Analysis of MyDoom.A

Zackery Adames
Austin Peay State University
601 College St
Clarksville TN, 37043, United States
(931) 624-0264
zadames@my.apsu.edu

Curtis Compton
Austin Peay State University
601 College St
Clarksville TN, 37043, United States
(931) 218-0924
ccompton6@my.apsu.edu

Douglas Ferguson
Austin Peay State University
601 College St
Clarksville TN, 37043, United States
(931) 436-8920
dferguson8@my.apsu.edu

Ethan McCrary
Austin Peay State University
601 College St
Clarksville TN, 37043, United States
(931) 272-1733
emccrary@my.apsu.edu

Curtis Roush
Austin Peay State University
601 College St
Clarksville TN, 37043, United States
(931) 302-9452
croush1@my.apsu.edu

ABSTRACT
Discussed in this paper are several types of analysis done with MyDoom.A. The analysis consisted of three parts: static, dynamic, and code analysis. For static analysis, a live copy of the MyDoom.A virus was taken. From there, hash values were checked and the executable was analyzed using hex editors. In dynamic analysis, MyDoom.A was ran in a virtual environment. From this, accessed files and generated processes were observed. For code analysis, the source code of MyDoom.A was analyzed and several key functions used to initiate myDoom.A's attacks were recorded. This project is a demonstration of the skills and techniques that were acquired in studying malware analysis.

General Terms
Malware Analysis, MyDoom.A, Distributed Denial of Service

Keywords
malware, analysis, static, dynamic, code, MyDoom.A, www.sco.com

1. INTRODUCTION
Malware, or malicious software, is [1] “software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system.” MyDoom.A, also known as Novarg, was created to send Distributed Denial of Service (DDoS) attacks to www.sco.com. This is caused by a worm that spreads via the Internet disguised as files attached to harmless looking messages. The worm only activates when a user opens the infected attachments. Once opened, the worm installs itself onto the system where its replication begins.

2. HISTORY
Thousands of malwares currently pollute the web, smart devices, and computers. In the past, MyDoom.A was one of the many malicious software that caused havoc around the Internet, specifically www.sco.com. First emerging around January 26th 2004, MyDoom.A was considered [2]”one of the fastest spreading and most destructive computer viruses of all time”. At its peak, [3] “one in every 12 email messages” contained a virus that would install a backdoor onto a person’s computer. While its origin is unknown, some earlier messages discovered seem to originate from Russia. Classified as a worm, that distributes a virus, the purpose of MyDoom.A was to hit the SCO website with a Distributed Denial of Service (DDoS) attack between February 1st and the 12th.

There are two main versions of the malware, MyDoom.A and MyDoom.b. Version A was designed to hit www.sco.com with a DDoS attack of 64 threads. Multiply that by several millions of infected computers, and you have a crashed website; specifically the mail server. While the World Wide Web portion is down, users can still navigate to just sco.com. The first few days of February is the infection process where the worm spreads from email to email, distributing a virus to any computer unlucky enough to be its host. As the number of infected victim’s increases, the number of threads hitting www.sco.com goes up as well.

Two to five years after its initial release in 2004, MyDoom.A is still not dead. Specialist in the security field are still discovering variants of the malware. Although the amount of attacks are not great in number as they were in 2004, they still could pose a threat to smaller companies or websites. Version B on the other hand, was created to crash Microsoft's website and mail server. While it managed to attack www.microsoft.com, it did not do as much damage like version A. The main cause of this is because there were not as many copies of version B distributed.

3. STATIC ANALYSIS
3.1 VirusTotal
3.1.1 Hash
Static analysis is the process of analyzing a file without interacting with it. For instance, searching the file for header information does not include running the file and allowing it to

make changes to the system. When Submitting the SHA-1 hash for MyDoom.A, only 4 engines in VirusTotal detected the file as malicious. Acrabit related it to a Trojan named Waledac, Fortinet related it back to the source – MyDoom.A, Endgame detected it as malicious with only moderate confidence, and NANO-Antivirus detected it as the trojan MyDoom.A.

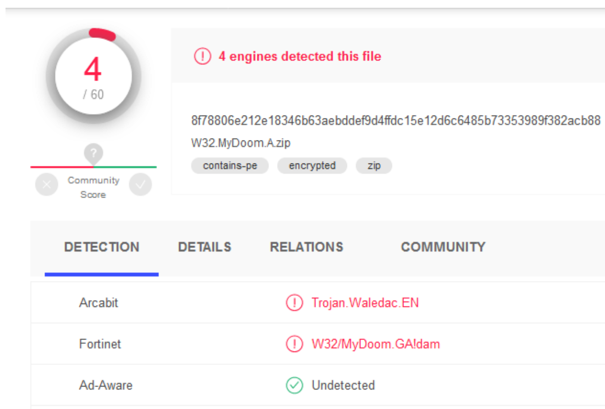

Figure 3.1 SHA-1 Hash Analysis (VirusTotal)

3.1.2 Details

Considering the last analysis of the hash was done on June 9th of 2019, there is a high possibility that the malware could have evolved or changed its hashed by this point. 61% of the files within the detected ZIP folder are of the filetype GIF. Around 19.5% of the files are unknown formats and another 19.5% are directory files, JPGs, and portable executables.

Contents Metadata

Contained Files	41
Uncompressed Size	707.26 KB
Earliest Content Modification	2004-05-31 23:00:48
Latest Content Modification	2019-01-09 22:40:54

Contained Files By Type

GIF	25
UNKNOWN	8
JPG	3
DIRECTORY	3
PORTABLE EXECUTABLE	2

Contained Files By Extension

GIF	25
HTM	3
JPG	3
JS	3
CSS	2
EXE	2

Figure 3.2. Metadata, File Type, and Extensions (VirusTotal)

VirusTotal was also able to analyze the details of the detected ZIP file and obtained some interesting information. For instance, the ZIP File Name is "Netcraft www_sco_com is a weapon of mass destruction.htm," which might be a hint that the file is dangerous. It was able to identify the name of the submitted hash as W32.MyDoom.A.zip, and even kept track of the latest modifications to the contents of the malware – January 9th, 2019. With that said, it is still a very active malware.

History

First Submission	2019-02-22 05:44:45
Last Submission	2019-06-09 20:14:30
Last Analysis	2019-06-09 20:14:30
Earliest Contents Modification	2004-05-31 23:00:48
Latest Contents Modification	2019-01-09 22:40:54

Figure 1 – History (VirusTotal)

3.1.3 Relations

While only four engines in VirusTotal were able to identify the hash as an identifier of a malicious software, MyDoom.A has related it to two other files that have had various malicious identities. One example is the file "strip-girl-2.0bdcom_patches.exe" – a Win32 executable. This executable alone has had 64/70 detections as a malicious trojan and worm. This is the real MyDoom.A executable.

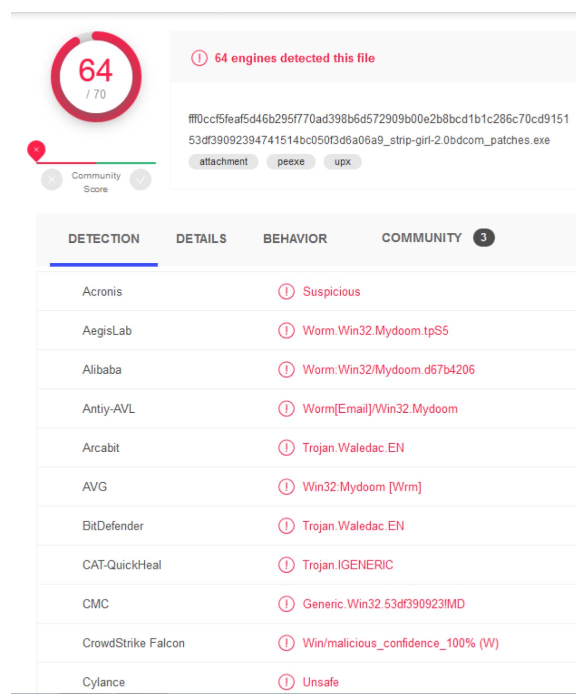

Figure 3.3. strip-girl-2.0bdcom_patches.exe (VirusTotal

3.2 CFF Explorer

3.2.1 File Analysis

The analysis of the strip-girl-2.0bdcom_patches.exe provides us with plenty of information, such as the date when it was created, the file size, the portable executable size, and the hashes (MD5 and SHA-1).

Property	Value
File Name	C:\Users\ethan\Downloads\theZoo-master\theZoo-master\malwares...
File Type	Portable Executable 32
File Info	No match found.
File Size	22.00 KB (22528 bytes)
PE Size	32.00 KB (32768 bytes)
Created	Wednesday 09 January 2019, 22.40.54
Modified	Saturday 23 November 2019, 15.14.06
Accessed	Wednesday 09 January 2019, 22.40.54
MD5	39A7D2BB5652C9D105C0D64A640C5A9D
SHA-1	E9FAB211F8DCAE2F118833042AAD6AE65EF6674D

Property	Value
Empty	No additional info available

Figure 3.4. File Analysis (CFF Explorer)

3.2.2 Imports

The following DLLs are all imports that were identified by CFF Explorer VII as imports stemming from strip-girl-2.0bdcom_patches.exe:

KERNEL32.DLL gives applications much of the 32-bit Windows base APIs, like memory management, input/output (I/O) operations, process and thread creation, and other functions.

ADVAPI32.DLL provides security calls and functions for manipulating the Windows Registry.

MSVCRT.DLL is the C standard library for the Visual C++ compiler from version 4.2 to 6.0. It provides programs compiled by these versions with most of the standard C library functions, such as string manipulation, memory allocation, C-style input/output calls, and others.

USER32.DLL is the Windows USER component that creates and manipulates elements of the Windows user interface, such as the desktop, windows, and menus.

WS2_32.DLL implements the Winsock API, which provides TCP/IP networking functions, which will be notable in the code analysis portion of this paper.

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.DLL	58	00000000	00000000	00000000	000085D4	0000101C
ADVAPI32.dll	6	00000000	00000000	00000000	000085E1	00001000
MSVCRT.dll	8	00000000	00000000	00000000	000085EE	00001108
USER32.dll	5	00000000	00000000	00000000	000085F9	0000112C
WS2_32.dll	15	00000000	00000000	00000000	00008604	00001144

Figure 3.5. Imports (CFF Explorer)

3.2.3 Strings

When analyzing the strings of strip-girl-2.0bdcom_patches.exe, we used the program CFF Explorer. Most of the strings that can be found throughout the executable are within the "Address Converter" section. Here, we can find several arrays of names and seemingly meaningless words and random combinations of text. The names are used within the code to randomize email headers and addresses. For example, one randomly generated email might look like jerry@aol.com. Certain keywords within the strings are filtered from the email addresses generated, such as "admin," "google," and "webmaster."

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
000019A0	70	65	74	65	72	00	00	00	74	6F	6D	00	72	61	79	00	peter...tom.ray.
000019B0	6D	61	72	79	00	00	00	00	73	65	72	67	00	00	00	00	mary....serg....
000019C0	62	72	69	61	6E	00	00	00	6A	69	6D	00	6D	61	72	69	brian...jim.mari
000019D0	61	00	00	00	6C	65	6F	00	6A	6F	73	65	00	00	00	00	a...leo.jose....
000019E0	61	6E	64	72	65	77	00	00	73	61	6D	00	67	65	6F	72	andrew..sam.geor
000019F0	67	65	00	00	64	61	76	69	64	00	00	00	6B	65	76	69	ge..david...kevi
00001A00	6E	00	00	00	6D	69	6B	65	00	00	00	00	6A	61	6D	65	n...mike....jame
00001A10	73	00	00	00	6D	69	63	68	61	65	6C	00	61	6C	65	78	s...michael.alex
00001A20	00	00	00	00	6A	6F	68	6E	00	00	00	00	61	63	63	6F	....john....acco
00001A30	75	6E	00	00	63	65	72	74	69	66	69	63	00	00	00	00	un..certific....
00001A40	6C	69	73	74	73	65	72	76	00	00	00	00	6E	74	69	76	listserv....ntiv
00001A50	69	00	00	00	73	75	70	70	6F	72	74	00	69	63	72	6F	i...support.icro
00001A60	73	6F	66	74	00	00	00	00	61	64	6D	69	6E	00	00	00	soft....admin...
00001A70	70	61	67	65	00	00	00	00	74	68	65	2E	62	61	74	00	page....the.bat.
00001A80	67	6F	6C	64	2D	63	65	72	74	73	00	00	63	61	00	00	gold-certs..ca..
00001A90	66	65	73	74	65	00	00	00	73	75	62	6D	69	74	00	00	feste...submit..
00001AA0	6E	6F	74	00	68	65	6C	70	00	00	00	00	73	65	72	76	not.help....serv
00001AB0	69	63	65	00	70	72	69	76	61	63	79	00	73	6F	6D	65	ice.privacy.some
00001AC0	62	6F	64	79	00	00	00	00	6E	6F	00	00	73	6F	66	74	body....no..soft
00001AD0	00	00	00	00	63	6F	6E	74	61	63	74	00	73	69	74	65	....contact.site
00001AE0	00	00	00	00	72	61	74	69	6E	67	00	00	62	75	67	73	....rating..bugs
00001AF0	00	00	00	00	6D	65	00	00	79	6F	75	00	79	6F	75	72	....me..you.your
00001B00	00	00	00	00	73	6F	6D	65	6F	6E	65	00	61	6E	79	6F	....someone.anyo
00001B10	6E	65	00	00	6E	6F	74	68	69	6E	67	00	6E	6F	62	6F	ne..nothing.nobo
00001B20	64	79	00	00	6E	6F	6F	6E	65	00	00	00	77	65	62	6D	dy..noone...webm
00001B30	61	73	74	65	72	00	00	00	70	6F	73	74	6D	61	73	74	aster...postmast
00001B40	65	72	00	00	73	61	6D	70	6C	65	73	00	69	6E	66	6F	er..samples.info
00001B50	00	00	00	00	72	6F	6F	74	00	00	00	00	0A	0A	62	65	....root......be
00001B60	5F	6C	6F	79	61	6C	3A	00	6D	6F	7A	69	6C	6C	61	00	_loyal:.mozilla.
00001B70	75	74	67	65	72	73	2E	65	64	00	00	00	74	61	6E	66	utgers.ed...tanf
00001B80	6F	72	64	2E	65	00	00	00	70	67	70	00	61	63	6B	65	ord.e...pgp.acke
00001B90	74	73	74	00	73	65	63	75	72	00	00	00	69	73	63	2E	tst.secur...isc.
00001BA0	6F	00	00	00	69	73	69	2E	65	00	00	00	72	69	70	65	o...isi.e...ripe
00001BB0	2E	00	00	00	61	72	69	6E	2E	00	00	00	73	65	6E	64	....arin....send
00001BC0	6D	61	69	6C	00	00	00	00	72	66	63	2D	65	64	00	00	mail....rfc-ed..
00001BD0	69	65	74	66	00	00	00	00	69	61	6E	61	00	00	00	00	ietf....iana....
00001BE0	75	73	65	6E	65	74	00	00	66	69	64	6F	00	00	00	00	usenet..fido....
00001BF0	6C	69	6E	75	78	00	00	00	6B	65	72	6E	65	6C	00	00	linux...kernel..
00001C00	67	6F	6F	67	6C	65	00	00	69	62	6D	2E	63	6F	6D	00	google..ibm.com.
00001C10	66	73	66	2E	00	00	00	00	67	6E	75	00	6D	69	74	2E	fsf.....gnu.mit.
00001C20	65	00	00	00	62	73	64	00	6D	61	74	68	00	00	00	00	e...bsd.math....
00001C30	75	6E	69	78	00	00	00	00	62	65	72	6B	65	6C	65	79	unix....berkeley
00001C40	00	00	00	00	0A	0A	0A	00	66	6F	6F	2E	00	00	00	00	........foo.....
00001C50	2E	6D	69	6C	00	00	00	00	67	6F	76	2E	00	00	00	00	.mil....gov.....
00001C60	2E	67	6F	76	00	00	00	00	72	75	73	6C	69	73	00	00	.gov....ruslis..
00001C70	6E	6F	64	6F	6D	61	69	00	6D	79	64	6F	6D	61	69	00	nodomai.mydomai.
00001C80	65	78	61	6D	70	6C	65	00	69	6E	70	72	69	73	00	00	example.inpris..
00001C90	62	6F	72	6C	61	6E	00	00	73	6F	70	68	6F	00	00	00	borlan..sopho...
00001CA0	70	61	6E	64	61	00	00	00	68	6F	74	6D	61	69	6C	00	panda...hotmail.
00001CB0	6D	73	6E	2E	00	00	00	00	69	63	72	6F	73	6F	66	00	msn.....icrosof.

Figure 3.6. ASCII Strings (CFF Explorer)

3.3 PEStudio

3.3.1 Indicators

When analyzing the file in PEStudio, we can find several indicators that the executable strip-girl-2.0bdcom_patches.exe is malicious. PEStudio gathered information from the SHA-256 hash that pointed towards blacklisted information, such as reference strings, symbols, and libraries. Other indicators were entry point locations, writability, and the executable sections. The sections above provide us with the most prominent indicators that a file is malicious.

xml-id	indicator (27)
1225	The location of the entry-point is suspicious
1223	The first section is writable
1430	The file references string(s) tagged as blacklist
1269	The file references blacklist library(ies)
1120	The file is scored by virustotal
1266	The file imports symbol(s) tagged as blacklist
2215	The file contains writable and executable section(s)
1631	The file contains self-modifying executable section(s)
1245	The file contains a blacklist section
1259	The dos-stub message is unusual
1265	The count of imported functions is suspicious
1321	The time-stamp of the compiler is suspicious

Figure 3.7. Indicators (PEStudio)

4. DYNAMIC ANALYSIS

It is important to remember that MyDoom.A is an old virus that was first discovered back in 2004, and because it has since be deactivated, the system clock must be set back to sometime in 2004. An attempt was made to dynamically analyze the program in the modern time (2019), to no avail. No evidence of the malware's digital footprint was found. Once the system clock was set back to 2004, however, we began to get very different results.

The dynamic analysis tools used below began to pick up traffic soon after the executable was run. The date used in the dynamic analysis below is the 27th of January, though any date in early 2004 would probably have produced similar results.

4.1 Process Monitor

Below is a screen capture of what process monitor picked up soon after the executable was run:

9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Internet Explorer	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer*	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer	SUCCESS	0: .., 1: en-US, 2: h...
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Internet Explorer\en-US	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\en-U...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Internet Explorer\en-US	SUCCESS	Offset: 0, Length: 4...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\en-US	SUCCESS	0: .., 1: hmmapi.dll....
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\en-US	NO MORE FILES	
9:48:0...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Internet Explorer\en-US	SUCCESS	
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Internet Explorer\SIGN...	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\SIGN...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\SIGN...	SUCCESS	0: .., 1: install.ins
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer\SIGN...	NO MORE FILES	
9:48:0...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Internet Explorer\SIGN...	SUCCESS	
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Internet Explorer	NO MORE FILES	
9:48:0...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Internet Explorer	SUCCESS	
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Microsoft Games	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games*	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games	SUCCESS	0: .., 1: FreeCell, 2:...
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	0: .., 1: desktop.ini,...
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	0: .., 1: FreeCell.ex...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	NO MORE FILES	
9:48:0...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Free...	NO MORE FILES	
9:48:0...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Microsoft Games\Hearts	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Hear...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Hearts	SUCCESS	0: .., 1: desktop.ini,...
9:48:0...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Microsoft Games\Hear...	SUCCESS	Desired Access: R...
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Hear...	SUCCESS	Filter: *, 1: .
9:48:0...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Microsoft Games\Hear...	SUCCESS	0: .., 1: Hearts.exe....

Figure 4.1. Queries made by MyDoom.A

The malware starts working right away, and the first thing it appears to do is to create a file that attempts to gain access to the Read Data/List Directory. The attempt is successful and the Share Mode is set to Read, Write, and Delete.

Date:	1/27/2004 9:48:09.3287954 AM
Thread:	2464
Class:	File System
Operation:	CreateFile
Result:	SUCCESS
Path:	C:\Program Files\Internet Explorer
Duration:	0.0000141
Desired Access:	Read Data/List Directory, Synchronize
Disposition:	Open
Options:	Directory, Synchronous IO Non-Alert
Attributes:	n/a
ShareMode:	Read, Write, Delete
AllocationSize:	n/a
OpenResult:	Opened

Figure 4.2. MyDoom.A successfully creates a file.

The executable then proceeds to call a number of dll's. Some of the notable dll's called include crypt32, kernel32, cryptbase, KernelBase, user32, and about 30 to 40 other dlls.

PID:	2892	Architecture:	32-bit
Parent PID:	2252	Virtualized:	False
Session ID:	1	Integrity:	High
User:	WIN-1RIE02FANBN\Binary		
Auth ID:	00000000:00014b5f		
Started:	1/27/2004 9:47:52 AM	Ended:	(Running)
Modules:			

Module	Address	Size	Path	Company	Version	Timestamp
sspicli.dll	0x753b0000	0x60000	C:\Windows\SysWOW64\sspicli.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 6:0...
msvcrt.dll	0x75510000	0xac000	C:\Windows\SysWOW64\msvcrt.dll	Microsoft Corpo...	7.0.7600.1638...	7/13/2009 7:07...
devobj.dll	0x755c0000	0x12000	C:\Windows\SysWOW64\devobj.dll	Microsoft Corpo...	6.1.7600.1638...	7/13/2009 7:05...
wininet.dll	0x755e0000	0xf5000	C:\Windows\SysWOW64\wininet.dll	Microsoft Corpo...	8.00.7600.163...	11/20/2010 6:0...
msctf.dll	0x75710000	0xcc000	C:\Windows\SysWOW64\msctf.dll	Microsoft Corpo...	6.1.7600.1638...	7/13/2009 7:07...
ws2_32.dll	0x757e0000	0x35000	C:\Windows\SysWOW64\ws2_32.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
iertutil.dll	0x75820000	0x1fb000	C:\Windows\SysWOW64\iertutil.dll	Microsoft Corpo...	8.00.7601.175...	11/20/2010 6:0...
ole32.dll	0x75b10000	0x15c000	C:\Windows\SysWOW64\ole32.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
cfgmgr32.dll	0x75c70000	0x27000	C:\Windows\SysWOW64\cfgmgr32.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 5:5...
KernelBase.dll	0x75ca0000	0x46000	C:\Windows\SysWOW64\KernelBase...	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:1...
oleaut32.dll	0x75cf0000	0x8f000	C:\Windows\SysWOW64\oleaut32.dll	Microsoft Corpo...	6.1.7601.17514	11/20/2010 6:0...
imm32.dll	0x75d80000	0x60000	C:\Windows\SysWOW64\imm32.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 6:0...
urlmon.dll	0x75de0000	0x136000	C:\Windows\SysWOW64\urlmon.dll	Microsoft Corpo...	8.00.7600.163...	11/20/2010 6:0...
nsi.dll	0x75fb0000	0x6000	C:\Windows\SysWOW64\nsi.dll	Microsoft Corpo...	6.1.7600.1638...	7/13/2009 7:09...
lpk.dll	0x75fc0000	0xa000	C:\Windows\SysWOW64\lpk.dll	Microsoft Corpo...	6.1.7600.1638...	7/13/2009 7:11...
sechost.dll	0x75fd0000	0x19000	C:\Windows\SysWOW64\sechost.dll	Microsoft Corpo...	6.1.7600.1638...	7/13/2009 7:10...
shlwapi.dll	0x76100000	0x57000	C:\Windows\SysWOW64\shlwapi.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
crypt32.dll	0x76160000	0x11d000	C:\Windows\SysWOW64\crypt32.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
setupapi.dll	0x76280000	0x19d000	C:\Windows\SysWOW64\setupapi.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
gdi32.dll	0x76420000	0x90000	C:\Windows\SysWOW64\gdi32.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 6:0...
usp10.dll	0x76610000	0x9d000	C:\Windows\SysWOW64\usp10.dll	Microsoft Corpo...	1.0626.7601.1...	11/20/2010 6:0...
shell32.dll	0x76760000	0xc4a000	C:\Windows\SysWOW64\shell32.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 6:0...
advapi32.dll	0x773b0000	0xa0000	C:\Windows\SysWOW64\advapi32.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 5:5...
user32.dll	0x77450000	0xfa000	C:\Windows\System32\user32.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 7:1...
kernel32.dll	0x77550000	0x11f000	C:\Windows\System32\kernel32.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 7:0...
ntdll.dll	0x77670000	0x1a9000	C:\Windows\System32\ntdll.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 7:1...
msasn1.dll	0x77820000	0xc000	C:\Windows\SysWOW64\msasn1.dll	Microsoft Corpo...	6.1.7601.1751...	11/20/2010 6:0...
ntdll.dll	0x77850000	0x180000	C:\Windows\SysWOW64\ntdll.dll	Microsoft Corpo...	6.1.7600.1638...	11/20/2010 6:0...
shimgapi.dll	0x7e1a0000	0x7000	C:\Windows\SysWOW64\shimgapi.dll			12/31/1969 6:0...

Figure 4.3. MyDoom.A calling .dll files.

The malware also begins to access Microsoft games creating files and accessing query directories.

CreateFile	C:\Program Files\Microsoft Games	SUCCESS	Desired Access: R...
QueryDirectory	C:\Program Files\Microsoft Games*	SUCCESS	Filter: *, 1: .
QueryDirectory	C:\Program Files\Microsoft Games	SUCCESS	0: .., 1: FreeCell, 2:...
CreateFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	Desired Access: R...
QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	Filter: *, 1: .
QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	0: .., 1: desktop.ini,...
CreateFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	Desired Access: R...
QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	Filter: *, 1: .
QueryDirectory	C:\Program Files\Microsoft Games\Free...	SUCCESS	0: .., 1: FreeCell.ex...
QueryDirectory	C:\Program Files\Microsoft Games\Free...	NO MORE FILES	
CloseFile	C:\Program Files\Microsoft Games\Free...	SUCCESS	

Figure 4.4. MyDoom.A accessing Microsoft games

Interestingly enough, the MyDoom.A executable also begins to access process hacker, another malware analysis tool that had been downloaded onto the virtual machine used for the analysis. While specific reasons for this are uncertain, we deduce that this is because it recognizes Process Hacker 2 as a malware analysis/detection tool and is trying to ensure it does not interrupt the infection process.

9:48:1...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Process Hacker 2	SUCCESS	Desired Access: R...
9:48:1...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Process Hacker 2*	SUCCESS	Filter: *, 1: .
9:48:1...	strip-girl-2.0bdc...	2892	QueryDirectory	C:\Program Files\Process Hacker 2	SUCCESS	0: .., 1: CHANGEL...
9:48:1...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Process Hacker 2\CH...	SUCCESS	Desired Access: G...
9:48:1...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Process Hacker 2\CH...	SUCCESS	Offset: 0, Length: 2...
9:48:1...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Process Hacker 2\CH...	SUCCESS	Offset: 0, Length: 2...
9:48:1...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Process Hacker 2\CH...	END OF FILE	Offset: 25,995, Len...
9:48:1...	strip-girl-2.0bdc...	2892	CloseFile	C:\Program Files\Process Hacker 2\CH...	SUCCESS	
9:48:1...	strip-girl-2.0bdc...	2892	CreateFile	C:\Program Files\Process Hacker 2\CO...	SUCCESS	Desired Access: G...
9:48:1...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Process Hacker 2\CO...	SUCCESS	Offset: 0, Length: 6...
9:48:1...	strip-girl-2.0bdc...	2892	ReadFile	C:\Program Files\Process Hacker 2\CO...	SUCCESS	Offset: 0, Length: 6...

Figure 4.5. MyDoom.A Accessing Process Hacker

One of MyDoom.A's primary purposes, performing a DOS attack, is ultimately internet based and this is reflected in our Process Monitor findings. The malware immediately starts trying to access the internet. After access has been gained the executable begins a TCP/UDP send receive sequence. This appears to be the handshake performed with the internet protocols before it begins the TCP flood.

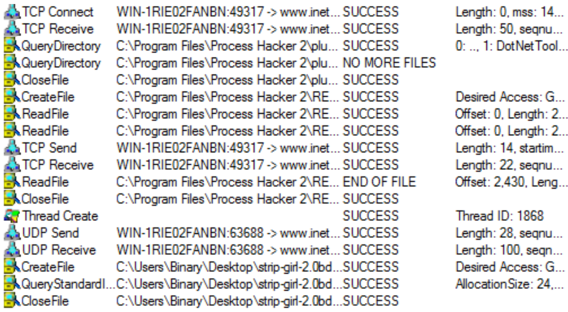

Figure 4.6. TCP Packet generation

After the handshake, MyDoom.A begins the TCP flood. The virtual machine used to analyze this malware is utilizing the INetSim tool, which creates a fake internet-like environment for the malware to interact with. The malware falls for the bait and begins attacking INetSim with its DOS.

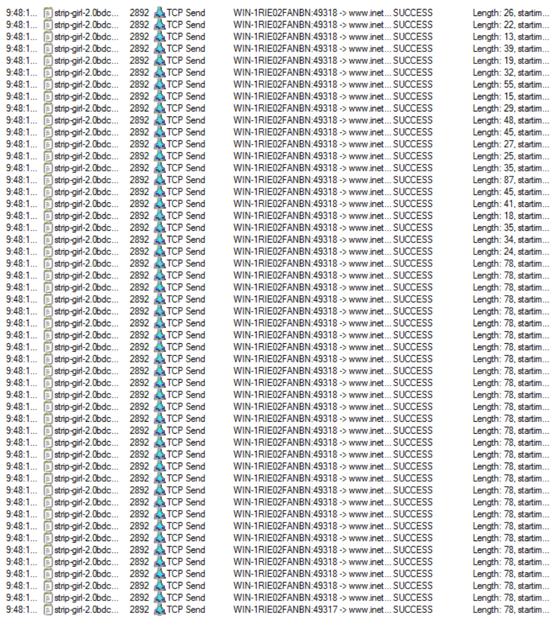
Figure 4.7 DDOS Attack

4.2 Process Explorer

Process explorer provides the malware analyst with a variety of options when analyzing a live malware sample. In this first image we are able to see the malware's performance as it attempts to attack INetSim. The performance spikes at regular intervals as MyDoom.A attempts a DOS attack.

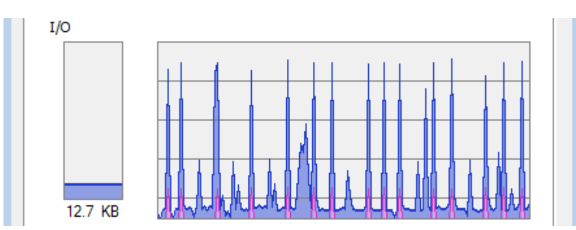

Figure 4.8. Resource usage

When process explorer executes, it allows the analyst to choose an executable to examine. Windows defender will shut down the MyDoom.A executable, so it was imperative to disable the windows feature before analyzing the executable. The threads used by the malware are also visible through process explorer.

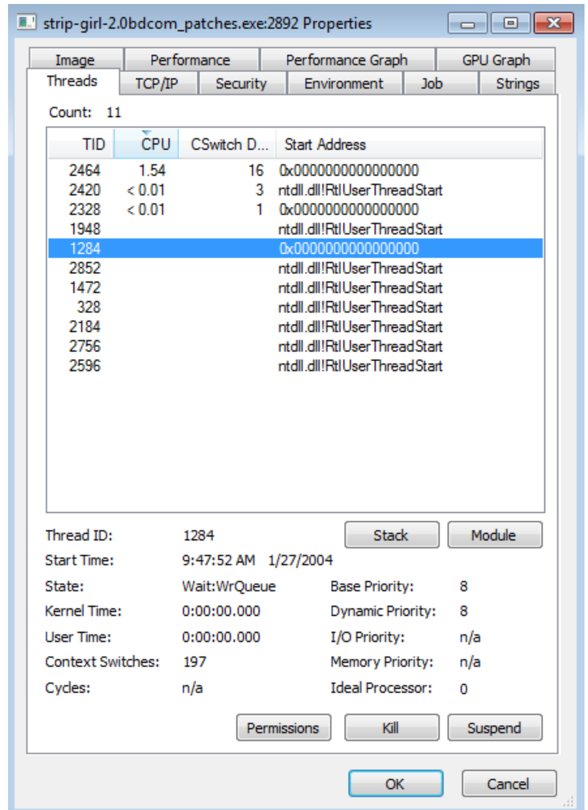

Figure 4.9. Threads created by MyDoom.A

More detailed performance details are available for analysis through the 'performance' tab. Information such as Kernel time, User Time, Virtual Memory, Physical Memory, I/O and Handles are all available through this tab.

5. CODE ANALYSIS

Code analysis of malware is a crucial step in malware analysis. Unfortunately, most malware authors do not publish their source code. With disassemblers like IDA and Ghidra, assembly code can be analyzed and reversed engineered into higher level languages, primarily C and C++. In the case of MyDoom.A, the original source code was distributed by MyDoom.C, also known as DoomJuice, with authorities believing [4] that "MyDoom's author is spreading the code to encourage others to write copy-cat viruses which try and mimic MyDoom. " The benefit to researchers, is that with the original source code, it becomes much easier to understand what occurs behind the scenes of this malware. As part of our code analysis we will be analyzing key function used by MyDoom.A in its attacks.

5.1 main.c

5.1.1 sync_main()

This function is the primary function of the MyDoom.A, and is responsible for initializing the worm. After verifying that MyDoom.A has been attached to the taskmon.exe process the

MyDoom.A begins its Distributed Denial of Service of www.sco.com.

5.1.2. sync_startup()

The sync_startup function is called by sync_main during the initialization of MyDoom.A. This function opens the registry and attaches the MyDoom.A worm to taskmon.exe, guaranteeing that the malware runs on startup.

```
if (RegOpenKeyEx(HKEY_LOCAL_MACHINE,
        regpath, 0, KEY_WRITE, &k) != 0)
    if (RegOpenKeyEx(HKEY_CURRENT_USER,
            regpath, 0, KEY_WRITE, &k) != 0)
        return;
RegSetValueEx(k, valname, 0, REG_SZ,
sync->sync_instpath,
lstrlen(sync->sync_instpath)+1);
RegCloseKey(k);
```

Figure 5.1. Attaching MyDoom.A to taskmon.exe

5.1.3. sync_check_frun()

This function begins by declares a char array that is 128 characters long, and passes it into a function named rot13, which utilizes a simple substitution cypher that replaces each letter in the string with the 13th letter after it. This prevents the strings from being easily read using debuggers. The resulting string, once decoded is:

“Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\ComDlg32\\Version”

After decoding the string explorer, MyDoom.A proceeds to open regedit and makes changes to the ComDlg32.ocx runtime library for Visual Basic. Follwing this procedures, the MyDoom.A virus creates a second Registry key for the ComDlg32.ocx.

5.2 scan.c

5.2.1 scan_dir_file(), scan_textfile(), scantext_textcvt()

This scan_dir_file function is used by MyDoom.A to scan the infected computer’s hard drives. Files with the extensions htmb, shtl, phpq, aspd, dbxn, tbbg, and adbh are located and passed into the scan_text file function. The scan_text function reads the contents of the file, and passes the text into the scantext_textcvt function. This function scans the text for email adresses and stores them in a list. MyDoom.A is then emailed to each of the addresses.

```
for (;;) {
    dwRead = 0;
    ReadFile(hFile, buf, sizeof(buf)-2, &dwRead, NULL);
    if (dwRead == 0 || dwRead >= sizeof(buf)) break;
    dwTotalRead += dwRead;
    buf[dwRead] = 0;
    scantext_textcvt(buf, dwRead);
    dwTotalFound += scantext_extract_ats(buf, dwRead);
    if ((dwTotalFound == 0) && (dwTotalRead > (300*1024)))
        break;
}
```

Figure 5.2. scan_textfile()

5.3 massmail.c

5.3.1. mm_gen(), massmail_addq(), email_filter()

Email distribution of MyDoom.A occurs when the mm_gen function is called. This function generates fake emails, based off the email addresses aquired by the scan functions. MyDoom.A is selective in which email addresses it forwards itself to, by calling the email_filter function. Each email address located by MyDoom.A is first checked against a list of keywords by the email_filter function to prevent sending MyDoom.A to targets that have email filters and IT departments.

```
static const char *nospam_domains[] = {
    "avp", "syma", "icrosof", "msn.", "hotmail",
    "panda", "sopho", "borlan", "inpris",
    "example", "mydomai", "nodomai", "ruslis",
    ".gov", "gov.", ".mil", "foo.", NULL, "\n\n\n"
};
static const char *loyal_list[] = {
    "berkeley", "unix", "math", "bsd", "mit.e",
    "gnu", "fsf.", "ibm.com", "google", "kernel",
    "linux", "fido", "usenet",  "iana", "ietf",
    "rfc-ed", "sendmail", "arin.", "ripe.",
    "isi.e", "isc.o", "secur", "acketst", "pgp",
    "tanford.e", "utgers.ed", "mozilla", NULL,
    "\n\nbe_loyal:"
};
```

Figure 5.3. Keywords filtered from email addresses

5.4 msg.c

5.4.1 select_exename()

The email sent by MyDoom.A attaches a file with a randomly generated name and file extension. MyDoom.A takes one of nine file names and five fake file extensions, and and passes them through a ROT13 cipher. After generating a filename, MyDoom.A renames itself and attaches itself to the email.

```
if ((xrand16() % 100) < 5) {
    j = 3 + (xrand16() % 15);
    for (i=0; i<j; i++)
        state->subject[i] = 'a' + (xrand16() % 26);
    state->subject[i] = 0;
} else {
    for (i=0, tot=1; subjs[i].pref != 0; i++) tot += subjs[i].pref;
    j = xrand16() % tot;
    for (i=0, tot=1; subjs[i].pref != 0; i++)
        if ((tot += subjs[i].pref) >= j) break;
    if (subjs[i].pref == 0) i = 0;
    rot13(state->subject, subjs[i].subj);
}
i = xrand16() % 100;
if ((i >= 50) && (i < 85))
    CharUpperBuff(state->subject, 1);
else if (i >= 85)
    CharUpper(state->subject);
```

Figure 5.4. Generating a fake filename for MyDoom.a

5.4.2 write_msgtext()

The email contents of MyDoom.A is generated by the write_msgtext function. This function takes an existing string, which is not encrypted using rot13. This text is used as the contents of the MyDoom.A emails. With each computer that is infected with MyDoom.A, a miniature email server is created to spread phishing emails.

```
struct {
    int pref;
    char *text;
} texts[] = {
    { 20, "" },
    { 5, "test" },
    { 40, "The message cannot be represented in 7-bit ASCII
            encoding and has been sent as a binary attachment." },
    { 40, "The message contains Unicode characters and has been
            sent as a binary attachment." },
    { 20, "Mail transaction failed. Partial message is available." },
    { 0, "" }
};
```

Figure 5.5. Email message text.

5.5 sco.c

5.5.1 scodos_th()

With this function we have finally reached where MyDoom.A begins its denial of service to www.sco.com. This function creates a low priority thread and creates an outward connection to www.sco.com.

```
SetThreadPriority(GetCurrentThread(),
THREAD_PRIORITY_BELOW_NORMAL);
if (pv == NULL) goto ex;
addr = *(struct sockaddr_in *)pv;
for (;;) {
    sock = connect_tv(&addr, 8);
    if (sock != 0) {
        send(sock, buf, lstrlen(buf), 0);
        Sleep(300);
        closesocket(sock);
    }
}
```

Figure 5.6. Beginning the DOS of www.sco.com

6. CONCLUSION

The MyDoom malware is a virus that is still in circulation today, with more variants appearing for different purposes. This virus is particularly resilient as it does not utilize exploits and relies on social engineering and phishing emails to infect computers. MyDoom's versatility has made it the most devastating computer virus to date. The virus' purpose is threefold, as it can be modified to create zombie email servers that can be used to spread spam email without needing to invest in infrastructure. This email functionality also can be used to further MyDoom's reach. The most obvious function was the distributed denial of service experienced by ww.sco.com and later www.microsoft.com. The most sinister aspect of MyDoom was the installation of backdoors on infected computers, which had the potential of letting hackers' access thousands of infected machines.

Analysis of MyDoom.A demonstrates the need for information security professionals to learn how to properly analyze malware when they come across it. With how dependent corporations and even government agencies are on technology, malware detection and removal cannot be relegated to antivirus engines. Malware analysis should become a priority in the training of security professionals to prevent attacks like MyDoom from compromising more organizations in the future.

7. REFERENCES

[1] "Malware: Definition of Malware by Lexico." *Lexico Dictionaries | English*, Lexico Dictionaries, www.lexico.com/en/definition/malware.

[2] Rochford, Louisa. "The Worst Computer Viruses in History." *CEO Today*, June 2019, www.ceotodaymagazine.com/2019/06/the-worst-computer-viruses-in-hist.

[3] Germain, Jack. "MyDoom: A Wrap-Up on the World's Most Vicious Worm." *TechNewsWorld.com*, 9 Mar. 2004, https://www.technewsworld.com/story/33068.html.

[4] Keizer, Gregg. "Why Is MyDoom Author Spreading Source Code?" *CRN*, 10 Feb. 2004, https://www.crn.com/news/security/18831436/why-is-mydoom-author-spreading-source-code.htm.

[5] ytisf. "ThZoo." *Git Hub*, Microsoft, 4 Sept. 2019, https://github.com/ytisf/theZoo.

[6] yorickdewid. "MyDoom." Git Hub, Microsoft, 16 Nov. 2015, https://github.com/yorickdewid/MyDoom.