GIAC Reverse Engineering Malware (GREM)
Practical Assignment
Version 1.0

Malware: msrll.exe

ILOT XII
James M. Balcik

12/8/2004

Table of Contents

Abstract

The intent of this paper is to partially fulfill the requirements of the GREM certificate and to demonstrate my knowledge of “Reverse Engineering Malware”. This paper documents the tools and processes used to analyze the msrll.exe malware.

Laboratory Setup

Hardware

The laboratory hardware consists of an HP OmniBook XE3 laptop running with the following specs:

Intel Pentium III Processor
700 MHz
384MB Memory
10GB Hard Disk Drive
DVD/CD-ROM Drive
1.44MB 3.5" Floppy drive
10/100MB Integrated Network Interface
56Kbps Integrated Modem

Networking

The laboratory networking setup uses a VMware virtual network. The virtual network allows all network activity to be contained on the laptop between the virtual PC's and the host system. To contain the malware fully the physical network interface on the laptop is not plugged in. The Vmware virtual network emulates a hub, which is convenient for sniffing network traffic.

Software

The key software in this laboratory is VMware Workstation 4.5.1. VMware allows you to run multiple operating systems on one physical PC by creating virtual PC's that share the physical resources of the host system.

The laptop's Microsoft Windows 2000 Server SP4 is the Host system that has VMware installed. There are 2 virtual PC's, one running Microsoft Windows XP SP1 and one running Red Hat Linux 9. Each virtual PC has been preconfigured with software tools to analyze the malware. Also, each virtual PC is contained in a folder on the laptop which has been backed up using WinZip for later recovery of the base or clean system. This allows for quick restores to a clean state for each virtual PC. The host system has been imaged using Ghost to allow for complete system recovery of the host system and all virtual systems.

Windows XP SP1 Software Tools:

Ghost
Version: 7.5
By: Symantec
http://www.symantec.com

Description:
Symantec Ghost is a disk imaging software that can backup a entire disk to a image file for later recovery on that image. I used Ghost to image the entire laboratory laptop hard disk so that in the event of infection at the host level I could restore the entire system back to a clean state.

md5sum
Version: GNU textutils 2.0
By: Ulrich Drepper
http://www.gnu.org/software/textutils/textutils.html

Description:
md5sum will calculate the md5 hash of a file. Knowing the md5 hash of a file will allow you to do file comparisons to determine if the files are the exact same. If a file has changed even in the slightest way the md5 hash of the two files should not match therefore revealing that the file has been modified in some way. I used md5sum to do file comparisons on the different copies of the malware msrll.exe.

BinText
Version: 3.00
By: Foundstone Inc.
http://www.foundstone.com

Description:
BinText allows you to view the ASCII text, Unicode text, and resource strings contained in any file. By viewing the ASCII text, Unicode text, and resource strings in a binary file you can begin to get hints about its functionality, if it is packed or unpacked, and the memory addresses of interesting functions. BinText was used in my analysis of the different copies of msrll.exe malware file to gain hints on packing method, functionality, and memory addresses of certain interesting code.

Regshot
Version: 1.61e5 Final
By: TiANWEi
http://regshot.yeah.net
http://regshot.ist.md

Description:
Regshot allows you to take 2 snap shots of the registry on a system and compare them. When you compare the snap shots you will get a list displaying the keys and

values that have been added, deleted, and modified. You can save your snap shots for later comparison. This is useful when you want to figure out what changes a malware made to a system. Regshot was used to first take a snap shot of the system before infection with the msrll.exe malware. Another snap shot of the system was taken after the msrll.exe malware ran. The comparison shows all the changes to the registry keys and values. This helps to figure out what the malware did and what filtering to do in examination of other log files like those from FileMon and RegMon.

FileMon
Version: 6.12
By: Mark Russinovich and Bryce Cogswell of Sysinternals
http://www.sysinternals.com

Description:
FileMon monitors and displays file system changes. You can save the logged changes to a file for later review. This is useful in finding detailed file system access during a specific period of time like during infection with the malware. A key area it shows is attempts not just successful file access. Sometimes the errors are more revealing than the successful entries. FileMon was used to record all file access during the initial infection of the system with the msrll.exe malware.

RegMon
Version: 6.12
By: Mark Russinovich and Bryce Cogswell of Sysinternals
http://www.sysinternals.com

Description:
RegMon will monitor all registry activity and display it on screen. You can save the log to a file for later review. RegMon will show what programs are accessing the registry and what registry keys and values they are reading or writing. RegMon was used to monitor the registry while infecting the system with the msrll.exe malware.

TDIMon
Version: 1.0
By: Mark Russinovich of Sysinternals
http://www.sysinternals.com

Description:
TDIMon is used to monitor TCP and UDP traffic on the system. This can help with monitoring what the malware does with network communications. An example would be opening a port on the system to listen for connections. TDIMon was used to monitor TCP and UDP traffic during the infection of the system with the msrll.exe malware.

Process Explorer
Version: 8.52
By: Mark Russinovich of Sysinternals
http://www.sysinternals.com

Description:
Process Explorer displays processes that are running on the local system along with their PID, description, and company name if any related to the process. It also shows in the lower window pane open handles or dlls depending on what mode it is in. Process Explorer allows you to drill down on each process listed by double clicking on the process. This reveals a great deal of information about the running process like TCP/IP connections or the path to the program that created the process. Process Explorer was used to monitor process the msrll.exe malware created and to end them during certain points of analysis.

AutoRuns
Version: 5.01
By: Mark Russinovich and Bryce Cogswell of Sysinternals
http://www.sysinternals.com

Description:
AutoRuns shows all the registry entries that are running programs during startup of the system. This is a common way for malware to auto start on a system. AutoRuns was used to check for msrll.exe changes to the auto starting entries in the registry.

UltraEdit-32
Version: 10.10a
By: IDM Computer Solutions Inc.
http://www.ultraedit.com

Description:
UltraEdit-32 can edit text, html, hex, and program source code. UltraEdit-32 was used to view files in hex mode and to view saved log files to search using it's advance search features.

nc (Netcat)
Version: 1.10
By: Hobbit
http://www.securityfocus.com/tools/139/scoreit
http://www.securityfocus.com/tools/137
http://netcat.sourceforge.net/

Description:
Netcat or nc is often called the network Swiss army knife because there are many uses for this tool. Netcat was used to transfer snort log files from the Red Hat 9 virtual

system to the Windows XP SP1 virtual system so that they could be viewed in UltraEdit-32 to allow for my preferred method of search and examining the file. Netcat was also used to setup listeners on the Linux system to capture any requests to certain ports from the infected system.

PESniffer
Version: 1.06
By: SkymarShall/CST
http://start.at/skymarshall (Not Active)

Description:
PE-Sniffer can scan a file for various packed executable encodings like ASPack. PE-Sniffer was used to scan msrll.exe malware for the packed executable encodings.

PEInfo
Version: unknown
By: Tom Liston
Not available to public. This tool was obtained from the SANS Reverse Engineering Malware instructor lead on-line training cd-rom.

Description:
PEInfo allows you to see the packed executable structure. By viewing the structure details you maybe given hints as to what method was used to pack the executable. PEInfo was used to figure out what packing method was used on msrll.exe malware.

ASPACKDIE
Version: 1.41
By: y0da
http://y0da.cjb.net

Description:
ASPACKDIE is an ASPACK packed executable unpacker. ASPACKDIE was used to unpack the msrll.exe malware.

IDA Pro
Version: 4.6
By: DataRescue
http://www.datarescue.com

Description:
IDA Pro is a disassembler and debugger. I used IDA Pro to sift through the msrll.exe disassembled code in search for clues of its functionality. I especially liked the flowchart feature which help me find the different decisions branches in the code.

OllyDbg
Version: 1.10
By: Oleh Yuschuk
http://home.t-online.de/home/ollydbg

Description:
OllyDbg is a 32-bit debugger that runs on Windows. OllyDbg was used to analyze msrll.exe disassembled code while running msrll.exe within OllyDbg. This allowed me to set break points at key areas in the code to further understand the functioning of the malware. OllyDbg was also used to patch msrll.exe so that it didn't require a proper password to authenticate.

Red Hat 9 Software:

SNORT
Version: 2.0.4
By: Martin Roesch
http://www.snort.org

Description:
SNORT is a network sniffer and an intrusion detection system or IDS. It is used here as a network sniffer to capture packets on the virtual network for analysis.

ircd
Version: 2.8/hybrid-6.3.1
http://www.ircd-hybrid.com

Description:
ircd is an IRC server daemon that runs on most UNIX based platforms. It is used here to run an irc server on the Red Hat 9 system to provoke additional behavior from the malware.

irc
Version: 20030709
http://www.eterna.com.au/ircii

Description:
irc is an IRC command line client for Unix/Linux. It is used here to interact with the ircd server and to further provoke and analyze the malware.

Properties of the Malware Specimen

Malware File: msrll.exe
File Type: executable
File Size: 41,984 bytes
MD5 Hash: 84acfe96a98590813413122c12c11aaa

Operating Systems Effected:
Microsoft Windows 9x, 2000, XP

Strings Embedded in File:

Fig. A shows the interesting strings found in the msrll.exe malware file before unpacking.

Fig. A

File pos	Mem pos	ID	Text
0000004D	0040004D	0	!This program cannot be run in DOS mode.
00000178	00400178	0	.text
000001A0	004001A0	0	.data
000001F0	004001F0	0	.idata
00000218	00400218	0	.aspack
00000240	00400240	0	.adata

Fig. B shows the interesting strings found in the msrll.exe malware file after it was unpacked using aspackdie.

Fig. B

File pos	Mem pos	ID	Text
00000000	00400000	0	MZ
0000004D	0040004D	0	!This program cannot be run in DOS mode.
00000080	00400080	0	PE
00000178	00400178	0	.text
000001A0	004001A0	0	.data
000001C8	004001C8	0	.bss
000001F0	004001F0	0	.idata
00000218	00400218	0	.aspack
00000240	00400240	0	.adata
00000268	00400268	0	.newIID
0000130D	0040130D	0	PW
00001326	00401326	0	?insmod
0000132E	0040132E	0	?rmmod
00001335	00401335	0	?lsmod
00001399	00401399	0	%s: <mod name>

```
000013A8   004013A8      0   %s: mod list full
000013BA   004013BA      0   %s: err: %u
000013C6   004013C6      0   mod_init
000013CF   004013CF      0   mod_free
000013D8   004013D8      0   %s: cannot init %s
000013EB   004013EB      0   %s: %s loaded (%u)
000013FE   004013FE      0   %s: mod allready loaded
00001416   00401416      0   %s:%s err %u
000015B5   004015B5      0   %s:%s not found
000015C5   004015C5      0   %s: unloading %s
000016AE   004016AE      0   [%u]: %s hinst:%x
00001712   00401712      0   unloading %s
000017A0   004017A0      0   %s: invalid_addr: %s
000017B5   004017B5      0   %s%s [port]
000018E8   004018E8      0   finished %s
00001A40   00401A40      0   %s <ip> <port> <t_time> <delay>
00001B32   00401B32      0   sockopt: %u
00001B3E   00401B3E      0   sendto err: %u
00001B4D   00401B4D      0   sockraw: %u
00001B59   00401B59      0   syn: done
00001FBC   00401FBC      0   %s <ip> <duration> <delay>
00002096   00402096      0   sendto: %u
000020A2   004020A2      0   jolt2: done
00002260   00402260      0   %s <ip> <p size> <duration> <delay>
00002356   00402356      0   Err: %u
0000235E   0040235E      0   smurf done
000025DE   004025DE      0   &err: %u
00002753   00402753      0   ?ping
00002759   00402759      0   ?udp
0000275E   0040275E      0   ?syn
00002763   00402763      0   ?smurf
0000276A   0040276A      0   ?jolt
00002820   00402820      0   PONG :%s
0000299D   0040299D      0   %s!%s@%s
00002B3D   00402B3D      0   %s!%s
00002BD7   00402BD7      0   irc.nick
00002BE0   00402BE0      0   NICK %s
00002C56   00402C56      0   MODE
00002E34   00402E34      0   ?bu
00002EEA   00402EEA      0   NETWORK=
00002FF8   00402FF8      0   irc.pre
000032A8   004032A8      0   %s_
000032AC   004032AC      0   %s
000032B0   004032B0      0   %s__
000032B5   004032B5      0   _%s
000032B9   004032B9      0   __%s
000032BF   004032BF      0   %s
000032C2   004032C2      0   %s
000032C7   004032C7      0   _%s
000032CC   004032CC      0   _%s__
000032D2   004032D2      0   __%s__
000032D9   004032D9      0   __%s___
000032E1   004032E1      0   NICK %s
000032EB   004032EB      0   NICK
000032F0   004032F0      0   %s %s
0000345C   0040345C      0   CmP
000036B0   004036B0      0   irc.chan
00003775   00403775      0   %s %s
```

```
0000377B   0040377B      0   WHO %s
00003A45   00403A45      0   USERHOST %s
00003A52   00403A52      0   logged into %s(%s) as %s
00003B99   00403B99      0   nick.pre
00003BA2   00403BA2      0   %s-%04u
00003BAA   00403BAA      0   irc.user
00003BB3   00403BB3      0   irc.usereal
00003BBF   00403BBF      0   irc.real
00003BC8   00403BC8      0   irc.pass
00003BE0   00403BE0      0   tsend(): connection to %s:%u failed
00003C20   00403C20      0   USER %s localhost 0 :%s
00003C38   00403C38      0   NICK %s
000040BA   004040BA      0   PING
000040BF   004040BF      0   PRIVMSG
000040C7   004040C7      0   001
000040CB   004040CB      0   JOIN
000040D0   004040D0      0   QUIT
000040D5   004040D5      0   352
000040D9   004040D9      0   302
000040DD   004040DD      0   303
000040E1   004040E1      0   005
000040E5   004040E5      0   PART
000040EA   004040EA      0   KICK
000040EF   004040EF      0   353
000040F3   004040F3      0   433
000040F7   004040F7      0   324
000040FD   004040FD      0   t&
00004100   00404100      0   trecv(): Disconnected from %s err:%u
0000446B   0040446B      0   NOTICE
00004472   00404472      0   %s %s :%s
0000447D   0040447D      0   %s
00004711   00404711      0   MODE %s -o+b %s *@%s
00004727   00404727      0   %s
000047E7   004047E7      0   MODE %s -bo %s %s
00004924   00404924      0   %s.key
00004AA8   00404AA8      0   sk#%u %s is dead!
00004ABA   00404ABA      0   s_check: %s dead? pinging...
00004AD7   00404AD7      0   PING :ok
00004B00   00404B00      0   s_check: send error to %s disconnecting
00004B28   00404B28      0   expect the worst
00004B39   00404B39      0   s_check: killing socket %s
00004B54   00404B54      0   irc.knick
00004B5E   00404B5E      0   jtr.%u%s.iso
00004B6B   00404B6B      0   ison %s
00004B74   00404B74      0   servers
00004B7C   00404B7C      0   s_check: trying %s
00004DAA   00404DAA      0   Ph9K@
00004ED5   00404ED5      0   PhkK@
00004F41   00404F41      0   ShtK@
00005052   00405052      0   %s.mode
0000505A   0040505A      0   MODE %s %s
00005078   00405078      0   ShRP@
000050DA   004050DA      0   Sh$I@
0000559F   0040559F      0   aop
000055A3   004055A3      0   mode %s +o %s
000055B2   004055B2      0   akick
000055B8   004055B8      0   mode %s +b %s %s
000055CA   004055CA      0   KICK %s %s
```

```
00005760   00405760      0   irc.pre
00005781   00405781      0   Set an irc sock to preform %s command on
000057AB   004057AB      0     Type
000057B3   004057B3      0   %csklist
000057BC   004057BC      0    to view current sockets, then
000057DC   004057DC      0   %cdccsk
000057E4   004057E4      0    <#>
000058B4   004058B4      0   %s: dll loaded
000058C3   004058C3      0   %s: %d
000059E1   004059E1      0   said %s to %s
000059EF   004059EF      0   usage: %s <target> "text"
00005A74   00405A74      0   %s not on %s
00005A81   00405A81      0   usage: %s <nick> <chan>
00005B1B   00405B1B      0   PASS
00005B20   00405B20      0   %s logged in
00005BA2   00405BA2      0   sys: %s bot: %s
00005BB2   00405BB2      0   preformance counter not avail
00005C2B   00405C2B      0   usage: %s <cmd>
00005C3B   00405C3B      0   %s free'd
00005C45   00405C45      0   unable to free %s
00005CAD   00405CAD      0   later!
00005CB4   00405CB4      0   unable to %s errno:%u
00005D40   00405D40      0   service:%c user:%s inet connection:%c
contype:%s reboot privs:%c
00005DAE   00405DAE      0   ???
00005E1E   00405E1E      0   kill
00005E23   00405E23      0    %-5u %s
00005F8F   00405F8F      0   %s: %s
00005F96   00405F96      0   %s: somefile
000060D4   004060D4      0   host: %s ip: %s
00006269   00406269      0   capGetDriverDescriptionA
00006282   00406282      0   9x
00006285   00406285      0   2k
00006288   00406288      0   XP
0000628B   0040628B      0   XP++
00006292   00406292      0   cpus:%u
0000629B   0040629B      0    CAM
000062A0   004062A0      0   WIN%s (u:%s)%s%s mem:(%u/%u) %u%% %s %s
000065C6   004065C6      0   open
000065CB   004065CB      0   %s: %s (%u)
00006703   00406703      0   NICK
00006708   00406708      0   %s %s
00006754   00406754      0   %s bad args
000067D6   004067D6      0   aop
000067DA   004067DA      0   akick
000067E0   004067E0      0   OP
000067E3   004067E3      0   KICK
000067E8   004067E8      0   %s[%u] %s
000067F2   004067F2      0   %s removed
000067FD   004067FD      0   couldnt find %s
0000680D   0040680D      0   %s added
00006816   00406816      0   %s allready in list
0000682A   0040682A      0   usage: %s +/- <host>
000069EB   004069EB      0   jtram.conf
000069F6   004069F6      0   %s /t %s
000069FF   004069FF      0   jtr.home
00006A08   00406A08      0   %s\%s
00006A0E   00406A0E      0   %s: possibly failed: code %u
```

```
00006A2B   00406A2B      0   %s: possibly failed
00006A3F   00406A3F      0   %s: exec of %s failed err: %u
00006A90   00406A90      0   u.exf
00006CBC   00406CBC      0   jtr.id
00006CC3   00406CC3      0   %s: <url> <id>
00006E79   00406E79      0   %s
00006EBD   00406EBD      0   IRC
00006EC2   00406EC2      0   DCC
00006EC8   00406EC8      0   DATH
00006ED0   00406ED0      0   IATH
00006ED7   00406ED7      0   IREG
00006EDD   00406EDD      0   CLON
00006EE3   00406EE3      0   ICON
00006EE9   00406EE9      0   RNL
00006EEE   00406EEE      0   RBN
00006EF3   00406EF3      0   WSN
00006EF8   00406EF8      0   WCON
00006EFE   00406EFE      0   LSN
00006F03   00406F03      0   SSL
00006F08   00406F08      0   S>S
00006F40   00406F40      0     #%u [fd:%u] %s:%u [%s%s] last:%u
00006F63   00406F63      0      |\=> [n:%s fh:%s] (%s)
00006F7D   00406F7D      0      |
00006F82   00406F82      0      |---[%s] (%u) %s
00006F96   00406F96      0      |     |-[%s%s] [%s]
00006FAD   00406FAD      0      |=> (%s) (%.8x)
00007360   00407360      0   %s <pass> <salt>
000073C8   004073C8      0   %s <nick> <chan>
00007435   00407435      0   !%s!
0000748B   0040748B      0   PING %s
000074C9   004074C9      0   mIRC v6.12 Khaled Mardam-Bey
000074E7   004074E7      0   VERSION %s
0000751C   0040751C      0   dcc.pass
00007525   00407525      0   temp add %s
0000766A   0040766A      0   %s%u-%s
00007672   00407672      0   wb
00007675   00407675      0   %s opened (%u)
000076A0   004076A0      0   %u bytes from %s in %u seconds saved to %s
000076CB   004076CB      0   (%s %s): incomplete! %u bytes
000076E9   004076E9      0   couldnt open %s err:%u
00007700   00407700      0   (%s) %s: %s
0000770C   0040770C      0   (%s) urlopen failed
00007720   00407720      0   (%s): inetopen failed
00007BE4   00407BE4      0   no file name in %s
00007DDB   00407DDB      0   %s created
00007E49   00407E49      0   %s %s to %s Ok
00007EE0   00407EE0      0   %0.2u/%0.2u/%0.2u  %0.2u:%0.2u  %15s  %s
00007F09   00407F09      0   %s (err: %u)
00008085   00408085      0   err: %u
000080F8   004080F8      0   %s %s :ok
00008165   00408165      0   unable to %s %s (err: %u)
000081F5   004081F5      0   %-16s %s
00008200   00408200      0   %-16s (%u.%u.%u.%u)
00008489   00408489      0   [%s][%s] %s
00008595   00408595      0   closing %u [%s:%u]
000085A8   004085A8      0   unable to close socket %u
000087E2   004087E2      0   using sock #%u %s:%u (%s)
000087FD   004087FD      0   Invalid sock
```

```
0000880B   0040880B      0   usage %s <socks #>
000088D7   004088D7      0   leaves %s
000088E1   004088E1      0   :0 * * :%s
000088EC   004088EC      0   hmm
00008A96   00408A96      0   joins: %s
00008B7D   00408B7D      0   chat
00008B82   00408B82      0   ACCEPT
00008B89   00408B89      0   resume
00008B90   00408B90      0   err: %u
00008B99   00408B99      0   DCC ACCEPT %s %s %s
00008BAE   00408BAE      0   dcc_resume: cant find port %s
00008BCC   00408BCC      0   send
00008BD1   00408BD1      0   dcc.dir
00008BD9   00408BD9      0   %s\%s\%s\%s
00008BE5   00408BE5      0   unable to open (%s): %u
00008BFD   00408BFD      0   resuming dcc from %s to %s
00008C19   00408C19      0   DCC RESUME %s %s %u
00009345   00409345      0   ?si
00009349   00409349      0   ?ssl
0000934E   0040934E      0   ?clone
00009355   00409355      0   ?clones
0000935D   0040935D      0   ?login
00009364   00409364      0   ?uptime
0000936C   0040936C      0   ?reboot
00009374   00409374      0   ?status
0000937C   0040937C      0   ?jump
00009382   00409382      0   ?nick
00009388   00409388      0   ?echo
0000938E   0040938E      0   ?hush
00009394   00409394      0   ?wget
0000939A   0040939A      0   ?join
000093A0   004093A0      0   ?op
000093A4   004093A4      0   ?aop
000093A9   004093A9      0   ?akick
000093B0   004093B0      0   ?part
000093B6   004093B6      0   ?dump
000093BC   004093BC      0   ?set
000093C1   004093C1      0   ?die
000093C6   004093C6      0   ?md5p
000093CC   004093CC      0   ?free
000093D2   004093D2      0   ?raw
000093D7   004093D7      0   ?update
000093DF   004093DF      0   ?hostname
000093E9   004093E9      0   ?fif
000093EE   004093EE      0   ?!fif
000093F4   004093F4      0   ?del
000093F9   004093F9      0   ?pwd
000093FE   004093FE      0   ?play
00009404   00409404      0   ?copy
0000940A   0040940A      0   ?move
00009410   00409410      0   ?dir
00009415   00409415      0   ?sums
0000941B   0040941B      0   ?ls
0000941F   0040941F      0   ?cd
00009423   00409423      0   ?rmdir
0000942A   0040942A      0   ?mkdir
00009431   00409431      0   ?run
00009436   00409436      0   ?exec
```

```
0000943C   0040943C      0   ?ps
00009440   00409440      0   ?kill
00009446   00409446      0   ?killall
0000944F   0040944F      0   ?crash
00009456   00409456      0   ?dcc
0000945B   0040945B      0   ?get
00009460   00409460      0   ?say
00009465   00409465      0   ?msg
0000946A   0040946A      0   ?kb
0000946E   0040946E      0   ?sklist
00009476   00409476      0   ?unset
0000947D   0040947D      0   ?uattr
00009484   00409484      0   ?dccsk
0000948B   0040948B      0   ?con
00009490   00409490      0   ?killsk
00009499   00409499      0   VERSION*
000094A3   004094A3      0   DCC
000094A8   004094A8      0   PING
000094AE   004094AE      0   IDENT
000096BE   004096BE      0   %ud %02uh %02um %02us
000096D4   004096D4      0   %02uh %02um %02us
000096E6   004096E6      0   %um %02us
000099E0   004099E0      0   jtram.conf
000099EB   004099EB      0   jtr.*
000099F1   004099F1      0   set
000099F5   004099F5      0   DiCHFc2ioiVmb3cb4zZ7zWZH1oM=
00009A16   00409A16      0   conf_dump: wrote %u lines
00009C8C   00409C8C      0   set
0000A270   0040A270      0   get of %s incomplete at %u bytes
0000A2B0   0040A2B0      0   get of %s completed (%u bytes), %u seconds %u
cps
0000A2F0   0040A2F0      0   error while writing to %s (%u)
0000A373   0040A373      0   Php
0000A65C   0040A65C      0   chdir: %s -> %s (%u)
0000A750   0040A750      0   dcc_wait: get of %s from %s timed out
0000A790   0040A790      0   dcc_wait: closing [#%u] %s:%u (%s)
0000A9DC   0040A9DC      0   SEND
0000A9E1   0040A9E1      0   GET
0000A9F0   0040A9F0      0   %4s #%.2u %s %ucps %u%% [sk#%u] %s
0000AA30   0040AA30      0   %u Send(s) %u Get(s) (%u transfer(s) total)
UP:%ucps DOWN:%ucps Total:%ucps
0000ACD0   0040ACD0      0   send of %s incomplete at %u bytes
0000AD10   0040AD10      0   send of %s completed (%u bytes), %u seconds %u
cps
0000AF50   0040AF50      0   cant open %s (err:%u) pwd:{%s}
0000AF70   0040AF70      0   DCC SEND %s %u %u %u
0000B751   0040B751      0   %s %s
0000B757   0040B757      0   %s exited with code %u
0000B76E   0040B76E      0   %s\%s
0000B774   0040B774      0   %s: %s
0000B77B   0040B77B      0   exec: Error:%u pwd:%s cmd:%s
0000BB40   0040BB40      0   dcc.pass
0000BB49   0040BB49      0   bot.port
0000BB52   0040BB52      0   %s bad pass from "%s"@%s
0000BCC9   0040BCC9      0   %s: connect from %s
0000BD33   0040BD33      0   jtr.bin
0000BD3B   0040BD3B      0   msrll.exe
0000BD45   0040BD45      0   jtr.home
```

```
0000BD4E   0040BD4E      0   mfm
0000BD52   0040BD52      0   2200
0000BD57   0040BD57      0   jtr.id
0000BD5E   0040BD5E      0   run5
0000BD63   0040BD63      0   irc.quit
0000BD6E   0040BD6E      0   servers
0000BD80   0040BD80      0
collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy0.com:8080
0000BDCA   0040BDCA      0   irc.chan
0000BDD3   0040BDD3      0   #mils
0000BDD9   0040BDD9      0   pass
0000BDE0   0040BDE0      0   $1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0
0000BE20   0040BE20      0   $1$KZLPLKDf$55isA1ITvamR7bjAdBziX.
0000BE43   0040BE43      0   m220
0000C02F   0040C02F      0   SSL_get_error
0000C03D   0040C03D      0   SSL_load_error_strings
0000C054   0040C054      0   SSL_library_init
0000C065   0040C065      0   SSLv3_client_method
0000C079   0040C079      0   SSL_set_connect_state
0000C08F   0040C08F      0   SSL_CTX_new
0000C09B   0040C09B      0   SSL_new
0000C0A3   0040C0A3      0   SSL_set_fd
0000C0AE   0040C0AE      0   SSL_connect
0000C0BA   0040C0BA      0   SSL_write
0000C0C4   0040C0C4      0   SSL_read
0000C0CD   0040C0CD      0   SSL_shutdown
0000C0DA   0040C0DA      0   SSL_free
0000C0E3   0040C0E3      0   SSL_CTX_free
0000C263   0040C263      0   kernel32.dll
0000C270   0040C270      0   QueryPerformanceCounter
0000C288   0040C288      0   QueryPerformanceFrequency
0000C2A2   0040C2A2      0   RegisterServiceProcess
0000C2B9   0040C2B9      0   jtram.conf
0000C5B1   0040C5B1      0   irc.user
0000C5BA   0040C5BA      0   %s : USERID : UNIX : %s
0000C6A4   0040C6A4      0   QUIT :FUCK %u
0000C742   0040C742      0   Killed!? Arrg! [%u]
0000C756   0040C756      0   QUIT :%s
0000C7E8   0040C7E8      0   SeShutdownPrivilege
0000C888   0040C888      0   %s\%s
0000C88E   0040C88E      0   %s\%s\%s
0000C897   0040C897      0   Rll enhanced drive
0000C8C0   0040C8C0      0   software\microsoft\windows\currentversion\run
0000C8EE   0040C8EE      0   /d "%s"
0000C8F8   0040C8F8      0   open
0000CF53   0040CF53      0   fNO
0000CF90   0040CF90      0   NO
0000D010   0040D010      0
./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
0000EA60   0040EA60      0   usage %s: server[:port] amount
0000EB2F   0040EB2F      0   all
0000EB33   0040EB33      0   %s: %s
0000EB3A   0040EB3A      0   die
0000EB3E   0040EB3E      0   %s %s %s <PARAM>
0000EB4F   0040EB4F      0   JOIN
0000EB55   0040EB55      0   %s
0000EB58   0040EB58      0   PART
0000EB5D   0040EB5D      0   raw
```

```
0000EB61   0040EB61      0   %s
0000EB65   0040EB65      0   msg
0000EB80   0040EB80      0   %s: [NETWORK|all] %s <"parm"> ...
0000EE20   0040EE20      0   USER %s localhost 0 :%s
0000EE38   0040EE38      0   NICK %s
0000F100   0040F100      0   md5
0000F140   0040F140      0   md5.c
0000F146   0040F146      0   md != NULL
0000F8F1   0040F8F1      0   buf != NULL
0000F99F   0040F99F      0   hash != NULL
0000FAC1   0040FAC1      0   abc
0000FAC5   0040FAC5      0   message digest
0000FAD4   0040FAD4      0   abcdefghijklmnopqrstuvwxyz
0000FB00   0040FB00      0
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
0000FB40   0040FB40      0
1234567890123456789012345678901234567890123456789012345678901234567890123456789012345
67890
0000FD11   0040FD11      0   sprng.c
0000FD19   0040FD19      0   buf != NULL
0000FD70   0040FD70      0   rc6
0000FDBC   0040FDBC      0   rc6.c
0000FDC2   0040FDC2      0   skey != NULL
0000FDCF   0040FDCF      0   key != NULL
0000FFD1   0040FFD1      0   ct != NULL
0000FFDC   0040FFDC      0   pt != NULL
000103C3   004103C3      0   desired_keysize != NULL
00010430   00410430      0   ctr.c
00010436   00410436      0   ctr != NULL
00010442   00410442      0   key != NULL
0001044E   0041044E      0   count != NULL
00010546   00410546      0   ct != NULL
00010551   00410551      0   pt != NULL
000106F0   004106F0      0
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
0001077F   0041077F      0   ?456789:;<=
000107B7   004107B7      0    !"#$%&'()*+,-./0123
00010850   00410850      0   base64.c
00010859   00410859      0   outlen != NULL
00010868   00410868      0   out != NULL
00010874   00410874      0   in != NULL
00010B30   00410B30      0   _ARGCHK '%s' failure on line %d of file %s
00010B8B   00410B8B      0   crypt.c
00010B93   00410B93      0   name != NULL
00010D79   00410D79      0   cipher != NULL
00010E70   00410E70      0   hash != NULL
00010F7A   00410F7A      0   prng != NULL
000110F0   004110F0      0   LibTomCrypt 0.83
00011102   00411102      0   Endianess: little (32-bit words)
00011123   00411123      0   Clean stack: disabled
00011139   00411139      0   Ciphers built-in:
0001114B   0041114B      0      Blowfish
00011157   00411157      0      RC2
0001115E   0041115E      0      RC5
00011165   00411165      0      RC6
0001116C   0041116C      0      Serpent
00011177   00411177      0      Safer+
00011181   00411181      0      Safer
```

```
0001118A   0041118A      0      Rijndael
00011196   00411196      0      XTEA
0001119E   0041119E      0      Twofish
000111AA   004111AA      0      CAST5
000111B3   004111B3      0      Noekeon
000111BF   004111BF      0   Hashes built-in:
000111D0   004111D0      0      SHA-512
000111DB   004111DB      0      SHA-384
000111E6   004111E6      0      SHA-256
000111F1   004111F1      0      TIGER
000111FA   004111FA      0      SHA1
00011202   00411202      0      MD5
00011209   00411209      0      MD4
00011210   00411210      0      MD2
00011218   00411218      0   Block Chaining Modes:
0001122E   0041122E      0      CFB
00011235   00411235      0      OFB
0001123C   0041123C      0      CTR
00011244   00411244      0   PRNG:
0001124A   0041124A      0      Yarrow
00011254   00411254      0      SPRNG
0001125D   0041125D      0      RC4
00011265   00411265      0   PK Algs:
0001126E   0041126E      0      RSA
00011275   00411275      0      DH
0001127B   0041127B      0      ECC
00011282   00411282      0      KR
00011289   00411289      0   Compiler:
00011293   00411293      0      WIN32 platform detected.
000112AF   004112AF      0      GCC compiler detected.
000112CA   004112CA      0   Various others:  BASE64  MPI  HMAC
00011313   00411313      0   /dev/random
00011430   00411430      0   Microsoft Base Cryptographic Provider v1.0
000114D2   004114D2      0   bits.c
000114D9   004114D9      0   buf != NULL
0001154A   0041154A      0   prng != NULL
00011A10   00411A10      0   -LIBGCCW32-EH-SJLJ-GTHR-MINGW32
00012091   00412091      0   %|
000120C1   004120C1      0   %$
000120E1   004120E1      0   %T
000120F1   004120F1      0   %h
00012111   00412111      0   %p
00012141   00412141      0   %4
00012151   00412151      0   %\
00012161   00412161      0   %t
000121A1   004121A1      0   %d
000121E1   004121E1      0   %(
000121F1   004121F1      0   %@
00012201   00412201      0   %X
00012231   00412231      0   %<
00012241   00412241      0   %D
00012251   00412251      0   %H
000122B1   004122B1      0   %l
000122C1   004122C1      0   %L
000122D1   004122D1      0   %8
000122E1   004122E1      0   %P
000122F1   004122F1      0   %x
00012301   00412301      0   %,
```

```
00012341   00412341      0   %\
00012351   00412351      0   %h
00012361   00412361      0   %8
00012371   00412371      0   %|
00012431   00412431      0   %t
00012451   00412451      0   %D
00012461   00412461      0   %p
00012491   00412491      0   %P
000124A1   004124A1      0   %
000124B1   004124B1      0   %x
000124F1   004124F1      0   %0
00012501   00412501      0   %$
00012511   00412511      0   %4
00012521   00412521      0   %,
00012591   00412591      0   %X
000125A1   004125A1      0   %T
000125E1   004125E1      0   %H
00012641   00412641      0   %L
00012671   00412671      0   %@
00012681   00412681      0   %l
00012691   00412691      0   %d
000126A1   004126A1      0   %(
000126C1   004126C1      0   %<
000130B0   004130B0      0    <ip> <total secs> <p size> <delay>
00013350   00413350      0    modem
00013358   00413358      0    Lan
0001335E   0041335E      0    Proxy
00013366   00413366      0    ??
0001336B   0041336B      0    none
00013390   00413390      0   m220 1.0 #2730 Mar 16 11:47:38 2004
000133D4   004133D4      0   unable to %s %s (err: %u)
00013420   00413420      0   unable to kill %s (%u)
00013437   00413437      0   %s killed (pid:%u)
00013470   00413470      0   AVICAP32.dll
0001347D   0041347D      0   unable to kill %u (%u)
00013494   00413494      0   pid %u killed
000134A2   004134A2      0   error!
000134A9   004134A9      0   ran ok
000134B0   004134B0      0   MODE %s +o %s
000134BF   004134BF      0   set %s %s
00013600   00413600      0   Mozilla/4.0
0001360C   0041360C      0   Accept: */*
0001361C   0041361C      0   <DIR>
0001362B   0041362B      0   Could not copy %s to %s
00013643   00413643      0   %s copied to %s
00013653   00413653      0   0123456789abcdef
00013664   00413664      0   %s unset
0001366D   0041366D      0   unable to unset %s
00013AD4   00413AD4      0   (%s) %s
00013ADD   00413ADD      0   %s %s
00013B30   00413B30      0   #:
00013BA0   00413BA0      0   libssl32.dll
00013BAD   00413BAD      0   libeay32.dll
00013BE0   00413BE0      0   <die|join|part|raw|msg>
0011B67A   0051B67A      0   AdjustTokenPrivileges
0011B692   0051B692      0   CloseServiceHandle
0011B6AA   0051B6AA      0   CreateServiceA
0011B6BE   0051B6BE      0   CryptAcquireContextA
```

0011B6D6	0051B6D6	0	CryptGenRandom
0011B6EA	0051B6EA	0	CryptReleaseContext
0011B702	0051B702	0	GetUserNameA
0011B712	0051B712	0	LookupPrivilegeValueA
0011B72A	0051B72A	0	OpenProcessToken
0011B73E	0051B73E	0	OpenSCManagerA
0011B752	0051B752	0	RegCloseKey
0011B762	0051B762	0	RegCreateKeyExA
0011B776	0051B776	0	RegSetValueExA
0011B78A	0051B78A	0	RegisterServiceCtrlHandlerA
0011B7AA	0051B7AA	0	SetServiceStatus
0011B7BE	0051B7BE	0	StartServiceCtrlDispatcherA
0011B7DE	0051B7DE	0	AddAtomA
0011B7EA	0051B7EA	0	CloseHandle
0011B7FA	0051B7FA	0	CopyFileA
0011B806	0051B806	0	CreateDirectoryA
0011B81A	0051B81A	0	CreateFileA
0011B82A	0051B82A	0	CreateMutexA
0011B83A	0051B83A	0	CreatePipe
0011B84A	0051B84A	0	CreateProcessA
0011B85E	0051B85E	0	CreateToolhelp32Snapshot
0011B87A	0051B87A	0	DeleteFileA
0011B88A	0051B88A	0	DuplicateHandle
0011B89E	0051B89E	0	EnterCriticalSection
0011B8B6	0051B8B6	0	ExitProcess
0011B8C6	0051B8C6	0	ExitThread
0011B8D6	0051B8D6	0	FileTimeToSystemTime
0011B8EE	0051B8EE	0	FindAtomA
0011B8FA	0051B8FA	0	FindClose
0011B906	0051B906	0	FindFirstFileA
0011B91A	0051B91A	0	FindNextFileA
0011B92A	0051B92A	0	FreeLibrary
0011B93A	0051B93A	0	GetAtomNameA
0011B94A	0051B94A	0	GetCommandLineA
0011B95E	0051B95E	0	GetCurrentDirectoryA
0011B976	0051B976	0	GetCurrentProcess
0011B98A	0051B98A	0	GetCurrentThreadId
0011B9A2	0051B9A2	0	GetExitCodeProcess
0011B9BA	0051B9BA	0	GetFileSize
0011B9CA	0051B9CA	0	GetFullPathNameA
0011B9DE	0051B9DE	0	GetLastError
0011B9EE	0051B9EE	0	GetModuleFileNameA
0011BA06	0051BA06	0	GetModuleHandleA
0011BA1A	0051BA1A	0	GetProcAddress
0011BA2E	0051BA2E	0	GetStartupInfoA
0011BA42	0051BA42	0	GetSystemDirectoryA
0011BA5A	0051BA5A	0	GetSystemInfo
0011BA6A	0051BA6A	0	GetTempPathA
0011BA7A	0051BA7A	0	GetTickCount
0011BA8A	0051BA8A	0	GetVersionExA
0011BA9A	0051BA9A	0	GlobalMemoryStatus
0011BAB2	0051BAB2	0	InitializeCriticalSection
0011BACE	0051BACE	0	IsBadReadPtr
0011BADE	0051BADE	0	LeaveCriticalSection
0011BAF6	0051BAF6	0	LoadLibraryA
0011BB06	0051BB06	0	MoveFileA
0011BB12	0051BB12	0	OpenProcess
0011BB22	0051BB22	0	PeekNamedPipe

0011BB32	0051BB32	0	Process32First
0011BB46	0051BB46	0	Process32Next
0011BB56	0051BB56	0	QueryPerformanceFrequency
0011BB72	0051BB72	0	ReadFile
0011BB7E	0051BB7E	0	ReleaseMutex
0011BB8E	0051BB8E	0	RemoveDirectoryA
0011BBA2	0051BBA2	0	SetConsoleCtrlHandler
0011BBBA	0051BBBA	0	SetCurrentDirectoryA
0011BBD2	0051BBD2	0	SetFilePointer
0011BBE6	0051BBE6	0	SetUnhandledExceptionFilter
0011BC06	0051BC06	0	Sleep
0011BC0E	0051BC0E	0	TerminateProcess
0011BC22	0051BC22	0	WaitForSingleObject
0011BC3A	0051BC3A	0	WriteFile
0011BC46	0051BC46	0	_itoa
0011BC4E	0051BC4E	0	_stat
0011BC56	0051BC56	0	_strdup
0011BC62	0051BC62	0	_stricmp
0011BC6E	0051BC6E	0	__getmainargs
0011BC7E	0051BC7E	0	__p__environ
0011BC8E	0051BC8E	0	__p__fmode
0011BC9E	0051BC9E	0	__set_app_type
0011BCB2	0051BCB2	0	_beginthread
0011BCC2	0051BCC2	0	_cexit
0011BCCE	0051BCCE	0	_errno
0011BCDA	0051BCDA	0	_fileno
0011BCE6	0051BCE6	0	_iob
0011BCEE	0051BCEE	0	_onexit
0011BCFA	0051BCFA	0	_setmode
0011BD06	0051BD06	0	_vsnprintf
0011BD16	0051BD16	0	abort
0011BD1E	0051BD1E	0	atexit
0011BD2A	0051BD2A	0	atoi
0011BD32	0051BD32	0	clock
0011BD3A	0051BD3A	0	fclose
0011BD46	0051BD46	0	fflush
0011BD52	0051BD52	0	fgets
0011BD5A	0051BD5A	0	fopen
0011BD62	0051BD62	0	fprintf
0011BD6E	0051BD6E	0	fread
0011BD76	0051BD76	0	free
0011BD7E	0051BD7E	0	fwrite
0011BD8A	0051BD8A	0	malloc
0011BD96	0051BD96	0	memcpy
0011BDA2	0051BDA2	0	memset
0011BDAE	0051BDAE	0	printf
0011BDBA	0051BDBA	0	raise
0011BDC2	0051BDC2	0	rand
0011BDCA	0051BDCA	0	realloc
0011BDD6	0051BDD6	0	setvbuf
0011BDE2	0051BDE2	0	signal
0011BDEE	0051BDEE	0	sprintf
0011BDFA	0051BDFA	0	srand
0011BE02	0051BE02	0	strcat
0011BE0E	0051BE0E	0	strchr
0011BE1A	0051BE1A	0	strcmp
0011BE26	0051BE26	0	strcpy
0011BE32	0051BE32	0	strerror

0011BE3E	0051BE3E	0	strncat
0011BE4A	0051BE4A	0	strncmp
0011BE56	0051BE56	0	strncpy
0011BE62	0051BE62	0	strstr
0011BE6E	0051BE6E	0	time
0011BE76	0051BE76	0	toupper
0011BE82	0051BE82	0	ShellExecuteA
0011BE92	0051BE92	0	DispatchMessageA
0011BEA6	0051BEA6	0	ExitWindowsEx
0011BEB6	0051BEB6	0	GetMessageA
0011BEC6	0051BEC6	0	PeekMessageA
0011BED6	0051BED6	0	GetFileVersionInfoA
0011BEEE	0051BEEE	0	VerQueryValueA
0011BF02	0051BF02	0	InternetCloseHandle
0011BF1A	0051BF1A	0	InternetGetConnectedState
0011BF36	0051BF36	0	InternetOpenA
0011BF46	0051BF46	0	InternetOpenUrlA
0011BF5A	0051BF5A	0	InternetReadFile
0011BF6E	0051BF6E	0	WSAGetLastError
0011BF82	0051BF82	0	WSASocketA
0011BF92	0051BF92	0	WSAStartup
0011BFA2	0051BFA2	0	__WSAFDIsSet
0011BFB2	0051BFB2	0	accept
0011BFBE	0051BFBE	0	bind
0011BFC6	0051BFC6	0	closesocket
0011BFD6	0051BFD6	0	connect
0011BFE2	0051BFE2	0	gethostbyaddr
0011BFF2	0051BFF2	0	gethostbyname
0011C002	0051C002	0	gethostname
0011C012	0051C012	0	getsockname
0011C022	0051C022	0	htonl
0011C02A	0051C02A	0	htons
0011C032	0051C032	0	inet_addr
0011C03E	0051C03E	0	inet_ntoa
0011C04A	0051C04A	0	ioctlsocket
0011C05A	0051C05A	0	listen
0011C066	0051C066	0	ntohl
0011C06E	0051C06E	0	recv
0011C076	0051C076	0	select
0011C082	0051C082	0	send
0011C08A	0051C08A	0	sendto
0011C096	0051C096	0	setsockopt
0011C0A6	0051C0A6	0	shutdown
0011C0B2	0051C0B2	0	socket
0011C0FC	0051C0FC	0	ADVAPI32.DLL
0011C1FC	0051C1FC	0	KERNEL32.dll
0011C21C	0051C21C	0	msvcrt.dll
0011C2E0	0051C2E0	0	msvcrt.dll
0011C2F0	0051C2F0	0	SHELL32.DLL
0011C30C	0051C30C	0	USER32.dll
0011C320	0051C320	0	VERSION.dll
0011C340	0051C340	0	WININET.DLL
0011C3B4	0051C3B4	0	WS2_32.DLL
0011D071	0051D071	0	VirtualAlloc
0011D07E	0051D07E	0	VirtualFree
0011D441	0051D441	0	kernel32.dll
0011D44E	0051D44E	0	ExitProcess
0011D45A	0051D45A	0	user32.dll

```
0011D465   0051D465      0   MessageBoxA
0011D471   0051D471      0   wsprintfA
0011D47B   0051D47B      0   LOADER ERROR
0011D488   0051D488      0   The procedure entry point %s could not be
located in the dynamic link library %s
0011D4D9   0051D4D9      0   The ordinal %u could not be located in the
dynamic link library %s
0011DF6C   0051DF6C      0   kernel32.dll
0011DF7B   0051DF7B      0   GetProcAddress
0011DF8C   0051DF8C      0   GetModuleHandleA
0011DF9F   0051DF9F      0   LoadLibraryA
0011E074   0051E074      0   advapi32.dll
0011E081   0051E081      0   msvcrt.dll
0011E08C   0051E08C      0   msvcrt.dll
0011E097   0051E097      0   shell32.dll
0011E0A3   0051E0A3      0   user32.dll
0011E0AE   0051E0AE      0   version.dll
0011E0BA   0051E0BA      0   wininet.dll
0011E0C6   0051E0C6      0   ws2_32.dll
0011E113   0051E113      0   AdjustTokenPrivileges
0011E12B   0051E12B      0   _itoa
0011E133   0051E133      0   __getmainargs
0011E143   0051E143      0   ShellExecuteA
0011E153   0051E153      0   DispatchMessageA
0011E166   0051E166      0   GetFileVersionInfoA
0011E17C   0051E17C      0   InternetCloseHandle
0011E192   0051E192      0   WSAGetLastError
001200C8   005200C8      0   advapi32.dll
001200D7   005200D7      0   AdjustTokenPrivileges
001200EF   005200EF      0   CloseServiceHandle
00120104   00520104      0   CreateServiceA
00120115   00520115      0   CryptAcquireContextA
0012012C   0052012C      0   CryptGenRandom
0012013D   0052013D      0   CryptReleaseContext
00120153   00520153      0   GetUserNameA
00120162   00520162      0   LookupPrivilegeValueA
0012017A   0052017A      0   OpenProcessToken
0012018D   0052018D      0   OpenSCManagerA
0012019E   0052019E      0   RegCloseKey
001201AC   005201AC      0   RegCreateKeyExA
001201BE   005201BE      0   RegSetValueExA
001201CF   005201CF      0   RegisterServiceCtrlHandlerA
001201ED   005201ED      0   SetServiceStatus
00120200   00520200      0   StartServiceCtrlDispatcherA
0012021C   0052021C      0   kernel32.dll
0012022B   0052022B      0   AddAtomA
00120236   00520236      0   CloseHandle
00120244   00520244      0   CopyFileA
00120250   00520250      0   CreateDirectoryA
00120263   00520263      0   CreateFileA
00120271   00520271      0   CreateMutexA
00120280   00520280      0   CreatePipe
0012028D   0052028D      0   CreateProcessA
0012029E   0052029E      0   CreateToolhelp32Snapshot
001202B9   005202B9      0   DeleteFileA
001202C7   005202C7      0   DuplicateHandle
001202D9   005202D9      0   EnterCriticalSection
001202F0   005202F0      0   ExitProcess
```

001202FE	005202FE	0	ExitThread
0012030B	0052030B	0	FileTimeToSystemTime
00120322	00520322	0	FindAtomA
0012032E	0052032E	0	FindClose
0012033A	0052033A	0	FindFirstFileA
0012034B	0052034B	0	FindNextFileA
0012035B	0052035B	0	FreeLibrary
00120369	00520369	0	GetAtomNameA
00120378	00520378	0	GetCommandLineA
0012038A	0052038A	0	GetCurrentDirectoryA
001203A1	005203A1	0	GetCurrentProcess
001203B5	005203B5	0	GetCurrentThreadId
001203CA	005203CA	0	GetExitCodeProcess
001203DF	005203DF	0	GetFileSize
001203ED	005203ED	0	GetFullPathNameA
00120400	00520400	0	GetLastError
0012040F	0052040F	0	GetModuleFileNameA
00120424	00520424	0	GetModuleHandleA
00120437	00520437	0	GetProcAddress
00120448	00520448	0	GetStartupInfoA
0012045A	0052045A	0	GetSystemDirectoryA
00120470	00520470	0	GetSystemInfo
00120480	00520480	0	GetTempPathA
0012048F	0052048F	0	GetTickCount
0012049E	0052049E	0	GetVersionExA
001204AE	005204AE	0	GlobalMemoryStatus
001204C3	005204C3	0	InitializeCriticalSection
001204DF	005204DF	0	IsBadReadPtr
001204EE	005204EE	0	LeaveCriticalSection
00120505	00520505	0	LoadLibraryA
00120514	00520514	0	MoveFileA
00120520	00520520	0	OpenProcess
0012052E	0052052E	0	PeekNamedPipe
0012053E	0052053E	0	Process32First
0012054F	0052054F	0	Process32Next
0012055F	0052055F	0	QueryPerformanceFrequency
0012057B	0052057B	0	ReadFile
00120586	00520586	0	ReleaseMutex
00120595	00520595	0	RemoveDirectoryA
001205A8	005205A8	0	SetConsoleCtrlHandler
001205C0	005205C0	0	SetCurrentDirectoryA
001205D7	005205D7	0	SetFilePointer
001205E8	005205E8	0	SetUnhandledExceptionFilter
00120606	00520606	0	Sleep
0012060E	0052060E	0	TerminateProcess
00120621	00520621	0	WaitForSingleObject
00120637	00520637	0	WriteFile
00120641	00520641	0	msvcrt.dll
0012064E	0052064E	0	_itoa
00120656	00520656	0	_stat
0012065E	0052065E	0	_mbsdup
00120668	00520668	0	_strcmpi
00120671	00520671	0	msvcrt.dll
0012067E	0052067E	0	__getmainargs
0012068E	0052068E	0	__p__environ
0012069D	0052069D	0	__p__fmode
001206AA	005206AA	0	__set_app_type
001206BB	005206BB	0	_beginthread

001206CA	005206CA	0	_cexit
001206D3	005206D3	0	_errno
001206DC	005206DC	0	_fileno
001206E6	005206E6	0	_iob
001206ED	005206ED	0	_onexit
001206F7	005206F7	0	_setmode
00120702	00520702	0	_vsnprintf
0012070F	0052070F	0	abort
00120717	00520717	0	atexit
00120720	00520720	0	atoi
00120727	00520727	0	clock
0012072F	0052072F	0	fclose
00120738	00520738	0	fflush
00120741	00520741	0	fgets
00120749	00520749	0	fopen
00120751	00520751	0	fprintf
0012075B	0052075B	0	fread
00120763	00520763	0	free
0012076A	0052076A	0	fwrite
00120773	00520773	0	malloc
0012077C	0052077C	0	memcpy
00120785	00520785	0	memset
0012078E	0052078E	0	printf
00120797	00520797	0	raise
0012079F	0052079F	0	rand
001207A6	005207A6	0	realloc
001207B0	005207B0	0	setvbuf
001207BA	005207BA	0	signal
001207C3	005207C3	0	sprintf
001207CD	005207CD	0	srand
001207D5	005207D5	0	_mbscat
001207DF	005207DF	0	strchr
001207E8	005207E8	0	strcmp
001207F1	005207F1	0	_mbscpy
001207FB	005207FB	0	strerror
00120806	00520806	0	strncat
00120810	00520810	0	strncmp
0012081A	0052081A	0	strncpy
00120824	00520824	0	strstr
0012082D	0052082D	0	time
00120834	00520834	0	toupper
0012083C	0052083C	0	shell32.dll
0012084A	0052084A	0	ShellExecuteA
00120858	00520858	0	USER32.dll
00120865	00520865	0	DispatchMessageA
00120878	00520878	0	ExitWindowsEx
00120888	00520888	0	GetMessageA
00120896	00520896	0	PeekMessageA
001208A3	005208A3	0	version.dll
001208B1	005208B1	0	GetFileVersionInfoA
001208C7	005208C7	0	VerQueryValueA
001208D6	005208D6	0	wininet.dll
001208E4	005208E4	0	InternetCloseHandle
001208FA	005208FA	0	InternetGetConnectedState
00120916	00520916	0	InternetOpenA
00120926	00520926	0	InternetOpenUrlA
00120939	00520939	0	InternetReadFile
0012094A	0052094A	0	ws2_32.dll

00120957	00520957	0	WSAGetLastError
00120969	00520969	0	WSASocketA
00120976	00520976	0	WSAStartup
00120983	00520983	0	__WSAFDIsSet
00120992	00520992	0	accept
0012099B	0052099B	0	bind
001209A2	005209A2	0	closesocket
001209B0	005209B0	0	connect
001209BA	005209BA	0	gethostbyaddr
001209CA	005209CA	0	gethostbyname
001209DA	005209DA	0	gethostname
001209E8	005209E8	0	getsockname
001209F6	005209F6	0	htonl
001209FE	005209FE	0	htons
00120A06	00520A06	0	inet_addr
00120A12	00520A12	0	inet_ntoa
00120A1E	00520A1E	0	ioctlsocket
00120A2C	00520A2C	0	listen
00120A35	00520A35	0	htonl
00120A3D	00520A3D	0	recv
00120A44	00520A44	0	select
00120A4D	00520A4D	0	send
00120A54	00520A54	0	sendto
00120A5D	00520A5D	0	setsockopt
00120A6A	00520A6A	0	shutdown
00120A75	00520A75	0	socket

Behavioral Analysis

Preparation for Infection

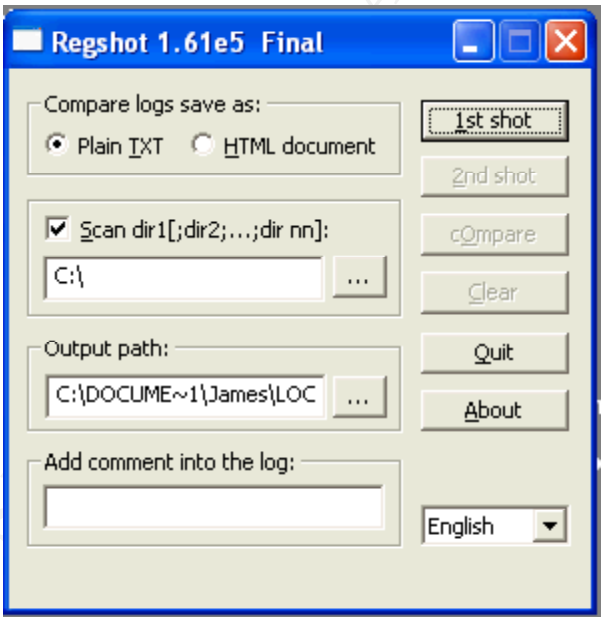
First I use md5sum to get the md5 hash of the msrll.exe malware file by running the following command:

```
c:\malware\exe>md5sum msrll.exe
84acfe96a98590813413122c12c11aaa *msrll.exe
```

The reason I did this first is so I have a baseline to compare to after executing the msrll.exe malware. If the malware modifies msrll.exe or copies itself somewhere else we will be able to verify if the file is the exact same as the original.

I then ran Regshot to get a snap shot of the registry on the clean system. Fig. 1 shows the options used to get the 1st snap shot.

Fig. 1



I click 1st shot and select shot and save to get a copy of the registry while the system is clean to compare later with the infected version.

Next I open FileMon, RegMon, and TDIMon and stop them from capturing and clear the display.

Process Explorer is then opened to show the processes running on the system.

On the VMware virtual Red Hat 9 system I start the sniffer by typing in the following

command

```
snort –vd | tee /tmp/sniffer1.log
```

This command will use snort to capture network traffic to the file sniffer1.log in the tmp directory.

Infection

Now everything is ready to infect the VMware Windows XP SP1 system. I start capturing on FileMon, RegMon, and TDIMon. I then execute msrll.exe malware by double clicking on it. After waiting about 30 seconds I view the processes using Process Explorer and see msrll.exe running under the parent process explorer. By highlighting the process and hitting the del key I kill the malware process.

Quickly I stop capturing on FileMon, RegMon, and TDIMon. Also, I switch over to the VMware Red Hat 9 system and hit CTRL+C to end the snort capture.

I run Regshot again this time clicking on 1st shot and selecting load. I browse to the saved registry file from the first time I ran Regshot on the clean system. Next I click on 2nd shot and select shot and save. Now we can click on compare and see the results shown in Fig. 2.

Fig. 2

```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Security\Security: 01 00 14 80 90 00 00 00 9C
00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF 01 0F 00 01 01 00 00 00 00 00 01 00
00 00 00 02 00 60 00 04 00 00 00 00 00 14 00 FD 01 02 00 01 01 00 00 00 00 00 05 12 00 00 00 00 00 18 00 FF
01 0F 00 01 02 00 00 00 00 00 05 20 00 00 00 20 02 00 00 00 00 14 00 8D 01 02 00 01 01 00 00 00 00 00 05 0B
00 00 00 00 00 18 00 FD 01 02 00 01 02 00 00 00 00 00 05 20 00 00 00 23 02 00 00 01 01 00 00 00 00 00 05 12
00 00 00 01 01 00 00 00 00 00 05 12 00 00 00
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Type: 0x00000120
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Start: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\ErrorControl: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\ImagePath:
"C:\WINDOWS\System32\mfm\msrll.exe"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\DisplayName: "Rll enhanced drive"
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\ObjectName: "LocalSystem"

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Security\Security: 01 00 14 80 90 00 00 00
9C 00 00 00 14 00 00 00 30 00 00 00 02 00 1C 00 01 00 00 00 02 80 14 00 FF 01 0F 00 01 01 00 00 00 00 00
01 00 00 00 00 02 00 60 00 04 00 00 00 00 00 14 00 FD 01 02 00 01 01 00 00 00 00 00 05 12 00 00 00 00 00 18
00 FF 01 0F 00 01 02 00 00 00 00 00 05 20 00 00 00 20 02 00 00 00 00 14 00 8D 01 02 00 01 01 00 00 00 00 00
05 0B 00 00 00 00 00 18 00 FD 01 02 00 01 02 00 00 00 00 00 05 20 00 00 00 23 02 00 00 01 01 00 00 00 00 00
05 12 00 00 00 01 01 00 00 00 00 00 05 12 00 00 00
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Type: 0x00000120
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Start: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ErrorControl: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ImagePath:
"C:\WINDOWS\System32\mfm\msrll.exe"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\DisplayName: "Rll enhanced drive"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ObjectName: "LocalSystem"
```

In examining the interesting results from the Regshot compare it looks like a service was created by the name of “Rll enhanced drive”. This service was confirmed to exist by opening the services window on the system and locating the service with that name. It was setup to start automatically on every boot up and ran with logon of local system. It was running the executable located in c:\windows\system32\mfm\msrll.exe. I verified the file existed there by navigating explorer to that location where I found msrll.exe and jtram.conf files.

I ran md5sum on msrll.exe in the c:\windows\system32\mfm location and compared the hash to the original msrll.exe file hash. The hashes matched so the file was an exact copy of the one I ran earlier. I then opened jtram.conf in UltraEdit-32. The jtram.conf file was not understandable and was probably encrypted.

I examined the FileMon log and found some interesting entries listed in Fig. 3.

Fig. 3

2:51:07 PM	msrll.exe:256	CREATE	C:\WINDOWS\System32\mfm SUCCESS Options: Create Directory Access: All
2:51:08 PM	msrll.exe:256	CREATE	C:\WINDOWS\System32\mfm\msrll.exe SUCCESS Options: OverwriteIf Sequential Access: All
2:51:08 PM	msrll.exe:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe SUCCESS Offset: 0 Length: 41984
2:51:29 PM	msrll.exe:952	OPEN	C:\WINDOWS\system32\mfm\jtram.conf FILE NOT FOUND Options: Open Access: All
2:51:29 PM	msrll.exe:952	CREATE	C:\WINDOWS\system32\mfm\jtram.conf SUCCESS Options: OverwriteIf Access: All

The FileMon log shows the creation of the directory mfm in c:\windows\system32 along with the creation of the files msrll.exe and jtram.conf.

Examination of the RegMon log files didn’t reveal much more than what Regshot did. One interesting entry found in the RegMon log is listed in Fig. 4.

Fig. 4

659.29827310	msrll.exe:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCCESS
E0 7F F7 64 D4 66 A1 09 ...				

This entry has something to do with cryptography.

Examination of TDIMon shows TCP activitiy being setup on the system. Entries in the TDIMon log show ports 2200 and 113 in use. TDIMon interesting log entries are displayed in Fig. 5.

Fig. 5

2:51:27 PM	msrll.exe:952	80D06A38	IRP_MJ_CREATE	TCP:0.0.0.0:2200 SUCCESS Address Open
2:51:27 PM	msrll.exe:952	80D06A38	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200 SUCCESS Error Event
2:51:27 PM	msrll.exe:952	80D06A38	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200 SUCCESS Disconnect Event
2:51:27 PM	msrll.exe:952	80D06A38	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200 SUCCESS Receive Event
2:51:27 PM	msrll.exe:952	80D06A38	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200 SUCCESS Expedited Receive Event
2:51:27 PM	msrll.exe:952	80D06A38	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200 SUCCESS Chained Receive

```
Event
2:51:27 PMmsrll.exe:952        80D06A38  TDI_QUERY_INFORMATION       TCP:0.0.0.0:2200              SUCCESS Query Address
2:51:27 PMmsrll.exe:952        FFB72AF0  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0x80DB9968
2:51:27 PMmsrll.exe:952        FFB72AF0  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:2200
2:51:27 PMmsrll.exe:952        FFBBD028  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0x80D46AA8
2:51:27 PMmsrll.exe:952        FFBBD028  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:2200
2:51:27 PMmsrll.exe:952        80D22628  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0xFFB7EBE8
2:51:27 PMmsrll.exe:952        80D22628  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:2200
2:51:27 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Connect Event

2:51:29 PMmsrll.exe:952        FFB70E00  IRP_MJ_CREATE       TCP:0.0.0.0:113             SUCCESS Address Open
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Error Event
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Disconnect Event
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Receive Event
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Expedited Receive
Event
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Chained Receive
Event
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_QUERY_INFORMATION       TCP:0.0.0.0:113               SUCCESS Query Address
2:51:29 PMmsrll.exe:952        80D8E2B8  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0x80D71008
2:51:29 PMmsrll.exe:952        80D8E2B8  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:113
2:51:29 PMmsrll.exe:952        80DB7900  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0x80DBA890
2:51:29 PMmsrll.exe:952        80DB7900  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:113
2:51:29 PMmsrll.exe:952        80DC4600  IRP_MJ_CREATE       TCP:Connection obj          SUCCESS Context:0x80D996D0
2:51:29 PMmsrll.exe:952        80DC4600  TDI_ASSOCIATE_ADDRESS       TCP:Connection obj          SUCCESS TCP:0.0.0.0:113
2:51:29 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Connect Event

2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Connect Event: NULL
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Connect Event: NULL
2:52:15 PMmsrll.exe:952        80D22628  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:2200              SUCCESS
2:52:15 PMmsrll.exe:952        80D22628  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80D22628  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        FFBBD028  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:2200              SUCCESS
2:52:15 PMmsrll.exe:952        FFBBD028  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        FFBBD028  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        FFB72AF0  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:2200              SUCCESS
2:52:15 PMmsrll.exe:952        FFB72AF0  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        FFB72AF0  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Error Event: NULL
2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Disconnect Event:
NULL
2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Receive Event: NULL
2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Expedited Receive
Event: NULL
2:52:15 PMmsrll.exe:952        80D06A38  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:2200              SUCCESS Chained Receive
Event: NULL
2:52:15 PMmsrll.exe:952        80D06A38  IRP_MJ_CLEANUP      TCP:0.0.0.0:2200            SUCCESS
2:52:15 PMmsrll.exe:952        80DC4600  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:113               SUCCESS
2:52:15 PMmsrll.exe:952        80DC4600  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80DC4600  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80DB7900  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:113               SUCCESS
2:52:15 PMmsrll.exe:952        80DB7900  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80DB7900  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80D8E2B8  TDI_DISASSOCIATE_ADDRESS  TCP:0.0.0.0:113               SUCCESS
2:52:15 PMmsrll.exe:952        80D8E2B8  IRP_MJ_CLEANUP      TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        80D8E2B8  IRP_MJ_CLOSE        TCP:Connection obj          SUCCESS
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Error Event: NULL
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Disconnect Event:
NULL
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Receive Event: NULL
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Expedited Receive
Event: NULL
2:52:15 PMmsrll.exe:952        FFB70E00  TDI_SET_EVENT_HANDLER       TCP:0.0.0.0:113               SUCCESS Chained Receive
Event: NULL
2:52:15 PMmsrll.exe:952        FFB70E00  IRP_MJ_CLEANUP      TCP:0.0.0.0:113             SUCCESS
```

I also ran AutoRuns just to check if any auto run registry entry had been added, but all looked well there.

Next I examined the snort capture file sniffer1.log. This capture file was completely empty. So it appears that no network traffic took place during the initial infection of the system.

I decided to run the service created “Rll Enhanced Drive”, but before I do I ready snort for another network capture with the command:

```
snort –vd | tee /tmp/sniffer2.log
```

I also start Process Explorer to monitor what processes are running.

Now I start the service and leave it run another 30 seconds. While I’m waiting for the 30 seconds I examine Process Explorer and see msrll.exe process running under parent service services.exe. By double clicking on the msrll.exe process in Process Explorer I get a window with various tabs to display information about the process. I select the TCP/IP tab and see that it is listening on ports 113 and 2200. After about 30 seconds pass I kill the process and switch to the VMware Red Hat 9 system and end the snort capture.

The listening ports 113 and 2200 correlates to the TDIMon log entries. This malware is listening on these ports.

Examination of the snort capture sniffer2.log shows the interesting entries displayed in Fig. 6.

Fig. 6

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-06:08:52.228500 192.168.62.129:1026 -> 192.168.62.1:53
UDP TTL:128 TOS:0x0 ID:786 IpLen:20 DgmLen:66
Len: 38
00 0D 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C  .............col
6C 65 63 74 69 76 65 37 04 7A 78 79 30 03 63 6F  lective7.zxy0.co
6D 00 00 01 00 01                                m.....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

This packet shows a DNS request coming from 192.168.62.129 port 1026 to 192.168.62.1 port 53 for the domain name collective7.zxy0.com. This DNS request is not answered however because 192.168.62.1 is the host computer that is running VMware and is not running a DNS server. 192.168.62.129 is the VMware virtual Windows XP SP1 system that is the infected system.

Since I have no DNS server I add to the hosts file on the infected system the entry listed in Fig. 7. The hosts file on the infected system is in c:\windows\system32\drivers\etc\hosts.

Fig. 7

```
192.168.62.128       collective7.zxy0.com
```

I put the address 192.168.62.128 in because I want to redirect any traffic going from the infected host 192.168.62.129 to my VMware Red Hat 9 system 192.168.62.128 so I can capture the requests the infected system is sending to collective7.zxy0.com.

I start another snort capture on the Red Hat 9 system using the following command:

```
snort –vd | tee /tmp/sniffer3.log
```

I also start Process Explorer to monitor the msrll.exe process.

I restart the "Rll Enhanced Drive" service and wait about 30 seconds. While waiting the 30 seconds I use Process Explorer to view the process msrll.exe TCP/IP activitiy. I observe it listening on ports 113 and 2200 like before, but during the wait I see it send a connection request to 192.168.62.128:6667 and then stop. I then see another connection request to 192.168.62.128:9999 and then stop. And another connection request to 192.168.62.128:8080 and then stop. The 30 seconds are past so I kill the msrll.exe process using Process Explorer and stop the snort capture on the Red Hat 9 system.

Examination of the snort capture file sniffer3.log I found the interesting entries listed in Fig. 8.

Fig. 8

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-10:38:58.493137 192.168.62.129:1103 -> 192.168.62.128:6667
TCP TTL:128 TOS:0x0 ID:1401 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8C1BF6D9  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-10:39:30.343066 192.168.62.129:1104 -> 192.168.62.128:9999
TCP TTL:128 TOS:0x0 ID:1403 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8C8951C2  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-10:40:01.274022 192.168.62.129:1105 -> 192.168.62.128:8080
TCP TTL:128 TOS:0x0 ID:1406 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8CF5FCD8  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Here there are connection requests from the infected system to the Red Hat 9 system on ports 6667, 9999, and 8080. The obvious port is 6667 which possibly could be an irc connection attempt. Port 9999 I'm not sure of, but listed as Distinct in the IANA list. Port 8080 is usually an alternate port for port 80 or proxy. These connection attempts listed in the snort capture correlate to the observed connection attempts in Process Explorer's TCP/IP section for the msrll.exe process.

Since port 6667 is normally irc port I decide to run an irc server on the Red Hat 9 system that the infected system already thinks is the collective7.zxy0.com. Also the observed connection attempts to 192.168.62.128 on ports 9999 and 8080 prompt me to run netcat to record the requests on the Red Hat 9 system.

I start the irc server which is configured to listen on port 6667 by issuing the following commands on the Red Hat 9 system:

```
su – ircd
./ircd
exit
```

I then start the irc client in another session on the Red Hat 9 system by issuing the command:

```
irc
```

I also start two more sessions on the Red Hat 9 system and issue the following commands:

```
nc –l –p 9999 >/tmp/port9999.txt
```

```
nc –l –p 8080 >/tmp/port8080.txt
```

These commands start netcat listening on ports 9999 and 8080. Whatever is sent to these ports will be written to the appropriate file port9999.txt or port8080.txt in the /tmp directory. This is in the hope to capture what type of requests are being sent to these ports.

Snort capture is also started again with the following command:

```
snort –vd | tee /tmp/sniffer4.log
```

Again we start the msrll.exe malware by starting the "Rll Enhanced Drive" service and wait about 30 seconds. After 30 seconds I kill the msrll.exe process and stopped the snort capture along with the two netcat listeners on the Red Hat 9 system.

First I examine the snort capture sniffer4.log and find the interesting sections displayed

in Fig. 9.

Fig. 9

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-10:55:25.907418 192.168.62.129:1107 -> 192.168.62.128:6667
TCP TTL:128 TOS:0x0 ID:1449 IpLen:20 DgmLen:53 DF
***AP*** Seq: 0x992158CC  Ack: 0xB8F21707  Win: 0xFAA3  TcpLen: 20
4A 4F 49 4E 20 23 6D 69 6C 73 20 3A 0A           JOIN #mils :.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/10-10:54:53.473313 192.168.62.129:1107 -> 192.168.62.128:6667
TCP TTL:128 TOS:0x0 ID:1444 IpLen:20 DgmLen:97 DF
***AP*** Seq: 0x99215880  Ack: 0xB8F210F6  Win: 0xFA7D  TcpLen: 20
55 53 45 52 20 56 52 55 74 6F 6E 6E 49 7A 65 77  USER VRUtonnIzew
46 20 6C 6F 63 61 6C 68 6F 73 74 20 30 20 3A 4D  F localhost 0 :M
74 65 62 70 6E 4C 0A 4E 49 43 4B 20 4B 4C 48 74  tebpnL.NICK KLHt
74 64 75 64 72 61 4B 44 0A                       tdudraKD.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Here it can be seen that the malware joined the channel #mils.

Next I examined the netcat capture files port9999.txt and port8080.txt and found them to be empty.

In the irc client I join the #mils channel by issuing the command:

```
/join #mils
```

This puts my client in the #mils channel.

I then run the malware again by starting the "Rll Enhanced Drive" service and wait in the irc client for the malware to join the #mils channel. After a few minutes it does join the #mils channel verifying that irc is in play with this malware.

I then try connecting to port 2200 on the infected system from the Red Hat 9 system using netcat command:

```
nc 192.168.62.129 2200
```

Netcat connects and a prompt appears #: that is awaiting my command. At this point I don't really know what it wants so I just hit enter then CTL+C to drop the connection.

I try to connect to port 113 on the infected system from the Red Hat 9 system by issuing the following command:

nc 192.168.62.129 113

Netcat connects but no prompt just a blinking cursor. I hit enter and nothing happens so I type x and press enter. This disconnects me with the following text:

x : USERID : UNIX : GPRdvDe

At this point I am done with my behavioral analysis since I cannot invoke anymore behavior.

The summary of what is known at this point is that the malware installs itself into the c:\windows\system32\mfm directory as msrll.exe and an encrypted configuration file jtram.conf. It sets up a Windows service called “Rll Enhanced Drive” that runs at Windows startup the msrll.exe file in the mfm directory. The malware then attempts to connect to a server on the internet called collective7.zxy0.com on port 6667, 9999, and 8080. The connection to port 6667 is an irc connection and the malware joins the channel #mils. The malware also listens on ports 2200 and 113 which can be connected to using netcat. The listener on port 2200 displays a prompt #: and the port 113 doesn’t display any prompt.

Code Analysis

The code analysis starts with examining the BinText listing of the msrll.exe file. Fig. A on page 11 displays the BinText output for the msrll.exe file. Here the segment .aspack leads me to believe that the executable has been packed with aspack. The rest of the strings are not understandable which also indicates that the file is packed.

PE-Sniffer is run on the msrll.exe file, but the scans do not reveal the packing technique used. I then load the file in PEInfo which displays the section aspack further indicating that aspack was used.

I load aspackdie with the msrll.exe and a file is created called unpacked.exe and seems to have unpacked the msrll.exe. A quick examination of the strings using BinText reveals more understandable strings. It looks like it worked, but an execute test should be done. I copy the file to the c:\windows\system32\mfm directory and rename the original msrll.exe to msrll.exe.org and then rename unpacked.exe to msrll.exe. I start the service while monitoring the irc channel #mils with the irc client on the Red Hat system and sure enough the malware joins the channel after a few minutes. It looks now like I have an unpacked version of the malware that still works.

Examination of the interesting strings reveals what looks like commands that start with a question mark (?). Commands like ?si, ?jolt, ?uptime, and ?login. Some of these commands look like denial of service attacks while others seem to be giving information about the current state of the system. The command ?login leads me to believe that some sort of authentication needs to be done.

While running the malware I tried some of the commands in the irc connection to the #mils channel. Every command I typed didn't elicit any response from the malware. I then try the following in the irc client:

```
?login
?login malware
? login a b
```

None of these generated a response from the malware. I tried these same commands using the netcat connection to port 2200 on the infected system from the Red Hat 9 system and got similar results.

The Search for Authentication Code

I'm going to cut to the chase here since there were many failed attempts at finding the authentication code. I used mainly the interesting strings found by BinText, see Fig. B on page 11, IDA Pro and OllyDbg to search the assembly instructions for possible areas where the malware is processing the authentication (?login).

I stumbled on the following string:

```
% bad pass from \"%s\"@%s
```

This looked like a place in the code you would go if your login failed.

I loaded the unpacked version of msrll.exe into IDA Pro and performed a text search looking for the string “bad pass”. The search had two hits, one at memory address 0040BB52 and another at memory address 0040BC6F displayed in Fig. 10.

Fig. 10

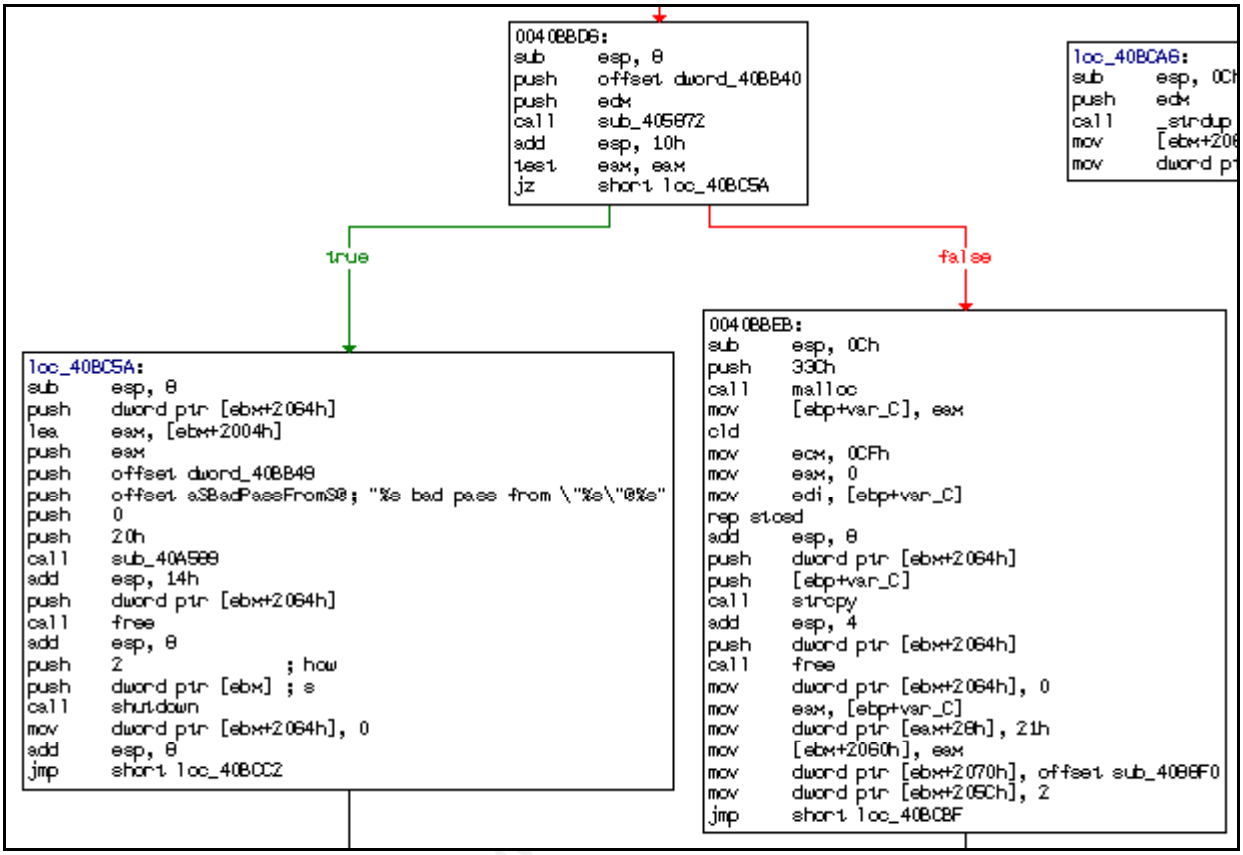
```
Address        Instruction
.text:0040BB52 aSBadPassFromS@ db '%s bad pass from \''%s\''@%s',0 ; DATA XREF: sub_40BB6B+104↓o
.text:0040BC6F         push    offset aSBadPassFromS@ ; '%s bad pass from \''%s\''@%s''
```

The instruction located at address 0040BC6F looked to be where bad pass would be pushed to the stack preparing to display on the screen possibly. This became the area of the code I focused on. I wanted to find how you ended up at this instruction and where the decision was made to branch to this section of the code. To find this branch I used IDA Pro to view a flowchart of the malware instructions. Clicking on view + graphs + flow chart I was able to view the code in a more visual way. I searched the flow chart manually until I found the instruction set that matched what was at address 0040BC6F. This was in a section labeled loc_40BC5A seen in Fig. 11. Going up the flowchart the instructions that decide what branch to jump to is in the section labeled 0040BBD6. In this section you can see the following instructions:

```
test    eax, eax
jz      short loc_40BC5A
```

It’s easy to see in the flowchart that if the result of the test is true we are going to jump to the section of code that contains the bad pass string which is where I’m assuming I don’t want to go if I want to get authenticated. If the result is false the code jumps to address 0040BBEB which might be where you go if you get successfully authenticated.

Fig. 11



I switch over to the code view and search for address 004BBD6 since this is the label for the beginning of the instructions that include the test and jz instructions. I find that the test instruction is at address 0040BBE7 which is where I decide I want to set a breakpoint, but not in IDA Pro. For this job I load msrll.exe into OllyDbg.

In OllyDbg I go to address 0040BBE7 and press F2 to set a breakpoint. Next I start ircd and irc on the Red Hat 9 system. In irc I join the #mils channel. Back in OllyDbg I hit F9 to run msrll.exe malware. Then I switch back to irc and wait for the malware to join the #mils channel. After a few minutes I see the malware join the #mils channel. I want to trigger the break point with the ?login command so I try the following commands:

```
?login malware
?login a b
```

Neither of these triggered the break point. I examined the code around 0040BBE7. The beginning of this section started at address 0040BBD6 which you can see in Fig. 11 above. Following the code I couldn't find any hints as to what the problem was.

I decided to connect using netcat on the Red Hat 9 system to port 2200 on the infected system. I issued the following command:

```
nc 192.168.62.129 2200
```

Netcat connected and I was prompted with a #:_. Next I issued the following commands:

```
?login malware <enter>
<enter>
<enter>
x <enter>
```

At the point I typed (x <enter>) I triggered the breakpoint in OllyDbg. Now I knew I was sitting at the TEST EAX, EAX instruction. From the flowchart research I knew I needed the result to be false. The EAX register contained 00000000 as its value which means the next instruction, JE SHORT msrll.0040BC5A, would be taken. This would be a true condition. So I right clicked on the EAX register and selected increment changing the EAX register's value to 00000001. This would now result in a false condition. I pressed F9 and continued running the malware.

Back on the Red Hat 9 system all I had was a cursor, but I was still connected. I typed in the following command:

```
?uptime <enter>
```

This returned the following line:

```
sys: 01h 49m 04s bot; 33m 09s
```

I was now authenticated, but just to make sure I issued the following command:

```
?status <enter>
```

This also returned information. It looks like it worked. I want to try all the commands to see what happens, but before I do that I'm going to patch the malware to bypass the authentication.

The instruction (JE SHORT, msrll.0040BC5A) needs to be replaced with a NOP so that this jump cannot happen since taking the jump is a true condition and I want a false condition to get authenticated.

To patch the malware I clicked on the JE SHORT, msrll.0040BC5A instruction in OllyDbg and hit the space bar. This opened an assembler window with a entry box that I typed in NOP. I made sure the "Fill with NOP's" box was checked in order to fill the

replaced instruction space properly. I then click assemble and cancel. This replaced the JE instruction with two NOP instructions. To save this patched version of the malware I right clicked on the assembler pane, select “Copy to executable”, select “All modifications”, clicked “Copy all”, and then a new disassembler pane opened. I right click on the pane and select “Save file”. This prompted me for the file name to save as. I used the name msrll-patched.exe and saved it.

Now I had a patched version of the program. I saved this version in place of the current msrll.exe in the c:\windows\system32\mfm directory.

Commands

Command	Description	Response
?insmod	Install loadable modules	?insmod: <mod name>
?rmmod	Remove loadable modules	?rmmod: <mod name>
?lsmod	List loadable modules	(No response)
?ping	ping DoS attack	?ping <ip> <total secs> <p size> <delay> [port]
?udp	udp DoS attack	?udp <ip> <total secs> <p size> <delay> [port]
?syn	syn DoS attack	?syn <ip> <port> <t_time> <delay>
?smurf	smurf DoS attack	?smurf <ip> <p size> <duration> <delay>
?jolt	jolt DoS attack	?jolt <ip> <duration> <delay>
?si	Displays system information	WINXP (u:James) mem:(52/127) 58% Genuine Intel (null)
?ssl	Something to do with ssl	?ssl: -1
?clone	?	usage ?clone: server[:port] amount
?clones	?	?clones: [NETWORK|all] <die|join|part|raw|msg> <"parm"> …
?login	Login command must be ?login <enter> user <enter> password <enter>	(No response on unsuccessful login)
?uptime	Displays system and bot uptime stats	sys: 58m 16s bot: 27m 14s
?reboot	Reboots the system	later!
?status	Displays status info	service:N user:James inet connection:Y contype: Lan reboot privs:Y
?jump	?	(No response)
?nick	Change nick on irc channel that bot is connected to, but you must first select the irc sock to perform the command on	Set an irc sock to perform ?nick command on Type .sklist to view current sockets, then .dccsk <#>

?echo	echos what you type after ?echo to the screen - if you typed ?echo hello <enter> the response would be what is in the response column to the right	hello
?hush	?	
?wget	When you type ?wget <enter> there is no response, but if you type ?wget x <enter> the reponse will be what is in the response column to the right	no file name in x
?join	irc join command to join other channels	(No output to screen - need to us the ?sklist and ?dccsk commands to find and connect to an irc sock first)
?op	irc command to become channel operator	?op bad args
?aop	Not sure what this does, but you can add or remove a host by the ?aop + <host> or ?aop - <host>	usage: ?aop +/- <host>
?akick	Not sure what this one does either, but has similar syntax as ?aop	usage: ?akick +/- <host>
?part	irc command to leave a channel	(No output to screen)
?dump	?	(No response)
?set	shows the jtram.conf contents plus can change settings by issueing the ?set <setting> <value>	set jtr.bin msrll.exe set jtr.home mfm set bot.port 2200 set jtr.id run5 set irc.quit set servers collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy0.com:8080 set irc.chan #mils set pass 1KZLPLKDf$W8kl8Jr1X8DOHZsmlp9qq0 set dcc.pass 1KZLPLKDf$55isA1lTvamR7bjAdBziX
?die	Kills msrll.exe process	(No output to screen)
?md5p	Displays the salt and md5 hash of whatever is typed in as <pass> parameter	?md5p <pass> <salt>
?free	?	usage: ?free <cmd>
?raw	?	(No output to screen)
?update	Possiably a command to update bot	?update <url> <id>

?hostname	Displays hostname and ip address	host: xxx.localdomain ip: 192.168.62.129
?fif	?	(No response)
?!fif	?	(No response)
?del	Delete a file	
?pwd	Display the current directory	c:\windows\system32\mfm
?play	?	(null): somefile
?copy	Copy a file	
?move	Move a file	
?dir	Display directory	(Displays directory listing of current directory)
?sums	Display the md5 hashs for all files in current directory	(Displays a file listing with it's md5 hash value next to it)
?ls	Displays directory listing of current directory	(Displays directory listing of current directory)
?cd	Changes directory	
?rmdir	Removes a directory	
?mkdir	Makes a new directory	
?run	Run a program (hidden) - syntax: ?run c:\windows\system32\notepad.exe <enter>	(Example: ?run c:\windows\system32\notepad.exe) ?run: ran ok (4022304)
?exec	irc command exec	(No response)
?ps	Display all processes running and their PID's	(lists active processes running on infected system)
?kill	Kill a process - ?kill 1448 <enter>	pid 1448 killed
?killall	?	(No response)
?crash	Crashes system	(No output to screen)
?dcc	irc command dcc direct connections to remote clients	(No response)
?get	?	(No response)
?say	irc say command - I believe this message would be said non-privately	usage: ?say <target> "text"
?msg	irc command to send private message to nick or list of nicks	usage: ?msg <target> "text"
?kb	?	?kb <nick> <chan>
?sklist	Display current socks	(A display numbering the different socks and connection information like ip address, nick, and irc chan
?unset	Un sets a set command refer to ?set command	(Example: ?unset pass <enter>) (This will remove the set pass parameter)
?uattr	?	usage: ?uattr <nick> <chan>
?dccsk	Set the irc sock to use	usage: ?dccsk <socks #>

?con	irc command (server operator command) connect two servers together	(No response)
?killsk	Kills a irc sock - Example: ?killsk 1 <enter>	closing 1 [collective7.zxy0.com:6667]

Analysis Wrap-Up

Once the malware is executed on a system, it will copy itself to the c:\%systemroot%\system32\mfm directory. It will then create jtram.conf file in the same directory. The jtram.conf file contains the encrypted configuration settings of the bot. The malware sets up a service, Rll enhanced drive, which starts automatically when the system is booted and runs with local system authority. The bot then attempts to connect to the irc server collective7.zxy0.com first on port 6667 then port 9999 and then port 8080. Once the bot is connected to the irc server it joins the #mils channel with a randomly generated nick. At this point the bot is awaiting orders from the bot commander/creator.

Analysis shows this bot is capable of receiving a connection on port 2200 using telnet or netcat. Connecting to this port presents a prompt #:_ awaiting authentication using the ?login command. Authentication allows you to execute numerous commands. These commands can setup denial of service attacks, run programs hidden to the user, update the bot, send irc related commands to the irc server from the bot system, get information on the infected system resources and configuration, kill processes, transfer files and change the bot configuration.

This bot army might have been created to sell or trade for something in return, to attack a specific website, to speed the spread of a future virus or worm, to steal financial information, to harvest email address, to spam, and the list goes on. This bot appears to have upgradeability build in so its purpose could change.

The first defensive tactic is to use a firewall to block outgoing traffic on port 6667, 9999, and 8080. If other outgoing ports are not being used they should be blocked as well to prevent the bot from reporting in. Next block incoming traffic to port 2200 as well as any other ports that are not required. Now you're left with the existing infected systems to detect and clean. To find the infected systems first run antivirus software, but if that doesn't detect it then you could run a port scanner on the network like nmap and look for systems listening on port 2200 or other odd ports. You could manually go to each one of the suspect systems and kill the msrll.exe process and remove the c:\%systemroot%\system32\mfm directory and files from the system. You'd also have the rll enhanced drive service to deal with by at least setting it to manual startup instead of automatic. You could also script the removal of the files since nothing should live in the mfm directory. The following batch file presents as an example. It would not matter if this ran on a system that wasn't infected; it would just not delete

anything.

```
@echo off
cls
echo Ready to delete mfm directory…
pause
c:
del /Q c:\windows\system32\mfm\*.*
del /Q c:\winnt\system32\mfm\*.*
attrib –r c:\windows\system32\mfm
attrib –r c:\winnt\system32\mfm
rd c:\windows\system32\mfm
rd c:\winnt\system32\mfm
```

Additional things you could do to prevent future attacks is to install a personal firewall on each system that can detect when unauthorized applications try to communicate on the network or Internet. Keep the antivirus software up to date. Monitor on regular bases the listening ports on each system. Create Snort IDS signatures to detect this activity.

List of Resources

Software Tools

vmware Workstation Product Web Site. 8 Dec. 2004
<http://www.vmware.com/products/desktop/ws_features.html>

Symantec Ghost Product Web Site. 8 Dec. 2004
<http://sea.symantec.com/content/product.cfm?productid=9>

Free Software Foundation, Inc. md5sum Web Site. 8 Dec 2004
<http://www.gnu.org/software/textutils/textutils.html>

Foundstone, Inc. BinText Web Site. 8 Dec. 2004
<http://www.foundstone.com/proddesc/bintext.htm>

TiANWEi. Regshot Download Web Site. 8 Dec. 2004
<http://www.pcworld.com/downloads/file_description/0,fid,19540,00.asp>

Russinovich, Mark and Cogswell, Bryce. Filemon Web Site. 8 Dec. 2004
<http://www.sysinternals.com/ntw2k/source/filemon.shtml>

Russinovich, Mark and Cogswell, Bryce. Regmon Web Site. 8 Dec. 2004
<http://www.sysinternals.com/ntw2k/source/regmon.shtml>

Russinovich, Mark. TDIMon Web Site. 8 Dec. 2004
<http://www.sysinternals.com/ntw2k/freeware/tdimon.shtml>

Russinovich, Mark. Process Explorer Web Site. 8 Dec. 2004
<http://www.sysinternals.com/ntw2k/freeware/procexp.shtml>

Russinovich, Mark and Cogswell, Bryce. Autoruns Web Site. 8 Dec. 2004
<http://www.sysinternals.com/ntw2k/freeware/autoruns.shtml>

GNU Netcat Web Site. 8 Dec. 2004 <http://netcat.sourceforge.net/>

SkymarShall. PESniffer Software. Reverse-Engineering Malware Tools and Techniques Hands-On SANS Institute Supplemental CD Lenny Zeltser v.9. CD-ROM.

Liston, Tom. PEInfo Software. Reverse-Engineering Malware Tools and Techniques Hands-On SANS Institute Supplemental CD Lenny Zeltser v.9. CD-ROM.

y0da. ASPACKDIE Web Site. 8 Dec. 2004 <http://scifi.pages.at/yoda9k/proggies.htm>

IDA Pro 4.6 Product Web Site. 8 Dec. 2004 <http://www.datarescue.com/idabase/index.htm>

Yuschuk, Oleh. OllyDbg Web Site. 8 Dec. 2004 <http://home.t-online.de/home/ollydbg/>

Snort Web Site. 8 Dec. 2004 <http://www.snort.org>

ircdhybrid Web Site. 8 Dec. 2004 <http://www.ircd-hybrid.com>

ircII Web Site. 8 Dec. 2004 <http://www.eterna.com.au/ircii