



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

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

## Interested in learning more?

Check out the list of upcoming events offering  
"Intrusion Detection In-Depth (Security 503)"  
at <http://www.giac.org/registration/gcia>

SANS GIAC  
Intrusion Detection Practical Assignment  
SANS Network Security 2000  
Mike Worman, 11/10/2000

© SANS Institute 2000 - 2002, Author retains full rights.

# Assignment 1 : Network Detects

All raw data in this section is comprised of Snort 1.6.3 packet and portscan logs.

## **Detect 1 – Portscan for lpd services**

Raw Data:

```
Nov 6 02:04:51 attacker.net:3999 -> my.net.118:515 SYN **S*****
Nov 6 02:04:51 attacker.net:4006 -> my.net.125:515 SYN **S*****
Nov 8 10:32:24 attacker.net:1300 -> my.net.241:5151 SYN **S*****
Nov 8 10:32:24 attacker.net:1304 -> my.net.245:5151 SYN **S*****
```

### 1. Source of Trace:

This trace was captured across a campus-wide network.

### 2. Detect was generated by:

Snort 1.6.3, using rule set 10042k, running as a host-based IDS on an administrative system.

### 3. Probability the source address was spoofed:

Low, since the scanning host must receive the response traffic generated by the scan. There is a small possibility that the hping2 “spoof scan” could be used.

### 4. Description of the Attack:

This is a SYN/FIN scan directed towards port 515. This is the well-known port for the printing daemon “lpd”. There are several well known vulnerabilities with OS-bundled lpd packages, such as those detailed in CVE-1999-0299, CAN-2000-0839, or CAN-2000-0879.

### 5. Attack Mechanism:

Most of the lpd exploits are buffer overflows : the attacker is hoping to send tainted data to the lpd daemon listening on port 515 in order to execute arbitrary code as the root user.

### 6. Correlations:

We correlated this scan with a residential-based IDS sensor, also running Snort 1.6.3. By using such remote sensors, we can verify the scope of this scan, which probably covered our entire class B address space. SANS GIAC also recently posted an advisory on these scans at <http://www.sans.org/newlook/alerts/port515.htm>. Interestingly enough, scans for port 5151 and 1515 have also been seen, suggesting either a type or a programming error on the part of the attacker.

### 7. Evidence of active targeting:

It is highly probable that the scan was targeted at the entire Class B address block used by the University, although the exploit being scanned for was very specific.

8. Severity:

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality : 5

The scan covered all of campus, including both administrative and residential networks.

Lethality : 5

There are an unknown number of Linux systems on campus, many of which are not secured. The lpd buffer overflow leads to immediate root compromise.

System Countermeasures : 2

Most administrative hosts are secured and patched, but student machines and academic departments may be exploited.

Network Countermeasures : 2

The campus network is open, thus there is no firewall and router filtering is minimal.

**Severity = (5 + 5) – (2 + 2) = 6**

9. Defensive recommendation:

It would be wise to re-check all administrative systems running print services to ensure that no vulnerable versions of lpd are running. For our purposes, blocking port 515 at the border router is not an option, but most organizations should not be running print services outside their internal LAN. In such instances, blocking port 515 at the router will provide a defense against this type of attack. The following ACL will do this:

```
access-list 11x deny tcp any any 515
```

10. Multiple choice test question:

What type of event does the following trace suggest?

```
Nov 6 02:04:51 attacker.net:3999 -> my.net.118:515 SYN **S*****  
Nov 6 02:04:51 attacker.net:4006 -> my.net.125:515 SYN **S*****  
Nov 8 10:32:24 attacker.net:1300 -> my.net.241:5151 SYN **S*****  
Nov 8 10:32:24 attacker.net:1304 -> my.net.245:5151 SYN **S*****
```

- a) A DNS lookup
- b) A FIN scan
- c) A Windows-type Traceroute
- d) A scan for exploitable print daemons

## Detect 2 – Full-force targeted system scan and OS fingerprint

Raw Data:

```
Oct 31 12:11:32 rude.com:2841 -> my.net.67:932 SYN **S*****
Oct 31 12:11:32 rude.com:2842 -> my.net.67:1552 SYN **S*****
Oct 31 12:11:32 rude.com:2843 -> my.net.67:863 SYN **S*****
Oct 31 12:11:32 rude.com:2844 -> my.net.67:174 SYN **S*****
Oct 31 12:11:33 rude.com:2855 -> my.net.67:47557 SYN **S*****
Oct 31 12:11:33 rude.com:2856 -> my.net.67:1672 SYN **S*****
Oct 31 12:11:33 rude.com:2857 -> my.net.67:2501 SYN **S*****
Oct 31 12:11:33 rude.com:2858 -> my.net.67:373 SYN **S*****
Oct 31 12:11:33 rude.com:2859 -> my.net.67:11 SYN **S*****
Oct 31 12:11:33 rude.com:2853 -> my.net.67:1533 SYN **S*****
Oct 31 12:11:34 rude.com:2860 -> my.net.67:4132 SYN **S*****
Oct 31 12:11:34 rude.com:2861 -> my.net.67:460 SYN **S*****
Oct 31 12:11:34 rude.com:2862 -> my.net.67:556 SYN **S*****
Oct 31 12:11:34 rude.com:2863 -> my.net.67:722 SYN **S*****
Oct 31 12:11:34 rude.com:2866 -> my.net.67:1110 SYN **S*****
Oct 31 12:11:34 rude.com:2868 -> my.net.67:215 SYN **S*****
Oct 31 12:11:34 rude.com:2869 -> my.net.67:85 SYN **S*****
Oct 31 12:11:34 rude.com:2870 -> my.net.67:675 SYN **S*****
```

### 1. Source of Trace:

This attack was detected by the host-based IDS on a network specialist's workstation.

### 2. Detect was generated by:

Snort 1.6.3, using rule set 10042k, running as a host-based IDS.

### 3. Probability that the source address was spoofed.

A brute force scan such as this requires responses to be directed back to the initiator, so it is unlikely that the source address was spoofed. In fact, the source address was later verified as correct.

### 4. Description of attack:

The attacker was actually a novice web administrator whose Linux web server had been compromised earlier in the week, and had decided to perform brute-force portscans against anyone visiting up web page. We politely pointed out how rude this was. The attack itself is a SYN scan.

### 5. Attack mechanism:

The source sends a connection request (A TCP segment with the SYN flag bit enabled) to the target, with the intent of collecting a list of listening ports. This scan is randomly stepping through all 65,535 TCP ports.

6. Correlations:

As this was a directed attack against one system, there were no other correlations.

7. Evidence of active targeting:

This scan was directed at the single host on our network that had accessed the attackers web site, so there was active targeting.

8. Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality : 3

The targeted system was a network administration workstation, but not a critical system.

Lethality : 1

The target system is not running any vulnerable services, nor is it susceptible to SYN flooding.

System Countermeasures : 5

The system is highly secured with no known exploitable services, and it is patched daily.

Network Countermeasures : 1

The campus network is open, thus there is no firewall and router filtering is minimal. Some of the SYN segments were blocked (NetBIOS, NFS, and portmap are blocked at the border).

**Severity = (3+1) – (5 + 1) = -2**

9. Defensive recommendation:

Overt portscans like this one are extremely common. For an open University environment, they are a daily occurrence. In this particular case we did not feel that a router ACL was justified, but if necessary the following ACL would do the trick:

```
access-list 11x deny ip aaa.bbb.ccc.0 0.0.0.255 any
```

where aaa.bbb.ccc.0 is the IP block assigned to rude.net.

10. Multiple choice test question:

```
Oct 31 12:11:33 rude.com:2858 -> my.net.67:373 SYN **S*****
```

```
Oct 31 12:11:33 rude.com:2859 -> my.net.67:11 SYN **S*****
```

```
Oct 31 12:11:33 rude.com:2853 -> my.net.67:1533 SYN **S*****
```

- a) A TCP retransmission
- b) A TCP traceroute
- c) A SYN scan for listening services
- d) A buffer overflow attempt









5. Attack mechanism:

Aside from “looking” like a standard SYN/FYN scan, the signature of this attack has the following interesting characteristics:

```
[**] SCAN-SYN FIN [**]  
10/24-07:20:51.260067 attacker.net:21 -> my.net.67:21  
TCP TTL:28 TOS:0x0 ID:39426  
**SF*** Seq: 0x536E7668 Ack: 0x384A53E3 Win: 0x404
```

First, the source and destination TCP ports are identical. Although some services (like DNS) operate this way, we already know this is very BAD traffic. In my opinion, the fact that the scan is looking for port 9704 (a known back door) is circumstantial. This scan is trying to **bypass firewall configurations**. Many firewalls are configured to let common services (FTP, HTTP, SSH, etc) through the firewall. Thus an administrator may have his firewall set to “allow incoming FTP connections”. If the firewall implements this as “allow incoming to TCP port 21 where SYN is enabled”, the SYN/FIN packet may well make it through. Once through, the firewall may not block outgoing traffic on the same port. If not, this attack can be used for network reconnaissance against a weak or misconfigured firewall.

Another signature of this packet is that ALL such scans reported to GIAC and elsewhere have seen:

```
Window Size = 0x404  
IP ID = 39426  
TCP Sequence Number = 0x536E7668
```

6. Correlations:

There have been correlations on this type of traffic on GIAC since late October.

```
http://www.sans.org/y2k/102500.htm  
http://www.sans.org/y2k/102600.htm  
http://www.sans.org/y2k/102700.htm
```

7. Evidence of active targeting:

Most of the scans have been across multiple hosts for arbitrary services or trojan ports.

8. Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality : 5

The scan covered all of campus, including both administrative and residential networks.

Lethality : 1

This attack is indicative of network reconnaissance, and is not particularly lethal.

System Countermeasures : 2

Although all core University systems are regularly patched and audited, many departmental and residential systems are not. Reconnaissance may provide the attacker with a list of possible victims.

Network Countermeasures : 2

The scan may allow reconnaissance of departmental network that are using insufficient firewall measures.

**Severity = (5 + 1) – (2 + 2) = 2**



# Assignment 2 : Evaluate an Attack

## The “QAZ” Trojan

### 1 - Attack Source :

We obtained this attack from our own network, later identifying it as a recently discovered trojan/worm. The network traces we obtained were captured by purposely infecting a pair of isolated lab systems. A detailed description of this attack can be found at:

[www.antivirus.com/pcillin/vinfo/virusencyclo/default5.asp?VName=TROJ\\_QAZ.A](http://www.antivirus.com/pcillin/vinfo/virusencyclo/default5.asp?VName=TROJ_QAZ.A)

### 2 - Attack Description :

The “QAZ” trojan is a Windows “worm” that affects all known versions of the Windows operating system. It was discovered in the wild in China around July 2000, and by last Fall it had infected thousands of systems. We discovered several dozen infections on our University campus, and just a few weeks later the Microsoft break-in was reported.

QAZ is an interesting example of a network-borne worm because of its ability to spread over an arbitrary number of Windows based hosts that use TCP/IP for file sharing. It also provides a “back door” interface for an attacker to remotely run commands and upload files. Thus, QAZ can be used as bounce point for installing more powerful trojan programs such as Back Orifice or SubSeven.

Its initial infection vector is variable, and can be caused by opening an executable email attachment or running a program downloaded from the web. Once running, the program takes the following actions:

-Creates the following Registry key:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run\startIE=%path%\Notepad.exe qazwsx.hsq
```

- Opens a listening TCP port at 7597. This port is a simple backdoor interface, accessible through telnet or netcat. The backdoor proves a “:” prompt and expects a password, which once entered will allow the remote host to perform scans and upload binaries.
- The trojan opens a TCP connection to port 25 (SMTP) of a remote IP and sends an email containing the infected machine’s IP address to a “collector” mail account. The initial account was a free email account provided by a Chinese ISP, but since QAZ’s release, several versions (including the Microsoft compromise) have used other email destinations.
- The infected host begins scanning for hosts, assuming a class B address (e.g. 192.168.0.0) with 24-bit subnet masks (e.g. 192.168.1.x, 192.168.2.x, etc). It searches for the Windows NetBIOS Session Service (TCP port 139) which is used for connecting to Windows folders that are shared on the network.
- If a scanned machine has TCP port 139 open, a connection will be made and all available shares will be searched. The trojan looks for any shares which contain the string “WIN”, which will match either “C:\WINDOWS” or “C:\WINNT” for most Microsoft systems.
- If the share allows write access and is not password-protected, the trojan will replace NOTEPAD.EXE with a copy of itself, and rename that copy NOTEPAD.EXE. It will move the original copy of NOTEPAD.EXE to NOTE.COM.

- Since the Windows registry is also located in the "C:\WIN\*" folders, it is modified to include the Registry entry that allows the trojan to start upon a reboot.

- Upon rebooting the newly infected machine, the QAZ trojan quietly starts, and the process continues.

### 3 – Network Trace:

The following network traffic was obtained using the following hosts:

- qaz.infected.sys** = A Windows 2000 system with the QAZ binaries available.
- qaz.victim.sys** = A Windows 98 system sharing its C drive with no password.
- Subnet.A.1, 2, 3** = Other Windows systems on the lab network, with no open shares.

The attack is initiated by running the QAZ NOTEPAD.EXE on qaz.infected.sys

Running "netstat -an" on qaz.infected.sys shows the following:

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:1027	0.0.0.0:0	LISTENING
TCP	128.119.175.126:139	0.0.0.0:0	LISTENING
<b>TCP</b>	<b>128.119.175.126:7597</b>	<b>0.0.0.0:0</b>	<b>LISTENING</b>
UDP	128.119.175.126:137	*.*	
UDP	128.119.175.126:138	*.*	

At this point, the infected machine will begin scanning for open shares.

Running "netstat -an" on qaz.infected.sys shows the following:

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:1027	0.0.0.0:0	LISTENING
TCP	128.119.175.126:139	0.0.0.0:0	LISTENING
TCP	128.119.175.126:7597	0.0.0.0:0	LISTENING
<b>TCP</b>	<b>128.119.175.126:1077</b>	<b>subnet.A.1:139</b>	<b>SYN_SENT</b>
<b>TCP</b>	<b>128.119.175.126:1078</b>	<b>subnet.A.2:139</b>	<b>SYN_SENT</b>
<b>TCP</b>	<b>128.119.175.126:1079</b>	<b>subnet.A.3:139</b>	<b>SYN_SENT</b>
...			
UDP	128.119.175.126:137	*.*	
UDP	128.119.175.126:138	*.*	

#### When the scan hits qaz.victim.sys, a typical NetBIOS session is created:

```
=====  
11/21-14:58:29.294279 qaz.infected.sys:1701 -> qaz.victim.sys:139  
TCP TTL:128 TOS:0x0 ID:11512 DF  
**S**** Seq: 0x4F5FDEE8 Ack: 0x0 Win: 0x4000  
TCP Options => MSS: 1460 NOP NOP SackOK  
=====  
11/21-14:58:29.294633 qaz.victim.sys:139 -> qaz.infected.sys:1701  
TCP TTL:128 TOS:0x0 ID:37121 DF  
**S***A* Seq: 0x18B513 Ack: 0x4F5FDEE9 Win: 0x2238  
TCP Options => MSS: 1460 NOP NOP SackOK  
=====  
11/21-14:58:29.294700 qaz.infected.sys:1701 -> qaz.victim.sys:139  
TCP TTL:128 TOS:0x0 ID:11513 DF  
*****A* Seq: 0x4F5FDEE9 Ack: 0x18B514 Win: 0x4470  
=====
```



**The trojan finds a folder which matches the "WIN" string...**

```
=====  
11/21-14:58:29.607054 128.119.175.126:1701 -> 128.119.175.100:139  
TCP TTL:128 TOS:0x0 ID:11527 DF  
****PA* Seq: 0x4F5FE33A Ack: 0x18C83F Win: 0x4435  
00 00 00 2B FF 53 4D 42 08 00 00 00 00 18 07 00 ...+.SMB.....  
00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE .....  
00 00 28 00 00 08 00 04 5C 57 49 4E 39 38 00 ..(.....\WIN98.
```

**The trojan finds its target and begins to download its payload, the trojan NOTEPAD.EXE.**

```
=====  
11/21-14:58:34.037057 qaz.infected.sys:1701 -> qaz.victim.sys:139  
TCP TTL:128 TOS:0x0 ID:11532 DF  
****PA* Seq: 0x4F5FE369 Ack: 0x18C87A Win: 0x43FA  
00 00 00 54 FF 53 4D 42 2D 00 00 00 00 18 07 00 ...T.SMB-.....  
00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE .....  
00 00 2C 00 0F FF 00 DE DE 01 00 40 00 16 00 00 ..,.....@....  
00 95 8D 1A 3A 01 00 00 00 00 00 FF FF FF FF 00 .....  
00 00 00 13 00 5C 57 49 4E 39 38 5C 6E 6F 74 65 ..... \WIN98\note  
70 61 64 2E 65 78 65 00 pad.exe.
```

**In this case, the victim machine actually had C:\WINDOWS and C:\WIN98 folders, and QAZ found them both:**

```
=====  
11/21-14:58:34.056394 128.119.175.126:1701 -> 128.119.175.100:139  
TCP TTL:128 TOS:0x0 ID:11534 DF  
****PA* Seq: 0x4F5FE3F2 Ack: 0x18C8DC Win: 0x4398  
00 00 00 56 FF 53 4D 42 2D 00 00 00 00 18 07 00 ...V.SMB-.....  
00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE .....  
00 00 34 00 0F FF 00 DE DE 01 00 40 00 16 00 00 ..4.....@....  
00 9A 8D 1A 3A 01 00 00 00 00 00 FF FF FF FF 00 .....  
00 00 00 15 00 5C 57 49 4E 44 4F 57 53 5C 6E 6F ..... \WINDOWS\no  
74 65 70 61 64 2E 65 78 65 00 tepad.exe.
```

**Next, the trojan initiates a move of the original NOTEPAD.EXE to NOTE.COM in the same folder.**

```
=====  
11/21-14:58:34.125952 qaz.infected.sys:1701 -> qaz.victim.sys:139  
TCP TTL:128 TOS:0x0 ID:11539 DF  
****PA* Seq: 0x4F5FE557 Ack: 0x18C9DB Win: 0x4299  
00 00 00 4E FF 53 4D 42 07 00 00 00 00 18 07 00 ...N.SMB.....  
00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE .....  
00 00 44 00 01 16 00 29 00 04 5C 57 49 4E 44 4F ..D...).\WINDO  
57 53 5C 6E 6F 74 65 70 61 64 2E 65 78 65 00 04 WS\notepad.exe..  
5C 57 49 4E 44 4F 57 53 5C 6E 6F 74 65 2E 63 6F \WINDOWS\note.co  
6D 00 m.
```

**The beginning of the actual data transfer of the trojan binary is show below:**

```
=====  
11/21-14:58:34.260172 qaz.infected.sys:1701 -> qaz.victim.sys:139  
TCP TTL:128 TOS:0x0 ID:11543 DF  
****PA* Seq: 0x4F5FEBEB Ack: 0x18CA70 Win: 0x4204  
28 FD FF FF 52 68 50 8A 41 00 8D 85 FC FD FF FF (...Rhp.A.....  
50 E8 4A 1E 00 00 83 C4 10 8D 8D FC FD FF FF 89 P.J.....  
8D 50 FE FF FF 8D 55 F8 52 8D 85 D8 FC FF FF 50 .P...U.R.....P  
E8 AB 1D 00 00 83 C4 08 89 85 4C FE FF FF 83 BD .....L.....  
4C FE FF FF 00 74 3D 8B 8D 4C FE FF FF 8D 95 D8 L...t=.L.....
```

**Finally, once transfer has completed the attacking host tears down the session:**

```

=====
11/21-14:58:53.816631 128.119.175.100:139 -> 128.119.175.126:1701
TCP TTL:128 TOS:0x0 ID:52737 DF
*****A* Seq: 0x18CC0A Ack: 0x4F60632B Win: 0x2238

=====
11/21-14:58:56.550590 128.119.175.100:139 -> 128.119.175.126:1701
TCP TTL:128 TOS:0x0 ID:53761 DF
****PA* Seq: 0x18CC0A Ack: 0x4F60632B Win: 0x2238
00 00 00 25 FF 53 4D 42 0B 00 00 00 00 98 07 00      ...%.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE      .....
00 00 50 00 01 28 0B 00 00                          ..P..(...)

=====
11/21-14:58:59.759458 128.119.175.100:139 -> 128.119.175.126:1701
TCP TTL:128 TOS:0x0 ID:54273 DF
****PA* Seq: 0x18CC0A Ack: 0x4F60632B Win: 0x2238
00 00 00 25 FF 53 4D 42 0B 00 00 00 00 98 07 00      ...%.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE      .....
00 00 50 00 01 28 0B 00 00                          ..P..(...)

=====
11/21-14:59:06.177082 128.119.175.100:139 -> 128.119.175.126:1701
TCP TTL:128 TOS:0x0 ID:54529 DF
****PA* Seq: 0x18CC0A Ack: 0x4F60632B Win: 0x2238
00 00 00 25 FF 53 4D 42 0B 00 00 00 00 98 07 00      ...%.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 01 C8 FF FE      .....
00 00 50 00 01 28 0B 00 00                          ..P..(...)

=====
11/21-14:59:06.177539 128.119.175.126:1701 -> 128.119.175.100:139
TCP TTL:128 TOS:0x0 ID:11589
****R*** Seq: 0x4F60632B Ack: 0x4F60632B Win: 0x0

=====

```

Since this was a lab event, neither infected machine was able to open an SMTP connection to the collector host. The email simply contains the IP address of the newly infected host.

“netstat -an” on qaz.victim.sys showed the following shortly after re-booting to Windows:

Proto	Local Address	Foreign Address	State
TCP	qaz.victim.sys:139	0.0.0.0:	LISTENING
TCP	qaz.victim.sys:7597	0.0.0.0:	LISTENING
TCP	<b>qaz.victim.sys:1077</b>	<b>subnet.A.1:139</b>	<b>SYN_SENT</b>
TCP	<b>qaz.victim.sys:1078</b>	<b>subnet.A.2:139</b>	<b>SYN_SENT</b>
TCP	<b>qaz.victim.sys:1079</b>	<b>subnet.A.3:139</b>	<b>SYN_SENT</b>
TCP	<b>qaz.victim.sys:1079</b>	<b>qaz.infected.sys:139</b>	<b>SYN_SENT</b>
...			
UDP	qaz.victim.sys:137	*:*	
UDP	qaz.victim.sys:138	*:*	

Thus, the cycle of proliferation continues.



## Assignment 3 : “Analyze This”

### **Threat Summary:**

After compiling the various data sources for GIAC Enterprises, the following suspicious network activity has been detected. Although these events and correlations are a subset of all the possible malicious network activity, they form a brief summary of immediate threats.

The following hosts/networks deserve special attention:

### **Major Trojan Activity:**

- 35.10.82.111 - This host performed a scan for SubSeven (port 27374) across the entire network on August 16<sup>th</sup>. See **Listing #1**.
- 168.120.0.0/16 - Several hosts on this network performed SubSeven scans to MY.NET.97.248 on August 18<sup>th</sup>. This network also performed WinGate Proxy scans for several days in September. It seems to be scanning the local network regularly for trojan ports. See **Listing #2**.
- 24.180.134.156 - Performed a directed NMAP fingerprinting and SYN portscan on September 11<sup>th</sup> against the MY.NET.208.x subnet. This is a possible precursor to an attack on that network. See **Listing #3**.

### **Major Port Scans / Network Reconnaissance:**

- 130.149.41.70 - Performed a number of OS fingerprinting techniques against MY.NET.217.46 from August 17<sup>th</sup> to August 18<sup>th</sup>. See **Listing #4**.
- 195.114.226.41 - Performed a SYN scan of the entire network for FTP on August 15<sup>th</sup>. See **Listing #5**.
- 206.186.79.9 - Stepped through the entire network, randomly switching looking for DNS servers. See **Listing #6**.
- 24.92.188.4 - Performed an OS fingerprinting scan directed at a particular host, MY.NET.106.164, on September 11<sup>th</sup> and 14<sup>th</sup>. See **Listing #7**.
- 205.238.205.3 - Performed a TELNET scan of the entire network on September 11<sup>th</sup>. See **Listing #8**.
- 210.61.144.125 - Performed a type of SYN/FIN scan on September 11<sup>th</sup> against the MY.NET.4.x subnet. This new scan is currently being analyzed by SANS GIAC. See **Listing #9**.
- 159.226.0.0/16 - A number of hosts on this network have been been accessing various ports on various hosts on this network. There has also been a large volume of UDP traffic to various high ephemeral ports. It is highly possible that this traffic is malicious, although no packet log data exists for this remote host. See **Listing #10**.

## Possible Employee Abuse:

The following systems seem to engage regularly in the Napster cooperative file sharing system, which may or may not violate your corporate policies. See Listing #11 for examples of this activity.

MY.NET.181.87  
MY.NET.221.94  
MY.NET.157.200

## Traces:

### Listing #1 – SubSeven Trojan Scan

08/16-05:06:07 35.10.82.111:1031 -> MY.NET.105.203:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1033 -> MY.NET.105.205:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1037 -> MY.NET.105.209:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1039 -> MY.NET.105.211:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1040 -> MY.NET.105.212:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1046 -> MY.NET.105.218:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1047 -> MY.NET.105.219:27374 SYN \*\*S\*\*\*\*\*  
08/16-05:06:07 35.10.82.111:1049 -> MY.NET.105.221:27374 SYN \*\*S\*\*\*\*\*

### Listing #2 – General Trojan Scanning

08/11-02:57:54.817032 [\*\*] WinGate 1080 Attempt [\*\*] 168.120.16.250:53391 ->  
MY.NET.98.197:1080  
08/11-03:58:08.734570 [\*\*] WinGate 1080 Attempt [\*\*] 168.120.16.250:56852 ->  
MY.NET.98.197:1080  
08/18-01:14:43 168.120.26.87:3497 -> MY.NET.97.248:44444 SYN \*\*S\*\*\*\*\*  
08/18-01:14:43 168.120.26.87:3498 -> MY.NET.97.248:12631 SYN \*\*S\*\*\*\*\*  
08/18-01:14:43 168.120.26.87:3504 -> MY.NET.97.248:6670 SYN \*\*S\*\*\*\*\*  
08/18-01:14:43 168.120.26.87:3506 -> MY.NET.97.248:5742 SYN \*\*S\*\*\*\*\*  
08/18-03:35:16 168.120.13.177:1755 -> MY.NET.97.248:27374 SYN \*\*S\*\*\*\*\*  
09/02-00:20:56.463518 [\*\*] WinGate 1080 Attempt [\*\*] 168.120.16.250:55419 ->  
MY.NET.97.212:1080  
09/02-01:45:08.742040 [\*\*] WinGate 1080 Attempt [\*\*] 168.120.16.250:58699 ->  
MY.NET.97.212:1080  
09/02-01:58:53.198145 [\*\*] WinGate 1080 Attempt [\*\*] 168.120.16.250:59371 ->  
MY.NET.97.212:1080

### Listing #3 – NMAP Fingerprint and Portscan

09/11-04:50:30 24.180.134.156:1328 -> MY.NET.208.21:1083 SYN \*\*S\*\*\*\*\*  
09/11-04:50:30 24.180.134.156:1329 -> MY.NET.208.21:215 SYN \*\*S\*\*\*\*\*  
09/11-04:50:30 24.180.134.156:1330 -> MY.NET.208.21:682 SYN \*\*S\*\*\*\*\*  
09/11-04:50:32.156163 [\*\*] Null scan! [\*\*] 24.180.134.156:50110 -> MY.NET.208.21:23  
09/11-04:50:32.156163 [\*\*] Null scan! [\*\*] 24.180.134.156:50110 -> MY.NET.208.21:23  
09/11-04:50:32.160549 [\*\*] Probable NMAP fingerprint attempt [\*\*] 24.180.134.156:50111 ->  
MY.NET.208.21:23  
09/11-04:50:32.160549 [\*\*] Probable NMAP fingerprint attempt [\*\*] 24.180.134.156:50111 ->  
MY.NET.208.21:23  
09/11-04:50:32 24.180.134.156:50109 -> MY.NET.208.21:23 SYN 2\*S\*\*\*\*\* RESERVEDBITS  
09/11-04:50:32 24.180.134.156:50110 -> MY.NET.208.21:23 NULL \*\*\*\*\*  
09/11-04:50:32 24.180.134.156:50111 -> MY.NET.208.21:23 NMAPID \*\*SF\*P\*U

#### Listing #4 – OS Fingerprinting

08/18-12:28:56 130.149.41.70:1123 -> MY.NET.217.46:994 INVALIDACK 2\*SFR\*A\*  
RESERVEDBITS  
08/18-12:29:09 130.149.41.70:1123 -> MY.NET.217.46:994 INVALIDACK \*\*\*FRPAU  
08/18-12:29:34 130.149.41.70:16 -> MY.NET.217.46:1123 INVALIDACK \*\*\*FRPAU  
08/18-12:29:37 130.149.41.70:1123 -> MY.NET.217.46:994 INVALIDACK 2\*SF\*\*A\*  
RESERVEDBITS  
08/18-12:31:13 130.149.41.70:1123 -> MY.NET.217.46:994 VECNA \*\*\*F\*\*\*U  
08/18-12:31:45 130.149.41.70:16 -> MY.NET.217.46:1123 INVALIDACK 2\*SF\*\*A\*  
RESERVEDBITS

#### Listing #5 – FTP Scan

08/15-00:52:54 195.114.226.41:4809 -> MY.NET.26.184:21 SYN \*\*S\*\*\*\*\*  
08/15-00:52:54 195.114.226.41:4810 -> MY.NET.26.185:21 SYN \*\*S\*\*\*\*\*  
08/15-00:52:54 195.114.226.41:4812 -> MY.NET.26.187:21 SYN \*\*S\*\*\*\*\*  
08/15-00:52:54 195.114.226.41:4813 -> MY.NET.26.188:21 SYN \*\*S\*\*\*\*\*  
08/15-00:52:54 195.114.226.41:4814 -> MY.NET.26.189:21 SYN \*\*S\*\*\*\*\*  
08/15-00:52:54 195.114.226.41:4815 -> MY.NET.26.190:21 SYN \*\*S\*\*\*\*\*

#### Listing #6 – DNS Scan

09/10-00:27:07 206.186.79.9:4836 -> MY.NET.1.0:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4838 -> MY.NET.1.2:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4840 -> MY.NET.1.4:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4841 -> MY.NET.1.5:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4843 -> MY.NET.1.7:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4847 -> MY.NET.1.11:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4856 -> MY.NET.1.20:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4857 -> MY.NET.1.21:53 SYN \*\*S\*\*\*\*\*  
09/10-00:27:07 206.186.79.9:4858 -> MY.NET.1.22:53 SYN \*\*S\*\*\*\*\*

#### Listing #7 – Directed OS Fingerprint

09/14-16:23:19 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS  
09/14-16:23:19 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS  
09/14-16:24:38 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS  
09/14-16:24:38 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS  
09/14-16:26:38 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS  
09/14-16:26:38 24.92.188.4:4269 -> MY.NET.106.164:6699 INVALIDACK 2\*S\*R\*A\*  
RESERVEDBITS

Listing #8 – Telnet Server Scan

```
09/11-13:19:29 205.238.205.3:10475 -> MY.NET.201.203:23 SYN **S*****
09/11-13:19:29 205.238.205.3:10476 -> MY.NET.201.204:23 SYN **S*****
09/11-13:19:29 205.238.205.3:10479 -> MY.NET.201.207:23 SYN **S*****
09/11-13:19:29 205.238.205.3:10480 -> MY.NET.201.208:23 SYN **S*****
09/11-13:19:30 205.238.205.3:10611 -> MY.NET.201.218:23 SYN **S*****
09/11-13:19:30 205.238.205.3:10613 -> MY.NET.201.220:23 SYN **S*****
09/11-13:19:30 205.238.205.3:10665 -> MY.NET.201.228:23 SYN **S*****
09/11-13:19:30 205.238.205.3:10667 -> MY.NET.201.230:23 SYN **S*****
09/11-13:19:30 205.238.205.3:10668 -> MY.NET.201.231:23 SYN **S*****
```

Listing #9 – SYNFIN FTP Scan with weird new signature

```
09/11-06:46:35 210.61.144.125:21 -> MY.NET.17.54:21 SYNFIN **SF****
09/11-06:46:35 210.61.144.125:21 -> MY.NET.17.63:21 SYNFIN **SF****
09/11-06:46:35 210.61.144.125:21 -> MY.NET.17.67:21 SYNFIN **SF****
09/11-06:46:35.752887 09/11-06:46:35.752887 [**] SYN-FIN scan! [**] 210.61.144.125:21 -> MY.NET.17.46:21
09/11-06:46:35.752887 [**] SYN-FIN scan! [**] 210.61.144.125:21 -> MY.NET.17.46:21
09/11-06:46:35.753115 [**] SYN-FIN scan! [**] 210.61.144.125:21 -> MY.NET.17.47:21
09/11-06:46:35.753115 [**] SYN-FIN scan! [**] 210.61.144.125:21 -> MY.NET.17.47:21

=====
09/11-06:45:39.310463 210.61.144.125:21 -> MY.NET.5.206:21
TCP TTL:25 TOS:0x0 ID:39426
**SF**** Seq: 0x7BE00D90 Ack: 0x35E5510F Win: 0x404
00 00 00 00 00 00 .....

=====
09/11-06:45:39.439890 210.61.144.125:21 -> MY.NET.5.212:21
TCP TTL:25 TOS:0x0 ID:39426
**SF**** Seq: 0x7BE00D90 Ack: 0x35E5510F Win: 0x404
00 00 00 00 00 00 .....
```

Listing #10 – Various questionable traffic to multiple hosts.

```
08/11-01:51:10.023783 [**] Watchlist 000222 NET-NCFC [**] 159.226.45.108:1051 ->
MY.NET.6.7:23
08/11-01:55:34.538623 [**] Watchlist 000222 NET-NCFC [**] 159.226.45.108:1051 ->
MY.NET.6.7:23
08/11-02:00:13.968903 [**] Watchlist 000222 NET-NCFC [**] 159.226.45.108:1051 ->
MY.NET.6.7:23
08/11-02:11:48.692532 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.200:1841 ->
MY.NET.253.42:25
08/16-02:30:11.291923 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1624 ->
MY.NET.253.43:25
08/16-02:38:35.674741 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1775 ->
MY.NET.253.43:25
08/16-02:38:36.381912 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1775 ->
MY.NET.253.43:25
08/16-02:39:59.218782 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1787 ->
MY.NET.253.42:25
08/16-02:40:03.987900 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1787 ->
MY.NET.253.42:25
```

08/17-05:55:28.725190 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.63.190:1654 ->  
MY.NET.253.42:25  
08/17-05:55:28.725235 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.63.190:1580 ->  
MY.NET.253.43:25  
08/18-18:29:37.695712 [\*\*] Watchlist 000222 NET-NCFC [\*\*]  
159.226.63.190:1380 -> MY.NET.253.43:25  
08/18-18:29:38.351170 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.63.190:1380 ->  
MY.NET.253.43:25  
08/18-18:29:38.354149 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.63.190:1380 ->  
MY.NET.253.43:25  
08/20-15:17:15.981404 [\*\*] Watchlist 000222 NET-NCFC [\*\*]  
159.226.114.129:37268 -> MY.NET.162.199:1097  
08/20-15:17:15.992617 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.114.129:37268 ->  
MY.NET.162.199:1097  
08/20-15:17:16.035221 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.114.129:37268 ->  
MY.NET.162.199:1097  
08/20-15:17:16.539426 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.114.129:37268 ->  
MY.NET.162.199:1097  
08/20-15:17:16.553265 [\*\*] Watchlist 000222 NET-NCFC [\*\*] 159.226.114.129:37268 ->  
MY.NET.162.199:1097

Listing #11 – Employee Use of Napster

08/17-12:45:31.229873 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:4807 ->  
MY.NET.181.87:6699  
08/17-12:45:36.048177 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:4807 ->  
MY.NET.181.87:6699  
08/17-12:45:36.584579 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:4807 ->  
MY.NET.181.87:6699  
08/17-12:45:36.678367 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:4807 ->  
MY.NET.181.87:6699  
08/17-12:45:37.273135 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:4807 ->  
MY.NET.181.87:6699  
09/09-10:46:12.672523 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/09-10:46:12.926072 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/09-10:46:13.354102 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/09-10:46:14.068419 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/09-10:46:16.077532 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/09-10:46:20.788141 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.66.2:22756 ->  
MY.NET.221.94:6699  
09/14-07:41:39.753385 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.58.174:2173 ->  
MY.NET.157.200:6699  
09/14-07:41:39.760943 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.58.174:2173 ->  
MY.NET.157.200:6699  
09/14-07:41:40.640329 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.58.174:2173 ->  
MY.NET.157.200:6699  
09/14-07:41:40.848808 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.58.174:2173 ->  
MY.NET.157.200:6699  
09/14-07:41:40.854809 [\*\*] Watchlist 000220 IL-ISDNNET-990517 [\*\*] 212.179.58.174:2173 ->  
MY.NET.157.200:6699

## Recommendations:

- 1) **Install a properly configured firewall**
- 2) **Implement a secure internal LAN with minimal exposure of external services**
- 3) **An Acceptable Use Policy with regard to employee use of non-work related protocols (Napster and ICQ were prevalent on the network)**
- 4) **Better log correlation.**

A large amount of unsolicited traffic is entering your network, much of which could be blocked with a properly configured firewall. Aside from regular trojan scanning, the network has been the target of several directed network mappings and may be a potential target. I suggest that there is enough illicit traffic entering your network to warrant the creation of a secured internal LAN, which I will gladly do for an extraordinary fee.

Also, there is noted usage of several non-work related protocols such as Napster and AOL ICQ. While not destructive, your company policy may prohibit them.

Finally, your organization has no IDS log correlation system. Although there are Snort IDS sensors in place, the data collected is not properly stored for easy analysis. The data could be concatenated, standardized, or recorded in a database for future reference. (These steps were taken by the Analyst).

© SANS Institute 2000 - 2002, Author retains full rights.

## Assignment 4 : Analysis Process

For this exercise, we were asked to analyze an enormous amount of Snort alert and packet log data. The objective was to assess any serious threats and to correlate as many of the events as possible. While the availability of special tools and procedures can be an enormous advantage to the Analyst, they will not always be present. Thus, the Analyst must sometimes be expected to do their best to make a manual analysis. This was the methodology I chose, thus any tools used are either standard utilities or quickly written perl scripts.

Using the vast amount of data at [www.sans.org/NS2000/snort/index.htm](http://www.sans.org/NS2000/snort/index.htm), we begin by determining how to parse the data. A quick perl script to “standardize” the log formats does the trick. The script below will parse all of the logs (Snort Packet logs, Snort Alert logs, and Snort Portscan logs) and combine all entries into two files, **Aug** and **Sept**, and standardizes the date/time format to “mm/dd-hh:mm:ss”. Thus, the files can now be sorted according to timestamp, allowing for easy correlation between events.

```
#!/usr/bin/perl

%months = ("Aug", "08", "Sep", "09");

$finisheddir = "./parsed";
$dir = ".";

opendir DH, $dir;
@files = readdir DH;
closedir DH;

foreach $month (sort keys %months) {
  open(MONTH, ">>$finisheddir/$month");
  foreach $file (@files) {
    open(LOG, "<$file");
    while ($line = <LOG>) {
      if (($line =~ /$months{s+d{2}}/) || ($line =~ /$months{$month}\d{2}/)) {
        $line =~ s/$month\s+(\d{2})s+/$months{$month}\V$1\-/;
        print MONTH $line;
      }
    }
  }
  close LOG;
  close MONTH;

  system("cat $finisheddir/$month | sort > $finisheddir/$month-sorted");
}
```

Standard shell tools such as **grep** or **sort** are extremely useful for parsing data. By combining ALL data into a single file sorted by timestamp, which we call **Both-sorted**, we can even use regular expressions to gleam whatever data we need. This type of standardization also makes import into a database such as MySQL or Oracle much easier.

For example:

To find all SubSeven traffic using the standard port 27374,  
**“cat Both-sorted | grep ‘27374’”**

To find all SYNFIN scans destined for the FTP port,  
**“cat Both-sorted | grep “:21 SYNFIN”**

To find all network traffic from a particular network or host,  
**“cat Both-sorted | grep “159.22.”**

Now, correlations such as the following can be made:

```
08/20-23:48:21.518799 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1648 -> MY.NET.253.42:25
08/20-23:49:50.132714 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1648 -> MY.NET.253.42:25
08/20-23:49:54.428298 [**] Watchlist 000222 NET-NCFC [**] 159.226.63.190:1648 -> MY.NET.253.42:25
09/02-02:43:22.703281 [**] Watchlist 000222 NET-NCFC [**] 159.226.124.58:2228 -> MY.NET.70.33:8765
09/02-02:43:23.270943 [**] Watchlist 000222 NET-NCFC [**] 159.226.124.58:2229 -> MY.NET.70.33:8765
09/02-02:43:24.214842 [**] Watchlist 000222 NET-NCFC [**] 159.226.124.58:2229 -> MY.NET.70.33:8765
```

The remote network 159.226.0.0/16 was responsible for a large portion of the traffic in the given logs, much of which was suspect. This method of log correlation allows us to see that this particular host has been sending traffic for several days, simply by placing all of the corresponding logs in the correct locality.

By analyzing the data in this manner, I looked for SPECIFIC signatures. If I were truly interested in securing the bid of GIAC Enterprises, what I need to do is to filter out the common, “du jour” events and present them with the serious, somewhat threatening events. Active targeting, exploit attempts, and trojan scans are the “meaty” material that I was looking for.

On the other hand, some organizations are more concerned with INTERNAL traffic, such as the misuse of resources by employees. Napster (Listing #11, above) is a good example of this, and has shown to be a huge tax on network bandwidth. IDS sensors on both sides of the firewall can monitor both malicious traffic as well as prohibited traffic.

© SANS Institute 2000 - 2002  
As part of GIAC practical repository.  
Author retains full rights.



# Upcoming Training

Click Here to  
**{Get CERTIFIED!}**



SANS Security East 2019	New Orleans, LA	Feb 02, 2019 - Feb 09, 2019	Live Event
Security East 2019 - SEC503: Intrusion Detection In-Depth	New Orleans, LA	Feb 04, 2019 - Feb 09, 2019	vLive
SANS Northern VA Spring- Tysons 2019	Tysons, VA	Feb 11, 2019 - Feb 16, 2019	Live Event
SANS London February 2019	London, United Kingdom	Feb 11, 2019 - Feb 16, 2019	Live Event
SANS Scottsdale 2019	Scottsdale, AZ	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS New York Metro Winter 2019	Jersey City, NJ	Feb 18, 2019 - Feb 23, 2019	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201902,	Feb 27, 2019 - Apr 04, 2019	vLive
SANS San Francisco Spring 2019	San Francisco, CA	Mar 11, 2019 - Mar 16, 2019	Live Event
SANS Madrid March 2019	Madrid, Spain	Mar 25, 2019 - Mar 30, 2019	Live Event
SANS 2019	Orlando, FL	Apr 01, 2019 - Apr 08, 2019	Live Event
Blue Team Summit & Training 2019	Louisville, KY	Apr 11, 2019 - Apr 18, 2019	Live Event
SANS Riyadh April 2019	Riyadh, Kingdom Of Saudi Arabia	Apr 13, 2019 - Apr 18, 2019	Live Event
Community SANS New York SEC503	New York, NY	Apr 29, 2019 - May 04, 2019	Community SANS
SANS Security West 2019	San Diego, CA	May 09, 2019 - May 16, 2019	Live Event
SANS Northern VA Spring- Reston 2019	Reston, VA	May 19, 2019 - May 24, 2019	Live Event
SANS Amsterdam May 2019	Amsterdam, Netherlands	May 20, 2019 - May 25, 2019	Live Event
SANS San Antonio 2019	San Antonio, TX	May 28, 2019 - Jun 02, 2019	Live Event
San Antonio 2019 - SEC503: Intrusion Detection In-Depth	San Antonio, TX	May 28, 2019 - Jun 02, 2019	vLive
SANS London June 2019	London, United Kingdom	Jun 03, 2019 - Jun 08, 2019	Live Event
SANSFIRE 2019	Washington, DC	Jun 15, 2019 - Jun 22, 2019	Live Event
Security Operations Summit & Training 2019	New Orleans, LA	Jun 24, 2019 - Jul 01, 2019	Live Event
SANS Paris July 2019	Paris, France	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS Rocky Mountain 2019	Denver, CO	Jul 15, 2019 - Jul 20, 2019	Live Event
SANS Columbia 2019	Columbia, MD	Jul 15, 2019 - Jul 20, 2019	Live Event
SANS Boston Summer 2019	Boston, MA	Jul 29, 2019 - Aug 03, 2019	Live Event
SANS Chicago 2019	Chicago, IL	Aug 19, 2019 - Aug 24, 2019	Live Event
SANS Copenhagen August 2019	Copenhagen, Denmark	Aug 26, 2019 - Aug 31, 2019	Live Event
SANS Network Security 2019	Las Vegas, NV	Sep 09, 2019 - Sep 16, 2019	Live Event
SANS Oslo September 2019	Oslo, Norway	Sep 09, 2019 - Sep 14, 2019	Live Event
SANS London September 2019	London, United Kingdom	Sep 23, 2019 - Sep 28, 2019	Live Event
SANS OnDemand	Online	Anytime	Self Paced