals/default.aspx>)
- **Wireshark** – invaluable network protocol analyzer to watch what's going on on the wire. (<http://www.wireshark.org>)
- **VirusTotal** – online tool that runs an executable through 30+ antivirus engines and several other useful tools for a good first-pass on suspicious files. (<http://www.virustotal.com>)
- **ThreatExpert** – online tool that reports on the behavior of an executable including network, registry and file activity. (<http://www.threatexpert.com>)
- **PEiD** – a signature-based tool to detect the type of packer used on a PE file. (<http://www.peid.info>).
- **Netcat** – the swiss army knife of network tools.. can be used to set up a quick network listener or connect to arbitrary ports on a remote computer. (<http://netcat.sourceforge.net>)
- **Regshot** – captures and compares the contents of a Windows registry and any file directory tree for changes. (<http://sourceforge.net/projects/regshot>)
- **IDA Pro Freeware** – a very handy disassembler and debugger free for non-commercial use. (<http://www.hex-rays.com/idapro/>)

Q2: What information can you gather about the malware without executing it?

Most antivirus software detects this malware as a bot or worm of some kind. Here is a sampling of what was found by VirusTotal:

- Kaspersky: Backdoor.Win32.Rbot.bzf
- Symantec: W32.Spybot.Worm
- Microsoft: Backdoor:Win32/Rbot.gen

The full results can be viewed online at the following URL:

<http://www.virustotal.com/analysis/4bc1fd9e5bdda6ee884eb3018d9d4681>

After unpacking the malware, an inspection of its strings yields a lot of information.

- This malware is identified as Crxbot Alias REalmbot -by Lindem-.
- The malware contains an apparent IRC server hostname, channel name and associated commands, which may mean it uses IRC for command and control.
- There is a laundry list of network and security related registry keys that this malware is programmed to manipulate.
- There is reference to an executable "Winsec32.exe" as well as a Windows service "Microsoft Svchost local services" which may be how the malware survives between reboots.
- There is a dictionary of common usernames and passwords included, in addition to references to several default Windows administrative shares.
- A list of keywords such as "Welcome to Gmail" and "PayPal" may mean that it watches for user activity and can capture credentials and/or account numbers.

Without ever running anything in my own lab, ThreatExpert can report on what sort of behavior the executable exhibits. While we'll still want to run it locally, it's helpful to have a basic idea of what's about to be unleashed in the lab. Amongst other things, several registry entries and a file are created -- specifics are discussed later.

The ThreatExpert results can be viewed online at the following URL:

<http://www.threatexpert.com/report.aspx?md5=59a95f668e1bd00f30fe8c99af675691>

Q3: Is the malware packed? If so, how did you determine what it was?

The malware is packed with UPX. Tools like TrID and PEiD can be used to guess the type of packer that was used. VirusTotal conveniently runs uploaded samples through these tools for a quick overview. In this case, PEiD could not guess what we were dealing with in its normal mode, which is what VirusTotal uses; however running a deeper scan in the lab resulted in detection of UPX. TrID decided that the sample looked mostly like it was packed with Yoda's Crypter, a result of some header obfuscation. PE Explorer saw through this trickery and effortlessly provided an unpacked binary to work with.

Question #4: Describe the malware's behavior. What files does it drop? What registry keys does it create and/or modify? What network connections does it create? How does it auto-start, etc?

Although the bot has the ability to do far more, I was able to detect the following with RegShot upon initial execution:

Drops a file:

- C:\WINDOWS\Winsec32.exe

Creates registry entries:

- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Microsoft Svchost local services: "Winsec32.exe"
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices\Microsoft Svchost local services: "Winsec32.exe"
- HKEY_CURRENT_USER\Software\Microsoft\OLE\Microsoft Svchost local services: "Winsec32.exe"

Makes a network connection to:

- testirc1.sh1xy2bg.NET (tcp/6667)

It is the registry changes and the Winsec32.exe executable stored on the computer that allow this malware to survive a reboot.

Q5: What type of command and control server does the malware use? Describe the server and interface this malware uses as well as the domains and URLs accessed by the malware.

Command and control is via an IRC channel #challenge on the server testirc1.sh1xy2bg.NET. The owner of the botnet interfaces with the bot by sending commands in private messages.

For instance, to authenticate to and begin working with any given bot, a message must be sent to that bot with the following syntax:

```
/msg <botnick> .login <password>
```

The bot responds through IRC to the user commanding it. Additional interfaces are available such as a web server and an ftp server that can be enabled to allow for file transfers to and from the host.

Q6: What commands are present within the malware and what do they do? If possible, take control of the malware and run some of these commands, documenting how you did it.

There is a long list of commands available within this malware. The attacker has complete control over the infected computer, including the ability to upload and download files, grab the contents of the clipboard, capture screenshots, control services, issue remote shell commands, log keystrokes and launch denial of service attacks on other hosts. There even appears to be functionality to lock the infected host down, perhaps to prevent losing control of the drone to another attacker, and also an option to terminate and even uninstall the malware from the host.

Here is a listing of most of the commands, with aliases shown in parenthesis:

action (irc.ac)	flushdns (util.fdns)	privmsg (irc.pm)
advscan (asc)	ftp.upload ()	proc.off (com.ps.off)
aliases (irc.al)	ftpd.on (d.ftpd.on)	proc.on (com.ps)
clearlog (clg)	getclip (com.gc)	prockillid (pkid)
clone.action (clone.ac)	gethost (irc.gh)	quit (irc.q)
clone.join (clone.j)	httpcon (util.hcon)	raw (irc.ra)
clone.make (clone.start)	httpd.on (web.on)	rawclone (clone.ra)
clone.mode (clone.m)	id (irc.i)	readfile (com.rf)
clone.nick (clone.ni)	ident ()	reboot ()
clone.part (clone.p)	irc.nick (irc.n)	reconnect (irc.r)
clone.privmsg (clone.pm)	irc.repeat (irc.rp)	redirect (daemon.rd)
clone.quit (clone.q)	irc.who ()	remove (rm)
clone.rndnick (clone.rn)	join (irc.j)	rename (com.mv)
cmd (cmd1)	keylog.on (cmd.kl.on)	rndnick (rn)
crash ()	killprocess (kpc)	secure (sec)
currentip (cip)	killthreads (killt)	setserver (irc.se)
cycle (irc.cy)	list (com.fl)	stats (st)
delay (irc.de)	log (irc.lg)	status (irc.s)
delete (del)	login (l)	stop (stop)
die (irc.di)	logout (lo)	synflood (ddos.ack)
disconnect (irc.d)	mirc.cmd (mirc.cmd)	sysinfo (sys)
dns (irc.dn)	mode (irc.m)	testdlls (com.dll)
download (dl)	netinfo (ni)	threads (threads.l)
driveinfo (com.driv)	open (com.o)	udpflood (ddos.udpf)
dump ()	opencmd (cmd1)	uptime (com.up)
encrypt (enc)	part (irc.pt)	versionship (ver)
execute (com.e)	pingflood (ddos.pingf)	
flusharp (farp)	prefix (irc.pr)	

A description for many of these commands including arguments and examples can be found on the web. The following is a command reference for a similar variant of Rbot:
<http://www.angelfire.com/theforce/travon1120/RxBotCMDLIST.html>

Controlling the bot requires setting a trap that makes it connect to a host of our choosing instead of the botnet C&C server. This could be accomplished using netcat listening on port 6667, but since the IRC protocol is laborious to type out long-hand, I opted to configure an IRC daemon in my lab. Redirecting the bot to this server simply involves placing an entry in C:\Windows\System32\drivers\etc\hosts with the IP address of the lab daemon.

To keep things simple, I loaded a Windows-based daemon called InspIRCd on the localhost itself. After configuring several options in the inspircd.conf file, I connected to the server and joined channel #challenge. Once the malware was executed, it also joined the channel. I first tried to logon with several possible passwords.

```
/msg USA[XP]4221645 .login somepass  
  
-USA[XP]4221645- Are you a Fucker?. (bothhunter!bothhunter@127.0.0.1).  
-USA[XP]4221645- No pass for you.
```

Several password attempts resulted in the same, until...

```
/msg USA[XP]0166582 .login gemp123  
  
-USA[XP]4221645- WTF!? no yet fucker!. (bothhunter!bothhunter@127.0.0.1).  
-USA[XP]4221645- Orders: No Talk with you.
```

Using the password gemp123 produces a different error, but at this point I'm not sure why because I don't know the program logic. Two options now would be to track down the source code or attach a debugger to the program to see what's going on. I opted first for the debugger, since I now have a better idea of where to set some breakpoints. This pays off.. the first test shows a string comparison involving the login password gemp123, and a second comparison against my hostname and the string *@legalize.it.

Luckily, InspIRCd allows server operators to mask their hostnames. I configure a mask of legalize.it and try to logon again, and this time it works.

```
/whois bothhunter  
bothhunter is bothhunter@legalize.it * bothhunter  
bothhunter is connecting from bothhunter@127.0.0.1 127.0.0.1  
  
/msg USA[XP]4221645 .login gemp123  
<USA[XP]4221645> [REALMBOT] : Thank for trying.
```

This now has unlocked the other commands.

Q7: How would you classify this malware? Why?

Antivirus detections, runtime behavior and strings analysis all indicate that this is a variant of Rbot, a common IRC controlled bot.

An extensive analysis of Rbot can be found on CA's website at the following URL:
<http://www.ca.com/us/securityadvisor/virusinfo/virus.aspx?id=39437>

The findings in this report closely match the characteristics of this malware sample.

Q8: What do you think the purpose of this malware is?

This is a multipurpose bot that allows an attacker to gain control over a host. Natively, it can be leveraged to steal data such as authentication credentials and financial information, or launch further attacks on other hosts. However, since it has file transfer and process controls, it can also be used to further escalate the intrusion to include additional functionality as needed by the attacker.

Bonus Question #1: Is it possible to find the malware's source code? If so, how did you do it?

Using Google search terms "Crxbot source code" provides a link to <http://www.blackhat-forums.com/index.php?showtopic=6032> which contains a RAR archive of many different bots. One of the bots included is "Crx-realmbot.VNC+RFI", which looks a lot like the malware provided for this challenge.

Bonus Question #2: How would you write a custom detection and removal tool to determine if the malware is present on the system and remove it?

This malware doesn't go to great lengths to cloak itself. Killing the process and deleting the associated files and registry keys seems to clean the infected machine up pretty well. Detection of those components looks straightforward too, but we might also employ some IDS signatures, since there are tell-tale strings passed in plaintext over the network.

The problem is that while this malware provides a great deal of functionality by itself, in reality it will serve as a means to an end. Once infected, the botnet owner will rent the bots out to handle all kinds of other distributed processing activities for a fee. This means two things: 1) additional malware is bound to follow and may not be as easy to detect or remove, and 2) this is an important investment that the botnet owner is going to want to protect, so it's possible a kernel level rootkit may be installed to help it hide its tracks.

If we're only concerned about this single piece of malware, then detection and remediation should not be too difficult. But for the above-stated reasons, a host that's been infected should no longer be trusted again.