Global Information Assurance Certification Paper

Reverse Engineering the *Msrll.exe* Trojan

Alan Ptak

January 2005

Table of Contents

Table of Figures

Table of Tables

1 Overview

This paper presents reverse engineering tools and techniques and the analysis of the malware file *msrll.exe*. The paper fulfills the practical component of the GIAC Reverse Engineering Malware (GREM) certificate, version 1.0, (23 July 2004). [1] The goals of this work are to demonstrate mastery of the subject area and to advance the state of the practice of information security. Actions and expected results are described in sufficient detail to allow others to repeat this work and obtain substantially equivalent results.

System monitoring tools, a disassembler and a debugger are used in a virtual analysis laboratory. Tools are used to analyze program code and monitor the malware as it runs in a controlled environment. Specific subroutines in *msrll.exe* are analyzed, referred to here as *RegMod, PassLogin, Pass* and *Compare*. *RegMod* creates registry keys to ensure that the malware runs whenever the system runs. *PassLogin, Pass* and *Compare* are involved in remote access via a backdoor network port.

Msrll.exe infects Win32 systems, including Windows 95, 98, NT, 2000, XP and Server 2003 platforms. *Msrll.exe* is presumably distributed through email, IRC or P2P networks. *Msrll.exe* appears to be used to store and distribute software, images and other (contraband) data. *Msrll.exe* connects to a specific IRC channel and server, presumably to participate in distributed denial of service (DDOS) attacks (e.g., jolt2, smurf) against arbitrary sites for financial or other gain.

This paper is organized as follows: Section 2 presents the laboratory environment and tools. Section 3 describes the static analysis. Section 4 describes the code analysis of *RegMod* and *PassLogin*. Section 5 presents the behavioral analysis. Section 6 presents the major conclusions and provides mitigating measures. Section 7 lists the references. Appendices A through J contain log files and related information and data.

2 Laboratory Setup

Dedicated laboratory systems provide the safe work environment needed to work with malicious code. The lab is equipped with tools for static, code and behavioral analyses. Static and code analyses provide detailed information on vulnerabilities exploited in the target system and the potential for damage. Programming skills and time are needed to understand the high-level meaning of low-level instructions and data. Behavioral analysis produces results quickly, but as with any experimental method, events depend heavily on conditions and evidence is buried in volumes of data. Together, these complementary methods produce a rapid and efficient evaluation of potential for harm, the techniques to restore infected systems and the means to mitigate future attacks.

2.1 VMware Virtual Workstation Environment

The lab uses VMware Workstation 4 [2] with virtual systems *Viroom* and *Victoom* connected to the virtual LAN *Vmnet1*. Figure 1 shows the virtual workstation environment, which runs on an Intel Pentium ® 4 platform. *Victoom* is the victim (target) of the attack. *Viroom* is used to interact with the malware and to monitor and capture

network traffic on *Vmnet1*. Table 1 summarizes the configuration for the host and virtual systems. Appendix A describes the virtual laboratory environment in greater detail.

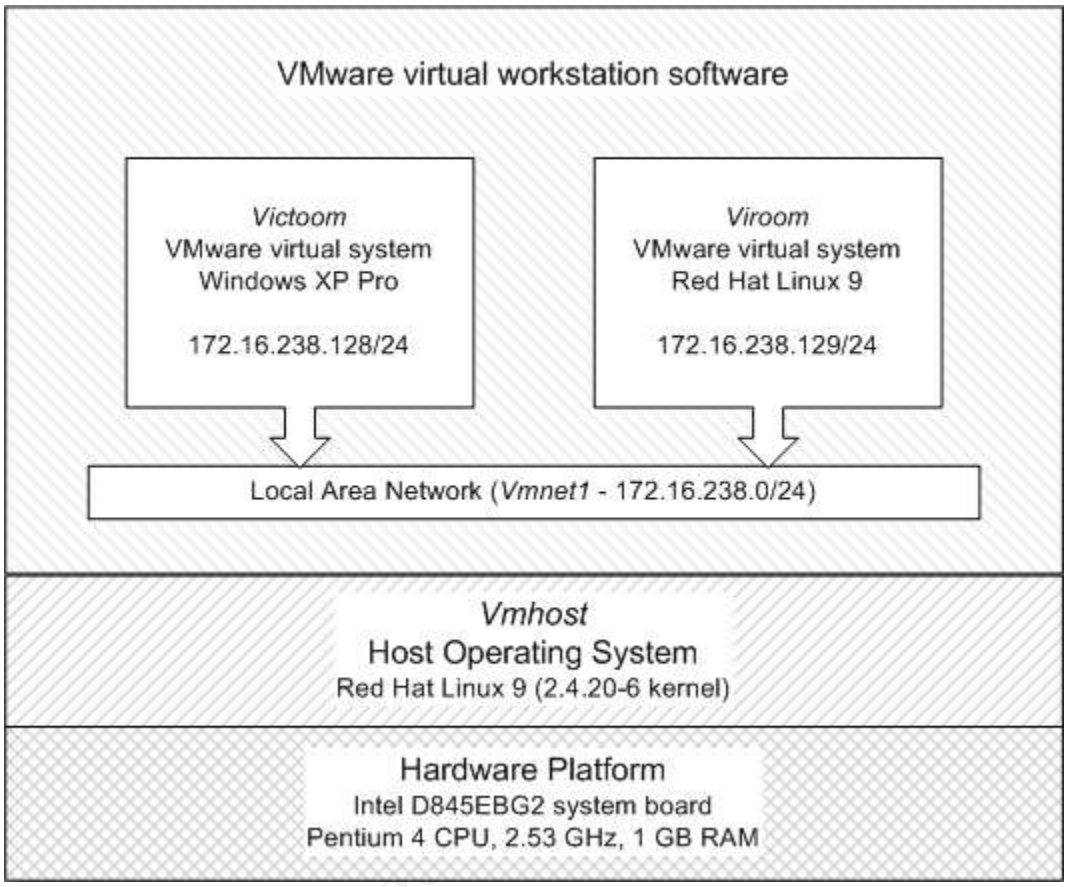

Figure 1: VMware laboratory environment

Table 1: VMware host and guest configuration

Hostname	OS	Machine specifications	IP address
Vmhost (VMware host)	Red Hat 9 2.4.20-6 kernel	Pentium 4 2.53 GHz CPU 533 MHz FSB 1024 MB RAM 80 GB EIDE disk drive (ATA-100)	*Vmnet1*: 172.16.238.1 (VMware virtual adapter)
Viroom (net monitor)	Red Hat 9 2.4.20-6 kernel	Virtual machine 256 MB RAM 2 GB SCSI virtual disk drive	172.16.238.129
Victoom (victim)	Windows XP Pro All updates prior to SP2	Virtual machine 256 MB RAM 4 GB SCSI virtual, pre-allocated	172.16.238.128

2.2 Analysis tools and software

Table 2 and Table 3 list the tools installed on *Victoom* and *Viroom*. All Windows tools, other than Norton Antivirus, can be obtained without cost on the Web for this project. All Linux tools are available at no cost on the web.

Table 2: Windows software tools installed on *Victoom*

Application	Tool Description, Usage and Source
7-Zip	Archive files. Program distributed under the GNU LGPL. Used to unpack malware from ZIP archive and to archive logs and data (TAR and Zip) Source: http://www.7-zip.org
AspackDie	Unpack Win32 PE programs compressed with AsPack. Used to unpack malware for static analysis. Source: http://www.exetools.com/files/unpackers/win/aspackdie13d.zip
BinText	Find ASCII, Unicode and Resource strings in a file. Used to find strings in malware. Source: http://www.foundstone.com/resources/termsofuse.htm?file=bintext.zip
FileMon	Monitor file system activity on a Windows system in real-time. Used to monitor filesystem activity while running malware. Source: http://www.sysinternals.com/files/NTFILMON.ZIP
IDA Pro	Disassemble and debug Win32 PE programs; features interactive, programmable, extendible user interface and extensive graphing functions. Used for static and behavioral analysis of binary malware code. Source: http://www.datarescue.be/downloaddemo.htm
LordPE	View or edit Win32 PE program headers. Used to analyze PE headers to determine code sections and parameters. Source: http://www.softpedia.com/get/Programming/File-Editors/LordPE.shtml
MD5	Compute message digests using the MD5 algorithm (RFC 1321). Used to identify identical copies of malware program and data files or files that have been changed. Source: http://www.fourmilab.ch/md5/md5.zip
Norton Antivirus	Remove viruses, worms, and Trojans from Win32 systems. Used to scan archives and files for known malware. Source: http://www.symantecstore.com/dr/v2/ec_MAIN.Entry17c?CID=39910&SID=27674&SP=10007&PN=5&PID=641220
OllyDbg	Disassemble and debug Win32 PE programs. Recognizes procedures, loops, API calls. Open architecture with plug-in support. Free software. Used for static and behavioral analysis of binary malware code. Source: http://home.t-online.de/home/OllyDbg/odbg110.zip
PSCP	Part of a free implementation of Telnet and SSH for Win32 and Unix. Used to securely transfer files between Windows and Linux system. Source: http://the.earth.li/~sgtatham/putty/latest/x86/putty.zip
Regmon	Monitor Registry activity by applications, keys and data in real-time. Used to monitor Registry activity while running malware. Source: http://www.sysinternals.com/files/ntregmon.zip
RegShot	Capture registry values and show differences between snapshots Used to find Registry and filesystem changes made by malware. Source: http://www.pcworld.com/downloads/file_download.asp?fid=19540&fileidx=1
TCPView	Show TCP and UDP connection state and endpoints, local and remote addresses, and associated processes in real-time. Used to monitor network connections while running malware. Source: http://www.sysinternals.com/files/tcpview.zip
TDIMon	Show TCP and UDP activity in real-time; operates at the OS kernel level. Used to monitor network activity while running malware. Source: http://www.sysinternals.com/files/tdimonnt.zip

Table 3: Linux software tools on *Viroom*

Application	Description
Ircd	Internet Relay Chat (IRC) server for text-based conferencing and file transfer Used to interact with malware during behavioral tests. Source: http://prdownloads.sourceforge.net/ircd-hybrid/ircd-hybrid-6.4.3.tgz
Ircii	Internet Relay Chat (IRC) client for text-based conferencing and file transfer Used to interact with malware during behavioral tests. Source: http://www.irchelp.org/irchelp/ircii/ircii-2.8.2.tar.gz
Netcat (nc)	Read or write data across networks using TCP or UDP connections; create interactive network listening servers. Used to interact with malware during behavioral tests. Source: http://www.securityfocus.com/data/tools/nc110.tgz
Opensshd	Secure remote access and file transfer tools. Used to securely transfer files between Windows and Linux systems. Source: ftp://ftp5.usa.openbsd.org/pub/OpenBSD/OpenSSH/portable/rpm/SRPMS/openssh-3.9p1-1.src.rpm
Snort	Capture, log and detect patterns in network traffic Used to monitor and capture network packets. Source: http://www.snort.org/dl/binaries/linux/snort-2.1.3-1.i386.rpm
VMware	Enables multiple virtual systems to run concurrently on a single physical machine. Used to create a secure isolated virtual environment for static and behavioral tests. Source: http://www.vmware.com/vmwarestore/newstore/wkst_eval_login.jsp

2.3 Precautions for working with malware

The malware lab is dedicated to malware analysis and isolated from live networks to protect others and ourselves. The potential for loss when working with malware is very real, as is the possibility of legal, social and possibly career-limiting consequences. Isolation protects the lab from outside influences that might interfere with the analysis and contaminate the results. This project adopted the following specific precautions:

- Network connectivity is limited at several levels.
  - The lab is isolated from other networks with an air gap.
- Media handling policies restrict data leaving the lab.
  - Only data files are allowed to leave the lab.
  - Only removable media (e.g., CDROM, floppy disk) are used to transfer data.
  - All media is scanned with antivirus software on a trusted system.
  - All systems used for documentation and data archiving use antivirus software.
- All storage devices are reformatted before reuse in the lab or elsewhere.

3 Static Analysis

Static analysis attempts to understand the malware's potential and its *modus operandi* by inspecting the code only, without allowing it to run. The analysis can also show how to remove the malware from infected systems and how to defend against future attacks.

The data collected includes metadata (file name, size and message digest signatures), embedded strings and disassembled code. Metadata helps researchers to uniquely

identify files. The MD5 message digest algorithm [3] was specifically designed to make the preparation of input text with a given signature computationally infeasible [4]. MD5 is not infallible but it is adequate for this project [5].

Text strings embedded in binary executable files provide clues to the program's origin and intent. Strings include *printf()* format strings, API and library names, hostnames, usernames, passwords, prompts and logging messages. The BinText utility lists ASCII and Unicode strings and reports their file position, memory position and type.

3.1 Collecting Metadata

Copy the malware provided by GIAC onto a CD. Place the CD in the CDROM drive on *Vmhost*. Navigate in VMware to *Victoom*, create the directory `c:\tmp` and copy the zip archive to `c:\tmp\msrll.zip`. Launch the 7-Zip File Manager by selecting **Start → Programs → 7-Zip → 7-Zip**. Browse to the file `c:\tmp\msrll.zip` and click Open. Click on Extract, enter the password, and click OK to extract *msrll.exe*.

Norton Antivirus identifies the file as the trojan *Backdoor.IRC.Bot* [6]. Norton Antivirus deletes the file and alerts the user (alert is similar to that shown in Figure 2). This is the first evidence that *msrll.exe* is malware. To proceed with the analysis, disable the antivirus software and repeat the steps in the preceding paragraph.

Launch a Windows Command Prompt. Select **Start → Run**, type `cmd.exe` and click OK. Type `cd \tmp`, `dir` and `md5 msrll.*` as shown in Figure 3. Table 4 lists the metadata for *msrll.zip*, *msrll.exe* and other malware files encountered later in this work.

Table 4: Metadata for malware files

File	Type	Size (bytes)	MD5 Hash
msrll.zip	Zip archive	39,100	`696c78651244b1ad0363a400a23d48ef`
msrll.exe	Extracted Win32 PE	41,984	`84acfe96a98590813413122c12c11aaa`
unpacked.ExE	Unpacked Win32 PE	1,175,552	`be5b7871ef6fe7dda24c2e11e83c3325`
msrll.exe	Created Win32 PE	1,175,552	`be5b7871ef6fe7dda24c2e11e83c3325`
jtram.conf	Created data file	1084	`c6f351047f463ed4da8f0c47cff24b14`

Figure 2: Norton Antivirus detection of *msrll.exe* being extracted from the zip archive

```
C:\tmp> dir
Volume in drive Z is alan
Volume Serial Number is 04D6-024D

Directory of c:\tmp

11/22/2004  07:56 PM    <DIR>          .
11/22/2004  07:59 PM    <DIR>          ..
04/15/2003  10:09 PM            49,152 md5.exe
05/10/2004  04:29 PM            41,984 msrll.exe
08/12/2004  01:25 AM            39,100 msrll.zip
2 File(s)         81,084 bytes
2 Dir(s)   5,337,251,840 bytes free

c:\tmp> md5 msrll.* > md5sums.txt

c:\tmp> type md5sums.txt
84ACFE96A98590813413122C12C11AAA  msrll.exe
696C78651244B1AD0363A400A23D48EF  msrll.zip

c:tmp> _
```

Figure 3: Computing MD5 message digests for malware files

3.2 Embedded Strings

Launch BinText. Select **Start → Run**, type `bintext.exe` and click OK. Click Browse, navigate to the `c:\tmp` directory, select *msrll.exe* from the list, click Open and click Go. Figure 4 shows the results. The main observations to note in Figure 4 are the following:

- **Text string *This program cannot be run in DOS mode.***
  This string is part of the *DOS stub* in the Win32 PE header**.** (see Appendix C)
- **Text strings *.text* and *.data***
  These strings identify the text and data code sections in the binary program.
- **Text string *.aspack***
  An *aspack* code section does not appear in ordinary Win32 PE files.
- **Most of the strings are meaningless.**
  Only 6 of the 185 strings found are intelligible.

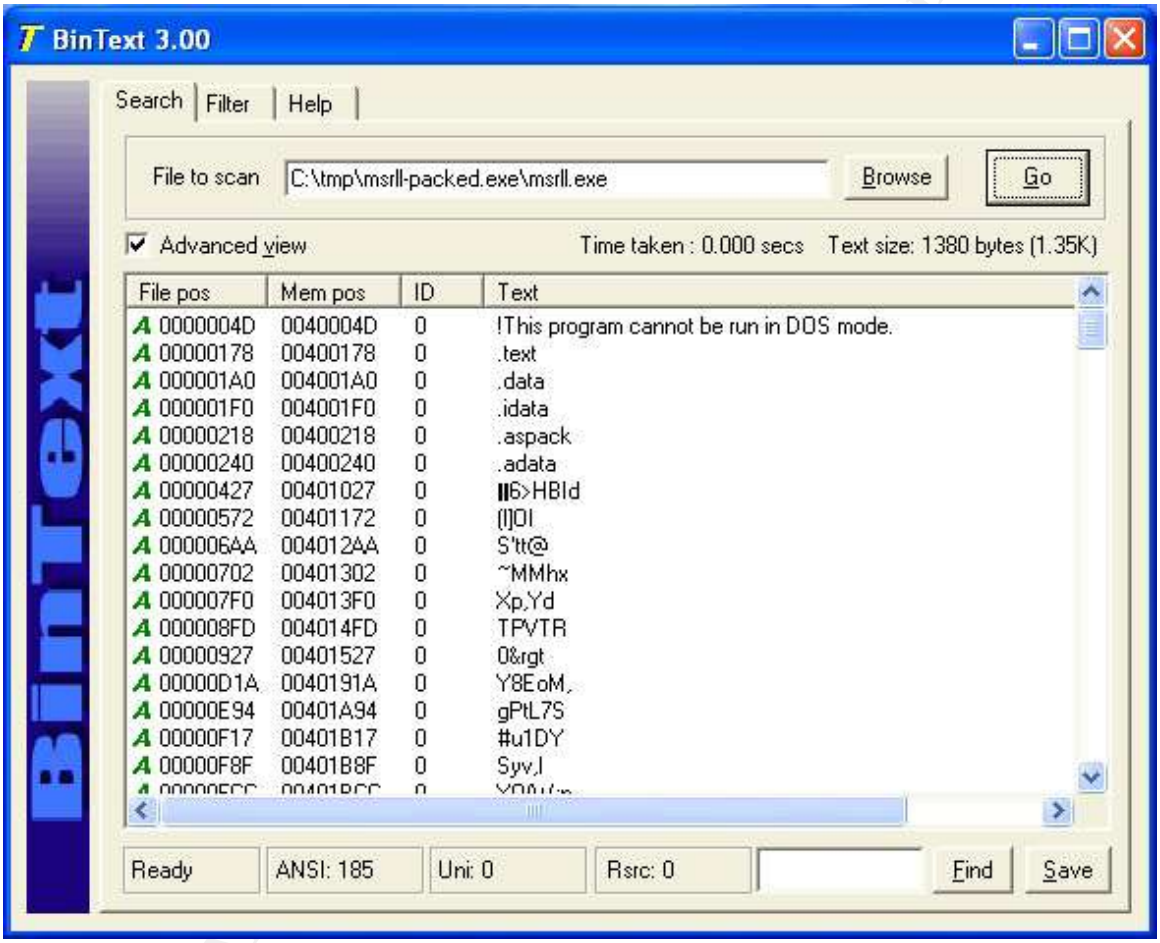

Figure 4: Embedded strings in msrll.exe (packed executable)

3.3 The Win32 PE Header

LordPE interprets files conforming to the Win32 PE specification (see Appendix C). Select **Start → Run**, type `lordpe.exe` and click OK. Click on PE Editor, browse to `c:\tmp`, select *msrll.exe* and click OK. Figure 5 shows the program entry point as 0x11d001. The program entry point normally resides in the text code section. Click on Sections to list the code sections (see Figure 6, also Figure 4). The text code section begins at 0x1000 (VOffset) and ends at 0x13000 (VSize + VOffset). The entry point of 0x11d001 resides in the aspack code section (VOffset = 0x11d000), not the text section.

Search for the term *aspack* with Google. [7] The link to the ASPACK web site [8] describes the AsPack tools that compress Win32 binary images. These tools add start-up code and adjust the section table and entry point in the PE header accordingly.

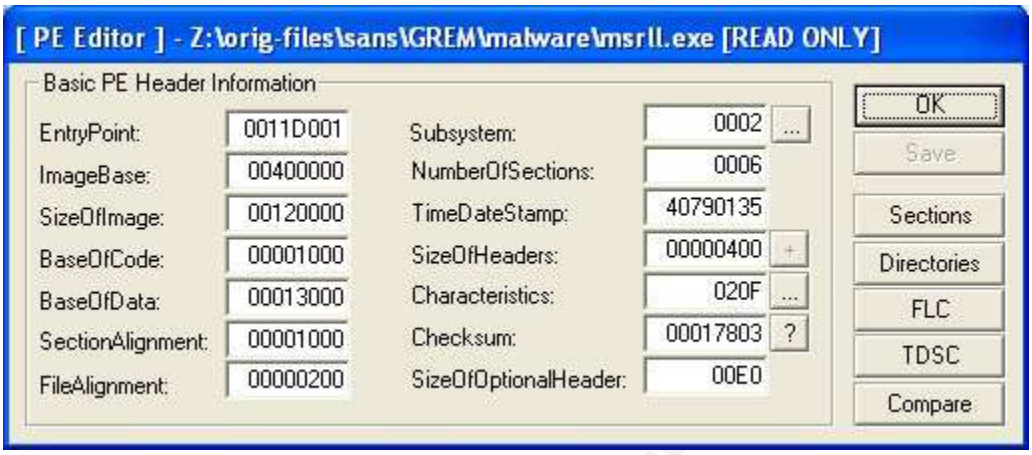

Figure 5: Basic header information for *msrll.exe*

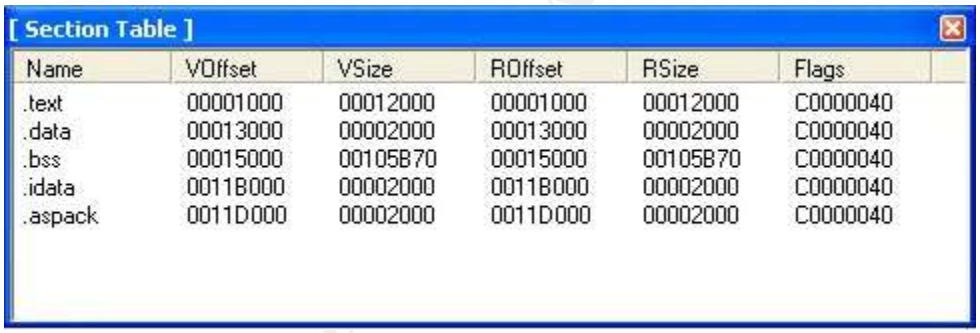

Figure 6: Code sections for *msrll.exe*

3.4 Unpacking msrll.exe

Google for *aspack unpacker* and select the link to Aaron's Homepage. [9] Aaron's site contains links to unpacking tools for AsPack. Find and download *aspackdie13d.zip*, transfer this file via CDROM to `c:\bin` on *Victoom* and use 7-Zip as before to extract *AspackDie.exe*. Select **Start → Run**, type `aspackdie.exe` and click OK. In the AsPack file browser, navigate to `c:\tmp`, select *msrll.exe* and click Open. A message similar to that seen in Figure 7 indicates that AspackDie unpacked *msrll.exe* and saved the result as *unpacked.ExE.* AspackDie also adjusts the Win32 PE header. Note that Norton Antivirus does not identify unpacked.ExE as the *Backdoor.IRC.Bot* trojan.

Collect the metadata for *unpacked.ExE* using md5 and LordPE as before. Table 4 contains the data collected. LordPE now shows the program entry point as 0x1240 (not shown). This address lies in the text code section, as was expected.

Figure 7: AspackDie output after unpacking *msrll.exe*

3.5 Embedded Strings II

Figure 8 shows the BinText output for *unpacked.ExE*. BinText lists 670 ASCII strings. All but a few irrelevant strings are listed in Appendix B. Table 5 lists selected strings that are used in this analysis. The significance of the selected strings is noted in the table.

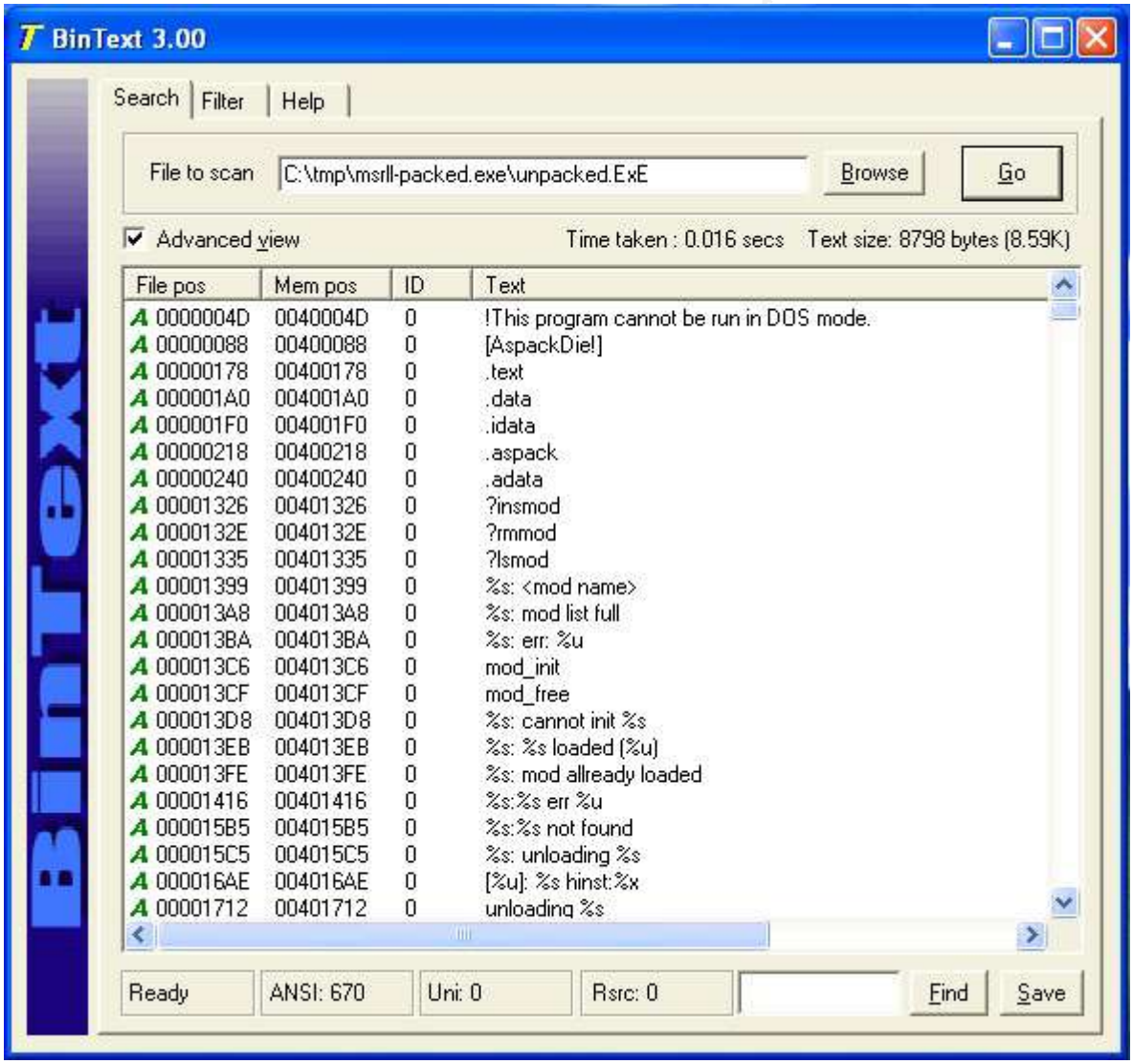

Figure 8: BinText string results for *unpacked.ExE*

Table 5: Selected strings from the unpacked trojan

Mem. Pos	String	Comments
0x4069eb 0x40bd38	*jtram.conf* *msrll.exe*	*jtram.conf* is created during installation Distribution file is copied as *msrll.exe*
0x40c8c0	software\microsoft\windows\currentversion\run	Registry key that causes a program to be run each time a user logs in.
0x40c897 0x40bd4e	Rll enhanced drive mfm	The names of legacy hard drive technologies are chosen to evade detection of the malware.
0x40BD80	collective7.zxy0.com, collective7.zxy0.com:9999!, collective7.zxy0.com:8080	Hostnames for IRC servers that the trojan will attempt to connect to when running.
0x40c5ba	%s : USERID : UNIX : %s	Response string sent on connection to the IDENT service (port 113)
0x403c20 0x403c38 0x40377b	USER %s localhost 0 :%s NICK %s WHO %s	User, nick and who are IRC commands. They appear here in *printf()* format strings, suggesting IRC capabilities.
00405B1B 00405B20	PASS %s logged in	Keyword associated with backdoor access on port 2200. Associated with *printf()* string to log connections made.
0x40BDE0 0x40BE20	1KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0 1KZLPLKDf$55isA1ITvamR7bjAdBziX.	Cryptic string containing password needed for connection on port 2200
0x401b59 0x40235e 0x4020a2	syn: done smurf done jolt2 done	*Printf()* format strings. Syn, smurf and jolt2 are names of denial of service attacks.
0x409394	?wget	wget retrieves files over HTTP, HTTPS and FTP

3.6 Finding Strings with OllyDbg

OllyDbg, like BinText, finds text strings in compiled programs. Unlike BinText, OllyDbg also disassembles the code, shows references to strings and provides context needed to understand their significance. This capability greatly improves static analysis work.

Launch OllyDbg. Select **Start → Run**, type `ollydbg.exe` and click OK. Select **File → Open**, browse to the directory in `c:\tmp`, select *unpacked.ExE* and click OK. Left-click in the top left quadrant of the CPU window, select **Search for → All referenced text strings**, and maximize the window, as shown in Figure 9. For instance, page down to the string *software\microsoft\windows\currentversion\run* at 0x40c8c0. This registry key is often targeted by malware to cause a program to run each time a user logs on. [10] As seen in Figure 9, instructions at 0x40cab0 and 0x40cadf reference this string.

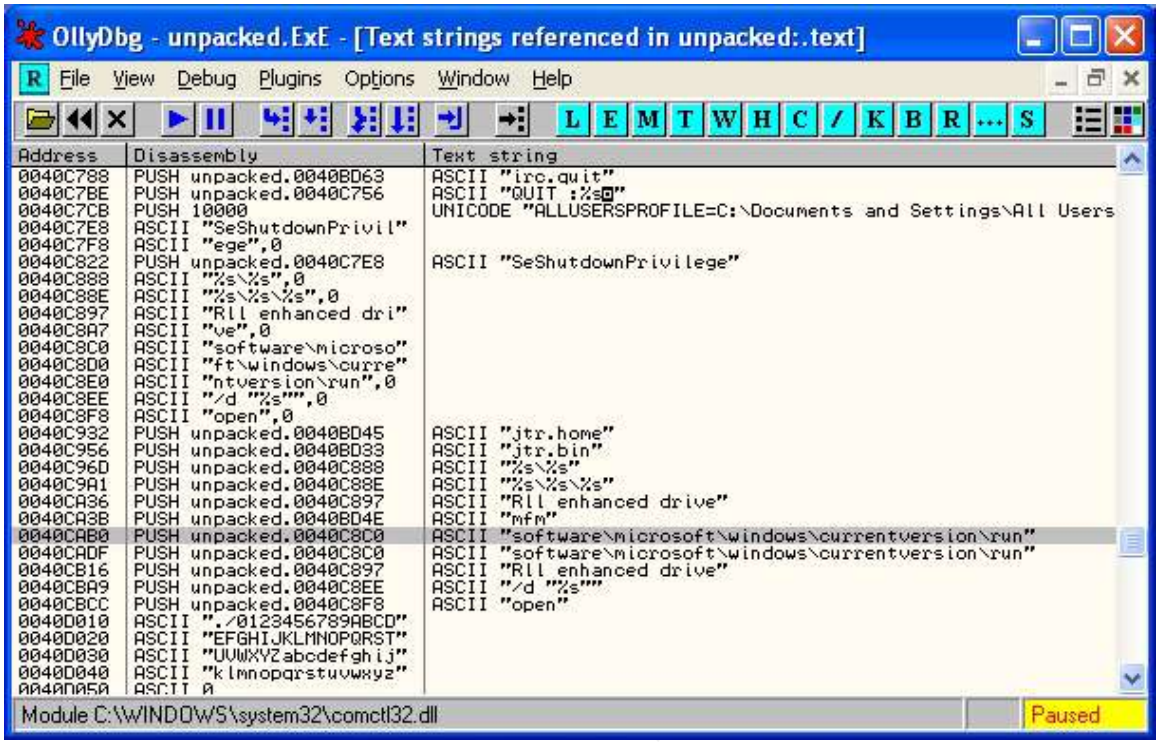

Figure 9: Strings displayed in OllyDbg

4 Code Analysis

4.1 The RegMod Subroutine

RegMod begins at 0x40c8fd is referred to here as. The registry key at 0x40c8c0 (noted in Figure 9, Table 5 and Section 3.6) raise suspicions about this subroutine's purpose.

In OllyDbg, highlight the string as shown in Figure 9 and press Enter. Figure 10 shows the context of the string in the program. OllyDbg adds the vertical bar in fourth column (from 0x40ca9a to 0x40caba) to highlight the arguments being passed to the function `RegCreateKeyExA`. [11], [12] This function call creates the registry key *HKLM\software\microsoft\windows\currentversion\run.* Similarly, the call at 0x40cae9 creates the key *HKCU\software\microsoft\windows\currentversion\run*.

The call at 0x40ca41 to `CreateServiceA` [13] also modifies the registry. `CreateServiceA` creates a service with the display name *Rll enhanced drive*. Table 6 lists the arguments that are passed to the function. Clearly, *RegMod* attempts to ensure that the trojan always runs. At least two methods are seen (as a start-up program or as a service). Both methods may not be needed or possible on all systems. The program may, for instance, select the appropriate method for victim's operating system version.

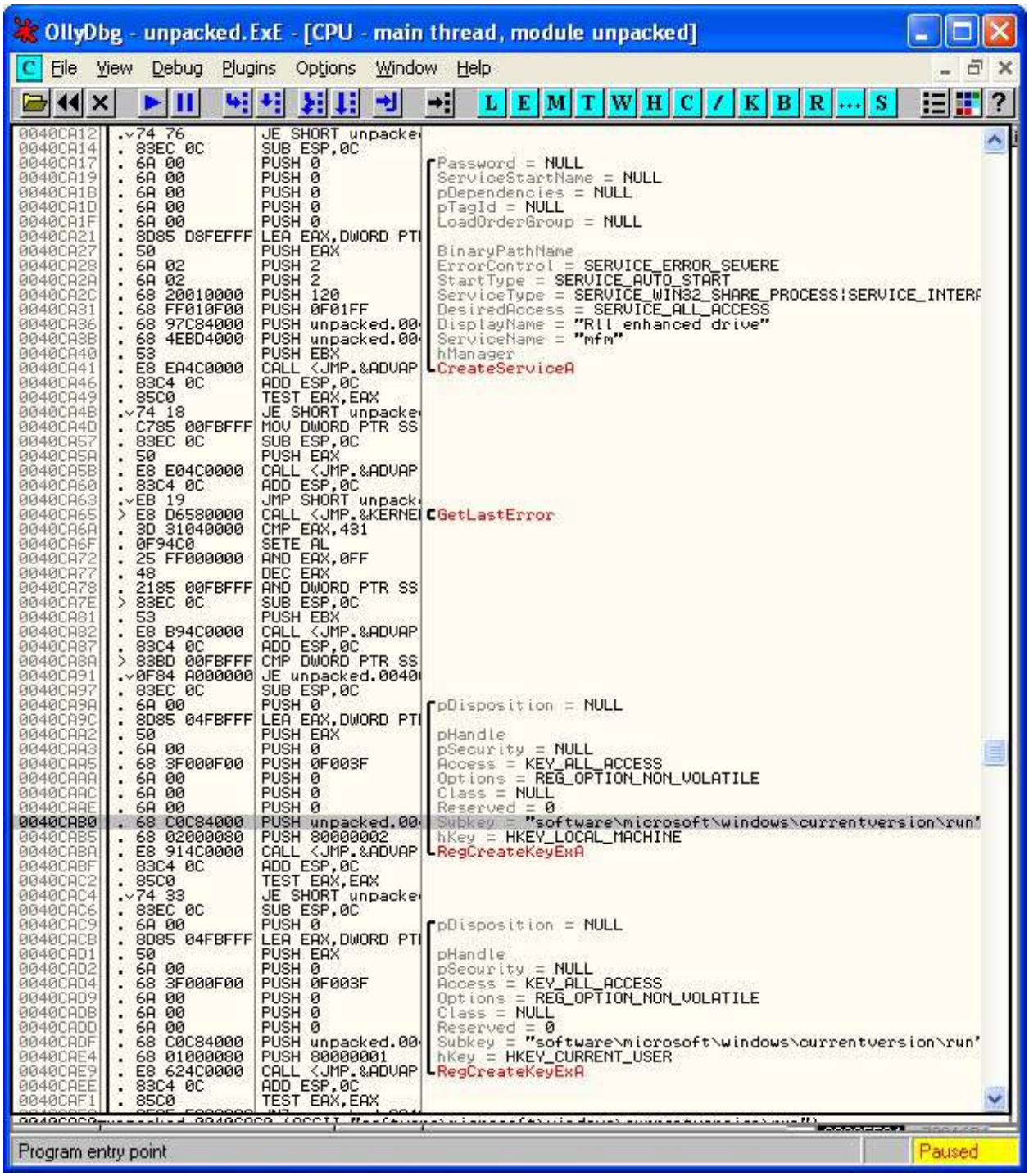

Figure 10: Service and registry key creation code fragment

Table 6: Library function *CreateServiceA* parameters

Parameter	Function Library Parameter Name	Parameter Value
BinaryPath	<none>	Unknown
ErrorControl	SERVICE_ERROR_SEVERE	0x2

Parameter	Function Library Parameter Name	Parameter Value
StartType	SERVICE_AUTO_START	0x2
ServiceType	SERVICE_WIN32_SHARE_PROCESS (logically OR'ed with) SERVICE_INTERACTIVE_PROCESS	0x120
DesiredAccess	SERVICE_AL_ACCESS	0x0f01ff
DisplayName	Rll enhanced drive	--
ServiceName	mfm	--

4.2 The PassLogin Subroutine

PassLogin begins at 0x405b2d. *PassLogin* appears suspicious because of the strings *PASS* and *%s logged in* being in close proximity (see Table 5). OllyDbg and IDA Pro are needed to study this further. Appendix C provides a brief introduction to the JUMP and CALL instructions and to OllyDbg.

Open *unpacked.ExE* in OllyDbg as before. Left-click in the disassembled code (top left) quadrant of the CPU window and select **Search for → All referenced text strings**. Maximize the window and highlight the top line. Left-click and select **Search for text**. Type *PASS*, clear the Entire Scope checkbox, check the Case Sensitive checkbox, and click OK. OllyDbg first finds this string at 0x415b1b. Press Ctrl-L to find the next instance of the string at 0x405b58 and press Enter. Figure 11 shows the context of the string in the program. Press Ctrl- - (Ctrl-minus) to move to the start of *PassLogin*.

4.2.1 The *strcmp()* function

Programmers often use the *strcmp()* function (e.g., [14]) to compare user passwords against a secret string. One hopes to find related subroutines that accept user input and then compare the user input with a string (the secret string) using *strcmp()*. Pointers to the two strings are passed as arguments to the function.

The user input and the secret string will not appear in the code directly, so some guesswork is involved. By locating *strcmp()* through code analysis, the argument values (and hence the password) can found by running the program in the OllyDbg debugger.

4.2.2 Jump and Call Instructions

PassLogin contains five flow control instructions and two subroutine calls. Recall that a JUMP conditionally breaks the sequential program flow depending on the CPU flags at that moment. JE and JNZ, seen here, are two of many variations of JUMP. [15] Recall that the sequential program flow resumes after the subroutine named in a CALL ends.

In *PassLogin*, execution flows sequentially from 0x4052bd to the JNZ instruction at 0x405b46. If the Z flag is not set (JNZ = **J**ump if **N**ot **Z**ero), the instruction at 0x405b9b is executed. Otherwise, the program flow continues to the CMP instruction at 0x405b48. The processor's Z flag is set on the result of the TEST instruction at 0x405b41.

The first CALL occurs at 0x405b60 to the *Pass* subroutine at 0x405872. As seen in Figure 11, *PASS* takes two arguments, labeled Arg1 and Arg2 by OllyDbg. The PUSH instruction [16] at 0x405b58 pushes Arg2 onto the stack. Arg2 is a 32-bit pointer to the string *PASS*. The value of this string can be seen by running the program in OllyDbg.

Arg1 is pushed onto the stack at 0x405b5d. Arg1 is clearly a variable. Knowledge of the Intel IA-32 segmented memory model and registers [17] is needed to understand this observation. The operand *DWORD PTR* forces the processor to treat Arg1 as a 32-bit pointer to a memory address. The expression “DS:[EDX+4]” is evaluated at run-time to determine this memory address. DS is the Data Segment register, typically used with global variables. [18], [19] The 32-bit EDX (Extended Data) register is typically used in data operations. Thus, the CPU evaluates the expression “(DS * 0x10) + EDX + 4”, fetches the value stored at that address and then pushes it on the stack. [20]

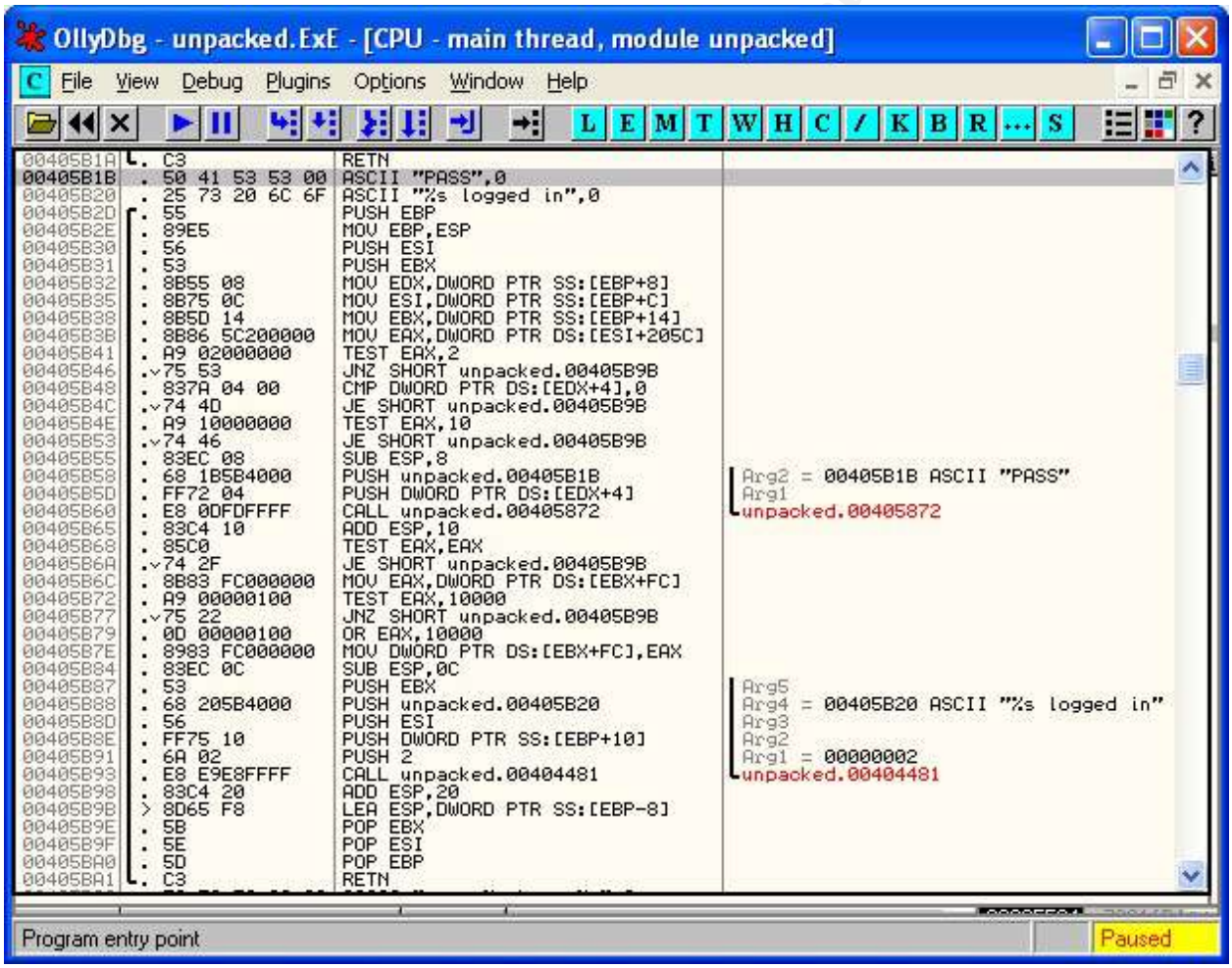

Figure 11: The *PassLogin* subroutine in OllyDbg

4.2.3 *PassLogin* Flowcharts and Graphs

The program logic just described is easier understood in diagrams than in text. IDA Pro has a powerful feature that produces flowcharts directly from program code. [21] Launch

IDA Pro by selecting **Start → Programs → IDA Pro Demo → IDA Pro Demo**. Click New on the Welcome to IDA Pro dialog box, navigate to the `c:\tmp` directory, select the file *unpacked.ExE* and click OK. Press the g key (go to a specific address) and type `405b2d` in the dialog box. Highlight all the lines of text between addresses 405b2d and 405ba1. Select **View → Graphs → Flowchart** to produce the flowchart shown in Figure 12. This graph breaks the code into blocks between branch points. Selecting **View → Display edge labels** shows the result needed to select a given branch. The result depends on the JUMP instruction and the processor flags at run-time. Figure 13 shows an equivalent flowchart produced by hand that emphasizes the branching structure.

Figure 12 and Figure 13 show that all “true” branches lead to the end of the subroutine. This likely indicates a failed login attempt. The graphs also show that the first three branches must evaluate to “false” before *Pass* is called. Assume for the moment that the necessary conditions exist to reach that point. Arg1 and Arg2 are pushed on the stack just prior to the CALL. If Arg1 is user input as hypothesized, then the *Pass* subroutine may include strcmp(), or it may call another subroutine that does.

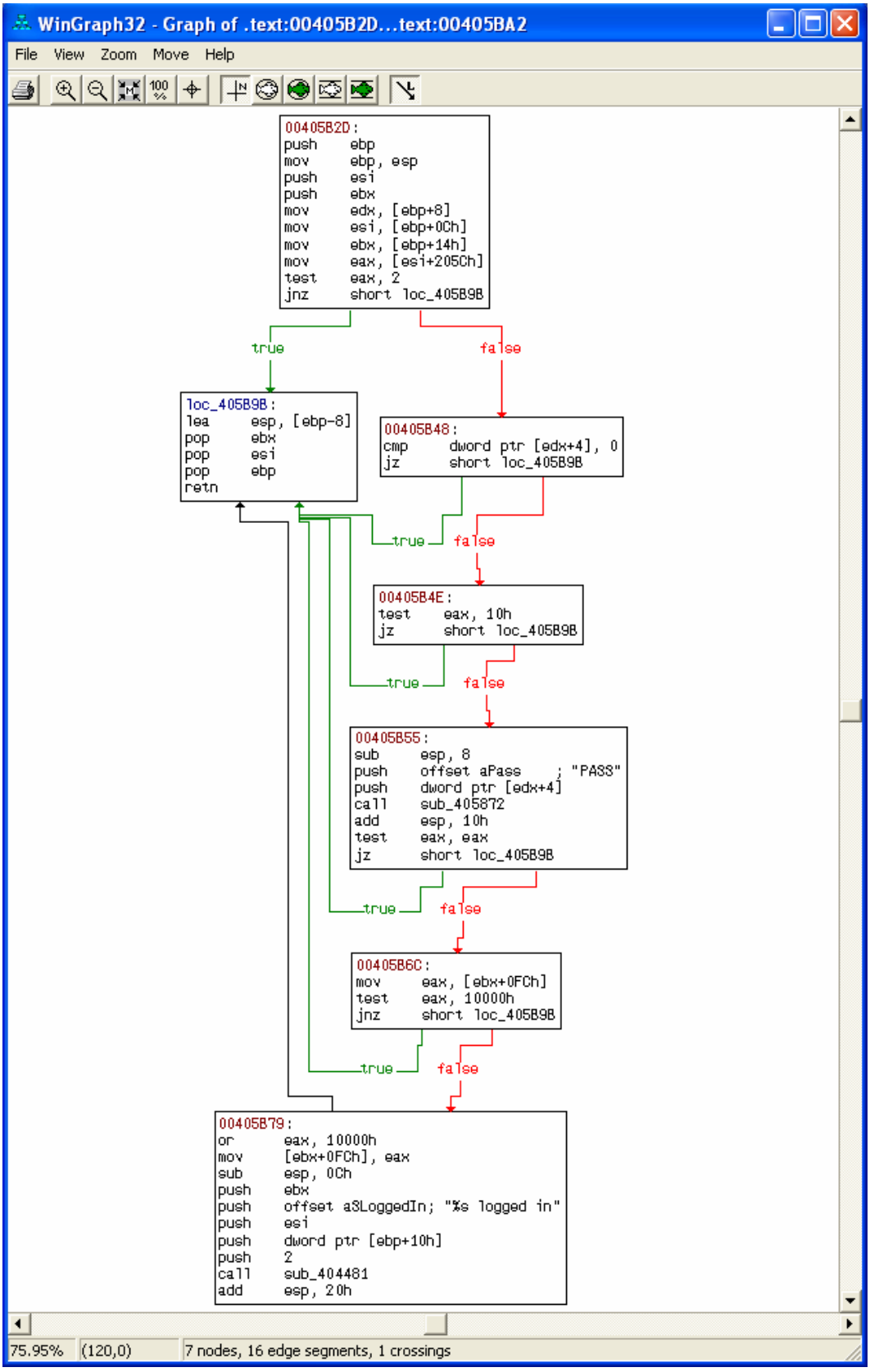

Figure 12: Flowchart for *PassLogin* generated by IDA Pro

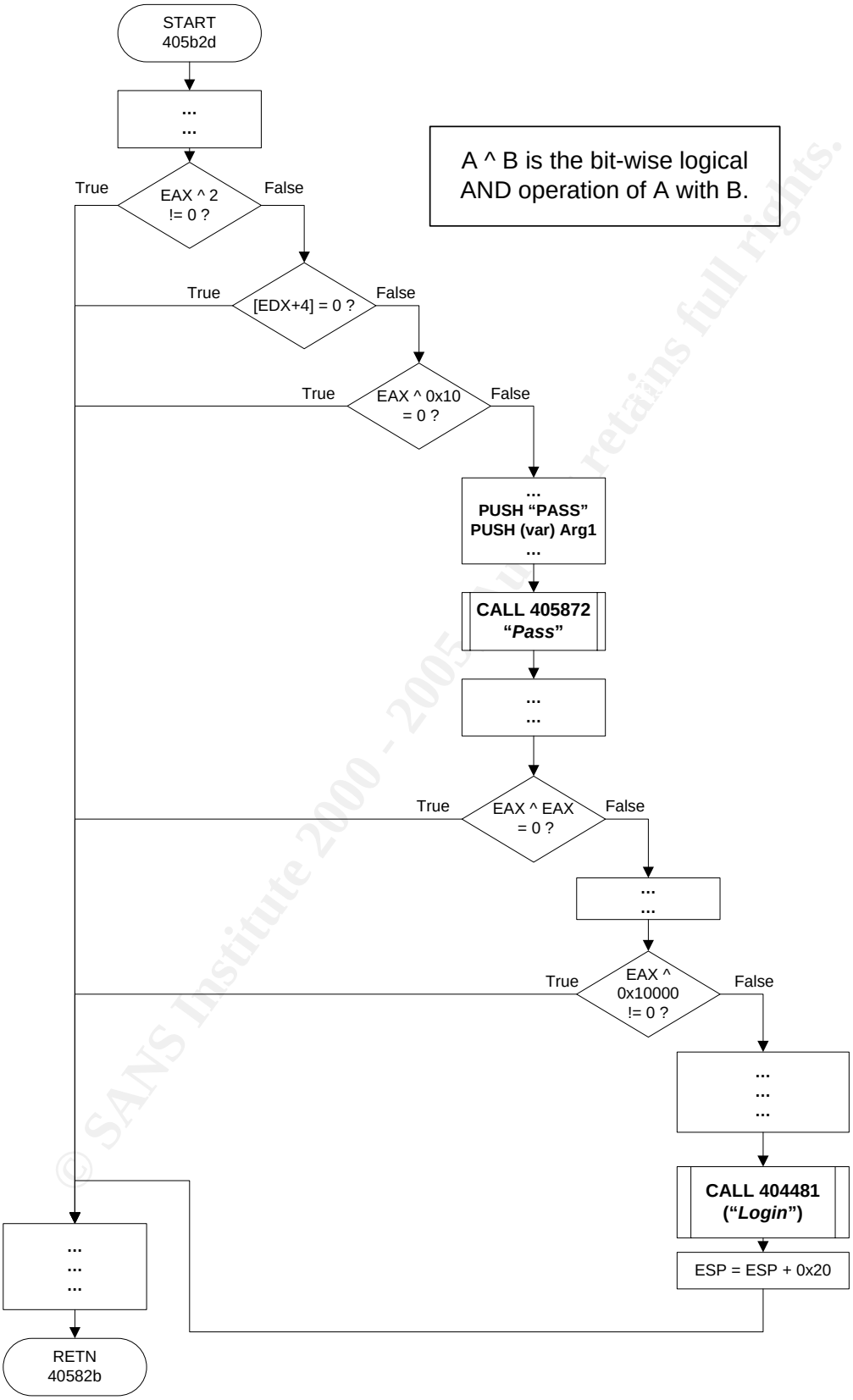

Figure 13: Flowchart for *PassLogin* created manually

4.2.4 The *Pass* subroutine

The *Pass* subroutine begins at 0x405872. Use IDA Pro as before to create the flowchart shown in Figure 14. *Pass* does not directly call *strcmp()*, so the analysis moves to *Compare* (at 0x40d611) and the subroutine at 0x40e7eb (named *Bidon*).

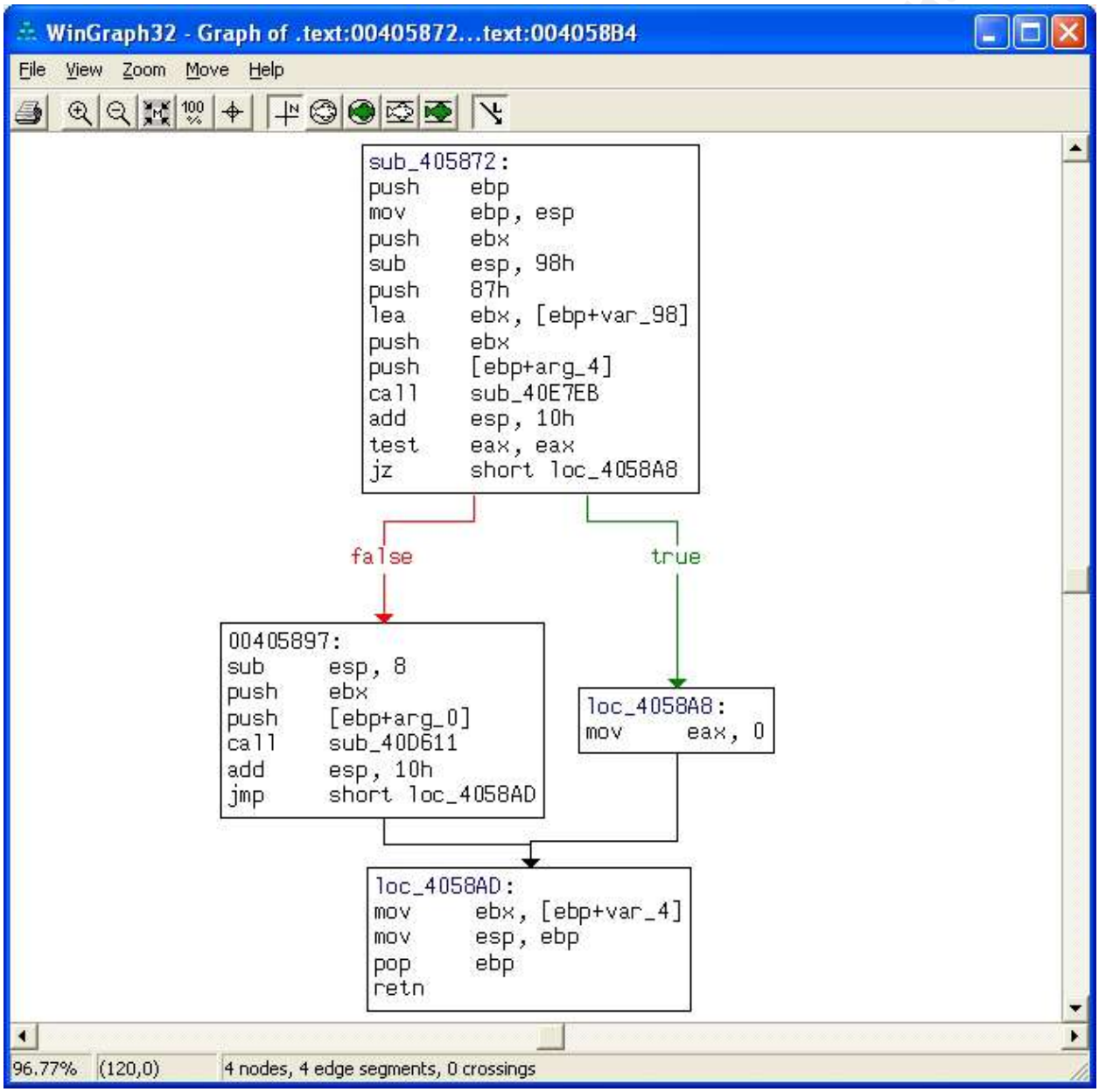

Figure 14: Flowchart for the *Pass* subroutine

4.2.5 The *Compare* Subroutine

Compare begins at 0x40d611. Press Ctrl-g in OllyDbg and type `406d11` to view the code (see Figure 15). *Compare* indeed calls *strcmp()* at 0x40d655. Arguments s1 and s2 are string pointers stored in registers EAX and ESI. The values of s1 and s2 can be

seen at run-time. Press F2 to set a breakpoint at 0x40d611. The analysis of *Compare* resumes in Section 5.4. There is no present need to examine *Login* or *Bidon*.

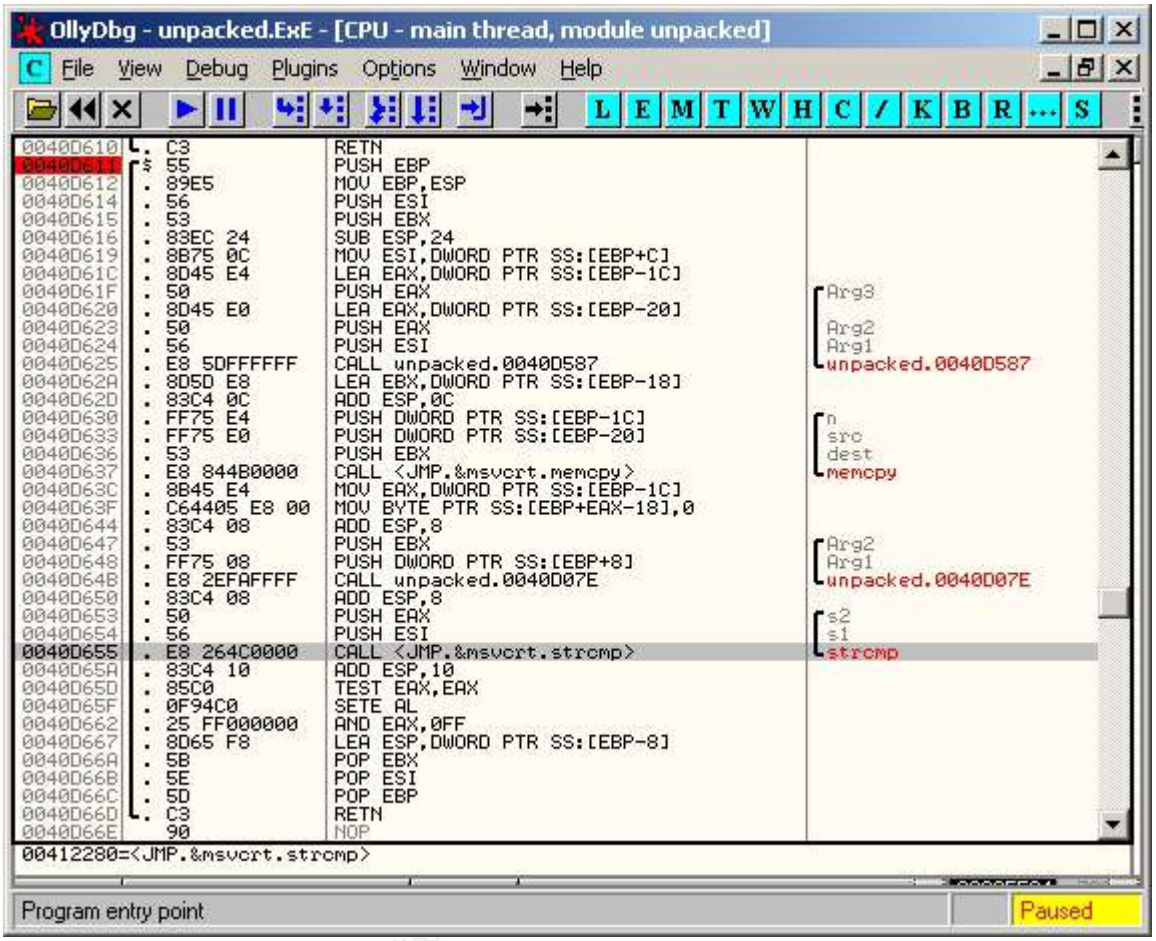

Figure 15: *Compare* in OllyDbg

5 Behavioral Analysis

5.1 Running the malware for the first time

5.1.1 Preparations on the target

This is the last opportunity to save the baseline system configuration for the analysis lab. Navigate to Victoom and select Snapshot → Save Snapshot in the drop-down menu on the VMware console. (This is purely for the reader's edification. The snapshot is not used nor discussed further here. To restore a snapshot, select Snapshot → Restore Snapshot.) This analysis uses the following monitoring tools:

- RegShot to find registry changes by comparing differences between snapshots.
- FileMon, RegMon and TDImon to log file system, registry and network events.
- TCPView to display running network connections and associated processes.

5.1.1.1 Launching monitoring tools

Launch RegShot by selecting **Start → Run**, typing `regshot.exe` and clicking OK. Click to check the option marked Scan dir1 and type `c:\` in the text box. Find the field labeled Output path: and type `c:\log`. Click 1st shot and then Shot and Save. Type `before-unpacked-exe` and click Save to save an image of the registry. (This registry snapshot is not used here.) Minimize the RegShot window.

Launch FileMon by selecting **Start → Run**, typing `filemon.exe` and clicking OK. Select **File → Capture Events** to stop capturing events, then **Edit → Clear Display**, and **Options → Show Milliseconds**.

Launch RegMon by selecting **Start → Run**, typing `regmon.exe` and clicking OK. Select **File → Capture Events** to stop capturing events, then **Edit → Clear Display**, **Options → Clock Time and Options → Show Milliseconds**.

Launch TDImon by selecting **Start → Run**, typing `tdimon.exe` and clicking OK. Select **Capture → Capture Events** to stop capturing events, then **Edit → Clear Display**. The TDImon clock resolution does not include millisecond precision.

Launch TCPview by selecting **Start → Run**, typing `tcpview.exe` and clicking OK. Select **Options → Resolve Addresses** to display a numeric listing of ports and IP addresses. Click on the column header labeled Process to sort the display by name.

5.1.2 Preparations on the network monitor

5.1.2.1 Stopping network services

Switch to *Viroom* and log in as root. At the shell prompt, type `ps -ef` to view all running processes. The output should be similar to the minimal list in Figure 16. Type `netstat --ip -nap` to view all network services and connections. No network services (e.g., ircd, named) should be running. Type `service sshd stop` to stop the SSH service. The netstat output should now be exactly as shown at the bottom of Figure 16. If netstat lists any service, stop the associated process before proceeding.

```
[root@viroom root]# ps –ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 Nov13 ?        00:00:04 init
root         2     1  0 Nov13 ?        00:00:01 [keventd]
root         3     1  0 Nov13 ?        00:00:00 [kapmd]
root         4     1  0 Nov13 ?        00:00:00 [ksoftirqd_CPU0]
root         9     1  0 Nov13 ?        00:00:00 [bdflush]
root         5     1  0 Nov13 ?        00:00:00 [kswapd]
root         6     1  0 Nov13 ?        00:00:00 [kscand/DMA]
root         7     1  0 Nov13 ?        00:15:53 [kscand/Normal]
root         8     1  0 Nov13 ?        00:00:00 [kscand/HighMem]
root        10     1  0 Nov13 ?        00:00:00 [kupdated]
root        11     1  0 Nov13 ?        00:00:00 [mdrecoveryd]
root        15     1  0 Nov13 ?        00:00:05 [kjournald]
root        73     1  0 Nov13 ?        00:00:00 [khubd]
root      2259     1  0 Nov13 ?        00:00:00 [kjournald]
root      2698     1  0 Nov13 ?        00:00:02 syslogd -m 0
root      2702     1  0 Nov13 ?        00:00:02 klogd –x
root      2859     1  0 Nov13 ?        00:00:01 /usr/sbin/sshd
root      3098     1  0 17:03 ?        00:00:00 login – root
root      3113  3098  0 17:03 tty1     00:00:00 –bash
root      3268  3113  0 17:05 pts/2    00:00:00 ps -ef
[root@viroom root]# netstat –-ip –nap
Active Internet connections (servers and established)
Proto  Recv-Q  Send-Q  Local Address  Foreign Address  State  PID/Program
tcp    0       0       0.0.0.0:22     0.0.0.0:*        LISTEN 2859/sshd
[root@viroom root]# service sshd stop
Stopping sshd:                                             [  OK  ]
[root@viroom root]# netstat –-ip –nap
Active Internet connections (servers and established)
Proto  Recv-Q  Send-Q  Local Address  Foreign Address  State  PID/Program

[root@viroom root]#
```

Figure 16: Processes and network services on *Viroom*

5.1.2.2 Starting Snort

Launch Snort by typing `snort -vd | tee /tmp/msrll-001.log` at the shell prompt. Figure 17 shows the Snort start-up message. Snort displays packets as they are captured and decoded. *Tee* writes the same output to the log file specified.

On *Viroom*, press Alt-F2 to open a new login session. Log in again as root and type `tail -f /tmp/msrll-001.tmp`. *Tail* displays captured packets as they are written to the log file by Snort. Tail can be interrupted without affecting the Snort session.

```
[root@viroom root]# snort -vd | tee /tmp/msrll-001.log
Running in packet dump mode
Log directory = /var/log/snort

Initializing Network Interface eth0

        --== Initializing Snort ==--
Initializing Output Plugins!
Decoding Ethernet on interface eth0

        --== Initialization Complete ==--

-*> Snort! <*-
Version 2.1.3 (Build 27)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)
_
```

Figure 17: Starting Snort in packet capture mode

5.1.3 Running the malware on *Victoom*

Switch back to *Victoom* and perform the following actions:

- Click Start → Run, type `cmd.exe` and click OK. Type `cd \tmp` and press Enter.
- Point the mouse to the Task Bar. Left-click and select **Task Manager**. Click on the Processes tab to view the running processes.
- In FileMon, select **File → Capture Events** to capture filesystem activity.
- In RegMon, select **File → Capture Events** to capture registry activity.
- In TDImon, select **Capture → Capture Events** to capture network activity.
- Click **Start → Run**, type *`unpacked.exe`* and click OK. The trojan is now running.

Within five to 10 seconds, *msrll.exe* will appear in the Task Manager and in TCPView. As seen in Figure 18, TCPview shows *msrll.exe* listening on TCP ports 113 and 2200. Select **File → Save As**, type `tcpview-001.log` and click OK to save this data.

Switch to *Viroom* and press Alt-F2. Within a minute or so, *Victoom* sends the DNS query for collective7.zyx0.com seen in Figure 19. Type `Ctrl-c` to interrupt tail. Type `less /tmp/msrll-001.log` to view packets captured so far. *Less* pauses the output after filling the screen. Type `/zxy0` to search through the text file for the string *zxy0*. If the string is not found, this event has not yet occurred. Type q to quit less. Type `tail -f /tmp/msrll-001.log` to resume watching the packets in real time. The ICMP response packet also shown in Figure 19 will follow the DNS query almost immediately.

Process	Protocol	Local Address	Remote Address	State
msrll.exe:1492	TCP	0.0.0.0:2200	0.0.0.0:0	LISTENING
msrll.exe:1492	TCP	0.0.0.0:113	0.0.0.0:0	LISTENING

Figure 18: TCPView entries for *unpacked.ExE* on *Victoom*

```
12/01-16:43:47.233759 172.16.238.128:1026 -> 172.16.238.1:53
UDP TTL:128 TOS:0x0 ID:1166 IpLen:20 DgmLen:66
Len: 38
01 F5 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C  .............col
6C 65 63 74 69 76 65 37 04 7A 78 79 30 03 63 6F  lective7.zxy0.co
6D 00 00 01 00 01                                m.....

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

12/01-16:43:47.237181 172.16.238.1 -> 172.16.238.128
ICMP TTL:255 TOS:0xC0 ID:21800 IpLen:20 DgmLen:94
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
172.16.238.128:1026 -> 172.16.238.1:53
UDP TTL:128 TOS:0x0 ID:1166 IpLen:20 DgmLen:66
Len: 38
** END OF DUMP
00 00 00 00 45 00 00 42 04 8E 00 00 80 11 01 76  ....E..B.......v
AC 10 EE 84 AC 10 EE 01 04 02 00 35 00 2E 2C FE  ...........5..,.
01 F5 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C  .............col
6C 65 63 74 69 76 65 37 04 7A 78 79 30 03 63 6F  lective7.zxy0.co
6D 00 00 01 00 01                                m.....

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

:
```

Figure 19: DNS request for collective7.zxy0.com in Snort log

5.1.4 Stopping the Malware

Switch back to *Victoom* when the DNS query and ICMP response are logged. In TCPView, left-click on the *msrll.exe* process, select End Process and click Yes on the TCPView prompt. *Msrll.exe* will end and drop off the process list.

5.1.5 Saving the event Logs

Save the event logs as follows:

- In FileMon, select **File → Capture Events** to stop capturing filesystem activity. Select **File → Save As …** to save the captured events to a text file. Navigate to the `c:\logs` directory. Type `Filemon-001` and click on Save.
- In RegMon, select **File → Capture Events** to stop capturing registry activity. Select File → Save As … to save the captured events to a text file. Navigate to the `c:\logs` directory. Type `Regmon-001` and click on Save.
- In TDImon, select **Capture → Capture Events** to stop capturing network activity. Select File → Save As … to save the captured events to a text file. Navigate to the `c:\logs` directory. Type `TDImon-001` and click on Save.
- In RegShot, click 2nd shot and then Shot and Save to save the current registry state. Type `after-unpacked-exe` and click OK for the registry snapshot file. Click on the HTML radio button and then on cOmpare. RegShot opens the report

in Internet Explorer. Select **File → Save** in the Internet Explorer window, navigate to `c:\log` and type `regshot-cmp-001` to save the report.

Use 7-Zip to collect all the log files into a TAR archive. Select **Start → Programs → 7-Zip → 7-Zip File Manager**, navigate to the `c:\` directory, and right-click on the log directory. Select **File → Add to archive** and type `c:\log-001.tar` for the archive file name. Select the Tar format from the Archive format drop-list, and click OK.

Switch to *Viroom* and press Alt-F1. Press `Ctrl-c` to stop Snort. Type `service sshd start` to restart the SSH daemon. Switch back to *Victoom* and use Putty SCP (pscp) as indicated in Figure 20 to copy the TAR archive to *Victoom*. Type the root password for *Viroom* when prompted to begin the transfer.

Switch again to *Viroom* and type `cd` to return to root's home directory. Type `tar xf log-001.tar` to extract the logs into the log directory created by tar. Move *msrll-001.log* to the `log` directory as shown in Figure 21.

```
C:\tmp>pscp c:\log.tar root@172.16.238.129:
root@172.16.238.129's password:
log-001.log      |         39 kB |  39.3 kB/s | ETA: 00:00:00 | 100%

C:\tmp>
```

Figure 20: Using pscp to copy files from *Victoom* to *Viroom*

```
[root@viroom root]# cd
[root@viroom root]# tar xf log-001.tar
[root@viroom root]# mv /tmp/msrll-001.log log
[root@viroom root]# ls –l log
total 18262
-rw-rw-r--    1 root    root   8227753 Nov 13 17:22 after–unpacked-exe.hiv
-rw-rw-r--    1 root    root   8227357 Nov 13 17:11 before–unpacked-exe.hiv
-rw-rw-r--    1 root    root    311676 Nov 13 17:19 Filemon-001.log
-rw-rw-r--    1 root    root     26880 Nov 13 17:58 msrll-001.log
-rw-rw-r--    1 root    root   1261824 Nov 13 17:19 Regmon-001.log
-rw-rw-r--    1 root    root     16771 Nov 13 17:27 regshot-cmp-001.html
-rw-rw-r--    1 root    root     95247 Nov 13 17:20 TDImon-001.log
-rw-rw-r--    1 root    root       105 Nov 13 17:20 TCPview-001.log
[root@viroom root]# _
```

Figure 21: Extracting the log files on *Viroom*

5.2 Analyzing the first run

5.2.1 Event Logs on the Victim

5.2.1.1 RegShot Snapshot Report

The RegShot report (see Appendix D) lists registry keys, data and values changed during the first run. Table 7 compares the observed events with those described in

Section 4.1 and Table 6. As expected, keys were created to run the trojan service *Rll enhanced drive* having the predicted attribute values. Events observed but not expected are shown with "---" in Table 7. Examples include the new files *msrll.exe* and *jtram.conf*. The analysis also predicted keys *HKLM\software\microsoft\windows\currentversion\run* and *HKCU\software\microsoft\windows\currentversion\run.* The keys were not created and may only be needed, for instance, on Windows versions that don't support services.

Table 7: Comparison of expected to observed registry events from RegShot

Object	Expected Event	Actual Event
C:\WINDOWS\system32\mfm	---	Dir added
C:\WINDOWS\system32\mfm\jtram.conf	---	File added
C:\WINDOWS\system32\mfm\msrll.exe	---	File added
C:\tmp\unpacked.ExE	---	File deleted
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm	---	Key Added
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\mfm\Security	---	Key Added
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm	Key Added	Key Added
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\Security	---	Key Added
SYSTEM\CurrentControlSet\Services\mfm\ErrorControl	0x2	0x2
SYSTEM\CurrentControlSet\Services\mfm\Start	0x2	0x2
SYSTEM\CurrentControlSet\Services\mfm\Type	0x120	0x120
SYSTEM\CurrentControlSet\Services\mfm\DisplayName	Rll enhanced drive	Rll enhanced drive
SYSTEM\CurrentControlSet\Services\mfm\ImagePath	Unknown	C:\WINDOWS\System32\mfm\msrll.exe

5.2.1.2 RegMon, FileMon, TDImon and TCPView logs

RegMon, FileMon, TDImon and TCPView logs are tab-separated ASCII files structured as shown in Table 8. This section describes the use of standard text processing utilities to efficiently reduce large logs into manageable datasets of relevant events.

Table 8: Log file format for RegMon, FileMon, TDImon and TCPView

Field	RegMon	FileMon	TDImon	TCPView
1	Event number	Event number	Event number	Process_name:PID
2	Timestamp	Timestamp	Timestamp	Protocol
3	Process_name:PID	Process_name:PID	Process_name:PID	Local_address:port
4	Event type	Event type	Event object	Remote_address:port
5	Event object	Event object	Event request	Status
6	Result	Result	Local_address:port	N/A
7	Other information	Other information	Remote_address:port	N/A
8	N/A	N/A	Result	N/A
9	N/A	N/A	Other information	N/A

5.2.1.3 Filtering Log Files with bash and egrep

Figure 22 shows egrep [22] used in a simple bash script [23], [24] to filter relevant events from multiple log files at once. The egrep filter selects only those lines containing *msrll* or *unpacked*. The bash script simply saves a lot of typing. The first line of the script selects each file in the current directory that ends in *.log*. In the next line, matching lines in that file are written to the terminal (standard out). The redirection character (>) sends the standard output to a file. The output filename is the input filename with the extension *.filt* added. The original file is unchanged. The last line ends the loop and the script.

Table 9 shows that the egrep filtered out up to 96.9% of the raw data logged (based on file size information from the `ls -l` command in Figure 22). Section 5.2.1.4 discusses the use of a text editor to cull the remaining relevant information from the logs.

```
[root@viroom root]# for myfile in `ls *.log` ;
> do egrep -i "*msrll*|*unpacked*" $myfile > $myfile.filt
> done
[root@viroom root]# ls –l
total 18262
-rw-rw-r--    1 root    root   8227753 Nov 13 17:22 after–unpacked-exe.hiv
-rw-rw-r--    1 root    root   8227357 Nov 13 17:11 before–unpacked-exe.hiv
-rw-rw-r--    1 root    root    311676 Nov 13 17:19 Filemon-001.log
-rw-rw-r--    1 root    root    151616 Nov 13 18:02 Filemon-001.log.filt
-rw-rw-r--    1 root    root     26880 Nov 13 17:58 msrll-001.log
-rw-rw-r--    1 root    root         0 Nov 13 17:58 msrll-001.log.filt
-rw-rw-r--    1 root    root   1261824 Nov 13 17:19 Regmon-001.log
-rw-rw-r--    1 root    root    725954 Nov 13 18:02 Regmon-001.log.filt
-rw-rw-r--    1 root    root     16771 Nov 13 17:27 regshot-cmp-001.html
-rw-rw-r--    1 root    root     95247 Nov 13 17:20 TDImon-001.log
-rw-rw-r--    1 root    root      2945 Nov 13 17:20 TDImon-001.log.filt
-rw-rw-r--    1 root    root       105 Nov 13 17:20 TCPview-001.log
-rw-rw-r--    1 root    root       105 Nov 13 17:20 TCPview-001.log.filt
[root@viroom root]# rm msrll-001.log.filt TCPview-001.log.filt
[root@viroom root]# vi Filemon-001.log.filt
```

Figure 22: Using egrep, a bash script and vi to find malware events in log files

Table 9: Data reduction using egrep on raw log files

File	Raw (bytes)	Filtered (bytes)	Reduction (%)
Filemon-001.log	311676	151616	51.4
Regmon-001.log	1261824	725954	42.5
TDImon-001.log	95247	2945	96.9

5.2.1.4 Filtering Log Files with vi

The last command shown in Figure 22 invokes the revered vi editor [25] (pronounced "vee-eye") on `filemon-001.log.filt`. Figure 23 shows vi's opening screen. The

bottom line is the status line, showing the file size (1377 lines, 151616 characters) and the cursor row and column location (1,1). Table 10 gives the essential vi commands to edit the log files. A count parameter with the *d* command deletes multiple lines at once. For instance, type d4d to delete the line under the cursor plus the 3 lines which follow.

Table 10: Essential vi editor commands

Command	Function
/	Defines a search string
n	Moves cursor to the next occurrence of the search string
N	Moves cursor to the previous occurrence of the search string
dd	Deletes the current line of text
u	Undoes the previous editor command
Ctrl-g	Shows file size and current cursor location
:q!	Quit without saving changes
:wq	Saves the file and exits

```
50      11:46:27.588 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
52      11:46:27.588 PM cmd.exe:1868    DIRECTORY       C:\tmp\ SUCCESS
FileBothDirectoryInformation: unpacked.ExE
54      11:46:27.604 PM cmd.exe:1868    OPEN    C:\tmp\unpacked.ExE
SUCCESS Options: Open  Access: All
55      11:46:27.604 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Length: 1175552
69      11:46:27.682 PM cmd.exe:1868    DIRECTORY       C:\tmp\ SUCCESS
FileBothDirectoryInformation: unpacked.ExE
71      11:46:27.682 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
76      11:46:27.682 PM cmd.exe:1868    DIRECTORY       C:\tmp\ SUCCESS
FileBothDirectoryInformation: unpacked.ExE
78      11:46:27.682 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
83      11:46:27.682 PM cmd.exe:1868    DIRECTORY       C:\tmp\ SUCCESS
FileBothDirectoryInformation: unpacked.ExE
85      11:46:27.682 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
86      11:46:27.682 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Length: 1175552
88      11:46:27.682 PM cmd.exe:1868    QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
"filemon-1204.log.filt" [dos] 1377L, 151616C                  1,1        Top
```

Figure 23: The vi editor on Filemon-001.log.filt

5.2.1.5 Filesystem Event Logs

Open `Filemon-001.log.filt` in vi, type */msrll.exe* and Enter. Vi skips to the first instance and highlights the string, as shown in Figure 24. The status line reports that this occurrence lies on line 75 between character 69 and character 81. This log entry (315) captures the event when *unpacked.ExE* creates the file

C:\WINDOWS\System32\mfm\msrll.exe. The log entry also confirms that the event succeeds. The four other occurrences of the string are also highlighted. In this fashion, search for *OPEN, CLOSE, READ, WRITE, CREATE* and *DELETE* to find other relevant events. Delete unrelated lines using `dd`. Save the file and exit vi using `:wq`. Table 11 summarizes the file `Filemon-001.log.filt`. The file is listed in Appendix E.

```
313     11:46:28.119 PM unpacked.ExE:256        QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
314     11:46:28.119 PM unpacked.ExE:256        QUERY INFORMATION
C:\tmp\unpacked.ExE     SUCCESS Attributes: A
315     11:46:28.119 PM unpacked.ExE:256        CREATE
C:\WINDOWS\System32\mfm\msrll.exe       SUCCESS Options: OverwriteIf
Sequential  Access: All
316     11:46:28.119 PM unpacked.ExE:256        OPEN
C:\WINDOWS\System32\mfm\        SUCCESS Options: Open  Access: 00000000
317     11:46:28.119 PM unpacked.ExE:256        CLOSE
C:\WINDOWS\System32\mfm\        SUCCESS
318     11:46:28.119 PM unpacked.ExE:256        QUERY INFORMATION
C:\WINDOWS\System32\mfm\msrll.exe       SUCCESS Attributes: A
319     11:46:28.119 PM unpacked.ExE:256        SET INFORMATION
C:\WINDOWS\System32\mfm\msrll.exe       SUCCESS Length: 1175552
320     11:46:28.119 PM unpacked.ExE:256        READ    C:\tmp\unpacked.ExE
SUCCESS Offset: 0 Length: 65536
321     11:46:28.119 PM unpacked.ExE:256        WRITE
C:\WINDOWS\System32\mfm\msrll.exe       SUCCESS Offset: 0 Length: 65536
322     11:46:28.119 PM unpacked.ExE:256        READ    C:\tmp\unpacked.ExE
SUCCESS Offset: 65536 Length: 65536
323     11:46:28.119 PM unpacked.ExE:256        WRITE
C:\WINDOWS\System32\mfm\msrll.exe       SUCCESS Offset: 65536 Length: 65536
/msrll.exe                                                75,69-81       4%
```

Figure 24: Searching for text string *msrll.exe* with vi

Table 11: Summary of malware events in Filemon-001.log

Log entry	Description
50 – 55, 97	cmd.exe queries the file system and launches unpacked.ExE
98	unpacked.ExE begins execution
268	unpacked.ExE creates directory C:\WINDOWS\system32\mfm
315	unpacked.ExE creates file C:\WINDOWS\system32\mfm\msrll.exe
319	unpacked.ExE sets file length of msrll.exe to 1175552 bytes
321	unpacked.ExE writes first 65536 bytes from file unpacked.ExE to file msrll.exe
323	unpacked.ExE writes second block of 65536 bytes to file msrll.exe
324 – 359	unpacked.ExE copies all remaining bytes from unpacked.ExE to msrll.exe
651 – 658	unpacked.ExE rereads parts of the first 224 bytes of msrll.exe (reason unknown)
723 – 725	msrll.exe begins execution
735 – 738	unpacked.ExE cleans up and terminates normally
1047 – 1089	msrll.exe deletes c:\tmp\unpacked.ExE
1270 – 1307	msrll.exe writes cookie, history and other data for user alan
1698 – 1701	msrll.exe creates file C:\WINDOWS\system32\mfm\jtram.conf

Log entry	Description
1756 – 1798	msrll.exe reads RSAENH.DLL crypto library routines
1855 – 2273	msrll.exe queries RSAENH.DLL and writes 1084 bytes to jtram.conf

5.2.1.6 Registry Event Logs

Edit `Regmon-001.log.filt` using vi as before to search the file for the strings *CreateKey, SetValue, CloseKey* and *DeleteKey*. Table 12 summarizes the most relevant registry events found. Delete the other lines using dd as before, save the file and exit. Appendix F lists the final file contents.

Log entry	Description
576 – 582	Unpacked.ExE creates and sets key for the Microsoft cryptographic provider
678 – 689	services.exe creates and sets keys for msrll.exe to run as a service with the display name of Rll enhanced drive
693 – 713	Unpacked.ExE creates and sets keys for the Microsoft cryptographic provider

Table 12: Summary of malware events in Regmon-001.log

5.2.1.7 Network Events Logs

File `TCPview-001.log`, listed in Figure 18, provided the initial evidence that the process listening on TCP ports 113 and 2200 is *msrll.exe*. The TDImon log corroborates this fact. Port 113 is officially assigned to the Identification (IDENT) service. [26], [27]

Use vi to search `TDImon-001.log` for *msrll.exe* as before. Table 13 summarizes the relevant network events. Delete the other lines as before, save the file and exit. Appendix G lists the final file `TDImon-001.log.filt`.

Log entry	Description
113 – 126	msrll.exe creates network connection object and configures the event handler for port 2200
183 – 196	msrll.exe creates network connection object and configures the event handler for port 113

Table 13: Summary of malware events in TDImon-001.log

5.2.2 Network Packet Log Files

Use vi as before to analyze the Snort log file `msrll-001.log`. Snort event logs span several lines, as seen in Figure 19, precluding use of the egrep filter used previously.

Figure 25 shows `msrll-001.log` open in vi. The first event is an ARP query for *Victoom*'s hardware address. The second is the reply from *Victoom* containing the 48-bit address. The third packet is a DNS query packet from *Victoom*. Snort captured the packet at 23:47:29.424418. The source address is 172.16.238.128 (*Victoom*) from port 1026. The destination is 172.16.238.1 to port 53. The packet payload includes the ASCII string *collective7.zxy0.com*.

By design, there is no DNS server in the lab environment to respond to the DNS query. Following another round of ARP, host 172.16.238.1 (*Vmhost*) sent an ICMP packet at 23:48:00.476980. *Vmhost* sent the ICMP type 3 code 3 packet [28] (*port unreachable*) to advise *Victoom* that there is no service listening on port 53. The payload of the rejected packet is also included in the ICMP response.

```
11/23-23:46:01.471743 ARP who-has 172.16.238.128 tell 172.16.238.1

11/23-23:46:01.473482 ARP reply 172.16.238.128 is-at 0:C:29:B1:63:AF

11/23-23:47:29.424418 172.16.238.128:1026 -> 172.16.238.1:53
UDP TTL:128 TOS:0x0 ID:887 IpLen:20 DgmLen:66
Len: 38
00 EB 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C  .............col
6C 65 63 74 69 76 65 37 04 7A 78 79 30 03 63 6F  lective7.zxy0.co
6D 00 00 01 00 01                                m.....

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

11/23-23:47:34.461658 ARP who-has 172.16.238.128 tell 172.16.238.1

11/23-23:47:34.462298 ARP reply 172.16.238.128 is-at 0:C:29:B1:63:AF

11/23-23:48:00.476980 172.16.238.1 -> 172.16.238.128
ICMP TTL:255 TOS:0xC0 ID:43677 IpLen:20 DgmLen:94
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
172.16.238.128:1026 -> 172.16.238.1:53
UDP TTL:128 TOS:0x0 ID:888 IpLen:20 DgmLen:66
Len: 38
** END OF DUMP
00 00 00 00 45 00 00 42 03 78 00 00 80 11 02 8C  ....E..B.x......
AC 10 EE 84 AC 10 EE 01 04 02 00 35 00 2E 2E 07  ...........5....
00 EC 01 00 00 01 00 00 00 00 00 00 0B 63 6F 6C  .............col
6C 65 63 74 69 76 65 37 04 7A 78 79 30 03 63 6F  lective7.zxy0.co
6D 00 00 01 00 01                                m.....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
                                                                 1,1           1%
```

Figure 25: Editing the Snort log file msrll-001.log with vi

5.2.3 Artifacts on the Victim

5.2.3.1 File *msrll.exe*

Msrll.exe is an artifact created by *unpacked.ExE*. The metadata in Table 4 shows that *msrll.exe* is identical to *unpacked.ExE*. The previous analysis in Section 5.2.1 shows that *unpacked.ExE* starts *msrll.exe* shortly after creating it. *Msrll.exe* then completes the installation process and begins to attempt to resolve the host *collective7.zxy0.com*.

BinText and OllyDbg reported the string *msrll.exe* (Sections 3.5 and 3.6), however its significance was not apparent at that time. Section 5.2.1.7 shows that *msrll.exe* listens on ports 113 and 2200. Other capabilities of *msrll.exe* are seen in later sections.

5.2.3.2 File *jtram.conf*

Msrll.exe creates the file `c:\WINDOWS\system32\mfm\jtram.conf` during installation. This is evident from RegShot report (Section 5.2.1.1) and the FileMon log (Section 5.2.1.5). After creating *jtram.conf*, *msrll.exe* alternates between queries to RSAENH.DLL [51] and writing blocks of data to *jtram.conf*. [1] Data is written in blocks of 53, 77 and 129 bytes. Table 14 shows the evidence supporting these observations.

Table 14: *jtram.conf* events from Filemon-001.log

ID	Timestamp	Event Description	Length	File Offset
1699	11:46:32.010	msrll.exe creates file C:\WINDOWS\system32\mfm\jtram.conf		
1861	11:46:32.697	Writes ASCII string	53	0
1887	11:46:32.979	Writes ASCII string	53	53
1916	11:46:33.276	Writes ASCII string	53	106
1917	11:46:33.276	Writes new-line (0x0a)	1	159
1969	11:46:34.041	Writes ASCII string	53	213
2004	11:46:34.604	Writes ASCII string	53	266
2005	11:46:34.604	Writes new-line (0x0a)	1	319
2036	11:46:34.947	Writes ASCII string	53	320
2062	11:46:35.213	Writes ASCII string	53	373
2067	11:46:35.260	Writes ASCII string	129	426
2068	11:46:35.260	Writes new-line (0x0a)	1	555
2099	11:46:35.697	Writes ASCII string	53	556
2120	11:46:35.994	Writes ASCII string	53	609
2147	11:46:36.041	Writes ASCII string	53	662
2148	11:46:36.041	Writes new-line (0x0a)	1	715
2179	11:46:36.057	Writes ASCII string	53	716
2208	11:46:36.057	Writes ASCII string	53	769
2213	11:46:36.057	Writes ASCII string	77	822
2214	11:46:36.057	Writes new-line (0x0a)	1	899
2245	11:46:36.072	Writes ASCII string	53	900
2266	11:46:36.088	Writes ASCII string	53	953
2271	11:46:36.088	Writes ASCII string	77	1006
2272	11:46:36.088	Writes new-line (0x0a)	1	1083
2273	11:46:36.088	msrll.exe closes file C:\WINDOWS\system32\mfm\jtram.conf		

Appendix H lists the contents of *jtram.conf* in hexadecimal and ASCII. Figure 26 shows the contents of *jtram.conf* in vi, arranged to emphasis the regularity in the data. The

[1] RSAENH.DLL can create and use keys for RSA Signature, RSA Key Exchange, RC2, RC4, DES, 3DES, and AES algorithms. [51]

space character marking the end of each data block is highlighted in yellow. All 1-byte strings are the new-line character (0x0a). The last three bytes in each block are "== " (0x3d 0x3d 0x20). The data resembles the base64-encoding scheme for translating binary data into printable ASCII characters. [29], [30] In this scheme, the "=" character is used for padding within the printable encoding procedure at the end of encoded strings. Additional investigation of this data file is beyond the scope of this analysis.

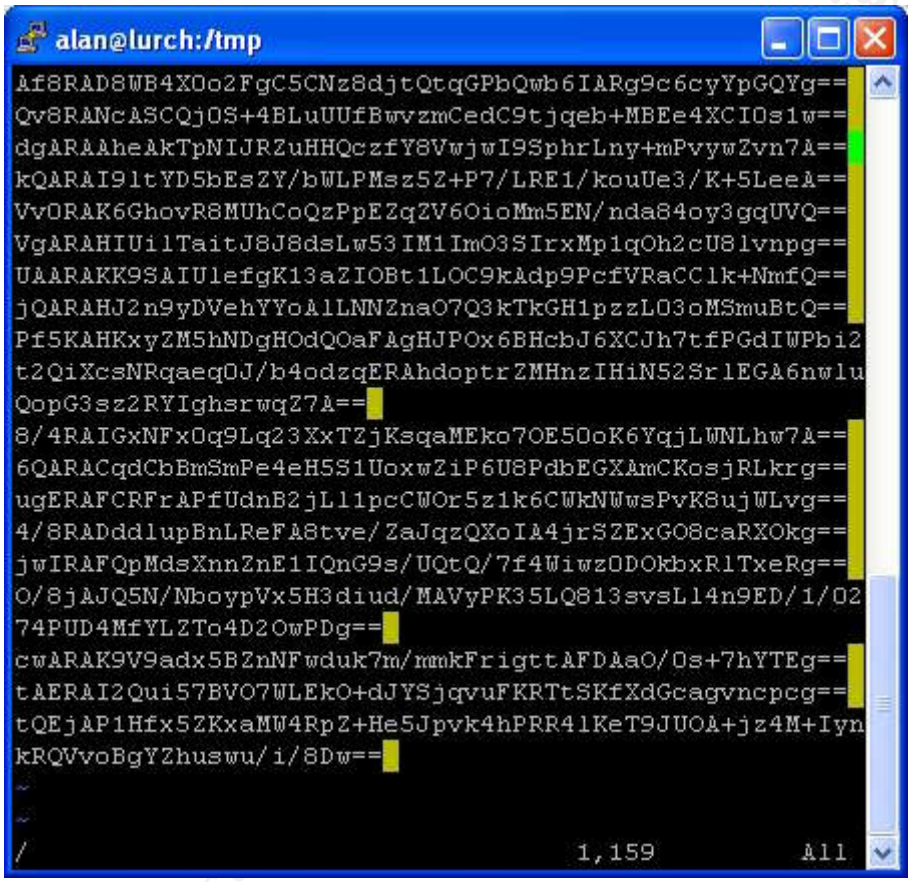

Figure 26: File *jtram.conf* arranged to show regular substrings in ASCII text

5.2.4 Combined Event Timeline

Table 15 shows these events in chronological order. The installation and startup for the malware required 1:01 minutes. Msrll.exe starts 2.0 seconds after the attack begins. Network services begin after 2.3 seconds, even before *jtram.conf* exists. The crypto functions involved in creating jtram.conf require 4.1 seconds to complete. This completes the analysis of the first run.

Table 15: Chronological Order of logged events

Time	Elapsed	Log File	ID	Event
11:46:27.697	00:00.0	FileMon	98	Unpacked.ExE starts
11:46:27.932	00:00.2	RegMon	576	Unpacked.ExE sets crypto keys

Time	Elapsed	Log File	ID	Event
11:46:28.072	00:00.4	RegMon	678	Services.exe sets keys for msrll.exe to run as a service
11:46:28.119	00:00.4	FileMon	315	Unpacked.ExE begins copying itself as msrll.exe
11:46:28.135	00:00.4	RegMon	693	Unpacked.ExE sets crypto keys
11:46:29.666	00:02.0	FileMon	723	Msrll.exe starts
11:46:29.744	00:02.0	FileMon	735	Unpacked.ExE terminates
11:46:29.994	00:02.3	FileMon	1047	Msrll.exe deletes unpacked.ExE
11:46:30.---	00:02.3	TDImon	113	Msrll.exe starts network service on port 2200
11:46:32.---	00:04.3	TDImon	183	Msrll.exe starts network service on port 113
11:46:32.010	00:04.3	FileMon	1699	Msrll.exe creates jtram.conf
11:46:36.088	00:08.4	FileMon	2272	Msrll.exe writes 1084 bytes to jtram.conf
11:47:29.424	01:01.7	Snort	---	DNS query for collective7.zxy0.com

5.3 Exploring the malware behavior with IRC

5.3.1 Resolving collective7.zxy0.com

The local hosts file and the host *Viroom* can be used to hijack the trojan's attempts to access collective7.zxy0.com. On *Victoom*, select **Start → Run**, type `notepad.exe` and press Return to open Notepad. Select **File → Open**, browse to the directory `C:\WINDOWS\system32\drivers\etc\`, select the file `hosts` and press OK. Figure 27 shows the only lines needed in the file. Save the file and exit Notepad.

```
127.0.0.1         localhost               # localhost
172.16.238.129    collective7.zxy0.com    # Viroom
```

Figure 27: Hosts file on *Victoom*

On *Viroom*, start Snort as before except with a log file name of `/tmp/msrll-002.log`.

On *Victoom*, start the trojan. Select **Start → Settings → Control Panel → Administrative Tools → Services**. Right-click on Rll enhanced drive and select Start. The process *msrll.exe* will appear in the Task Manager as soon as the trojan starts.

Back on *Viroom*, monitor the Snort output for connection attempts from *Viroom*. Figure 28 shows the Snort output that appears within several minutes. The first packet is a TCP packet from *Victoom* to port 6667 on *Viroom*. The asterisks on the third line represent the TCP flags. The S indicates that this was a SYN packet attempting to establish a TCP connection to port 6667. The second packet is the response from *Viroom*. The characters A and R indicate that the Acknowledgement and Reset flags were set by *Viroom*. The reset flag immediately terminates the connection attempt.

The change in behavior is a direct result of the host entry added on *Victoom*. *Msrll.exe* tries to connect to what it believes to be collective7.zxy0.com. In `msrll-002.log` (see Appendix I), Snort logs three attempts each to port 6667, 9999 and 8080. Port 6667 is

officially assigned to IRC. [31] Many trojans use IRC to alert the attacker that the infected system is available to participate in malicious activities. [32]

```
12/01-16:45:19.574746 172.16.238.128:1050 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:1168 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7BF18060  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:19.576865 172.16.238.129:6667 -> 172.16.238.128:1050
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7BF18061  Win: 0x0  TcpLen: 20

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

Figure 28: Snort output on *Viroom*

5.3.2 Listening on port 6667 with netcat

On *Viroom*, start netcat to listen on port 6667 as shown in Figure 29. The NICK [33] keyword in Figure 29 associates a nickname to a user on Internet Relay Chat (IRC). The trojan uses what appear to be random ASCII strings with the NICK and USER commands. The trojan closes the connection after a few seconds. The strings NICK and USER were found with BinText and OllyDbg as noted in Section 3.5.

```
[root@viroom root]# nc –l –p 6667
USER SalmSAAsR localhost 0 :xBemrbscSivenNXCKjXuiTvteofG
NICK kjkUxFiVSI
[root@viroom root]#
```

Figure 29: Using netcat to listen on port 6667

5.3.3 Running an IRC server

On *Viroom*, type `service ircd start` as root to start an IRC server. Start Snort as before with output file `/tmp/msrll-003.log`. Monitor network activity using tail and netstat for a connection from the trojan to the IRC server on port 6667. Stop Snort after a minute or two. Edit the log file in vi, keeping packets with source port 113 or 6667. Appendix J lists the edited file. Table 16 and Table 17 summarize the IRC session.

As expected, *msrll.exe* connects to an IRC channel (#*mils*) on collective7.zxy0.com. The IRC server and channel name are set when the trojan is compiled. After the exchange captured and described here, the trojan then simply listens on the channel. Presumably it is waiting for a specific message to initiate some other behavior.

Table 16: Description of packet payloads for IRC client/server exchange

ID	Description
1	Trojan acting as an IRC client sends IRC USER and NICK commands as in Figure 29, except with new random ASCII strings as parameters: • IRC username: Zetjd • Real name: OJnsKzIPB • Nickname: lETUSENtC
2	IRC server notifies the IRC client that it is trying to resolve the client's hostname using an inverse query to DNS. Note that the DNS query and response exchange is similar to that shown in Figure 25 and fails as before. These packets are not shown here.
3	Server sends IDENT query to client for the username owning the process associated with the IRC connection on port 1171.
4	Trojan listening on port 113 (see Figure 18) sends bogus IDENT reply to server with user ID pSaPvz. IDENT response format is UNIX.
5	Server notifies the client that it requested and received an IDENT response from the client's host.
6	Server notifies the client that the DNS query failed. Regardless, the IRC server allows the connection and continues.
7	Server sends sign-on message, usage statistics and message of the day (MOTD) to the client. The server response continues in packet 9.
8	Trojan sends USERHOST lETUSENtC to server. The reason for this may be to check if the server accepted the IRC and IDENT exchanges previously sent.
9	Continuation of sign-on message and MOTD response to client with MODE information. Mode for nickname lETUSENtC is invisible (:+ i)
10	Server responds to USERHOST command with data from the IDENT response (pSaPvz) and from the TCP packet stream. The server uses the IP address of client, since the hostname is unknown to the server. [34]
11	Trojan joins the IRC channel #mils.
12	Server response to JOIN. Response includes the JOIN message as required by IRC RFC 2812. [35]
13	Client queries server for the channel mode and list of users on the channel (MODE and WHO commands).
14	Server response to client. The channel does not accept external messages (mode n) and only channel operators can change the topic (mode t). [36] The channel users are listed one per line. The trojan is the only user. The letter H (**H**ere) denotes that the user (pSaPvz from host 172.16.238.128 with nickname lETUSENtC) is on-line. The symbol @ indicates that lETUSENtC is a channel operator.

Table 17: Payload from TCP packets for trojan IRC session

ID	Time	Src Port	Packet payload
1	18:11:12.595395	1171	`USER Zetjd localhost 0 :OJnsKzIPB` `NICK lETUSENtC`
2	18:11:12.596598	6667	`NOTICE AUTH :*** Looking up your hostname`
3	18:11:12.711235	32771	`1171 , 6667`
4	18:11:12.715972	113	`1171 , 6667 : USERID : UNIX : pSaPvz`

ID	Time	Src Port	Packet payload
5	18:11:12.741535	6667	NOTICE AUTH :*** Checking Ident NOTICE AUTH :*** Got Ident response
6	18:11:39.668168	6667	NOTICE AUTH :*** Couldn't look up your hostname
7	18:11:40.298319	6667	:localhost.localdomain 001 lETUSENtC :Welcome to the Internet Relay Network lETUSENtC :localhost.localdomain 002 lETUSENtC :Your host is localhost.localdomain[localhost.localdomain/6667], running version 2.8/hybrid-6.3.1 NOTICE lETUSENtC :*** Your host is localhost.localdomain[localhost.localdomain/6667], running version 2.8/hybrid-6.3.1 :localhost.localdomain 003 lETUSENtC :This server was created Tue Jun 4 2 002 at 16:59:45 EDT :localhost.localdomain 004 lETUSENtC localhost.localdomain 2.8/hybrid-6.3.1 oOiwszcrkfydnxb biklmnopstve :localhost.localdomain 005 lETUSENtC WALLCHOPS PREFIX=(ov)@+ CHANTYPES=#& MAXCHANNELS=20 MAXBANS=25 NICKLEN=9 TOPICLEN=120 KICKLEN=90 NETWORK=EFnet CHANMODES=b,k,l,imnpst MODES=4 :are supported by this server. :localhost.localdomain 251 lETUSENtC :There are 0 users and 1 invisible on 1 servers :localhost.localdomain 255 lETUSENtC :I have 1 clients and 0 servers :localhost.localdomain 265 lETUSENtC :Current local users: 1 Max: 1 :localhost.localdomain 266 lETUSENtC :Curren
8	18:11:40.307318	1171	USERHOST lETUSENtC
9	18:11:40.307449	6667	t global users: 1 Max: 1 :localhost.localdomain 250 lETUSENtC :Highest connection count: 1 (1 clients) (1 since server was (re)started) :localhost.localdomain 375 lETUSENtC :- localhost.localdomain Message of the Day - :localhost.localdomain 372 lETUSENtC :- This is an IRC server. Authorized users only. :localhost.localdomain 376 lETUSENtC :End of /MOTD command. :lETUSENtC MODE lETUSENtC :+ i
10	18:11:40.668153	6667	:localhost.localdomain 302 lETUSENtC :lETUSENtC=+pSaPvz@172.16.238.128
11	18:11:44.639459	1171	JOIN #mils :
12	18:11:44.639889	6667	<join response> :lETUSENtC!pSaPvz@172.16.238.128 JOIN :#mils :localhost.localdomain MODE #mils +nt :localhost.localdomain 353 lETUSENtC = #mils :@lETUSENtC :localhost.localdomain 366 lETUSENtC #mils :End of /NAMES list.
13	18:11:47.628415	1171	MODE #mils WHO #mils

ID	Time	Src Port	Packet payload
14	18:11:47.678104	6667	<MODE WHO response> :localhost.localdomain 324 lETUSENtC #mils +tn :localhost.localdomain 329 lETUSENtC #mils 1101 942704 :localhost.localdomain 352 lETUSENtC #mils pSaPvz 172.16.238.128 localhost.localdomain lETUSENtC H@ :0 OJnsKzIPB :localhost.localdomain 315 lETUSENtC #mils :End of /WHO list.

5.3.4 Joining the #mils channel

Connect to the IRC server as a client and join the channel with the trojan. At a shell prompt on *Viroom*, create a normal user account rather than running an IRC client as root. Type `useradd toto` and press Enter to create a user account. Type `su - toto` to login as toto. Now type `irc MyNickName 172.16.238.129` to start the IRC client with nickname MyNickName and connecting to the IRC server on *Viroom*. Figure 30 shows the startup screen on the IRC client. The client screen is divided into two regions separated by the highlighted status bar. All user input appears below the status bar and is cleared after each command. Comments added for clarity are shown in bold red text.

The MOTD (ending with *Authorized users only*.) is the last of the initial start-up output. The IRC client automatically sets our mode to invisible. The *message *** Mode change +i for user MyNickName by MyNickName* is the server's confirmation of the change.

Repeat the three IRC commands issued by the trojan. Type `/JOIN #mils`, `/MODE #mils` and `/WHO #mils` (each followed by Enter). The server responds to confirm the JOIN. The dialog closely resembles that presented to *unpacked.ExE*, for instance, the channel mode (refer to Table 16 and compare with Figure 30). The server now lists two users, *toto* (this connection) and *lETUSENtC* (*msrll.exe*).

This exercise shows that merely the presence of another user on the channel is not sufficient to elicit additional behavior from the trojan. Further exploration of the trojan's behavior requires a detailed code and behavioral analysis using OllyDbg (beyond the scope of this work). This concludes the analysis of the trojan's behavior using IRC.

```
*** Connecting to port 6667 of server 172.16.238.129
*** Looking up your hostname...
*** Checking Ident
*** No Ident response
*** Couldn't look up your hostname
=== Setting this servers nickname to "MyNickName"
*** Welcome to the Internet Relay Network MyNickName (from
localhost.localdomain)
*** If you have not already done so, please read the new user information with
/HELP +NEWUSER
*** Your host is localhost.localdomain[localhost.localdomain/6667], running
version +2.8/hybrid-6.3.1
*** Your host is localhost.localdomain[localhost.localdomain/6667], running
version +2.8/hybrid-6.3.1
*** This server was created Tue Jun 4 2002 at 16: 59:45 EDT
*** umodes available oOiwszcrkfydnxb, channel modes available biklmnopstve
*** WALLCHOPS PREFIX=(ov)@+ CHANTYPES=#& MAXCHANNELS=20 MAXBANS=25 NICKLEN=9
+TOPICLEN=120 KICKLEN=90 NETWORK=EFnet CHANMODES=b,k,l,imnpst MODES=4 are
supported by
+this server
*** There are 0 users and 2 invisible on 1 servers
*** 1 channels have been formed
*** This server has 2 clients and 0 servers connected
*** Current local  users:  2  Max: 2
*** Current global users:  2  Max: 2
*** Highest connection count:  1 (1 clients) (2 since server was (re)started)
*** - localhost.localdomain Message of the Day -
*** - This is an IRC server. Authorized users only.              ← MOTD
*** Mode change "+i" for user MyNickName by MyNickName          ← Invisible mode
*** MyNickName (~toto@172.16.238.129) has joined channel #mils  ← JOIN confirmation
*** #mils 1102276196
*** Mode for channel #mils is "+tn"                             ← Channel mode
*** #mils 1102276196
#mils      MyNickNam H   ~toto@172.16.238.129 (*Unknown*)      ← toto
#mils      lETUSENtC H@  pSaPvz@172.16.238.128 (OJnsKzIPB)     ← trojan (unpacked.ExE)

[1] 15:00 MyNickNam (+i) on #mils (+nt) * type /help for help
_
```

Figure 30: ircII IRC client screen on channel #mils

5.4 Exploring the malware behavior on the backdoor port

Section 4.2 described the static code analysis of *PassLogin*, thought to be associated with the remote access backdoor on port 2200. This section describes the behavioral analysis of *PassLogin*, with the specific objective of finding the password needed to interact with the trojan.

This exercise can be performed on *Victoom*. Launch OllyDbg as before and select **File → Attach**. Highlight the process *unpacked.ExE* from the list of running processes displayed and click Attach. OllyDbg determines the memory address of the running process from the kernel and disassembles the program found in memory rather than

reading it from disk as before. Also, the OllyDbg status line in the lower right now reads *Running* whereas it previously read *Paused.* This indicates that the process is running.

Select **View → Breakpoints** to confirm that the breakpoint set earlier in Section 4.2.5 is active, as shown in Figure 31.

Figure 31: OllyDbg breakpoints

Launch a command shell. Type `nc 127.0.0.1 2200` to start netcat and connect to port 2200 on the local system. The characters *#:* appear on a new line. Type an arbitrary string, such as *aaaa bbbb* and then Enter. Note that nothing interesting happens. Type *ccccccccc* and press Enter. The breakpoint at 0x40d611 causes OllyDbg to pause the program as shown in Figure 32. The stack (seen in the lower right of the CPU window) at address 0x22cdc0 contains a pointer to the string *ccccccccc*. The pointer value is 0x41e2b0, which is the address of the first character of the string.

Press F8 to single-step through the program one instruction at a time, stepping over any CALL instructions. Press F8 repeatedly until the program advances to 0x40d64b as shown in Figure 33. Note again the contents of the program stack. The previous string *ccccccccc* is still on the stack. The eight-character string *KZLPLKDf* has been pushed on the stack. This string is presumed to be the password. The stack also contains pointers to two strings noted in the Section 3.5. Note that the password is a substring of these longer strings. The general structure of these longer strings is unknown.

Press Ctrl-F2 to stop and reload the process in OllyDbg. Click Yes to allow OllyDbg to terminate the process. Use netcat as before in the same command window to connect to the backdoor. Type the same first line as before and press Enter. Then type the presumed password *KZLPLKDf* and press Enter. The text-based dialog box containing the string *Enter command number:* now appears as shown in Figure 34. Note that this string was not reported by either BinText or OllyDbg.

Type 1 and press Enter. The trojan echoes back our string *ccccccccc* as shown in Figure 35. This string was evidently logged elsewhere by *msrll.exe*. The attacker may have included this feature as a user convenience for fast repetition of manually-entered commands to perform some specific need, for instance as part of a network attack or file transfer operation. This concludes the malware analysis.

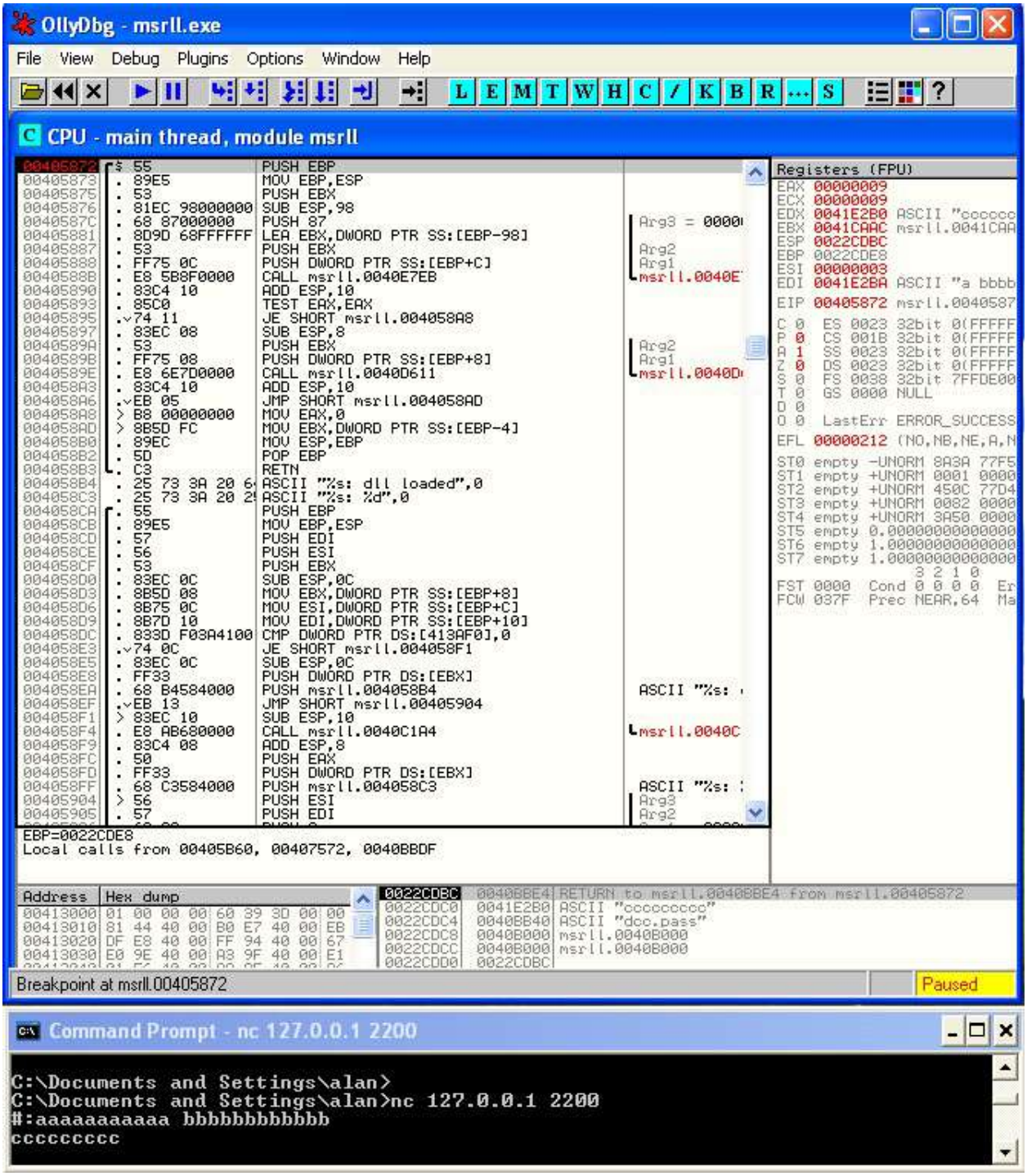

Figure 32: Execution paused at the *Compare* subroutine

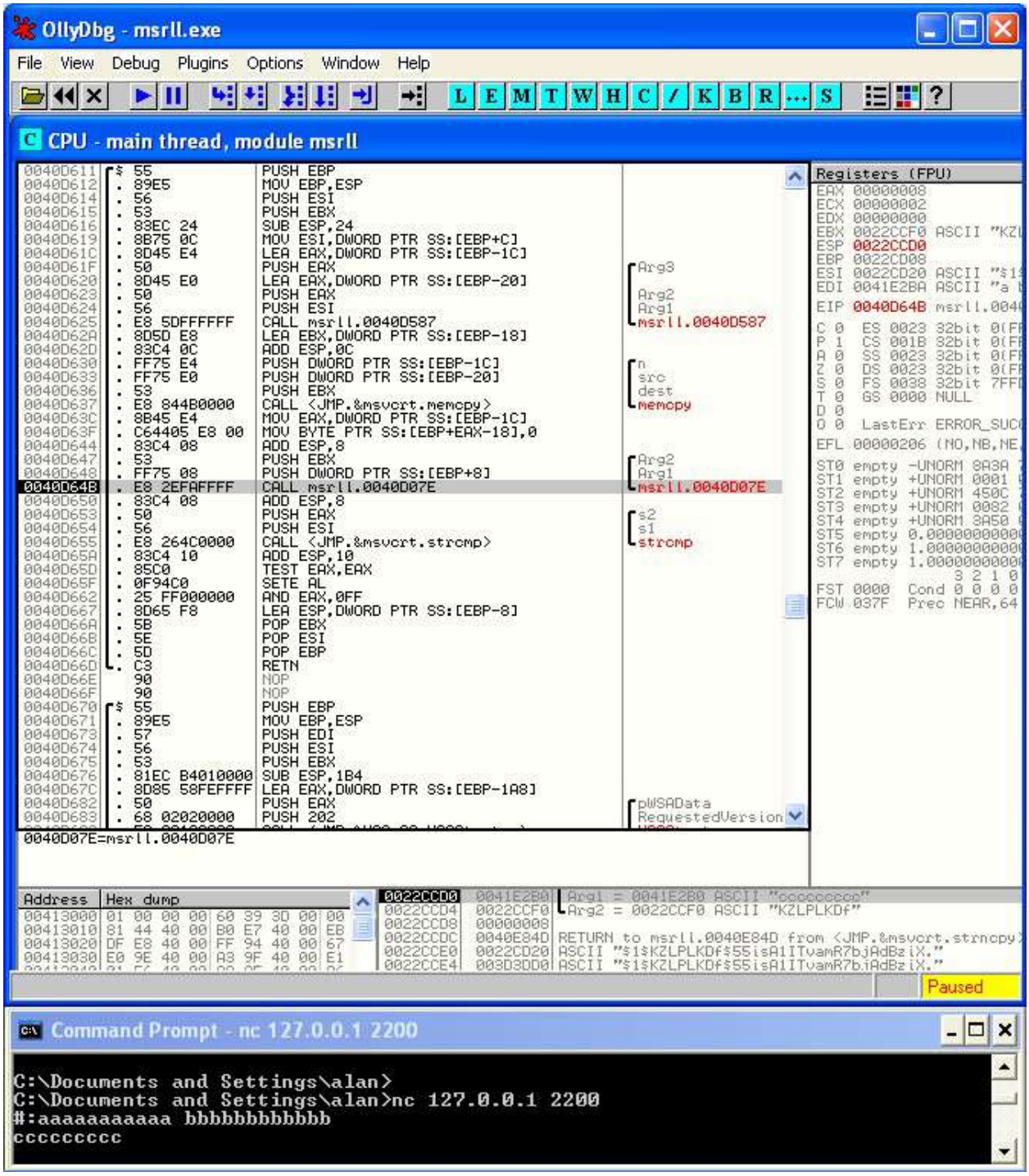

Figure 33: Trojan password for backdoor port revealed in debugger

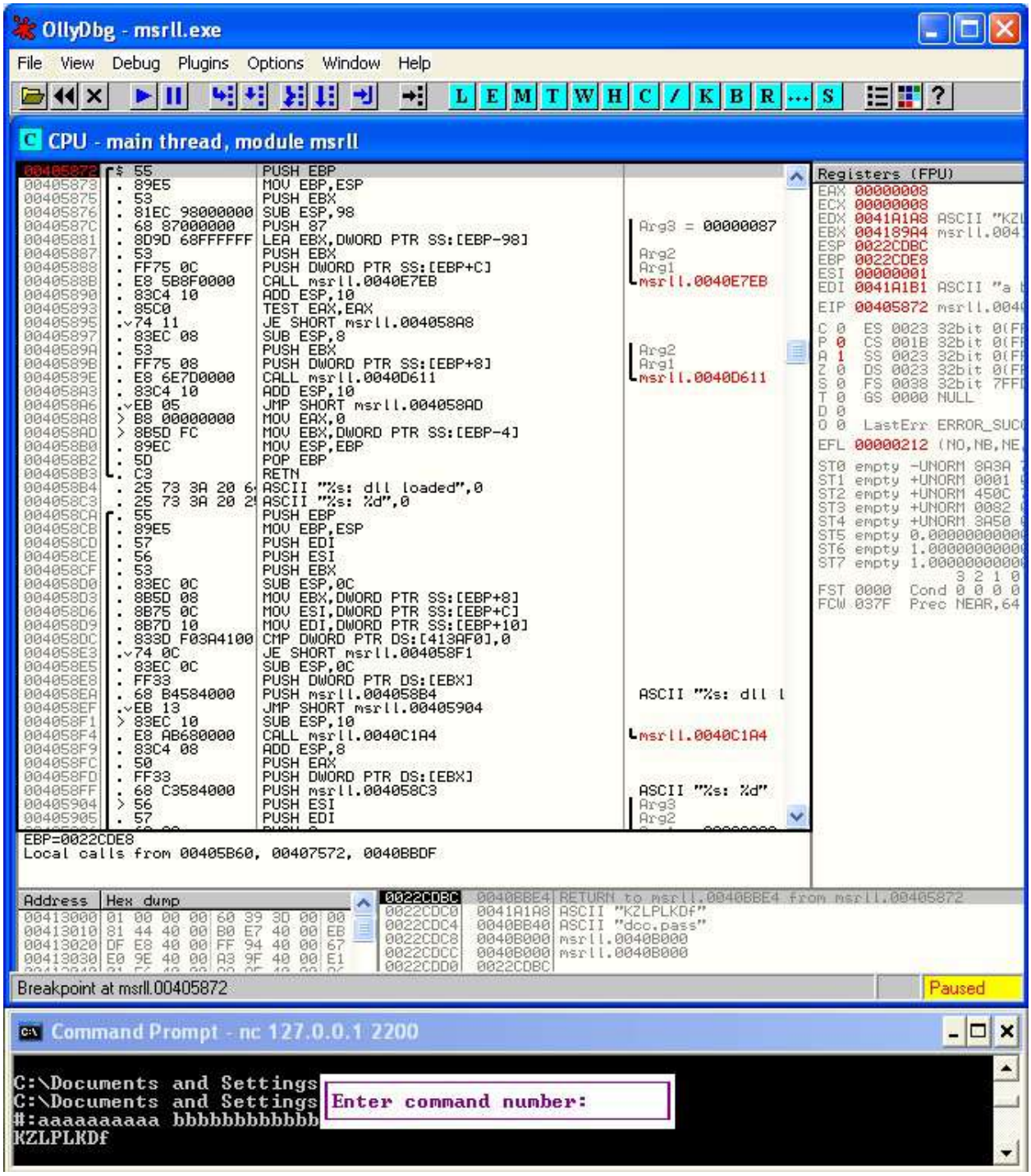

Figure 34: Trojan response using the presumed password

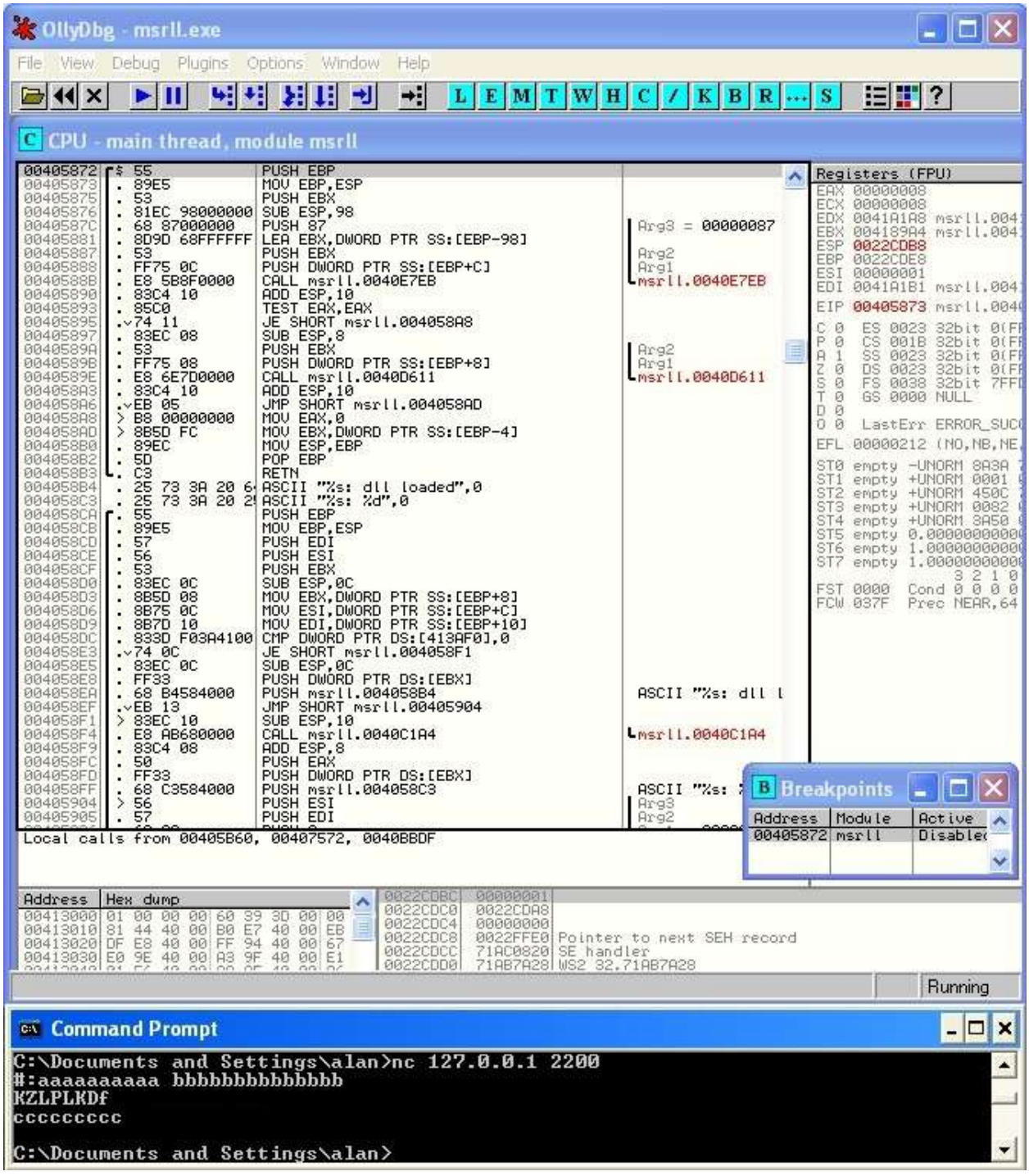

Figure 35: Trojan response to command number 1

6 Analysis Wrap-Up

Msrll.exe infects Win32 systems, including Windows 95, 98, NT, 2000, XP and Server 2003 platforms. This trojan is presumably distributed through email, IRC or P2P networks. *Msrll.exe* has been packed with AsPack to reduce file size and hamper

reverse engineering with disassembler and debugger tools. *Msrll.exe* installs itself in the `Windows\system32` folder. It configures registry keys to ensure that it runs when the system starts. On some platforms, a user may have to log in to start the trojan. *Msrll.exe* uses the strong encryption library RSAENH.DLL. The trojan stores what appear to be encryption keys or encoded data in the file *jtram.conf*.

Msrll.exe has been seen in the lab to connect to the host *collective7.zxy0.com* on the IRC channel #*mils*. The trojan provides an IDENT response for IRC servers. The attacker presumably controls #mils channel as a means to simultaneously control all infected systems. *Msrll.exe* allows backdoor remote access to infected systems by telnet or netcat connection to TCP port 2200. *Msrll.exe* appears to be designed to store and distribute software, photographs and other (contraband) data. *Msrll.exe* may also be designed to participate in distributed denial of service (DDOS) attacks (e.g., jolt2, smurf) on targets for financial or other gain. This behavior was not seen in the lab.

User awareness training is essential to stop the spread of *msrll.exe*. Site acceptable use policies should prohibit users from downloading software from unknown or untrusted sources. *Msrll.exe* is likely disguised as a benign file (e.g., “the dancing pigs” screensaver [37]) and distributed by attachment to email, downloaded from the Web or shared through IRC or peer-to-peer networks. It does not appear to be self-propagating.

Antivirus software can detect and remove *msrll.exe. Msrll.exe* can also be manually removed by deleting the appropriate registry keys and files. Antivirus software does not however detect *msrll.exe* when unpacked, nor can it protect against any malware that has not already been included in the antivirus signature database (*zero-day* exploits).

Host-based firewalls (e.g., ZoneAlarm [38]) configured to block outbound TCP ports 6667, 8080, 9999 (IRC) and inbound TCP port 2200 (telnet, netcat) will isolate infected systems and prevent them from acting in DDOS attacks. Perimeter firewalls with a deny-all policy or specific deny rules for these ports will achieve the same effect. Such sites and systems will however suffer degraded network and system performance.

Infected systems can be identified locally by the presence of the files *msrll.exe* and *jtram.conf*, by finding *msrll.exe* in the Task Manager process list, or by finding *msrll.exe* listening on TCP ports 113 or 2200 in netstat or TCPview. Remote detection using intrusion detection systems (IDS) or network port-scanning tool (e.g., nmap [39]) may be more effective for large or distributed sites provided that the IDS can handle the bandwidth at the network sensor. Port scanners that simply identify systems listening on both TCP ports 113 and 2200 as being infected may lead to false positives.

7 References

1 SANS Institute. “GIAC Reverse Engineering Malware (GREM).” GIAC Individual Subject Certifications. 2005. 5 Jan 2005. <http://www.giac.org/subject_certs.php#grem>.

2 Products -- VMware Workstation 4.5. 2005. VMware, Inc. 5 Jan 2005. <http://www.vmware.com/products/desktop/ws_features.html>.

3 Rivest, R. “RFC 1321 – The MD5 Message Digest Algorithm.” Network Working Group Request for Comments. April 1992. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc1321.html>.

4 Walker, John. “MD5 Command Line Message Digest Utility”. 15 April 2003. 5 Jan 2005. <http://www.fourmilab.ch/md5/>.

5 Kaminsky, Dan. “MD5 to be Considered Harmful Someday” 6 Dec 2004. 5 Jan 2005. <http://www.doxpara.com/md5_someday.pdf>.

6 Symantec Corporation. “Backdoor.IRC.Bot” 27 Oct 2003. 5 Jan 2005. <http://securityresponse.symantec.com/avcenter/venc/data/backdoor.irc.bot.html>.

7 Google Home Page. 2005. Google, Inc. 5 Jan 2005. <http://www.google.com>.

8 ASPACK SOFTWARE - Best Choice Compression and Protection Tools for Software Developers. 2005. 5 Aspack Software. Jan 2005. < http://aspack.com/aspack.html>.

9 Aaron’s Homepage. 5 Jan 2005. <http://www.exetools.com/unpackers.htm>.

10 Microsoft Corporation. “Run and RunOnce Registry Keys.” MSDN Library. Oct 2004. 5 Jan 2005. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/setupapi/setup/run_and_runonce_registry_keys.asp>.

11 Microsoft Corporation. “Registry Functions.” MSDN Library. Dec 2004. 5 Jan 2005. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/regcreatekeyex.asp>.

12 Microsoft Corporation. “Understand Common Virus Attacks Before They Strike to Better Protect Your Apps.” MSDN Magazine. May 2003. 5 Jan 2005 <http://msdn.microsoft.com/msdnmag/issues/03/05/virushunting/>

13 Microsoft Corporation. “Service Functions.” MSDN Library. Dec 2004. 5 Jan 2005. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/createservice.asp.>

14 Microsoft Corporation. “strcmp, wcscmp, _mbscmp” MSDN Library. 2005. 5 Jan 2005. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vccore98/HTML/_crt__stricmp.2c_._wcsicmp.2c_._mbsicmp.asp>.

15 Microsoft Corporation. “Chapter 7: Controlling Program Flow.” Programmer's Guide Microsoft® MASM Assembly-Language Development System Version 6.1 For MS-DOS® and Windows Operating Systems. 1992. pp. 161–84. 5 Jan 2005. <http://webster.cs.ucr.edu/AsmTools/MASM/MASMDoc/ProgrammersGuide/Chap_07.htm>

16 Intel Corporation. IA-32 Intel® Architecture Software Developer’s Manual Volume 2B: Instruction Set Reference, N-Z. 2004. p. 4–141. 5 Jan 2005. <ftp://download.intel.com/design/Pentium4/manuals/25366714.pdf>

17 Hyde, Randall. “4.3, Segments on the 80x86.” The Art of Assembly Language Programming. 30 Sep 1996. 5 Jan 2005. <http://cs.smith.edu/~thiebaut/ArtOfAssembly/CH04/CH04-1.html#HEADING1-2>.

18 Hyde, Randall. “4.5, Segment Registers on the 80x86.” The Art of Assembly Language Programming. 30 Sep 1996. 5 Jan 2005. <http://cs.smith.edu/~thiebaut/ArtOfAssembly/CH04/CH04-1.html#HEADING1-12>.

19 Hyde, Randall. “4.5, Segment Registers on the 80x86.” The Art of Assembly Language Programming. 30 Sep 1996. 5 Jan 2005. <http://cs.smith.edu/~thiebaut/ArtOfAssembly/CH04/CH04-1.html#HEADING1-102>.

20 Hyde, Randall. “6.3.5, The PUSH and POP Instructions.” The Art of Assembly Language Programming. 30 Sep 1996. 5 Jan 2005. <http://cs.smith.edu/~thiebaut/ArtOfAssembly/CH06/CH06-1.html#HEADING1-160>

21 DataRescue sa/nv. “Graphing in IDA.” 28 Jan 2003. 5 Jan 2005. <http://www.datarescue.com/idabase/graphs/index.htm>.

22 Free Software Foundation, Inc. “grep – GNU Project – Free Software Foundation (FSF).” 28 Dec 2002. 5 Jan 2005. <http://www.gnu.org/software/grep/grep.html>.

23 Free Software Foundation, Inc. “bash – GNU Project – Free Software Foundation (FSF).” 6 Nov 2003. 5 Jan 2005. <http://www.gnu.org/software/bash/bash.html>

24 Garrels, Machtelt. “9.1. The for loop.” Bash Guide for Beginners. 6 Dec 2004. 5 Jan 2005. <http://www.tldp.org/LDP/Bash-Beginners-Guide/html/Bash-Beginners-Guide.html#sect_09_01>.

25 Vasudevan, Alavoor. “Vim Color Editor HOW-TO (Vi Improved with syntax color highlighting).” 14 May 2003. 5 Jan 2005. < http://www.faqs.org/docs/Linux-HOWTO/Vim-HOWTO.html>.

26 The Internet Corporation for Assigned Names and Numbers. “Port Numbers.” 15 Nov 2004. 5 Jan 2005. <http://www.iana.org/assignments/port-numbers>

27 St. Johns, M. “RFC 1413 – Identification Protocol.” Network Working Group Request for Comments. Feb 1993. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc1413>.

28 Postel, J. “RFC 792 – Internet Control Message Protocol.” The Internet Engineering Task Force Network Working Group. Sep 1981. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc792.html>.

29 Freed, N., Borenstein, N. “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies” (RFC 2045). The Internet Engineering Task Force Network Working Group. Nov 1996. p. 24. 5 Jan 2005 <http://www.ietf.org/rfc/rfc2045.txt>

30 Linn, J. “Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures.” (RFC 1421) The Internet Engineering Task Force Network Working Group. Feb 1993. p. 13. 5 Jan 2005 <http://www.ietf.org/rfc/rfc1421.txt>

31 The Internet Corporation for Assigned Names and Numbers. “Port Numbers.” 15 Nov 2004. 5 Jan 2005. <http://www.iana.org/assignments/port-numbers>

32 Wikimedia Foundation, Inc. “Botnet.” Wikipedia. 27 Dec 2004. 5 Jan 2005. <http://en.wikipedia.org/wiki/Botnet>.

33 Kalt, C. “RFC 2812 - Internet Relay Chat: Client Protocol.” The Internet Engineering Task Force Network Working Group. April 2000. Pp. 10–11. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc2812.html>.

34 Kalt, C. “RFC 2812 - Internet Relay Chat: Client Protocol.” The Internet Engineering Task Force Network Working Group. April 2000. Pp. 43–52. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc2812.html>.

35 Kalt, C. “RFC 2812 - Internet Relay Chat: Client Protocol.” The Internet Engineering Task Force Network Working Group. April 2000. p. 16. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc2812.html>

36 Kalt, C. “RFC 2811, Internet Relay Chat: Channel Management.” The Internet Engineering Task Force Network Working Group. April 2000. p. 4–5. 5 Jan 2005. <http://www.faqs.org/rfcs/rfc2811.html. 9 Dec. 2004>.

37 Schneier, B. "Security in the Real World: How to Evaluate Security Technology," Computer Security Journal, v 15, n 4, 1999, pp. 1-14. 5 Jan 2005. <http://www.schneier.com/essay-031-ft.txt>.

38 Zone Labs LLC. “Zone Labs: Download & Buy.” 2005. 5 Jan 2005. <http://www.zonelabs.com/store/content/catalog/products/sku_list_za.jsp>

39 Fyodor. “Nmap – Free Security Scanner for Network Exploration & Security Audits.” 2004. 5 Jan 2005. <http://www.insecure.org/nmap/index.html>

40 Mills, David. “NTP: The Network Time Protocol.” 14 Dec 2004. 5 Jan 2005. <http://www.ntp.org>.

41 VMware Inc. “Try VMware Workstation 4.” 2004. 5 Jan 2005. <http://www.vmware.com/vmwarestore/newstore/wkst_eval_login.jsp>

42 VMware Inc. VMware Workstation 4.5.2 User’s Manual. 2004. 5 Jan 2005. <http://www.vmware.com/support/pubs/ws_pubs.html>.

43 VMware Inc. “Starting a Virtual Machine on a Linux Host.” VMware Workstation 4.5.2 User’s Manual. 2004. Pp. 78 – 83. 5 Jan 2005. <http://www.vmware.com/support/ws45/doc/running_startvm_lin_ws.html#1054201>

44 Microsoft Corporation. “How To Manage Environment Variables in Windows XP.” Microsoft TechNet. 15 July 2004. 5 Jan 2005. <http://support.microsoft.com/default.aspx?scid=kb;en-us;310519&sd=tech>.

45 VMware Inc. “Using Virtual Ethernet Adapters in Promiscuous Mode on a Linux Host.” VMware Workstation 4.5.2 User’s Manual. 2004. P. 243. 5 Jan 2005. <http://www.vmware.com/support/ws45/doc/network_promiscuous_ws.html#1062038>

46 Wikimedia Foundation, Inc. “Portable Executable.” Wikipedia. 9 Dec 2004. 5 Jan 2005. <http://en.wikipedia.org/wiki/Portable_Executable>.

47 Yuschuk, Oleh. “OllyDbg v1.10.” 2004. 5 Jan 2005. <http://home.t-online.de/home/OllyDbg/>

48 Microsoft Corporation. “Introduction.” Programmer's Guide Microsoft® MASM Assembly-Language Development System Version 6.1 For MS-DOS® and Windows Operating Systems. 1992. 5 Jan 2005. <http://webster.cs.ucr.edu/AsmTools/MASM/MASMDoc/ProgrammersGuide/Introduction.htm>

49 Russinovich, Mark. “Windows NT and Windows 2000 – Inside the Registry”. Microsoft TechNet. May 1999. 5 Jan 2005. <http://www.microsoft.com/technet/archive/winntas/tips/winntmag/inreg.mspx>.

50 Microsoft Corporation. “Description of the Microsoft Windows Registry”. 21 Dec 2004. 5 Jan 2005. <http://support.microsoft.com/kb/256986>.

51 Microsoft Corporation. “Microsoft Enhanced Cryptographic Provider, Windows 2000 Cryptographic Providers.” Microsoft TechNet. 9 Aug 1999. 5 Jan 2005. <http://www.microsoft.com/technet/security/topics/issues/wn2cspsp.mspx#EAAA>

Appendix A VMware Virtualization Software

VMware Workstation [2] allows virtual systems run independently and transparently share the real resources of the host system. Virtual systems can be configured with no external connections or allowed to communicate over virtual networks using networked file systems (e.g., NFS, CIFS/SMB shared folders), or other conventional network services (e.g., SSH, IRC, HTTP, etc). VMware offers a 30-day no-cost fully-functional evaluation of their product.

A virtual lab is preferred for malware analysis. It is easier to manage than the equivalent collection of physical systems, and it requires less total hardware, physical space and power. Note that logged events from either virtual system can be used without adjusting for clock skew, since all timed events are based on the host system clock. Clock skew is a major issue when multiple physical systems are used, even when using clock synchronization protocols such as ntp. [40]

This appendix provides general and specific guidance to install, configure and test the virtual laboratory environment used to reverse engineering malware such as that presented in this paper.

These instructions apply to VMware Workstation version 4.5.2 build 8848. [41] The principle source of this information is the VMware Workstation 4.5 User's Manual. [42] The hostnames *Vmhost, Victoom* and *Viroom* refer to the VMware host, Windows XP virtual system and Linux Red Hat 9 virtual system, respectively. More information on these systems can be found in the body of this paper.

A.1 Installing and Configuring VMware on a Linux Host

VMware Workstation is installed first on the host system. Virtual systems are then installed using the VMware application. For *Vmhost*, follow the procedure in the section *Installing VMware Workstation 4 on a Linux Host* beginning on page 36 of the VMware Users Manual.

Launch the VMware application by opening a command shell as the root user and typing `vmware &`.

A.2 Installing Tools on Victoom and Viroom

To install the virtual systems *Victoom* and *Viroom,* follow the procedures in the section *Installing a Guest Operating System and VMware Tools.* [43] Add *Victoom* and *Viroom* to the VMware Favorites List as described in the manual. Be sure to install the VMware tools on all virtual machines as instructed.

Download and install all the packages in Table 2 on *Victoom*. Certain tools (e.g., FileMon) install into a directory. Each may be installed into separate directory or all such tools may all be installed into a single directory (e.g., `c:\bin`). Add the directory (or

directories) to the system path [44] by selecting **Start → Settings → Control Panel → System**. Click the Advanced tab and the Environment Variables button. Select Path in the System Variables list and click Edit. Press the End key and type `;c:\bin` and press Return. Click OK twice to close the properties window. If individual directories are used, add each directory to the path in the same manner.

A.3 Switching Between Virtual Machines

With VMware running on *Vmhost*, select *Viroom* from the Favorites list and click the Power On button to start the virtual machine. Click anywhere inside the virtual machine window to give the virtual machine control of your mouse and keyboard. Virtual machines run faster in full screen mode. To do so, select **View → Full Screen** from the VMware tool bar at the top of the screen. To get out of full screen mode, press the Ctrl and Alt keys at the same time. The virtual machine now appears inside a VMware Workstation window. Now select *Victoom* from the Favorites list and click Power on. Both machines are now running in the virtual environment.

A.4 Virtual Networking

Configure a host-only network as described on page 38 the VMware Users Manual, using the network configuration settings described in Table 1. As an added precaution to ensure that network traffic is contained within the virtual environment, disable the host's connection to the virtual network *Vmnet1* as follows: At a command shell on *Vmhost* as root, type `ifconfig Vmnet1 down`.

At the shell prompt on *Viroom*, type `ping 172.16.238.128` to verify the virtual network *Vmnet1* between it and *Victoom*. Figure 36 shows the ping replies that indicate the network is correct. If replies are not received, resolve the problem before proceeding.

At a command prompt on *Victoom*, type `ping 172.16.238.129` to verify the network to *Viroom*. If any replies time out, debug and resolve the problem before proceeding.

```
[root@Viroom root]# ping 172.16.238.128
PING 172.16.238.128 (172.16.238.128) 56(84) bytes of data.
64 bytes from 172.16.238.128: icmp_seq=1 ttl=150 time=0.484 ms
64 bytes from 172.16.238.128: icmp_seq=2 ttl=150 time=0.362 ms
64 bytes from 172.16.238.128: icmp_seq=3 ttl=150 time=0.360 ms
64 bytes from 172.16.238.128: icmp_seq=4 ttl=150 time=0.360 ms

--- 172.16.238.128 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3997ms
rtt min/avg/max/mdev = 0.360/0.392/0.484/0.054 ms

[root@Viroom root]#
```

Figure 36: Confirming network activity using ping

A.5 Performance Considerations

The *Performance Tuning* section beginning on page 307 of the VMware Users Manual recommends a number of settings for both the host and guest systems to improve performance for the virtual workstation software environment.

Allocation of memory to the virtual systems tends to have the greatest impact. For best results, allocate as much memory to each virtual system as would be needed for a real system. The VMware host will therefore need enough physical RAM for itself and the sum of that needed for the virtual systems. Stopping all unneeded processes on the VMware host and virtual systems will also improve performance.

VMware recommends pre-allocating disk space when installing the virtual system to gain additional performance, particularly when the application will be reading or writing frequently to the disk. This is a decision made during the installation of the virtual system, as explained on page 75 of the Users Manual.

A.6 Promiscuous-mode Virtual Interfaces with VMware

By default, a network interface will only report to its host the network activity that is either broadcast to all hosts or specifically addressed to itself. Network interfaces can be set to promiscuous-mode operation that allows it to report any and all network activity to the host. This functionality is required to allow *Viroom* to capture all network activity .

There are two options to enable promiscuous-mode interfaces in VMware. One option is to simply run the VMware application as the root user. Running applications as root violates the security principle of least privilege*.* A second option is to change the ownership of network devices on the host system to allow specified regular users to set interfaces into promiscuous mode. This option is highly recommended and described in detail in the documentation. [45]

Appendix B Strings in *unpacked.ExE*

This section lists the majority of the strings extracted from *unpacked.ExE* by BinText. Certain irrelevant strings are not included in this list.

Mem. Pos	Text
0040004D	!This program cannot be run in DOS mode.
00400088	[AspackDie!]
00400178	.text
004001A0	.data
004001F0	.idata
00400218	.aspack
00400240	.adata
00401326	?insmod
0040132E	?rmmod
00401335	?lsmod
00401399	%s: <mod name>
004013A8	%s: mod list full
004013BA	%s: err: %u
004013C6	mod_init
004013CF	mod_free
004013D8	%s: cannot init %s
004013EB	%s: %s loaded (%u)
004013FE	%s: mod allready loaded
00401416	%s:%s err %u
004015B5	%s:%s not found
004015C5	%s: unloading %s
004016AE	[%u]: %s hinst:%x
00401712	unloading %s
004017A0	%s: invalid_addr: %s
004017B5	%s%s [port]
004018E8	finished %s
00401A40	%s <ip> <port> <t_time> <delay>
00401B32	sockopt: %u

Mem. Pos	Text
00401B3E	sendto err: %u
00401B4D	sockraw: %u
00401B59	syn: done
00401FBC	%s <ip> <duration> <delay>
00402096	sendto: %u
004020A2	jolt2: done
00402260	%s <ip> <p size> <duration> <delay>
00402356	Err: %u
0040235E	smurf done
004025DE	&err: %u
00402753	?ping
00402763	?smurf
0040276A	?jolt
00402820	PONG :%s
0040283A	0h (@
0040299D	%s!%s@%s
00402B3D	%s!%s
00402BD7	irc.nick
00402BE0	NICK %s
00402EEA	NETWORK=
00402FF8	irc.pre
004032E1	NICK %s
004036B0	irc.chan
00403A45	USERHOST %s
00403A52	logged into %s(%s) as %s
00403B99	Nick.pre
00403BAA	irc.user
00403BB3	irc.usereal

Mem. Pos	Text
00403BBF	irc.real
00403BC8	irc.pass
00403BE0	tsend(): connection to %s:%u failed
00403C20	USER %s localhost 0 :%s
00403C38	NICK %s
004040BF	PRIVMSG
00404100	trecv(): Disconnected from %s err:%u
0040446B	NOTICE
00404472	%s %s :%s
00404711	MODE %s -o+b %s *@%s
004047E7	MODE %s -bo %s %s
00404924	%s.key
00404AA8	sk#%u %s is dead!
00404ABA	s_check: %s dead? pinging...
00404AD7	PING :ok
00404B00	s_check: send error to %s disconnecting
00404B28	expect the worst
00404B39	s_check: killing socket %s
00404B54	irc.knick
00404B5E	jtr.%u%s.iso
00404B6B	ison %s
00404B74	servers
00404B7C	s_check: trying %s
0040505A	MODE %s %s
004055A3	mode %s +o %s
004055B2	akick
004055B8	mode %s +b %s %s
004055CA	KICK %s %s
00405781	Set an irc sock to preform %s command on
004057BC	to view current sockets, then
004057DC	%cdccsk
004058B4	%s: dll loaded
004058C3	%s: %d

Mem. Pos	Text
004059E1	said %s to %s
004059EF	usage: %s <target> "text"
00405A74	%s not on %s
00405A81	usage: %s <nick> <chan>
00405B1B	PASS
00405B20	%s logged in
00405BA2	sys: %s bot: %s
00405BB2	preformance counter not avail
00405C2B	usage: %s <cmd>
00405C3B	%s free'd
00405C45	unable to free %s
00405CAD	later!
00405CB4	unable to %s errno:%u
00405D40	service:%c user:%s inet connection:%c contype:%s reboot privs:%c
004060D4	host: %s ip: %s
00406269	capGetDriverDescriptionA
00406292	cpus:%u
004062A0	WIN%s (u:%s)%s%s mem:(%u/%u) %u%% %s %s
004065CB	%s: %s (%u)
00406708	%s %s
00406754	%s bad args
004067DA	akick
004067E8	%s[%u] %s
004067F2	%s removed
004067FD	couldnt find %s
0040680D	%s added
00406816	%s allready in list
0040682A	usage: %s +/- <host>
004069EB	*jtram.conf*
004069FF	jtr.home
00406A0E	%s: possibly failed: code %u
00406A2B	%s: possibly failed

Mem. Pos	Text
00406A3F	%s: exec of %s failed err: %u
00406CBC	jtr.id
00406CC3	%s: <url> <id>
00406ED7	IREG
00406EDD	CLON
00406EE3	ICON
00406EF8	WCON
00406F40	#%u [fd:%u] %s:%u [%s%s] last:%u
00406F63	|\=> [n:%s fh:%s] (%s)
00406F82	|---[%s] (%u) %s
00406F96	| |-[%s%s] [%s]
00406FAD	|=> (%s) (%.8x)
0040716E	B$PRhco@
00407360	%s <pass> <salt>
004073C8	%s <nick> <chan>
0040748B	PING %s
004074C9	mIRC v6.12 Khaled Mardam-Bey
004074E7	VERSION %s
0040751C	dcc.pass
00407525	temp add %s
00407675	%s opened (%u)
004076A0	%u bytes from %s in %u seconds saved to %s
004076CB	(%s %s): incomplete! %u bytes
004076E9	couldnt open %s err:%u
00407700	(%s) %s: %s
0040770C	(%s) urlopen failed
00407720	(%s): inetopen failed
00407BE4	no file name in %s
00407DDB	%s created
00407E49	%s %s to %s Ok
00407EE0	%0.2u/%0.2u/%0.2u %0.2u:%0.2u %15s %s
00407F09	%s (err: %u)
00408085	err: %u

Mem. Pos	Text
004080F8	%s %s :ok
00408165	unable to %s %s (err: %u)
004081F5	%-16s %s
00408200	%-16s (%u.%u.%u.%u)
00408489	[%s][%s] %s
00408595	closing %u [%s:%u]
004085A8	unable to close socket %u
004087E2	using sock #%u %s:%u (%s)
004087FD	Invalid sock
0040880B	usage %s <socks #>
004088D7	leaves %s
004088E1	:0 * * :%s
00408A96	joins: %s
00408B82	ACCEPT
00408B89	resume
00408B90	err: %u
00408B99	DCC ACCEPT %s %s %s
00408BAE	dcc_resume: cant find port %s
00408BD1	dcc.dir
00408BD9	%s\%s\%s\%s
00408BE5	unable to open (%s): %u
00408BFD	resuming dcc from %s to %s
00408C19	DCC RESUME %s %s %u
0040934E	?clone
00409355	?clones
0040935D	?login
00409364	?uptime
0040936C	?reboot
00409374	?status
0040937C	?jump
00409382	?nick
00409388	?echo
0040938E	?hush

Mem. Pos	Text
00409394	?wget
0040939A	?join
004093A9	?akick
004093B0	?part
004093B6	?dump
004093C6	?md5p
004093CC	?free
004093D7	?update
004093DF	?hostname
004093EE	?!fif
004093FE	?play
00409404	?copy
0040940A	?move
00409415	?sums
00409423	?rmdir
0040942A	?mkdir
00409436	?exec
00409440	?kill
00409446	?killall
0040944F	?crash
0040946E	?sklist
00409476	?unset
0040947D	?uattr
00409484	?dccsk
00409490	?killsk
00409499	VERSION*
004094AE	IDENT
004096BE	%ud %02uh %02um %02us
004096D4	%02uh %02um %02us
004099E0	*jtram.conf*
004099EB	jtr.*
004099F5	DiCHFc2ioiVmb3cb4zZ7zWZH1oM=
00409A16	conf_dump: wrote %u lines

Mem. Pos	Text
0040A270	get of %s incomplete at %u bytes
0040A2B0	get of %s completed (%u bytes), %u seconds %u cps
0040A2F0	error while writing to %s (%u)
0040A65C	chdir: %s -> %s (%u)
0040A750	dcc_wait: get of %s from %s timed out
0040A790	dcc_wait: closing [#%u] %s:%u (%s)
0040A9F0	%4s #%.2u %s %ucps %u%% [sk#%u] %s
0040AA30	%u Send(s) %u Get(s) (%u transfer(s) total) UP:%ucps DOWN:%ucps Total:%ucps
0040ACD0	send of %s incomplete at %u bytes
0040AD10	send of %s completed (%u bytes), %u seconds %u cps
0040AF50	cant open %s (err:%u) pwd:{%s}
0040AF70	DCC SEND %s %u %u %u
0040B77B	exec: Error:%u pwd:%s cmd:%s
0040BB40	dcc.pass
0040BB49	bot.port
0040BB52	%s bad pass from "%s"@%s
0040BCC9	%s: connect from %s
0040BD33	jtr.bin
0040BD3B	*msrll.exe*
0040BD45	jtr.home
0040BD57	jtr.id
0040BD63	irc.quit
0040BD6E	servers
0040BD80	collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy0.com:8080
0040BDCA	irc.chan
0040BDD3	#mils
0040BDE0	1KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0
0040BE20	1KZLPLKDf$55isA1ITvamR7bjAdBziX.
0040C02F	SSL_get_error

Mem. Pos	Text
0040C03D	SSL_load_error_strings
0040C054	SSL_library_init
0040C065	SSLv3_client_method
0040C079	SSL_set_connect_state
0040C08F	SSL_CTX_new
0040C09B	SSL_new
0040C0A3	SSL_set_fd
0040C0AE	SSL_connect
0040C0BA	SSL_write
0040C0C4	SSL_read
0040C0CD	SSL_shutdown
0040C0DA	SSL_free
0040C0E3	SSL_CTX_free
0040C263	kernel32.dll
0040C270	QueryPerformanceCounter
0040C288	QueryPerformanceFrequency
0040C2A2	RegisterServiceProcess
0040C2B9	*jtram.conf*
0040C5B1	irc.user
0040C5BA	%s : USERID : UNIX : %s
0040C6A4	QUIT :FUCK %u
0040C742	Killed!? Arrg! [%u]
0040C756	QUIT :%s
0040C7E8	SeShutdownPrivilege
0040C888	%s\%s
0040C88E	%s\%s\%s
0040C897	Rll enhanced drive
0040C8C0	software\microsoft\windows\currentversion\run
0040D010	./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
0040EA60	usage %s: server[:port] amount
0040EB33	%s: %s

Mem. Pos	Text
0040EB3E	%s %s %s <PARAM>
0040EB80	%s: [NETWORK|all] %s <"parm"> ...
0040EE20	USER %s localhost 0 :%s
0040EE38	NICK %s
0040F140	md5.c
0040F146	md != NULL
0040F8F1	buf != NULL
0040F99F	hash != NULL
0040FAC5	message digest
0040FAD4	abcdefghijklmnopqrstuvwxyz
0040FB00	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
0040FB40	12345678901234567890123456789012345678901234567890123456789012345678901234567890
0040FCE0	sprng
0040FD11	sprng.c
0040FD19	buf != NULL
0040FDBC	rc6.c
0040FDC2	skey != NULL
0040FDCF	key != NULL
0040FFD1	ct != NULL
0040FFDC	pt != NULL
004103C3	desired_keysize != NULL
00410430	ctr.c
00410436	ctr != NULL
00410442	key != NULL
0041044E	count != NULL
00410546	ct != NULL
00410551	pt != NULL
004106F0	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
0041077F	?456789:;<=
004107B7	!"#$%&'()*+,-./0123

Mem. Pos	Text
00410850	base64.c
00410859	outlen != NULL
00410868	out != NULL
00410874	in != NULL
00410B30	_ARGCHK '%s' failure on line %d of file %s
00410B8B	crypt.c
00410B93	name != NULL
00410D79	cipher != NULL
00410E70	hash != NULL
00410F7A	prng != NULL
004110F0	LibTomCrypt 0.83
00411102	Endianess: little (32-bit words)
00411123	Clean stack: disabled
00411139	Ciphers built-in:
0041114B	Blowfish
00411157	RC2
0041115E	RC5
00411165	RC6
0041116C	Serpent
00411177	Safer+
00411181	Safer
0041118A	Rijndael
00411196	XTEA
0041119E	Twofish
004111AA	CAST5
004111B3	Noekeon
004111BF	Hashes built-in:
004111D0	SHA-512
004111DB	SHA-384
004111E6	SHA-256
004111F1	TIGER
004111FA	SHA1
00411202	MD5

Mem. Pos	Text
00411209	MD4
00411210	MD2
00411218	Block Chaining Modes:
0041122E	CFB
00411235	OFB
0041123C	CTR
00411244	PRNG:
0041124A	Yarrow
00411254	SPRNG
0041125D	RC4
00411265	PK Algs:
0041126E	RSA
00411275	DH
0041127B	ECC
00411282	KR
00411289	Compiler:
00411293	WIN32 platform detected.
004112AF	GCC compiler detected.
004112CA	Various others: BASE64 MPI HMAC
00411313	/dev/random
00411430	Microsoft Base Cryptographic Provider v1.0
004114D2	bits.c
004114D9	buf != NULL
0041154A	prng != NULL
00411A10	-LIBGCCW32-EH-SJLJ-GTHR-MINGW32
004130B0	<ip> <total secs> <p size> <delay>
00413350	modem
00413358	Lan
0041335E	Proxy
00413390	m220 1.0 #2730 Mar 16 11:47:38 2004
004133D4	unable to %s %s (err: %u)
00413420	unable to kill %s (%u)
00413437	%s killed (pid:%u)

Mem. Pos	Text
00413470	AVICAP32.dll
0041347D	unable to kill %u (%u)
00413494	pid %u killed
004134A2	error!
004134A9	ran ok
004134B0	MODE %s +o %s
004134BF	set %s %s
00413600	Mozilla/4.0
0041360C	Accept: */*

Mem. Pos	Text
0041361C	<DIR>
0041362B	Could not copy %s to %s
00413643	%s copied to %s
00413653	0123456789abcdef
00413664	%s unset
0041366D	unable to unset %s
00413BA0	libssl32.dll
00413BAD	libeay32.dll
00413BE0	<die|join|part|raw|msg>

Appendix C An Introduction to OllyDbg and the Win32PE

C.1 The Windows Portable Executable

Wikipedia describes the Windows Portable Executable file format as follows [46] :

> The Portable Executable (PE) format is an executable file format used in 32-bit and 64-bit versions of Windows operating systems. The term "portable" refers to the format's portability across all 32-bit (and by extension 64-bit) Windows operating systems. The PE format is basically a data structure that encapsulates the information necessary for the Windows OS loader to manage the wrapped executable code. This includes dynamic library references for linking, API export and import tables, and resource management data. On NT operating systems, the PE format supports EXE, DLL, OBJ, and other file types.
>
> PE is a modified version of the Unix COFF file format. PE/COFF is an alternate term in Windows development.
>
> Microsoft migrated to the PE format with the introduction of the Windows NT and Windows 95/98/ME operating systems (a hallmark of the transition to 32-bit systems). The format has retained limited legacy support to bridge the gap between DOS-based and NT systems. For example, PE/COFF Headers still include an MS-DOS executable header (or "stub") that displays the simple message "This program cannot be run in MSDOS mode", or similar, as backwards-compatible error output. PE also continues to serve the changing Windows platform. Some extensions include the .NET PE format (see below) and a 64-bit version called PE+ (sometimes PE32+).

Program instructions and data are grouped into code sections having the same attributes. An arbitrary number of code sections may exist. Figure 6 shows the code sections for *msrll.exe*. The *text* code section (having read-only attributes) contains instructions, string constants and other data that does not change. The *data* code section contains declared variables whose size is known at compile time but whose values change as the program runs. The data section has read-write attributes.

C.2 OllyDbg

OllyDbg [47] is both a disassembler and debugger. OllyDbg allows users to view the program code in binary and assembly format, change instructions and data, and add comments directly to the code. OllyDbg is primarily used in this work to disassemble a compiled Win32 PE into assembly language. OllyDbg analyzes the code as it disassembles it to identify subroutines, text strings, external libraries and function calls and other objects. OllyDbg automatically adds comments for subroutine calls and code structures where possible. The Microsoft ASM (MASM) assembly language syntax will be used throughout this work because of its widespread use and availability of documentation (e.g., [48]).

Launch OllyDbg by selecting **Start → Run**, typing `OllyDbg.exe` and clicking OK. Click File → Open, browse to the malware directory in `c:\tmp`, select *unpacked.ExE* and click OK. Figure 37 shows the OllyDbg CPU window. The CPU window is organized in four quadrants. The top left quadrant contains the disassembled object code. The top right shows the CPU registers. The bottom left shows the contents of memory regions. The bottom right quadrant shows the processor stack. The status bar along the bottom of the window includes warning and activity messages. The Paused flag in the lower right corner shows that the debugger has loaded and analyzed the program and suspended its execution.

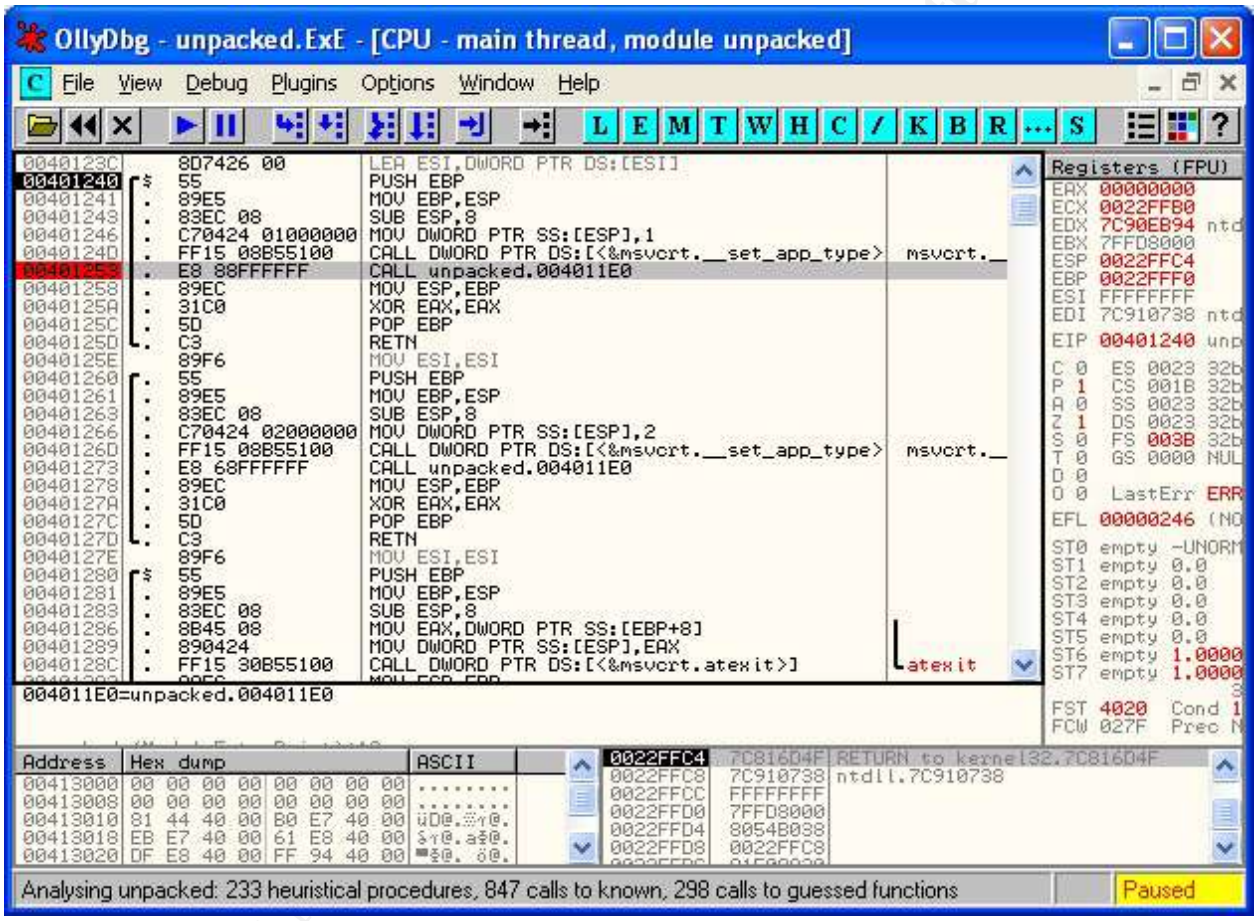

Figure 37: *unpacked.ExE* loaded and analyzed in OllyDbg

The top left quadrant of the CPU window is further divided into four columns. The first column is the memory position at run-time, the second is the object code (in hexadecimal) at that position, and the third is the disassembled representation of the object code for the Intel x86 processor. The fourth column is an area for comments. Vertical bars in the second column delineate subroutines. Vertical bars in the fourth column delineate subroutine calls and arguments.

In Figure 37, the address shown in white text on black background in the first column (0x401240) is the current value of the instruction pointer. The instruction at that address

will be executed next. The address shown with red background has a breakpoint set, so that the debugger with halt the program execution as the instruction pointer reaches that location. Breakpoints are set or cleared at whatever instruction is highlighted by pressing the F2 key. OllyDbg sets a breakpoint at the program entry point when the program is first loaded which keeps its execution under the user's control at all times.

The CPU register display allows the user to monitor the processor state as execution proceeds. Note that the instruction pointer register (EIP) contains 0x401240, corresponding to the instruction pointer location indicated in the code display (as explained in the preceding paragraph). The register display also shows the processor flags. Flags are set by the results of certain comparison and calculation operations and determine the program execution flow at jump instructions.

When a CALL is reached, the processor stores (pushes) the current address on the stack and then begins executing at the address specified in the CALL instruction. Execution continues sequentially in the subroutine until a RETN (return) instruction is reached. The processor then retrieves (pops) the address previously stored on the stack to return to the calling subroutine. Execution then resumes sequentially from the point where the CALL occurred.

Arguments to the subroutine, if any, are passed by pushing them on the stack, usually just before the CALL instruction. Within the called subroutine, they are popped off the stack as needed. OllyDbg labels the arguments passed to subroutines and functions as Arg1, Arg2 … Arg***N***. The integer suffix seems to increase by one for each argument passed. This labeling scheme is tool-specific. IDA Pro, for instance, labels arguments passed to subroutines as arg_0, arg_4 … arg_***N***. The integer suffix in IDA Pro seems to increase by 1 for each byte passed.

Left-click with the mouse anywhere within the four quadrants to show OllyDbg's context-sensitive command menu. To view the complete listing of disassembled code, left-click and select **Copy → Select all**, then left-click and select **Copy → To file**. Type `unpacked-ExE.txt` and click Save. The listing can be viewed with any text editor.

Appendix D RegShot Snapshot Comparison Report

The Windows Registry is a system-defined database where application and system software store and retrieve configuration information. [49], [50] The Registry keys are structured in a tree format, with top-level branches as shown in Figure 38. The key Software\Microsoft\Windows\CurrentVersion\Run in the HKEY_LOCAL_MACHINE (HKLM) and HKCU registry branches defines programs that start and run when a user logs in to the system. The values set for these keys are the path and file name of the program. Objects configured in this way do not show up in the Startup group of the Start menu and are difficult to locate unless you're acquainted with this registry key. As a result, these keys are frequently targeted by malware.

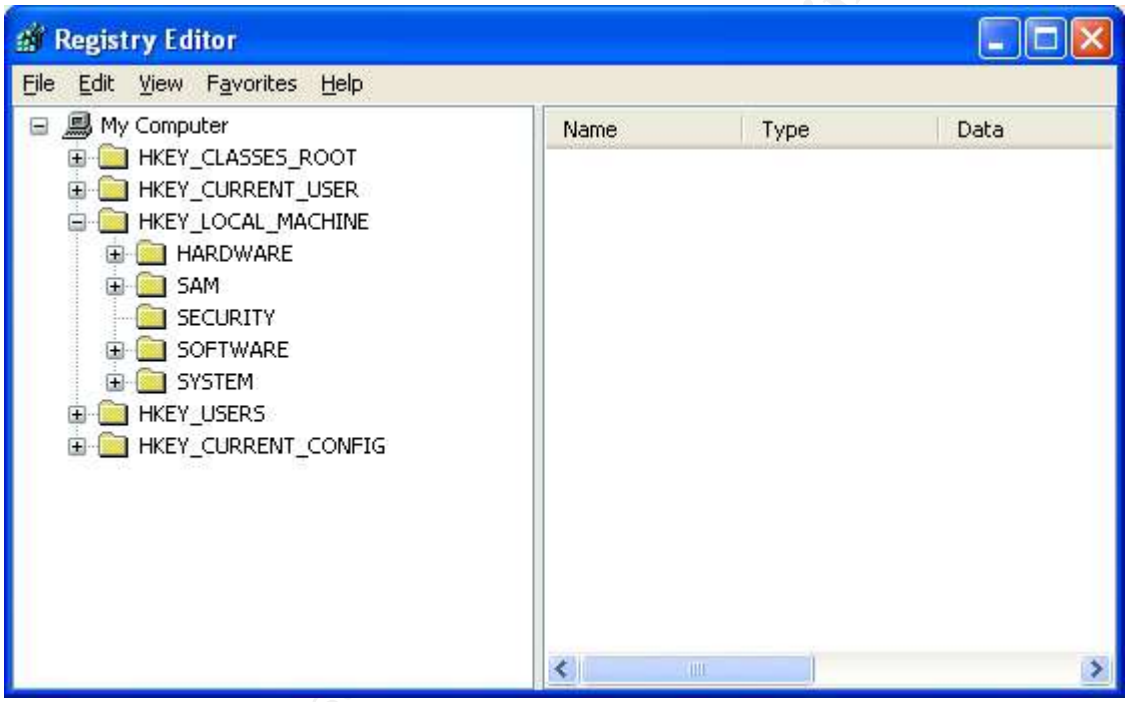

Figure 38: Top-level registry branches in *regedit*

Data not relevant to the malware analysis have been deleted from this report. Some additional unimportant information may remain.

REGSHOT LOG 1.61e5

Comments:

Datetime: 2004/9/16 07:55:41 , 2004/9/16 08:00:17

Computer:VICTOOM **,** VICTOOM

Username:

Relevant Keys deleted:0 Total Keys deleted:4

<no relevant keys deleted>

Relevant Keys added:4 Total Keys added:14

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

Relevant Values deleted:2 Total Values deleted:23

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\kmixer\Enum\0: SW\{b7eafdc0-a680-11d0-96d8-00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\0: SW\{b7eafdc0-a680-11d0-96d8-00aa0051e51d}\{9B365890-165F-11D0-A195-0020AFD156E4}

Relevant Values added:14 Total Values added:57

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

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ErrorControl: 0x00000002
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ImagePath: C:\WINDOWS\System32\mfm\msrll.exe
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\DisplayName: Rll enhanced drive
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\mfm\ObjectName: LocalSystem

Relevant Values modified:10 Total Values modified:30

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: 5D F2 CF 07 1F 2D 10 23 77 0D F9 38 1A 0F E3 29 7F 40 99 B7 4A 1B 98 91 AD FF 7A 7C 6B B3 57 18 1B C9 C2 6B 3F 87 71 A3 F4 10 D3 B3 99 1D A3 25 7C FB EE 1C ED 8B C6 D9 87 BF B3 48 47 BD 57 38 E5 D6 23 E6 AC D4 F2 F9 C0 45 11 6F 7F C1 43 08
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\RNG\Seed: F4 FE B9 FC 22 81 94 7C 59 03 B7 41 58 7B A6 35 34 AE 5D 6E 83 12 D1 C9 B6 D4 2C 6B 6F 93 75 C8 28 20 17 A3 3E 0A 28 9D 88 09 8B 85 AF 34 CD 90 B7 E6 CE 28 CE 68 CA C2 AF 67 03 F8 87 FC 89 A2 3E 11 D0 EF B2 36 97 79 55 9C 24 B9 C8 5F 1D F2
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\kmixer\Enum\Count: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\kmixer\Enum\Count: 0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\kmixer\Enum\NextInstance: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\kmixer\Enum\NextInstance: 0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\Count: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\Count: 0x00000000
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\NextInstance: 0x00000001
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\kmixer\Enum\NextInstance: 0x00000000

Relevant Files added:2 Total Files added:6

C:\WINDOWS\system32\mfm\jtram.conf
C:\WINDOWS\system32\mfm\msrll.exe

Relevant Files deleted:1 Total Files deleted:1

C:\tmp\unpacked.ExE

Relevant Files[attr] modified:2 Total Files[attr] modified:13

C:\WINDOWS\system32\config\software.LOG
C:\WINDOWS\system32\config\system.LOG

Relevant Folders added:1 Total Files added:3

C:\WINDOWS\system32\mfm

Total changes:151

Bug reports to:regshot.yeah.net

Appendix E Filtered File Filemon-001.log.filt

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
50	11:46:27.588	cmd.exe:1868	QUERY INFO	C:\tmp\unpacked.ExE	SUCCESS	Attributes: A
52	11:46:27.588	cmd.exe:1868	DIR	C:\tmp\	SUCCESS	FileBothDIRInformation: unpacked.ExE
54	11:46:27.604	cmd.exe:1868	OPEN	C:\tmp\unpacked.ExE	SUCCESS	Options: Open Access: All
55	11:46:27.604	cmd.exe:1868	QUERY INFO	C:\tmp\unpacked.ExE	SUCCESS	Length: 1175552
97	11:46:27.682	cmd.exe:1868	CLOSE	C:\tmp\unpacked.ExE	SUCCESS	
98	11:46:27.697	unpacked.ExE:256	QUERY INFO	C:\tmp\unpacked.ExE	SUCCESS	FileNameInformation
216	11:46:27.932	unpacked.ExE:256	QUERY INFO	C:\tmp\unpacked.ExE	BUFFER OVERFLOW	FileNameInformation
218	11:46:27.932	unpacked.ExE:256	SET INFO	C:\WINDOWS\system32\config\software.LOG	SUCCESS	Length: 8192
219	11:46:27.932	unpacked.ExE:256	SET INFO	C:\WINDOWS\system32\config\software.LOG	SUCCESS	Length: 8192
220	11:46:27.932	unpacked.ExE:256	SET INFO	C:\WINDOWS\system32\config\software.LOG	SUCCESS	Length: 16384
268	11:46:28.026	unpacked.ExE:256	CREATE	C:\WINDOWS\System32\mfm	SUCCESS	Options: Create DIR Access: All
315	11:46:28.119	unpacked.ExE:256	CREATE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Options: OverwriteIf Sequential Access: All
318	11:46:28.119	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Attributes: A
319	11:46:28.119	unpacked.ExE:256	SET INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Length: 1175552
320	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 0 Length: 65536
321	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 0 Length: 65536
322	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 65536 Length: 65536
323	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 65536 Length: 65536
324	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 131072 Length: 65536
325	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 131072 Length: 65536
326	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 196608 Length: 65536
327	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 196608 Length: 65536
328	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 262144 Length: 65536
329	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 262144 Length: 65536
330	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 327680 Length: 65536

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
331	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 327680 Length: 65536
332	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 393216 Length: 65536
333	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 393216 Length: 65536
334	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 458752 Length: 65536
335	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 458752 Length: 65536
336	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 524288 Length: 65536
337	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 524288 Length: 65536
338	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 589824 Length: 65536
339	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 589824 Length: 65536
340	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 655360 Length: 65536
341	11:46:28.119	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 655360 Length: 65536
342	11:46:28.119	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 720896 Length: 65536
343	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 720896 Length: 65536
344	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 786432 Length: 65536
345	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 786432 Length: 65536
346	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 851968 Length: 65536
347	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 851968 Length: 65536
348	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 917504 Length: 65536
349	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 917504 Length: 65536
350	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 983040 Length: 65536
351	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 983040 Length: 65536
352	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 1048576 Length: 65536
353	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 1048576 Length: 65536
354	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	SUCCESS	Offset: 1114112 Length: 65536
355	11:46:28.135	unpacked.ExE:256	WRITE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 1114112 Length: 61440
356	11:46:28.135	unpacked.ExE:256	READ	C:\tmp\unpacked.ExE	END OF FILE	Offset: 1175552 Length: 65536
357	11:46:28.135	unpacked.ExE:256	SET INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	FileBasicInformation
358	11:46:28.135	unpacked.ExE:256	CLOSE	C:\tmp\unpacked.ExE	SUCCESS	
359	11:46:28.135	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	
454	11:46:28.729	unpacked.ExE:256	READ	C:\Documents and Settings\alan\My Documents\desktop.ini	SUCCESS	Offset: 0 Length: 75
460	11:46:28.729	unpacked.ExE:256	READ	C:\Documents and Settings\alan\My Documents\desktop.ini	SUCCESS	Offset: 0 Length: 75
480	11:46:28.729	unpacked.ExE:256	READ	C:\Documents and Settings\alan\My Documents\desktop.ini	SUCCESS	Offset: 0 Length: 75
508	11:46:28.744	unpacked.ExE:256	READ	C:\Documents and Settings\All Users\Documents\desktop.ini	SUCCESS	Offset: 0 Length: 62
651	11:46:29.416	unpacked.ExE:256	OPEN	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Options: Open Access: All

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
652	11:46:29.416	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Attributes: A
653	11:46:29.416	unpacked.ExE:256	SET INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	FileBasicInformation
654	11:46:29.416	unpacked.ExE:256	READ	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 0 Length: 64
655	11:46:29.416	unpacked.ExE:256	READ	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 128 Length: 64
656	11:46:29.416	unpacked.ExE:256	READ	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 200 Length: 4
657	11:46:29.416	unpacked.ExE:256	READ	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Offset: 220 Length: 4
658	11:46:29.416	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	
698	11:46:29.557	unpacked.ExE:256	OPEN	C:\WINDOWS\System32\	SUCCESS	Options: Open DIR Access: All
699	11:46:29.557	unpacked.ExE:256	DIR	C:\WINDOWS\System32\	SUCCESS	FileBothDIRInformation: mfm
700	11:46:29.557	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\	SUCCESS	
701	11:46:29.557	unpacked.ExE:256	OPEN	C:\WINDOWS\System32\mfm\	SUCCESS	Options: Open DIR Access: All
702	11:46:29.557	unpacked.ExE:256	DIR	C:\WINDOWS\System32\mfm\	SUCCESS	FileBothDIRInformation: msrll.exe
703	11:46:29.557	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\mfm\	SUCCESS	
704	11:46:29.557	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Attributes: A
705	11:46:29.557	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	Length: 1175552
706	11:46:29.557	unpacked.ExE:256	CLOSE	C:\WINDOWS\AppPatch\sysmain.sdb	SUCCESS	
707	11:46:29.635	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\system32\mfm\msrll.exe	SUCCESS	Attributes: A
708	11:46:29.635	unpacked.ExE:256	OPEN	C:\	SUCCESS	Options: Open DIR Access: All
709	11:46:29.635	unpacked.ExE:256	DIR	C:\	SUCCESS	FileBothDIRInformation: WINDOWS
710	11:46:29.635	unpacked.ExE:256	CLOSE	C:\	SUCCESS	
711	11:46:29.635	unpacked.ExE:256	OPEN	C:\WINDOWS\	SUCCESS	Options: Open DIR Access: All
712	11:46:29.635	unpacked.ExE:256	DIR	C:\WINDOWS\	SUCCESS	FileBothDIRInformation: system32
713	11:46:29.635	unpacked.ExE:256	CLOSE	C:\WINDOWS\	SUCCESS	
714	11:46:29.635	unpacked.ExE:256	OPEN	C:\WINDOWS\system32\	SUCCESS	Options: Open DIR Access: All
715	11:46:29.635	unpacked.ExE:256	DIR	C:\WINDOWS\system32\	SUCCESS	FileBothDIRInformation: mfm
716	11:46:29.635	unpacked.ExE:256	CLOSE	C:\WINDOWS\system32\	SUCCESS	
717	11:46:29.635	unpacked.ExE:256	OPEN	C:\WINDOWS\system32\mfm\	SUCCESS	Options: Open DIR Access: All
718	11:46:29.635	unpacked.ExE:256	DIR	C:\WINDOWS\system32\mfm\	SUCCESS	FileBothDIRInformation: msrll.exe
719	11:46:29.635	unpacked.ExE:256	CLOSE	C:\WINDOWS\system32\mfm\	SUCCESS	
720	11:46:29.666	unpacked.ExE:256	OPEN	C:\WINDOWS\System32\mfm\msrll.exe.Manifest	FILE NOT FOUND	Options: Open Access: All
721	11:46:29.666	unpacked.ExE:256	QUERY INFO	C:\WINDOWS\System32\mfm	SUCCESS	Attributes: D
722	11:46:29.666	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
723	11:46:29.666	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	FileNameInformation
725	11:46:29.666	msrll.exe:1492	OPEN	C:\WINDOWS\System32\mfm	SUCCESS	Options: Open DIR Access: Traverse
735	11:46:29.744	unpacked.ExE:256	CLOSE	C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.1515_x-ww_7bb98b8a	SUCCESS	
736	11:46:29.760	unpacked.ExE:256	CLOSE	C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.1515_x-ww_7bb98b8a	SUCCESS	
737	11:46:29.760	unpacked.ExE:256	CLOSE	C:\WINDOWS\System32\mfm	SUCCESS	
738	11:46:29.776	unpacked.ExE:256	CLOSE	C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.2600.1515_x-ww_7bb98b8a	SUCCESS	
893	11:46:29.807	msrll.exe:1492	OPEN	C:\WINDOWS\system32\SHELL32.DLL	SUCCESS	Options: Open Access: All
998	11:46:29.916	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	BUFFER OVERFLOW	FileNameInformation
999	11:46:29.916	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\mfm\msrll.exe	SUCCESS	FileNameInformation
1047	11:46:29.994	msrll.exe:1492	OPEN	C:\tmp\unpacked.ExE	SUCCESS	Options: Open Access: All
1048	11:46:29.994	msrll.exe:1492	DELETE	C:\tmp\unpacked.ExE	SUCCESS	
1049	11:46:29.994	msrll.exe:1492	CLOSE	C:\tmp\unpacked.ExE	SUCCESS	
1051	11:46:30.010	msrll.exe:1492	OPEN	C:\tmp\unpacked.ExE	FILE NOT FOUND	Options: Open Access: All
1089	11:46:30.010	msrll.exe:1492	OPEN	C:\tmp\unpacked.ExE	FILE NOT FOUND	Options: Open Access: All
1106	11:46:30.041	msrll.exe:1492	OPEN	C:\WINDOWS\system32\mfm\jtram.conf	FILE NOT FOUND	Options: Open Access: All
1270	11:46:31.322	msrll.exe:1492	OPEN	C:\Documents and Settings\alan\Local Settings\Temporary Internet Files	SUCCESS	Options: Open Access: All
1271	11:46:31.322	msrll.exe:1492	SET INFO	C:\Documents and Settings\alan\Local Settings\Temporary Internet Files	SUCCESS	FileBasicInformation
1272	11:46:31.338	msrll.exe:1492	CLOSE	C:\Documents and Settings\alan\Local Settings\Temporary Internet Files	SUCCESS	

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
1279	11:46:31.338	msrll.exe:1492	OPEN	C:\Documents and Settings\alan\Local Settings\History	SUCCESS	Options: Open Access: All
1280	11:46:31.338	msrll.exe:1492	SET INFO	C:\Documents and Settings\alan\Local Settings\History	SUCCESS	FileBasicInformation
1281	11:46:31.338	msrll.exe:1492	CLOSE	C:\Documents and Settings\alan\Local Settings\History	SUCCESS	
1282	11:46:31.338	msrll.exe:1492	QUERY INFO	C:\Documents and Settings\alan\Local Settings\History\desktop.ini	SUCCESS	Attributes: HSA
1283	11:46:31.338	msrll.exe:1492	OPEN	C:\Documents and Settings\alan\Local Settings\Temporary Internet Files\Content.IE5\	SUCCESS	Options: Open DIR Access: All
1284	11:46:31.338	msrll.exe:1492	CLOSE	C:\Documents and Settings\alan\Local Settings\Temporary Internet Files\Content.IE5\	SUCCESS	
1301	11:46:31.338	msrll.exe:1492	OPEN	C:\Documents and Settings\alan\Cookies\	SUCCESS	Options: Open Access: All
1302	11:46:31.338	msrll.exe:1492	SET INFO	C:\Documents and Settings\alan\Cookies\	SUCCESS	FileBasicInformation
1303	11:46:31.338	msrll.exe:1492	CLOSE	C:\Documents and Settings\alan\Cookies\	SUCCESS	
1304	11:46:31.338	msrll.exe:1492	OPEN	C:\Documents and Settings\alan\Cookies\index.dat	SUCCESS	Options: OpenIf Access: All
1305	11:46:31.338	msrll.exe:1492	SET INFO	C:\Documents and Settings\alan\Cookies\index.dat	SUCCESS	FileBasicInformation
1306	11:46:31.338	msrll.exe:1492	QUERY INFO	C:\Documents and Settings\alan\Cookies\index.dat	SUCCESS	Length: 32768
1307	11:46:31.338	msrll.exe:1492	CLOSE	C:\Documents and Settings\alan\Cookies\index.dat	SUCCESS	
1698	11:46:32.010	msrll.exe:1492	OPEN	C:\WINDOWS\system32\mfm\jtram.conf	FILE NOT FOUND	Options: Open Access: All
1699	11:46:32.010	msrll.exe:1492	CREATE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Options: OverwriteIf Access: All
1700	11:46:32.010	msrll.exe:1492	OPEN	C:\WINDOWS\system32\mfm\	SUCCESS	Options: Open Access: 00000000
1701	11:46:32.010	msrll.exe:1492	CLOSE	C:\WINDOWS\system32\mfm\	SUCCESS	
1756	11:46:32.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1757	11:46:32.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Length: 133632
1758	11:46:32.041	msrll.exe:1492	CLOSE	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	
1759	11:46:32.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1760	11:46:32.041	msrll.exe:1492	OPEN	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Options: Open Access: All
1761	11:46:32.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1762	11:46:32.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Length: 133632
1763	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 0 Length: 336

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
1764	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 340 Length: 4096
1765	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 4436 Length: 4096
1766	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 8532 Length: 4096
1767	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 12628 Length: 4096
1768	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 16724 Length: 4096
1769	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 20820 Length: 4096
1770	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 24916 Length: 4096
1771	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 29012 Length: 4096
1772	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 33108 Length: 4096
1773	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 37204 Length: 4096
1774	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 41300 Length: 4096
1775	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 45396 Length: 4096
1776	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 49492 Length: 4096
1777	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 53588 Length: 4096
1778	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 57684 Length: 4096
1779	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 61780 Length: 4096
1780	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 65876 Length: 4096
1781	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 69972 Length: 4096
1782	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 74068 Length: 4096
1783	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 78164 Length: 4096
1784	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 82260 Length: 4096
1785	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 86356 Length: 4096
1786	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 90452 Length: 4096
1787	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 94548 Length: 4096
1788	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 98644 Length: 4096
1789	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 102740 Length: 4096
1790	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 106836 Length: 4096
1791	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 110932 Length: 4096
1792	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 115028 Length: 4096
1793	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 119124 Length: 4096
1794	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 123220 Length: 4096
1795	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 127316 Length: 884
1796	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 128360 Length: 4096
1797	11:46:32.041	msrll.exe:1492	READ	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Offset: 132456 Length: 1176
1798	11:46:32.041	msrll.exe:1492	CLOSE	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	
1855	11:46:32.635	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1860	11:46:32.666	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1861	11:46:32.697	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 0 Length: 53

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
1863	11:46:32.713	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1870	11:46:32.776	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1872	11:46:32.807	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1874	11:46:32.838	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1876	11:46:32.854	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1878	11:46:32.885	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1880	11:46:32.901	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1882	11:46:32.916	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1884	11:46:32.947	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1886	11:46:32.963	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1887	11:46:32.979	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 53 Length: 53
1889	11:46:32.979	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1891	11:46:32.994	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1893	11:46:33.010	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1895	11:46:33.041	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1897	11:46:33.057	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1899	11:46:33.072	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1901	11:46:33.088	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1905	11:46:33.151	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1907	11:46:33.182	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1909	11:46:33.213	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
1911	11:46:33.229	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1913	11:46:33.244	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1915	11:46:33.276	msrll.exe:1492	QUERY INFO	C:\WINDOWS\System32\rsaenh.dll	SUCCESS	Attributes: A
1916	11:46:33.276	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 106 Length: 53
1917	11:46:33.276	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 159 Length: 1
1969	11:46:34.041	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 213 Length: 53
2004	11:46:34.604	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 266 Length: 53
2005	11:46:34.604	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 319 Length: 1
2036	11:46:34.947	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 320 Length: 53
2062	11:46:35.213	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 373 Length: 53
2067	11:46:35.260	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 426 Length: 129
2068	11:46:35.260	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 555 Length: 1
2099	11:46:35.697	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 556 Length: 53
2120	11:46:35.994	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 609 Length: 53
2147	11:46:36.041	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 662 Length: 53
2148	11:46:36.041	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 715 Length: 1
2179	11:46:36.057	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 716 Length: 53
2208	11:46:36.057	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 769 Length: 53
2213	11:46:36.057	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 822 Length: 77
2214	11:46:36.057	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 899 Length: 1
2245	11:46:36.072	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 900 Length: 53
2266	11:46:36.088	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 953 Length: 53
2271	11:46:36.088	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 1006 Length: 77
2272	11:46:36.088	msrll.exe:1492	WRITE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	Offset: 1083 Length: 1
2273	11:46:36.088	msrll.exe:1492	CLOSE	C:\WINDOWS\system32\mfm\jtram.conf	SUCCESS	

Appendix F Filtered File Regmon-001.log.filt

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
576	11:46:27.932	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1A5D138
577	11:46:27.932	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	B9 5F 65 0F 6E A4 1A C0 ...
582	11:46:27.932	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1A5D138
678	11:46:28.072	services.exe:656	OpenKey	HKLM\System\CurrentControlSet\Services	SUCC	Key: 0xE1B7C468
679	11:46:28.072	services.exe:656	CreateKey	HKLM\System\CurrentControlSet\Services\mfm	SUCC	Key: 0xE1AEE650
680	11:46:28.072	services.exe:656	CloseKey	HKLM\System\CurrentControlSet\Services	SUCC	Key: 0xE1B7C468
681	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\Type	SUCC	0x120
682	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\Start	SUCC	0x2
683	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\ErrorControl	SUCC	0x2
684	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\ImagePath	SUCC	C:\WINDOWS\System32\mfm\msrll.exe
685	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\DisplayName	SUCC	Rll enhanced drive
686	11:46:28.072	services.exe:656	CreateKey	HKLM\System\CurrentControlSet\Services\mfm\Security	SUCC	Key: 0xE1104C48
687	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\Security\Security	SUCC	01 00 14 80 90 00 00 00 ...
689	11:46:28.072	services.exe:656	SetValue	HKLM\System\CurrentControlSet\Services\mfm\ObjectName	SUCC	LocalSystem
690	11:46:28.119	services.exe:656	FlushKey	HKLM\System\CurrentControlSet\Services\mfm	SUCC	Key: 0xE1AEE650
691	11:46:28.119	services.exe:656	CloseKey	HKLM\System\CurrentControlSet\Services\mfm	SUCC	Key: 0xE1AEE650
693	11:46:28.135	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
694	11:46:28.135	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	DB 7B 60 C9 AA 32 A1 C7 ...
695	11:46:28.151	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
696	11:46:28.151	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
697	11:46:28.151	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	3B 96 32 FF EC CE 99 97 ...
698	11:46:28.166	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
699	11:46:28.166	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
700	11:46:28.166	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	5E 56 A6 2C 92 97 FA B1 ...
701	11:46:28.182	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
702	11:46:28.182	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
703	11:46:28.182	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	DF FD 12 10 3B 39 84 5F ...
704	11:46:28.182	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
705	11:46:28.197	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650

ID	Timestamp	Process and PID	Event	Object	Result	Other Information
706	11:46:28.197	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	ED 06 60 4B B8 DC D8 A1 ...
707	11:46:28.197	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
708	11:46:28.213	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
709	11:46:28.213	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	A0 15 6E 95 33 99 4D 2C ...
710	11:46:28.213	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
711	11:46:28.213	unpacked.ExE:256	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650
712	11:46:28.213	unpacked.ExE:256	SetValue	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	SUCC	5A DE C8 8D 79 44 CA 14 ...
713	11:46:28.229	unpacked.ExE:256	CloseKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG	SUCC	Key: 0xE1AEE650

Appendix G Filtered File TDImon-001.log.filt

ID	Time	Process	Object	Request	Local	Remote	Result	Other
113	11:46:30	msrll.exe:864	815EF750	IRP_MJ_CREATE	TCP:0.0.0.0:2200		SUC	Address Open
114	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Error Event
115	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Disconnect Event
116	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Receive Event
117	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Expedited Receive Event
118	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Chained Receive Event
119	11:46:30	msrll.exe:864	815EF750	TDI_QUERY_INFORMATION	TCP:0.0.0.0:2200		SUC	Query Address
120	11:46:30	msrll.exe:864	815FD498	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x8144DE10
121	11:46:30	msrll.exe:864	815FD498	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:2200
122	11:46:30	msrll.exe:864	815E3D10	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x8144DD70
123	11:46:30	msrll.exe:864	815E3D10	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:2200
124	11:46:30	msrll.exe:864	814A3938	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x8144DB90
125	11:46:30	msrll.exe:864	814A3938	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:2200
126	11:46:30	msrll.exe:864	815EF750	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:2200		SUC	Connect Event
…								
183	11:46:32	msrll.exe:864	817D1330	IRP_MJ_CREATE	TCP:0.0.0.0:113		SUC	Address Open
184	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Error Event
185	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Disconnect Event
186	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Receive Event
187	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Expedited Receive Event
188	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Chained Receive Event
189	11:46:32	msrll.exe:864	817D1330	TDI_QUERY_INFORMATION	TCP:0.0.0.0:113		SUC	Query Address
190	11:46:32	msrll.exe:864	817D34E8	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x8144D5F0
191	11:46:32	msrll.exe:864	817D34E8	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:113
192	11:46:32	msrll.exe:864	817D3450	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x817D36D0
193	11:46:32	msrll.exe:864	817D3450	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:113
194	11:46:32	msrll.exe:864	8165CAE0	IRP_MJ_CREATE	TCP:Connection obj		SUC	Context:0x817D3008
195	11:46:32	msrll.exe:864	8165CAE0	TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUC	TCP:0.0.0.0:113
196	11:46:32	msrll.exe:864	817D1330	TDI_SET_EVENT_HANDLER	TCP:0.0.0.0:113		SUC	Connect Event

Appendix H File *jtram.conf*

Jtram.conf is listed below in ASCII and hexadecimal format. The structure, contents and use of *jtram.conf* was not determined in this work. Further analysis of the disassembled code as described in Section 4 may show the encryption functions used. Also, debugging the running program using OllyDbg may show the data being encrypted and decrypted and their role in the application.

Offset		Data (hexadecimal notation)		ASCII
Hex	Dec			
0x000	0	41 66 38 52 41 44 38 57	42 34 58 30 6f 32 46 67	Af8RAD8WB4X0o2Fg
0x010	16	43 35 43 4e 7a 38 64 6a	74 51 74 71 47 50 62 51	C5CNz8djtQtqGPbQ
0x020	32	77 62 36 49 41 52 67 39	63 36 63 79 59 70 47 51	wb6IARg9c6cyYpGQ
0x030	48	59 67 3d 3d 20 51 76 38	52 41 4e 63 41 53 43 51	Yg== Qv8RANcASCQ
0x040	64	6a 30 53 2b 34 42 4c 75	55 55 66 42 77 76 7a 6d	j0S+4BLuUUfBwvzm
0x050	80	43 65 64 43 39 74 6a 71	65 62 2b 4d 42 45 65 34	CedC9tjqeb+MBEe4
0x060	96	58 43 49 30 73 31 77 3d	3d 20 64 67 41 52 41 41	XCI0s1w== dgARAA
0x070	112	68 65 41 6b 54 70 4e 49	4a 52 5a 75 48 48 51 63	heAkTpNIJRZuHHQc
0x080	128	7a 66 59 38 56 77 6a 77	49 39 53 70 68 72 4c 6e	zfY8VwjwI9SphrLn
0x090	144	79 2b 6d 50 76 79 77 5a	76 6e 37 41 3d 3d 20 0a	y+mPvywZvn7A== .
0x0a0	160	6b 51 41 52 41 49 39 6c	74 59 44 35 62 45 73 5a	kQARAI9ltYD5bEsZ
0x0b0	176	59 2f 62 57 4c 50 4d 73	7a 35 5a 2b 50 37 2f 4c	Y/bWLPMsz5Z+P7/L
0x0c0	192	52 45 31 2f 6b 6f 75 55	65 33 2f 4b 2b 35 4c 65	RE1/kouUe3/K+5Le
0x0d0	208	65 41 3d 3d 20 56 76 30	52 41 4b 36 47 68 6f 76	eA== Vv0RAK6Ghov
0x0e0	224	52 38 4d 55 68 43 6f 51	7a 50 70 45 5a 71 5a 56	R8MUhCoQzPpEZqZV
0x0f0	240	36 4f 69 6f 4d 6d 35 45	4e 2f 6e 64 61 38 34 6f	6OioMm5EN/nda84o
0x100	256	79 33 67 71 55 56 51 3d	3d 20 56 67 41 52 41 48	y3gqUVQ== VgARAH
0x110	272	49 55 69 6c 54 61 69 74	4a 38 4a 38 64 73 4c 77	IUilTaitJ8J8dsLw
0x120	288	35 33 49 4d 31 49 6d 4f	33 53 49 72 78 4d 70 31	53IM1ImO3SIrxMp1
0x130	304	71 4f 68 32 63 55 38 6c	76 6e 70 67 3d 3d 20 0a	qOh2cU8lvnpg== .
0x140	320	55 41 41 52 41 4b 4b 39	53 41 49 55 6c 65 66 67	UAARAKK9SAIUlefg
0x150	336	4b 31 33 61 5a 49 4f 42	74 31 4c 4f 43 39 6b 41	K13aZIOBt1LOC9kA
0x160	352	64 70 39 50 63 66 56 52	61 43 43 6c 6b 2b 4e 6d	dp9PcfVRaCClk+Nm
0x170	368	66 51 3d 3d 20 6a 51 41	52 41 48 4a 32 6e 39 79	fQ== jQARAHJ2n9y
0x180	384	44 56 65 68 59 59 6f 41	6c 4c 4e 4e 5a 6e 61 4f	DVehYYoAlLNNZnaO
0x190	400	37 51 33 6b 54 6b 47 48	31 70 7a 7a 4c 30 33 6f	7Q3kTkGH1pzzL03o
0x1a0	416	4d 53 6d 75 42 74 51 3d	3d 20 50 66 35 4b 41 48	MSmuBtQ== Pf5KAH

Offset			
0x1b0	432	4b 78 79 5a 4d 35 68 4e 44 67 48 4f 64 51 4f 61	KxyZM5hNDgHOdQOa
0x1c0	448	46 41 67 48 4a 50 4f 78 36 42 48 63 62 4a 36 58	FAgHJPOx6BHcbJ6X
0x1d0	464	43 4a 68 37 74 66 50 47 64 49 57 50 62 69 32 74	CJh7tfPGdIWPbi2t
0x1e0	480	32 51 69 58 63 73 4e 52 71 61 65 71 30 4a 2f 62	2QiXcsNRqaeq0J/b
0x1f0	496	34 6f 64 7a 71 45 52 41 68 64 6f 70 74 72 5a 4d	4odzqERAhdoptrZM
0x200	512	48 6e 7a 49 48 69 4e 35 32 53 72 6c 45 47 41 36	HnzIHiN52SrlEGA6
0x210	528	6e 77 6c 75 51 6f 70 47 33 73 7a 32 52 59 49 67	nwluQopG3sz2RYIg
0x220	544	68 73 72 77 71 5a 37 41 3d 3d 20 0a 38 2f 34 52	hsrwqZ7A== .8/4R
0x230	560	41 49 47 78 4e 46 78 30 71 39 4c 71 32 33 58 78	AIGxNFx0q9Lq23Xx
0x240	576	54 5a 6a 4b 73 71 61 4d 45 6b 6f 37 4f 45 35 30	TZjKsqaMEko7OE50
0x250	592	6f 4b 36 59 71 6a 4c 57 4e 4c 68 77 37 41 3d 3d	oK6YqjLWNLhw7A==
0x260	608	20 36 51 41 52 41 43 71 64 43 62 42 6d 53 6d 50	6QARACqdCbBmSmP
0x270	624	65 34 65 48 35 53 31 55 6f 78 77 5a 69 50 36 55	e4eH5S1UoxwZiP6U
0x280	640	38 50 64 62 45 47 58 41 6d 43 4b 6f 73 6a 52 4c	8PdbEGXAmCKosjRL
0x290	656	6b 72 67 3d 3d 20 75 67 45 52 41 46 43 52 46 72	krg== ugERAFCRFr
0x2a0	672	41 50 66 55 64 6e 42 32 6a 4c 6c 31 70 63 43 57	APfUdnB2jLl1pcCW
0x2b0	688	4f 72 35 7a 31 6b 36 43 57 6b 4e 57 77 73 50 76	Or5z1k6CWkNWwsPv
0x2c0	704	4b 38 75 6a 57 4c 76 67 3d 3d 20 0a 34 2f 38 52	K8ujWLvg== .4/8R
0x2d0	720	41 44 64 64 6c 75 70 42 6e 4c 52 65 46 41 38 74	ADddlupBnLReFA8t
0x2e0	736	76 65 2f 5a 61 4a 71 7a 51 58 6f 49 41 34 6a 72	ve/ZaJqzQXoIA4jr
0x2f0	752	53 5a 45 78 47 4f 38 63 61 52 58 4f 6b 67 3d 3d	SZExGO8caRXOkg==
0x300	768	20 6a 77 49 52 41 46 51 70 4d 64 73 58 6e 6e 5a	jwIRAFQpMdsXnnZ
0x310	784	6e 45 31 49 51 6e 47 39 73 2f 55 51 74 51 2f 37	nE1IQnG9s/UQtQ/7
0x320	800	66 34 57 69 77 7a 30 44 4f 6b 62 78 52 6c 54 78	f4Wiwz0DOkbxRlTx
0x330	816	65 52 67 3d 3d 20 4f 2f 38 6a 41 4a 51 35 4e 2f	eRg== O/8jAJQ5N/
0x340	832	4e 62 6f 79 70 56 78 35 48 33 64 69 75 64 2f 4d	NboypVx5H3diud/M
0x350	848	41 56 79 50 4b 33 35 4c 51 38 31 33 73 76 73 4c	AVyPK35LQ813svsL
0x360	864	6c 34 6e 39 45 44 2f 31 2f 30 32 37 34 50 55 44	l4n9ED/1/0274PUD
0x370	880	34 4d 66 59 4c 5a 54 6f 34 44 32 4f 77 50 44 67	4MfYLZTo4D2OwPDg
0x380	896	3d 3d 20 0a 63 77 41 52 41 4b 39 56 39 61 64 78	== .cwARAK9V9adx
0x390	912	35 42 5a 6e 4e 46 77 64 75 6b 37 6d 2f 6d 6d 6b	5BZnNFwduk7m/mmk
0x3a0	928	46 72 69 67 74 74 41 46 44 41 61 4f 2f 30 73 2b	FrigttAFDAaO/0s+
0x3b0	944	37 68 59 54 45 67 3d 3d 20 74 41 45 52 41 49 32	7hYTEg== tAERAI2
0x3c0	960	51 75 69 35 37 42 56 4f 37 57 4c 45 6b 4f 2b 64	Qui57BVO7WLEkO+d
0x3d0	976	4a 59 53 6a 71 76 75 46 4b 52 54 74 53 4b 66 58	JYSjqvuFKRTtSKfX
0x3e0	992	64 47 63 61 67 76 6e 63 70 63 67 3d 3d 20 74 51	dGcagvncpcg== tQ
0x3f0	1008	45 6a 41 50 31 48 66 78 35 5a 4b 78 61 4d 57 34	EjAP1Hfx5ZKxaMW4
0x400	1024	52 70 5a 2b 48 65 35 4a 70 76 6b 34 68 50 52 52	RpZ+He5Jpvk4hPRR

Offset			
0x410	1040	34 6c 4b 65 54 39 4a 55 4f 41 2b 6a 7a 34 4d 2b	4lKeT9JUOA+jz4M+
0x420	1056	49 79 6e 6b 52 51 56 76 6f 42 67 59 5a 68 75 73	IynkRQVvoBgYZhus
0x430	1072	77 75 2f 69 2f 38 44 77 3d 3d 20 0a	wu/i/8Dw== .
0x43c	1084		

Possible structure of jtram.conf:

```
Af8RAD8WB4X0o2FgC5CNz8djtQtqGPbQwb6IARg9c6cyYpGQYg==
Qv8RANcASCQj0S+4BLuUUfBwvzmCedC9tjqeb+MBEe4XCI0s1w==
dgARAAheAkTpNIJRZuHHQczfY8VwjwI9SphrLny+mPvywZvn7A==
kQARAI9ltYD5bEsZY/bWLPMsz5Z+P7/LRE1/kouUe3/K+5LeeA==
Vv0RAK6GhovR8MUhCoQzPpEZqZV6OioMm5EN/nda84oy3gqUVQ==
VgARAHIUilTaitJ8J8dsLw53IM1ImO3SIrxMp1qOh2cU8lvnpg==
UAARAKK9SAIUlefgK13aZIOBt1LOC9kAdp9PcfVRaCClk+NmfQ==
jQARAHJ2n9yDVehYYoAlLNNZnaO7Q3kTkGH1pzzL03oMSmuBtQ==
Pf5KAHKxyZM5hNDgHOdQOaFAgHJPOx6BHcbJ6XCJh7tfPGdIWPbi2t2QiXcsNRqaeq0J/b4odzqERAhdoptrZMHnzIHiN52SrlEGA6nwlu
QopG3sz2RYIghsrwqZ7A==
8/4RAIGxNFx0q9Lq23XxTZjKsqaMEko7OE50oK6YqjLWNLhw7A==
6QARACqdCbBmSmPe4eH5S1UoxwZiP6U8PdbEGXAmCKosjRLkrg==
ugERAFCRFrAPfUdnB2jLl1pcCWOr5z1k6CWkNWwsPvK8ujWLvg==
4/8RADddlupBnLReFA8tve/ZaJqzQXoIA4jrSZExGO8caRXOkg==
jwIRAFQpMdsXnnZnE1IQnG9s/UQtQ/7f4Wiwz0DOkbxRlTxeRg==
O/8jAJQ5N/NboypVx5H3diud/MAVyPK35LQ813svsLl4n9ED/1/0274PUD4MfYLZTo4D2OwPDg ==
cwARAK9V9adx5BZnNFwduk7m/mmkFrigttAFDAaO/0s+7hYTEg==
tAERAI2Qui57BVO7WLEkO+dJYSjqvuFKRTtSKfXdGcagvncpcg==
tQEjAP1Hfx5ZKxaMW4RpZ+He5Jpvk4hPRR4lKeT9JUOA+jz4M+IynkRQVvoBgYZhuswu/i/8Dw==
```

Appendix I *msrll-002.log* (Snort Log of IRC server polling)

This Appendix lists the packets captured by Snort as *unpacked.ExE* attempts to connect to the server collective7.zxy0.com on ports 6667, 8080 and 9999. Refer to Section 5.3 for a discussion of these events.

Except for ARP entries, Snort displays captured packets on several lines in the log file. The notable exception is the ARP packet display, which consists of a single line or text. ARP packets map IP addresses to the hardware (MAC) addresses for network interface adapters.

All Snort log entries begin with a timestamp, e.g., 12/01-16:45:19.572216 as shown below. The string “ARP” immediately follows the timestamp for ARP packets.

Log entries for TCP, UDP and ICMP packets span several lines. The timestamp is followed by the packet’s source IP address, source port, and destination IP address and destination port, all on one line. The second line contains the string UDP, TCP or ICMP corresponding to the packet protocol, followed by additional information as appropriate for the packet type. Snort then lists the packet payload and header information in both hexadecimal and ASCII. A separator string of repeated “=+=+” characters indicates the end of the packet dump.

The Snort capture log file `msrll-002.log` follows:

```
12/01-16:45:19.572216 ARP who-has 172.16.238.129 tell 172.16.238.128

12/01-16:45:19.573311 ARP reply 172.16.238.129 is-at 0:C:29:B9:A2:63

12/01-16:45:19.574746 172.16.238.128:1050 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:1168 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7BF18060  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:19.576865 172.16.238.129:6667 -> 172.16.238.128:1050
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7BF18061  Win: 0x0  TcpLen: 20

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

12/01-16:45:20.069877 172.16.238.128:1050 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:1169 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7BF18060  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:20.070104 172.16.238.129:6667 -> 172.16.238.128:1050
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7BF18061  Win: 0x0  TcpLen: 20
```

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

12/01-16:45:20.512278 172.16.238.128:1050 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:1170 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7BF18060  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:20.512350 172.16.238.129:6667 -> 172.16.238.128:1050
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7BF18061  Win: 0x0  TcpLen: 20

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

12/01-16:45:24.575275 ARP who-has 172.16.238.128 tell 172.16.238.129

12/01-16:45:24.576110 ARP reply 172.16.238.128 is-at 0:C:29:B1:63:AF

12/01-16:45:50.524975 172.16.238.128:1051 -> 172.16.238.129:9999
TCP TTL:128 TOS:0x0 ID:1171 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7C5E4029  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:50.525062 172.16.238.129:9999 -> 172.16.238.128:1051
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7C5E402A  Win: 0x0  TcpLen: 20

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

12/01-16:45:50.942431 172.16.238.128:1051 -> 172.16.238.129:9999
TCP TTL:128 TOS:0x0 ID:1172 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7C5E4029  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:50.942467 172.16.238.129:9999 -> 172.16.238.128:1051
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7C5E402A  Win: 0x0  TcpLen: 20

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

12/01-16:45:51.452772 172.16.238.128:1051 -> 172.16.238.129:9999
TCP TTL:128 TOS:0x0 ID:1173 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7C5E4029  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:45:51.452837 172.16.238.129:9999 -> 172.16.238.128:1051
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7C5E402A  Win: 0x0  TcpLen: 20

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

```
12/01-16:46:21.498446 172.16.238.128:1052 -> 172.16.238.129:8080
TCP TTL:128 TOS:0x0 ID:1174 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7CCAF45E  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:46:21.498618 172.16.238.129:8080 -> 172.16.238.128:1052
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7CCAF45F  Win: 0x0  TcpLen: 20

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

12/01-16:46:21.950481 172.16.238.128:1052 -> 172.16.238.129:8080
TCP TTL:128 TOS:0x0 ID:1175 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7CCAF45E  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:46:21.950525 172.16.238.129:8080 -> 172.16.238.128:1052
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7CCAF45F  Win: 0x0  TcpLen: 20

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

12/01-16:46:22.382037 172.16.238.128:1052 -> 172.16.238.129:8080
TCP TTL:128 TOS:0x0 ID:1176 IpLen:20 DgmLen:48 DF
******S* Seq: 0x7CCAF45E  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

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

12/01-16:46:22.382079 172.16.238.129:8080 -> 172.16.238.128:1052
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x0  Ack: 0x7CCAF45F  Win: 0x0  TcpLen: 20

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

Appendix J *msrll-003.log* (Snort Log of IRC Connection)

This Appendix lists the packets captured by Snort as *msrll.exe* connects to the IRC server on collective7.zxy0.com. Refer to Section 5.3 for a discussion of these events. Table 16 and Table 17 summarize the IRC session and interpret the IRC dialog in the captured packets.

The Snort capture log file `msrll-003.log` follows:

```
12/01-18:11:12.595395 172.16.238.128:1171 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:50734 IpLen:20 DgmLen:89 DF
***AP*** Seq: 0x7747C70F  Ack: 0x3C6010FB  Win: 0xFAF0  TcpLen: 20
55 53 45 52 20 5A 65 74 6A 64 20 6C 6F 63 61 6C  USER Zetjd local
68 6F 73 74 20 30 20 3A 4F 4A 6E 73 4B 7A 49 50  host 0 :OJnsKzIP
42 0A 4E 49 43 4B 20 6C 45 54 55 53 45 4E 74 43  B.NICK lETUSENtC
0A                                               .

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

12/01-18:11:12.595605 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47206 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x3C6010FB  Ack: 0x7747C740  Win: 0x16D0  TcpLen: 20

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

12/01-18:11:12.596598 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47207 IpLen:20 DgmLen:86 DF
***AP*** Seq: 0x3C6010FB  Ack: 0x7747C740  Win: 0x16D0  TcpLen: 20
4E 4F 54 49 43 45 20 41 55 54 48 20 3A 2A 2A 2A  NOTICE AUTH :***
20 4C 6F 6F 6B 69 6E 67 20 75 70 20 79 6F 75 72   Looking up your
20 68 6F 73 74 6E 61 6D 65 2E 2E 2E 0D 0A         hostname.....

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

3-way handshake omitted

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

12/01-18:11:12.711235 172.16.238.129:32771 -> 172.16.238.128:113
TCP TTL:64 TOS:0x0 ID:42238 IpLen:20 DgmLen:65 DF
***AP*** Seq: 0x3C6E0C63  Ack: 0x77489313  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 567465 0
31 31 37 31 20 2C 20 36 36 36 37 0D 0A           1171 , 6667..

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

12/01-18:11:12.715972 172.16.238.128:113 -> 172.16.238.129:32771
TCP TTL:128 TOS:0x0 ID:50736 IpLen:20 DgmLen:89 DF
***AP*** Seq: 0x77489313  Ack: 0x3C6E0C70  Win: 0xFAE3  TcpLen: 32
TCP Options (3) => NOP NOP TS: 6878986 567465
31 31 37 31 20 2C 20 36 36 36 37 20 3A 20 55 53  1171 , 6667 : US
45 52 49 44 20 3A 20 55 4E 49 58 20 3A 20 70 53  ERID : UNIX : pS
61 50 76 7A 0A                                   aPvz.
```

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

12/01-18:11:12.741535 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47208 IpLen:20 DgmLen:110 DF
***AP*** Seq: 0x3C601129  Ack: 0x7747C740  Win: 0x16D0  TcpLen: 20
4E 4F 54 49 43 45 20 41 55 54 48 20 3A 2A 2A 2A  NOTICE AUTH :***
20 43 68 65 63 6B 69 6E 67 20 49 64 65 6E 74 0D   Checking Ident.
0A 4E 4F 54 49 43 45 20 41 55 54 48 20 3A 2A 2A  .NOTICE AUTH :**
2A 20 47 6F 74 20 49 64 65 6E 74 20 72 65 73 70  * Got Ident resp
6F 6E 73 65 0D 0A                                onse..

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

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

12/01-18:11:39.668168 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47209 IpLen:20 DgmLen:89 DF
***AP*** Seq: 0x3C60116F  Ack: 0x7747C740  Win: 0x16D0  TcpLen: 20
4E 4F 54 49 43 45 20 41 55 54 48 20 3A 2A 2A 2A  NOTICE AUTH :***
20 43 6F 75 6C 64 6E 27 74 20 6C 6F 6F 6B 20 75   Couldn't look u
70 20 79 6F 75 72 20 68 6F 73 74 6E 61 6D 65 0D  p your hostname.
0A                                               .

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

12/01-18:11:40.298319 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47210 IpLen:20 DgmLen:1064 DF
***AP*** Seq: 0x3C6011A0  Ack: 0x7747C740  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 30 30 31 20 6C 45 54 55 53  domain 001 lETUS
45 4E 74 43 20 3A 57 65 6C 63 6F 6D 65 20 74 6F  ENtC :Welcome to
20 74 68 65 20 49 6E 74 65 72 6E 65 74 20 52 65   the Internet Re
6C 61 79 20 4E 65 74 77 6F 72 6B 20 6C 45 54 55  lay Network lETU
53 45 4E 74 43 0D 0A 3A 6C 6F 63 61 6C 68 6F 73  SENtC..:localhos
74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 30 30  t.localdomain 00
32 20 6C 45 54 55 53 45 4E 74 43 20 3A 59 6F 75  2 lETUSENtC :You
72 20 68 6F 73 74 20 69 73 20 6C 6F 63 61 6C 68  r host is localh
6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 5B  ost.localdomain[
6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61 6C 64  localhost.locald
6F 6D 61 69 6E 2F 36 36 36 37 5D 2C 20 72 75 6E  omain/6667], run
6E 69 6E 67 20 76 65 72 73 69 6F 6E 20 32 2E 38  ning version 2.8
2F 68 79 62 72 69 64 2D 36 2E 33 2E 31 0D 0A 4E  /hybrid-6.3.1..N
4F 54 49 43 45 20 6C 45 54 55 53 45 4E 74 43 20  OTICE lETUSENtC 
3A 2A 2A 2A 20 59 6F 75 72 20 68 6F 73 74 20 69  :*** Your host i
73 20 6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61  s localhost.loca
6C 64 6F 6D 61 69 6E 5B 6C 6F 63 61 6C 68 6F 73  ldomain[localhos
74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 2F 36 36  t.localdomain/66
36 37 5D 2C 20 72 75 6E 6E 69 6E 67 20 76 65 72  67], running ver
73 69 6F 6E 20 32 2E 38 2F 68 79 62 72 69 64 2D  sion 2.8/hybrid-
36 2E 33 2E 31 0D 0A 3A 6C 6F 63 61 6C 68 6F 73  6.3.1..:localhos
74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 30 30  t.localdomain 00
33 20 6C 45 54 55 53 45 4E 74 43 20 3A 54 68 69  3 lETUSENtC :Thi
73 20 73 65 72 76 65 72 20 77 61 73 20 63 72 65  s server was cre
61 74 65 64 20 54 75 65 20 4A 75 6E 20 34 20 32  ated Tue Jun 4 2
30 30 32 20 61 74 20 31 36 3A 35 39 3A 34 35 20  002 at 16:59:45 
45 44 54 0D 0A 3A 6C 6F 63 61 6C 68 6F 73 74 2E  EDT..:localhost.
```

```
6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 30 30 34 20  localdomain 004
6C 45 54 55 53 45 4E 74 43 20 6C 6F 63 61 6C 68  lETUSENtC localh
6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20  ost.localdomain
32 2E 38 2F 68 79 62 72 69 64 2D 36 2E 33 2E 31  2.8/hybrid-6.3.1
20 6F 4F 69 77 73 7A 63 72 6B 66 79 64 6E 78 62   oOiwszcrkfydnxb
20 62 69 6B 6C 6D 6E 6F 70 73 74 76 65 0D 0A 3A   biklmnopstve..:
6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61 6C 64  localhost.locald
6F 6D 61 69 6E 20 30 30 35 20 6C 45 54 55 53 45  omain 005 lETUSE
4E 74 43 20 57 41 4C 4C 43 48 4F 50 53 20 50 52  NtC WALLCHOPS PR
45 46 49 58 3D 28 6F 76 29 40 2B 20 43 48 41 4E  EFIX=(ov)@+ CHAN
54 59 50 45 53 3D 23 26 20 4D 41 58 43 48 41 4E  TYPES=#& MAXCHAN
4E 45 4C 53 3D 32 30 20 4D 41 58 42 41 4E 53 3D  NELS=20 MAXBANS=
32 35 20 4E 49 43 4B 4C 45 4E 3D 39 20 54 4F 50  25 NICKLEN=9 TOP
49 43 4C 45 4E 3D 31 32 30 20 4B 49 43 4B 4C 45  ICLEN=120 KICKLE
4E 3D 39 30 20 4E 45 54 57 4F 52 4B 3D 45 46 6E  N=90 NETWORK=EFn
65 74 20 43 48 41 4E 4D 4F 44 45 53 3D 62 2C 6B  et CHANMODES=b,k
2C 6C 2C 69 6D 6E 70 73 74 20 4D 4F 44 45 53 3D  ,l,imnpst MODES=
34 20 3A 61 72 65 20 73 75 70 70 6F 72 74 65 64  4 :are supported
20 62 79 20 74 68 69 73 20 73 65 72 76 65 72 0D   by this server.
0A 3A 6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61  .:localhost.loca
6C 64 6F 6D 61 69 6E 20 32 35 31 20 6C 45 54 55  ldomain 251 lETU
53 45 4E 74 43 20 3A 54 68 65 72 65 20 61 72 65  SENtC :There are
20 30 20 75 73 65 72 73 20 61 6E 64 20 31 20 69   0 users and 1 i
6E 76 69 73 69 62 6C 65 20 6F 6E 20 31 20 73 65  nvisible on 1 se
72 76 65 72 73 0D 0A 3A 6C 6F 63 61 6C 68 6F 73  rvers..:localhos
74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 32 35  t.localdomain 25
35 20 6C 45 54 55 53 45 4E 74 43 20 3A 49 20 68  5 lETUSENtC :I h
61 76 65 20 31 20 63 6C 69 65 6E 74 73 20 61 6E  ave 1 clients an
64 20 30 20 73 65 72 76 65 72 73 0D 0A 3A 6C 6F  d 0 servers..:lo
63 61 6C 68 6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D  calhost.localdom
61 69 6E 20 32 36 35 20 6C 45 54 55 53 45 4E 74  ain 265 lETUSENt
43 20 3A 43 75 72 72 65 6E 74 20 6C 6F 63 61 6C  C :Current local
20 20 75 73 65 72 73 3A 20 31 20 20 4D 61 78 3A    users: 1  Max:
20 31 0D 0A 3A 6C 6F 63 61 6C 68 6F 73 74 2E 6C   1..:localhost.l
6F 63 61 6C 64 6F 6D 61 69 6E 20 32 36 36 20 6C  ocaldomain 266 l
45 54 55 53 45 4E 74 43 20 3A 43 75 72 72 65 6E  ETUSENtC :Curren

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

12/01-18:11:40.307318 172.16.238.128:1171 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:50742 IpLen:20 DgmLen:59 DF
***AP*** Seq: 0x7747C740  Ack: 0x3C6015A0  Win: 0xF64B  TcpLen: 20
55 53 45 52 48 4F 53 54 20 6C 45 54 55 53 45 4E  USERHOST lETUSEN
74 43 0A                                         tC.

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

12/01-18:11:40.307449 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47211 IpLen:20 DgmLen:443 DF
***AP*** Seq: 0x3C6015A0  Ack: 0x7747C753  Win: 0x16D0  TcpLen: 20
74 20 67 6C 6F 62 61 6C 20 75 73 65 72 73 3A 20  t global users:
31 20 20 4D 61 78 3A 20 31 0D 0A 3A 6C 6F 63 61  1  Max: 1..:loca
6C 68 6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69  lhost.localdomai
6E 20 32 35 30 20 6C 45 54 55 53 45 4E 74 43 20  n 250 lETUSENtC
3A 48 69 67 68 65 73 74 20 63 6F 6E 6E 65 63 74  :Highest connect
69 6F 6E 20 63 6F 75 6E 74 3A 20 31 20 28 31 20  ion count: 1 (1
63 6C 69 65 6E 74 73 29 20 28 31 20 73 69 6E 63  clients) (1 sinc
```

```
65 20 73 65 72 76 65 72 20 77 61 73 20 28 72 65  e server was (re
29 73 74 61 72 74 65 64 29 0D 0A 3A 6C 6F 63 61  )started)..:loca
6C 68 6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69  lhost.localdomai
6E 20 33 37 35 20 6C 45 54 55 53 45 4E 74 43 20  n 375 lETUSENtC
3A 2D 20 6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63  :- localhost.loc
61 6C 64 6F 6D 61 69 6E 20 4D 65 73 73 61 67 65  aldomain Message
20 6F 66 20 74 68 65 20 44 61 79 20 2D 20 0D 0A   of the Day - ..
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 37 32 20 6C 45 54 55 53  domain 372 lETUS
45 4E 74 43 20 3A 2D 20 54 68 69 73 20 69 73 20  ENtC :- This is
61 6E 20 49 52 43 20 73 65 72 76 65 72 2E 20 41  an IRC server. A
75 74 68 6F 72 69 7A 65 64 20 75 73 65 72 73 20  uthorized users
6F 6E 6C 79 2E 0D 0A 3A 6C 6F 63 61 6C 68 6F 73  only...:localhos
74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 33 37  t.localdomain 37
36 20 6C 45 54 55 53 45 4E 74 43 20 3A 45 6E 64  6 lETUSENtC :End
20 6F 66 20 2F 4D 4F 54 44 20 63 6F 6D 6D 61 6E   of /MOTD comman
64 2E 0D 0A 3A 6C 45 54 55 53 45 4E 74 43 20 4D  d...:lETUSENtC M
4F 44 45 20 6C 45 54 55 53 45 4E 74 43 20 3A 2B  ODE lETUSENtC :+
69 0D 0A                                         i..

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

12/01-18:11:40.668153 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47212 IpLen:20 DgmLen:113 DF
***AP*** Seq: 0x3C601733  Ack: 0x7747C753  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 6C 45 54 55 53  domain 302 lETUS
45 4E 74 43 20 3A 6C 45 54 55 53 45 4E 74 43 3D  ENtC :lETUSENtC=
2B 70 53 61 50 76 7A 40 31 37 32 2E 31 36 2E 32  +pSaPvz@172.16.2
33 38 2E 31 33 30 20 0D 0A                       38.128 ..

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

12/01-18:11:44.639459 172.16.238.128:1171 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:50745 IpLen:20 DgmLen:53 DF
***AP*** Seq: 0x7747C753  Ack: 0x3C60177C  Win: 0xFAA7  TcpLen: 20
4A 4F 49 4E 20 23 6D 69 6C 73 20 3A 0A           JOIN #mils :.

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

12/01-18:11:44.639889 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47213 IpLen:20 DgmLen:249 DF
***AP*** Seq: 0x3C60177C  Ack: 0x7747C760  Win: 0x16D0  TcpLen: 20
3A 6C 45 54 55 53 45 4E 74 43 21 70 53 61 50 76  :lETUSENtC!pSaPv
7A 40 31 37 32 2E 31 36 2E 32 33 38 2E 31 33 30  z@172.16.238.128
20 4A 4F 49 4E 20 3A 23 6D 69 6C 73 0D 0A 3A 6C   JOIN :#mils..:l
6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61 6C 64 6F  ocalhost.localdo
6D 61 69 6E 20 4D 4F 44 45 20 23 6D 69 6C 73 20  main MODE #mils
2B 6E 74 0D 0A 3A 6C 6F 63 61 6C 68 6F 73 74 2E  +nt..:localhost.
6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 33 35 33 20  localdomain 353
6C 45 54 55 53 45 4E 74 43 20 3D 20 23 6D 69 6C  lETUSENtC = #mil
73 20 3A 40 6C 45 54 55 53 45 4E 74 43 20 0D 0A  s :@lETUSENtC ..
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 36 36 20 6C 45 54 55 53  domain 366 lETUS
45 4E 74 43 20 23 6D 69 6C 73 20 3A 45 6E 64 20  ENtC #mils :End
6F 66 20 2F 4E 41 4D 45 53 20 6C 69 73 74 2E 0D  of /NAMES list..
0A                                               .
```

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

12/01-18:11:47.628415 172.16.238.128:1171 -> 172.16.238.129:6667
TCP TTL:128 TOS:0x0 ID:50747 IpLen:20 DgmLen:61 DF
***AP*** Seq: 0x7747C760  Ack: 0x3C60184D  Win: 0xF9D6  TcpLen: 20
4D 4F 44 45 20 23 6D 69 6C 73 0A 57 48 4F 20 23  MODE #mils.WHO #
6D 69 6C 73 0A                                   mils.

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

12/01-18:11:47.678104 172.16.238.129:6667 -> 172.16.238.128:1171
TCP TTL:64 TOS:0x0 ID:47215 IpLen:20 DgmLen:321 DF
***AP*** Seq: 0x3C60184D  Ack: 0x7747C775  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 32 34 20 6C 45 54 55 53  domain 324 lETUS
45 4E 74 43 20 23 6D 69 6C 73 20 2B 74 6E 20 0D  ENtC #mils +tn .
0A 3A 6C 6F 63 61 6C 68 6F 73 74 2E 6C 6F 63 61  .:localhost.loca
6C 64 6F 6D 61 69 6E 20 33 32 39 20 6C 45 54 55  ldomain 329 lETU
53 45 4E 74 43 20 23 6D 69 6C 73 20 31 31 30 31  SENtC #mils 1101
39 34 32 37 30 34 0D 0A 3A 6C 6F 63 61 6C 68 6F  942704..:localho
73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 33  st.localdomain 3
35 32 20 6C 45 54 55 53 45 4E 74 43 20 23 6D 69  52 lETUSENtC #mi
6C 73 20 70 53 61 50 76 7A 20 31 37 32 2E 31 36  ls pSaPvz 172.16
2E 32 33 38 2E 31 33 30 20 6C 6F 63 61 6C 68 6F  .238.128 localho
73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E 20 6C  st.localdomain l
45 54 55 53 45 4E 74 43 20 48 40 20 3A 30 20 4F  ETUSENtC H@ :0 O
4A 6E 73 4B 7A 49 50 42 0D 0A 3A 6C 6F 63 61 6C  JnsKzIPB..:local
68 6F 73 74 2E 6C 6F 63 61 6C 64 6F 6D 61 69 6E  host.localdomain
20 33 31 35 20 6C 45 54 55 53 45 4E 74 43 20 23   315 lETUSENtC #
6D 69 6C 73 20 3A 45 6E 64 20 6F 66 20 2F 57 48  mils :End of /WH
4F 20 6C 69 73 74 2E 0D 0A                       O list...

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