Global Information Assurance Certification Paper

GCIA Certified Intrusion Analyst

Practical Assignment SANS 2001 New Orleans

5 Intrusion Detects

Option 1 – Intrusion Detection Technology
Running Snort on Microsoft NT 2000

Analyze This

Loras R. Even

Assignment 1

Executive Summary:
These scans were collected on my home network, which is connected to the Internet by a cable modem. The cable modem has been installed for approximately 18 months and I have a "test" domain that resolves to a couple of my servers. Normally, I have a Linksys router/firewall in place but I removed it for purposes of this test. (I backed up all files to CD before removing the Linksys box!)

The system used to collect the detects is configured as follows:

- Windows 2000 Professional
- Snort version 1.7-Win32
- Packet Driver v2.02 by Netgroup
- IDS Center 1.08 (Useful Notification Tool)

I used the AllRules.lib rule set that was located on Snort.org. The rule set I used is dated 01/23/2001

DETECT #1:

Two detects were discovered in my alert.ids indicating my system had been scanned by attackers looking for backdoors Subseven and Subseven v2.1. The detects appeared as follows:

```
[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:12.166332 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:27293 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:12.682419 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:33181 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:13.205036 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:42909 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:13.785977 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:56733 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

The following entries were also created in the alert.ids file approximately two hours later:

```
[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:12.867466 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
```

```
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:41031 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:13.462276 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:50759 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:13.969695 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:55623 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:14.468822 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:57927 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

1. Source of Trace:
Home Network.

2. Detect was generated by:
Snort intrusion detection system.

Notes regarding the detects:

1. Source Ports: Snort detects IDS50 and IDS279 do not look for a particular port. The source ports were 4979 for the first attack and 1847 for the second.
2. Destination Ports: IDS50 looks for the destination port of 1243 and IDS279 looks for the destination port of 27374. Both of these port requests can be seen in the alerts and log files produced.
3. Time to Live (TTL): The time to live values (120 & 116) indicate that the originating hosts were most likely hosts running Windows NT 4.0 or newer.
4. Type of Service (TOS), ID, IpLen: I did not notice anything unusual about these values.
5. Don't Fragment (DF): Set to don't fragment.
6. Flags (Syn): As these appear to be service requests, I would expect to see the Syn flag as part of a service set-up request.

3. Probability that the address was spoofed:

I think the likelihood that the source addresses were spoofed is low due to the following:

1. It appears as though the attackers are scanning for hosts that have the backdoor Subseven installed and they will need to get the response back to know when a host responds on port number 27374 or 1243 requests.

2. The TTL seems to be legitimate given that Subseven affects Windows systems and the default TTL from a Windows based attacker would initially be 128. The TTL in the Subseven attack is 120 and the TTL in the Subseven 2.1 attack is 116.

4. Description of Attack

This appears to be a scan for Subseven servers by sending a SYN. If a Subseven server is installed on the destination host it will respond with a normal TCP handshake (SYN, ACK) and then the attackers machine should return an ACK.

Once a server is discovered, Subseven client is used to manipulate the Subseven server. Some of the capabilities of Subseven include searching/retrieving/sending files, stealing passwords, changing the colors/resolution, playing sounds and changing the date/time.

This attack has been issued a candidate for review designation (CAN-1999-0660) and had three votes to accept it.

5. Attack Mechanism

This is a scan to find hosts that appear to be active Subseven hosts. A logical assumption is that the attackers are looking for candidates during these scans and if they see a positive response from a host, they will return to attempt a more thorough exploit.

6. Correlations

The detailed log files for the above alerts are as follows:

```
[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:12.166332 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:27293 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:12.682419 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:33181 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:13.205036 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:42909 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS50 - BACKDOOR ATTEMPT- Subseven [**]
03/05-00:27:13.785977 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.6.26.3:4979 -> 24.6.201.43:1243 TCP TTL:120 TOS:0x0 ID:56733 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1532640  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
```

```
03/05-02:44:12.867466 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:41031 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:13.462276 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:50759 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:13.969695 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:55623 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:14.468822 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:57927 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The first item of information I found of interest is the first octet of the source addresses. I at first thought I was dealing with an attack from one ISP, but reviewing the American Registry for Internet Addresses revealed the following:

Database query for: 24.6.26.3

@Home Network (NETBLK-ATHOME)
450 Broadway Street
Redwood City, CA 94063
US

Netname: ATHOME
Netblock: 24.0.0.0 - 24.23.255.255
Maintainer: HOME

Database query for: 24.24.152.173

ServiceCo LLC - Road Runner (NET-ROAD-RUNNER-1)
13241 Woodland Park Road
Herndon, VA 20171
US

Netname: ROAD-RUNNER-1
Netblock: 24.24.0.0 - 24.31.255.255
Maintainer: SCRR

The Snort rules that were matched by the inbound packets that triggered the alerts were:

Alert ID	Port	Flag
IDS50	1243	SYN
IDS279	27374	SYN

Although not used by Snort to trigger alerts, both scans indicate that the scans use the same sequence number within their scan. Normally, you would expect some variation in the sequence number from normal traffic.

Further analysis of the detailed trace log indicates that both versions of the Subseven Trojan create very similar signatures other than the port differences.

7. Evidence of Active Targeting

If Snort had detected other variants of the IDS50 and IDS279 alert (FTP Activity, etc.) I would have thought this to be more of a focused attack. Since both alerts were created but a scan match in the rule set, I don't believe there is evidence of active targeting and this is only a scan of a wide range of addresses looking for a response.

8. Severity

The standard formula for severity is:
(System Criticality + Attack Lethality) – (System Counter measures + Network Countermeasures)

System Criticality: 1 -- It hit my desktop, which is NT2000
Attack Lethality: 3 – It was a targeted scan as opposed to a general scan
System Counter Measures: 4 – All service packs/Virus Signatures up-to-date.
Network Counter Measures: 0 – No Measures had been taken

(1+3) – (4+0) = 0

9. Defensive Recommendation

Normally, these port scans could have been easily dropped by a good firewall that would drop these packets on the external interface. If the packets are dropped on the external interface, the attacker gathers no information regarding the internally connected hosts. (Assuming it is configured correctly!) The scan makes me feel somewhat justified that normally I have a LYNKSYS cable/dell firewall in place.

There are several low cost SOHO hardware solutions available today for cable users that are considered to be superior to the SOHO software solutions commonly in use today.

In addition to network defenses, virus protection can help to prevent accidental installations of Trojan programs. For example, if you try to install Back Orifice, Subseven, etc. when you have virus protection active you will receive and error.

10. Test Question

The following trace indicates that the attacker is trying to locate a Subseven v 2.1 server. What else in the trace indicates that this could be a "crafted" packet?

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:12.867466 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:41031 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
_
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
03/05-02:44:13.462276 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3E
24.24.152.173:1847 -> 24.6.201.43:27374 TCP TTL:116 TOS:0x0 ID:50759 IpLen:20 DgmLen:48
DF
******S* Seq: 0xDB48A4  Ack: 0x0  Win: 0x2000  TcpLen: 28
```

A. The TTL number is too high
B. The IpLen is invalid
C. The sequence number does not vary
D. The SYN flag is set

DETECT #2:

The following alerts came from an attempt to get into my FTP server. (I added FTP server 3 days earlier to get some more interesting traffic for this project)

```
[**] FTP - INFO - Anonymous FTP [**]
03/08-14:26:12.676883 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x46
4.34.169.203:2985 -> 24.6.201.43:21 TCP TTL:114 TOS:0x0 ID:48843 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0xD5F30BA2  Ack: 0xE225B4  Win: 0x444A  TcpLen: 20

[**] IDS364 - FTP - Bad Login [**]
03/08-14:26:13.023160 0:50:DA:6B:F7:63 -> 0:30:7B:FB:1C:54 type:0x800 len:0x4C
24.6.201.43:21 -> 4.34.169.203:2985 TCP TTL:128 TOS:0x0 ID:9200 IpLen:20 DgmLen:62 DF
***AP*** Seq: 0xE2260A  Ack: 0xD5F30BC8  Win: 0x444B  TcpLen: 20
```

The alerts correlated to entries in the log as follows:

```
[**] FTP - INFO - Anonymous FTP [**]
03/08-14:26:12.676883 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x46
4.34.169.203:2985 -> 24.6.201.43:21 TCP TTL:114 TOS:0x0 ID:48843 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0xD5F30BA2  Ack: 0xE225B4  Win: 0x444A  TcpLen: 20
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS364 - FTP - Bad Login [**]
03/08-14:26:13.023160 0:50:DA:6B:F7:63 -> 0:30:7B:FB:1C:54 type:0x800 len:0x4C
24.6.201.43:21 -> 4.34.169.203:2985 TCP TTL:128 TOS:0x0 ID:9200 IpLen:20 DgmLen:62 DF
***AP*** Seq: 0xE2260A  Ack: 0xD5F30BC8  Win: 0x444B  TcpLen: 20
35 33 30 20 4C 6F 67 69 6E 20 69 6E 63 6F 72 72  530 Login incorr
65 63 74 2E 0D 0A                                ect...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

1. Source of Trace:
Home Network.

2. Detect was generated by:
Snort intrusion detection system.

Notes regarding the detects:

7. Source Ports: These Snort rules do not look at the source port. The original source port from the attacker is at an expected high-level port. And when my server responds, it is at the expected port 21. IDS364 looks for port 21 as the source of my FTP server with payload content of "`530 Login`" which indicates a logon failure.
8. Destination Ports: FTP – Info looks for a destination port (on my server) of 21 with a payload of "anonymous".
9. Time to Live (TTL): The time to live value of 114 indicates that the originating hosts were most likely hosts running Windows NT 4.0 or newer based clients or perhaps VMS or Solaris.
10. Type of Service (TOS), ID, IpLen: I did not notice anything unusual about these values.
11. Don't Fragment (DF): Set to don't fragment.

Flags (ACK,PSH): As these are packets from an already setup conversation, I would expect to see acknowledgement and push set.

3. Probability that the address was spoofed:
I think the likelihood that the source address was spoofed is low. It looks as though an attacker is looking for a host that has anonymous FTP enabled, when they get a result that indicates that anonymous is blocked they leave and continue on to greener pastures. That is why there was only one attempt.

4. Description of Attack

This appears to be part of a search for anonymous FTP servers. It's relatively easy to write a batch or script file that will attempt to log into an FTP server anonymously.

I suspect an automated tool was used since it tried once instead of repeatedly and may be an attackers method of evading detection; instead of port scanning systems for FTP and risking detection, simply attempt to log into FTP hosts anonymously. Many IDS may not log user authentication failure as it is an *application* detect and not a network protocol detect.

I happen to use Typesoft's free FTP server, which by default enabled an Anonymous user. Luckily I disabled the user during the initial configuration, I think I'll leave it disabled.

5. Attack Mechanism

I suspect an automated tool is used for the attempt at logon. While writing this, I received more anonymous logon attempts from various addresses. Unfortunately, I didn't have snort running so I don't have a detect log. I did gather the following information from the FTP server log:

```
[15:22:15] - [0] Connect to 64.70.191.22. Get Username.
[15:22:15] - [0] User ANONYMOUS Connected
[15:22:15] - [0] Client 64.70.191.22 Disconnected (00:00:00 Min)
[05:36:43] - [0] Connect to 199.67.65.41. Get Username.
[05:36:43] - [0] Connect to 199.67.65.41. Get Username.
[05:36:43] - [0] Client 199.67.65.41 Disconnected (00:00:00 Min)
[05:36:43] - [0] Client 199.67.65.41 Disconnected (00:00:00 Min)
[03:12:01] - [0] Connect to 210.112.236.8. Get Username.
[03:12:02] - [0] Client 210.112.236.8 Disconnected (00:00:00 Min)
[04:33:28] - [0] Connect to 210.112.236.8. Get Username.
```

6. Correlations

I do not have anything to correlate this to other than when we setup a test server in our lab at work and had an attempted anonymous FTP logon within 30 minutes of connecting the test system to the Internet, from Germany of all places. The test system was given an address that had never been used for over 2 years! So, seeing activity within days of placing an FTP server on my network should have been no surprise.

The more I think about an application scanner, based on my home network experience and the lab at work, the more I suspect there is a lot of this type activity. Assuming an FTP server is in place, a carefully crafted logon script would not show up in many IDS's or firewall logs. The only errors that might be generated would be failed attempts in the FTP server logs. Even if a detect IS generated in the IDS logs, would the network or security personnel even take note? They may assume that yet another "dumb" user had forgotten their password!

I wish I had time to fire up an SMTP server during this project as I suspect I would have seen some activity on it looking for another SPAM zombie.

7. Evidence of Active Targeting

Based on my discussions above, it's obviously a targeted attack at the application layer (FTP) but I don't think my host address had been targeted as the attacker had never port scanned me.

8. Severity

The standard formula for severity is:
(System Criticality + Attack Lethality) – (System Counter measures + Network Countermeasures)

System Criticality:	1 -- It hit my system (Windows2000)
Attack Lethality:	3 – Attempting user access
System Counter Measures:	4 – All service packs/Virus Signatures up-to-date
Network Counter Measures:	0 – No Measures had been taken

(1+3) – (4+0) = 0

9. Defensive Recommendation

Disable user anonymous! I can't think of a good reason to leave anonymous enabled. Other measures include ensuring that your IDS, firewall and applications server is capable of logging authentication failures and that the logs are looked at and reviewed.

10. Test Question

How do intrusion detection systems like Snort detect invalid user authentication events at the application layers?

A. Read the IP Header Flags to look for an I
B. Decode the payload looking for ASCII matches
C. Check for a DF
D. Application layer checks rely on the application server logs

DETECT #3:

The following detect appeared in my alert.ids file:

```
[**] IIS - Possible Attempt at NT INETINFO.EXE 100% CPU Utilization [**]
03/06-20:24:52.175112 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
209.180.36.97:59557 -> 24.6.201.43:1031 TCP TTL:25 TOS:0x0 ID:3746 IpLen:20 DgmLen:40
******S* Seq: 0x86886095  Ack: 0x0  Win: 0xC00  TcpLen: 20
```

The corresponding log entry:

```
[**] IIS - Possible Attempt at NT INETINFO.EXE 100% CPU Utilization [**]
03/06-20:26:14.221332 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
209.180.36.97:59557 -> 24.6.201.43:1031 TCP TTL:25 TOS:0x0 ID:53909 IpLen:20 DgmLen:40
******S* Seq: 0x86886095  Ack: 0x0  Win: 0xC00  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

1. Source of Trace:
Home Network.

2. Detect was generated by:
Snort intrusion detection system.

The rule that created the alert is as follows:

```
alert tcp any 1024: -> 24.6.201.43/32 1031:1035 (msg:"IIS - Possible Attempt at NT
INETINFO.EXE 100% CPU Utilization"; flags:S;)
```

The rule looks for service requests on TCP ports 1031 – 1035. This alert was triggered due to the service request being directed to port 1031.

Notes regarding the detects:

1. Source Ports: Snort detect IIS does not look for a particular port. In my detect the source port was 59557.
2. Destination Ports: Snort detect IIS looks for the destination ports of 1031 – 1035. A request for 1031 can be seen in the trace file.
3. Time to Live (TTL): The time to live value of 25 indicates the originating host was most likely an older version of Windows(95, NT 3.51). It is also quite possible that this may have been crafted as well.
4. Type of Service (TOS), ID, IpLen: I did not notice anything unusual about these values.
5. Don't Fragment (DF): Not set.
6. Flags (Syn): As these appear to be service requests, I would expect to see the Syn flag as part of a service set-up request.

3. Probability that the address was spoofed:

The likelihood of this being a spoofed source address is low. The SYN flag indicates that this is an attempt to determine if a service is active on port 1031. If the service is active, then an attack can be launched. In order to receive the SYN/ACK, the attacker would need to pass their source address.

4. Description of Attack

This appears to be a probe to determine whether a service is active on TCP port 1031. If a service is discovered, the attacker can attempt to "telnet" to the port. If they are able to enter/pass garbage information the IIS server's utilization peeks to 100%, effectively causing a denial of service.

I could not find a CVE or CAN entry for this specific attack even though many IIS vulnerabilities do have CAN and CVE references.

5. Attack Mechanism

This is a scan to find hosts that appear to respond to TCP port 1031. A logical assumption is that the attackers are looking for candidates during these scans and if they see a positive response from a host, they will return to attempt a more thorough exploit.

6. Correlations

Database query for: 209.180.36.97
Bob XXXX (NETBLK-USW-XXXXX)
XXX XXXXX Avenue
XXXXX, IA XXXXX
US

Netname: USW-XXXXX
Netblock: 209.180.36.96 - 209.180.36.103

I recognized the name of the individual that ARIN indicates is registered to these addresses (We work for the same company). A phone call confirmed what he was attempting.

7. Evidence of Active Targeting
I feel that this was a scan targeting a service (IIS) and a host because I had used IIS to run a small web site from my home in the recent past.

8. Severity
The standard formula for severity is:
(System Criticality + Attack Lethality) – (System Counter measures + Network Countermeasures)

System Criticality:	1 -- It hit my desktop, which is NT2000
Attack Lethality:	4 – It was a targeted scan as opposed to a general scan
System Counter Measures:	4 – All service packs/Virus Signatures up-to-date.
Network Counter Measures:	0 – No Measures had been taken

(1+4) – (4+0) = 0

9. Defensive Recommendation
Normally, these port scans could have been easily dropped by a good firewall that would drop these packets on the external interface. If the packets are dropped on the external interface, the attacker gathers no information regarding the internally connected hosts. (Assuming it is configured correctly!) The scan makes me feel somewhat justified that normally I have a LYNKSYS cable/dsl firewall in place.

There are several low cost SOHO hardware solutions available today for cable users that are considered to be superior to the SOHO software solutions commonly in use today.

10. Test Question

The following trace indicates that the attacker is trying to converse with a Microsoft NT IIS server on port 1031. Is this a request for service of our Home_net server or a response to a request from our server?

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IIS - Possible Attempt at NT INETINFO.EXE 100% CPU Utilization [**]
03/06-20:26:14.221332 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
209.180.36.97:59557 -> Home_net:1031 TCP TTL:25 TOS:0x0 ID:53909 IpLen:20 DgmLen:40
******S* Seq: 0x86886095  Ack: 0x0  Win: 0xC00  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

A. Request
B. Response

DETECT #4:

The following is from my Alert.ids file:

```
[**] IDS05 - SCAN-Possible NMAP Fingerprint attempt [**]
03/06-20:27:57.155738 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x4A
209.180.36.97:59566 -> 24.6.201.43:135 TCP TTL:25 TOS:0x0 ID:23409 IpLen:20 DgmLen:60
**U*P*SF Seq: 0xF5290FFC  Ack: 0x0  Win: 0xC00  TcpLen: 40  UrgPtr: 0x0
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The truncated corresponding entry from the snort_portscan.log file:

```
Mar  6 20:23:25 209.180.36.97:59557 -> 24.6.201.43:2469 SYN ******S*
Mar  6 20:23:25 209.180.36.97:59557 -> 24.6.201.43:3456 SYN ******S*
Mar  6 20:23:25 209.180.36.97:59557 -> 24.6.201.43:3441 SYN ******S*
Mar  6 20:23:25 209.180.36.97:59557 -> 24.6.201.43:4945 SYN ******S*
…
Mar  6 20:27:26 209.180.36.97:59557 -> 24.6.201.43:1982 SYN ******S*
Mar  6 20:27:57 209.180.36.97:59564 -> 24.6.201.43:135 SYN *2****S* RESERVEDBITS
Mar  6 20:27:57 209.180.36.97:59565 -> 24.6.201.43:135 NULL ********
Mar  6 20:27:57 209.180.36.97:59566 -> 24.6.201.43:135 NMAPID **U*P*SF
Mar  6 20:27:57 209.180.36.97:59568 -> 24.6.201.43:2469 SYN ******S*
Mar  6 20:27:57 209.180.36.97:59557 -> 24.6.201.43:2469 UDP
Mar  6 20:28:06 209.180.36.97:59558 -> 24.6.201.43:135 SYN ******S*
Mar  6 20:28:13 209.180.36.97:59561 -> 24.6.201.43:135 SYN ******S*
Mar  6 20:28:17 209.180.36.97:59563 -> 24.6.201.43:135 SYN ******S*
```

1. Source of Trace:
Home Network.

2. Detect was generated by:
Snort intrusion detection system. A portscan alert was generated before the fingerprint alert. I will focus on the fingerprint alert.

The rule that detected this alert is:

```
alert tcp any any -> 24.6.201.43/32 any (msg:"IDS05 - SCAN-Possible NMAP Fingerprint
attempt";flags:SFPU;)
```

Notes regarding the detects:

1. Source Ports: The snort detect doesn't really look for any information on the source port. I think it's interesting, however, to note that the source ports for this trace are very close to sequential; especially when the whole 20 page log is reviewed.
2. Destination Ports: The snort detect doesn't consider the destination ports although a port scan alert was triggered as the 3 ports/3 second threshold had been crossed.
3. Time to Live (TTL): The TTL of 25 indicates that this packet is crafted (As I suspect) or the attacker is using an old version of Microsoft Windows or HP/DEC OS.
4. Type of Service (TOS), ID, IpLen: I did not notice anything unusual about these values.
5. Don't Fragment (DF): Not set.
6. Flags (SYN, FIN, PSH, URG): Snort detect IDS05 looks for the presence of flags SUN, FIN, PUSH and URGENT in the same packet. The SYN FIN alone are

invalid, but the combination of PUSH and URGENT are used to "guess" the operating system based on the response the attacker gets from the "invalid Packet".

3. Probability that the address was spoofed:
I think the likelihood that the source addresses was spoofed is very low due to the following:

1. In order for the attacker to learn information about the host subject to the O.S. guessing, they would have to receive the packets.

4. Description of Attack
This is an attempt to learn the operating system in use on the host under attack. Through unusual flag manipulation, it is possible for NMAP and other tools such as QUESO to guess which operating system they are attacking. They compare the responses received from the host to a signature of expected responses from various systems.

An attacker uses this reconnaissance or foot printing information to focus their attempts to known exploits and hacks that are useful against the identified operating system.

This attack has been issued a candidate for review designation (CAN-1999-0454).

5. Attack Mechanism

This is a scan to identify the operating system through header flag manipulation. By comparing operating system responses to flag manipulations to known signature responses, it is possible to identify an operating system.

6. Correlations

I obtained a copy of NMAP for NT from a site on the Internet. After scanning the files, I installed them and ran them against my test system from a remote system. The responses I received were similar to those above with respect to the flags and creating an alert in Snort.

7. Evidence of Active Targeting

Since NMAP and NMAPNT both support network scanning (more than one host) and the detect snort created covered a wide range of ports, I believe this was not active targeting but more of a reconnaissance effort by the attacker.

8. Severity
The standard formula for severity is:

(System Criticality + Attack Lethality) – (System Counter measures + Network Countermeasures)

System Criticality: 1 -- It hit my desktop, which is NT2000
Attack Lethality: 2 – General scan, Reconnaissance
System Counter Measures: 4 – All service packs/Virus Signatures up-to-date.
Network Counter Measures: 0 – No Measures had been taken

(1+2) – (4+0) = -1

9. Defensive Recommendation

Normally, these port scans could have been easily dropped by a good firewall that would drop these packets on the external interface. If the packets are dropped on the external interface, the attacker gathers no information regarding the internally connected hosts. The scan makes me feel somewhat justified that normally I have a LYNKSYS cable/dsl firewall in place.

There are several low cost SOHO hardware solutions available today for cable users that are considered to be superior to the SOHO software solutions commonly in use today.

In addition to network defenses, virus protection can help to prevent accidental installations of Trojan programs. For example, if you try to install Back Orifice, Subseven, etc. when you have virus protection active you will receive and error.

10. Test Question

Below is a detect from snort that indicates an attacker is attempting to identify your hosts operating system. What item(s) the detect most likely keying on?

```
[**] IDS05 - SCAN-Possible NMAP Fingerprint attempt [**]
03/06-20:27:57.155738 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x4A
209.180.36.97:59566 -> 24.6.201.43:135 TCP TTL:25 TOS:0x0 ID:23409 IpLen:20 DgmLen:60
**U*P*SF Seq: 0xF5290FFC  Ack: 0x0  Win: 0xC00  TcpLen: 40  UrgPtr: 0x0
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

A. TTL = 25
B. The sequence number
C. The flags SYN, FIN, PSH, URG
D. IpLen: 20

DETECT #5:

The following is from my Alert.ids file:

```
[**] IDS80 - BACKDOOR ATTEMPT-Netbus/GabanBus [**]
03/07-02:50:09.906013 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
216.154.19.80:3862 -> 24.6.201.43:12345 TCP TTL:111 TOS:0x0 ID:1685 IpLen:20 DgmLen:44 DF
******S* Seq: 0x35ADE7A6  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460

[**] IDS80 - BACKDOOR ATTEMPT-Netbus/GabanBus [**]
03/07-02:50:10.582367 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
216.154.19.80:3862 -> 24.6.201.43:12345 TCP TTL:111 TOS:0x0 ID:17813 IpLen:20 DgmLen:44
DF
******S* Seq: 0x35ADE7A6  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460

[**] IDS80 - BACKDOOR ATTEMPT-Netbus/GabanBus [**]
03/07-02:50:11.273155 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
216.154.19.80:3862 -> 24.6.201.43:12345 TCP TTL:111 TOS:0x0 ID:32917 IpLen:20 DgmLen:44
DF
******S* Seq: 0x35ADE7A6  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460

[**] IDS80 - BACKDOOR ATTEMPT-Netbus/GabanBus [**]
03/07-02:50:12.009227 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
216.154.19.80:3862 -> 24.6.201.43:12345 TCP TTL:111 TOS:0x0 ID:48533 IpLen:20 DgmLen:44
DF
******S* Seq: 0x35ADE7A6  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
```

1. Source of Trace:
Home Network.

2. Detect was generated by:
Snort intrusion detection system.

```
alert tcp any any -> 24.6.201.43/32 12345 (msg:"IDS80 - BACKDOOR ATTEMPT-
Netbus/GabanBus";flags:S;)
```

Notes regarding the detects:

1. Source Ports: The snort detect doesn't really look for any information on the source port.
2. Destination Ports: This snort detect was triggered due to a match on the destination port (12345).
3. Time to Live (TTL): The TTL of 111 indicates that this may have originated on a host running Windows NT 4.0 or newer.
4. Type of Service (TOS), ID, IpLen: I did not notice anything unusual about these values.
5. Don't Fragment (DF): Set to Don't Fragment.
6. Flags (SYN): The SYN flag is set as this is an initial packet being sent to the targeted host.

3. Probability that the address was spoofed:

The likelihood that the address was spoofed is very low:

1. It appears as though the attacker is scanning for hosts that have the backdoor Netbus/GabanBus installed and they will need to get the response back to know when a host responds on port number 12345 requests.
2. The TTL seems to be legitimate given that Netbus/GabanBus affects Windows systems and the default TTL from a Windows based attacker would initially be 128.

4. Description of Attack

This appears to be a scan for Netbus/GabanBus servers by sending a SYN to a particular destination port. If a Netbus/GabanBus server is installed on the destination host it will respond with a normal TCP handshake (SYN, ACK) and then the attackers machine should return an ACK.

Once a server is discovered, Netbus/GabanBus client is used to manipulate the Netbus/GabanBus.

This attack has been issued a candidate for review designation (CAN-1999-0660).

5. Attack Mechanism

This is a scan to look for services on a particular port. If initial scans detect a response to the port request, follow-ups attempts are usually made to connect to the service.

6. Correlations

The Snort rules that were matched by the inbound packets that triggered the alerts were:

Alert ID	Port	Flag
IDS80	12345	SYN

Although not used by Snort to trigger alerts, both scans indicate that the scans use the same sequence number within their scan. Normally, you would expect some variation in the sequence number from normal traffic.

7. Evidence of Active Targeting

I don't believe there is evidence of active targeting and this is only a scan of a wide range of addresses looking for a response. If the attackers had found a response, we may have seen additional traffic in addition to the detect discovered.

8. Severity

The standard formula for severity is:

(System Criticality + Attack Lethality) – (System Counter measures + Network Countermeasures)

System Criticality: 1 -- It hit my desktop, which is NT2000
Attack Lethality: 3 – General scan, Reconnaissance
System Counter Measures: 4 – All service packs/Virus Signatures up-to-date.
Network Counter Measures: 0 – No Measures had been taken

(1+3) – (4+0) = 0

9. Defensive Recommendation

Normally, these port scans could have been easily dropped by a good firewall that would drop these packets on the external interface. If the packets are dropped on the external interface, the attacker gathers no information regarding the internally connected hosts. The scan makes me feel somewhat justified that normally I have a LYNKSYS cable/dsl firewall in place.

There are several low cost SOHO hardware solutions available today for cable users that are considered to be superior to the SOHO software solutions commonly in use today.

10. Test Question

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] IDS80 - BACKDOOR ATTEMPT-Netbus/GabanBus [**]
03/07-02:50:10.582367 0:30:7B:FB:1C:54 -> 0:50:DA:6B:F7:63 type:0x800 len:0x3C
216.154.19.80:3862 -> 24.6.201.43:12345 TCP TTL:111 TOS:0x0 ID:17813 IpLen:20 DgmLen:44
DF
******S* Seq: 0x35ADE7A6  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

A. Use a proxy server
B. Deny access to port 12345
C. Deny packets with the SYN flag set
D. Use and update virus protection

ANSWERS:

Detect #1: C is correct.
Detect #2: B is correct.
Detect #3: A is correct
Detect #4: C is correct
Detect #5: D is correct

Assignment 2

Executive Summary

As a long time implementer and user of Microsoft products (Starting with NT version 3.0) I've long been frustrated with the wealth of security products available for the Unix operating system as compared those available for Microsoft. The Microsoft compatible solutions line have suffered with bug filled and expensive security solution alternatives to solutions that are relatively inexpensive and work trouble-free on the Unix platform.

As part of my practical, I have had an opportunity to work with the Microsoft NT version of the popular intrusion detection tool called Snort along with many other tools designed to work with Snort.

I will discuss the tools used, review the installation of the tools, discuss configuration considerations and provide a brief opinion regarding my opinions regarding the products capabilities.

Tools Utilized

Snort: Snort's official web site is: http://www.snort.org. This site has links to most tools you'll need to get snort up and running. A brief review of the site confirms that many of snort's tools will run on only Unix, but closer inspection still turns up some excellent tools that you will want to use as part of your snort installation.

The current version is 1.7 as of the time of this report.

The binaries needed to install Snort can be found in the downloads section. The link is: http://download.datanerds.net/binaries/snort-1.7-win32-static.zip

WinPCap: This is the REQUIRED promiscuous driver used to sniff the packets. Installation is easy and it can be found at: http://netgroup-serv.polito.it/winpcap/install/2.1beta/WinPcap.exe.

IDSCenter: IDS Center (Version 1.08) is an excellent tool that provides alert notification among other niceties. Some of the more useful features I found in IDSCenter are:

- Graphical access to most Snort command-line options
- Ability to restart Snort if it dies
- Alerts can be configured to use "Net Send" to send an SMB alerts
- Audio and visual alerts can be configured
- Snort can be configured to start at host startup

- Your configuration can be tested within IDS Center

I also tried Snort Panel which has many features similar to IDSCenter but I preferred some of the features in IDSCenter. You can also find it at: http://www.xato.net/downloads.

Installation

After you have downloaded the files referenced above, installation is pretty straightforward. I'd recommend the following process:

1. Install the WinPcap driver. This is a pretty straightforward step. All you need to do is double click on the WinPcap.exe file you downloaded from the link: http://netgroup-serv.polito.it/winpcap/install/2.1beta/WinPcap.exe. It self installs and asks you if you want to reboot (Required) when installation is complete. There is no additional configuration needed for the driver.

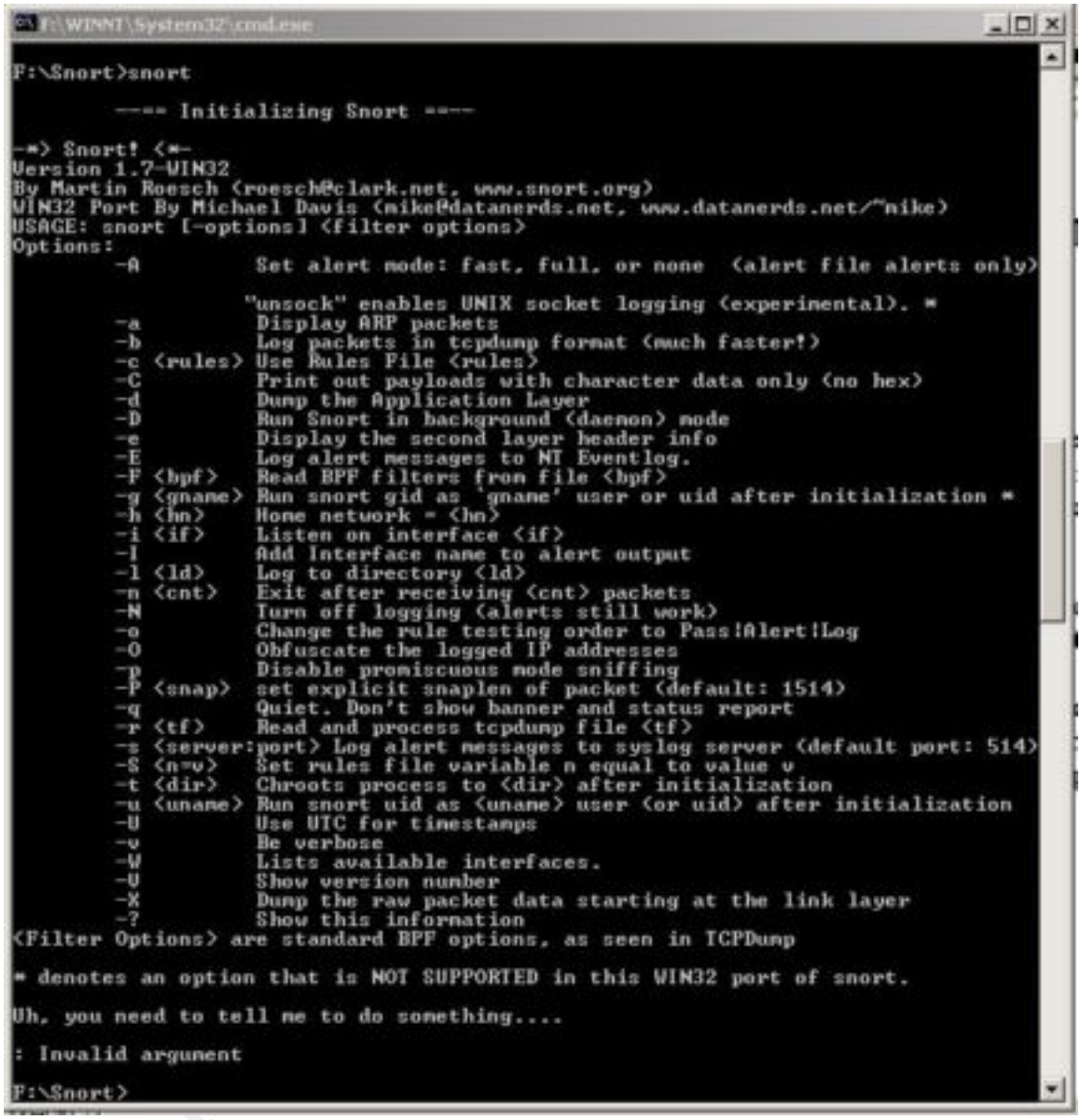
   After you reboot your system, you can verify that the driver was installed by checking the properties of your LAN connection. If the driver is properly installed, your properties should look as follows:

Note: If you need to run the Novell Client for Windows 2000, you may have problems getting the WinPcap driver to work correctly. The version of Netware Client for Windows 2000 I was trying to use is:v4.71.20000312. There is reportedly a new driver available from Novell that may help things. I had to remove the Netware client completely to get Snort/WinPcap to work on my system.

2. Next, you will want to install the Snort application itself. The zip file I downloaded (The standard version, there are additional versions for SQL and FlexRESP) was called snort-1.7-win32-static.zip. You will need an unzip utility to extract the zipped files to a directory that you want snort to be installed into.

   Snort installs itself with many example rules files and other documentation. The Win32 port of snort is different from the Unix version in that there is not a lot of configuration during install of the Win32 port. The snort application requires that the user enter the correct command line options and have a properly configured rules file.

   If you type snort at the command line alone you will get a display like the following:



As you can see, there are MANY command line options! Snort is a very command line oriented which means that the user needs to understand what they want to get out of the tool and how the tool is to report detects before implementing it.

Snort.conf contains most of the configuration parameters specific to your system that are needed by snort to properly process packets as they are captured by the driver. The snort.conf file is very well documented. The only change you will most likely need is to change the "var HOME_NET" to reflect your systems IP address.

If desired, you can update additional information in the snort.conf file to reflect your DNS servers and other information specific to your environment. The

snort.conf includes enough information for you to modify as much or as little as possible. I recommend making as few modifications as possible in the beginning stages of your testing to keep troubleshooting minimal. Once you have had more experience with the product, additional modifications can be made.

Although snort does support SMB messages, (Swatch and some of the other Unix add-on tools have not been ported to Windows as this time) I still desired a little more in the way of "active" alerting. By active alerting, I wanted snort to give me some audible notification that it had found an alert.

In order to get the additional alerting, I tried Snort Panel and IDSCenter.

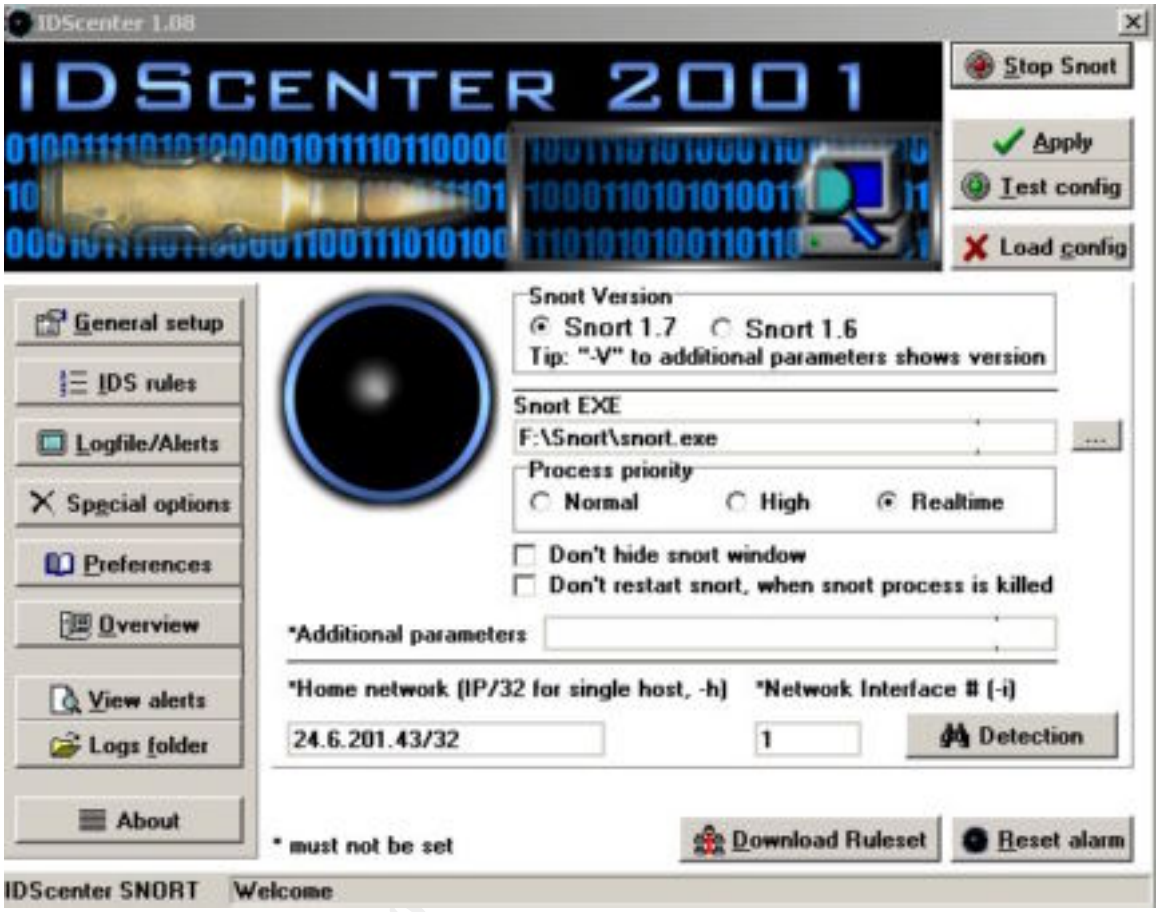
3. I installed both Snort Panel and IDSCenter. I will cover only IDSCenter as I decided to use it and it performed fairly well. The file you download from http://www.snort.org/Files/idscenter.zip is a zip file containing an installation file called setup.exe. Simply extract setup.exe to a temporary directory. Begin the installation by double clicking on setup.exe.

   The installation program prompts you through some fairly simple prompts asking where to install the files and where to put the shortcut in your "Start" menu.

   After installation, you can simply start IDSCenter by selecting Snort IDSCenter 2001 from your "Start" menu.

   When you run IDSCenter, it installs itself in your systems toolbox and looks like a small black dot. If you right click on the dot, you will be prompted with a menu. Since this is the first time you have run IDSCenter and snort, you will want to select settings so you can configure the system.

   The main menu of IDSCenter looks as follows:



The main items to be configured in IDSCenter to get it running are:

a. General Setup Tab
    i. Selecting the version of Snort you are using.
    ii. Enter the path of Snort.exe.
    iii. Select process priority.
    iv. I recommend using the detection button to input your IP address and subnet.
b. IDS Rules Tab
    i. Select the rules file you want to use.
    ii. Enter the name and location of the external editor you want to use.
c. Logfile/Alerts
    i. Select your root Snort log directory.
    ii. Select your alert file type (Full or fast).
    iii. Select the level of protocol analysis (Arp, application, etc.).
    iv. Select "Start Alarm Beep" to get that audible alarm!

There are many other options in IDSCenter you can select to tailor Snort to your systems requirements. I recommend experimenting with each to see which works best for you.

Using Snort (Observations and Tips)

I quickly discovered that snort is a very useful intrusion detection system; it's not only very small and reliable but very extensible and configurable with it's use of rule sets. I found the rule sets so easy to understand that I was able to create my own fairly quickly and easily.

After some experimenting with the default rule sets over a two week period, I can make the following recommendations regarding how you may want to configure snort to help avoid missing information while still not being overwhelmed in false positives.

1. Be careful with the ICMP rules. It seemed as though I was overwhelmed with "ICMP Unknown, unreachable, etc." error messages during my testing. I eventually deleted many of the ICMP rules.
2. Review your alert.ids file often during the first few days to identify potential false positives.
3. Be sure and identify your DNS servers as documented in the snort.conf file. Until I updated my snort.conf file to reflect my DNS servers, I had several "false" UDP port scans in the alerts.ids file. Another approach might have been to increase the thresholds for port scans in the snort.conf file, but this may have made it possible for legitimate ports scans to sneak in below the threshold.
4. A symptom that sometimes occurred with IDSCenter seemed to be a problem with the way it recursively calls the graphic system. If left to run for several days, my display would become corrupt. What I eventually have done is to copy the command-line for snort from IDSCenter and put it in a batch file that is run during boot in a minimized window. This gives me the capability of running snort for extended periods of time with little concern of it's impact on my graphics display.

As much as I like the Microsoft version of snort, there are still some limitations to what you can do with it. We've also installed the Unix variant of snort on one of our FreeBSD systems at work. When using the basic capabilities of snort, they perform essentially the same. The real difference exists in the number of "add-on" utilities that are available for Unix and not for the Microsoft platform.

Even though the Microsoft system port of snort is very useable, if you wish to build a true IDS with capabilities such as Secure Shell management, e-mail notification and others using snort, you will still need to use the Unix version.

Opinion of Snort as an IDS

There are currently over 92 IDS systems available for a variety of platforms. As an IDS, the Win32 version of snort is excellent: it is very extensible with it's rule set support which means that it can be updated quickly when new exploits are released.

When it comes to network traffic analysis the ability of snort to detect crafted packets is limited only by your ability to create (or download) a rule to detect the packet. Luckily, snort was written with a rule interpreter that is very easy to learn. Most users that have written a couple of lines of code or created a script or batch file will be writing their own detects within minutes of reviewing the examples.

I have and use an older version of Data Generals Sniffer, which allows me to capture, edit and replay Ethernet traffic. The sniffer is very handy if you want to test the ability of a system to actually detect "crafted" packets. We found that snort was able to detect any "crafted" packet we could throw at it as long as the rule is correctly defined.

I also monitored resource utilization of the snort application. My system is configured as follows:

- Pentium III 850 MHz
- 256 MB RAM
- 10 MBps Ethernet network

Running at "real-time" prioritization, Task Manager reported less than 1% CPU utilization and less than 500k of memory was in use. You may expect higher utilization in a higher traffic environment, but considering the resource requirements of some of the commercial packages (RealSecure, etc.) compared to those used by snort, the Win32 version still deserves the "light weight" tag

In addition to network traffic analysis, content analysis is also desirable in an IDS. Snort is also able to perform content filtering as evidenced by the number of "anonymous ftp" alerts I received from starting my FTP server on my snort-enabled system. Further inspection of the downloaded rule sets reveals many other content inspection alerts, such as HTTP content. Some of the commercial systems we have considered implementing also perform content filtering in addition to network traffic analysis but for server thousands of dollars!

Alert reporting is another important element in an IDS. The reporting capabilities of snort are somewhat limited and can seem confusing to the average user due to it's design of creating a new directory for every host detect and the technical detail in the alerts.ids file. I did not have time to experiment with the SQL version of snort but I did download and run the tool Win32 version Snort2Html which creates a much friendlier HTML version report of the alerts in the alerts.ids file. Network administrators might find it easier to browse an HTML formatted report than the alerts.ids file itself.

Overall, I'm relatively pleased with the results I've been able to obtain with snort on my Windows 2000 system. As an IDS, snort fulfills many of the basic requirements. Long-term, heavy weight analysis of attacks may require a more through analysis tools such as Shadow, RealSecure or other commercial offerings.

References:

http://www-rnks.informatik.tu-cottbus.de/~sobirey/ids.html
http://www.nswc.navy.mil/ISSEC/CID/step.htm
http://www.snort.org/Files/ws2html.zip
http://www.snort.org
http://packetstorm.securify.com/papers/IDS/lisapaper.txt

Assignment 3

Analyze This

Thank you for the opportunity to bid on your project. You have supplied us with approximately one month's worth of data from a Snort system. The data is assumed to be somewhat incomplete (Power/Disk Full, etc.) as a few days of data are missing but we will work with what you have provided.

You provided us with 45 Snort Alert files. The file creation date on the files was approximately the same, so we interrogated the beginning and end dates within the alert files from which we determined the following correlation between the files and the dates the alerts were created:

Snort File	Begin Date	End Date
A6	11/24/00	11/24/00
A9	11/26/00	11/26/00
A3	11/28/00	11/28/00
A2	11/29/00	11/30/00
A4	12/1/00	12/1/00
A7	12/2/00	12/2/00
A8	12/3/00	12/3/00
A10	12/4/00	12/4/00
A5	12/5/00	12/5/00
A31	12/6/00	12/6/00
A29	12/7/00	12/7/00
A26	12/8/00	12/8/00
A27	12/9/00	12/9/00
A24	12/10/00	12/10/00
A17	12/12/00	12/13/00
A14	12/13/00	12/13/00
A13	12/15/00	12/16/00
A11	12/16/00	12/17/00
A12	12/17/00	12/18/00
A20	12/20/00	12/21/00
A15	12/21/00	12/22/00
A46	12/22/00	12/23/00
A41	12/23/00	12/24/00
A44	12/24/00	12/25/00
A36	12/26/00	12/27/00
A37	12/28/00	12/28/00
A25	12/29/00	12/30/00
A21	12/30/00	12/31/00
A23	12/31/00	1/1/01
A35	1/1/01	1/2/01

A16	1/2/01	1/3/01
A19	1/3/01	1/4/01
A51	1/4/01	1/5/01
A50	1/5/01	1/6/01
A47	1/6/01	1/7/01
A45	1/7/01	1/8/01
A43	1/8/01	1/9/01
A40	1/9/01	1/10/01
A38	1/10/01	1/11/01
A34	1/11/01	1/12/01
A30	1/12/01	1/13/01
A32	1/13/01	1/14/01
A18	1/15/01	1/16/01
A52	1/16/01	1/17/01
A48	1/18/01	1/19/01

You also provided us with 22 OOSche*.txt files which contained what appeared to be Snort log files and 31 Port scan files for further analysis.

We will attempt to provide information regarding:

1. Top Alert Hosts
    a. Destination (Your Network)
    b. Source (Attackers Host(s))
2. Top Port Attacks
3. Reconnaissance activity
4. Suspected Compromised Hosts
5. Watch list Activity
6. Identify any "False" alerts if possible

Methodology

Due to the massive amount of data provided (>50 MB), we spent a little time planning and experimenting with the best methodology of analyzing the data. The only available workstation was a Windows 2000 based system. Most tools designed to automate the analysis process are designed to be run on Unix based workstations.

Our first thought was to use a tool that has been ported to Windows from the Unix world called 'grep". Initial analysis of the alert files indicated a large amount of syn-fin port scans. We decided to pull the scan records out of the alert files so that we would end up with just exploits. We performed this by:

1. We first pulled them out of the snorta*.txt files by grepping(?) the summary and syn-fin records to files called syn-fin.txt and endofportscan.txt.
2. We then did an inverse grep against the snorta*.txt files, creating one large text file with only exploits.

We then used grep in recursive steps to extract data from the large text file created above into individual files containing one exploit. We hoped to end up with files sizes that would allow ASCII import into Microsoft Excel for further manipulation and analysis.

An example of a typical grep command line looked as follows:

```
grep "Back Orifice" snorta*.txt > temp\backorifice.txt
```

After repeated passes (and time) we ended up with the following individual files grouped by detect:

```
03/20/2001  21:11       <DIR>          .
03/20/2001  21:11       <DIR>          ..
03/19/2001  22:08              237,356 Wingate.txt
03/19/2001  22:14              261,913 WatchList222.txt
03/19/2001  22:17              613,319 Tinyfragments.txt
03/19/2001  22:31                  404 wuftpd.txt
03/19/2001  22:01            5,034,997 synfin.txt
03/19/2001  22:09              253,952 SunRPC.txt
03/19/2001  22:49               11,047 SMTPSource.txt
03/19/2001  22:46               16,243 Broadcast.txt
03/19/2001  22:29               17,434 connect515inside.txt
03/19/2001  22:06              478,994 connect515out.txt
03/19/2001  22:53            1,948,548 DNSUDP.txt
03/19/2001  21:58            4,668,736 EndofPortscan.txt
03/19/2001  22:25                6,046 ExternalRPC.txt
03/19/2001  22:47                  961 NMAPFinger.txt
03/19/2001  22:18               54,082 NMAPPing.txt
03/19/2001  22:22               78,806 NullScan.txt
03/19/2001  22:27               74,517 queso.txt
03/19/2001  22:20               22,621 RPCAccess.txt
03/19/2001  22:51               66,612 Russia.txt
03/19/2001  22:23               53,055 SMBName.txt
03/19/2001  22:50                7,748 BackOrifice.txt
03/19/2001  22:30               62,051 SNMPpublic.txt
03/21/2001  14:26           12,939,015 Watchlist220.txt
              23 File(s)     26,908,457 bytes
               2 Dir(s)  27,341,225,984 bytes free
```

Unfortunately, some of the files were still too large to import into Excel and we did not have another similar tool at our disposal.

After some quick research, we found that a free perl interpreter is available from ActiveState Corp. (http://www.activestate.com/) which should allow us to run the perl script tools available for the Unix platforms. We downloaded and installed their perl interpreter version 5.6 build 623. *We were happy to see the installation complete since*

we could never get build 616 to install correctly. We then downloaded SnortSnarf from http://www.snort.org.

Initially, when we unzipped SnortSnarf and tried to run it against the alert files you provided us with we found we had to combine the individual alert files into one, as SnortSnarf would not allow the input of file wildcards. We combined your alert files with the following simple file manipulation:

```
Copy snorta*.txt snortall.txt
```

SnortSnarf then was able to run against the combined file (Thank goodness we have a lot of memory (512MB RAM) and processing capacity, the author warned us in the documentation of processing requirements and he was correct!) but it could not parse the source or destination addresses. We spent a considerable amount of time tracking this down until we ran it against our own alert files and did not experience the same problem.

We discovered the difference between our alert files and yours is that our addresses have not been resolved while yours have been resolved partially (i.e. MY.NET.0.0). We were not sure if this was a "Windows" V.S. Unix issue but we came up with a work around.

It seemed from testing on one snort log that if we substituted MY.NET with a unique octet pair SnortSnarf worked. We grepped the combined file several times looking for an octet pair that was not to be found in the file. We discovered that 123.123 did not match anything in the combined file so we were able to open it with Qedit (An old DOS tool), do a search and replace and then save it as a text file.

We then re-ran SnortSnarf against the combined file and were able to get address resolution! As a final confirmation, we ran grep against our Backorifice.txt detect file to make sure the detects matched what SnortSnarf was reporting:

```
grep -c "[**]" backorifice.txt
                    :77
```

This matched our Backorifice detect in the SnortSnarf html pages so it appeared as though the data had survived the manipulation.

Top Alerts

From our analysis of the alert logs (SnortA*.txt) it was evident that 194,039 alerts occurred while the logs were being created. A breakdown of attacks is as follows:

Signature	# Alerts	# Sources	# Destinations
STATDX UDP attack	1	1	1
SITE EXEC - Possible wu-ftpd exploit - GIAC000623	1	1	1
Happy 99 Virus	1	1	1
site exec - Possible wu-ftpd exploit - GIAC000623	2	2	2
Probable NMAP fingerprint attempt	8	5	6
External RPC call	59	15	25
Back Orifice	77	10	71
TCP SMTP Source Port traffic	100	5	88
Broadcast Ping to subnet 70	154	24	1
connect to 515 from inside	159	10	98
SUNRPC highport access!	204	25	19
SMB Name Wildcard	515	93	171
Russia Dynamo - SANS Flash 28-jul-00	546	2	2
NMAP TCP ping!	558	47	156
SNMP public access	591	20	7
Queso fingerprint	710	52	72
Null scan!	826	527	173
Attempted Sun RPC high port access	2053	16	23
WinGate 1080 Attempt	2239	474	572
Watchlist 000222 NET-NCFC	2401	31	19
connect to 515 from outside	4238	10	2877
Tiny Fragments - Possible Hostile Activity	5340	27	13
DNS udp DoS attack described on unisog	16146	8	6
SYN-FIN scan!	51192	37	27067
Watchlist 000220 IL-ISDNNET-990517	105918	46	100

Total Alerts	**194,039**		

Discussion of Alerts

*** Per Snort rules definition, ALL tests of a rule must be TRUE for an alert to trigger ***
*** We are also assuming that you have not significantly modified the standard rule set***

STATDX UDP attack

The first alert is most a true attempt to determine whether or not this host is vulnerable to this buffer overflow exploit. The rule that triggered this alert is as follows:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"RPC EXPLOIT statdx"; flags: A+; content: "/bin|c74604|/sh";reference:arachnids,442;)

There are several packet tests in the above rule. The most interesting is that to trigger this alert, all flags are set, so we doubt that it is a false alarm.

Source of attack: 206.210.80.6 (North America Registration)
Destination: MY.NET.6.15

SITE EXEC - Possible wu-ftpd exploit - GIAC000623

This could be a compromised system but there was only one detect on inbound traffic so we doubt that the system is compromised at this time. We suggest assessing this host (Virus scanner, etc.) to determine if there is evidence of a compromise. If testing does not indicate a compromise, close monitoring is warranted.

We could not find a standard rule to trigger this alert in our standard rule set although there are several similar wu-ftpd messages.

Source: 209.162.94.11 (North America Registration)
Destination: MY.NET.156.127

Happy 99 Virus

This was most likely an attempt to pass an email to your host with an attached virus. Snort can decode to the application layer through proper rule testing. The rule that caused this alert is found below:

alert tcp any 110 -> any any (msg:"Virus - Possible Happy99 Virus"; content:"X-Spanska\:Yes"; reference:MCAFEE,10144;)

The enclosed alert entry indicates that this packet was intended for an SMTP server although we suspect you may have modified your rule to monitor port 25 instead of our default rules port 110.

12/22-20:25:10.840208 [**] Happy 99 Virus [**] 63.216.198.158:2239-> MY.NET.6.47:25

It is unlikely that your host is compromised, although virus scanning the mail database is warranted.

Source: 63.216.198.158 (North America Registration)

site exec - Possible wu-ftpd exploit - GIAC000623

This could be a compromised system but there was only two detects on inbound traffic so we doubt that the systems are compromised at this time. We suggest assessing these hosts (Virus scanner, etc.) to determine if there is evidence of a compromise. If testing does not indicate a compromise, close monitoring is warranted.

We could not find a standard rule to trigger this alert in our standard rule set although there are several similar wu-ftpd messages.

Source: 24.23.255.246 (North America Registration)
64.217.116.106 (North America Registration)

Probable NMAP fingerprint attempt

These are reconnaissance attempts to identify your hosts operating system so that the attacker can narrow their focus of vulnerability testing to the operating system being used. The attempts came from five different source addresses to six different destination addresses and some can from domestic sources while some came from questionable registries.

Enclosed is the rule that most likely triggered this alert:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN nmap fingerprint attempt";flags:SFPU; reference:arachnids,05;)

As you can see, NMAP manipulates flags to see how the operating system reacts and the rule is testing for the presence of odd flag combinations.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
24.113.198.51	4	MY.NET.105.120	3
130.239.129.109	1	MY.NET.98.154	1
211.109.37.120	1	MY.NET.201.222	1
153.19.144.207	1	MY.NET.209.78	1
206.205.246.2	1	MY.NET.98.147	1

External RPC call

These can be attempts to exploit weaknesses in the RPC mechanism that could allow attackers to change file permissions or execute arbitrary commands. The attempts came from 15 different source addresses to 25 different destination addresses and some can from domestic sources while some came from questionable registries.

It is difficult to know if a host has been compromised. If tripwire or a similar tool has been used to set a baseline, we suggest running tripwire to verify file and system integrity.

We could not find an identical message in our rule set but we suspect that the rule tests for a destination port of 111 and a content test.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
148.228.125.215	13	MY.NET.6.15	26
195.116.66.14	8	MY.NET.15.127	6
206.210.80.6	8	MY.NET.100.130	5
63.11.25.117	7	MY.NET.133.103	1
130.212.20.72	5	MY.NET.133.104	1

Back Orifice

Analysis of the alerts indicates that external parties checking for the existence of a Back Orifice server created all of the messages as all packets were inbound and none of the outbound rules were triggered.

We suggest checking the hosts for listening ports and running a virus scanner on them.

The typical alert message looked as follows:

12/09-22:23:21.058011 [**] Back Orifice [**] 209.94.199.202:31338-> MY.NET.60.7:31337

Source	# Alerts (sig)	Destinations	# Alerts (sig)
209.94.199.202	32	MY.NET.202.94	3
62.136.71.93	20	MY.NET.60.8	2
209.94.199.143	14	MY.NET.60.152	2
216.99.200.242	3	MY.NET.60.22	2
207.253.109.40	2	MY.NET.60.36	2

TCP SMTP Source Port traffic

This alert is most likely caused by the external host trying to either relay mail or test for the capability to relay mail since the source and destination ports were both port 25. 5 External hosts attempted to reach 88 internal hosts Ensure that your hosts do not have relay turned on if not needed.

We did not find a matching message in our alert logs.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
63.11.25.117	84	MY.NET.253.42	11
165.112.79.25	11	MY.NET.199.71	2
213.74.161.214	2	MY.NET.5.27	2
206.132.27.156	2	MY.NET.140.35	1
64.161.240.254	1	MY.NET.140.36	1

Broadcast Ping to subnet 70

There were several sources that attempted to ping the entire 70 subnet range. This was most likely a reconnaissance effort with little risk of there being a compromised system.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
213.154.131.131	52	MY.NET.70.255	154
194.102.93.101	26		
193.231.220.91	17		
193.231.220.137	12		
193.231.220.238	8		

Connect to 515 from inside

Port 515 is a traditional printer or spooler port. This alert is triggered because internal hosts are trying to connect to external devices at port 515. While this could be considered normal traffic, it could also be a tool such as Netcat pushing data back out a "normal" port.

We find it particularly "interesting" that many of the external hosts are located outside North America.

Use a toolkit to test the internal hosts to check for a root kit or existence of a backdoor-like tool.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
MY.NET.70.38	137	216.181.129.185	9
MY.NET.98.151	9	MY.NET.0.114	4
MY.NET.253.12	3	MY.NET.0.22	3
MY.NET.60.38	3	64.23.4.67	3
MY.NET.99.244	2	MY.NET.0.32	3

SUNRPC highport access!/ Attempted Sun RPC high port access

Name	CVE-1999-0189
Description	Solaris rpcbind listens on a high numbered UDP port, which may not be filtered since the standard port number is 111.

This could be an indication of reconnaissance or active targeting. If these are Sun workstations, we suggest ensuring that RPC is filtered to port 111 on the firewall.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
205.188.153.139	91	MY.NET.213.158	104
24.180.202.45	35	MY.NET.99.51	42
64.4.13.74	19	MY.NET.98.199	19
206.196.168.157	7	MY.NET.17.44	6
216.99.200.242	6	MY.NET.202.94	6

SMB Name Wildcard

We could not find an alert trigger in our rule set however this indicates that an external party has tried to attack an SMB vulnerability that exists on many platforms and deal with their inability to sometimes parse and filter wildcards correctly.

The alerts indicate that 93 external sources hit 171 internal addresses.

We think this is a low risk item (Assuming a firewall) but should be monitored for additional traffic.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
141.157.104.204	62	MY.NET.6.15	67
MY.NET.101.160	58	MY.NET.101.192	58
132.239.165.19	23	MY.NET.98.212	23

MY.NET.111.156	17	MY.NET.100.130	10
130.54.113.11	16	MY.NET.50.239	10

Russia Dynamo - SANS Flash 28-jul-00

We think that the internal host is compromised as there are hundreds of packets back and forth between the internal host and the external host. We recommend the host be assessed immediately!

Source	# Alerts (sig)	Destinations	# Alerts (sig)
MY.NET.205.138	442	194.87.6.38	442
194.87.6.38	104	MY.NET.205.138	104

NMAP TCP ping!

This appears to be a very general reconnaissance effort. There are many external source addresses attempting to identify whether a host is live or not by sending an unusual TCP packet. Our rule set identifies that the packet is sent with the Ack flag set.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
MY.NET.70.38	262	MY.NET.1.8	63
192.102.197.234	55	MY.NET.1.3	38
194.133.58.129	46	MY.NET.100.165	27
63.119.91.2	41	MY.NET.253.125	27
64.64.226.2	19	MY.NET.60.14	23

SNMP public access

These alerts appear to be generated by external hosts attempting to connect to internal hosts at port 161 (udp or tcp). Some routers and hosts suffer from denial of service by malformed or unusual fragmentation to their SNMP port.

We would suggest that SNMP access be filtered from the Internet and if possible, eliminate the use of "public" as a community string.

Source(s): Many
Destinations: Many

Queso fingerprint

Like NMAP, Queso is able to fingerprint the version of operating system you are using to further focus the attackers exploits. Due to the large number of sources and destinations, we suspect this is a reconnaissance effort.

WinGate 1080 Attempt

We suspect these are attempts to find a Wingate Proxy server. Wingate by default does not log connections so if an attacker finds a vulnerable proxy server they can use this as a launch-point to attack others.

We could not find the rule in our set but from the data you sent us, this appears to have been triggered by an attempt to connect to port 1080.

Analysis of the OOSch*.txt files indicates that the flags are set to "urgent" on the packets.

If you are running Wingate, please make sure you have upgraded to version 2.1 or better.

Source	# Alerts	Destinations	# Alerts (sig)
209.212.128.47	111	MY.NET.60.8	134
207.114.4.46	91	MY.NET.208.22	110
12.77.204.44	91	MY.NET.60.11	96
204.117.70.5	67	MY.NET.60.38	95
212.72.75.236	65	MY.NET.15.178	75

Watchlist 000222 NET-NCFC and Watchlist 000220 IL-ISDNNET-990517

These alerts were triggered because of traffic from know suspicious sources.

Watchlist 000220 is triggered from addresses in the 212.179.79.0 range which originates in Israel. Watchlist 000222 is triggered from addresses in the 159.226.0.0 range which originates in China.

Traffic generated by these source addresses needs to be watched as it is extremely suspect.

Connect to 515 from outside

Port 515 is a traditional printer or spooler port. This alert is triggered because external hosts are trying to connect to internal devices at port 515. This would be unusual activity as most we assume you don't allow external partners to print to your printers.

In addition, this can be a denial of service attempt if arbitrary codes are passed to the printing service. Cert note: 382365 addresses this issue.

We recommend that you filter access to port 515 from access by external networks.

Primary external access is from the University of Michigan.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
141.211.176.99	2236	MY.NET.100.209	405
216.119.15.88	1273	MY.NET.99.104	403
209.217.166.69	713	MY.NET.130.86	259
192.118.36.9	7	MY.NET.214.166	209
62.46.70.175	4	MY.NET.20.1	7

Tiny Fragments - Possible Hostile Activity

Packets that are too small and are generally considered to be a denial of service attempt generate this alert. Various ports were targeted so this may be an attempt to break your firewall as discussed in Cert vulnerability note 35958.

Suggest that if you are not experiencing performance issues on your firewall, monitor for increased traffic from top 5 sources.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
65.4.87.43	733	MY.NET.1.8	3148
202.205.5.10	521	MY.NET.1.10	1264
202.101.43.222	460	MY.NET.217.162	727
61.134.9.133	458	MY.NET.60.11	168
61.140.75.3	415	MY.NET.1.9	8

DNS udp DoS attack described on unisog

These appear to be concentrated attacks on your dns server as port 53 is targeted. CERT Incident Note IN-2000-04 discusses in detail how these are most likely spoofed source addresses and that the dns server is effectively flooded with spoofed name requests.

Recommended action includes perhaps placing a flood handler on your exterior router, such as implementing Cisco's FW feature set, etc.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
209.67.50.203	16132	MY.NET.1.3	5411
209.67.50.253	4	MY.NET.1.4	5390
209.67.50.85	3	MY.NET.1.5	5331
209.67.50.209	2	MY.NET.1.8	6
209.67.50.241	2	MY.NET.1.10	6

SYN-FIN scan!

These scans are attempts to map your network in a "stealthy" manner. The packets are unusual in that they have both the SYN and FIN flags set. The look for active ports based on the OS response to the unusual flag combination. (Definitely crafted and we suspect NMAP is a potential tool that was used).

The scans were typical in that your whole network experience the scan as the attacker is trying to map your network.

Source	# Alerts (sig)	Destinations	# Alerts (sig)
211.34.40.1	17604	MY.NET.253.112	19
195.56.182.206	9878	MY.NET.21.15	8
194.234.48.26	8565	MY.NET.5.125	7
147.8.182.157	4096	MY.NET.11.212	7
194.204.224.131	3052	MY.NET.232.35	6

Top Alert Hosts

Unfortunately, we could not get the Snort_Stat or Snort_Sort perl scripts to work on the alert files you provided us. All we can assume is that the scripts don't work because we are trying to run them on Windows or because the alert file format is different than expected.

We manually reviewed the information provided by SnortSnarf to generate the top five hosts based on alerts.

External Hosts

Address	Alert #	Ports
212.179.X.X	105,918	Various
211.34.40.1	17,604	Primarily Portscan activity
209.67.50.203	16,132	53
159.226.X.X	2,401	143, 443, +
141.211.176.99	2,236	515

Internal Hosts

Address	Alert #	Ports
MY.NET.201.222	37,609	6688, +
MY.NET.202.94	5,253	31337, 1080, +
MY.NET.201.130	2,047	515, 137, +
MY.NET.217.138	1,447	1080
MY.NET.5.29	1,429	Primarily Portscan Activity

Last Updated: May 26th, 2017

Upcoming Training

Security Operations Center Summit & Training	**Washington, DC**	**Jun 05, 2017 - Jun 12, 2017**	**Live Event**
SANS Houston 2017	**Houston, TX**	**Jun 05, 2017 - Jun 10, 2017**	**Live Event**
SANS Columbia, MD 2017	**Columbia, MD**	**Jun 26, 2017 - Jul 01, 2017**	**Live Event**
SANS London July 2017	**London, United Kingdom**	**Jul 03, 2017 - Jul 08, 2017**	**Live Event**
SANSFIRE 2017	**Washington, DC**	**Jul 22, 2017 - Jul 29, 2017**	**Live Event**
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	**Washington, DC**	**Jul 24, 2017 - Jul 29, 2017**	**vLive**
SANS San Antonio 2017	**San Antonio, TX**	**Aug 06, 2017 - Aug 11, 2017**	**Live Event**
SANS Boston 2017	**Boston, MA**	**Aug 07, 2017 - Aug 12, 2017**	**Live Event**
SANS Adelaide 2017	**Adelaide, Australia**	**Aug 21, 2017 - Aug 26, 2017**	**Live Event**
SANS Virginia Beach 2017	**Virginia Beach, VA**	**Aug 21, 2017 - Sep 01, 2017**	**Live Event**
SANS Network Security 2017	**Las Vegas, NV**	**Sep 10, 2017 - Sep 17, 2017**	**Live Event**
SANS vLive - SEC503: Intrusion Detection In-Depth	**SEC503 - 201709,**	**Sep 11, 2017 - Oct 18, 2017**	**vLive**
Baltimore September 2017 - SEC503: Intrusion Detection In-Depth	**Baltimore, MD**	**Sep 25, 2017 - Sep 30, 2017**	**vLive**
SANS Baltimore Fall 2017	**Baltimore, MD**	**Sep 25, 2017 - Sep 30, 2017**	**Live Event**
SANS London September 2017	**London, United Kingdom**	**Sep 25, 2017 - Sep 30, 2017**	**Live Event**
Community SANS Scottsdale SEC503	**Scottsdale, AZ**	**Oct 02, 2017 - Oct 07, 2017**	**Community SANS**
SANS October Singapore 2017	**Singapore, Singapore**	**Oct 09, 2017 - Oct 28, 2017**	**Live Event**
Community SANS Boston SEC503	**Boston, MA**	**Oct 09, 2017 - Oct 14, 2017**	**Community SANS**
Community SANS Ottawa SEC503	**Ottawa, ON**	**Oct 16, 2017 - Oct 21, 2017**	**Community SANS**
SANS San Diego 2017	**San Diego, CA**	**Oct 30, 2017 - Nov 04, 2017**	**Live Event**
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	**San Diego, CA**	**Oct 30, 2017 - Nov 04, 2017**	**vLive**
SANS Seattle 2017	**Seattle, WA**	**Oct 30, 2017 - Nov 04, 2017**	**Live Event**
SIEM & Tactical Analytics Summit & Training	**Scottsdale, AZ**	**Nov 28, 2017 - Dec 05, 2017**	**Live Event**
SANS Cyber Defense Initiative 2017	**Washington, DC**	**Dec 12, 2017 - Dec 19, 2017**	**Live Event**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s Only**	**Anytime**	**Self Paced**