

Team Lipstick

Malware Challenge 2008 write-up:

- Describe your malware lab:

Stand alone laptop with the following tools installed:

FlexHex	MAPS
Comodo	PEiD
Sandboxie	PE Explorer
Wireshark	IRCD
OllyDbg	XChat

- What information can you gather about the malware without executing it?
 - File created on October 1st 2008
 - Does not run in compatibility mode
 - No summary information present
 - MD5=59a95f668e1bd00f30fe8c99af675691 (Unpacked)
 - MD5=31d2ec3b312d0fd27940aae5c89e3787 (Website)
 - Ran file through VirusScan.org, discovered it has already been processed and identified as IRC Bot.
 - Pulled strings from binary findings are aligned with VirsuScan.org prognosis.
 - Paypal, Yahoo - potentially for targeted attack
 - [home] [down] etc - potentially key logger
 - Appears to identify as Mozilla - potentially browses the web
 - Presence of HTML - potentially builds website
 - Presence of registry keys - potentially modifies the registry
- Is the malware packed? If so, how did you determine what it was?

PEiD detected that the malware was packed. Identifies as UPX Markus & Lazlo

- 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?

The following are observations made after executing the malware:

- Writes file C:\Windows\Winsec32.exe
- File is marked as hidden system protected file
- MD5=59a95f668e1bd00f30fe8c99af675691 (Same MD5 as original sample)
- Winsec32.exe is executed
- Writes file a.bat and 1.reg used to write to registry

- Writes to registry: HKUS\SW\MS\Windows\CV\Run\Microsoft svchost local services to autostart
- Writes to registry: HKUS\SW\MS\Windows\CV\Run\Microsoft svchost local services to autostart as service
- Initiates IRC (TCP 6667) to testirc1.sh1xy2bg.net
- Sniffing the IRC request indicates:
 - o PING 127.0.0.1
 - o PONG 127.0.0.1
 - o JOIN #challenge happy12
 - o topic #challenge :.asc vnc 100 0 0 -r -b
 - o MODE #challenge +mnst
- If windows network connection status is disconnected, the malware does not attempt to connect to IRC.

- 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.

This malware leverages IRC for C&C. It looks for a specific room (#challenge) although the password for the room (happy12) is not required to be set. Upon connecting to the room, the bot listens for a host connecting via *@legalize.it with the password of “gemp123”. Once a user is authenticated, control is obtained and commands can be issued.

- 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.

First using PE Explorer we were able to isolate strings that seemed related to authentication. We did this by reversing our searches from the known error strings as well as common authentication strings (auth, logged, log, success, etc). After identifying the areas in memory that we were primarily interested in, we opened Olly and set breakpoints around them. From here we were able to obtain the needed information such as password and required auth_host. A local instance of IRCd was configured to match the required environment, and host file entries were set to direct the request to testirc1.sh1xy2bg.net onto the loopback.

One obstacle encountered was syntax. After locating the source code, and comparing the data, we were able to successfully authenticate and begin issuing commands.

Confirmed command list:

This is quite a versatile piece of malware. There are over 100 confirmed distinct commands, with 180 distinct execution strings. This table shows the execution strings, see appendix 1 for analysis. The malicious payload is concerned with keylogging, DOS, dataleakage, and continued compromise.

login	l	logout	lo	versionship	ver
speedtest	test	chghhttp	secure	sec	lockdown.off
ld.off	visit	irc.v	web.off	ftpd.off	log.off
proxy.redirect.off	ddos.off	syn.off	udp.off	ping.off	proc.off
com.ps.off	clone.off	secure.stop	scanstop	stats	st
reconnect	irc.r	disconnect	irc.d	quit	irc.q
status	irc.s	id	irc.i	reboot	threads
threads.l	aliases	irc.al	log	irc.lg	clearlog
clg	netinfo	ni	supersyn	sysinfo	sys
remove	rm	proc.on	com.ps	uptime	com.up
driveinfo	com.driv	encrypt	enc	opencmd	cmdl
closecmd	irc.who	getclip	com.gc	die	irc.di
flushdns	util.fdns	currentip	cip	httpd.on	web.on
crash	ftpd.on	d.ftpd.on	irc.nick	irc.n	join
irc.j	part	irc.pt	raw	irc.ra	killthreads
killt	flusharp	farp	clone.rndnick	clone.rn	prefix
irc.pr	open	com.o	setserver	irc.se	dns
irc.dn	killprocess	kpc	prockillid	pkid	delete
del	list	com.fl	mirr.cmd	mirr.cmd	cmd
cmdl	readfile	com.rf	clone.quit	clone.q	keylog.on
cmd.kl.on	net start	net pause	net continue	net delete	net share
net user	net send	gethost	irc.gh	addalias	irc.aa
privmsg	irc.pm	action	irc.ac	cycle	irc.cy
mode	irc.m	rawclone	clone.ra	clone.mode	clone.m
clone.nick	clone.ni	clone.part	clone.p	irc.repeat	irc.rp
delay	irc.de	execute	com.e	rename	com.mv
clone.make	clone.start	synflood	ddos.ack	ddos.random	download
dl	redirect	daemon.rd	clone.privmsg	clone.pm	clone.action
clone.ac	advscan	asc	udpfflood	ddos.udpf	u
pingflood	ddos.pingf	p	httpcon	util.hcon	ftp.upload

➤ How would you classify this malware? Why?

This is an IRC Bot. While it provides some limited Trojan functions, it is mainly IRC focused and appears to be built around the C&C vs. individual element control.

➤ What do you think the purpose of this malware is?

This malware is built for data theft and denial of service attacks. Many of the commands located focused around these areas and no SMTP services were detected during testing.

➤ Is it possible to find the malware's source code? If so, how did you do it?

Yes, we were able to download a copy from www.bottalk.us. It was fairly trivial to uncover the version after reviewing the strings present in the binary “Crxbot Alias REalmbot - by Lindem”

- How would you write a custom detection and removal tool to determine if the malware is present on the system and remove it?

The delivery mechanism of a tool to detect and remove malware will not be discussed due to the broad range of frameworks that can be leveraged. The method would be the same across all frameworks:

- Detect for the malware via:
 - MD5 hash scans to detect executable
 - Monitor network egress for string “.asc vnc” outbound destined for port 6667
- Remove the malware via:
 - Delete file matching MD5 of malware
 - Delete registry entries where the name of the malware executable is present.

Appendix 1

Command details

login || l <pass> - logs you in to have some fun
logout || lo - logs you out no more fun
rndnick || rn - new random nick
die || ircdi || quit || crash- kill winsec32 process
versionship || ver - display version
chghttp <host> - change VNC HTTP host
secure || sec - disables DCOM, restricts \$IPC,C\$, removes network shares
lockdownoff || ldoff - enables DCOM, unrestricts \$IPC,C\$, allows network shares
visit <URL>|| ircv <URL> - gets requested URL (can't get to work)
httpd.on - starts web server
web.off - stops web server
ftpd.on - starts ftp server
ftpdoff - stops ftp server
logoff - stops log thread???
Proxy.redirect.off - terminates tcp redirect thread
ddosoff - terminates DDos flood thread
synoff - terminates SYN flood thread
udpoff - terminates UDP flood thread
pingoff - terminates PING flood thread
procoff ||compsoff - terminates process list thread
cloneoff - terminates clone thread
securestop - terminates secure thread (via secure||sec)
scanstop - terminates VNC sweep threads
stats || st - show exploit scan stats
reconnect || ircr - disconnects then reconnects to IRC
disconnect || ircd - disconnects from IRC winsec32 process continues
status || ircs - display winsec32 uptime
id || ircid - display winsec32 ID
reboot - hard reboot of host
threads || threadsl - display list of threads
aliases || ircal - display list of aliases
log || irclg - display activity log
clearlog || clg - delete activity log
netinfo || ni - display NIC status, IP address, and hostname
supersyn - <targetIP> <targetport> <duration in sec> - execute SYN flood attack against target IP:port for duration
sysinfo || sys - display CPU, RAM free, RAM total, disk free, disk total, OS, SYSDIR, hostname, current user, date, time, uptime
remove || rm - terminates winsec32 and deletes file
procon || comps - display process (PID) list
uptime || comup - display host uptime
driveinfo || comdrv - display host disk free, disk total

testdll || comdll - display DLL test ???
**encrypt || enc
opencmd || cmd1 - launches remote shell
cmd <operation> - execute operation via remote shell result displayed
closecmd - terminate remote shell
ircwho - display logged in users
getclip || comgc - display host clipboard
flusharp || farp - flush ARP cache
flushdns || utilfdns - flush DNS cache
currentip || cip - display current IP address
httpdon || webon - launch HTTP server on port configured in cfgh
ftpdon || dftpdon - launch FTP server (on port 0 ???)
ircnick || ircn <NICK> - changes nick
ircj || join <#CHANNEL> - joins channel
part || ircpt <#CHANNEL> - parts channel
raw || ircra <irc command> - bot issues irc command
**clone.quit || clone.q
**clone.rndnick || clone.rn
prefix || ircpr <letter> - changes command prefix to letter
**setserver || ircse
killprocess || kpc <NAME> - terminates named process
prockillid || pkid <PID> - terminates process by pid
delete <file> - deletes file
list || com.fl <string> - find and display files matching string (use wildcards *)
readfile || com.rf <file> - display contents of file
keylog.on || cmd.kl.on <mode> - logs keystroke based on diff modes : pay, normal
net stop as MSFT command
net pause as MSFT command
net continue as MSFT command
net delete as MSFT command
net share as MSFT command
net user as MSFT command
net send as MSFT command
addalias || irc.aa <arg> <arg> - adds alias ?
privmsg || irc.pm <nick> <msg> - send <nick> a private message <msg>
cycle || irc.cy <#channel1> <#channel2> - joins both channels
mode || irc.m <arg> <arg> - changes mode???
**rawcone <arg> <arg> || clone.ra <arg> <arg> - ???
clone.mode || clone.m <arg> <arg> - ???
clone.nick || clone.ni <arg> <arg> - ???
clone.join || clone.j <arg> <arg> - ???
clone.part || clone.p <arg> <arg> - ???
irc.repeat || irc.rp <arg> <arg> - ???
delay <arg> <arg> || irc.de <arg> <arg> - ???
update <url> <name of exe without .exe> - downloads and executes exe from url
execute || com.e <path> <name of exe without .exe> - executes local file

rename || com.mv <path to orig file> <path to new file> - moves orig file to new path - can't create new dirs
clone.make || clone.start <irchost> <ircport> <ircchannel> - starts new thread (not new process) on specified dest
synflood || ddos.ack || ddos.random <targetIP> <targetport> <duration in sec>
download || dl <srcurl> <destpath> <execute after dl 1/0> - downloads a file from the specified URL saves locally and optionally executes
redirect || daemon.rd <local port> <dest IP> <dest port> - tunnels local port to remote port
clone.privmsg || clone.pm - can't get to work
clone.action || clone.ac - can't get to work
advscan || asc <port> <delay> <minutes> <threads> <startingIP> - sequential scan of port for minutes iterating up from startingIP every delay seconds using specified threads
udpflood || ddos.udpf <targetIP> <#packets> <packetSize> <delay> <port> - send #packets of size to host:port every delay ms
pingflood || ddos.ping <targetIP> <#pings> <packetSize> <delay> - send #pings of size to host every delay ms
httpcon || util.hcon <IP> <port> <HTTPmethod> <URL> <refer> - HTTP connection to IP:port using method to resolve URL with refer
ftp.upload <host> <user> <pass> <option> <file> - puts file to host over ftp as user/pass allows 1 optional command like cd