Global Information Assurance Certification Paper

GREM Assignment Version 1.0

October 2004

SANS NS2004

Title : GREMlins Are you taking the mIRC

Author: Adrian Hammill

• Table of Contents

1 Introduction\Abstract

1.1 This is a submission for the GREM certificate, for Reverse Engineering of Malware. The Malware specimen used for this practical was version 1.0 msrll.zip. The GREM certificate is used to demonstrates an understanding of the procedures and methods that are needed to be followed to Reverse Engineer a piece of Malicious code. These are having a secure labaratory environment, evaluating Behavioural Analysis and conducting Code Analysis.

2 Laboratory Environment

2.1 The following section describes the laboratory environment and the precautions that I have taken to ensure that no networks other than those which were intended as part of the Lab environment were infected. The whole of this practical will be carried out on one Laptop.

2.2 Laptop Specification:

Make: Sony Vaio Laptop VGN-S1HP,
RAM: 512MB
RAM Hard Disk: 40GB
CPU: Intel Pentium M715 1.5Ghz
Graphics: ATI Mobility Radeon 9200 1280x800
Ports: Ethernet Port, W LAN, Internal Modem, Bluetooth. 2 x USB, Firewire
Peripherals: DVD/CD-RW USB-Floppy Drive

2.3 Software Installed:

Host Name: 'Shaggy'
Operating system: Redhat 9.0 (shrike) Kernel 2.4.20-31.9
Software: VMware Workstation 4.5.2 Build-8848

2.4 VMware environment:

The VMware environment has a network of four machines, their configurations are listed below. The VMware environment has been tailored to suit a Windows executable following the work carried out in section 3 in looking at the properties of the malware specimen. Had the malware specimen been a linux executable the configuration would have been less Windows orientated.

2.4.1 Machine 1: Name 'grem'

Purpose: Network Traffic Analysis and Response
Operating System: Redhat 9.0 (Shrike) kernel 2.4.20-8 text only install.[1]
Software: Snort, IRC
Ram 64MB Virtual Hard Disk size 2GB
Network: Host Only Network Vmnet 1
IP Address: 192.168.226.134
Extra Information: Running IP Tables.

1 As provided on the GREM course CD with local name and IP Address changes implemented.

2.4.2 Machine 2: Behavioural Analysis Machine
Name: 'Scrappy'
Operating System: Windows XP SP2
Software: SYSinternal Utilities TDImon, FileMon, RegMon, Procxp, Regshot.
mIRC V6.12
Ram: 76MB Hard Disk: 2GB
Network: Host only Network Vmnet 1
IP Address: 192.168.226.131.
Extra Information: Firewall was disabled.

2.4.3 Machine 3: Code Analysis Machine.
Name: 'mutley'
Operating System: Windows XP SP2
Software: IDA Pro, Ollydebug, BinText, aspackdie1.41, PE Module Explorer
Ram: 128MB
Hard Disk: 3GB
Network: Host only Network Vmnet 1
IP Address vmnet 192.168.226.136.
Extra Information: Firewall was disabled.

2.1.1 Machine 4: Alternative platform for infection
Name: 'Dastardly'
Operating System: Windows 2000 Professional
Software: Various SYSinternal Utilities TDImon, FileMon, RegMon, Regshot.
Ram: 64MB
Hard Disk: 2GB
Network: Host only Network Vmnet 1
IP Address vmnet 192.168.226.133.

Tools Description.
TDImon - TDImon is a useful utility which monitors TCP and UDP activity on your local system.
RegMon - RegMon is a utility which monitors for any changes made to the Registry.
FileMon – FileMon is a utility which monitors for any changes to files, whether that be creation, deletion or modification.
Regshot – A utility which takes two snapshots of a filesystem and provides a differential output file detailing changes which occured to the system in the time between the between the two snapshots.
SNORT – an IDS used in this instance for packet sniffing.
PE Module Explorer – A utility to explore the sections of a Portable Executable, breaking the file down into its DOS,COFF and data sections.
BinText – A tool for stripping and displaying the printable text strings from a file.
IDA Pro – An Interactive Disassembler for investigating compiled programs.
OllyDebug - 32 bit Assembler Level Debugger used to set trace points and breakpoints in an executable,enabling the stopping of the program at any location.
mIRC V6.12 – Real IRC client for behaviour comparison.
Aspackdie – Unpacker for programs obfuscated with Aspack.

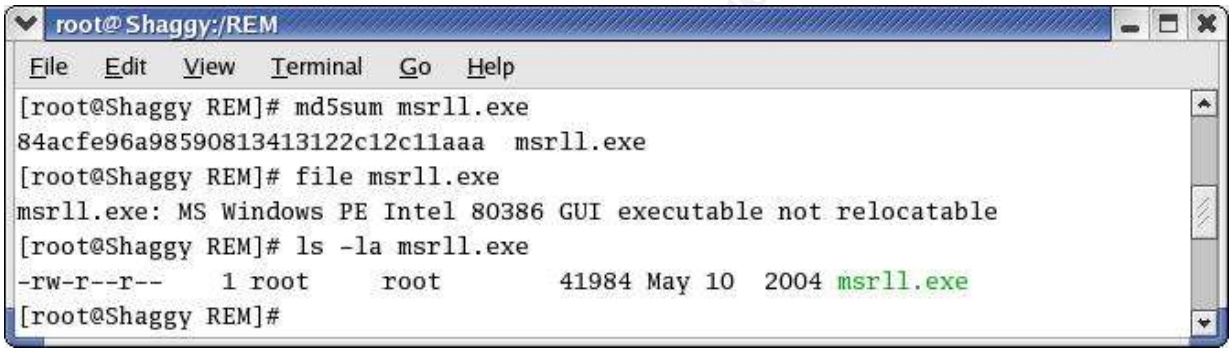
Precautions

The following steps were taken as precautionary measures:

- No Ethernet cables were ever connected to the Laptop a dummy plug was placed into the Ethernet port to prevent any connections being made by accident.
- No Modem cables were ever connected to the Laptop as for the Ethernet port a dummy plug was placed into the port to prevent any accidental connections being made.
- Wireless LAN drivers were not loaded into any of the Linux environments to prevent any inadvertent network connections occurring.
- IP Tables were configured on the Host operating system as a precautionary measure should the malware attempt to escape its sandbox environment.

3 Properties of Malware Specimen

3.1 The following section describes the file properties which relate to the malware specimen. I require these details to enable the tailoring of my laboratory environment to this particular piece of malware. I first needed to establish whether or not it is a Windows or Linux based attack. I used the Linux host operating system of my laptop which as I have described before is separate from all networks. I downloaded the msrll.zip file on to the Linux operating system, using the password provided I unzipped the malware and copied into a directory called REM. I then used the Linux commands *file, MD5Sum* and *ls -la*, to determine as much as I could about the executable before carrying out behavioural or code analysis. The screen dump of the output of these three commands is shown in Illustration 1.



- *Illustration 1Display of Linux commands 'md5sum, file & ls -la'*

3.2 The *file* command makes an educated guess based on file content and headers as to what type of file it is looking at, whether it is a text file or binary executable, if it is binary which format windows or elf.

3.3 The md5sum command calculates the MD5 message digest of a file which we can use later for comparison purposes to establish identical files.

3.4 I used *ls -la* to list the file displaying its file size and date last modified. What the output shows is that the file is a windows portable executable, which was last modified on May 10th 2004 and has a file size of 41984 bytes.

Its MD5 hash is 84acfe96a98590813413122c12c11aaa.

3.5 I then went on to extract strings of text from the now unzipped executable using the 'strings' command. Using 'strings -a' on the command line prints to the screen strings of printable characters from a file. A list of the printable characters which were of interest are printed below.

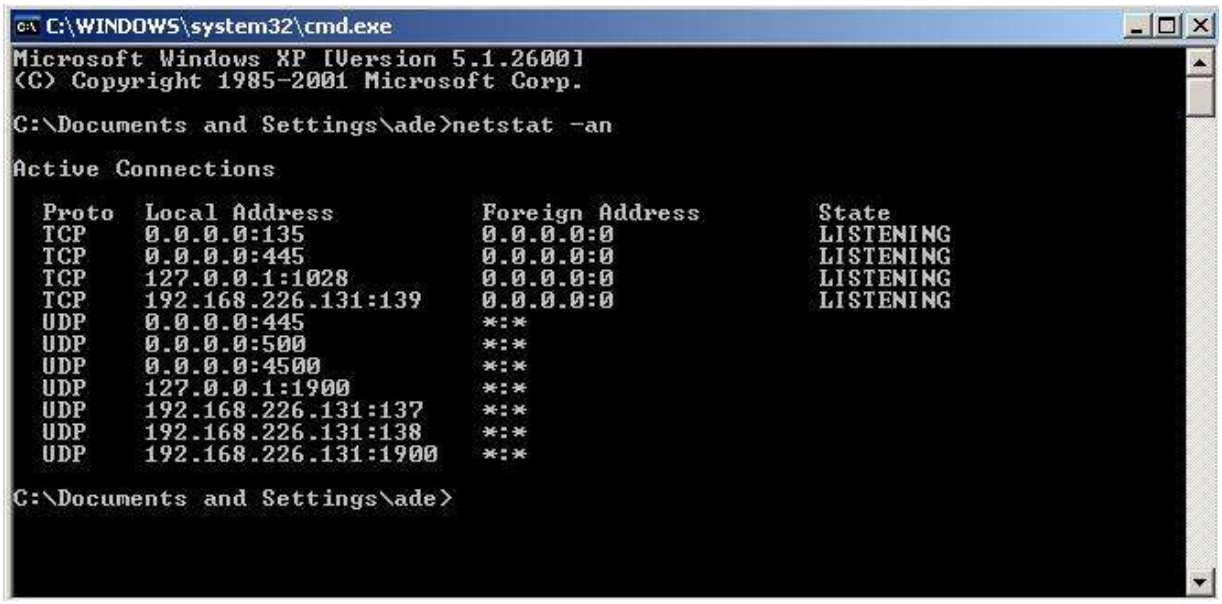
```
!This program cannot be run in DOS mode.
.text
.data
.idata
.aspack
.adatakernel32.dll
GetProcAddress
GetModuleHandleA
LoadLibraryA
advapi32.dll
msvcrt.dll
shell32.dll
user32.dll
version.dll
wininet.dll
ws2_32.dll
AdjustTokenPrivileges
```

3.6 The strings of text above confirm that the program is in fact a Windows based executable, this can be drawn from the 'This program cannot be run in DOS mode', along with all of the references to windows Dlls. The majority of the strings output were complete garbage and this is indicative of a packer being used to obfuscate the malicious code, obfuscation is used to prevent Anti Virus products detecting it. There are various packers available, during the code analysis section I will try to identify which one has been used.

4 Behavioural Analysis

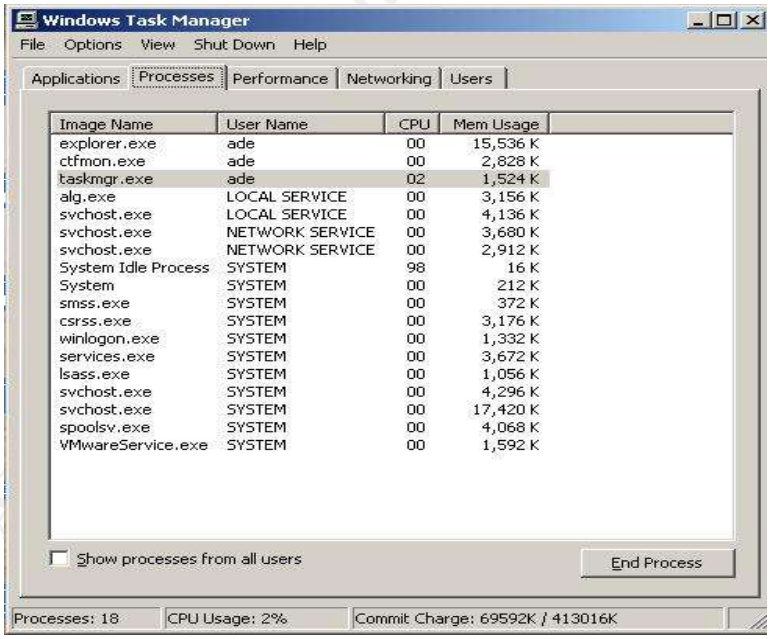
4.1 In this section I will attempt to describe the procedures I took in analysing the behavioural processes of the malware.

4.2 To begin with I created a baseline of the VMware Machine 2 'Scrappy' so that I could determine the changes which had taken place after running the executable. The tools I used to create a baseline were common applications found on any Windows XP installation. Illustration 2 below shows the command netstat -an being run from a command prompt and displaying all ports currently open and their current state. The arguments '-an' were used as they display Protocol, local and remote IP Address and port state.



- *Illustration 2Default Ports open on Windows XP*

4.3 I then started up Windows Task Manager to identify which applications and processes were already running and to get a feel for the level of Network and CPU activity. I also took a screen dump of the processes tab in Task Manager so that I had a reference point prior to execution of the malware, this is shown in Illustration 3.

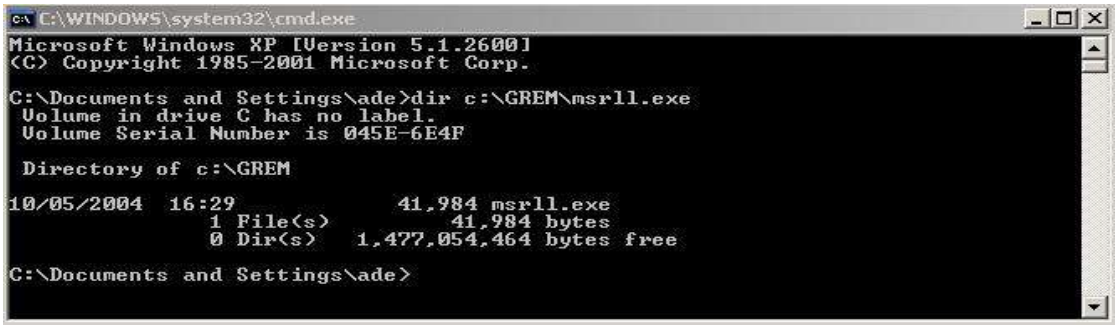


- *Illustration 3Default Processes running in Task Manager*

4.4 Now that I had a system baseline I copied the Zipped msrll file to be used for the exercise onto the Windows XP machine Scrappy. I copied it zipped up so that it wasn't inadvertently executed during the copying process. I unzipped the msrll.zip using Winzip 9.0 and the appropriate password into a directory called GREM. The output from msrll.zip was as before msrll.exe, I then created a shortcut on the Desktop to this executable. I used WinMD5 to create an MD5 hash of the newly created file 'msrll.exe' to verify that it was an identical file to the one previously unzipped. A screen dump of the output of WinMD5 is given in Illustration

4. WinMD5 is a free windows tool downloaded from the Internet which can calculate the MD5 hash sum of any file. Illustration 5 shows the file size and last modified date of the unzipped malware msrll.exe. As you can see they match exactly the details obtained in Illustration 1.

- *Illustration 4WinMD5 output of Msrll.exe*



- *Illustration 5Directory listing of Msrll.exe*

4.5 Now that I had a system prepared I used the Snapshot facility of VMware so that I had a clean image to return to without having to rebuild from scratch. This is a useful function of VMware which can save an Analyst time and aggravation, it can restore your environment known working point within seconds. One thing to be aware of is that once you revert back to a previous version any changes will be lost, although this may sound obvious it is very simple to revert and then realise you have lost any logs which you may have created. I recommend that you save any logs off platform (e.g. On a shared folder with the host operating system, floppy disk, or USB memory stick).

4.6 I started the SYSinternal applications TDImon, RegMon and FileMon pausing their capture functions using Ctrl-E and clearing the displays on all three applications. I then used a fourth utility called Regshot to take a snapshot of the installation before the malware was executed. The snapshot taken with Regshot should not in any way be confused with that taken by VMware, Regshot's snapshot takes a time slice look at all of the files and registry settings on a system so that a second time slice can be used for comparison at a later point in time. I started the capture processes of the TDImon, RegMon and FileMon applications and then executed the malware from the shortcut previously created on the Desktop.

4.7 The malware was allowed to run for approximately 40 seconds before being terminated using Task Manager. The capture process was then stopped on all TDImon, RegMon and FileMon. Regshot was then used to take the second snapshot of the installation. The comparison function of Regshot was then used to compare the before and after images. The output files were then saved to a directory called “logs” on shaggy the Linux host operating system under the name of 'comparison_result131104.txt'. The captures from the SYSinternal tools were also saved under similarly obvious names (tdimonoutput131101.txt, regmonoutput131103.txt & filemonoutput131104.txt) in the same location. I use application name and dates in my file names so that I have a reference point of when the tests were run, if

any test is run several times on the same day a further number is added to the string to indicate chronological order.

5 Log Analysis

5.1 RegShot -The next step in behavioural analysis is to read through the log files that have been created. The first log to be examined was the comparison_result131104.txt file. This file was generated by Regshot it is a differential file based on the comparison of the two snapshots. The format of the text output of this file is quite helpful in that it groups together various events making it easy to understand. In total Regshot identified that 59 changes had occurred since the initial baseline had been taken. Of those 59 changes that had taken place the most relevant were as follows.

5.1.1 In no particular order, 'msrll.exe' had;

5.1.1.1 Created an 'mfm' directory under c:\windows\system32 in which it had placed a copy of itself and a new file called jtram.conf.

5.1.1.2 Deleted the original copy of itself from c:\grem

5.1.1.3 Created a service called 'Rll enhanced drive' which was to run automatically with Local System privileges, the underlying executable for this service was c:\windows\systems32\mfm\msrll.exe

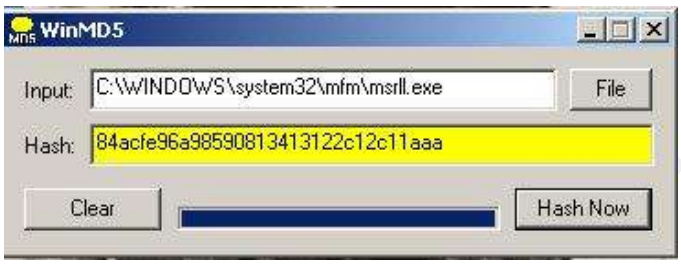
5.1.1.4 Created an entry in the Registry called 'seed' under HkeyLocalMachine\Software\Microsoft\Cryptography\Rng.

5.2 TDImon - The next log to be analysed was that provided by TDImon, as I have written earlier TDImon monitors the TCP and UDP activity on the local system. Looking through the output provided by TDImon the items of interest worth noting were that msrll.exe had opened two ports to listen on these were TCP 113 and TCP 2200. TCP 113 is (according to RFC 1413) the port assigned to the Identification Protocol. The Identification Protocol is also known as the “ident” protocol, it is used to provide a means of identifying the user of a particular TCP connection. TCP 2200 is in the unallocated range of port numbers and is therefore specifically associated with this malware. The only other activity identified by TDImon was traffic which appeared to be DNS name resolution queries, which were being targeted at the default gateway. As there is no DNS server on this network at present these requests were torn down. When scrappy was next brought up I ran a 'netstat -an' command which listed ports 113 and 2200 as listening.

5.3 RegMon - identified that registry changes previously seen by Regshot, with the creation of the new service 'Rll enhanced drive' being visible and the seed value being changed under the HkeyLocalMachine\Software\Microsoft\Cryptography\RNG. RegMon also identified that msrll.exe queried the majority of settings and configuration information for the machine, it checked version numbers, security settings, IP configuration and many other items.

5.4 FileMon - Reading through the FileMon output confirmed the information which had been previously been observed in the RegShot comparison exercise. The creation of the mfm directory under the system32 directory, the copying into the directory of the msrll.exe file and the creation of the new jtram.conf file. It also corroborated the deleting of the original malware specimen.

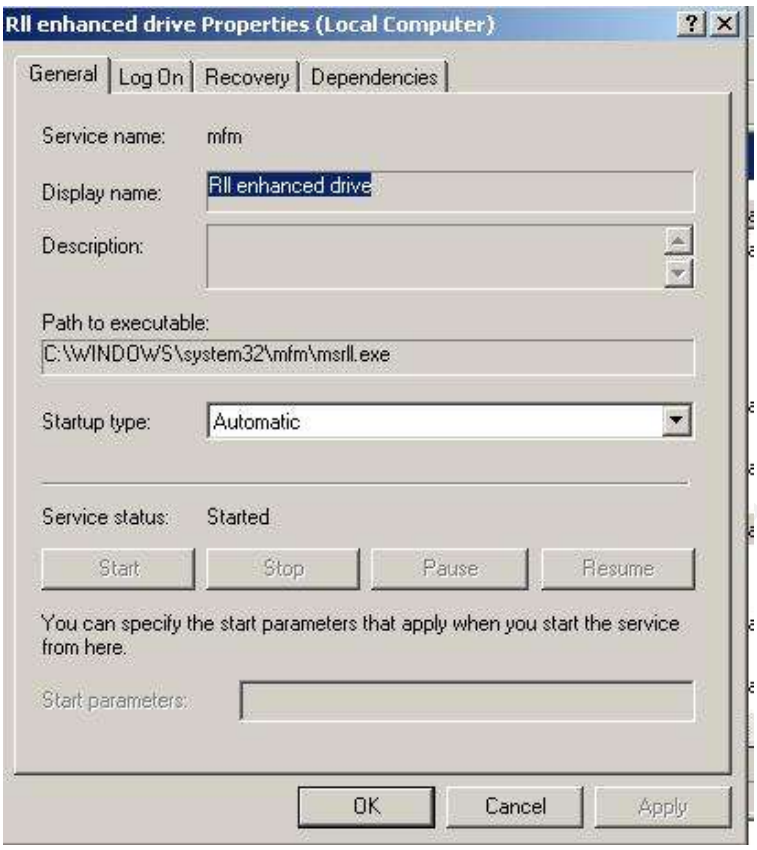
5.5 I then looked into the newly created 'mfm' directory and found the two files which Regshot had identified as being created, I carried out a WinMD5 on both of these files. By comparing the screen dump of the WinMD5 hash of the msrll.exe in the mfm directory shown in Illustration 6 with that of the msrll.exe that had existed in the GREM directory shown in Illustration 4 it is possible to verify that they are identical files because their MD5 digests are the same. I then turned to the jtram.conf file and attempted to open it using notepad, the file appeared to be encrypted.



- *Illustration 6WinMD5 of msrll.exe under mfm directory*

5.6 All log files were saved away and 'scrappy' was rebooted, by looking at the process list through Task Manager it was possible to see that the msrll.exe was running, by navigating to the services application under Admin Tool (from within Control panel) it was possible to see the newly created service ' Rll enhanced drive' while clicking on the properties of the service I was able to confirm that the service was indeed running under the local system account and unusually for a service the start, stop and pause function had been disabled. I was also able to determine that the service is actually called 'mfm' and that it is its "display name" that is 'Rll enhanced drive'. A screen dump of the services applet is shown in Illustration 7.

5.7 I closed down the infected machine and brought up my linux network monitor 'grem' and started Snort running. Snort can be used to capture network traffic by setting the network card into promiscuous mode and sniffing all the Ethernet packets off the LAN. Using the command "snort -vd | tee > /tmp/'date_01.txt' " I started packet sniffing. I redirected the output into a text file for filtering and analysis. Once I was sure that snort was running I started the Infected Windows XP machine 'scrappy'. Once 'scrappy' was up and running I logged in and brought up task manager to ensure that the malicious code was indeed running. Once I had observed that it was I returned to 'grem' to watch the packet capture.



- *Illustration 7 Rll Enhanced Drive Service properties*

5.8 The first thing that I observed whilst reviewing the packet captures was a DNS resolution request for collective7.zxy0.com . With no DNS server present on the network it was unlikely to find the server, so an entry was entered into the Windows XP hosts file (c:\winnt\system32\drivers\etc\hosts) pointing the name collective7.zxy0.com to the ip address of 192.168.226.134 (grem). I restarted Snort on grem and monitored the traffic a second time, this time scrappy had used its internal host file to resolve the IP Address of 'Collective7.zxy0.com' and was trying to initiate a connection to Port 6667 on 192.168.226.134. By looking through the port assignment list issued at www.iana.org by The Internet Assigned Numbers Authority for Port 6667 was identified for Internet Relay Chat, and so it was looking likely that we would need to introduce an IRC server onto the network to further monitor the activity of the malicious code.

5.9 Before introducing an IRC server onto the network I continued to look through the initial capture. There were several other connection attempts being initiated from the infected host. Two of the connections being attempted were to the same address of Collective7.zxy0.com but to the alternative ports of 9999 and 8080. These ports are not commonly associated with IRC channels, the IANA lists port 8080 as being associated as an alternate port for HTTP traffic, and port 9999 as being 'distinct' (although I did find several IRC channels which were using port 9999.

5.10 Other connection attempts worth noting were initiated to 239.255.255.250 port 1900, A quick search on Google highlighted that Port 1900 is commonly associated with SSDP, SSDP is Simple Service Discovery Protocol, windows messenger uses SSDP to attempt to locate upstream Internet gateways on UDP 1900. An article at http://support.microsoft.com/default.sapx?scid=kb;en-us;317843 describes the service in detail. This is normal activity on a windows client and so has been discounted as being

relevant, this activity reinforces to the analyst the need for a known base and activity line so as not to be distracted by activity which you believe to be irregular which is in fact normal behaviour.

5.11 Before starting an IRC server I placed a NetCat listener on all three ports which had been identified (6667,8080,9999) on the Grem server and directed the output to three text files. I then started the malware running on the infected machine and observed the three ports. The screen capture of the output of these text files is shown in Illustration8. As you can see the same type of connection was being attempted to all three ports with only the details of the connection being different.

```
[root@grem /]# more 8080.txt
USER Uqlnt localhost 0 :XoOubFI
NICK YCYoMirixnvb
[root@grem /]# more 9999.txt
USER taVBy localhost 0 :fAMflFKHAxQaIDScXVbqXFwpg
NICK TGZNByOjh
[root@grem /]# more 6667.txt
USER YULOmCUuzmv localhost 0 :tKeMHbcxzndxiERufcHMqmDGucGWtWWJBiwSDES
NICK pEgVtcPffa
[root@grem /]# _
```

- *Illustration8 Capture of 3 Netcat Listeners on Ports 6667, 8080 & 9999*

5.12 I then initialised an IRC server on 'grem' to try to continue the IRC communication that was being initiated. The IRC daemon was started under the ircd account and a new account 'ade' created locally was used to join to the IRC. I used an account other than root in case the malware would fail to run should root be present. With the IRC server running and Snort capturing the traffic again, traffic analysis showed in Illustration 9 that the irc channel that the malware was trying to connect to was #mils.

```
04/17-16:18:40.338023 192.168.226.192:1026 -> 192.168.226.134:6667
TCP TTL:128 TOS:0x0 ID:82 IpLen:20 DgmLen:53 DF
***AP*** Seq: 0x617723B1  Ack: 0x99F57B2B  Win: 0xFAF0  TcpLen: 20
4A 4F 49 4E 20 23 6D 69 6C 73 20 3A 0A          JOIN #mils :.
```

- *Illustration 9Malware joining channel #mils*

5.13 The local account 'ade' joined the channel #mils and monitored the logins, 'scrappy' was started and the msrll.exe connected to the #mils channel with the account name 'FmnoTaUap' on this occasion, each time the process is restarted a new name is selected to join the channel with. Attempts were made to communicate with the malware on the IRC channel #mils but I was not able to gain any response or reaction from it.

5.14 Whilst reading through the Snort output file and working through the log on process of the malware it was possible to trace the connection taking place, the one thing that stood out was that immediately after joining the #mils channel the malware requested a 'who' lookup on the channel, the irc channel responded by informing the irc client exactly who was logged on, at this point the malware seemed to go in to a sleep mode. This leads me to believe that the malware client is looking for a particular user to be logged on to the channel. All that was visible from then on in was the PING \ PONG between the server and client which occurred approximately every 3 minutes where the server was ensuring that the client was still alive. As an operator on the channel I kicked the malware client off the channel in the hope of provoking some reaction, all that occurred was that the client immediately rejoined. I also reset the client and server letting the malware join the channel first to see if that made the malware behave any

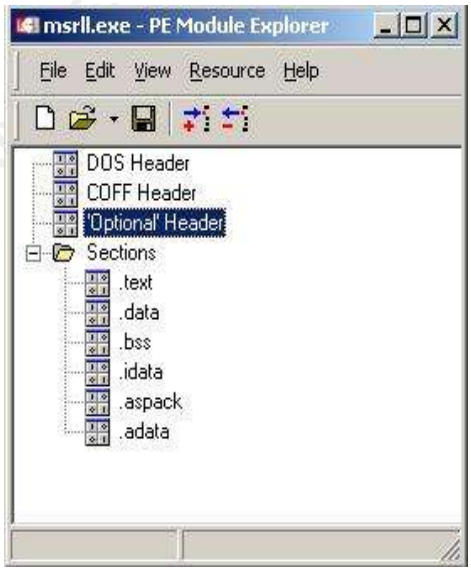
differently, once again this had no effect.

5.15 It is also worth noting at this point that the malware is only effective if the account that it is executed under is a privileged account, the series of tests conducted above were repeated using a normal limited user account and the malware was unable to successfully install itself onto the system, or open any network ports. As soon as the user logged off, the process which had been unsuccessful simply terminated itself.

6 Code Analysis

6.1 Having been unable to obtain any reaction from the malware and without any more information to work on I headed for Code Analysis to see if I could gather any more information to aid the reverse engineering of the malware. I had already prepared a machine for the job, It was a Windows XP machine called 'mutley'. I had preloaded 'mutley' with an evaluation version of IDA pro, Ollydebug, BinText and PE Module Explorer.

6.2 While looking at the properties of the malware I had discovered that it had been packed with an unknown packer for obfuscation purposes. Therefore one of the first things I needed to do was see if I could ascertain which packer it was. When looking through the strings earlier I had noticed that .aspack was a visible character pattern. Aspack is a freely available packer which could have been used to pack this executable, however it could also be specifically placed tect to draw the analyst away from the true packer. I needed more details and with that in mind I loaded the malware specimen into PE Module Explorer to look at the sections within the portable executable. Illustration 10 is a screen dump of what was found. As you can see one of the sections is called .aspack, adding this with the .aspack string from earlier made me believe that Aspack had been used for obfuscation. I then needed to work out how to unpack it, unlike a lot of packers Aspack is unable to pack itself, using google I found several versions of a tool called Aspackdie, The latest version of Aspackdie was version 1.41.

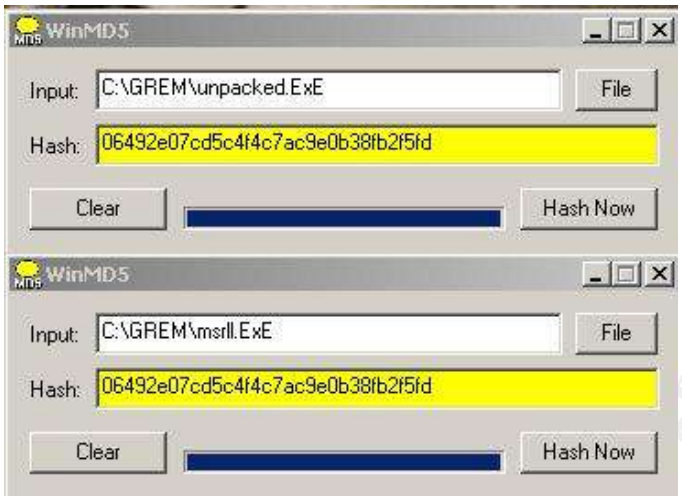


- *Illustration 10PE Module Explorer*

6.3

6.4 I installed Aspackdie1.41 on to Mutley and then once again unzipped the msrll.exe from its zip wrapper. I verified that I was still working with exactly the same file by using WINMD5 to calculate the MD5 message digest, which I then compared to the MD5 hashes previously produced. I opened Aspackdie and navigated to the unzipped msrll.exe. I clicked on ok and received a message box confirming that the msrll.exe had successfully been unpacked and named unpacked.exe. In preparation for future execution of the code I renamed the original

msrll.exe to msrll.exe.old and renamed unpacked.exe to msrll.exe. I used WinMD5 to calculate the MD5 message digest of the new msrll.exe. It is shown in Illustration 11.



- *Illustration 11Comparison of Unpacked.exe and msrll.exe*

6.5 Strings Analysis -Now that I had an unpacked version of the malware I decided to run it through strings again to see if any more useful printable strings were visible. As I was now on a windows platform I used the BinText utility to carry out the strings test. Appendix B has the full 10 page print out of printable strings from BinText, I have included the ones which caught my attention below.

6.5.1 Smurf is an ICMP DoS attack which takes advantage of directed broadcasts to flood a network with ICMP traffic,and Jolt is an IP fragmentation DoS attack that is directed at Windows NT4 and 2000. Are the strings that we are seeing commands ready to be called by the malware?

```
00002763   00402763      0   ?smurf
0000276A   0040276A      0   ?jolt
```

6.5.2 We have already established that the malware is a modified IRC client, the string below leads me to believe that it is based on mIRC v6.12 by Khaled Mardam-Bey.

```
000074C9   004074C9      0   mIRC v6.12 Khaled Mardam-Bey
```

6.5.3 The following text confirms the name of the servers which I had identified during behavioural analysis as being collective7.zxy0.com. It also explains why when the malware fails to receive a response at one port it moves on to another.

```
000BD80   0040BD80      0   collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy0.co m:8080
```

6.5.4 I found the following date mixed within the strings and it occurred to me that it could be an activation date, I reset all of the VMware sessions and altered the date and time in all environments to be Mar 16 11:45 2004 and reran all of the tests I had tried to see if it would be relevant in waking up the client. Unfortunately this was once again another dead end.

```
00013390   00413390      0   m220 1.0 #2730 Mar 16 11:47:38 2004
```

6.5.5 The next sequence of strings all appeared to be commands that I was expecting the irc channel to respond to, I tried each and every one of them in various formats e.g. with a / or a ! or with nothing in front of them.

```
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
000093A9   004093A9      0   ?akick
000093B0   004093B0      0   ?part
000093B6   004093B6      0   ?dump
000093C6   004093C6      0   ?md5p
000093CC   004093CC      0   ?free
000093D7   004093D7      0   ?update
000093DF   004093DF      0   ?hostname
000093EE   004093EE      0   ?!fif
000093FE   004093FE      0   ?play
00009404   00409404      0   ?copy
0000940A   0040940A      0   ?move
00009415   00409415      0   ?sums
00009423   00409423      0   ?rmdir
0000942A   0040942A      0   ?mkdir
00009436   00409436      0   ?exec
00009440   00409440      0   ?kill
00009446   00409446      0   ?killall
0000944F   0040944F      0   ?crash
0000946E   0040946E      0   ?sklist
00009476   00409476      0   ?unset
0000947D   0040947D      0   ?uattr
00009484   00409484      0   ?dccsk
00009490   00409490      0   ?killsk
```

6.5.6 The last set of strings which were of interest given the time constraints imposed were those relating to DCC, DCC is a Direct client to client protocol. It allows you to send and receive files privately and securely over IRC. From the strings below it would appear that the malware is expecting to use DCC to perform some file transfers. It would also appear that for the DCC connection a password is required with the 'dcc.pass and % bad pass from' strings being visible.

```
00008C19   00408C19      0   DCC RESUME %s %s %u
00008B99   00408B99      0   DCC ACCEPT %s %s %s
00008BAE   00408BAE      0   dcc_resume: cant find port %s
00008BD1   00408BD1      0   dcc.dir
00008BFD   00408BFD      0   resuming dcc from %s to %s
00008C19   00408C19      0   DCC RESUME %s %s %u
0000BB40   0040BB40      0   dcc.pass
0000BB49   0040BB49      0   bot.port
0000BB52   0040BB52      0   %s bad pass from "%s"@%s
```

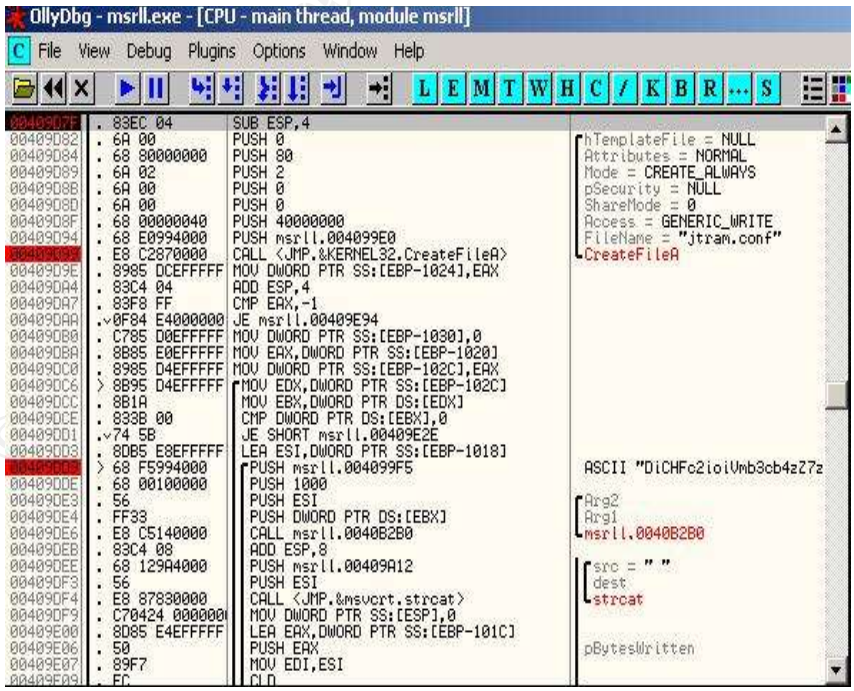
6.6 Further Analysis – I described earlier that one of the actions carried out by the malware is to create an encrypted file called JTRAM.CONF it is located in the 'system32\mfm' directory with

msrll.exe. I opened the file with Notepad to see a file which had several lines of what I believed to be encrypted text. I decided to try and find the encryption routine to see if I could find out the true content of the file. A group of strings which caught my attention were

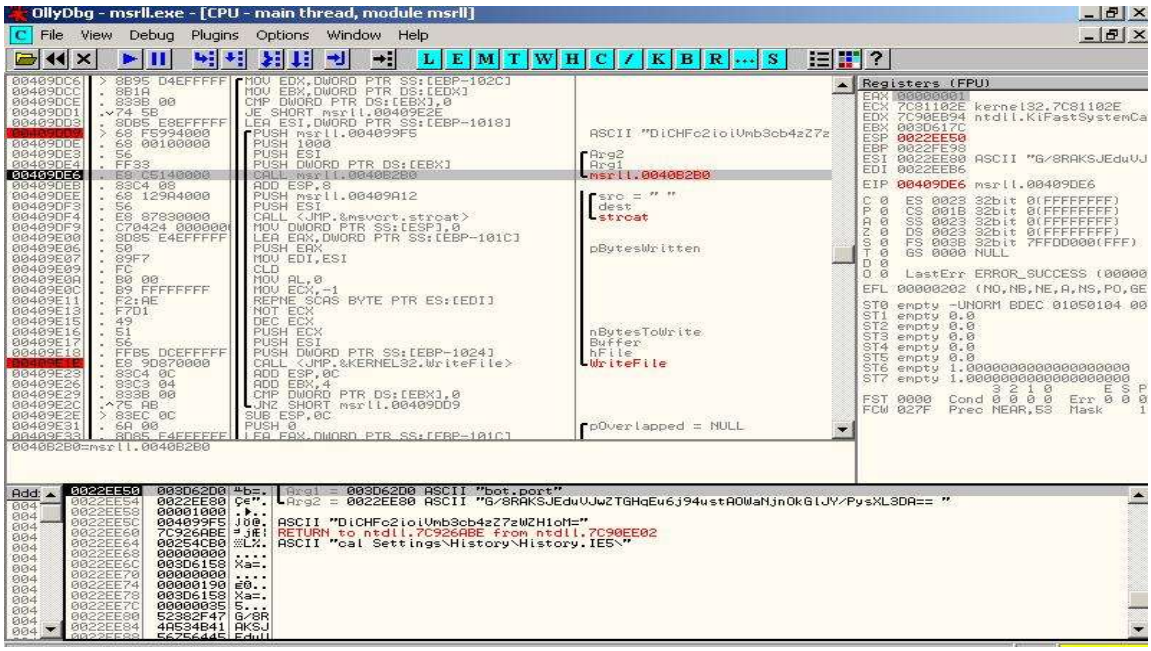
```
000099E0   004099E0     0   jtram.conf
000099EB   004099EB     0   jtr.*
000099F5   004099F5     0   DiCHFc2ioiVmb3cb4zZ7zWZH1oM=
00009A16   00409A16     0   conf_dump: wrote %u lines
```

6.7 What was interesting about these lines for me was that they had the filename that I was interested in, an indication of lines being written (wrote %u lines) and a string of characters which looked suspiciously like an encryption key. I located the code in IDA Pro using the find function and found the initial process for creating the jtram.conf file at 409D7F for in the next few line calls were made for CreatingfileA and the attributes of the file are listed e.g. Generic file, FileName. Using these location details as a starting point I set a breakpoint at 409D7F to try and capture the routine before it started. My first instinct was to remove what I believed to be an Encryption string therefore I used the edit function of Ollydebug to replace DiCHFc2ioiVmb3cb4zZ7zWZH1oM= with zeros and ticked the 'keep size' box. I deleted the jtram.conf file and then executed the program and waited for it to create a new conf file. I opened the new jtram.conf file with notepad to find a single line of repeated clear text of "collective7.Zxy0.com", I also observed that the file was no longer 2kb in size it was 1kb. From this I deduced that I had found the correct place of encryption but that by changing it to zeros I had shortened the process of encrypting information.

6.8 My next step in trying to decrypt the jtram.conf file was to interrupt the execution of the encryption routine using a breakpoint, step through using F8 and then looking at the contents of the stack window in the CPU pane to look at the parameters which were passed to it. Illustration 12 Is a screen captures of the breakpoints used 409D7F, 409D99 and 409DD9 to capture the jtram.conf creation and view the lines as they are encrypted. Illustration 13 Demonstrates bot.port being encrypted.



- *Illustration 12Ollydebug Decryption Breakpoints*



- *Illustration 13Ollydebug showing encryption of bot.port*

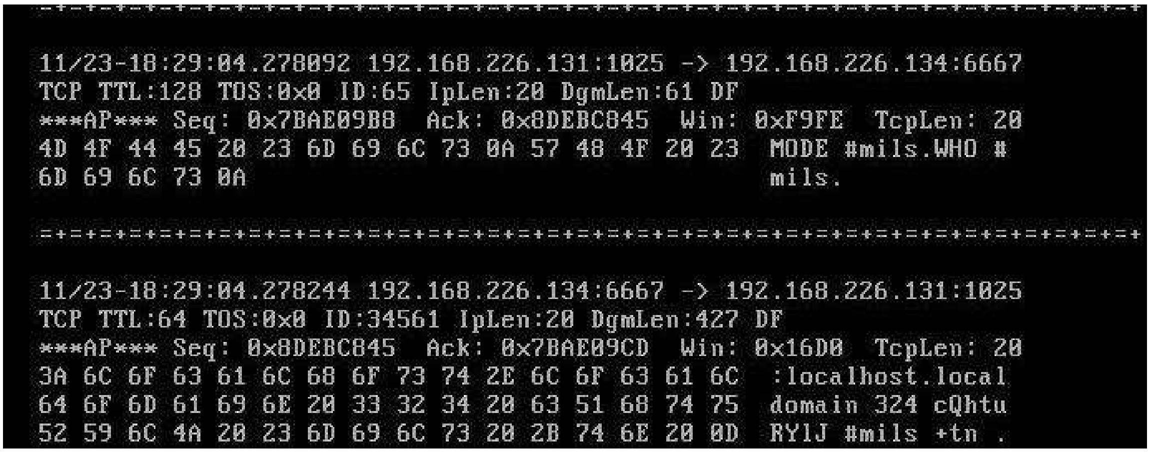
6.9 By continuing to use F8 to step through the jtram.conf encryption process I was able to recover each of the lines of the file. It appears that this is a configuration file for the malware lkisting the server to connect to and which bot port to have open. It also lists which channel to connect to. There are two interesting strings beginnning with the same characters 1KZLPLKDf$, I wondered whether these were the pass keys to the #mils channel or if they referenced the account to log on to the #mils channel with. I attempted connections using all permutations but was unsuccesssful.

6.10 The following strings are the decrpted text from jtram.conf.

Collective7.zxy0.com
bot.port
2200
irc.quit
servers
collective7.zxy0.com,collective7.zxy0.com9999!,collective7.zxy0.com8080!
Irc.chan
#mils
pass
1KZLPLKDf$W8kl8Jr1X8DOHZsmip9qq0
set
dcc.pass
1KZLPLKDf$55isA1ITvamR7bjAdBziX

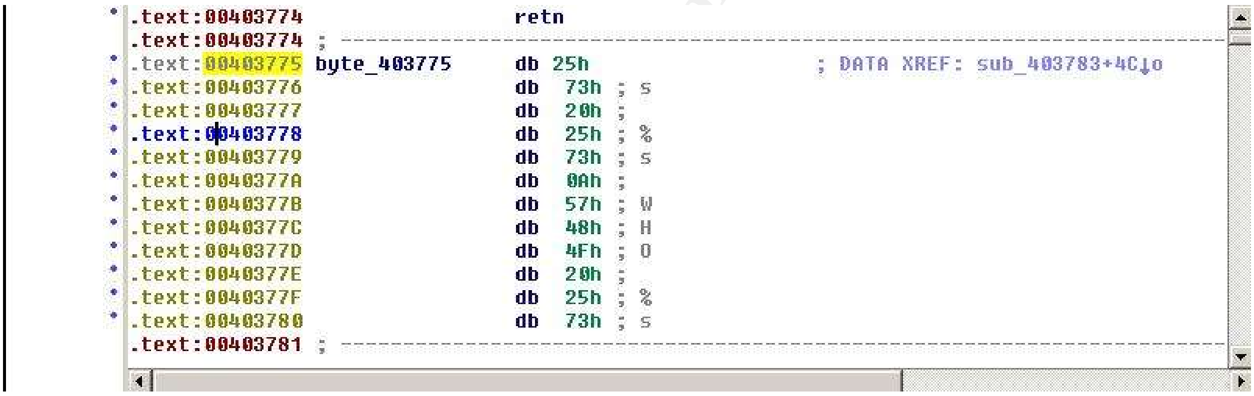
6.11 Moving away from the jtram.conf file I attempted to work out the authentication mechanism for the irc channel. The first thing I attempted was to work out the username for the irc channel. I believed that there was a specific username required for the irc channel as I observed it running a /who command when it first connected to the #mils channel. An example of the / who command being issued is shown in Illustration 14, I was uncertain as to whether this was normal behaviour for an IRC client. As the strings above had identified the mIRC client to be based on mIRC v6.12 I searched the Internet and located another copy the mIRC client. I installed this new client onto 'mutley'. I then started the newly downloaded mIRC client on 'mutley' and connected to the #mils irc channel, whilst at the same time running a snort session on the 'grem' server to capture the network traffic of the connection. The clean install of the

mIRC client did not follow the same pattern as the malware and no '/who' command was issued. This reinforced my belief that the malware was looking for a specific user to be logged onto the channel before it became truly active.



- *Illustration 14 Snort capture of /who command*

6.12 After reviewing the contents of strings for the newly unpacked malware I went on to load it into IDA Pro disassembler and searched for the '/who' string. The intention was to analyse the code around this command, my thought process being that if a '/who' command was being issued then some form of comparison might being made to verify the user names.

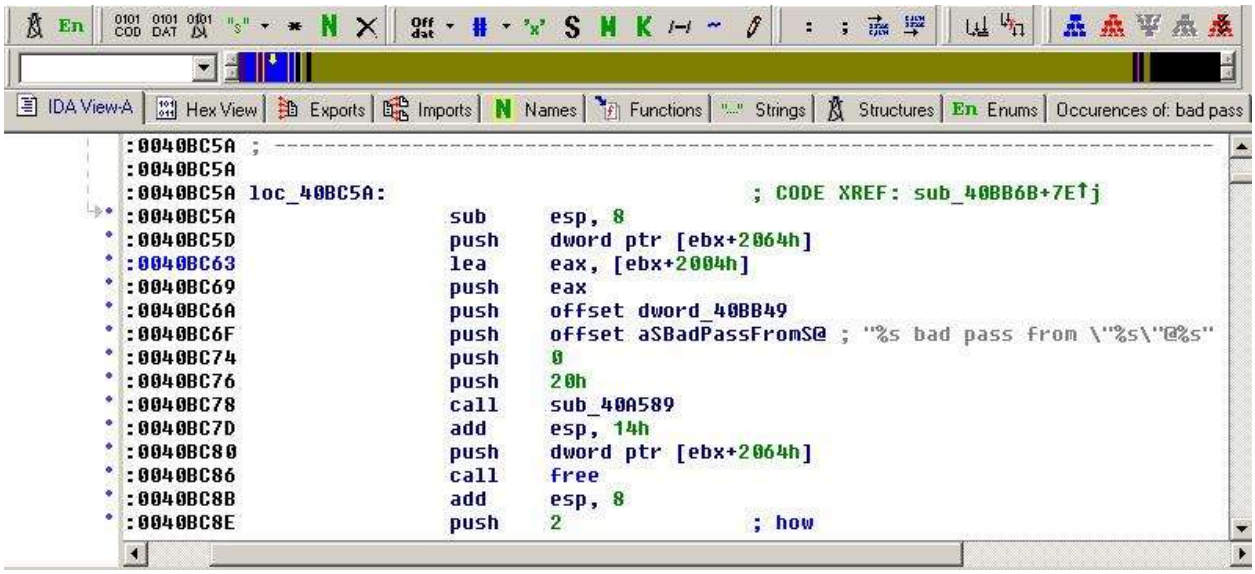


- *Illustration 15 IDA Pro /who command*

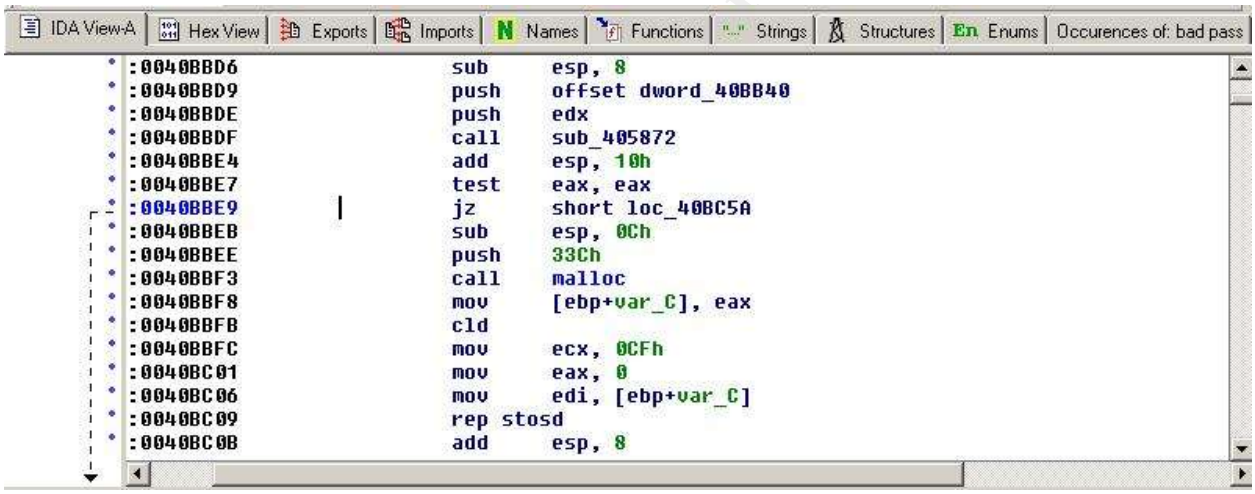
6.13 I used the search utility to find 'who',it was not available through the first text search but when the search was directed at the hex window in IDA pro the text was found. I then correlated the text from the hex screen to the IDA-View-A screen, the location was at 00403775 Illustration 15 however this related just to the text string not to the command calling it.I was able to trace back in the code to where the command was called from this was at 004037CF, I then set breakpoints in Ollydebug at this point so that I could step through the calls which followed to find any comparison routines. I was unable to determine any useful comparisons through this technique.

6.14 As I could not locate the entry points for the string comparison for the 'who' command I turned my attention to the DCC communication. From the BinText output I could see the 'dcc.pass' and the 'bad pass' strings, this was indicative of a log on and password being expected. I searched through the malware in IDA Pro using the find utility to find the location of these two strings. I found the location of bad pass and this is shown below in Illustration 16. From here I traced the steps back to see which sections of code caused the bad pass output to be called. This section is shown in Illustration 17. Although I determined the location of the

DCC threads in the code I was unable to initiate any communication with it, I believe that once you have authenticated with the malware a second authentication is required to use the DCC communication.



- *Illustration 16DCC bad pass*



- *Illustration 17Call to DCC bad pass*

7 Analysis Wrap-Up

7.1 Malware Capabilities – From observing the malware executing and the investigations that have taken place the capabilities of the malware include creating a local system level service, creating and deleting files and XP firewall subversion. I assert that the malware is a multi purpose tool that is capable of being used to establish a Distributed Denial of Service attack using at least three preloaded attack tools (smurf, syn and jolt). The malware also opens a listening port on TCP 2200 which has an as yet undetermined program associated with it. There is the capability to remove and place files onto the client using DCC which allows you the freedom to send and receive files. The malware is persistent in that once it has been kicked of a channel it automatically reconnects, and should you stop the process using task manager it

will re establish itself once the machine has been rebooted.

7.2 The assigment asked the question who would use the program? I can find no valid reason why any normal system administrator would deploy this executable on a network. Therefore the only reasons such an executable would be deployed on a network is to steal corporate\personal information, and to amass an army of machines to conduct a coordinated denial of service attack. Which leaves a script kiddie or hacker as the attacker.

7.3 Defensive measures - From tests that I conducted infection only occurred when a user with Administrative privileges was logged in, by sticking with the rules of common sense and only logging on with Admin privileges when necessary you would mitigate the risk of infection. (Now if we can only convince the administrators to use least priv accounts). I had intended to write that by having a personal firewall turned on that protection would be provided but I thought I had better check before making such a statement. With Windows XP SP2 Firewall enabled and no exemptions allowed msrll.exe continued to function totally uninhibited. The best defence in such a case is to implement Boundary Firewalls blocking connections to IRC channels 6667 and 9999 when working from a corporate environment, this partially prevents an attack from this malware from being effective, however if you allow port 8080 through as an alternative HTTP which the majority of companies do then you are still at risk. Blocking and dropping inbound connections to TCP 113 and 2200 from an external network would prevent the malware from being successful in the first instance. It would also be sensible practice idea to implement a Global Policy through Active Directory that limits which services are able to run automatically and even which executables a user can run thus preventing the infection in the first place.

7.4 To eliminate current infections I would create a script which stops the service from running, distributing the file manually, via SMS or some other software management distribution tool. It would stop and then delete the service and then delete the executable msrll.exe and the directory c:\windows\system32\mfm. An alternative regression path is to use the restore utility built in to Windows XP whereby it is possible to revert to a checkpoint, however this does rely on you knowing when the infection took place. An example of a short removal script is below, to be honest it is brutal and takes no prisoners, it makes no checks to see if a user is logged on or if the machine is infected, those refinements could be carried out by the software management system looking for the installed service. Pskill is a sysinternals utility allowing you to kill running processes. The three other commands are standard files found on a Window XP machine.

```
Pskill msrll.exe
sc delete mfm
rmdir /s /q  c:\windows\system32\mfm
shutdown -r
```

Appendix A RegShot Output File

Comments:
Datetime:2004/11/13 11:46:39 , 2004/11/13 11:48:16
Computer:SCRAPPY , SCRAPPY
Username:ade , ade
REGSHOT LOG 1.61e5

Keys added:4

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\Security

---
Values added:18

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 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 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 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 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"
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU:Fubegphg gb zfeyy.yax: 08 00 00 00 06 00 00 00 F0 95 00 73 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU:P:\TERZ\zfeyy.rkr: 08 00 00 00 06 00 00 00 A0 1E 1D 73 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\GREM\msrll.exe: "msrll"
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\ShellNoRoam\MUICache\C:\WINDOWS\system32\mfm\msrll.exe: "msrll"

Values modified:4

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 90 1D 98 02 B7 45 C3 68 AE 2A 32 2A 4C C2 D4 53 D7 F2 8B 93 71 C2 75 25 3B 78 FD B3 D5 98 92 8C B2 96 42 D9 B3 66 8A A3 DD FD 9A 6A C7 7D B9 3F 13 81 6D 26 A8 51 7D 1F A1 74 5C 09 86 26 98 07 51 9A BA 8F 2A 53 3C 7D 5F 44 B3 98 C7 D0 C4 44
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: FE F8 4A D1 F1 19 84 3F 93 59 82 77 F7 20 18 92 C5 E7 81 C0 58 9A 10 70 B5 30 59 25 71 03 55 FE FE E4 34 ED CC 37 F7 39 2D E9 30 BF 5C 34 0F EC 71 6C 8B D7 33 5B 0E F4 CA 48 A0 8C 73 A0 76 97 DF 0E C4 54 FB 42 62 EB 93 05 99 88 5C 9A 7B AC
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU: 08 00 00 00 3A 00 00 00 F0 6B 74 4C 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU: 08 00 00 00 3C 00 00 00 A0 1E 1D 73 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_HVFPHG: 08 00 00 00 13 00 00 00 20 E8 5E 4C 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_HVFPHG: 08 00 00 00 14 00 00 00 F0 95 00 73 09 E1 C4 01
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections\SavedLegacySettings: 3C 00 00 00 03 00 00 00 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
HKEY_USERS\S-1-5-21-725345543-113007714-839522115-1003\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections\SavedLegacySettings: 3C 00 00 00 04 00 00 00 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Files added:6

C:\WINDOWS\Prefetch\MSRLL.EXE-03966588.pf
C:\WINDOWS\system32\mfm\jtram.conf
C:\WINDOWS\system32\mfm\msrll.exe

C:\WINDOWS\Prefetch\MSRLL.EXE-03966588.pf
C:\WINDOWS\system32\mfm\jtram.conf
C:\WINDOWS\system32\mfm\msrll.exe

Files deleted:1

C:\GREM\msrll.exe

Files [attributes?] modified:20

C:\WINDOWS\Prefetch\MSRLL.EXE-1068ACA9.pf
C:\WINDOWS\Prefetch\TASKMGR.EXE-20256C55.pf
C:\WINDOWS\system32\config\software.LOG
C:\WINDOWS\system32\config\system.LOG
C:\WINDOWS\system32\wbem\Repository\FS\INDEX.MAP
C:\WINDOWS\system32\wbem\Repository\FS\MAPPING.VER
C:\WINDOWS\system32\wbem\Repository\FS\MAPPING1.MAP
C:\WINDOWS\system32\wbem\Repository\FS\OBJECTS.MAP
C:\Documents and Settings\ade\Cookies\index.dat
C:\Documents and Settings\ade\Local Settings\History\History.IE5\index.dat
C:\Documents and Settings\ade\Local Settings\Temporary Internet Files\Content.IE5\index.dat
C:\Documents and Settings\ade\NTUSER.DAT.LOG
C:\WINDOWS\Prefetch\MSRLL.EXE-1068ACA9.pf
C:\WINDOWS\Prefetch\TASKMGR.EXE-20256C55.pf
C:\WINDOWS\system32\config\software.LOG
C:\WINDOWS\system32\config\system.LOG
C:\WINDOWS\system32\wbem\Repository\FS\INDEX.MAP
C:\WINDOWS\system32\wbem\Repository\FS\MAPPING.VER
C:\WINDOWS\system32\wbem\Repository\FS\MAPPING1.MAP
C:\WINDOWS\system32\wbem\Repository\FS\OBJECTS.MAP

Folders added:6

C:\WINDOWS\system32\mfm
C:\WINDOWS\system32\mfm\.
C:\WINDOWS\system32\mfm\..
C:\WINDOWS\system32\mfm
C:\WINDOWS\system32\mfm\.
C:\WINDOWS\system32\mfm\..

Total changes:59

Appendix B Bintext Output

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

0000004D   0040004D     0   !This program cannot be run in DOS mode.
00000088   00400088     0   [AspackDie!]
00000178   00400178     0   .text
000001A0   004001A0     0   .data
000001F0   004001F0     0   .idata
00000218   00400218     0   .aspack
00000240   00400240     0   .adata
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
00002567   00402567     0   PhV#@
000025DE   004025DE     0   &err: %u
0
00002820   00402820     0   PONG :%s
0000283A   0040283A     0   0h (@
0000299D   0040299D     0   %s!%s@%s
00002B3D   00402B3D     0   %s!%s
00002BB6   00402BB6     0   SVh=+@
00002BD7   00402BD7     0   irc.nick
00002BE0   00402BE0     0   NICK %s
00002EEA   00402EEA     0   NETWORK=
00002FF8   00402FF8     0   irc.pre
000032CC   004032CC     0   _%s__
000032D2   004032D2     0   __%s__
000032D9   004032D9     0   __%s___
000032E1   004032E1     0   NICK %s
000032F0   004032F0     0   %s %s
000036B0   004036B0     0   irc.chan
00003775   00403775     0   %s %s
WHO %s
000037C8   004037C8     0   PPhV,@
00003A45   00403A45     0   USERHOST %s
```

File pos	Mem pos	ID	Text
00003A52	00403A52	0	logged into %s(%s) as %s
00003A97	00403A97	0	<$hE:@
00003ABB	00403ABB	0	PhR:@
00003B99	00403B99	0	nick.pre
00003BA2	00403BA2	0	%s-%04u
00003BAA	00403BAA	0	irc.user
00003BB3	00403BB3	0	irc.usereal
00003BBF	00403BBF	0	irc.real
00003BC8	00403BC8	0	irc.pass
00003BE0	00403BE0	0	tsend(): connection to %s:%u failed
00003C20	00403C20	0	USER %s localhost 0 :%s NICK %s
00003DF5	00403DF5	0	Ph <@
000040BF	004040BF	0	PRIVMSG
00004100	00404100	0	trecv(): Disconnected from %s err:%u
0000446B	0040446B	0	NOTICE
00004472	00404472	0	%s %s :%s
00004615	00404615	0	Ph}D@
00004711	00404711	0	MODE %s -o+b %s *@%s
00004798	00404798	0	C'PSWh
000047B4	004047B4	0	Sh'G@
000047E7	004047E7	0	MODE %s -bo %s %s
0000487B	0040487B	0	Sh'G@
00004924	00404924	0	%s.key
00004A63	00404A63	0	Ph'G@
00004AA8	00404AA8	0	sk#%u %s is dead!
00004ABA	00404ABA	0	s_check: %s dead? pinging...
00004AD7	00404AD7	0	PING :ok
00004B00	00404B00	0	s_check: send error to %s disconnecting
00004B28	00404B28	0	expect the worst
00004B39	00404B39	0	s_check: killing socket %s
00004B54	00404B54	0	irc.knick
00004B5E	00404B5E	0	jtr.%u%s.iso
00004B6B	00404B6B	0	ison %s
00004B74	00404B74	0	servers
00004B7C	00404B7C	0	s_check: trying %s
00004DAA	00404DAA	0	Ph9K@
00004ED5	00404ED5	0	PhkK@
00004F41	00404F41	0	ShtK@
00004FD8	00404FD8	0	uYVh|K@
00005052	00405052	0	%s.mode
0000505A	0040505A	0	MODE %s %s
00005078	00405078	0	ShRP@
000050DA	004050DA	0	Sh$I@
000051A8	004051A8	0	PShZP@
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
000057AA	004057AA	0	
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
0000597B	0040597B	0	RhHY@
000059C6	004059C6	0	RhHY@
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>
00005B20	00405B20	0	%s logged in
00005B87	00405B87	0	Sh [@
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

File pos	Mem pos	ID	Text
00005C45	00405C45	0	unable to free %s
00005C6F	00405C6F	0	0h+\@
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
00005E09   00405E09     0   Ph@]@
00005E23   00405E23     0    %-5u %s
00005F8F   00405F8F     0   %s: %s
00005F96   00405F96     0   %s: somefile
0000603F   0040603F     0   PhHY@
000060D4   004060D4     0    host: %s ip: %s
00006269   00406269     0   capGetDriverDescriptionA
00006292   00406292     0   cpus:%u
000062A0   004062A0     0    WIN%s (u:%s)%s%s mem:(%u/%u) %u%% %s %s
000065CB   004065CB     0    %s: %s (%u)
00006708   00406708     0   %s %s
00006754   00406754     0   %s bad args
000067BC   004067BC     0   3hTg@
000067DA   004067DA     0    akick
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>
0000696F   0040696F     0   7h*h@
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
00006C2D   00406C2D     0   Ph+j@
00006C82   00406C82     0   Ph?j@
00006CBC   00406CBC     0    jtr.id
00006CC3   00406CC3     0   %s: <url> <id>
00006ED7   00406ED7     0   IREG
00006EDD   00406EDD     0   CLON
00006EE3   00406EE3     0   ICON
00006EF8   00406EF8     0   WCON
00006F40   00406F40     0    #%u [fd:%u] %s:%u [%s%s] last:%u
00006F63   00406F63     0     |\=> [n:%s fh:%s] (%s)
00006F82   00406F82     0     |---[%s] (%u) %s
00006F96   00406F96     0     |    |-[%s%s] [%s]
00006FAD   00406FAD     0     |=> (%s) (%.8x)
0000716E   0040716E     0   B$PRhco@
00007360   00407360     0   %s <pass> <salt>
000073C8   004073C8     0   %s <nick> <chan>
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
000075BD   004075BD     0   $h%u@
0000766A   0040766A     0   %s%u-%s
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
00007798   00407798     0   Whjv@
00007B9D   00407B9D     0   Ph w@
00007BE4   00407BE4     0   no file name in %s
00007DDB   00407DDB     0   %s created
00007E49   00407E49     0   %s %s to %s Ok
00007E8F   00407E8F     0   3hI~@
00007EE0   00407EE0     0   %0.2u/%0.2u/%0.2u  %0.2u:%0.2u  %15s  %s
00007F09   00407F09     0   %s (err: %u)
0000806B   0040806B     0   ShHY@

File pos   Mem pos      ID   Text
========   =======      ==   ====
00008085   00408085     0   err: %u
000080F8   004080F8     0   %s %s :ok
00008165   00408165     0   unable to %s %s (err: %u)
000081C3   004081C3     0   ShHY@
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
00008A96 00408A96 0 joins: %s
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
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
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
000093A9 004093A9 0 ?akick
000093B0 004093B0 0 ?part
000093B6 004093B6 0 ?dump
000093C6 004093C6 0 ?md5p
000093CC 004093CC 0 ?free
000093D7 004093D7 0 ?update
000093DF 004093DF 0 ?hostname
000093EE 004093EE 0 ?!fif
000093FE 004093FE 0 ?play
00009404 00409404 0 ?copy
0000940A 0040940A 0 ?move
00009415 00409415 0 ?sums
00009423 00409423 0 ?rmdir
0000942A 0040942A 0 ?mkdir
00009436 00409436 0 ?exec
00009440 00409440 0 ?kill
00009446 00409446 0 ?killall
0000944F 0040944F 0 ?crash
0000946E 0040946E 0 ?sklist
00009476 00409476 0 ?unset
0000947D 0040947D 0 ?uattr
00009484 00409484 0 ?dccsk
00009490 00409490 0 ?killsk
00009499 00409499 0 VERSION*
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.*
000099F5 004099F5 0 DiCHFc2ioiVmb3cb4zZ7zWZH1oM=
00009A16 00409A16 0 conf_dump: wrote %u lines
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)

File pos	Mem pos	ID	Text
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)
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
0000AC94	0040AC94	0	
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
0000BD57   0040BD57    0   jtr.id
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
0000BDE0   0040BDE0    0   $1$KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0
0000BE20   0040BE20    0   $1$KZLPLKDf$55isA1ITvamR7bjAdBziX.
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"
0000CE3D   0040CE3D    0   < u&
0000D010   0040D010    0   ./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
0000EA60   0040EA60    0   usage %s: server[:port] amount
0000EB33   0040EB33    0   %s: %s
0000EB3E   0040EB3E    0   %s %s %s <PARAM>
0000EB80   0040EB80    0   %s: [NETWORK|all] %s <"parm"> ...
0000EE20   0040EE20    0   USER %s localhost 0 :%s
0000EEE4   0040EEE4    0   PSVh
0000F140   0040F140    0   md5.c
0000F146   0040F146    0   md != NULL
0000F8F1   0040F8F1    0   buf != NULL

File pos   Mem pos      ID   Text
========   =======     ==   ====
0000F99F   0040F99F    0   hash != NULL
0000FAC5   0040FAC5    0   message digest
0000FAD4   0040FAD4    0   abcdefghijklmnopqrstuvwxyz
0000FB00   0040FB00    0   ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
0000FB40   0040FB40    0   1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
0000FCE0   0040FCE0    0   sprng
0000FD11   0040FD11    0   sprng.c
0000FD19   0040FD19    0   buf != NULL
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
0001023E   0041023E    0   #4EVgx
00010256   00410256    0   $5FWhy
00010282   00410282    0   #4EVgx
0001029A   0041029A    0   $5FWhy
000102C6   004102C6    0   #4EVgx
000102DE   004102DE    0   $5FWhy
000102F8   004102F8    0   gN]HU
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
Endianess: little (32-bit words)
Clean stack: disabled
Ciphers built-in:
   Blowfish
   RC2
   RC5
   RC6
   Serpent
   Safer+
   Safer
   Rijndael
   XTEA
   Twofish
   CAST5
   Noekeon
Hashes built-in:
   SHA-512
   SHA-384
   SHA-256
   TIGER
   SHA1
   MD5
   MD4
   MD2
Block Chaining Modes:
   CFB
   OFB
   CTR
PRNG:
   Yarrow
   SPRNG
   RC4
File pos   Mem pos     ID   Text
========   ========     ==   ====

PK Algs:
   RSA
   DH
   ECC
   KR
Compiler:
   WIN32 platform detected.
   GCC compiler detected.
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
000114F6   004114F6     0   t9VWS
0001154A   0041154A     0    prng != NULL
00011832   00411832     0   <"tx< tf< t
00011846   00411846     0   < tV<       t
00011852   00411852     0   < tJ<       tF
00011A10   00411A10     0   -LIBGCCW32-EH-SJLJ-GTHR-MINGW32
000130B0   004130B0     0    <ip> <total secs> <p size> <delay>
00013350   00413350     0   modem
00013358   00413358     0   Lan
0001335E   0041335E     0    Proxy
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
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

File pos	Mem pos	ID	Text
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
0011BCEE 0051BCEE 0 _onexit
0011BCFA 0051BCFA 0 _setmode
0011BD06 0051BD06 0 _vsnprintf
0011BD16 0051BD16 0 abort
0011BD1E 0051BD1E 0 atexit
0011BD32 0051BD32 0 clock
0011BD3A 0051BD3A 0 fclose
0011BD46 0051BD46 0 fflush

File pos	Mem pos	ID	Text
0011BD52	0051BD52	0	fgets
0011BD5A	0051BD5A	0	fopen
0011BD62	0051BD62	0	fprintf
0011BD6E	0051BD6E	0	fread
0011BD7E	0051BD7E	0	fwrite
0011BD8A	0051BD8A	0	malloc
0011BD96	0051BD96	0	memcpy
0011BDA2	0051BDA2	0	memset
0011BDAE	0051BDAE	0	printf
0011BDBA	0051BDBA	0	raise
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
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
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
0011C076   0051C076     0   select
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

File pos   Mem pos      ID   Text
========   =======      ==   ====
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
0011D6E6   0051D6E6     0   (08@P
0011D874   0051D874     0   D4l|M
0011D9C0   0051D9C0     0   ;;F,s
0011D9CF   0051D9CF     0   ,;F0s
0011D9DB   0051D9DB     0   ;F4s
0011DCB5   0051DCB5     0   D$$W3
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
```

- References

The following sources were used for reference in the writing of this paper.

Counter Hack – Ed Skoudis
Anti Hacker Tool Kit – Jones Schema & Johnson
Hacking Exposed – McClure, Scambray & Kurtz
Hacker Disassembling Uncovered – K. Kaspersky
http://support.microsoft.com/default.sapx?scid=kb;en-us;317843
http://www.bekkoame.ne.jp/~s_ita/port/port1900-1999.html
http://www.theeldergeek.com/ssdp_discovery_service.htm
http://www.seifried.org/security/ports/9000/9999.html
www.mirc.com/faq.html#section1
http://www.faqs.org/rfcs/rfc2812.html
http://www.faqs.org/rfcs/rfc1459.html
http://www.faqs.org/rfcs/rfc1413.html
http://www.faqs.org/docs/Linux-mini/IRC.html
http://www.iana.org/assignments/port-numbers

Last Updated: July 24th, 2017

Upcoming Training

SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Community SANS Ottawa FOR610	Ottawa, ON	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Baltimore Fall 2017 - FOR610: Reverse-Engineering Malware: Malware Analysis Tools and Techniques	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Boston FOR610	Boston, MA	Oct 09, 2017 - Oct 14, 2017	Community SANS
SANS Tokyo Autumn 2017	Tokyo, Japan	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
Cyber Threat Intelligence Summit & Training 2018	Bethesda, MD	Jan 29, 2018 - Feb 05, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced