Software Engineering Institute
Carnegie Mellon University

2000 CERT Incident Notes

CERT Division

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

http://www.sei.cmu.edu

REV-03.18.2016.0

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

This report was prepared for the

SEI Administrative Agent
AFLCMC/AZS
5 Eglin Street
Hanscom AFB, MA 01731-2100

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM17-0052

Table of Contents

1 IN-2000-01:Windows Based DDOS Agents 1

2 IN-2000-02: Exploitation of Unprotected Windows Networking Shares 3

3 IN-2000-03: 911 Worm 6

4 IN-2000-04: Denial of Service Attacks using Nameservers 8

5 IN-2000-05: "mstream" Distributed Denial of Service Tool 10

6 IN-2000-06: Exploitation of "Scriptlet.Typelib" ActiveX Control 15

7 IN-2000-07: Exploitation of Hidden File Extensions 18

8 IN-2000-08: Chat Clients and Network Security 21

9 IN-2000-09: Systems Compromised Through a Vulnerability in the IRIX telnet daemon 23

10 IN-2000-10: Widespread Exploitation of rpc.statd and wu-ftpd Vulnerabilities 25

1 IN-2000-01:Windows Based DDOS Agents

Updated: Tuesday, October 3, 2000
Date: Monday February 28, 2000

Description:

We have received reports indicating intruders are beginning to deploy and utilize windows based denial of service agents to launch distributed denial of service attacks. On Feburary 16th we began receiving reports of a program called "service.exe" that appears to be a Windows version of trinoo. This program listens on UDP port 34555. More details about this tool are available on Gary Flynn's web site at:
http://www.jmu.edu/computing/info-security/engineering/issues/wintrino.shtml

We have seen two almost identical versions of the "service.exe" program to date (they vary by 12 bytes but produce the same results for strings(1)). The binaries we have seen have one of the following MD5 checksums:

MD5 (service.exe) = 03fe58987d7dc07e736c13b8bee2e616

MD5 (service.exe) = 1d45f8425ef969eba40091e330921757

In at least one incident, machines runing the "service.exe" program were also running backoriface. We have also received reports of administrators finding other "remote administration" intruder tools on machines that were running "service.exe".

Note that the tool TFN2K, first released in December 1999, will run on Windows NT. The existance of distributed denial of service tools for Windows platforms is not new; however, we are beginning to receive reports of these tools being installed on compromised systems.

Impact:

Windows machines have been used as intermediaries in various types of denial of service attacks for years; however, the development and deployment of the technology to use Windows machines as agents in a distributed denial of service attacks represents an overall increase in the threat of denial of service attacks.

Solution:

Standard safe computing practices will prevent intruders from installing the service.exe program on your machine(s).

- Don't run programs of unknown origin, regardless of who sent you the program. Likewise, don't send programs of unknown origin to your friends or coworkers simply because they are amusing -- it might be a Trojan horse.
- Before opening any email attachments, be sure you know what the source of the attachment was. It is not enough that the mail originated from an address you recognize. The Melissa virus spread precisely because it originated from a familiar address. Malicious code might be distributed in amusing or enticing programs. If you must open an attachment before you can verify the source, do so in an isolated environment. If you are unsure how to proceed, contact your local technical support organization.
- Be sure your anti-virus software is, and remains, up-to-date.
- Some products, such as Microsoft Office, Lotus Notes and others, include the ability to execute code embedded in documents. For any such products you use, disable the automatic execution of code embedded in documents. For example, in Microsoft Word 97, enable the "Macro Virus Protection" feature by choosing Tools->Options->General and selecting the appropriate checkbox. In Lotus Notes 4.6, set a restrictive Execution Control List (ECL) by setting the options found in File->Tools->User Preferences->Security Options to restrict the execution of code to trusted signers. For other products, consult your documentation.
- Use data-integrity tools. Data-integrity tools use strong cryptography to help you determine which files, if any, may have changed on a system. This may be crucial information to determine the most appropriate response to a security event. The use of these tools requires that they be installed before a security event has taken place.
- Avoid the use of MIME types that cause interpreters or shells to be invoked.
- Be aware of the risks involved in the use of "mobile code" such as Active X, Java, and JavaScript. It is often the case that electronic mail programs use the same code that web browsers use to render HTML. Vulnerabilities that affect ActiveX, Java, and Javascript often are applicable to electronic mail as well as web pages.

Author: Jed Pickel

2 IN-2000-02: Exploitation of Unprotected Windows Networking Shares

Updated: Friday, April 7, 2000
Date: Friday, March 3, 2000

Overview

Intruders are actively exploiting Windows networking shares that are made available for remote connections across the Internet. This is not a new problem, but the potential impact on the overall security of the Internet is increasing.

Description

We have received reports indicating a rise in activity related to a malicious Visual Basic Script (VBScript) known as "network.vbs". The malicious script is similar to a harmless example script distributed with some versions of Windows 98, found as:

c:\windows\samples\wsh\network.vbs

The malicious network.vbs script attempts to do the following things:

- Open C:\network.log on the local machine
- Generate a random /24 network address block. The algorithm we have seen used to generate addresses is:
  - the first octet will be randomly selected between 199 and 214 the first 50 times, after which is it randomly selected between 1 and 254
  - the second and third octet are randomly selected between 1 and 254
  - the fourth octet begins at 1
- The generated /24 address is written to C:\network.log
- For each host address from 1 to 254 in the generated /24 range, network.vbs attempts to remotely mount a share named "C" from the remote computer as J: on the local computer.
  - If the "C" share of a remote computer is mounted successfully, copies network.vbs to the following locations on the remotely mounted filesystem:

    "j:\"

    "j:\windows\startm~1\programs\startup\"

    "j:\windows\"

    "j:\windows\start menu\programs\startup\"

    "j:\win95\start menu\programs\startup\"

    "j:\win95\startm~1\programs\startup\"

    "j:\wind95\"

If the first copy is successful, the address of the target system is written to C:\network.log.

- network.vbs then generates a new random /24 network address range and starts the process over. It will continue to cycle through random address space implanting copies of itself onto vulnerable computers until administrative intervention prevents further execution.

When configuring the C: drive of a Windows 9x machine to be shared, the default share name assigned is "C". If this default share name is used on a vulnerable computer, network.vbs performs it's file copies on the C: drive of the remote system. If network.vbs is successfully copied into a Windows startup folder on a remote system, the remote system could execute network.vbs when the system reboots or a new user logs into the system.

We have also seen variations of network.vbs that perform different actions, such as:

- Create deceptively titled malicious items in the Windows startup folder and in the user start menu
- Deploy distributed encryption cracking tools on vulnerable systems

The network.vbs script demonstrates one pervasive method of propagation intruders can leverage to deploy tools on Windows-based computer systems connected to the Internet. We are aware of one infected computer that attempted to infect a range of at least 2,400,000 other IP addresses before being detected and stopped. There may also be denial of service issues due to packet traffic if network.vbs is able to infect and execute from a large number of machines in a concentrated area.

Abe Singer from the San Diego Supercomputer Center has also published an analysis of network.vbs, available at: http://security.sdsc.edu/publications/network.vbs.shtml

Impact

Unprotected Windows networking shares can be exploited by intruders in an automated way to place tools on large numbers of Windows-based computers attached to the Internet. Because site security on the Internet is interdependent, a compromised system not only creates problems for the system's owner, but it is also threat to other sites on the Internet. The greater immediate risk to the Internet community is the potentially large number of systems attached to the Internet with unprotected Windows networking shares combined with distributed attack tools such as those described in IN-2000-01, Windows Based DDOS Agents

Another threat includes malicious and destructive code, such as viruses or worms, which leverage unprotected Windows networking shares to propagate. One such example is the 911 worm described in IN-2000-03, 911 worm

There is great potential for the emergence of other instances of intruder tools that leverage unprotected Windows networking shares on a widespread basis.

Solutions

Removing the network.vbs script from an infected computer involves removing the running image from memory and deleting the copies of network.vbs from the hard drive. Other tools installed using the same method of propagation may be more difficult to detect and remove.

You may wish to insure your anti-virus software is configured to test file names ending in .VBS to help detect virus outbreaks involving malicious VBScript code.

Several steps can be taken to prevent exploitation of the larger problem of unprotected Windows networking shares:

- Disable Windows networking shares in the Windows network control panel if the ability to share files is not needed. Or, you may choose to entirely disable NETBIOS over TCP/IP in the network control panel.
- When configuring a Windows share, require a password to connect to the share. The use of sound password practices is encouraged. It is also important to consider trust relationships between systems. Malicious code may be able to leverage situations where a vulnerable system is trusted by and already authenticated to a remote system.
- Restrict exported directories and files to the minimum required for an application. In other words, rather than exporting an entire disk, export only the directory or file needed. Export read-only where possible.
- If your security policy is such that Windows networking is not used between systems on your network and systems outside of your network, packet filtering can be used at network borders to prevent NETBIOS packets from entering and/or leaving a network. Alternatively, use packet filtering to allow NETBIOS packets only between those sites with whom you want to do file sharing. The following ports are commonly associated with Windows networking:

```
netbios-ns      137/tcp      # NETBIOS Name Service
netbios-ns      137/udp
netbios-dgm     138/tcp      # NETBIOS Datagram Service
netbios-dgm     138/udp
netbios-ssn     139/tcp      # NETBIOS session service
netbios-ssn     139/udp
```

  Keep in mind that packet filtering alone may not provide complete protection. Malicious code can enter a network through portable code downloaded from web sites or through email containing portable code or executable file attachments. For more information about Trojan horses and suggested strategies, please see CA-99-02, Trojan Horses.

  In the case of a tool like network.vbs, packet filtering may be most effective against preventing the exit of malicious packets from your network, thus preventing malicious code like network.vbs from spreading from your site to others.

Acknowledgments

We thank Abe Singer and the San Diego Supercomputer Center for contributions to this Incident Note.

Author: Kevin Houle

3 IN-2000-03: 911 Worm

Date: April 4, 2000

Overview

A worm with variants known as "chode," "foreskin," "dickhair", "firkin," or "911" has received some attention over the last week. The National Infrastructure Protection Center issued a bulletin regarding this worm, available at http://www.nipc.gov/nipc/advis00-038.htm.

This worm spreads by taking advantage of unprotected Windows shares. For more information on a similar problem and relevant solutions, please see http://www.cert.org/incident_notes/IN-2000-02.htm.

Description

The "chode" worm affects Windows 98 systems with unprotected shares. It does not function properly on Windows NT systems. We have not completed testing on Windows 95 systems or Windows 2000 systems.

As of this writing, CERT/CC has not received any direct reports of systems infected with this worm, though we have received a small number of second-hand reports.

The worm consists of several batch files, and it takes the following steps.

CHODE.BAT calls RANDOM.BAT, which picks a target network and initial host from a set of predefined networks.

Once RANDOM.BAT picks an initial machine, CHODE.BAT increments over the addresses, and for each address it

- pings a machine and listens for an answer
- on machines that answer the ping, looks for any shares using "net view \\< ip-addr>"
- tries to map the C drive on any machine with shares using "net use /yes j: \\< ip-addr>\c"
- looks for j:\windows\win.com

If it maps C and finds win.com, it then

- checks for and deletes instances of "foreskin"
- checks for and deletes instances of "mstum.pif"
- checks for and deletes instances of "dickhair"
- checks for instances of chode

If chode is not found, it begins the process of trying to infect/replicate. It

- makes the directory j:\zx
- copies test.txt to j:\zx\test.txt

If the copy is successful, it

- deletes the zx directory
- makes the directory j:\progra~1/chode
- sets chode hidden using "attrib j:\progra~1\chode +h"
- copies all chode files to j: using "copy /y c:\progra~1\chode\*.* j:\progra~1\chode"

It then selects a random number based on the time. During this process, it creates a file called "cu##ent.bat", a file called "current.bat", and an environment variable called "time".

Based on the random number, it appends a file named "chocher.bat" to autoexec.bat with probability 1/10. The new autoexec.bat (with chocher.bat appended) then

- calls 911 with a probability of 3/6, attempting to use each of COM1 through COM4
- formats D,E,F,G,H drives, issues the message *tHE cHOdE gOTcHA yOu sTUpID mOThER fUCKeR!!!!!!!!!!!!!!*, and then formats the C drive, all with probability 1/6

Chode then copies ashield.pif, netstat.pif, and winsock.vbs to the startup folder on the victim machine. When Windows next starts on the victim machine, these files begin the process again.

The winsock.vbs file then deletes all files on the C drive on the 19th day of the month.

The initiating machine then starts again with a new IP address.

We encourage you to read CERT Incident Note IN-2000-02 for information on general solutions to the problem of unprotected Windows shares.

One notable variant (foreskin) of the worm described in this document randomly copies one of a set of batch files (named A.BAT, B.BAT, C.BAT...J.DAT) to a file called MSTUM.BAT. Other variants named dickhair and firkin are similar.

Other information

Additional information about this and similar viruses and worms is available at

- http://www.antivirus.com/pc-cillin/vinfo/virusencyclo/default5.asp?VName=BAT_CHODE911
- http://vil.mcafee.com/dispVirus.asp?virus_k=98557
- http://www.sarc.com/avcenter/venc/data/bat.chode.worm.html
- http://www.sans.org/newlook/alerts/911worm.htm
- http://www.sophos.com/virusinfo/analyses/911a.html
- http://www.sophos.com/virusinfo/analyses/911b.html

Author: Shawn Hernan

4 IN-2000-04: Denial of Service Attacks using Nameservers

Updated: Monday, January 15, 2001 (changed RFC 2267 to RFC 2827/BCP 38)
Date: Friday, April 28, 2000

Overview

Intruders are using nameservers to execute packet flooding denial of service attacks.

Description

We are receiving an increasing number of reports of intruders using nameservers to execute packet flooding denial of service attacks.

The most common method we have seen involves an intruder sending a large number of UDP-based DNS requests to a nameserver using a spoofed source IP address. Any nameserver response is sent back to the spoofed IP address as the destination. In this scenario, the spoofed IP address represents the victim of the denial of service attack. The nameserver is an intermediate party in the attack. The true source of the attack is difficult for an intermediate or a victim site to determine due to the use of spoofed source addresses.

Because nameserver responses can be significantly larger than DNS requests, there is potential for bandwidth amplification. In other words, the responses may consume more bandwidth than the requests. We have seen intruders utilize multiple nameservers on diverse networks in this type of an attack to achieve a distributed denial of service attack against victim sites.

In incidents we have seen as of the date of publication, the queries are usually crafted to request the same valid DNS resource record from multiple nameservers. The result is many nameservers receiving queries for resources records in zones for which the nameserver is not authoritative. The response of the nameserver depends on it's configuration.

- If the target nameserver allows the query and is configured to be recursive or to provide referrals, the nameserver's response could contain significantly more data than the original DNS request, resulting in a higher degree of bandwidth amplification.
- A target nameserver configured without restrictions on DNS query sources may not log malicious queries at all.
- If the target nameserver is configured to restrict DNS queries by source, and the source IP address is not allowed to make queries, the nameserver's response will be a reject message with little to no bandwidth amplification. Also, the nameserver can log the malicious queries. An example syslog entry looks like this:

  ```
  Apr 27 14:26:12 intermediary.example.com named[pid]: unapproved
  recursive query from [10.1.2.3].udp-port for resource.example.net
  ```

  In this example, the IP address "10.1.2.3" represents the victim of the denial of service attack. The name "intermediary.example.com" represents an intermediary nameserver used in the attack. The name "resource.example.net" represents the DNS resource record being queried in the DNS request. Some reports we have received indicate logging malicious DNS queries at a rate as high as 5 per second during an attack.

The intermediary nameserver may receive packets back from the victim host. In particular, ICMP port unreachable packets may be returned from the victim to the intermediary in response to an unexpected UDP packet sent from the intermediary nameserver to the victim host.

Impact

Sites with nameservers used as intermediaries may experience performance degradation and a denial of DNS service as a result of an increase in DNS query traffic. It is also possible to experience higher bandwidth consumption and a bandwidth denial of service attack on the intermediary nameserver's network.

Victim sites may experience a bandwidth denial of service attack due to a high volume of DNS response packets being forwarded by one or more intermediary nameservers.

Solutions

AusCERT published an advisory in 1999 discussing denial of service attacks that utilize DNS and nameservers. For more information about the attack method, and for BIND 8 configuration strategies to mitigate the effectiveness of attacks, see

> AL-1999.004, Denial of Service (DoS) attacks using the Domain Name System (DNS)

For information about using packet filtering to prevent denial of service attacks based on IP source spoofing, see

> RFC2827/BCP 38, Defeating Denial of Service Attacks which employ IP Source Address Spoofing

> CA-96.21, TCP SYN Flooding and IP Spoofing Attacks

Author: Kevin Houle

5 IN-2000-05: "mstream" Distributed Denial of Service Tool

Date: Tuesday, May 2, 2000

Overview

In late April 2000, we began receiving reports of sites finding a new distributed denial of service (DDOS) tool that is being called "mstream". The purpose of the tool is to enable intruders to utilize multiple Internet connected systems to launch packet flooding denial of service attacks against one or more target systems.

Description

The "mstream" tool consists of a handler and an agent portion, much like previously known DDOS tools such as Trinoo. We have seen both the agent and the handler running as "rpc.wall" in binary form. The source code we have seen names the handler "master.c" and the agent "server.c".

The handler does not require administrative privileges and can function under a regular user login on a Unix system. The agent crafts forged packet headers and requires administrative (e.g., root) privileges to function.

The handler can be controlled remotely by one or more intruders using a password-protected interactive login to a running handler. Simple commands issued to the handler cause instructions to be sent to agents deployed on compromised systems. The communications between intruder and handler, and the handler and agents, are configurable at compile time and have varied significantly from incident to incident. The default protocol and destination socket numbers in source code recently released to the public are

```
intruder --------- 6723/tcp -> handler
handler  --------- 7983/udp -> agent
agent    --------- 9325/udp -> handler
```

It is important to note that any of these socket numbers can easily be altered to any value at compile-time by an intruder. For example, we have seen the handler compiled to listen for communications from the agent on UDP socket 6838 rather than 9325.

Agent binaries contain a list of handlers that are defined at compile-time by the intruder. The list of handlers is visible by running 'strings' against the agent binary. Here is an example of the output that has been edited to show easily identifiable items, including a sample list of mstream handlers.

```
192.168.1.2
192.168.3.4
192.168.5.6
Must be ran as root.
socket
```

```
        bind
        setsockopt
        newserver
        stream
        mstream
        ping
        pong
        fork
        Forked into background, pid %d
```

When an agent is first executed, it will send a "newserver" message via UDP to all known handlers. Any handlers receiving the "newserver" message record the agent in a list of known agents. The IP address of the agent is written to a disk file using a simple ASCII rotation to obscure the IP address. The contents of the file can be recovered using the following command

```
cat <filename> | tr 'b-k`' '0-9.' | sed 's/<$//'
```

IP addresses contained in this file may represent compromised hosts running mstream agents. The filename is configurable at compile-time by the intruder and we have seen various names used. Some examples we have seen are

/usr/bin/...

.sr [found in the directory containing the handler binary]

The payload of a mstream network is a packet flooding denial of service attack using TCP packets with the ACK flag set. Other observed attributes of the payload packet headers include

- random source IP address (all octets) for each packet
- random source TCP socket number for the initial packet, then incrementing for each additional packet
- random destination TCP socket number for each packet
- IP header type-of-service (TOS) field set to "0x08" for each packet
- IP header ID field random for initial packet, then incrementing for each additional packet
- IP header time-to-live (TTL) field set to 255 for each packet
- TCP header window size set to 16384 for each packet
- TCP header sequence number random for initial packet, then incrementing for each additional packet
- TCP header acknowledgment number set to 0 for each packet
- no data in the data portion of the packet

The handler can be instructed to initiate an attack using the commands 'stream' or 'mstream'. However, in versions analyzed by the CERT/CC, the 'stream' command does not function as intended due to coding errors by the author. The apparent intent for 'stream' is to cause the handler to instruct all known agents to launch a TCP ACK flood against a single target IP address for a specified duration. Future versions of the tool may correctly implement this function. The 'mstream' command causes the handler to instruct all known agents to launch a TCP ACK flood against one or more target IP addresses.

Here is sample tcpdump output showing the attack pattern. In this example, handler.example.net is running the handler and agent.example.net is running the agent. The IP addresses 10.1.1.2 and 10.1.1.3 are the victims of the attack.

- intruder sending 'mstream 10.1.1.2:10.1.1.3 5' command to handler

  ```
  11:58:43.530004 lo > intruder.example.com.1044 > handler.example.net.6723: P
  769187158:769187187(29) ack 770575957 win 31072 < nop,nop,timestamp 207945850
  207939664> (DF) (ttl 64, id 54036)
  ```

- handler echoing commands back to intruder

  ```
  11:58:43.530301 lo > handler.example.net.6723 > intruder.example.com.1044: P 1:45(44)
  ack 29 win 31072 < nop,nop,timestamp 207945850 207945850> (DF) (ttl 64, id 54037)
  ```

- handler sending 'mstream/10.1.1.2:10.1.1.3/5' command to agent

  ```
  11:58:43.530648 lo > handler.example.net.1035 > agent.example.net.7983: udp 28 (ttl 64,
  id 54038)
  ```

- agent beginning to attack two victim hosts; each source IP address and destination socket number is random

  ```
  11:58:43.531109 eth0 > xxx.xxx.xxx.xxx.2458 > 10.1.1.2.51479: .
  2110392958:2110392958(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 12979)
  11:58:43.531116 eth0 > xxx.xxx.xxx.xxx.2714 > 10.1.1.3.29405: .
  2127170174:2127170174(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 13235)
  11:58:43.531136 eth0 > xxx.xxx.xxx.xxx.2970 > 10.1.1.2.29837: .
  2143947390:2143947390(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 13491)
  11:58:43.531186 eth0 > xxx.xxx.xxx.xxx.3226 > 10.1.1.3.10268: .
  2160724606:2160724606(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 13747)
  11:58:43.531192 eth0 > xxx.xxx.xxx.xxx.3482 > 10.1.1.2.16764: .
  2177501822:2177501822(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 14003)
  11:58:43.531211 eth0 > xxx.xxx.xxx.xxx.3738 > 10.1.1.3.34732: .
  2194279038:2194279038(0) ack 0 win 16384 [tos 0x8] (ttl 255, id 14259)
  ```

Output of 'strings' run against the handler binary produces some easily recognizable output. Here is an example:

```
You're too idle !
 Connection from %s
 newserver
 New server on %s.
 pong
 Got pong number %d from %s
 %s has disconnected (not auth'd): %s
 Invalid password from %s.
 Password accepted for connection from %s.
 Lost connection to %s: %s
 stream
```

```
Usage: stream < hostname> < seconds>
Unable to resolve %s.
stream/%s/%s
Streaming %s for %s seconds.
quit
%s has disconnected.
servers
Server file doesn't exist, creating ;)
The following ips are known servers:
help
commands
Available commands:
stream          --      stream attack !
servers         --      Prints all known servers.
ping            --      ping all servers.
who             --      tells you the ips of the people logged in
mstream         --      lets you stream more than one ip at a time
Currently Online:
Socket number %d        [%s]
ping
Pinging all servers.
mstream
Usage: mstream < ip1:ip2:ip3:...> < seconds>
MStreaming %s for %s seconds.
mstream/%s/%s
fork
Forked into background, pid %d
Caught SIGHUP, ignoring.
Caught SIGINT, ignoring.
Segmentation Violation, Exiting cleanly..
Caught unknown signal, This should not happen.
__exit_dummy_decl
_send2server
_sendtoall
```

Impact

Distributed denial of service (DDOS) tools in general are capable of producing high magnitude packet flooding denial of service attacks. At the time of this writing, the "mstream" tool is capable of producing a severe denial of service condition against one or more victim sites, including sites being used as hosts for portions of a "mstream" DDOS network. However, at this time, "mstream" does not contain any functionality that significantly adds to the overall threat posed by DDOS tools in general.

Based on differences observed during analysis, we believe the code for "mstream" to be under active testing and development. The functionality of the tool may diverge from the functionality described in this Incident Note as the tool evolves.

Solutions

The CERT/CC has previously published several resources discussing distributed denial of service tools. These resources contain advice on handling distributed denial of service attacks and the associated tools.

CA-2000-01, Denial-of-Service Developments

CA-99-17, Denial-of-Service Tools

IN-99-07, Distributed Denial of Service Tools

For general information about distributed system intruder tools, please see the results of the CERT-sponsored DSIT workshop from November 2, 1999.

Results of the Distributed-Systems Intruder Tools Workshop

An independent analysis of "mstream" was produced and made available by David Dittrich - University of Washington, George Weaver - Pennsylvania State University, Sven Dietrich - NASA Goddard Space Flight Center, and Neil Long - Oxford University. It is available from http://staff.washington.edu/dittrich/misc/mstream.analysis.txt

Authors: Kevin Houle, Chad Dougherty

6 IN-2000-06: Exploitation of "Scriptlet.Typelib" ActiveX Control

Date: Tuesday, June 6, 2000

Overview

We have received reports of email-borne viruses that exploit a vulnerability created by unsafe configuration of the Microsoft ActiveX control named "Scriptlet.Typelib".

Description

The Microsoft ActiveX control Scriptlet.Typelib allows local files to be created or modified, so it is unsafe to allow untrusted programs to access this control. The control is incorrectly marked "safe for scripting" as shipped with Internet Explorer versions 4.0 and 5.0. As a result, malicious programs may be able to execute the control without requesting approval from the user. For example, an HTML-format email message that is rendered using Internet Explorer may be able to execute the Scriptlet.Typelib control to create and modify local files.

We are aware of two email-borne viruses that are designed to exploit this vulnerability. Malicious VBScript programs known as Bubbleboy and kak are designed to infect systems by altering the Windows registry and propagating themselves through email. In both cases, a malicious VBScript is delivered in the form of an HTML-format email message with characteristics that might entice a user to view the message. If the HTML in the email message is rendered by Internet Explorer, the VBScript may be executed. In vulnerable configurations, the Scriptlet.Typelib ActiveX control can be called by the malicious program to create and modify local files.

It is important to note that some mail user agents, such as Outlook 2000 and Outlook Express 5, use Internet Explorer to render HTML-format email messages. Rather than explicitly executing a malicious file attachment, a user may cause a malicious program to execute simply by viewing a message.

It is possible that other methods of delivering and executing malicious code can be used to exploit vulnerable configurations of Scriptlet.Typelib; for example, through a maliciously crafted web page.

We began receiving reports of kak and kak variants in late February 2000, and we continue to receive reports of new infections. As of this writing, we have not received any direct reports of Bubbleboy infections.

Information about kak and its variants can be found at

> **Aladdin Knowledge Systems:**
> http://www.ealaddin.com/home/csrt/valerts.asp#VBS_KAK

Computer Associates International, Inc.:
http://www.cai.com/virusinfo/encyclopedia/descriptions/wscript.htm

F-Secure:
http://www.f-secure.com/v-descs/kak.htm

Network Associates (McAfee & Dr. Solomon):
http://vil.nai.com/villib/dispVirus.asp?virus_k=10509&

Norman Data Defense Systems:
http://www.norman.no/virus_info/js_kak_worm.shtml

Proland Software:
http://www.pspl.com/virus_info/worms/kak.htm

Sophos Anti-Virus:
http://www.uk.sophos.com/virusinfo/analyses/vbskakworm.html

Symantec:
http://www.symantec.com/avcenter/venc/data/wscript.kakworm.html

Information about BubbleBoy can be found at

Central Command, Inc.:
http://www.avpve.com/viruses/worms/bubblebo.html

Computer Associates International, Inc.:
http://www.cai.com/virusinfo/encyclopedia/descriptions/bubble.htm

F-Secure:
http://www.f-secure.com/v-descs/bubb-boy.htm

Network Associates, Inc. (McAfee & Dr. Solomon's Software):
http://vil.nai.com/villib/dispVirus.asp?virus_k=10418

Norman Data Defense Systems:
http://www.norman.no/virus_info/vbs_bubble.shtml

Proland Software:
http://www.pspl.com/trojan_info/win32/bubbleboy.htm

Sophos Anti-Virus:
http://www.uk.sophos.com/virusinfo/analyses/vbsbubbleboy.html

Symantec:
http://www.symantec.com/avcenter/venc/data/vbs.bubbleboy.html

Trend Micro, Inc.:
http://www.antivirus.com/vinfo/security/sa110999.htm

Impact

Viruses or other malicious code contained in HTML-format email or web pages can exploit Scriptlet.Typelib to create and modify local files.

Solutions

Microsoft produced a patch that will remove the "safe for scripting" marking from the Scriptlet.Typelib ActiveX control. More information about the vulnerable condition and the patch is available from Microsoft at:

http://www.microsoft.com/security/bulletins/ms99-032.asp

http://www.microsoft.com/technet/security/bulletin/fq99-032.asp

http://support.microsoft.com/support/kb/articles/q240/3/08.asp

With the patch applied, the default action is for the user to be prompted before Scriptlet.Typelib is executed. Even with the patch installed, a user can choose to allow the control to be executed. If the control is allowed to execute, local files can still be created and modified.

Authors: Kevin Houle, Chad Dougherty, Brian King

7 IN-2000-07: Exploitation of Hidden File Extensions

Updated: Thursday, July 27, 2000
Date: Monday, June 19, 2000

Overview

There have been a number of recent malicious programs exploiting the default behavior of Windows operating systems to hide file extensions from the user. This behavior can be used to trick users into executing malicious code by making a file appear to be something it is not.

Description

Multiple email-borne viruses are known to exploit the fact that Microsoft Windows operating systems hide certain file extensions. The first major attack incorporating an element of file extension obfuscation was the VBS/LoveLetter worm which contained an email attachment named "LOVE-LETTER-FOR-YOU.TXT.vbs". Other malicious programs have since incorporated similar naming schemes.

- Downloader (MySis.avi.exe or QuickFlick.mpg.exe)
- VBS/Timofonica (TIMOFONICA.TXT.vbs)
- VBS/CoolNote (COOL_NOTEPAD_DEMO.TXT.vbs)

The files attached to the email messages sent by these viruses may appear to be harmless text (.txt), MPEG (.mpg), AVI (.avi) or other file types when in fact the file is a malicious script or executable. For further information about these specific viruses, please visit the sites listed on our Computer Virus Resource page.

Windows operating systems contain an option to "Hide file extensions for known file types". The option is enabled by default, but a user may choose to disable this option in order to have file extensions displayed by Windows. After disabling this option, there are still some file extensions that, by default, will continue to remain hidden from the user.

There is a registry value which, if set, will cause Windows to hide certain file extensions regardless of user configuration choices elsewhere in the operating system. The "NeverShowExt" registry value is used to hide the extensions for basic Windows file types. For example, the ".LNK" extension associated with Windows shortcuts remains hidden even after a user has turned off the option to hide extensions.

We have seen attacks which leverage file extensions that are, by default, hidden using the "NeverShowExt" registry value. One such extension, ".SHS", is associated with Shell Scrap Object files. SHS files are typically associated with OLE objects and can include executable contents. Reports indicate that SHS files are being used to distribute malicious code in email attachments. One recent example is a malicious VBScript program wrapped in a Shell Scrap Object file that is sent as an email file attachment named "LIFE_STAGES.TXT.SHS".

Impact

Users can be tricked into opening a file that appears to be something it is not. A file that appears to be innocent based on it's viewable file name may contain malicious executable code.

Solutions

In an environment where file types are mapped to functionality by the extension used in the file name, it is important for the user to know the complete and unobfuscated file name in the course of making informed decisions impacting security.

The CERT/CC encourages sites to evaluate the following suggested steps against security and usability policies at your site. To configure Windows operating systems to display entire and complete file names for all files to the user:

- **Configure Windows to show all files and extensions**

  Windows 9x and Windows NT 4.0:

  - Open the Windows Start menu
  - Select "Settings -> Control Panel" to open the control panel
  - From the "View" menu, select "Options..."
  - Click on the "View" tab
  - Insure "Hide files of these types" and "Hide file extensions for known file types" are both unchecked
  - Insure "Show all files" is selected
  - Click "OK" to complete the changes

  Windows 2000:

  - Open the Windows Start menu
  - Select "Settings -> Control Panel" to open the control panel
  - From the "Tools" menu, select "Folder options"
  - Click on the "View" tab
  - Under "Hidden files and folders", insure "Show hidden files and folders" is selected
  - Insure "Hide file extensions for known file types" is unchecked
  - Insure "Hide protected operating system files" is unchecked. Note, Windows 2000 will display a dialog asking for confirmation. Be sure to read and understand the information contained in the dialog and then click on "Yes".
  - Click "OK" to complete the changes
- **Remove all occurrences of the value "NeverShowExt" from the registry**
  - Open the Windows Start menu
  - Select "Run" and enter "regedit" to open the registry editor
  - From the "Edit" menu, select "Find"
  - Uncheck the "Keys" and "Data" entries under "Look at", and insure the "Values" entry is checked
  - Enter "NeverShowExt" in the "Find What" box and click "Find Next"
  - When a value is found, right click on the value name and select "Delete"
  - Press F3 to find the next occurrence of "NeverShowExt".

- Repeat the previous two steps until all occurrences of "NeverShowExt" have been deleted from the registry
- The computer will need to be rebooted for changes to take effect

Authors: Brian King, Kevin Houle

8 IN-2000-08: Chat Clients and Network Security

Date: Wednesday, June 21, 2000

The CERT/CC has received reports and inquiries regarding the security issues inherent in the use of chat clients.

Internet chat applications, such as instant messenging applications and Internet Relay Chat (IRC) networks, provide a mechanism for information to be transmitted between computers within a network and computers at remote sites across network borders in both directions. Chat clients provide groups of individuals the means to exchange dialog, Web URL's, and in many cases, files of any type. As with any similar networked application (e.g., email), chat applications pose security risks when used in a networked environment.

The security model of chat clients is one that relies on each end-user to make independent security decisions rather than relying on a central enforceable security policy. The result is a broader base of exposure to risk across a network with less central control, making security policies that allow chat client usage difficult to implement and enforce.

There are several general security issues network and system administrators can consider when evaluating security policies and the use of chat clients.

- Software flaws, such as buffer overflows or insecure configurations, may be present in client software and may provide a means for remote users to initiate attacks that execute code on internal systems. The configuration of chat software should be reviewed; check security settings and insure security issues have been addressed with work arounds or patches.
- Social engineering attacks may entice users into taking insecure actions, such as communicating sensitive information with outsiders or executing untrusted software. Users should be aware of the potential for social engineering attacks and use caution in releasing information and executing untrusted software.
- Information, including passwords, may be passed across untrusted networks (both domestic and international) in clear text, making them subject to interception. Strong encryption, if available, should be used to secure sensitive communications.
- For sensitive communications, it may be difficult to strongly authenticate the identity of remote parties using only the information provided in most chat clients. Strong authentication, if available, should be used to establish trusted communications.
- Attacks involving Trojan horse programs have been known to leverage chat networks to enable intruders to coordinate the actions of compromised computers in attacks against other Internet sites.

A general security practice for system configuration is to disable all services that are not needed. The same concept can be applied to network configuration. Unless the services provided by chat clients are needed in your environment, we encourage you to consider disabling chat client functionality on your network.

Author: Kevin Houle

9 IN-2000-09: Systems Compromised Through a Vulnerability in the IRIX telnet daemon

Original release date: Thursday, August 31, 2000
Last revised: Thursday, September 7, 2000
Source: CERT/CC

Overview

We have received reports of intruder activity involving the telnet daemon on SGI machines running the IRIX operating system. Intruders are actively exploiting a vulnerability in *telnetd* that is resulting in a remote root compromise of victim machines.

Information about the vulnerability we have seen exploited as a part of these attacks can be found at SGI Security Advisory 20000801-01-P, IRIX telnetd vulnerability (http://www.securityfocus.com/bid/1572).

Description

Reports of successful exploitations of the vulnerability in *telnetd* have included some or all of the following attack characteristics:

- Generation of a syslog message similar to

  ```
  overly long syslog message detected, truncating
  telnetd[xxxxx]: ignored attempt to setenv (_RLD,     ^?D^X^\
  ^?D^X^^   ^D^P^?^?$^B^Cs#^?^B^T#d~^H#e~^P/d~^P/`~^T#`~^O
  ^C ^?^?L/bin/sh
  ```

  or

  ```
  overly long syslog message, integrity compromised, aborting
  ```

- Addition of accounts with root privileges to /etc/passwd
- Remote retrieval and installation of additional intruder tools, including root kits that contain replacements for various system binaries, including *telnetd*
- Installation of packet sniffers
- Installation of irc proxy programs such as *bnc*

Solutions

Patch or disable the telnetd service

Patches for this vulnerability have been released by SGI. Sites are encouraged to follow the instructions outlined in the SGI advisory for specific instructions on how to obtain the patches. For sites that cannot immediately apply the patches, instructions for disabling the telnet service are also provided.

Restrict access to the telnetd service

Sites can employ the use of access control mechanisms, such as packet filtering, firewalls, or application-layer controls to manage the risk of intrusion on vulnerable systems.

As a good security practice in general, the CERT/CC recommends blocking unneeded ports at your network border(s). In particular to this vulnerability, sites should block TCP port 23 (telnet).

For sites which this is not feasible, the CERT/CC recommends applying an access control mechanism such as tcp_wrappers or tcpserver for the telnet service. The tcp_wrappers package can be found at ftp://ftp.porcupine.org/pub/security/index.html.

The ucspi-tcp package, including tcpserver, can be found at http://cr.yp.to/ucspi-tcp.html.

If you believe a host has been compromised, we encourage you to disconnect the host from the network and review our steps for recovering from a root compromise:
http://www.cert.org/tech_tips/root_compromise.html.

We also encourage you to ensure that your hosts are current with security patches or workarounds for well-known vulnerabilities and to regularly review security related patches released by your vendors.

Author: Chad Dougherty

Revision History

```
August 31, 2000: Initial Release
September 7, 2000: Updated information in solutions section upon
SGI's release
of patches for this vulnerability, and updated the SGI advisory num-
ber.
```

10 IN-2000-10: Widespread Exploitation of rpc.statd and wu-ftpd Vulnerabilities

Date: Friday, September 15, 2000

Overview

Recent reports involving intruder exploitation of two vulnerabilities have involved very similar intruder activity. The level of activity and the scope of the attacks suggests that intruders are using scripts and toolkits to automate attacks.

Vulnerabilities we have commonly seen exploited as a part of these attacks include:

CA-2000-17, Input Validation Problem in rpc.statd

CA-2000-13, Two Input Validation Problems In FTPD

Of the two vulnerabilities discussed in CA-2000-13, the "Site exec" vulnerability is the one we are seeing exploited as a part of this activity.

Description

Sites involved in related incidents are reporting finding hosts compromised through one of these two vulnerabilities. In several cases, hundreds of compromised hosts have been involved in single incidents. Intruders appear to be using automated tools to probe for and exploit vulnerable hosts on a widespread scale.

A large majority of the compromised hosts involved in this activity have been running various versions of Red Hat Linux. Insecure default configurations in some versions, especially with respect to the vulnerable rpc.statd service often being enabled during automated installation and upgrade processes, have contributed to the widespread success of these attacks.

Intruders searching for vulnerable machines are performing widespread scanning for vulnerable systems across large blocks of address space. The scans target the following services:

- sunrpc (e.g., portmap) on ports 111/udp and 111/tcp
- ftp on port 21/tcp

In many cases, sites report receiving exploit attempts against both rpc.statd and wu-ftpd immediately after receiving probes. There is evidence to suggest intruders may be developing worm-like attack tools based on exploitations of rpc.statd and wu-ftpd.

Once hosts are compromised, there are several common patterns in the tools being installed by intruders.

't0rnkit' rootkit

Since May of 2000, we have observed more than six different versions of a rootkit being called 't0rnkit', or 'tornkit'. Rootkits are not a new idea and have been employed by intruders for several years. The important thing here is to be aware of the widespread nature of this particular activity and to insure compromised hosts are recovered using appropriate procedures and techniques. Various versions of 't0rnkit' include an installation script which attempts many of the following things

- killing syslogd
- alerting the intruder to remote logging facilities by searching the syslog configuration file for the '@' character
- storing an intruder-supplied password for trojan horse programs in /etc/ttyhash
- installing a trojan horse version of sshd configured to listen on an intruder-supplied port number with intruder-supplied SSH keys stored in a directory named '/usr/info/.t0rn'. The trojan horse binary is installed as /usr/sbin/nscd and started using '/usr/sbin/nscd -q'. The same command is appended to /etc/rc.d/rc.sysinit to start the daemon at system boot time.
- locating trojan horse configuration files to hide file names, process names, etc. in a directory named '/usr/src/.puta'
- replacing the following system binaries with trojan horse copies
  - /bin/login
  - /sbin/ifconfig
  - /bin/ps
  - /usr/bin/du
  - /bin/ls
  - /bin/netstat
  - /usr/sbin/in.fingerd
  - /usr/bin/find
  - /usr/bin/top
- installing a password sniffer, sniffer logfile parser, and system logfile cleaning tool in /usr/src/.puta
- attempting to enable telnet, shell, and finger in /etc/inetd.conf by removing any leading '#' comment characters
- alerting the intruder about the word 'ALL' appearing in /etc/hosts.deny
- some versions attempt to patch rpc.statd and wu-ftpd with versions that are not vulnerable.
- restarting /usr/sbin/inetd
- starting syslogd

Most versions also include a trojan horse version of tcp_wrappers in RPM format named 'tcpd.rpm'. There is strong evidence that 't0rnkit' is undergoing active development at the time of this writing, so the exact composition of the rootkit may vary from this description over time.

Distributed Denial of Service Tools

In addition to the installation of rootkits, we have observed a significant increase in the installation of distributed denial of service (DDoS) tools on hosts compromised through these two vulnerabilities. In one incident, we recorded over 560 hosts at 220 Internet sites around the world as being a part of a Tribe Flood Network 2000 (TFN2K) DDoS network. The hosts we were able to identify were compromised via either the rpc.statd or wu-ftpd vulnerabilities. We have commonly seen the following DDoS tools installed by intruders.

- Tribe Flood Network (TFN) - see IN-99-07, Distributed Denial of Service Tools
- Tribe Flood Network 2000 (TFN2K) - see CA-99-17, Denial-of-Service Tools
- Stacheldraht 1.666+smurf+yps - modified version of the tool discussed in CA-2000-01 Denial-of-Service Developments

For more information about distributed denial of service attacks, please see

Results of the Distributed-Systems Intruder Tools Workshop - HTML format
Results of the Distributed-Systems Intruder Tools Workshop - PDF format

Impact

The combination of widespread, automated exploitation of two common vulnerabilities and an associated increase in distributed denial of service tool installation poses a significant threat to Internet sites and the Internet infrastructure.

Solutions

The CERT/CC encourages all Internet sites to review the rpc.statd advisory (CA-2000-17) and the wu-ftpd advisory (CA-2000-13) and insure workarounds or patches have been applied on all affected hosts on your network.

If you believe your host has been compromised, please follow the steps outlined in Steps for Recovering From a Root Compromise.

Author: Kevin Houle