Global Information Assurance Certification Paper

GIAC Reverse Engineering Malware Practical Assignment

GREM Version 1.0

Joseph Fresch
SANS - GREM /
Las Vegas NV
2004

Submitted
March 3, 2005

Table of Contents

List of Figures

Abstract

Reverse Engineer an unknown piece of Malware using different methods of analysis. Dangerous malicious code needs to have proper precautions to prevent it from entering the wild of the Internet. To this end a set of Virtual Machines have been brought together to form a test environment, with the emphasis placed on safe controlled analysis of this unknown Malware. Once enough data is collected surmise the capabilities of the Malware specimen, what does it do, who would use the program. What can be done to set up defensive measures can be and what can be derived from the analysis to prevent future attacks and eliminate current infections.

Document Convention

When you read this practical assignment, you will see that certain words are represented in different fonts and typefaces. The types of words that are represented this way include the following:

`command`	Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell.
`filename`	Filenames, paths, and directory names are represented in this style.
`computer output`	The results of a command and other computer output are in this style
URL	Web URL's are shown in this style.
Quotation	A citation or quotation from a book or website is in this style.

Laboratory Setup

To afford some protection and allow for a controlled analysis of the Malware, software called VMWare was used to create the test environment. The test environment was created using VMWare version 4.5.2 build 8848 on a dual Athlon MP system with 512 MB of RAM and 120GB hard drive. This software allows for the emulation of different hardware and allows for installation of different Operating System software into those emulated collections of hardware.
Configuration and installation of the Operating Systems in the VMWare environment requires knowledge of both VMWare and of Operating System Installation. The network was setup to use Host Only Networking.

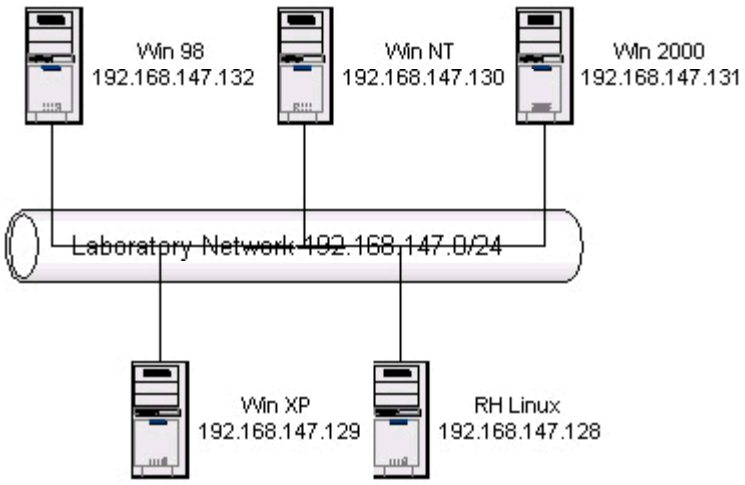
Host-only — When you use this type of network connection, the virtual machine is connected to the host operating system on a virtual private network, which normally is not visible outside the host. Multiple virtual machines configured with host-only networking on the same host are on the same network[1].

OS	Version	Service Pack	Memory (MB)	Hard Disk1 (GB)	Hard Disk2 (GB)
Win 98	4.10.1998		128	2	none
Win NT 4	4.00.1381	IE 5 6.0.2800.1106	256	1.29 (4.0)	0.126 (0.512)
Win 2K	5.00.2195	SP 4	256	2	none
Win XP	2002	SP 1	128	2	1
Linux RH 9			284	2	none

Figure 1: Machine specifics for lab environment.

As an added precaution it is advised that the researcher install and use a local firewall application like ZoneAlarm[2]. This will help prevent any accidental releases into the LAN or out onto the Internet. The operating systems focused on in this research were Windows XP Professional (Win XP) and Linux Red Hat 9.0 running kernel 2.4.20-8 (RH Linux). The Malware was tested for launch on each of the other four Operating Systems, Windows 98, Windows NT and Windows 2000 and Windows XP Professional. The RH Linux allowed for interaction with the Malware as well as use of tools such as Snort, Net Cat and Internet Relay Chat (IRC). The Windows machines will provide the environment for running the Malware. This is also where the investigation of the specimen will be conducted.

[1] Host-only information from the VMWare help, search host only network and displaying the first entry, Configuring a Network Adapter (NIC).
[2] http://www.zonealarm.com



2: Lab Setup with Host Only Networking – 192.168.147.0/24.

Tools for Analysis:

- `WinZip.exe 9.0 (6028)` – Used to unpack the Malware Specimen.
- `MD5sum.exe` – Used to create MD5 hash of files.
- `RegShot.exe 1.61e5 Final 2003/1/1` – Used for comparison of Registry before and after Malware is launched.
- `RegMon.exe` – Used to show access to the Registry.
- `FileMon.exe 6.07` – Used to show file access.
- `TDIMon.exe 1.01` – Used to view network access similar to netstat –an.
- `UPX 1.24` – Unpacking tool.
- `BinText.exe 3.0` – Used to inspect strings found in the Malware Specimen.
- `LordPE.exe RoyalTS` – Used to Dump the decrypted Malware Specimen from memory.
- `Unaspack.exe 2.0` – BinText.exe showed the string “aspack” a packing software was used to pack the Malware.
- `AspackDie.exe 1.3d` – Utility used to unpack the Malware Specimen.
- `OllyDB.exe 1.0.10.0` – 32 bit Assembler Level Debugger.
- `IDAPro.exe 4.0` – Used to disassemble the Malware Specimen.
- `Snort 2.04 (Build 96)` – Used to monitor network traffic.
- `NetCat.exe 1.10` – Used to set listening ports and move files from Linux to Win XP machine.
- `Ircd-hybrid 2.8/hybrid-6.3.1` – IRC server.
- `Listdlls.exe` – Lists Dynamically Link Libraries and process that is using them.
- `TCPView.exe 2.34` – For viewing TCP/UDP ports being used and the process with ID.
- `VMWare 4.52 build 8848` – Used for the creation of the testing environment.

Properties of the Malware Specimen

The Malware `msrll.exe`, is an application file, as seen in Figure 3 the `msrll.exe` Properties.

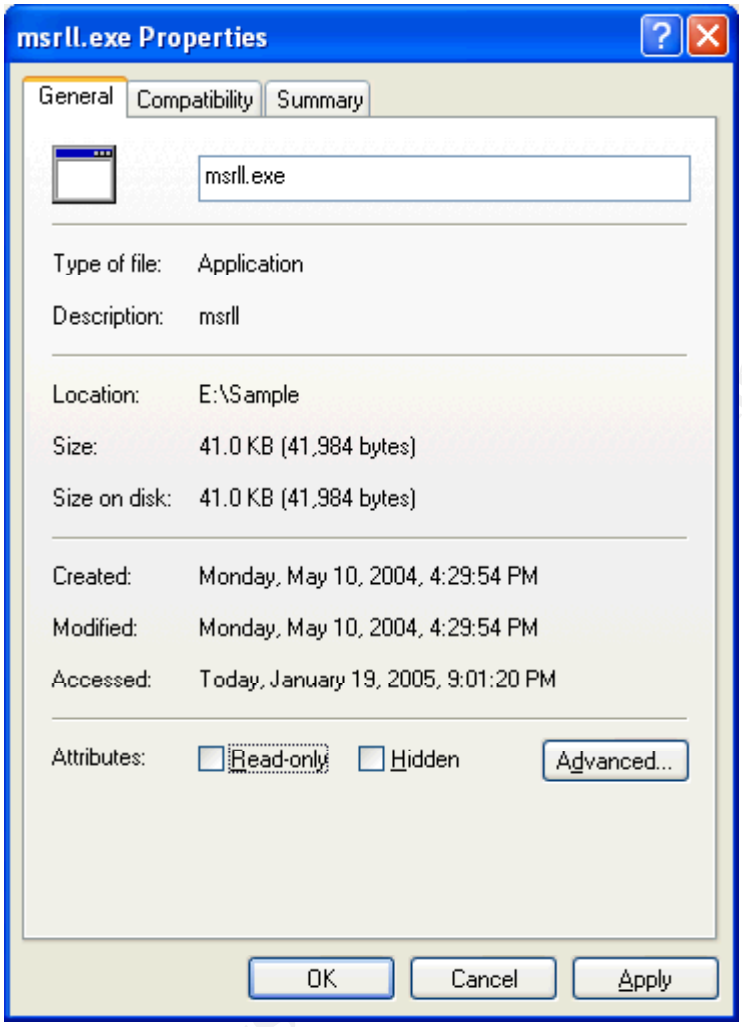


Figure 3: Properties of `msrll.exe`.

From the Properties this researcher was able to determine the file size is 41KB (41,984 bytes). It was created on May 10, 2004 at 4:29:54 PM. It was downloaded from the GIAC website as `msrll.zip`. It was unzipped with WinZip using the password "malware" as stated in the GREM Practical Assignment documentation. The MD5 hash for msrll.exe is 84acfe96a98590813413122c12c11aaa, which was found using the `md5sum.exe` application. The Malware `msrll.exe` does not run in DOS mode, as seen in the strings found using `BinText.exe`. The Malware does, however, run on Windows 98, Windows NT, Windows 2000 and Windows XP Professional, as it

was launched in each within the lab setup VMWare instances. From OllyDbg CPU window we can see in the code that different Operating Systems are usable by the Malware to run. This information also helps the Malware determine the Registry Keys to write to the infected machines Registry to allow it to start up when the system reboots.

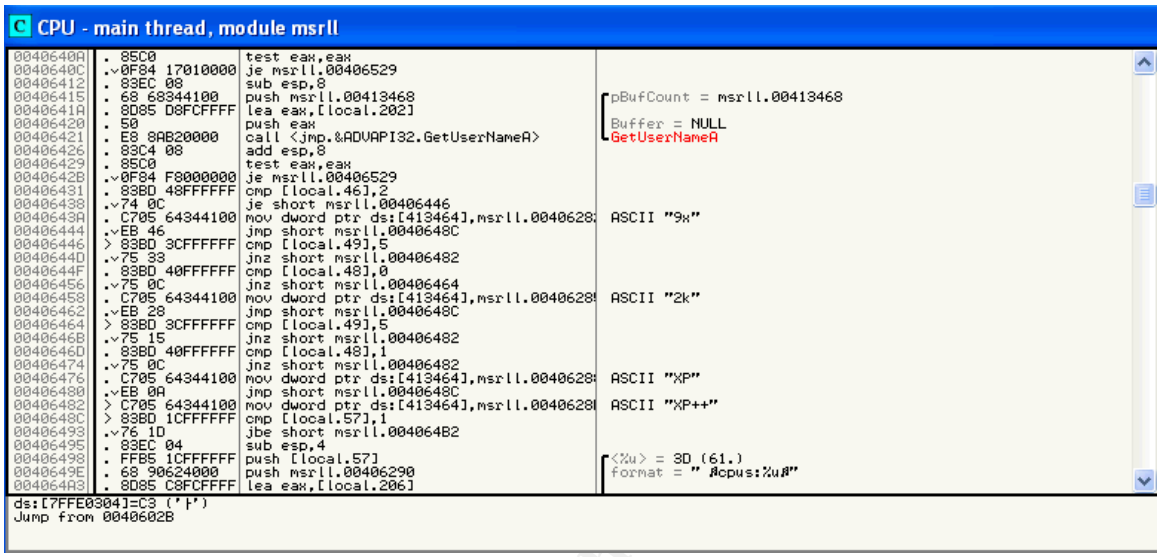


Figure 4: Operating System environments in which the Malware can execute.

.

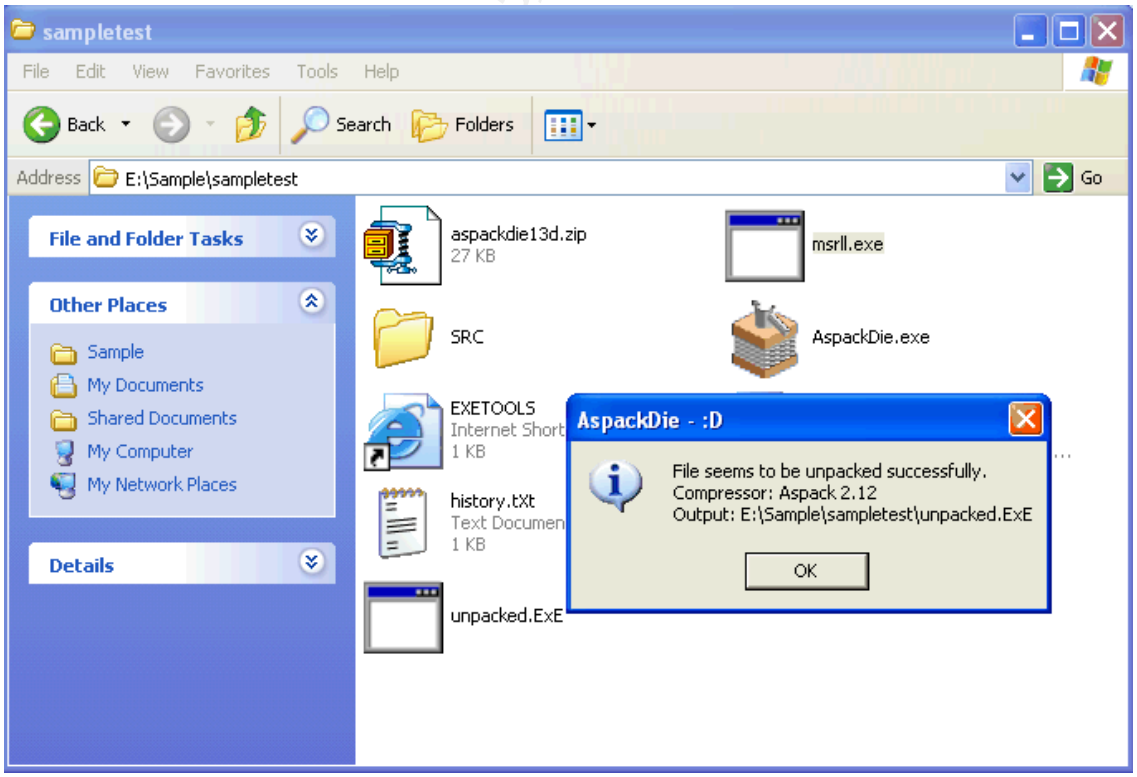


Figure 5: Unpacking the `msrll.exe` with AspackDie 1.3d.
It is packed with Aspack.exe 2.12, which AspackDie 1.3d listed as it unpacked the Malware specimen. Simply drag the Malware sample over the AspackDie.exe application and release the mouse button and it will begin processing the Malware file if it is able to unpack the specimen.

The Malware contains the following significant embedded strings, found by using OllyDB.exe and IDAPro.exe to investigate the unpacked specimen:

```
.aspack
```

Location	Command	Location	Command	Location	Command
409345	"?si",0	004093C1	"?die",0	409436	"?exec",0
409349	"?ssl",0	004093C6	"?md5p",0	0040943C	"?ps",0
0040934E	"?clone",0	004093CC	"?free",0	409440	"?kill",0
409355	"?clones",0	004093D2	"?raw",0	409446	"?killall",0
0040935D	"?login",0	004093D7	"?update",0	0040944F	"?crash",0
409364	"?uptime",0	004093DF	"?hostname",0	409456	"?dcc",0
0040936C	"?reboot",0	4.09E+12	"?fif",0	0040945B	"?get",0
409374	"?status",0	004093EE	"?!fif",0	409460	"?say",0
0040937C	"?jump",0	004093F4	"?del",0	409465	"?msg",0
409382	"?nick",0	004093F9	"?pwd",0	0040946A	"?kb",0
409388	"?echo",0	004093FE	"?play",0	0040946E	"?sklist",0
0040938E	"?hush",0	409404	"?copy",0	409476	"?unset",0
409394	"?wget",0	0040940A	"?move",0	0040947D	"?uattr",0
0040939A	"?join",0	409410	"?dir",0	409484	"?dccsk",0
004093A0	"?op",0	409415	"?sums",0	0040948B	"?con",0
004093A4	"?aop",0	0040941B	"?ls",0	409490	"?killsk",0
004093A9	"?akick",0	0040941F	"?cd",0	409499	"VERSION*",0
004093B0	"?part",0	409423	"?rmdir",0	004094A8	"PING",0
004093B6	"?dump",0	0040942A	"?mkdir",0	004094AE	"IDENT",0
004093BC	"?set",0	409431	"?run",0		

```
servers
collective7.zxy0.com,collective7.zxy0.com:9999!,collective
7.zxy0.com:8080

irc.chan
#mils

mIRC v6.12 Khaled Mardam-Bey

m220 1.0 #2730 Mar 16 11:47:38 2004

jtr.bin
msrll.exe

jtr.id
m220

!This program cannot be run in DOS mode.
```

Behavioral Analysis

Environment for Analysis

Launch the VMWare Application and launch the WinXP Pro (Created for this analysis) and Linux Red Hat 9.0 (From the CD handed out at the SANS Session in Las Vegas) instances. Log into each accordingly.

Obtaining Malware Specimen

Log into the GIAC website[3] using momgate and the given login information. From section 24.1.5 entitled "Malware Specimen for GREM Practical Assignment", download the Malware specimen. The Malware specimen is in a file named msrll.zip that is protected with the password "malware."
Once downloaded, use the "Drag and Drop" functionality available in VMWare to copy the Malware zip file to the WinXP Pro instance. For my Analysis the zip file was placed into `E:\Sample`. Using the built-in WinZip[4] menu feature, perform an extract of the file from the `E:\Sample` directory. WinZip will require the password "malware". There should now be a file named `msrll.exe` in the `E:\Sample` directory.

Checking the Specimen

From the Start/Run prompt, type in `cmd` and hit "Enter". Within this Command window type `e:\> md5sum msrll.exe`. The MD5 hash for `msrll.exe` is 84acfe96a98590813413122c12c11aaa. Launching `BinText 3.0` on the newly uncompressed file reveals that the executable is packed with Aspack.

Readying for the First Run

Launch `RegShot.exe` and select the Shot 1 button to take a snapshot of the Registry before the Malware Specimen is launched. Launch `Filemon.exe`, `TDImon.exe`, and `Regmon.exe` and clear each after pausing the capture of the data. Once all three applications have been cleared, restart capturing for each. Double click on the Malware Specimen `msrll.exe`. After approximately four seconds it will disappear from the `E:\Sample` directory. Launch the Task Manager and note that the Malware `msrll.exe` now runs under its own name. Select the `msrll.exe` process from the Task Manager list and click on the "End Process" button and click the "Yes" button to stop the Malware from running. Stop capturing with `Filemon.exe`, `TDImon.exe` and `Regmon.exe`. Save these logs to separate files and name them accordingly for the respective application, the malware and the date (Ex: `Filemon-msrll-20041204.log`). In

[3] http://giactc.giac.org/cgi-bin/momgate
[4] http://www.winzip.com

RegShot.exe click on the "Shot 2" button to capture the state of the Registry after the Malware was launched and the process was ended. Click on the "Compare" button and then save the resulting Log file following the above naming convention.

Analyze the Log files

The Malware will move itself from its point of origin on the Windows computer to the %windows%\System32\mfm\ as a file named msrll.exe with a matching MD5 hash. Note the change in Process ID as the location and active msrll.exe changes.

From the Filemon.exe log file:

```
569   9:22:13 AM msrll.exe:1020   CLOSE E:\Sample\msrll.exe
      SUCCESS
570   9:22:13 AM msrll.exe:1020   CLOSE
      C:\WINDOWS\System32\mfm\msrll.exe      SUCCESS

1042  9:22:14 AM msrll.exe:1168   DELETE      E:\Sample\msrll.exe
      SUCCESS
1043  9:22:14 AM msrll.exe:1168   CLOSE E:\Sample\msrll.exe
      SUCCESS
```

In addition, the Malware also creates a file named jtram.conf, which appears to be encrypted.

From the Filemon.exe log file:

```
1615  9:22:30 AM msrll.exe:1168   WRITE
      C:\WINDOWS\system32\mfm\jtram.conf    SUCCESS    Offset: 0
Length: 53
```

The Regmon.exe log file shows the Malware querying many Dynamically Linked Libraries for information on the System settings. The TDImon.exe log file shows the Malware Specimen is listening on TCP port 2200 as shown below:

```
      156   49.49251315     msrll.exe:1168   81697028
            TDI_SET_EVENT_HANDLER TCP:0.0.0.0:2200
            SUCCESS         Error Event: NULL
      157   49.49256037     msrll.exe:1168   81697028
            TDI_SET_EVENT_HANDLER TCP:0.0.0.0:2200
            SUCCESS    Disconnect Event: NULL
      158   49.49259277     msrll.exe:1168   81697028
            TDI_SET_EVENT_HANDLER TCP:0.0.0.0:2200
            SUCCESS    Receive Event: NULL
      159   49.49263133     msrll.exe:1168   81697028
            TDI_SET_EVENT_HANDLER TCP:0.0.0.0:2200
            SUCCESS    Expedited Receive Event: NULL
```

```
160   49.49307272     msrll.exe:1168  81697028
      TDI_SET_EVENT_HANDLER TCP:0.0.0.0:2200
      SUCCESS    Chained Receive Event: NULL
161   49.49316575     msrll.exe:1168  81697028
      IRP_MJ_CLEANUP  TCP:0.0.0.0:2200
```

This may be easier to view using the “netstat –an” command.

Network Analysis with SNORT

Extract another `msrll.exe` from the `msrll.zip` file using the password “malware” without the quotation marks. The `MD5sum` creates the same MD5 hash as before. Login to the Red Hat Linux 9.0 – REM VMWare Instance and launch the SNORT[5] network sniffer software using the tee command to split the output to both STDOUT and a file. To do so, execute the following command:

```
#  snort –vd | tee /tmp/sniff-msrll.log
```

Nothing much is occurring at all

The original VMWare session for the WinXP Pro instance is set to VMNet5. When the Red Hat Linux 9.0 – REM is set to this network as well communications between the two are lost. Moving both back to the Host Only Network requires shutting down both instances. The commands `Start Shutdown` can be used for the WinXP Pro instance and `shutdown –h now` can be used on the Red Hat Linux instance. Settings also need to be adjusted to use the Host Only Network. (NOTE: I noticed at this point in the practical that the WinXp Pro instance was also missing USB Controller Hardware so I added it as well).

Continuing on

Start up both instances again and login to each with each respective user and password pair. Launch SNORT on the Linux instance using the following command:

```
#  snort –vd | tee /tmp/sniff-msrll2.log
```

Switch to the WinXP Pro instance, log in and launch the previously unzipped msrll.exe file by double clicking it. After a few seconds the file will disappear. Confirm the move of the file to the %windir%\System32\mfm directory and MD5sum to make sure it is still the same file.
Switch back to the Red Hat Linux session. SNORT log file information should

[5] http://www.snort.org

be visible on the screen. The SNORT log shows a request for DNS information using UDP port 53 from the infected machine to the 192.168.147.x subnets gateway located at 192.168.147.1. The request will be for the IP address of `collective7.zxy0.com`. Use "Control-z" to pause the SNORT application. Next at the command prompt type in:

```
# ps  -ux
```

This command will give the Process Identification number (PID) for the SNORT process that has just been stopped. Use the `kill` command:

```
# kill -9 (SNORT PID)
```

The following shows the DNS request on UDP port 53 in the SNORT log file:

```
12/06-15:05:38.058348 192.168.147.129:1027 -> 192.168.147.1:53
UDP TTL:128 TOS:0x0 ID:374 IpLen:20 DgmLen:66
Len: 38
00 39 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C
.9...........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.....
```

Next, use the following command to determine the IP address of the Red Hat Linux 9.0 VMWare instance.

```
Ipconfig eth0
```

The IP address for the Red Hat Linux instance will display as 192.168.147.128. Switch to the WinXP Pro VMWare instance and launch The Task Manager and end the `msrll.exe` process. Edit the `hosts` file located in the `%windir%\System32\dirvers\etc` directory. Add to the end of the hosts file the IP address of the Red Hat Linux instance and match this to the `collective7` name as follow:

```
192.168.147.128 collective7.zxy0.com  collective7
```

Save this change. Next, unzip another `msrll.exe` to the `E:\Sample` directory. Double click the file to launch this application. After a few seconds it will disappear again . Switch to the Linux session and launch SNORT using:

```
#  snort -vd | tee /tmp/sniff-msrll3.log
```

The same DNS request for `collective7.zxy0.com` will appear followed by attempts to connect to `collective7.zxy0.com` on TCP port 6667, a typical IRC port. The following SNORT log file excerpt shows the Malware looking for the IRC server:

12/06-15:07:51.457856 ARP who-has 192.168.147.128 tell 192.168.147.129

12/06-15:07:51.457973 ARP reply 192.168.147.128 is-at 0:C:29:A5:4E:4E

```
12/06-15:07:51.458245 192.168.147.129:1033 ->
192.168.147.128:6667
TCP TTL:128 TOS:0x0 ID:375 IpLen:20 DgmLen:48 DF
******S* Seq: 0x3EF4C409  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Stop the SNORT application using the following commands:

```
Control-z
ps –ux
kill –9 (SNORT PID)
kill –9 (PID of the tee /tmp/sniff-msrll3.log)
```

Switch to the WinXP Pro VMWare instance and launch the The Task Manager and end the msrll.exe process. Extract another msrll.exe from the msrll.zip file using the password "malware" minus the quotation marks. The `MD5sum` creates the same MD5 hash as before. Switch to the Red Hat Linux instance and start the IRC server with the following steps:

```
# SU – ircd
$ ./ircd
$ exit
# ps –u ircd
PID  TTY        TIME       CMD
1638 ?          00:00:00   ircd
# irc
```

This will connect you to the Internet Relay Chat server version 2.8 / hybrid – 6.3.1 which was created on Tue June 4, 2002 at 16:59:45. If the last line is not completed, you can use a Windows client like `mIRC 6.12` to connect to the `collective7.zxy0.com 6667` IRC server. In the Red Hat Linux instance start the SNORT sniffer application and begin logging to the STDOUT and to a log file using the following command.

```
# snort –vd | /tmp/sniff-msrll4.log
```

Extract another `msrll.exe` from the `msrll.zip` file using the password "malware" minus the quotation marks. The `MD5sum` will create the same MD5 hash as before.

IRC session

Connect to the IRC server either through the Red Hat Linux instance using the command:

```
# irc
```

or by launching a Windows32-based IRC client like `mIRC 6.12`. From the SNORT log file you will see that the Malware specimen connects to the IRC server on TCP port 6667.

The IRC server connection being established is displayed below:

```
12/06-15:07:51.469686 192.168.147.128:6667 ->
192.168.147.129:1033
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
***A**S* Seq: 0xBC345638  Ack: 0x3EF4C40A  Win: 0x16D0  TcpLen:
28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

12/06-15:07:51.481980 192.168.147.129:1033 ->
192.168.147.128:6667
TCP TTL:128 TOS:0x0 ID:376 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x3EF4C40A  Ack: 0xBC345639  Win: 0xFAF0  TcpLen:
20
```

The infected machine sets up an ident server that listens on TCP port 113. This will gather information on computers accessing the infected machine.

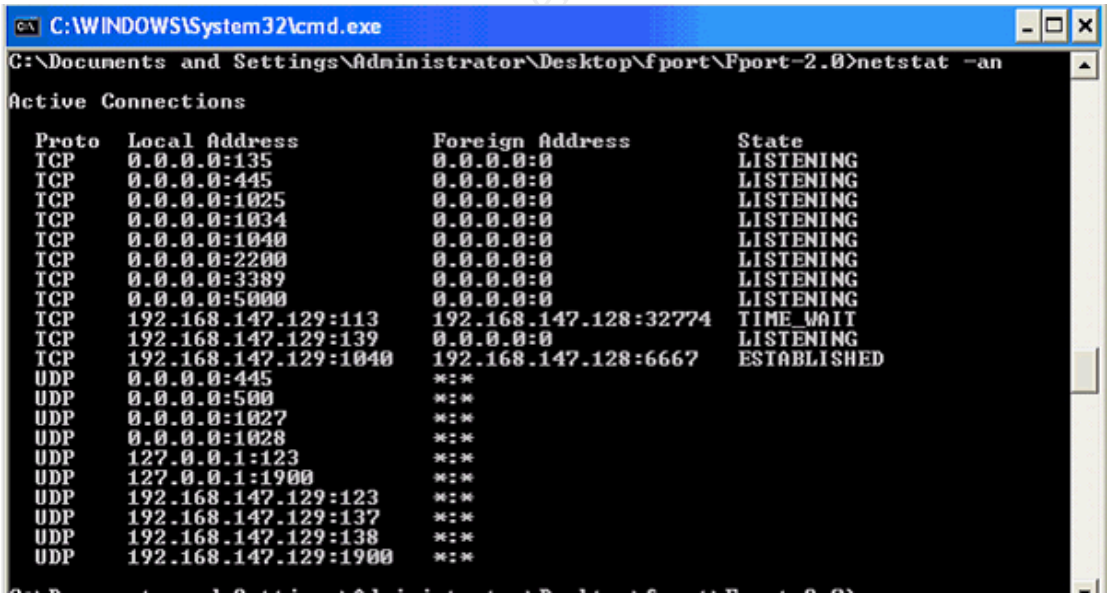


Figure 6: IRC connection and request for identd information.

The same identd information is displayed below as seen in the SNORT log file:

```
12/06-15:07:51.498045 192.168.147.128:32771 ->
192.168.147.129:113
```

```
TCP TTL:64 TOS:0x0 ID:13924 IpLen:20 DgmLen:60 DF
******S* Seq: 0xBCCFEE13  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 448631 0 NOP WS: 0

12/06-15:07:51.507899 192.168.147.129:113 ->
192.168.147.128:32771
TCP TTL:128 TOS:0x0 ID:377 IpLen:20 DgmLen:64 DF
***A**S* Seq: 0x3EF61675  Ack: 0xBCCFEE14  Win: 0xFAF0  TcpLen:
44
TCP Options (9) => MSS: 1460 NOP WS: 0 NOP NOP TS: 0 0 NOP NOP
SackOK

12/06-15:07:51.508153 192.168.147.128:32771 ->
192.168.147.129:113
TCP TTL:64 TOS:0x0 ID:13925 IpLen:20 DgmLen:52 DF
***A**** Seq: 0xBCCFEE14  Ack: 0x3EF61676  Win: 0x16D0  TcpLen:
32
TCP Options (3) => NOP NOP TS: 448632 0
```

The Malware will next join the #mils IRC channel using a randomly generated user and nick.

```
[root@localhost root]# nc -l -p 8080
USER XdRkDBbKP localhost 0 :apIPRdAVbnqbaQzTRdSXTLxDSBsV
NICK MfEbeNhulhUF
[root@localhost root]# nc -l -p 9999
USER IryMtmOdwIlWa localhost 0 :vBTRCrwUcUskorkVShslyxGeVvkDhrrhnq
NICK gADXZKNBEAk
[root@localhost root]# _
```

Figure 7: Randomness of user and nick

The TCP ports 6667, 9999 and 8080 are cycled through when connecting to the IRC server. It will stop once it has found the IRC server on the hosting machine, collective7.zxy0.com.

Attempted Interaction with the Malware Specimen

On the listening TCP port 2200 attempts were made from the RH Linux machine to connect to the infected machine using telnet and ftp. This attempt resulted in a shell like prompt of "#:". Several attempts to type commands did not result in any information returned to the sessions on the RH Linux machine.

```
root      1544  0.0  0.1  1344  392 tty6     S    10:07   0:00 /sbin/mingetty tt
root      1702  0.0  0.3  2252  976 ?        S    11:03   0:00 login -- root
root      1703  0.0  0.4  4308 1376 tty1     S    11:04   0:00 -bash
root      1963  0.0  0.2  2612  664 tty1     R    13:50   0:00 ps -ux
[root@localhost root]# telnet 192.168.147.129 2200
Trying 192.168.147.129...
Connected to 192.168.147.129.
Escape character is '^]'.
#:_
```

Figure 8: Attempted connection on port 2200 using telnet

```
[root@localhost root]# ftp 192.168.147.129 2200
Connected to 192.168.147.129 (192.168.147.129).
#:
```

Figure 9: Attempted connection on port 2200 using ftp

Moving logs from the Linux Instance

At the Las Vegas SANS Conference in the Fall of 2004, Lenny Zeltzer taught a Reverse-Engineering Malware class[6] that illustrated how `NetCat` can be used to create a unidirectional connection between the Red Hat Linux instance and the WinXP Pro instance. This type of connection can be utilized to move log files from the Red Hat Linux to the WinXP Pro Instances.
To create this connection in the WinXP Pro instance perform the following command:

```
C:\>nc -l -p 5555
```

In the Linux instance perform the following command:

```
# cat /tmp/sniff-msrll.log | nc 192.168.147.129 5555
```

Do the same for the remaining `sniff-msrll`(numbered).log files

Dynamically Link Libraries

[6] SANS Reverse-Engineering Malware: Tools and Techniques, Hands-On, 2004 Pg 2-41.

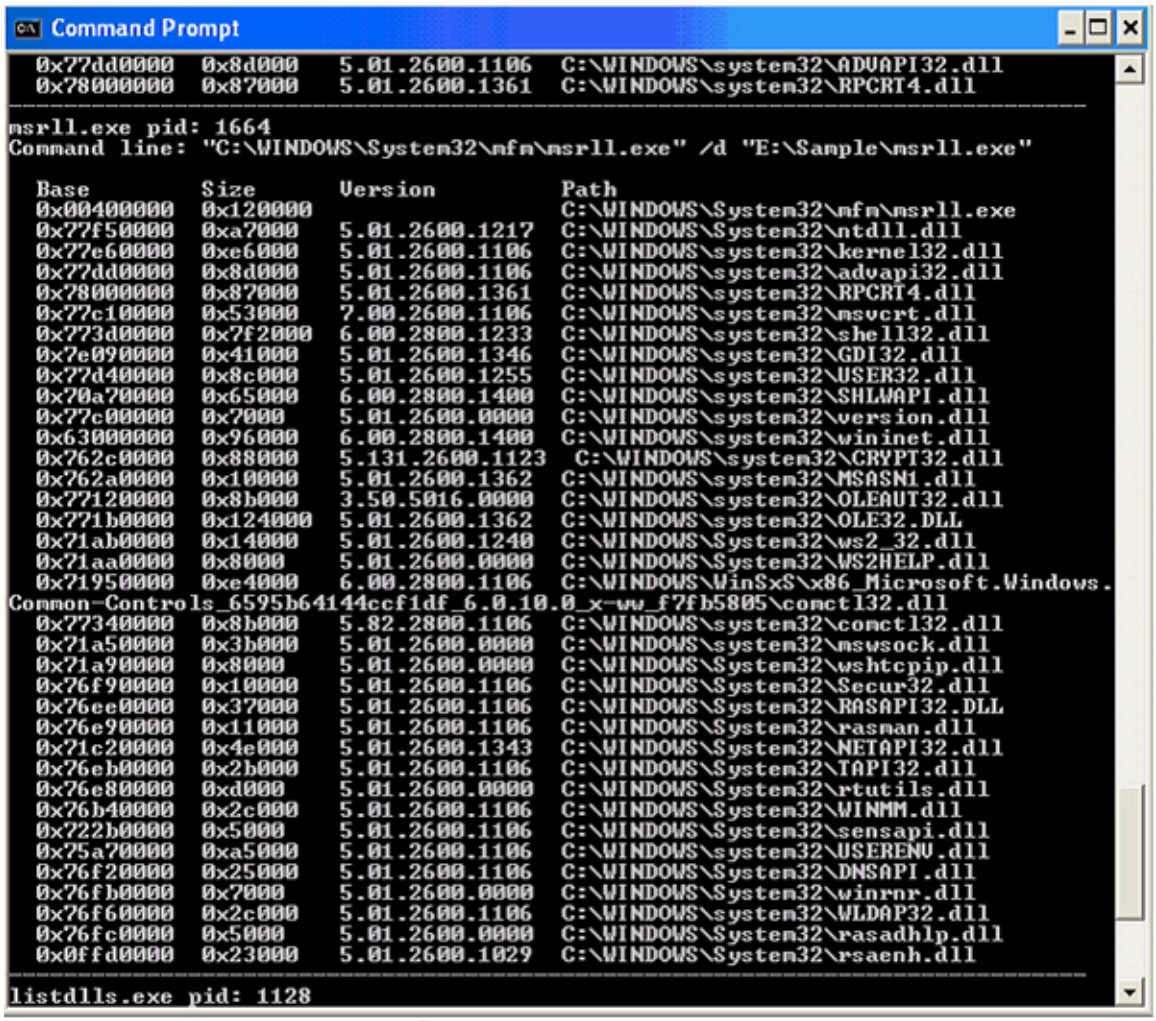


Figure 10: `listdlls.exe`, list of Dynamically Link Libraries used by msrll.exe

The system's Dynamically Link Libraries (DLLs), as seen in Figure 10, will be used by the Malware specimen to interact with the system and connect through the network to the IRC server. The Malware specimen `msrll.exe` can be seen in this screenshot below from the program `listdlls.exe`.

The Malware will add registry entries to run itself as a service when the infected machine is started up. It will add values to the registry entries it creates. It changes the values in the
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed`

In Windows 9x environments the Malware will create a Registry entry in:

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run`

Using the value:

```
Rll enhanced drive = "%System%\mfm\msrll.exe"
```

In Windows NT, Windows 2000 and Windows XP the Malware will create an entry in:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm
```

This will cause the Malware in either environment to restart when the machine it has infected is rebooted.

The `RegShot.exe` compare log from the Windows XP is shown below:

```
REGSHOT LOG 1.61e5
Comments:
Datetime:2004/12/4 15:21:34  ,  2004/12/4 15:25:37
Computer:VMWAREXP , VMWAREXP
Username: ,

----------------------------------
Keys added
----------------------------------
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Security
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Securit
y
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
OpenSaveMRU\LOG
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\
.LOG
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\
.LOG\OpenWithList
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDoc
s\.LOG
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\ShellNoRoam\BagMRU\4
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\ShellNoRoam\Bags\27
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\ShellNoRoam\Bags\27\Shell
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows NT\CurrentVersion\TaskManager

----------------------------------
Values deleted
----------------------------------
HKEY_USERS\S-1-5-21-1935655697-1715567821-725345543-
500\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDoc
```

```
s\Folder\3: 4E 00 65 00 74 00 43 00 61 00 74 00 00 00 46 00 32
00 00 00 00 00 00 00 00 00 00 00 4E 65 74 43 61 74 2E 6C 6E 6B
00 00 2C 00 03 00 04 00 EF BE 00 00 00 00 00 00 00 00 14 00 00
00 4E 00 65 00 74 00 43 00 61 00 74 00 2E 00 6C 00 6E 00 6B 00
00 00 1A 00 00 00

----------------------------------
Values added
----------------------------------
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Security\Se
curity: 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\ErrorContro
l: 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\Securit
y\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\ErrorCo
ntrol: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ImagePa
th: "C:\WINDOWS\System32\mfm\msrll.exe"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Display
Name: "Rll enhanced drive"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ObjectN
ame: "LocalSystem"
```

```
---------------------------------
Values modified
---------------------------------
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 05
A9 8E C4 4A 70 B9 28 E5 23 3F 0C E7 50 3D 36 58 CA 39 65 66 46
46 2E 9B B0 E1 B1 48 9F AA D7 21 06 1A 81 CB BB CA DA C9 9B 28
AC F2 43 6D 60 FF 9A E1 26 6A F4 4B 89 96 AA 4C 41 F6 A6 8C 33
6E 90 55 7A 2A E8 7B CE 9F 20 A3 6A 5B C1 D6 70
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: F0
CB 37 6E AB 38 EF 3B 26 9A C4 02 F0 DE A3 D3 5C A4 ED 20 B2 99
B9 F6 11 99 00 77 4D 08 76 74 FD 25 18 F2 C2 E9 68 BA 8D BB 5E
13 9D 36 02 4F E4 77 DD D0 E8 6B 3E 6E 5E 77 87 21 50 8D C8 C7
50 41 BA 72 C2 00 E2 5B 45 95 23 CA 5A 9D 31 7B
```

The information written to the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm` is information needed to start the service. The information written to the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet001\Services\mfm` helps the Malware to still start as a service even after the owner of the infected machine attempts to use the F8 troubleshooting option and then selects boot with last known good.

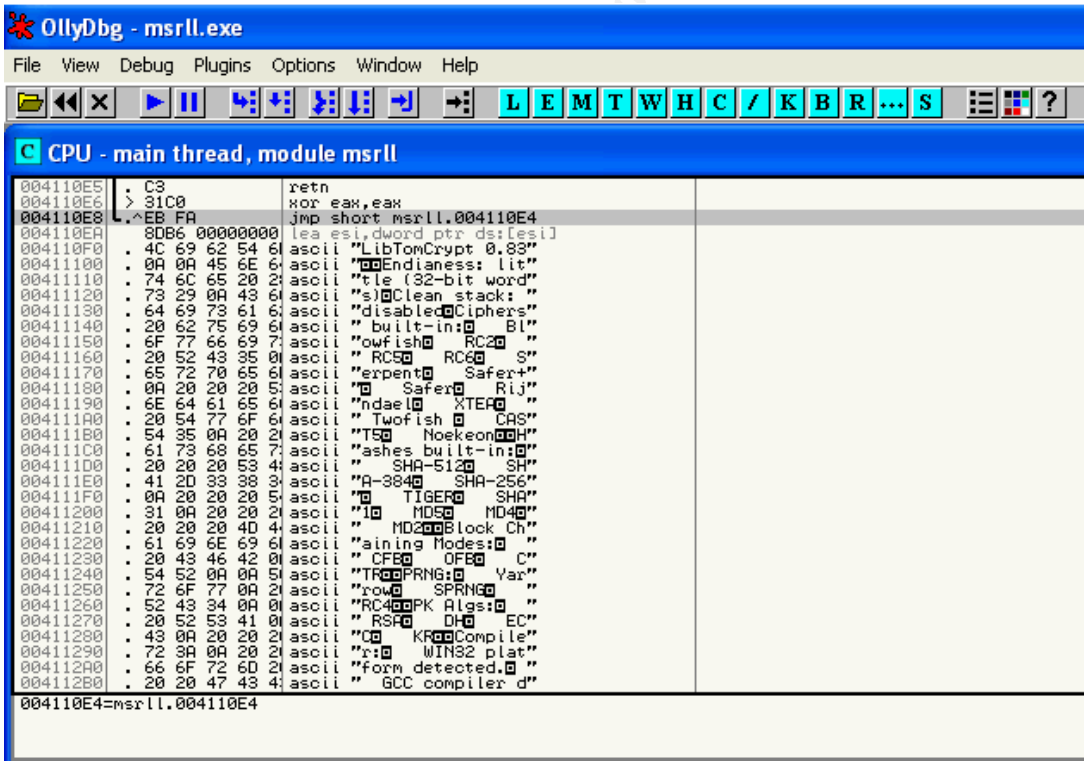


Figure 11: The Malware's Built-in Ciphers.

The Cryptographic sections help with encrypting the `jtram.conf` data file and decrypting it. There are several encryption algorithms available in the Malware.

The built in Ciphers from the output from `BinText 3.0`:

```
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:

File pos   Mem pos      ID   Text
========   =======      ==   ====

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
```

Code Analysis

Checking for Strings and Unpacking

Loading the Malware specimen into `BinText 3.0` just after it is unzipped from the `msrll.zip` file will show several readable strings, including `Aspack`. A Google.com[7] web search for "Aspack unpacking" resulted in Aaron's Homepage located at http://www.exetools.com/unpackers.htm[8] which included several unpackers. The first unpacker attempted is `UPX` to confirm that it is not packed with this packing tool. This was an attempt to follow the procedures from the SANS Lecture. The result will be a failure to unpack the `msrll.exe` file. The next attempted unpacker is `Aspack unpacker v2.0`. It will also fail to unpack the `msrll.exe` Malware specimen. Looking further down the page at Aaron's Homepage you will find a listing for `AspackDie 1.3d`, which contains support for an unknown Aspack version. `AspackDie 1.3d` is a program created by Yoda the creator of `LordPE` Memory dumping software. An attempt to unpack the Malware specimen using `AspackDie 1.3d` will prove successful. `AspackDie 1.3d` will find the Malware file is packed with ASPack version 2.12. The unpacked version of the Malware specimen will be 1.12 MB (1,175,552 bytes) and have an MD5 hash of:

```
E:\Sample>md5sum unpacked.exe
dc0c6b598c87f8a7d5c0bcb75ee5d6ea *unpacked.exe
```

Unpacking with AspackDie 1.3d can be completed by dragging the Malware specimen over the top of the AspackDie application and releasing the mouse button. The result is a small window generated by the AspackDie application that describes the results of the unpacking process, See Figure 5 above.

Further searching after unpacking

Within the unpacked Malware sample there will be a line that containing mIRC v6.12 Khaled Mardam-Bey. A Google.com[9] search for this string will supply a link to http://www.mirc.com/index.html[10] which will have information about the IRC client and the creator of the software. The site will include a useful help file on mIRC 6.12 software that explains the usage of common IRC commands. This can lead one to believe that the Malware uses this sting to represent itself to other IRC channel users if they query the version of the Malware IRC user, like camouflage.

[7] http://www.google.com
[8] http://www.exetools.com/unpackers.htm
[9] http://www.google.com
[10] http://www.mirc.com/index.html

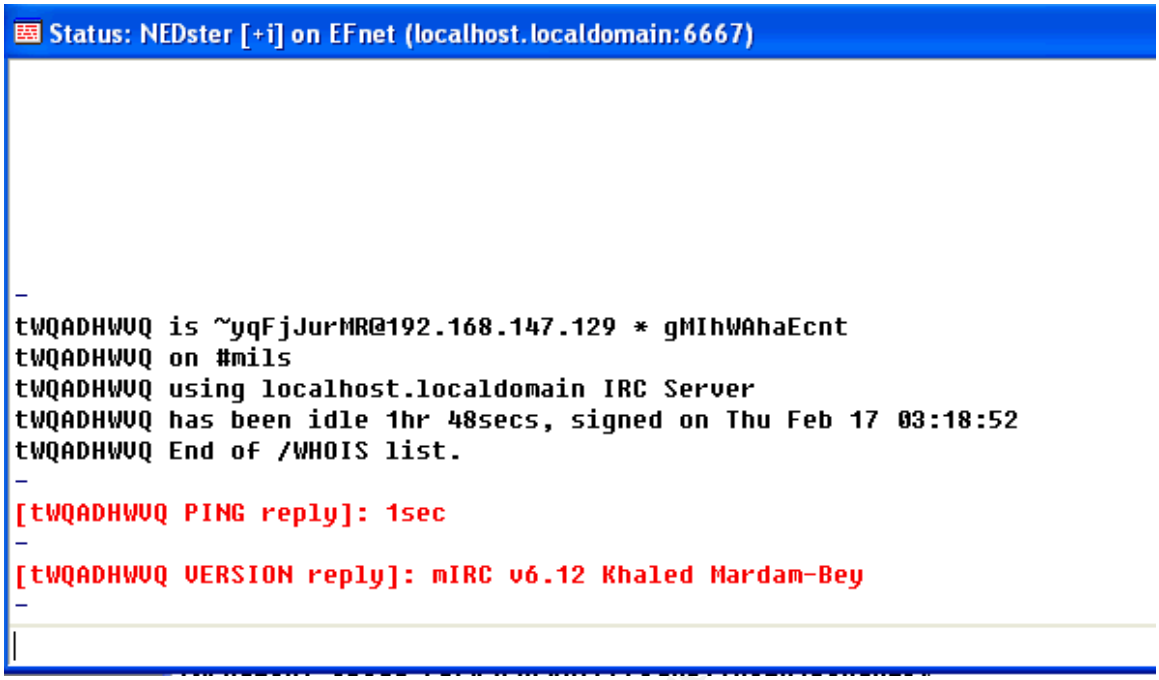


Figure 12: Whois, ping and version performed on the Malware client.

From the version command we can see that the `mIRC v6.12 Khaled Mardam-Bey` found earlier in the strings of the Malware is used as camouflage, hiding the fact that it is Malware.

Another interesting string is `m220 1.0 #2730 Mar 16 11:47:38 2004`, which looks similar to the name, version, build and build date of the Malware.

```
0040BED6                call    sub_40E7B0
0040BEDB                add     esp, 0Ch
0040BEDE                mov     edx, offset aM220 ; "m220"
0040BEE3                test    eax, eax
0040BEE5                jz      short loc_40BEEA
0040BEE7                mov     edx, [eax+4]
0040BEEA
0040BEEA loc_40BEEA:                            ; CODE XREF: sub_40BE48+9D↑j
0040BEEA                push    edx             ; lpName
0040BEEB                push    1               ; bInitialOwner
0040BEED                lea     eax, [ebp+MutexAttributes]
0040BEF0                push    eax             ; lpMutexAttributes
0040BEF1                call    CreateMutexA
0040BEF6                mov     ds:hMutex, eax
0040BEFB                add     esp, 4
0040BEFE                test    eax, eax
0040BF00                jz      short loc_40BF0E
0040BF02                call    GetLastError
```

Figure 13: Mutex “m220” setting code from IDA Pro.

The Malware also will set a Mutex of “m220” to keep other instances of it from starting up. There are several commands in the strings of the `msrll.exe` file that look like they are able to control and infect a machine.

Below is a list of embedded strings that may be special or unique to the

Malware and code:

Location	Command	Location	Command	Location	Command
409345	"?si",0	004093C1	"?die",0	409436	"?exec",0
409349	"?ssl",0	004093C6	"?md5p",0	0040943C	"?ps",0
0040934E	"?clone",0	004093CC	"?free",0	409440	"?kill",0
409355	"?clones",0	004093D2	"?raw",0	409446	"?killall",0
0040935D	"?login",0	004093D7	"?update",0	0040944F	"?crash",0
409364	"?uptime",0	004093DF	"?hostname",0	409456	"?dcc",0
0040936C	"?reboot",0	4.09E+12	"?fif",0	0040945B	"?get",0
409374	"?status",0	004093EE	"?!fif",0	409460	"?say",0
0040937C	"?jump",0	004093F4	"?del",0	409465	"?msg",0
409382	"?nick",0	004093F9	"?pwd",0	0040946A	"?kb",0
409388	"?echo",0	004093FE	"?play",0	0040946E	"?sklist",0
0040938E	"?hush",0	409404	"?copy",0	409476	"?unset",0
409394	"?wget",0	0040940A	"?move",0	0040947D	"?uattr",0
0040939A	"?join",0	409410	"?dir",0	409484	"?dccsk",0
004093A0	"?op",0	409415	"?sums",0	0040948B	"?con",0
004093A4	"?aop",0	0040941B	"?ls",0	409490	"?killsk",0
004093A9	"?akick",0	0040941F	"?cd",0	409499	"VERSION*",0
004093B0	"?part",0	409423	"?rmdir",0	004094A8	"PING",0
004093B6	"?dump",0	0040942A	"?mkdir",0	004094AE	"IDENT",0
004093BC	"?set",0	409431	"?run",0		

Ports will also show up in the unpacked Malware specimen. These will be backup ports to connect to an IRC server. If the IRC server is down on port 6667 then the Malware will cycle through the alternate TCP ports, 9999 and 8080. Setting `NetCat` up on the Red Hat Linux instance to listen on these ports will reveal whether or not this is the information sent to the ports. Listening using `NetCat` can be done using the following command statement:

```
For port 8080: nc -l -p 8080
For port 9999: nc -l -p 9999
```

```
[root@localhost root]# nc -l -p 8080
USER XdRkDBbKP localhost 0 :apIPRdAVbnqbaQzTRdSXTLxDSBsV
NICK MfEbeNhulhUF
[root@localhost root]# nc -l -p 9999
USER IryMtmOdwIlWa localhost 0 :vBTRCrwUcUskorkVShslyxGeVvkDhrrhnq
NICK gADXZKNBEAk
[root@localhost root]# _
```

Figure 14: `NetCat.exe` commands with results.

More embedded strings from the Malware are below:

```
servers
collective7.zxy0.com,collective7.zxy0.com:9999!,collective
7.zxy0.com:8080
```

The Malware will connect to the IRC channel `#mils` which will also be seen in

the SNORT log files.

The SNORT log file will look as follows:

```
12/06-15:08:19.580789 192.168.147.128:6667 ->
192.168.147.129:1033
TCP TTL:64 TOS:0x0 ID:36437 IpLen:20 DgmLen:113 DF
***AP*** Seq: 0xBC345C71  Ack: 0x3EF4C45F  Win: 0x16D0  TcpLen:
20
3A 6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61 6C
:localhost.local
64 6F 6D 61 69 6E 20 33 30 32 20 41 55 78 73 45  domain 302
AUxsE
6E 51 53 4D 20 3A 41 55 78 73 45 6E 51 53 4D 3D  nQSM
:AUxsEnQSM=
2B 72 47 73 73 4E 40 31 39 32 2E 31 36 38 2E 31
+rGssN@192.168.1
34 37 2E 31 32 39 20 0D 0A                       47.129 ..
```

Malware will join the #mils IRC Channel and perform the /who #mils command:

```
12/06-15:08:23.568340 192.168.147.129:1033 ->
192.168.147.128:6667
TCP TTL:128 TOS:0x0 ID:388 IpLen:20 DgmLen:53 DF
***AP*** Seq: 0x3EF4C45F  Ack: 0xBC345CBA  Win: 0xFAA7  TcpLen:
20
4A 4F 49 4E 20 23 6D 69 6C 73 20 3A 0A           JOIN #mils :.
```

Also evidenced in the output from BinText 3.0 as seen below:

```
irc.chan
#mils
```

When the Malware is renamed and launched on a Windows machine it will name itself back to the original msrll.exe as it is moved into the %windir%\System32\mfm directory.

```
jtr.bin
msrll.exe
```

The Malware application msrll.exe will not run in DOS mode but will run on the Windows 32 platforms.

```
!This program cannot be run in DOS mode.
```

The Malware specimen will create several Registry entries as well as querying the Registry for System Setting information. This can be seen in the screen shot below from IDA Pro.

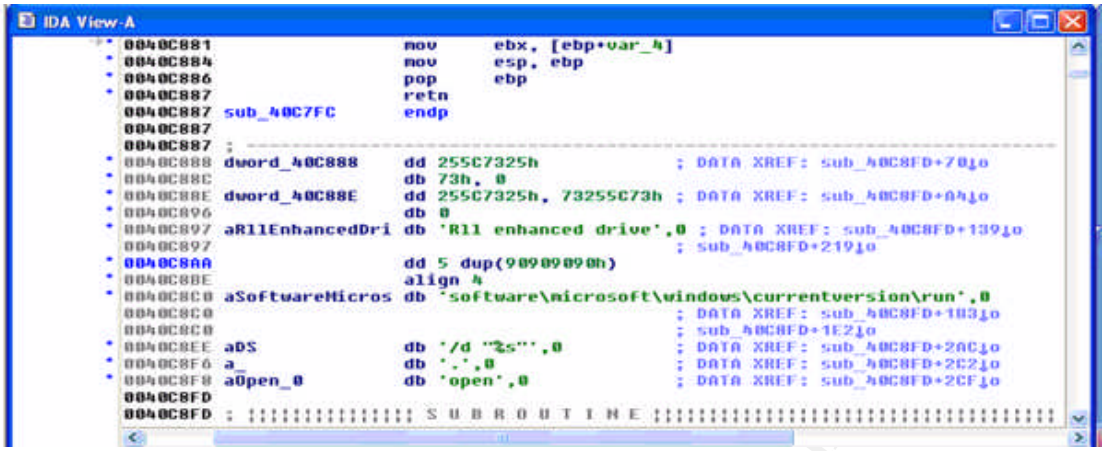


Figure 15: Code related to the Registry

The Connection to the IRC server through port 6667 will allow the controller of the Malware to get information to and from the infected system. The listening on TCP port 2200 created by the Malware allows the controller of the Malware to connect to the infected machine. The Malware continues to listen on TCP port 1033 for information from the IRC session on channel #mils. This can be seen in the screenshot from `TCPview.exe` in Figure 16.

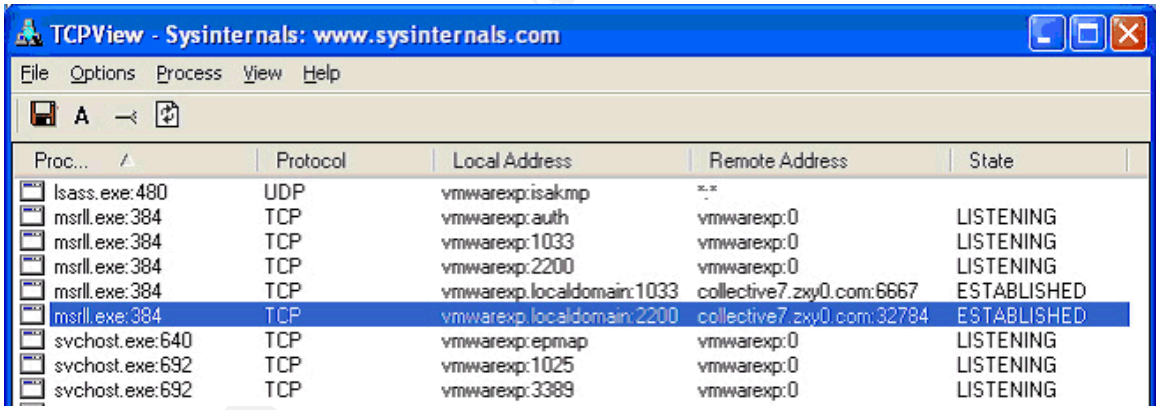


Figure 16: `TCPView.exe` look at listening and IRC server.[11]

Researching the content of the `jtram.conf` file

After far too many F7 key presses, within OllyDBg, to count had elapsed, many starting from a copy of the `unpacked.exe` file and and dragging it to the OllyDBg application. This however does not work well, in the process of launching the `unpacked.exe` malware it copies itself to the directory structure it creates `C:\Windows\System32\mfm` (jtr.home) as `msrll.exe` then it deletes

[11] http://www.sysinternals.com

itself and removes itself from memory as the newly created `msrll.exe` starts up. It will build Registry entries to cause itself to launch when the infected machine is rebooted, either as an application to launch or as a service depending on the Windows Operating System of the machine the Malware is infecting. For Windows NT and above it even sets Registry entries that go back to the last known good, using the ControlSet001 under services. With the original unpacked.exe deleted OllyDBg is no longer able to track the process. With this learned, using the msrll.exe that is located in the `jtr.home` directory structure is a better method for researching the contents of the `jtram.conf` file.

First run of `msrll.exe` gives the following `jtram.conf` file:

```
e/8RAOYI3FLMh+yKj9pMUCVRkuDs5h5wngcxcaiQdHjcphb4PQ==
mAMRAEVLDpg9BZxz9sd0fHavgu9hmXsW0eBgy8Y0XXtLAGN1YQ==
ZAARAPPHUHQYTgCAPVh2cSYd32RjzrSSEtiGkYgC1Rd/6/2k9A==
5/4RAIDqE+vgYRE0k1nHs1+ChPdAoB4HOtWB1XYLhJwxQlJTAw==
fwARALLMuphCrKzYrrbw9vxucWEIP34RMD9q19pq61894yaiqg==
CP8RAEKZS7CN7YVDWXGpoallRbZfVkUjhp+nbSIV1MZA4IHRKQ==
hgERAKdZVm0LFPR6keqMUS/EsSUtHiqZYg77Vu1vm+RcIokXPA==
7QQRADsf/WwjQkQPO3OjNPZrbEzDE13JbjWYa6sZf08eNmIrrg==
4P1KAKtctX/nAPsmZVL1k2LzVgucUR0HXbUog4DbAaupBzbNTLDjiEg62f0++yF0
4MxGRPiE2OnG1LAbKBUz4+Kjq8RBoCSFR9yws9Vs5Er4CCoBVbOypGXQq167KQ==
bwIRALvnK3B4N/gxQwx/dYmlEORH1hxqbRjgW8NNWqK9jq1BPA==
gf0RAASOxOfjVWTiHNjlDcsgLccsY5L5u3GYjJLFhpKRwiRxLw==
If4RAOJRzZMgZvTM1E6nK4lyDF+QhGqmmV+HyLb9J8VSp0qGCg==
XwARAJhrzuNYZARJAJOgpUiVze8hcnPUh5J8RBxVQOFov8rs/g==
h/4RAAgFeBkoFXgx9ISom82cvkQ8sVDVThOlzqe/ENeXSbyJdA==
7AIjAPsO35efvxXMxBruZWibzVTMa+hvALGKEkz49/o9RSPyb+RBJ2VbsxV8DjTL
LjRQVWhDkQ==
6gARAKpsNKR24L6CYMvWs09HbM5ss6t6FXHsqCThdzAtwX29nw==
nAERAFwd6X74Lq5qbbBsvp4efDhl3CDzIPunB60nJehOxV2mog==
OQEjAN4noYvRhk6SPWr8vU9YF80+d86wMHTNCzWpvXG0QR9UP9C8VgIBaNQM+Z+0
PvhEsnn/RA==
```

Second run of `msrll.exe` gives the following `jtram.conf` file:

```
+v0RAJ2f67PEhA0AvSLRjT/x0MqUP3UnkOGvPKR/U1ntC7AmSw==
mgARAC4uGR1kve76IYiRQZBRCV/LrokwRrBUOOq2c0wRJ/jOGQ==
hv8RAFUY4QqccSpOlhpmn1eM0Cus3pzTXau+l/lj3PLYNAxhDA==
awERAPVEqxMIxeSOKa5o942ycetPNjBi7gJHyekNIH2kcbVhfg==
6v4RAJ7oMAvcKJcSkxYWe8s88/cnXv7UhjqHHELkgKj940PzIg==
awARAA6XCSq+mRVwvVcULH57RLzD47SujopwTSsttzMYkiogRg==
8v0RAKeO8dvaGOtlwgc0yrqOKHjqsP0x0KWONFeT33+2dLO4Ew==
Jf0RAO6Ol+n0/+ILbZ/R4KoCaemm5XcoluCJOOOkPIbjzy1Npw==
Lv5KAMshsBBJBF2bcpwHTAht/mPdOorW2oMViF10p9/rQo2hSVL/IkWlkiZ1LPTp
Mw+4R5761noB12GwgfDVHe5MGzxSLkesDlMIlp+6+K7XCfCJp87146n7tylurg==
kv8RAA7EY/3mmrLOdSyianpbLYFNvQAus4WtaZi6xal+Eev3aA==
4QARAPMmOrYPLwvVCDJbe2fiHGab8DyVvj3qbSFzSYKpjhGM7Q==
df4RAL1rI9O5slESRmSOCPoIgbaJotqoStexBL7mQ/IqNdvsfQ==
LAARAJXhEhWlCcxBH93Dva55p0xg8nvG+crZa/yOClZIKwnZVA==
bgIRAFk+rF0Q6uIs9RVlTzNp/0/QlHjll1/XYocQ3qJv4CcbdQ==
```

```
hwAjAOJBFn529ZIUoAccONjqxYDN2XWrVwhQmR1qq0RQnnfxZBo7jA7MwS59cZGX
+rmyNMwoDw==
av8RAMWT6ATYqw98Htz51rnH7Ik8S7Bxy3i1FQ7KZwXzh/Jmfg==
CQERAKvOIqtavymlPNJMKBF09iIA1oxGcUZw7eyJOGDosn3T+Q==
YP4jAFM19kCjRb6Cz5WBktuIZrkPQLlQ/B87YWxLQ5XQUTxnwLmO5/yF8t0mxKcv
8BW7wHznGw==
```

Checking to see what the Malware did if the `jtram.conf` file was deleted and it had to build another copy of the data file was the starting point. This file was removed from the `C:\Windows\System32\mfm` directory. To begin debugging with OllyDbg, the current running Malware process was ended using Task Manager. Highlight the `msrll.exe` process and hit the End Process button and then the yes button when asked if ending the process is really what is wanted. Once the current process is ended debugging with OllyDbg can begin by simply dragging and dropping the Malware from the `jtram.home` directory, `C:\Windows\System32\mfm` onto the OllyDbg application and release the mouse button. OllyDbg will launch and the focus will be the `msrll.exe` Malware specimen.

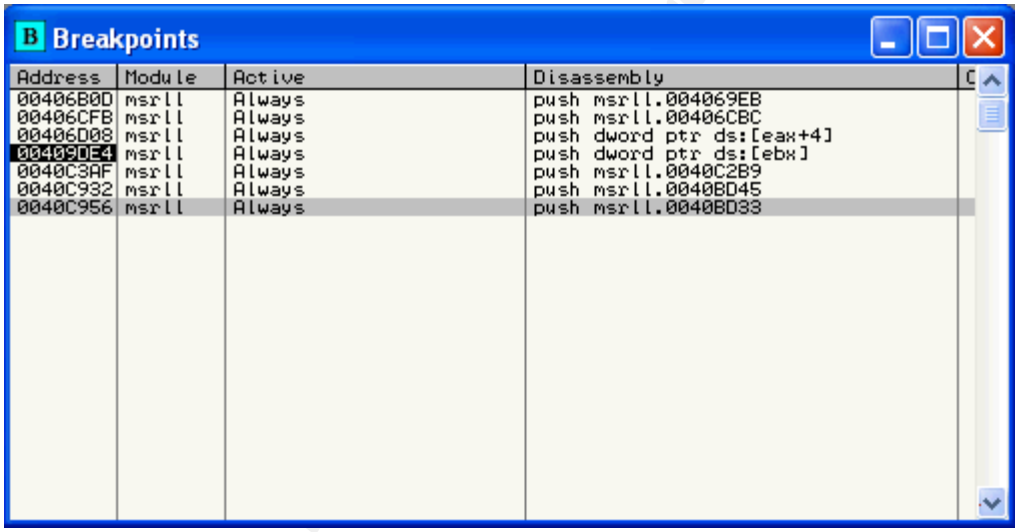


Figure 17: Breakpoints used in the investigation.

Looking at memory marker `00409DE4` in the `msrll.exe` module near a piece of coded data `"DiCHFc2ioiVmb3cb4zZ7zWZH1oM="` a section with `Arg1` and `Arg2` shows promise. Setting a breakpoint at this memory marker and running the Malware back to the breakpoint was very helpful and informative.

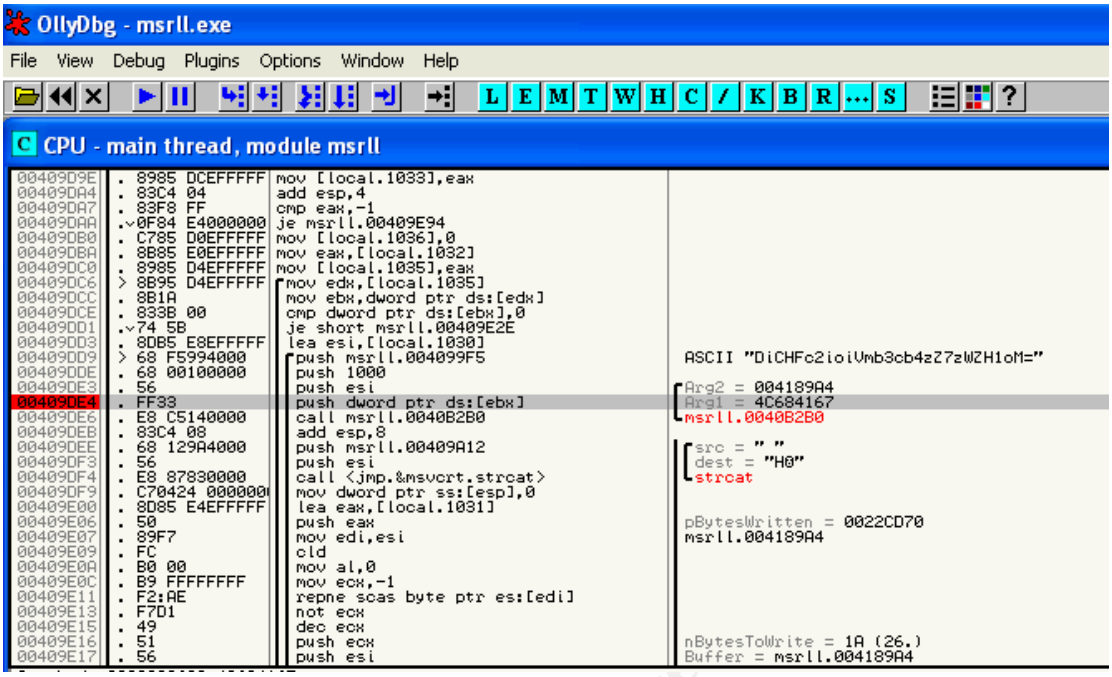


Figure 18: Section of code where the `jtram.conf` information is encoded.

Setting a breakpoint on the `00409DE4` memory address and hitting the F9 key allowing the running of the Malware and stopping at this set breakpoint results in the following information being revealed.

```
00409DE3  |. 56              ||push esi
; /Arg2 = 0022EE80 ASCII "collective7.zxy0.com"
00409DE4  |. FF33            ||push dword ptr ds:[ebx]
; |Arg1 = 003D5858 ASCII "set"

00409DE3  |. 56              ||push esi
; /Arg2 = 0022EE80 ASCII
"e/8RAOYI3FLMh+yKj9pMUCVRkuDs5h5wngcxcaiQdHjcphb4PQ== "
00409DE4  |. FF33            ||push dword ptr ds:[ebx]
; |Arg1 = 003D5878 ASCII "bot.port"

00409DE3  |. 56              ||push esi
; /Arg2 = 0022EE80 ASCII
"mAMRAEVLDpg9BZxz9sd0fHavgu9hmXsW0eBgy8Y0XXtLAGN1YQ== "
00409DE4  |. FF33            ||push dword ptr ds:[ebx]
; |Arg1 = 003D58A0 ASCII "2200"

00409DE3  |. 56              ||push esi
; /Arg2 = 0022EE80 ASCII
"ZAARAPPHUHQYTgCAPVh2cSYd32RjzrSSEtiGkYgC1Rd/6/2k9A== "
00409DE4  |. FF33            ||push dword ptr ds:[ebx]
; |Arg1 = 003D5910 ASCII "set"

00409DE3  |. 56              ||push esi
```

```
; /Arg2 = 0022EE80 ASCII
"5/4RAIDqE+vgYRE0k1nHs1+ChPdAoB4HOtWB1XYLhJwxQlJTAw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5930 ASCII "irc.quit"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"fwARALLMuphCrKzYrrbw9vxucWEIP34RMD9q19pq61894yaiqg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5958

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"CP8RAEKZS7CN7YVDWXGpoallRbZfVkUjhp+nbSIV1MZA4IHRKQ== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D59C8 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"hgERAKdZVm0LFPR6keqMUS/EsSUtHiqZYg77Vu1vm+RcIokXPA== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D59E8 ASCII "servers"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"7QQRADsf/WwjQkQPO3OjNPZrbEzDE13JbjWYa6sZf08eNmIrrg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5A08 ASCII
"collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy
0.com:8080"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"4P1KAKtctX/nAPsmZVL1k2LzVgucUR0HXbUog4DbAaupBzbNTLDjiEg62f0++yF
04MxGRPiE2OnG1LAbKBUz4+Kjq8RBoCSFR9yws9Vs5Er4CCoBVbOypGXQq167KQ=
= "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5AC0 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"bwIRALvnK3B4N/gxQwx/dYmlEORH1hxqbRjgW8NNWqK9jq1BPA== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5AE0 ASCII "irc.chan"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"gf0RAASOxOfjVWTiHNjlDcsgLccsY5L5u3GYjJLFhpKRwiRxLw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5B08 ASCII "#mils"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"If4RAOJRzZMgZvTM1E6nK4lyDF+QhGqmmV+HyLb9J8VSp0qGCg== "
```

```
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5B78 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"XwARAJhrzuNYZARJAJOgpUiVze8hcnPUh5J8RBxVQOFov8rs/g== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5B98 ASCII "pass"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"h/4RAAgFeBkoFXgx9ISom82cvkQ8sVDVThOlzqe/ENeXSbyJdA== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5BB8 ASCII "$1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"7AIjAPsO35efvxXMxBruZWibzVTMa+hvALGKEkz49/o9RSPyb+RBJ2VbsxV8DjT
LLjRQVWhDkQ== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5C48 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"6gARAKpsNKR24L6CYMvWs09HbM5ss6t6FXHsqCThdzAtwX29nw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5C68 ASCII "dcc.pass"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"nAERAFwd6X74Lq5qbbBsvp4efDhl3CDzIPunB60nJehOxV2mog== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5C90 ASCII "$1$KZLPLKDf$55isA1ITvamR7bjAdBziX."


Take2:

00409DE3  |. 56             ||push esi
; /Arg2 = 003D6308
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6308

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII "collective7.zxy0.com"
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5930 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"+v0RAJ2f67PEhA0AvSLRjT/x0MqUP3UnkOGvPKR/U1ntC7AmSw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5980 ASCII "bot.port"

00409DE3  |. 56             ||push esi
```

```
; /Arg2 = 0022EE80 ASCII
"mgARAC4uGR1kve76IYiRQZBRCV/LrokwRrBUOOq2c0wRJ/jOGQ== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D59D0 ASCII "2200"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"hv8RAFUY4QqccSpOlhpmn1eM0Cus3pzTXau+l/lj3PLYNAxhDA== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5B48 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"awERAPVEqxMIxeSOKa5o942ycetPNjBi7gJHyekNIH2kcbVhfg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5B98 ASCII "irc.quit"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"6v4RAJ7oMAvcKJcSkxYWe8s88/cnXv7UhjqHHELkgKj940PzIg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5BE8

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"awARAA6XCSq+mRVwvVcULH57RLzD47SujopwTSsttzMYkiogRg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5D60 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"8v0RAKeO8dvaGOtlwgc0yrqOKHjqsP0x0KWONFeT33+2dLO4Ew== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5DB0 ASCII "servers"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"Jf0RAO6Ol+n0/+ILbZ/R4KoCaemm5XcoluCJOOOkPIbjzy1Npw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5E00 ASCII
"collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy
0.com:8080"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"Lv5KAMshsBBJBF2bcpwHTAht/mPdOorW2oMViF10p9/rQo2hSVL/IkWlkiZ1LPT
pMw+4R5761noB12GwgfDVHe5MGzxSLkesDlMIlp+6+K7XCfCJp87146n7tylurg=
= "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D5FC8 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"kv8RAA7EY/3mmrLOdSyianpbLYFNvQAus4WtaZi6xal+Eev3aA== "
```

```
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6018 ASCII "irc.chan"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"4QARAPMmOrYPLwvVCDJbe2fiHGab8DyVvj3qbSFzSYKpjhGM7Q== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6068 ASCII "#mils"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"df4RAL1rI9O5slESRmSOCPoIgbaJotqoStexBL7mQ/IqNdvsfQ== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D61E0 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"LAARAJXhEhWlCcxBH93Dva55p0xg8nvG+crZa/yOClZIKwnZVA== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6230 ASCII "pass"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"bgIRAFk+rF0Q6uIs9RVlTzNp/0/QlHjll1/XYocQ3qJv4CcbdQ== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6280 ASCII "$1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"hwAjAOJBFn529ZIUoAccONjqxYDN2XWrVwhQmR1qq0RQnnfxZBo7jA7MwS59cZG
X+rmyNMwoDw== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6410 ASCII "set"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"av8RAMWT6ATYqw98Htz51rnH7Ik8S7Bxy3i1FQ7KZwXzh/Jmfg== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D6460 ASCII "dcc.pass"

00409DE3  |. 56             ||push esi
; /Arg2 = 0022EE80 ASCII
"CQERAKvOIqtavymlPNJMKBF09iIA1oxGcUZw7eyJOGDosn3T+Q== "
00409DE4  |. FF33           ||push dword ptr ds:[ebx]
; |Arg1 = 003D64B0 ASCII "$1$KZLPLKDf$55isA1ITvamR7bjAdBziX."
```

Pulling the important information together from this harvested from the running Malware during take2:

```
Take2:
"set"
"+v0RAJ2f67PEhA0AvSLRjT/x0MqUP3UnkOGvPKR/U1ntC7AmSw== "
```

```
 "bot.port"
"mgARAC4uGR1kve76IYiRQZBRCV/LrokwRrBUOOq2c0wRJ/jOGQ== "

"2200"
"hv8RAFUY4QqccSpOlhpmn1eM0Cus3pzTXau+l/lj3PLYNAxhDA== "

"set"
"awERAPVEqxMIxeSOKa5o942ycetPNjBi7gJHyekNIH2kcbVhfg== "

"irc.quit"
"6v4RAJ7oMAvcKJcSkxYWe8s88/cnXv7UhjqHHELkgKj940PzIg== "

003D5BE8
"awARAA6XCSq+mRVwvVcULH57RLzD47SujopwTSsttzMYkiogRg== "

"set"
"8v0RAKeO8dvaGOtlwgc0yrqOKHjqsP0x0KWONFeT33+2dLO4Ew== "

"servers"
"Jf0RAO6Ol+n0/+ILbZ/R4KoCaemm5XcoluCJOOOkPIbjzy1Npw== "


"collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy
0.com:8080"
"Lv5KAMshsBBJBF2bcpwHTAht/mPdOorW2oMViF10p9/rQo2hSVL/IkWlkiZ1LPT
pMw+4R5761noB12GwgfDVHe5MGzxSLkesDlMIlp+6+K7XCfCJp87146n7tylurg=
= "

 "set"
 "kv8RAA7EY/3mmrLOdSyianpbLYFNvQAus4WtaZi6xal+Eev3aA== "

 "irc.chan"
 "4QARAPMmOrYPLwvVCDJbe2fiHGab8DyVvj3qbSFzSYKpjhGM7Q== "

 "#mils"
"df4RAL1rI9O5slESRmSOCPoIgbaJotqoStexBL7mQ/IqNdvsfQ== "

 "set"
"LAARAJXhEhWlCcxBH93Dva55p0xg8nvG+crZa/yOClZIKwnZVA== "

 "pass"
 "bgIRAFk+rF0Q6uIs9RVlTzNp/0/QlHjll1/XYocQ3qJv4CcbdQ== "

 "$1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0"
"hwAjAOJBFn529ZIUoAccONjqxYDN2XWrVwhQmR1qq0RQnnfxZBo7jA7MwS59cZG
X+rmyNMwoDw== "

 "set"
"av8RAMWT6ATYqw98Htz51rnH7Ik8S7Bxy3i1FQ7KZwXzh/Jmfg== "

 "dcc.pass"
"CQERAKvOIqtavymlPNJMKBF09iIA1oxGcUZw7eyJOGDosn3T+Q== "
```

```
"$1$KZLPLKDf$55isA1ITvamR7bjAdBziX."
"YP4jAFM19kCjRb6Cz5WBktuIZrkPQLlQ/B87YWxLQ5XQUTxnwLmO5/yF8t0mxKc
v8BW7wHznGw== "
```

When the `jtram.conf` file is not deleted it still rewrites the file. Each time the Malware launches it rewrites the `jtram.conf` file. There are several built-in ciphers that might aid in this process. Though the resulting content of the file is different each time the data encoded is the same. The Registry entry that had the SEED written to it is used to generate the encryption of the `jtram.conf` file's data. This would make it near impossible to just decode the `jtram.conf` file with some scripted algorithm. This SEED Registry entry, `HKEY_LOCAL_MACHINE\ SOFTWARE\Microsoft\Cryptography\RNG\Seed`, is modified with the current SEED for each launching of the Malware specimen.

From this exercise we now have much needed information to help control and interact with the Malware specimen.

```
"pass" = "$1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0"

"dcc.pass" = "$1$KZLPLKDf$55isA1ITvamR7bjAdBziX."
```

Next we can attempt taking over the clients and controlling them through the IRC session on the IRC channel `#mils`. From information about controlling viruses with IRC found at `http://swatit.org/bots/`[12] and at the Las Vegas SANS Conference in the Fall of 2004, Lenny Zeltzer taught a Reverse-Engineering Malware class[13] that illustrated how to control a Malware specimen with IRC. Within the IRC session Commands that were found embedded in the Malware as strings should be able to run and reveal more information about the Malware and its functionality. This process started with trying the `PASS` command. `PASS` alone or with the password found in the `jtram.conf` file with variations of leading characters like `"!@", "!?", "?", "$!$","!@?"` nor `"!"` did not return results like those from the SANS Conference, while controlling the `Tnnbtib.exe` Trojan file. In the lecture Lenny Zeltzer was able to use the `!@login` command to connect to the Malware being investigated.

If the use of `"!pass $1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0"` had worked and control of the bots logged into the IRC session was obtained, the next steps would be to interact with the Malware and then remove the threat on the infected machines. Using embedded commands like `?rmdir` to remove the `C:\%windows%\System32\mfm` directory and then running `?die` or `?kill` to stop the `msrll.exe` process.

[12] http://swatit.org/bots/
[13] SANS Reverse-Engineering Malware: Controlling the Trojan, 2004 Pg 54-56.

Analysis Wrap-Up

Capabilities and What It Does

The Malware `msrll.exe` operates as a backdoor on the computer it is infecting. It is able to report statistics to the Malware controller via its connection to `collective7.zxy0.com`. It sets up a listening TCP port on 2200 to accept connection from the Malware controller. It is able to create directories, move files, copy files, compute MD5 sums, crash systems, reboot the infected machine, and accept login information from the controller. When launched initially it moves itself to a directory it creates at `%windir%\system32\mfm`. Malware `msrll.exe` runs under its file name in the The Task Manager and does not attempt to hide its presence. The Malware used mIRC version 6.12 created by Khaled Mardam-Bey[14] as camouflage, to hide that it was Malware, when a request for version information is sent to it. The Malware `msrll.exe` referred to as "m220" in the Malware code includes its own version information: `m220 1.0 #2730 Mar 16 11:47:38 2004`. The Malware creates a mutex called `m220` to keep other instances from starting. This m220 also might also relate to the TCP port 2200 that the Malware uses to listen for connections from its creator. It requests information on `collective7.zxy0.com` and once it locates the IP address it connects to a running IRC server with a default TCP port of 6667. The code also listed TCP ports 8080 and 9999, if the default IRC port is not found it will cycle through the other two TCP ports, looking for an IRC server on one of these ports on the server set in the code "collective7.zxy0.com". The Malware uses random usernames, nicks and joins the IRC channel `#mils`. Registry entries are created and populated with values that allow the Malware `msrll.exe` to run on reboot for Windows 98, Windows NT, Windows 2000 and Windows XP Professional machines that are able to do this functionality. Machines that are able to run services get `msrll.exe` setup to start as a service and display "Rll enhanced drive". Other Operating Systems were not available for installing into the VMWare environment as test machines.

Who would use the program?

The user of this malicious program would be someone with several possible goals. They might be using this program for bragging rights either having the biggest BOT army or to have completed the challenge of creating and controlling such an enterprise. They may also want to use hard disk space on machines that are not theirs for purposes like file sharing.

Defensive measures and elimination of current infections

Defending against this and similar Malware can begin with simply adding a firewall to the environment to introduce a barrier to intrusion. The firewall would be used to control traffic in and out of the protected internal environment.

[14] http://www.mirc.com/index.html

Blocking TCP and UDP traffic at a firewall would eliminate the beginning steps such Malware use to identify themselves and set up connections with their creators, namely the IRC servers and the created listening port on the infected machines. On a PIX firewall these rules can be added to stop internal traffic from infected machines from contacting the IRC session on the collective7.zxy0.com server.

```
access-list inside deny tcp any any eq 6667
access-list inside deny tcp any any eq 8080
access-list inside deny tcp any any eq 9999
```

Access from outside on the internet to the 2200 port would be in a state of deny by default. Additionally there would have to be a NAT set to allow those attempts and a rule to allow these connections from outside to inside.
Add adware and spyware removal software to the desktop and laptop machines that reside both inside and outside the firewall. Include antivirus software in this strategy. Most importantly, keep the environment updated with current patches and up-to-date signature files (virus.dat files). Establish regularly scheduled machine scans with these defenses. Keep Operating System software patched against the latest known exploits. Research on the web to ensure that vulnerabilities are known and defenses can be put in place if possible before malicious code is written to take advantage of it. Most viruses are designed to exploit sometimes known and sometimes unknown vulnerabilities in software that may be running on machines in the company's enterprise environment. Finally, be sure to include web browser software in any update and patching strategy.

Current Removal

To remove the current infection do debugging with the AspackDie 1.3d unpacked sample and determine the login password to access the Malware on the TCP port 2200 that is set to listening or send a command through IRC to killall. Use the IRC command `/who #mils` to get a listing of the machine IP addresses that are infected and use an application like DameWare[15] to login remotely and edit the Registry and remove the files through the hidden share C$, that are placed there by the Malware.
Sending a sample of the virus to the antivirus vendor so that a signature file can be made to detect and cure the infection. Once a signature is created it can be distributed through the company's Antivirus software already existing signature distribution method.
Building a script that is accessed by the login script could help with removal. It would need to remove the %windir%\system32\mfm directory and the files within it as well as the registry entries created by msrll.exe noted with Regshot.exe. Since the Malware `msrll.exe` does not have a built in network distribution system it would not be necessary to add a dummy file with the same

[15] http://www.dameware.com/

name as the Malware but as an empty read-only file.

Deduced Information

The m220 command set that was found in the mIRC v6.12 code could be used to perform Social Engineering. Commands like "`?echo`" could be used to talk on the IRC and get real users to accept DCC connections and thus do file transfer to move the Malware to a new machine.
If the use of "`!pass $1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0`" had worked and control of the bots logged into the IRC session was obtained, the next steps would be to interact with the Malware and then remove the threat on the infected machines. Using embedded commands like `?rmdir` to remove the `C:\%windows%\System32\mfm` directory and then running `?die` or `?kill` to stop the `msrll.exe` process.

References

GIAC login, 14 Dec 2004
http://giactc.giac.org/cgi-bin/momgate

WinZip, 14 Dec 2004
www.winzip.com

Sysinternals Analysis tools, 14 Dec 2004
www.sysinternals.com

SNORT, 14 Dec 2004
www.snort.org

Google, 14 Dec 2004
www.google.com

Hybrid 7 IRC, 14 Dec 2004
http://irc.carnet.hr/docs/hybrid7docs

SANS Reverse-Engineering Malware: Tools and Techniques, Hands-On, 2004 Pg 2-41.

SANS Reverse-Engineering Malware: Controlling the Trojan, 2004 Pg 54-56.

Aaron's Homepage, 14 Dec 2004
www.exetools.com/unpackers.htm

mIRC version 6.12, 14 Dec 2004
www.mirc.com/index.html

A parked domain name listed in the Malware msrll.exe strings, 14 Dec 2004
http://collective7.zxy0.com

DameWare Development
http://www.dameware.com/

All About Bots. Trojans And Worms! – section 4.d.
http://swatit.org/bots/