GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Reverse-Engineering Malware: Malware Analysis Tools and Techniques (Forens
at http://www.giac.org/registration/grem

GIAC Reverse Engineering Malware (GREM)

Practical Assignment (v1.0)

16 September 2004

Analyzing a BackDoor-CGM Trojan

Prepared by:

Andrew Mackie

Abstract: This report describes an analysis process applied to an unknown malware specimen. The report begins with a detailed description of the laboratory set-up established to contain and examine the malware. The findings of each step of the analysis are documented and, where possible, observations are made about the implications of the findings. Some of the inner workings of this Trojan were successfully revealed. The report concludes with an assessment of the capabilities of the malware and the threat it poses to my organization.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

1 INTRODUCTION

The sharing of information is essential to business operations. Unfortunately the creativity of those developing malicious software (malware) is making it ever more difficult for businesses to protect themselves while they share information between their IT systems. The Internet has become a perilous place for the unwary.

The IT security community must continue to analyse the techniques and strategies used by malware authors so that defences can be improved. Fundamental to this is the reverse engineering of malware and sharing the knowledge gained so that there is benefit to the entire IT community.

This report provides a detailed assessment of an unknown specimen of malware. The report details the laboratory techniques used to isolate the specimen, describes the analysis techniques used and the information gained as a result of the analysis.

The practical assignment report begins with a description of the laboratory set-up in section 2.

Section 3 describes the properties of the malware specimen provided for analysis.

Section 4 offers a behavioural analysis of the malware specimen infecting a system in the laboratory.

Section 5 provides the results of a code analysis of the malware specimen.

The report concludes with an analysis wrap-up in section 6.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

2 LABORATORY SETUP

A Dell Dimension XPS T500 computer (500 MHz PIII with 512 MB memory) with a new 40 GB Maxtor hard drive was set up as a dedicated malware analysis computer. Windows 2000 Professional was installed as the host operating system. The required display driver was installed to support the ATI Radeon 7500 add-in video controller board for this computer. The latest Windows service pack (SP4) was applied from a distribution CD. This machine was used as the physical host computer onto which guest virtual machines were installed.

Virtual Machines

Guest virtual machines are accomplished using a licensed version of VMware workstation 4 running on the host computer. VMware workstation software allows a single physical host computer to concurrently run a variety of guest operating systems and their applications. Host-only networking creates a network that is completely contained within the physical host computer [VM]. Configuring each virtual machine to use host-only networking allows each to inter-communicate among themselves and with the underlying physical host as though they shared the same network hub. The IP addressing for host-only networking is provided by Vmware's DHCP server service. The result of this configuration is a completely isolated networked environment of heterogeneous systems on one stand-alone computer.

As the analysis progressed, virtual machines were installed and/or configured to provide services that the malware attempted to contact.

Configuration of Windows Malware Analysis Virtual Machine

A Windows 2000 Professional virtual machine was created and installed with host-only networking. It was configured with 64MB RAM and 1.0GB disk space. Once the virtual machine had been installed the IP address was determined to be 192.168.92.128 and connectivity between physical host and virtual machine was confirmed. This machine provided an environment for analysis of Windows-based malware. The VMware Tools was installed and the display settings re-configured to 800x600.

The ILOT X CD, provided as course material, was the source of most of the utilities. An evaluation version of WinZip [WZ] was installed on the virtual machine so that other tools could be extracted from the CD and installed. The executables for the following tools were installed directly into the c:\WINNT directory of the virtual machine:

- upx.exe [UP] compressed file unpacker;

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

- md5sum.exe [MD] file checksum calculator;
- strings.exe [ST] string extraction utility; and
- nc.exe [NC1] NetCat multi-purpose network utility.

All files for these additional tools were extracted from archives on the CD into folders under c:\Program Files:

- LordPE [LP] Portable Execution (PE) file editing and dumping tool;
- Filemon [SI1] file activity monitoring utility;
- Regmon [SI2] registry activity monitoring utility;
- TDIMon [SI3] network activity monitoring utility;
- Regshot [RS] registry snapshot comparison tool;
- BinText [BT] binary to text conversion tool; and
- OllyDbg [OD1] executable debugging tool.

All files from the OllyDbg plug-in [OD2] ZIP archive on the CD were extracted into the folder for OllyDbg. Desktop shortcuts were created for each of the above tools. Finally IDA Pro [ID] was installed from the CD using its installation wizard.

As a precaution the integrity of the source file for each installed tool was verified using the md5sum utility and comparing its results to the values provided in the associated md5 text files on the CD.

Returning to the physical host, a VMware snapshot was made of the virtual machine before shutting this machine down and making a copy of the Windows 2000 Professional folder under c:\My Documents\My Virtual Machines. This provided two levels of backup from which the trusted image of the Windows virtual machine could be restored.

An additional PsKill [SI4] utility from Sysinternals was added later in the analysis process. It was obtained from the Internet to allow killing of the malware process.

Configuration of Linux Target Virtual Machine

The ILOTX CD provided the folder for a pre-configured Red Hat Linux 9.0 – REM virtual machine. It had been configured with 64MB RAM and 0.6GB disk space. The folder of files for this virtual machine was extracted into c:\My Documents\My Virtual Machines on the physical host. This RedHat-REM machine was accessed using VMware's File/Open and its host-only configuration was confirmed under its virtual machine settings. The machine was started and logged onto successfully using the root password provided in the course notes. It's IP

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

address of 192.168.92.130 was noted and connectivity with the underlying physical host was confirmed.

These pre-installed Linux system utilities were used in this analysis:

- NetCat [NC2];
- IRCD and its IRCII client [IR]; and
- Snort [SN].

Configuration of Physical Host

The NetCat (nc.exe) utility from the ILOT X CD was installed in c:\WINNT on the physical host. Figure 1 below shows the lab environment with the physical Windows 2000 host attached to a virtual hub network with its three guest virtual machines.

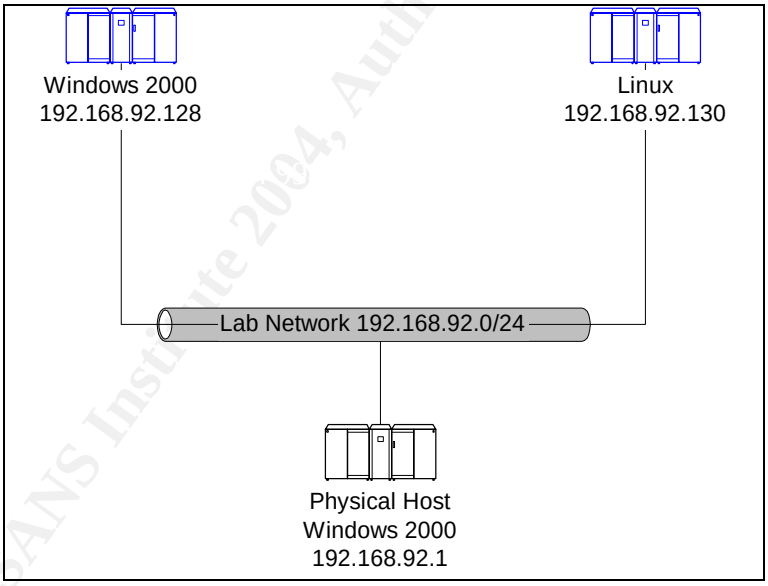

Figure 1 Isolated Malware Analysis Laboratory Network

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

3 PROPERTIES OF THE MALWARE SPECIMEN

The specimen of malware was provided in a password-protected archive. This archive file was downloaded from the course site and transferred to CD for transfer to the isolated analysis computer. The specimen was extracted to the Windows 2000 virtual machine. The Windows file properties of the extracted file (msrll.exe) indicated it was an application of size 41,984 bytes.

The bintext utility provided the following strings in the malware specimen:

0040004D !This program cannot be run in DOS mode.
00400178 .text
004001A0 .data
004001F0 .idata
00400218 .aspack
00400240 .adata

The following strings were found at the end of the file:

0051D071 VirtualAlloc
0051D07E VirtualFree
0051D441 kernel32.dll
0051D44E ExitProcess
0051D45A user32.dll
0051D465 to 0051DF6C showed various messages, system calls and data
0051EC81 msvcrt.dll
0051E08C msvcrt.dll
0051E097 shell32.dll
0051E0A3 user32.dll
0051E0AE version.dll
0051E0BA wininet.dll
0051E0C6 ws2_32.dll
0051E113 to 0051E192 showed various messages, system calls and data

The suspicion that msrll.exe had been compressed was confirmed by loading the file for OllyDbg analysis. OllyDbg statistical testing suggested that the code section might have been compressed or encrypted.

An MD5 hash of the file was obtained using md5sum.exe. Its value was 84acfe96a98590813413122c12c11aaa.

Opening the malware specimen with the DOS editor showed the first two bytes as “MZ” which indicated that the file was most likely an executable. The first

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

string of the file and those at the end of the file indicated msrll.exe was likely a Windows executable.

4 BEHAVIORAL ANALYSIS

The behavioural analysis of the specimen began by using the RegShot utility to determine the changes made to the target Windows system by the specimen. This was followed by a more detailed analysis using monitoring tools to track changes to the file system, changes to the registry and any network activity. Initial observations were drawn from the results of this monitoring and the laboratory environment was adjusted to accommodate the expectations of the specimen. This process of giving the malware what it expects then observing the results was repeated until it was felt that little more could be obtained from such observation.

Changes Made by Malware

A snapshot of the Windows 2000 virtual machine was made using Regshot.exe before the malware specimen was first executed. The *scan dir* parameter was set to c:\ for this first shot.

In a DOS command window the *netstat –a* command was run to identify the initial network connections before infection. The Task Manager monitoring utility was activated and the malware specimen was executed by double-clicking on its file icon. A new entry (msrll.exe) was added to list of active processes. After giving the malware sufficient time to contaminate the machine (45 seconds) the msrll.exe process was selected in the process list and its processing was stopped. A second Regshot capture was made and compared to the first to give the following important changes:

Table 1 Summary of Changed Values

Change	Description	
Key added	HKLM\SYSTEM\ControlSet001\Services\mfm	
	HKLM\SYSTEM\ControlSet001\Services\mfm\Security	
	HKLM\SYSTEM\CurrentControlSet\Services\mfm	
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\Security	
Change	Description	Value
Key Values Added	HKLM\SYSTEM\ControlSet001\Services\mfm\Security\Security	String of hex values
	HKLM\SYSTEM\ControlSet001\Services\mfm\Type	0x00000120
	HKLM\SYSTEM\ControlSet001\Services\mfm\Start	0x00000002
	HKLM\SYSTEM\ControlSet001\Services\mfm\ErrorControl	0x00000002
	HKLM\SYSTEM\ControlSet001\Services\mfm\ImagePath	“C:\WINNT\System32\mfm\msrll.exe”

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

	HKLM\SYSTEM\ControlSet001\Services\mfm\DisplayName	“Rll enhanced drive”
	HKLM\SYSTEM\ControlSet001\Services\mfm\ObjectName	“LocalSystem”
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\Security\Security	String of hex values
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\Type	0x00000120
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\Start	0x00000002
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\ErrorControl	0x00000002
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\ImagePath	“C:\WINNT\System32\mfm\msrll.exe”
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\DisplayName	“Rll enhanced drive”
	HKLM\SYSTEM\CurrentControlSet\Services\mfm\ObjectName	“LocalSystem”
	HKU\S-1-5-21-507921405-1614895754-1801674531-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU:P:\Qbphzragf naq Frggvatf\ive20001\Zl Qbphzragf\zfeyy.rkr	02 00 00 00 06 00 00 00 40 A9 D9 F8 E3 8A C4 01
Key Values Modified	HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed	Changed twice
	HKU\S-1-5-21-507921405-1614895754-1801674531-1000\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{75048700-EF1F-11D0-9888-006097DEACF9}\Count\HRZR_EHACNGU	Changed twice
	HKU\S-1-5-21-507921405-1614895754-1801674531-1000\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections\SavedLegacySettings	Changed twice
Change	Description	
Files Added	C:\WINNT\system32\mfm\jtram.conf	
	C:\WINNT\system32\mfm\msrll.exe	
Files Deleted	C:\Documents and Settings\vir20001\My Documents\msrll.exe	
Files [attributes?] Modified	C:\Documents and Settings\vir20001\Cookies\index.dat	
	C:\Documents and Settings\vir20001\Local Settings\History\History.IE5\index.dat	
	C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files\Content.IE5\index.dat	
	C:\Documents and Settings\vir20001\NTUSER.DAT	
	C:\Documents and Settings\vir20001\ntuser.dat.LOG	
	C:\WINNT\system32\config\software	
	C:\WINNT\system32\config\software.LOG	
	C:\WINNT\system32\config\system	
	C:\WINNT\system32\config\SYSTEM.ALT	
Folder Added	C:\WINNT\system32\mfm	
	C:\WINNT\system32\mfm\.	
	C:\WINNT\system32\mfm\..	

The reasons for modifying all the keys and file attributes listed in Table 3 were not immediately obvious. The creation of the new “c:\WINNT\system32\mfm” folder and dropping of two new files into this folder was noted. The removal of the original infecting malware file was also noted. Attempts to access some of the

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

files (eg., NTUSER.DAT and ntuser.dat.log under the vir20001 user folder) indicated some of the key settings applied were self-protective access controls.

The *netstat –a* command was re-run on the infected Windows virtual machine to see what new connections might have been established by the malware. It was noted that two new TCP connections were now in the listening state on TCP ports 113 and 2200.

Secondary Monitoring

The original VMware snapshot was restored and the malware specimen extracted from CD in preparation for more extensive analysis. Monitoring was started using the Filemon, Regmon and TDImon utilities on the Windows 2000 virtual machine. The monitoring for each was paused while the windows were sized and positioned so that the folder containing the malware specimen remained visible. The task manager was activated before re-activating each of the paused monitoring. Finally the malware specimen (msrll.exe) was executed for another 45 seconds before stopping the capturing of events by the three monitoring utilities and ending the msrll.exe process.

It was noted that the file for the malware specimen was removed from its original location. The Regmon monitoring showed that msrll.exe checks and often changes many registry settings, some of the more interesting of which are summarized in the following table:

Table 2 Summary of Regmon Monitored Events

Process	Action	Key	Status
msrll.exe	Create	HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer	SUCCESS
msrll.exe	Create	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings	SUCCESS
services.exe	CreateKey	HKLM\System\CurrentControlSet\Services\mfm	SUCCESS
services.exe	CreateKey	HKLM\System\CurrentControlSet\Services\mfm\Security	SUCCESS
services.exe	SetValue	HKLM\System\CurrentControlSet\Services\mfm\Security\Security	SUCCESS
msrll.exe	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG (multiple times)	SUCCESS
msrll.exe	Enumerate Value	HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Prefixes\ftp	SUCCESS "ftp://"
msrll.exe	Enumerate Value	HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Prefixes\gopher	SUCCESS "gopher://"
msrll.exe	Enumerate Value	HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Prefixes\home	SUCCESS "http://"
msrll.exe	Enumerate Value	HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Prefixes\mosaic	SUCCESS "http://"
msrll.exe	Enumerate Value	HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Prefixes\www	SUCCESS "http://"

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Process	Action	Key	Status
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints\C	SUCCESS
msrll.exe	Create	HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer	SUCCESS
msrll.exe	Create	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders	SUCCESS
msrll.exe	CreateKey	KLM\Software\Microsoft\Tracing	SUCCESS
msrll.exe	QueryValue	HKLM\Software\Microsoft\Tracing\RASAPI32\FileDirectory	SUCCESS "%windir%\tracing"
msrll.exe	CreateKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders	SUCCESS
msrll.exe	CreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\Parameters	SUCCESS
msrll.exe	CreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\Parameters	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS
msrll.exe	CreateKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings	SUCCESS
msrll.exe	SetValue	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyEnable	SUCCESS
msrll.exe	DeleteValueKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyServer	NOTFOUND
msrll.exe	DeleteValueKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyOverride	NOTFOUND
msrll.exe	DeleteValueKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\AutoConfigURL	NOTFOUND
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\Connections	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\Internet Settings\Connections	SUCCESS

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Process	Action	Key	Status
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\ Internet Settings	SUCCESS
msrll.exe	CreateKey	HKCC\Software\Microsoft\windows\CurrentVersion\ Internet Settings	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\ Internet Settings\Connections	SUCCESS
msrll.exe	CreateKey	HKCU\Software\Microsoft\windows\CurrentVersion\ Internet Settings\Connections	SUCCESS
msrll.exe	CreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\P arameters	SUCCESS
msrll.exe	CreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\P arameter	SUCCESS
msrll.exe	CreateKey	HKLM\System\CurrentControlSet\Services\Tcpip\P arameters	SUCCESS
msrll.exe	CreateKey	HKLM\SOFTWARE\Microsoft\Cryptography\RNG (many times)	SUCCESS
msrll.exe	OpenKey	HKLM\SOFTWARE\Microsoft\Cryptography\Default s\Provider\Microsoft Base Cryptographic Provider v1.0	SUCCESS
msrll.exe	QueryValue	HKLM\SOFTWARE\Microsoft\Cryptography\Default s\Provider\Microsoft Base Cryptographic Provider v1.0\Image Path (4 times)	SUCCESS
msrll.exe	QueryValue	HKLM\Software\Microsoft\Cryptography\MachineGu id (4 times)	SUCCESS

Again the reasons for most of these registry manipulations were not readily apparent at this stage of the analysis. As noted before, some appear to set strong access controls on files the malware uses. The Filemon monitoring revealed the interesting activities summarized in the table below:

Table 3 Summary of Filemon Monitored Events

Process	Action	Key	Status
msrll.exe	Create	C:\WINNT\System32\mfm	SUCCESS
msrll.exe	Create	C:\WINNT\System32\mfm\msrll.exe	SUCCESS
msrll.exe	Write	C:\WINNT\System32\mfm\msrll.exe	SUCCESS
msrll.exe	Set Information	C:\WINNT\System32\shell32.dll	SUCCESS
msrll.exe	Set Information	C:\WINNT\System32\mfm\msrll.exe	SUCCESS
msrll.exe	Delete	C:\Documents and Settings\vir20001\My Documents\msrll.exe	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\History	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files\Content.IE5\	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Process	Action	Key	Status
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Cookies\	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Cookies\index.dat	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\History\History.IE5\	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\History\History.IE5\index.dat	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files\Content.IE5\	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\Local Settings\History\History.IE5\	SUCCESS
msrll.exe	Set Information	C:\Documents and Settings\vir20001\ntuser.dat.LOG	SUCCESS
msrll.exe	Query Information	C:\WINNT\System32\rsabase.dll	SUCCESS
msrll.exe	Set Information	C:\WINNT\system32\config\software.LOG	SUCCESS Length: 4096
msrll.exe	Set Information	C:\WINNT\system32\config\software.LOG	SUCCESS Length: 8192
msrll.exe	Open	C:\dev\random (many times)	NOT FOUND
msrll.exe	Write	C:\WINNT\system32\mfm\jtram.conf (many sequential writes)	SUCCESS

This file activity indicates which files are important to the malware's operations. After the malware had been running for a while it carried out periodic queries of the length of the hidden system file C:\Documents and Settings\vir20001\Local Settings\Temporary Internet Files\content.IE5\index.dat. This may be used for logging of status. The TDImon monitoring showed attempted UDP probing of the physical host (192.168.92.1) on port 53 (DNS).

The Windows 2000 virtual machine was restarted and the Task Manager was used to determine that msrll.exe was running. Attempts to end this process failed due to access controls applied by the malware during its infection process. This was overcome by using the PsKill utility to successfully stop the process.

The C:\WINNT\system32\mfm\jtram.conf file created by the malware was opened for editing. It appeared to be scrambled or encoded in some way. By the extension it appeared to be a configuration file. The contents of this file are shown in Table 4 however the original format had a table-like structure of three columns and six rows.

Table 4 First Version of jtram.conf

```
CAARAMsx4wHZq1JKZ3pnbtbqpP9XHbDnzZP6cnaOqpkOOQi6tw==
QAIRABNuE0hETp5PKLxeZwi4CEnA9+ujR0UHuUDn2k4wfejV0A==
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
c/8RACx04pna+dK312xzuyoG8T7CoduakxUk06aAejA2xmUJag==
nQARAGNaync3y7HnndgQk/U2XOJ5P+xIKZ5A/CWnLb0yjwauWQ==
VP8RADfdkAznDweBRVPhMreMMHGp6sRzUazkINo1ZPTt+s+FGQ==
5v0RAIHf5zrQT0eHXeFkqOqksV0XttyIq+ypQdQJ1jUWjtQTXA==
SAARALBrthRXUmYwJGFt1WmsvKSR6ocl+oYU77dS9/Y8PsLIMw==
tgARAG7HnuSHy1wdim8XjNVQ+58C9fdsWUSZ0zbNayAOIXQGLg==
cgFKAKoWujcgHZ4Jwm8Z5Qh3F7FMy6DC4OxuBZN3YLPsXtWo8iQSdnLVa97QQpoNNU9Ykrcml+/x
dKD9GqNZJN8QRNbm1e34CifqmkZYy88FOSOXjlZd4F/kw/rcag==
a/4RAA1xUAYY/gSTk/zMBV7inH7cETQU5aY5mv2IpgMH/sSBFA==
UwARAB4ZLmEiME4x0FvAHSAOg7CzqZB/nzxoh/B28MXFLQ8izw==
jf4RAKzzTs8xuuyPYGjDqiiVV/QfQfKKCMsGHVqSghpkrQYckA==
xP8RAHtNmGn0IQfOIjK3mWbRDH2fx7p/W0meUZ2VbrONjnif5Q==
GgARAB4ebEfaNXfu5JUQPkBfbwG1/dc3w4qh7AzQpAh2nM8NwA==
vv8jAM+IAND3LC9i9slEdpRFhFzAEhI5+LLlcezdoSsOk6KlhFQX7m6kb3Ni7JlyTpKhBW8E0Q==
YP8RAIVqhybFU3U101LKwDBVs5qIpewS4UV/xQOSBiMzARDd/A==
kf4RAOUGXbUbEm5hBOGVdgCA9za6waE0u8PEnCuI7hIGdLFbgw==
UwIjAHjTfmcoeL5cZT7brDNa+6dVI+PHxVksDcYdiTgxUcW6PqywMSt54OofcdB4oKTsxA8aoQ==
```

This file was renamed and the malware was re-run. A new version of jtram.conf was created, this time with differently coded contents with the same basic structure. This indicates that the malware probably requires the presence of this file.

Monitoring Network Activity

The Linux virtual machine was started and the snort utility was run to capture network activity using the command *snort –vd | tee <filename>*. The Windows virtual machine was reverted to its clean state and re-infected by repeating the extraction and running of msrll.exe. The infected machine quickly identified that physical machine was present on the network and sent a DNS query for "collective7.zxy0.com".

The Trojan specimen was killed using the DOS command *pskill msrll.exe*. The entry "192.168.92.130 collective7.zxy0.com" was added to the c:\WINNT\system32\drivers\etc\hosts file on the infected machine. The snort monitoring on the Linux machine was re-started using a different filename and the Trojan executable was run from its new location of "\WINNT\system32\mfm\msrll.exe". The snort logs showed the expected ARP and NetBIOS broadcasts as well as a repeated pattern of network activity between the infected Windows virtual machine and the Linux virtual machine. These were at IP addresses 192.168.92.128 and 192.168.92.130 respectively. The infected machine expected collective7.zxy0.com to be at 192.168.92.130. This repeated polling activity is shown in Table 5 below.

Table 5 Summary of Repeated Network Activity

Source Address	Protocol	Port	Destination Address	Port	Description
192.168.92.128	TCP	1070	192.168.92.130	6667	Infected machine contacts IRC service on Linux machine

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Source Address	Protocol	Port	Destination Address	Port	Description
192.168.92.130	TCP	6667	192.168.92.128	1070	No service—reset connection
The above attempted dialogue occurred twice more					
192.168.92.128	TCP	1071	192.168.92.130	9999	Attempt to contact unknown service
192.168.92.130	TCP	9999	192.168.92.128	1071	No service—reset connection
The above attempted contact occurred twice more					
192.168.92.128	TCP	1072	192.168.92.130	8080	Attempt to contact port 8080
192.168.92.130	TCP	8080	192.168.92.128	1072	No service—reset connection
The above attempted contact occurred twice more					

Capturing Service Activities

The probes of TCP port 6667 were likely attempts to contact an IRCD server. The IRCD service was started on the Linux virtual machine to confirm this. The logged in root account was switched to the ircd user (*su – ircd*), the IRC service was started (*./ircd*) and control was returned to the root account (*exit*). Snort monitoring showed that the infected machine joined the #mils channel.

Table 6 Joining IRC #mils Channel

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
08/28-19:57:11.607975 192.168.92.128:2471 -> 192.168.92.130:6667
TCP TTL:128 TOS:0x0 ID:19956 IpLen:20 DgmLen:53 DF
***AP*** Seq: 0xC24569EC  Ack: 0xF456862D  Win: 0x4423  TcpLen: 20
4A 4F 49 4E 20 23 6D 69 6C 73 20 3A 0A          JOIN #mils :.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

Restarting the infected Windows machine several times showed that the usernames and nicknames used to join this channel were different each time and appeared to be comprised of random letters.

The IRC server was stopped and NetCat was run and the following user and nickname announcements were captured on TCP port 6667 using the command *nc –p <port #> –l –n .*

USER ZftQReBb localhost 0 : MhemUxTTxIxEUr
NICK   uScZBauPzktW

NetCat captures on ports 9999 and 8080 showed the same pattern of user and nickname transmissions with random upper and lower case values. This indicates that msrll.exe is looking for IRC servers on all three ports (6667, 9999 and 8080) and it confirms that the malware repeated its probed for these services.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The IRC server was stopped, the *ircd.conf* file was altered to switch the port on which it listens from 6667 to 9999 and the IRC service was re-started. Snort monitoring was used to observe the activities of the malware when it was re-run. The steps of this test were repeated, this time changing the IRC port from 9999 to 8080. The malware joined the same #mils channel for all three IRC ports. There did not appear to be a password involved in any of these attempts. The malware may use some other security mechanism to protect this channel.

The malware specimen was stopped using PsKill. The IRC server was restarted to provide services on TCP port 6667 and the malware was re-run. The IRC service was accessed with a Linux client (*irc*) and the channel was joined successfully (*/join #mils*). The presence of the malware was confirmed with a *who* command (/who #mils) but attempts to contact this other user (eg., */msg #mils hello there*, */msg <nick> hello there*) elicited no response.

Network monitoring recorded that when the malware joined the #mils channel the IRC server looked up the hostname of the source using authentication port 113 on the infected machine. Once the lookup had been successfully completed, *netstat* showed the infected machine eventually removed this active port. This is shown below in the *netstat - a* monitoring snapshot:

Table 7 Authentication by IRC Server

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
09/05-23:48:12.906718 192.168.92.130:1024 -> 192.168.92.128:113
TCP TTL:64 TOS:0x0 ID:56011 IpLen:20 DgmLen:65 DF
***AP*** Seq: 0x4DDEBFF3  Ack: 0x76FE5997  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 25928 0
31 33 30 37 20 2C 20 36 36 36 37 0D 0A          1307 , 6667..
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/05-23:48:12.914351 192.168.92.128:113 -> 192.168.92.130:1024
TCP TTL:128 TOS:0x0 ID:1195 IpLen:20 DgmLen:97 DF
***AP*** Seq: 0x76FE5997  Ack: 0x4DDEC000  Win: 0x4463  TcpLen: 32
TCP Options (3) => NOP NOP TS: 120617 25928
31 33 30 37 20 2C 20 36 36 36 37 20 3A 20 55 53  1307 , 6667 : US
45 52 49 44 20 3A 20 55 4E 49 58 20 3A 20 6F 62  ERID : UNIX : ob
71 6D 4A 78 5A 46 53 76 6A 59 4F 6D 0A          qmJxZFSvjYOm.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

There were also repeated “PING” and “PONG” message exchanges from the server to the client to maintain the IRC session.

Table 8 IRC Server Session Keep Alive

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
08/28-20:00:14.446451 192.168.92.130:6667 -> 192.168.92.128:2471
TCP TTL:64 TOS:0x0 ID:253 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0xF4568834  Ack: 0xC2456A0E  Win: 0x16D0  TcpLen: 20
50 49 4E 47 20 3A 6C 6F 63 61 6C 68 6F 73 74 2E  PING :localhost.
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
6C 6F 63 61 6C 64 6F 6D 61 69 6E 0D 0A          localdomain..
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
08/28-20:00:14.455177 ARP who-has 192.168.92.130 tell 192.168.92.128

08/28-20:00:14.456546 ARP reply 192.168.92.130 is-at 0:C:29:6A:59:5F

08/28-20:00:14.457940 192.168.92.128:2471 -> 192.168.92.130:6667
TCP TTL:128 TOS:0x0 ID:19960 IpLen:20 DgmLen:68 DF
***AP*** Seq: 0xC2456A0E  Ack: 0xF4568851  Win: 0x41FF  TcpLen: 20
50 4F 4E 47 20 3A 6C 6F 63 61 6C 68 6F 73 74 2E  PONG :localhost.
6C 6F 63 61 6C 64 6F 6D 61 69 6E 0A          localdomain.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

Since it was noted that the IRC client indicated the localhost address, the /etc/hosts table on the Linux guest machine was edited to remove localhost (127.0.0.1). The IRC client had to use the */server 192.168.92.130* command to connect. As a result the client session showed the IP address of collective7.zxy0.com. Unfortunately this made no noticeable difference to the malware; it still did not respond to messages or even to attempts at direct communications using */msg* or */dcc chat* commands. The localhost address entry was restored.

The malware's use of TCP ports 113, 6667, 9999 and 8080 had been verified. This left TCP port 2200 to be investigated. A connect to this port was established using the command *telnet 192.168.92.128 2200* from the Linux machine. Upon successful connection the prompt "#:" was displayed. Typing anything failed to elicit a response.

A copy of the malware msrll.exe file was extracted to a floppy disk and inserted into a stand-alone computer protected by McAfee Anti-Virus [NA1]. The virus scanner automatically detected and cleaned the file a:/msrll.exe. The label associated with the malware was "BackDoor-CGM Trojan" [MA2]. The McAfee description of this Trojan was very limited, indicating only that it was of unknown origin, was a remote access sub-type and had no known aliases.

5 CODE ANALYSIS

The malware appeared to be packed or encrypted. A copy was made of the msrll.exe specimen before attempting to unpack it with the *upx* unpacker supplied on the course CD (*upx -d msrll-new.exe*). The utility indicated that the specimen was not packed by *upx*.

The earlier text analysis showed no strings that indicated the "upx" packer had been used but there was an "aspack" string. Assuming that this meant the

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

ASPack compression program [AP1] had been applied to compress and obscure the executable, attempts were made to find a native decompression that would work for the type and version of compression used on msrll.exe. The AspackDie 1.3d utility [AD] was found to successfully unpack the specimen. The decompression was carried out with a simple file selection operation using the utility's graphical user interface. Since a native unpacker was found it was not necessary to use the OllyDbg utility to dump the running image of the executable for analysis.

The bintext utility was used to analyse the decompressed executable. The following is a list of the observations that could be made about the malware specimen from its strings:

- Several strings (eg., "smurf" [AO1] and "jolt" [AO2]) indicated the malware could have a comprehensive set of denial of service tools
- A long list of what appeared to be commands included such interesting actions as: "?rmdir", "?clone", "?clones", "?login", "?reboot", "?update", "?exec", "?kill", "?killall" and "?crash"
- A section of the code had the strings "?insmod", "?rmmod", and "?lsmod" which indicated an ability to modify Linux kernels
- There appeared to be extensive IRC DCC commands for sending and receiving files
- There appeared to be extensive cryptographic capabilities including SSL-protected communications, twelve crypto algorithms, eight different built-n hash algorithms, three separate block chaining modes, several pseudo-random number generators and public key (PK) algorithms
- The "GCC compiler detected" string indicated the malware could possibly detect the presence of a GCC compiler
- There appeared to be a version indication of "m220 1.0 #2730 Mar 16 11:47:38 2004"
- There was the string "Mozilla/4.0" which might be a client indicator in HTTP requests

IDAPro was used to analyse the decompressed executable, develop a comprehensive list of strings and provide contextual information that could be referred to during the actual debugging of the AspackDie unpacked executable using OllyDbg.

Debugging a complex specimen of code without source code is a matter of building up a larger picture of its operations using many successive observations of how small parts of the program work. The debugging started by letting the executable run on its own then pausing execution (F12) and stepping over (F8) or sometimes into (F7) the code, watching what gets put on the stack or into registers. It was found that this main program loop begins at 0040C438. IDA was used to determine that the loop involves the following procedural steps:

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Table 9 Mail Code Loop of Malware

loc_40C438:	Call PeekMessageA – obtain a message if one exists in the message queue of the calling thread [MS1]
	Jump to loc_40C49D if no message
	Call GetMessageA – wait for and obtain a message from the message queue [MS2]
	Jump to loc_40C49D if no message
	Call DispatchMessageA – send the message data to a windows procedure [PM]
loc_40C49D:	Call sub_40D8CE – operations of subroutine A to be determined
	Jump to loc_40C361 based on a result of the previous subroutine
	Call sub_40426D – operations of subroutine B to be determined
	Call msvcrt.time procedure [MS3] – returns elapsed time
	Jump to loc_40C4F8 based on a result of the previous subroutine
	Call sub_40492B – operations of subroutine C to be determined
	Call sub_404B8F – operations of subroutine D to be determined
	Call sub_40A7B3 – operations of subroutine E to be determined
	Call msvcrt.time procedure – returns elapsed time
loc_40C4F8:	Call msvcrt.time procedure – returns elapsed time
	Jump to loc_40C438 based on a result of the previous subroutine
	Call msvcrt.time procedure – returns elapsed time
	Call sub_409A30 – operations of subroutine F to be determined
	Jump to loc_40C438 (start of main loop)

The jump to location loc_40C361 in the code made a call to an earlier part of the program (loc_40171F) and returns. This indicated that the malware looped waiting to receive some sort of communications. While it waited it appeared to carry out activities based on a schedule.

The next step in the debugging was to investigate each subroutine further. First the jtram.conf file was removed to observe which subroutine would replace the file. Each subroutine was stepped into (F7) and the code was executed until return (Ctrl + F9) repeatedly. Notes were made of the stack values.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

The calls to subroutines A, B, C and E (see Table 9 above) were "stepped into" using F7 and allowed to run until each return using <Ctrl + F9>. This was repeated until control was returned to the main loop. These subroutines appeared to do little when there was no network environment (eg., IRC or port 2200) with which to interact. Prior to the call to sub_40D8CE the stack had the UNICODE value of "WS2HELP.DLL". WS2HELP.DLL contains functions used by the Windows Sockets API used by Internet and network applications [LI].

Subroutine D (loc_40C4DD)
The following stack values were noted during debugging of this subroutine:

- ASCII "C:\WINNT\system32\mfm\msrll.exe"
- UNICODE "RASAPI32.DLL", "RASMAN.DLL",
- UNICODE "C:\WINNT\system32\mfm;.;C:\WINNT\system32;C:\WINNT\system; C:\WINNT;C:\WINNT\system32;C:\WINNT;C:\WIN"
- UNICODE "TAPI32.DLL", "RTUTILS.DLL", "RasPbFile"
- ASCII "Settings\Vir20001\Local Settings\History\History.IE5\MSHist012004091120040912"
- UNICODE "sensapi.dll", "ntdll.dll", "USERENV.DLL", "USER32.DLL"
- ASCII "ExpandEnvironmentStringsForUserW"
- UNICODE "%ALLUSERSPROFILE%\Application Data"
- ASCII "{W#"
- UNICODE "netapi32.dll", "SECUR32.DLL", "NETRAP.DLL"
- UNICODE "AP.DLL", SAMLIB.DLL", Wbem", WLDAP32.DLL"
- UNICODE "DNSAPI.DLL", WSOCK32.DLL", VIR2000"
- UNICODE "C:\Documents and Settings\All Users\Application Data"
- ASCII "C:\WINNT\system32\winrnr.dll"
- UNICODE "%SystemRoot%\system32\winrnr.dll"
- ASCII "collective7.zxy0.com"
- UNICODE "rasadhl.dll", "File Directory"
- ASCII "s_check: trying %s"

Subroutine F (loc_40C51F)
The following stack values were noted during debugging of this subroutine:

- ASCII "PE"
- UNICODE "rsabase.dll"
- UNICODE "C:\WINNT\system32\mfm;.;C:\WINNT\system32;C:\WINNT\system; C:\WINNT;C:\WINNT\system32;C:\WINNT;C:\WIN"
- UNICODE "ole32.dll", "CRYPT32.dll", "MSASN1.DLL", "NTDLL.DLL"
- ASCII "Microsoft Base Cryptographic Provider v1.0"
- ASCII "d/8RAAbq2j7J/KK/jtAwjOhCydI2V6466uYLNNoCrAbvVix7rA=="
- ASCII "DiCHFc2ioiVmb3cb4zZ7zWZH1oM="

During the execution of subroutine F a new jtram.conf file with different values was dropped into the mfm folder. Looking at the list of values shown above and knowing that the jtram.conf file was re-created made it appear likely that this subroutine carried out the infection of the system. The last two ASCII values in the list were probably key values for the cryptographic algorithm. Restarting the

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

malware and retracing the above process showed that the first ASCII value (key) changed but the second stayed constant.

Debugging with the IRC service on TCP port 6667 available on the Linux guest machine allowed further analysis to see how this changed the operation. Subroutine D carried out activities involving “RPCRT4.DLL” when it could interact with IRC.

Attempts were made to step through with breakpoints (F2) on the main subroutines and observe the IRC interaction. Unfortunately the delays involved made debugging difficult. Instead a search was made to find the “#:” prompt used on TCP port 2200. This was found at 0040BD04. A breakpoint was set at 0040BD10, after the routine that issues the prompt and the malware was allowed to run.

When Telnet was used successfully to contact the infected machine’s port 2200, the breakpoint was triggered. Single stepping (F8) through subroutine after this location (at 0040BD27) showed that register eventually contained the string “bot.port: connect from 192.168.92.130”. Register EDI contained “***”. This confirmed that the malware detected the contact on port 2200. It also supported the likelihood that the malware was an IRC “bot”.

There was no success in eliciting a response from the malware on TCP port 2200 using Telnet. Reviewing strings from this and other areas of the code provided possible strings (eg., pass, jtr.bin, msrll.exe, jtr.home, mfm, 220, jtr.id, run5, irc.quit). The “***” string noticed in the register was also tried to no avail. The connection consistently accepted two lines of input before closing. The code was unsuccessfully searched for the appropriate condition that would unlock access.

The names such as “jtr.bin”, “jtr.home”, “jtr.id”, and “ïrc.quiet” that were noticed in the code were intriguing. It was unclear whether these identified file names or referred to variables. A reference to “jtr.*” in the code supported the fact they might be filenames, however a search of the infected system did not find any files with “jtr” before any extension. A search of IRC references did not reveal any use for an “irc.quit” file. If they were referring to variables, the name “jtr” must have been important to the code’s author(s). They could even be the initials of the original author.

Breakpoints were set on the calls to the three message routines (ie., PeekMessageA, GetMessageA) and at the top of the main loop to attempt to catch reception of incoming messages submitted via the IRC or Telnet channels. This became a very frustrating and time-consuming process that was eventually abandoned. Single stepping through the code altered the timing of communications and made it difficult to synchronize the submission of a

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

message from the Linux client applications and detection by the malware executable.

The key (as it were) to this malware appeared to involve the jtram.conf file. IDA was used to search for references to the filename and identify the location where this file is created (00409FD5). Setting a breakpoint at this location in OllyDbg, removing the file from its "mfm" folder, restarting debug execution and then stepping through the code allowed tracking of activities involving the "jtram.conf" file.

Observing stack values with breakpoints at locations 00409DE6 and 00409DF4 caught the contents prior to their being written to the new "jtram.conf" file and revealed the values they represented. Arranging these values into the original structure (three columns by six rows) resulted in following mapping for values in the file (see Table 10):

Table 10 Map of Values in Configuration File

Command	Variable	Value
set	bot.port	2200
set	irc.quit	<blank>
set	servers	collective7.zxy0.com,collective7.zxy0.com:9999!,collective7.zxy0.com:8080
set	irc.chan	#mils
set	pass	1KZLPLKDf$W8kl8Jr1X8DOHZsmIp9qq0
set	dcc.pass	1KZLPLKDf$55isA1ItvamR7bjAdBziX.

This mapping of values in Table 10 confirmed many of the characteristics observed in the analysis. Some sort of channel, named the "bot.port. can be used to communicate with the malware on port 2200. The malware looks for servers on three separate ports (default 6667, 9999 and 8080) at "collective7.zxy0.com". The malware uses IRC channel "#mils". The use of the value "irc.quit" is unknown. The "dcc.pass" name indicates there may be a password for IRC DCC communications. The remaining "pass" variable may indicate there is a password for port 2200.

The two passwords were of most interest. The lengths and complexity of these passwords, along with the length of the IRC nicknames would make it a painful process to establish communication sessions unless some utility can be used to do this automatically.

OllyDbg was used to create simpler passwords within a new "jtram.conf" file in the hope that the passwords could be used to finally establish successful remote access to the malware. The executable was restarted with the pre-existing breakpoints in place to catch writes to the file. The "jtram.conf" file was deleted and debugging run until the first breakpoint. The debugging was re-run until the

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

stack pointers to the passwords could be recorded as 003F53C0 and 003F5498 for “pass” and “dcc.pass” respectively. The above steps were repeated until first breakpoint for writing to the file.

The memory map was activated and the memory range 003F0000 was selected for editing. All but the first three values (“1”) of the passwords were selected for editing and filled with zeros using the right-click activations for the selection. Execution was continued but the values on the stack did not get altered.

The memory range was revisited and the two original passwords were found to be also at locations 003F39F8 and 003F3A88. The steps were repeated again, this time zero filling all but the first three characters for all four locations. This time stepping through the writes to the file appeared to set the value “1” for both “pass” and “dcc.pass”. The “jtram.conf” had now been configured to suit further manual testing.

Unsuccessful attempts were made to communicate with the malware using IRC messages to the channel (eg., */msg #mils login 1*) and private messages (eg., */msg <nick> login 1*). Telnet sessions to the malware were established on port 2200 and attempts to elicit a response were attempted (eg., *login 1*), again unsuccessfully. Lenny Zeltser’s SANS Malware FAQ [LZ] indicated communications with another malware specimen required that the password precede commands so the tests on IRC and Telnet sessions were attempted with the “1” password first. There was still no success gaining a response. The Malware FAQ analysis also indicated that communications had to be encrypted as well. Possibly this is also the case for msrll.exe. This was set aside as something for later analysis when there was more time.

6 ANALYSIS WRAP-UP

The malware specimen appears to be a complex Windows Trojan (msrll.exe) with an extensive set of potentially malicious capabilities ranging from controlling a victim system to killing its processes and launching distributed denial of service (DdoS) attacks against other systems. The common IRC channel for all infected systems could be used to create a network of malicious “bots” that can launch coordinated attacks from a large number of sources, making the attacks more difficult to defend against. People use such attacks to extort money from businesses [SO1], to make some statement [WN], to gain stature within their community or just for fun [SW].

The Trojan removes the infecting file, installs itself within a system directory and configures registry settings to ensure that it always starts upon system start-up. It

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

protects itself with access controls to prevent the killing of its running process and it periodically takes steps to ensure that its executable and an associated file remain in the system folder, even if deleted. The associated “jtram.conf” file is a cryptographically obscured configuration file. This file sets encrypted configuration parameters for the malware operation (eg., bot.port set to 2200 and irc.chan set to “#mils”). The “bot.port” value has the malware listen for incoming communications from a remote user on port 2200.

Strings within the executable indicate the Trojan has an extensive set of commands. The most interesting of these appear to be the infection of remote Windows systems (ie., “?clone”) and some ability to manipulate Linux/Unix systems (ie,. “?insmod”, “?rmmod” and GCC compiler detection).

The analysis indicated that theTrojan would likely respond to specific commands via the #mils channel on IRC served up by collective7.zxy0.com as well as directly on TCP port 2200. It is highly likely that the commands require a password however testing with control over the passwords was still unsuccessful. It is quite possible that the communications sessions must be properly encrypted.

While interaction with the malware was unsuccessful, the analysis did provide sufficient information to use in protecting my organization. Certainly the fact that our corporate McAfee virus scanner detected its signature is encouraging. Should the Trojan escape this detection, its telltale use of IRC services and communications to TCP port 2200 would be blocked by the highly restrictive policies enforced by our corporate perimeter firewalls. Outbound IRC is not allowed. The firewalls block all but selected the incoming ports necessary to carry out business.

Since malware authors will re-use code to create new variants with different signatures and communications channels, it is important to reduce the chance of infection by controlling the methods of infection. Given that the corporate firewalls block all peer-to-peer and instant messaging activity, most viruses and Trojans are likely to arrive either as e-mail attachments or as downloads from web sites. Users need to be reminded about proper precautions for their handling of e-mail attachments and file downloads. They also must be aware of the policy to report suspect or unusual activities on their computer to the corporate Information Protection Centre.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

REFERENCES

[VM] Vmware, Inc. “Vmware Workstation 4 User’s Manual” v4.5.2. URL: http://vmware-svca.www.conxion.com/software/ws45_manual.pdf . 6 September 2004.

[WZ] Winzip Computing, Inc. Evaluation copy of WinZip 9.0 for Windows systems. URL: http://www.winzip.com . 6 September 2004.

[UP] Oberhumer, Markus. Molnár, László. UPX version 1.21 for Windows operating systems. URL: http://upx.sourceforge.net/download/00-OLD-VERSIONS . 6 September 2004.

[MD] A port of the GNU "md5sum" utility to Windows. Extracted from the distribution file UnxUtils.zip. URL: http://www.weihenstephan.de/~syring/win32 . 6 September 2004.

[ST] A port of the UNIX "strings" utility to Windows. Extracted from MinGW distribution file binutils-2.11.92-20011113.tar.gz. URL: http://www.mingw.org . 6 September 2004.

[NC1] @Stake, Inc. NetCat 1.1 for Win9x/Me/NT/2000/XP. URL: http://www.atstake.com/research/tools/network_utilities . 6 September 2004.

[LP] Yoda. Lord PE version RolyalTS. URL: http://mitglied.lycos.de/yoda2k/LordPE/info.htm . 6 September 2004.

[SI1] Russinovich, Mark. Cogswell, Bryce. Filemon v6.06 for Windows NT/9x. URL: http://www.sysinternals.com/ntw2k/source/filemon.shtml . 6 September 2004.

[SI2] Russinovich, Mark. Cogswell, Bryce. Regmon v6.06 for Windows NT/9x. URL: http://www.sysinternals.com/ntw2k/source/regmon.shtml . 6 September 2004.

[SI3] Russinovich, Mark. TDImon v1.01 for Windows NT/9x. URL: http://www.sysinternals.com/ntw2k/source/regmon.shtml . 6 September 2004.

[RS] TiANWEi. RegShot 1.61e5 Final for Windows operating systems.

[BT] Foundstone, Inc. BinText version 3.00. URL: http://www.foundstone.com/knowledge/forensics.html . 6 September 2004.

[OD1] Clarke, Alex. OllyDbg version 1.09 for Windows operating systems. URL: http://home.t-online.de/home/Ollydbg . 6 September 2004.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

[OD2] Clarke, Alex. OllyDump v2.20.108. URL: http://home.t-online.de/home/Ollydbg . 6 September 2004.

[ID] Evaluation copy of IDA Pro v4.6.

[SI4] Russinovich, Mark. PsKill v1.03 for Windows NT/2000/XP. URL: http://www.sysinternals.com/ntw2k/freeware/pskill.shtml . 6 September 2004.

[NC2] @Stake, Inc. NetCat 1.10 for Unix platforms. URL: http://www.atstake.com/research/tools/network_utilities . 6 September 2004.

[IR] RedHat, Inc. Linux 9.0 operating system utilities. Internet Relay Chat. URL: http://www.redhat.com . 6 September 2004.

[SN] Caswell, Brian. Roesch, Marty. Snort.org. Snort. URL: http://www.snort.org . 6 September 2004.

[NA1] Network Associates, Inc. McAfee AntiVirus System Protection. URL: http://www.mcafeesecurity.com/us/products/mcafee/antivirus/category.htm . 15 September 2004.

[NA2] Network Associates, Inc. BackDoor-CGM Trojan. URL: http://vil.mcafeesecurity.com/vil/content/v_126653.htm . 6 September 2004.

[AP1] Aspack Software. Aspack for Windows 95/98/NT4/2000/XP. URL: www.aspack.com . 6 September 2004.

[AD] Yoda. AspackDie 1.3d for PE files compressed by Aspack 2.11/2.11c/2.11d/2.12. URL: http://www.exetools.com/unpackers.htm . 6 September 2004.

[AO1] TFreak. Attrition.org. Smurf TCP/IP flooding utility. URL: http://www.attrition.org/security/denial/w/jolt.dos.html . 6 September 2004.

[AO2] Attrition.org. Jolt 1.0 Denial of Service utility. URL: http://www.attrition.org/security/denial/w/jolt.dos.html . 6 September 2004.

[MS1] Microsoft.com. MSDN Library. PeekMessageA of text services framework. URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tsf/tsf/itfmessagepump_peekmessagea.asp . 12 September 2004.

[MS2] Microsoft.com. MSDN Library. GetMessageA of text services framework. URL:

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tsf/tsf/itfmessagepump_getmessagea.asp . 12 September 2004.

[PM] McKevitt, Paul. Windows Api Programming in C#, VB and VB6. DispatchMessage. URL: http://custom.programming-in.net/articles/art9-1.asp?f=DispatchMessage . 12 September 2004.

[MS3] Microsoft.com. MSDN Library. Visual C and C++ Libraries. Time function. URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vccore98/html/_crt_time.asp . 12 September 2004.

[LI] Uniblue Systems, Ltd. LIUtilities. WinTasks DLL Library. ws2help - ws2help.dll - DLL Information. URL: http://www.liutilities.com/products/wintaskspro/dlllibrary/ws2help/ . 12 September 2004.

[SO1] Sophos, Plc. Virus information Articles. Police crack suspected online extortion ring, Sophos reports. 23 July 2004. URL: http://www.sophos.com/virusinfo/articles/extortion.html . 14 September 2004.

[LZ] Zeltser, Lenny. SANS Malware FAQ: Reverse Engineering Srvcp.exe. URL: http://www.sans.org/resources/malwarefaq/srvcp.php . 15 September 2004.

[WN] Shachtman, Noah. Wired News. Hackers Take Aim at GOP. 17 August 2004. URL: http://www.wired.com/news/politics/0,1283,64602,00.html?tw=wn_story_top5 . 14 September 2004.

[SW] mobman. Author of SubSeven Trojan. SwatIt Anti Trojan and Bot Scanner and Remover. Bots. URL: http://swatit.org/bots/mobman.html . 14 September 2004.

© SANS Institute 2004, Author retains full rights.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.