Global Information Assurance Certification Paper

10 Detects Submitted for GCIA

Hsiao-Yuan Wang
June 10, 2000

Detect 1:

21:24:33.343044 192.168.4.210.2885 > mail.my.net.25: S 271691229:271691229(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 44060
)
21:24:34.927064 192.168.4.210.2888 > mail.my.net.25: S 271691229:271691229(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 44097
)
21:24:35.938229 192.168.4.210.2891 > mail.my.net.25: S 271691229:271691229(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 44137
)
21:24:36.938132 192.168.4.210.2894 > mail.my.net.25: S 271691229:271691229(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 44375
)
21:24:37.940907 192.168.4.210.2897 > mail.my.net.25: S 271691229:271691229(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 44624
)

21:25:33.318610 192.168.4.210.2957 > mail.my.net.25: S 289476050:289476050(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 45264
)
21:25:34.986594 192.168.4.210.2960 > mail.my.net.25: S 289476050:289476050(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 45307
)
21:25:35.947508 192.168.4.210.2963 > mail.my.net.25: S 289476050:289476050(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 45378
)
21:25:36.958746 192.168.4.210.2966 > mail.my.net.25: S 289476050:289476050(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 45563
)
21:25:37.986394 192.168.4.210.2969 > mail.my.net.25: S 289476050:289476050(0)
win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp[|tcp]> (DF) (ttl 51, id 45854
)

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, using –vv options

3. Probability the source address was spoofed
   The IP address 192.168.4.210 is a private IP address, which is not to be appeared on public Internet; therefore, the source IP address is definitely spoofed
   .

4. Description of attack
   An attempted SYN attack, possibly to cause denial-of-service attack against our mail server.

5. Attack mechanism
   The attack apparently originated from an Ethernet network (inferred from the mss option.) The attacker sent 2 bursts of SYN packets to our mail server. Each burst consisted of 5 packets, and all 5 packets shared the same sequence number, with source port incrementing by 3 for the following packets. The listening server sent SYN-ACKs in response to the connections, it then waited for ACKs from the source to arrive to complete 3-way TCP handshake. However, since the source IP is an invalid public IP address, no ACKs ever came back. The result was that those incomplete connections remained in the server's incomplete-connection queue until they were timed out. In doing so, the attacker hoped to fill the server's queue completely; therefore, causing denial-of-service against legitimate connections.

   All packets appear to be crafted by a program.

6. Correlation
   The attack was also logged at another sensor. SYN attacks are also well documented at many web sites, including http://www.cert.org/advisories/CA-96.21.tcp_syn_flooding.html.

7. Evidence of active targeting
   Yes, the attack was logged both at the sensor and the host. It appears that the attacker was actively targeting our mail server.

8. Severity
   Critical: 5—Attempted attack on a core server
   Lethal: 4 – The attack can cause total lockout of the server
   System countermeasure: 5—The server uses a modern operating system with patches installed; in addition, the attack did not knock out the server.
   Net countermeasure: 1 – The gateway router did not block reserved IP addresses—such as the one used in this attack.

   (Critical + lethal) – (system + net countermeasures) = severity
   (5 + 4) – (5 + 1) = 3

9. Defensive recommendation
   Private IP addresses should be blocked at gateway routers.
   Using CISCO router and the above private IP address (192.168.x.x) as an example,

Access-list acl# deny ip 192.168.0.0 0.0.255.255 any log
Apply it at the inbound interface from your ISP.

10 . Multiple choice test question
This is an example of:
A). Attempted SYN attacks against mail server
B). Port scanning
C). Attempted SYN attacks against ftp server
D) The last part of TCP 3-way handshake
Answer: A

Detect 2:

Hostname of 194.217.242.92 is anchor-post-34.mail.demon.net

10:57:34.503708 194.217.242.92.1085 > my.net.110.80: S 797574:799026(**1452**) w
in 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 49, id 7542)
4500 05dc 1d76 4000 3106 b5cd c2d9 f25c
xxxx xxxx 043d 0050 000c 2b86 0000 0000
7002 2000 eb29 0000 0204 0218 0101 0402
6f74 5273 5a53

10:57:34.532273 194.217.242.92.7770 > my.net.110.7770: SP 22334604:22336040(
1436) win 11344 urg **23888** <[**bad opt**]> (DF) (ttl 49, id 7546)
4500 05dc 1d7a 4000 3106 b5c9 c2d9 f25c
xxxx xxxx 1e5a 1e5a 0154 cc8c 1f00 d6ec
bcaa 2c50 49f1 5d50 f562 78e8 3d1b 6fe2
aabb 0d57 5b67

The 13th and 14th byte of the above packet have some of the reserved bits set.

10:57:43.254608 194.217.242.92.1255 > my.net.110.80: P 25435910:25437370(**146
0**) ack 2689617729 win 8760 (DF) (ttl 49, id 12347)
4500 05dc 303b 4000 3106 a308 c2d9 f25c
xxxx xxxx 04e7 0050 0184 1f06 a050 4f41
5018 2238 910a 0000 4745 5420 2f64 6569
6e6c 616b 6169

10:57:43.458552 194.217.242.92.2077 > my.net.110.80: . 5014349:5015809(**1460**)
ack 3566870142 win 8760 (DF) (ttl 49, id 12439)
4500 05dc 3097 4000 3106 a2ac c2d9 f25c
xxxx xxxx 081d 0050 004c 834d d49a 1e7e
5010 2238 80b5 0000 4a32 7376 515a 7345
3561 5477 7750
10:57:45.178315 194.217.242.92.2816 > my.net.110.80: S 534860820:534862272(**1**

452) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 49, id 13375)
4500 05dc 343f 4000 3106 9f04 c2d9 f25c
xxxx xxxx 0b00 0050 1fe1 5414 0000 0000
7002 2000 cf5e 0000 0204 0218 0101 0402
7277 7274 7737

10:57:52.992781 194.217.242.92.3244 > my.net.110.80: S 3785276770:3785278226
(**1456**) win 8192 <mss 1460> (DF) (ttl 49, id 18024)
4500 05dc 4668 4000 3106 8cdb c2d9 f25c
xxxx xxxx 0cac 0050 e19e bd62 0000 0000
6002 2000 3b83 0000 0204 05b4 7772 6478
7a32 6b6c 2f71
10:57:56.702896 194.217.242.92.16314 > my.net.110.80: S 2989244844:298924630
0(**1456**) win 8192 <mss 1460> (DF) (ttl 49, id 19849)
4500 05dc 4d89 4000 3106 85ba c2d9 f25c
xxxx xxxx 3fba 0050 b22c 41ac 0000 0000
6002 2000 0067 0000 0204 05b4 4870 6b31
734c 3234 4554
10:57:57.512281 194.217.242.92.27950 > my.net.110.27960: S 6409912:6411392(**1**
480) win 36612 urg **25606** (DF) (ttl 49, id 20153)
4500 05dc 4eb9 4000 3106 848a c2d9 f25c
xxxx xxxx 6d2e 6d38 0061 ceb8 9c32 0000
05a2 8f04 8796 6406 c23f 718b 9611 ccbd
ce5a 899a 67c0

The 13th byte and the 14th byte in above packet have some suspicious settings, specifically, the TCP header length is 0, which should be at least 5. In addition, some of the reserved bits are set.

10:57:57.768186 194.217.242.92.2995 > my.net.110.80: . 19485443:19486891(**144**
8) ack 1244631975 win 8576 <nop,nop,sack **38335@18991 38479@18991**> (DF) (ttl 49,
id 20253)
4500 05dc 4f1d 4000 3106 8426 c2d9 f25c
xxxx xxxx 0bb3 0050 0129 5303 4a2f 93a7
8010 2180 7002 0000 0101 050a 4a2f 95bf
4a2f 964f 6e52

10:58:01.505205 194.217.242.92.7777 > my.net.110.1037: S 2738850:2740330(**148**
0) win 0 (DF) (ttl 49, id 22078)
4500 05dc 563e 4000 3106 7d05 c2d9 f25c
xxxx xxxx 1e61 040d 0029 caa2 c510 30e0
0602 0000 0000 55ff 1060 1100 e602 c4ff
59ff 03e4 fc49

The 13th byte and the 14th byte in above packet have some suspicious settings, specifically, the TCP header length is 0, which should be at least 5. In addition, some of the reserved bits are set

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, with –vv and –x options

3. Probability the source address was spoofed
   The probability is fairly low. The attacker needed to get responses back.

4. Description of attack
   A) SYN packets with payload
   B) Some packets had "impossible" TCP header length
   C) Some packets had reserved bits set
   D) Abnormal options

5. Attack mechanism
   When the attacker initiated connections at port 80, 1037, 27960, 7770 (why those ports were chosen remains unknown to me), he/she also included data in those packets—possibly with malicious content. When the receiving host received these data, it would include them in the data after the 3-way TCP handshake was completed—therefore help the attacker circumvent some Intrusion Detection Systems that inspect data only after 3-way handshake completes.

   Another reason might be that the attacker wanted to do some OS fingerprinting. Be examining the response that were sent back, the attacker might be able to learn the OS the target host is running. (The host attacked is running Linux 2.2 kernel, and never responded to any of the packets above.)

6. Correlation
   None, I have never seen traffic these in our network. An extensive search over the Internet did not yield any results.

7. Evidence of active targeting
   Yes, the attacker did a scanning of our network about 3 days ago.

8. Severity
   Critical: 4—The attacked host is a core web server
   Lethal: 3 – The attack could gather some useful information about the host's operating system
   System countermeasure: 5—The web server is running the latest operating system, with all available patches installed.
   Net countermeasure: 2 – The firewall did allow these packets to go through.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 3) – (5 + 2) = 0

9. Defensive recommendation
   Add the attacker's IP address to the gateway router's deny list.

10. Multiple choice test question
    The minimum value of TCP header length filed is:
    A) 0
    B) 2
    C) 5
    D) 8

    Answer: C

Detect 3:

Hostname of source IP 209.250.35.192 is ohiper2-192.apex.net

14:10:51.024376 209.250.35.192.3174 > my.net.**216**.27374: S 9822753:9822753(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 52013)
14:10:51.038632 209.250.35.192.3176 > my.net.**218**.27374: S 9822782:9822782(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 52525)
14:10:51.058572 209.250.35.192.3177 > my.net.**219**.27374: S 9822787:9822787(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 52781)
14:10:51.060295 209.250.35.192.3178 > my.net.**220**.27374: S 9822793:9822793(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 53037)

…

14:10:51.296996 209.250.35.192.3190 > my.net.**232**.27374: S 9823055:9823055(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 56877)
14:10:51.305375 209.250.35.192.3191 > my.net**.233**.27374: S 9823060:9823060(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 57133)
14:10:51.306382 my.net.**233**.27374 > 209.250.35.192.3191: R 0:0(0) ack 9823061 win 0 (ttl 30, id 1830)

…

14:10:53.854670 209.250.35.192.3202 > my.net.**254**.27374: S 9823206:9823206(0) win 8192 <mss 536,nop,nop,sackOK> (DF) (ttl 117, id 4142)
14:10:53.854912 my.net.254.27374 > 209.250.35.192.3202: R 0:0(0) ack 1 win 0 (ttl 128, id 60750)

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump

3. Probability the source address was spoofed
   The source IP address does not appear to be spoofed. The attacker obviously needed to get responses back from his/her targets.

4. Description of attack
   This appears to be a scan for SubSeven.

5. Attack mechanism
   The attacker initiated connections to port 27374 to a group of hosts in our network (host my.net.216-my.net.254). By analyzing the responses that were sent by those scanned hosts, the attacker could learn if SubSeven was running at a particular host. For those hosts that didn't exist, no responses were sent to the attacker; for those that did exist, a RESET was sent back to attacker, notifying the attacker that SubSever was not running at that host. However, if a SYN/ACK was sent back, the attacker could learn that SubSeven was indeed running at that host, which might lead to exploitation by the attacker.

   Most of the hosts that the attacker scanned were not existent, for those that did exist, all sent a RESET to the attacker.

6. Correlation
   SubSeven is described at URL, http://www.datafellows.com/v-descs/subseven.htm

7. Evidence of active targeting
   Yes, the attack was logged at both at the sensor and the scanned hosts.

8. Severity
   Critical: 2—The scanned host were either non-existent or user desktops
   Lethal: 4 – The attacker could gain remote control of a host
   System countermeasure: 5—The existing hosts use modern operating systems with patches installed.
   Net countermeasure: 2 –Firewall was too permissive

   (critical + lethal) – (system + net countermeasures) = severity
   (2 + 4) – (5 + 2) = -1

9. Defensive recommendation
   Block port 27374 at the gateway router.

10. Multiple choice test question
    This is a probe of:
    A). Portmapper
    B) SubSeven
    C.) BackOrifice
    D) RingZero

    Answer: B

Detect 4:

Hostname of 199.3.121.201 is Non-Existent.

02:30:25.571585 199.3.121.201 > my.net.110: icmp: echo request (frag 13719:5
52@0+) (ttl 114)
02:30:28.898251 199.3.121.201 > my.net.110: icmp: echo request (frag 31127:5
52@0+) (ttl 114)
02:30:55.567503 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 34685)
02:30:58.887495 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 34860)
02:30:59.195144 199.3.121.201 > my.net.110: icmp: echo request (frag 5528:55
2@0+) (ttl 114)
02:31:02.353880 199.3.121.201 > my.net.110: icmp: echo request (frag 12952:5
52@0+) (ttl 114)
02:31:25.094405 199.3.121.201 > my.net.110: icmp: echo request (frag 39064:5
52@0+) (ttl 114)
02:31:27.896184 199.3.121.201 > my.net.110: icmp: echo request (frag 56216:5
52@0+) (ttl 114)
02:31:29.186362 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 36518)
02:31:32.346269 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 36710)
02:31:35.366798 199.3.121.201 > my.net.110: icmp: echo request (frag 58008:5
52@0+) (ttl 114)
02:31:38.852312 199.3.121.201 > my.net.110: icmp: echo request (frag 62872:5
52@0+) (ttl 114)
02:31:49.254478 199.3.121.201 > my.net.110: icmp: echo request (frag 3737:55
2@0+) (ttl 114)
02:31:52.342467 199.3.121.201 > my.net.110: icmp: echo request (frag 9881:55
2@0+) (ttl 114)
02:31:55.085615 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 38422)
02:31:57.885430 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 38611)
02:32:05.365131 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 39531)
02:32:08.845018 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 39606)
02:32:19.244714 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 39878)
02:32:22.334670 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 39913)
02:32:23.778790 199.3.121.201 > my.net.110: icmp: echo request (frag 47769:5
52@0+) (ttl 114)

02:32:26.990691 199.3.121.201 > my.net.110: icmp: echo request (frag 49561:5
52@0+) (ttl 114)
02:32:53.773515 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 41303)
02:32:56.983423 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 41442)
02:33:41.119683 199.3.121.201 > my.net.110: icmp: echo request (frag 31131:5
52@0+) (ttl 114)
02:33:44.450053 199.3.121.201 > my.net.110: icmp: echo request (frag 34203:5
52@0+) (ttl 114)
02:34:11.110956 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 44997)
02:34:14.441013 my.net.110 > 199.3.121.201: icmp: ip reassembly time exceeded
d [tos 0xc0] (ttl 255, id 45325)
…

32 more echo requests

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, with –vv option

3. Probability the source address was spoofed
   The probability of a spoofed source IP address is fairly high. The attacker may not be interested in getting any responses back.

4. Description of attack
   All echo requests were abnormally large, and were fragmented. The last fragment never arrived.

5. Attack mechanism
   The attacker sent 46 fragmented echo requests to one of our core servers, and in all 46 echo requests, the last fragment was never received, causing our server to send 'ip reassembly time exceeded' error messages. Maybe the attacker wanted to launch a small scale denial-of-service attack by filling the server's buffer space completely.

6. Correlation
   Many fragmented echo requests, similar to the ones shown above, have been detected. However, in all those cases, the last fragment was always received. For example,

   13:11:08.525101 nas-4-122.nyc-t.navipath.net > bay36.qetc.com: icmp: echo request (frag 30220:552@0+)
   13:11:08.535938 nas-4-122.nyc-t.navipath.net > bay36.qetc.com: (frag 30220:156@552)

… 81 more echo requests … (last fragment was received in all echo requests)

7. Evidence of active targeting
   Yes, we have seen a lot of fragmented echo requests from this IP address.

8. Severity
   Critical: 4—The scanned host is a web server
   Lethal: 4 – The attack can possibly cause denial-of-service.
   System countermesaure: 5—The existing hosts use modern opertating systems with patches installed
   Net countermeasure: 2 –Firewall was too permissive, it allowed those packets to go through.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 4) – (5 + 2) = 1

9. Defensive recommendation
   One defensive measure would be to block incoming ping packets.

10. Multiple choice test question
    02:32:26.990691 199.3.121.201 > my.net.110: icmp: echo request (frag 49561:5 52@0+) (ttl 114)
    What does 552:@0+ mean?
        A). The first fragment with a packet size of 552 bytes
        B). The last fragment with a packet size of 552 bytes
        C) The 552nd byte of data of the packet
        D) The value at the 0^{th} byte is 552

    Answer: A

Detect 5:

Hostname of 207.157.100.21 is Non-existent.
Hostname of 205.173.93.34 is mason.ge.com.

my-web-host-access_log:207.157.100.21 - - [29/Mar/2000:17:53:47 -0500]
"GET /cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd HTTP/1.0" 404 282
my-web-host-access_log:205.173.93.34 - - [19/May/2000:04:02:25 -0400]
"GET /cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd HTTP/1.0" 404 282
my-web-host-access_log:205.173.93.34 - - [19/May/2000:04:02:38 -0400]
"GET /cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd HTTP/1.0" 404 282

1. Source of trace

Detected from our network

2. Detect was generated by
   Apache server log

3. Probability the source address was spoofed
   The probability of a spoofed source address is 50-50. The attackers could have used an intermediate site to do the attack—in this case, the source IP could be that of the intermediate site.

4. Description of attack
   The attackers tried to get a password file through a web browser.

5. Attack mechanism
   The way this attack works is as follows: If a web server has an executable file named phf in its cgi-bin directory, then an attacker may be able to use a web browser to manipulate files in the system, including executing commands in the system at the same privileges as the owner of the running httpd. This vulnerability existed before Apache 1.0.5. The attackers in this detect tried to exploit this. However, our web server is not vulnerable to this exploit. The 404 code indicates that our web server responded by "File Not Found" error to the attackers.

6. Correlation
   PHF Cgi-bin attack has been known for quite some time, sites that have information on this kind of attack include http://packetstorm.securify.com/mag/DoJ/DoJ-2/DoJ2-05.txt.

7. Evidence of active targeting
   Yes, the attacks were logged at the web server.

8. Severity
   Critical: 4—The attacked host is one of our core web servers
   Lethal: 3 – The attack could gain all user information on the server if successful, but would not get encrypted passwords (shadow password is used in this host.)
   System countermeasure: 5—The web server did not have the file phf in its cgi-bin directory, and was not vulnerable to this attack.
   Net countermeasure: 2 – The firewall did allow these packets to go through, but it's our company's policy to allow all traffic to this web server.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 3) – (5 + 2) = 0

9. Defensive recommendation
   A). Run the latest version of web server.
   B) Subscribe to the security alert mailing-lists to keep the system updated.

10. Multiple choice test question
    This is an example of:

A). PHF CGI-BIN exploit
B). Easier system administration through a web browser
C). User login through a web browser
D). Testing to verify the web server is functioning

Answer: A

Detect 6:

Hostname of 206.176.81.2 is Non-existent.

```
May 28 09:35:33 zion snort[27540]: spp_portscan:
        PORTSCAN DETECTED from 206.176.81.2
      May 28 09:35:33 206.176.81.2:4074 -> x.y.z.102:110 SYN **S*****
      May 28 09:35:36 206.176.81.2:4072 -> x.y.z.100:110 SYN **S*****
      May 28 09:35:33 206.176.81.2:4077 -> x.y.z.104:110 SYN **S*****
      May 28 09:35:33 206.176.81.2:4086 -> x.y.z.110:110 SYN **S*****
      May 28 09:35:36 zion snort[27540]: spp_portscan: portscan status
        from 206.176.81.2: 9 connections across 9 hosts: TCP(9), UDP(0)
      May 28 09:35:36 206.176.81.2:4074 -> x.y.z.102:110 SYN **S*****
      May 28 09:35:36 206.176.81.2:4077 -> x.y.z.104:110 SYN **S*****
      May 28 09:35:36 206.176.81.2:4086 -> x.y.z.110:110 SYN **S*****
      May 28 09:35:36 206.176.81.2:4111 -> x.y.z.125:110 SYN **S*****
      May 28 09:36:12 zion snort[27540]: spp_portscan: portscan status
        from 206.176.81.2: 8 connections across 8 hosts: TCP(8), UDP(0)
      May 28 09:37:13 zion snort[27540]: spp_portscan: End of portscan
        from 206.176.81.2
```

1. Source of trace
   Detect taken from GIAC website: http://www.sans.org/y2k/060100-1400.htm

2. Detect was generated by
   Snort 1.6

3. Probability the source address was spoofed
   The probability is fairly low as the attacker needed to get responses back from those scanned hosts. However, if the source host was compromised by the attacker prior to the scan, then his/her true origin is questionable.

4. Description of attack
   The attacker might be looking for buffer overflow exploits in POP3.

5. Attack mechanism

The attacker tried to connect to a number of hosts in a network at port 110, hoping to determine if POP3 is accepting connections. For those hosts do respond by sending back SYS/ACK, the attacker might be able to learn the version of the running programs. If that particular version exhibits certain vulnerabilities, the attacker can then exploit them.

The exploit the attacker might be looking for could be the smashing bug found in Qpopper. For versions prior to 2.4.1, an attack against this program could yields a root shell; for versions before 2.53, an attacker who has a valid account may obtain a shell with group ID of 'mail', therefore allowing him/her to access all mails. (Source: http://www.eudora.com/freeware/qpop.html#CURRENT).

6. Correlation
   Exploits about Qpopper can be found at http://www.eudora.com/freeware/qpop.html#BUFFER

7. Evidence of active targeting
   The attacker is definitely targeting the network to find POP3 vulnerabilities.

8. Severity
   Critical: 5—The attacker is scanning multiple hosts in a network for a service that has a long history of security problems.
   Lethal: 5 – The attack can gain root access if he managed to find vulnerabilities
   System countermeasure: I would give it a 5 if the system is running the modern operating system and latest version of POP3, 1 otherwise.
   Net countermeasure: 2 – If the firewall did let these traffic to go through, I would give it a score of 1; however, if it did, then I would give it a score of 5.

   (critical + lethal) – (system + net countermeasures) = severity

   Best case scenario:
   (5 + 5) – (5 + 5) = 0

   Worst case scenario:
   (5 + 5) – (1 + 2) = 7

9. Defensive recommendation

   Block TCP port 110 at the gateway router or allow only authorized hosts to connect to it.

10. Multiple choice test question
    TCP port 110 is:
    A) POP3
    B) POP2
    C) IMAP
    D) DNS

    Answer: A

Detect 7:

Hostname of 165.247.1.145 is user-2ive0ch.dialup.mindspring.net.

```
 [**] SNMP public access [**]
06/04-22:38:50.794940 165.247.1.145:1197 -> my.net.236:161
UDP TTL:117 TOS:0x0 ID:33361
Len: 155
30 81 90 02 01 00 04 06 70 75 62 6C 69 63 A0 81  0.......public..
82 02 01 07 02 01 00 02 01 00 30 77 30 0F 06 0B  ..........0w0...
2B 06 01 04 01 85 01 01 07 01 00 05 00 30 0F 06  +............0..
0B 2B 06 01 04 01 85 01 01 07 04 00 05 00 30 0F  .+............0.
06 0B 2B 06 01 04 01 85 01 01 07 05 00 05 00 30  ..+............0
0F 06 0B 2B 06 01 04 01 85 01 01 06 01 00 05 00  ...+............
30 0F 06 0B 2B 06 01 04 01 85 01 01 01 01 00 05  0...+...........
00 30 0F 06 0B 2B 06 01 04 01 85 01 01 01 0B 00  .0...+..........
05 00 30 0F 06 0B 2B 06 01 04 01 85 01 01 01 0A  ..0...+.........
00 05 00                                         ...

06/04-22:38:52.086832 165.247.1.145:1202 -> my.net.236:161
UDP TTL:117 TOS:0x0 ID:34641
Len: 68
30 3A 02 01 00 04 06 70 75 62 6C 69 63 A0 2D 02  0:.....public.-.
01 0A 02 01 00 02 01 00 30 22 30 0F 06 0B 2B 06  ........0"0...+.
01 04 01 85 01 01 01 0D 00 05 00 30 0F 06 0B 2B  ...........0...+
06 01 04 01 85 01 01 01 0F 00 05 00              ............

06/04-22:38:52.289570 165.247.1.145:1203 -> my.net.236:161
UDP TTL:117 TOS:0x0 ID:34897
Len: 155
30 81 90 02 01 00 04 06 70 75 62 6C 69 63 A0 81  0.......public..
82 02 01 0B 02 01 00 02 01 00 30 77 30 0F 06 0B  ..........0w0...
2B 06 01 04 01 85 01 01 07 01 00 05 00 30 0F 06  +............0..
0B 2B 06 01 04 01 85 01 01 07 04 00 05 00 30 0F  .+............0.
06 0B 2B 06 01 04 01 85 01 01 07 05 00 05 00 30  ..+............0
0F 06 0B 2B 06 01 04 01 85 01 01 06 01 00 05 00  ...+............
30 0F 06 0B 2B 06 01 04 01 85 01 01 01 01 00 05  0...+...........
00 30 0F 06 0B 2B 06 01 04 01 85 01 01 01 0B 00  .0...+..........
05 00 30 0F 06 0B 2B 06 01 04 01 85 01 01 01 0A  ..0...+.........
00 05 00                                         ...


06/04-22:38:52.875276 165.247.1.145:1206 -> my.net.236:161
```

UDP TTL:117 TOS:0x0 ID:35665
Len: 72
30 3E 02 01 00 04 06 70 75 62 6C 69 63 A0 31 02 0>.....public.1.
01 0E 02 01 00 02 01 00 30 26 30 11 06 0D 2B 06 ....0&0...+.
01 04 01 85 01 01 06 02 01 09 01 05 00 30 11 06 ....0..
0D 2B 06 01 04 01 85 01 01 06 02 01 0A 01 05 00 .+..............
.

1. Source of trace
   Detected from our network

2. Detect was generated by
   Snort 1.6

3. Probability the source address was spoofed
   The probability of a spoofed source IP address is fairly low, as the attacker's probably interested in getting reponses back.

4. Description of attack
   Illegal access to a router using the default SNMP community name.

5. Attack mechanism
   The attacker tried to connect to one of our internal routers, using the default SNMP community name "public", probably in hope to obtain system information, such as routing tables. If successful, the attacker could monitor, modify, or even take down our network completely. Fortunately, all of the attacker's connections were blocked at the gateway router, i.e, they did not reach the intended router. In addition, the SNMP community name in our network is not "public."

6. Correlation
   This kind of attack is also mentioned at GIAC-San Jose Conference by Stephen Northcutt, SANS Institute. (*2.4 Network –Based Intrusion Detection Analysis, Stephen Northcutt, The SANS Institute, pp 169)*

7. Evidence of active targeting
   Yes, the unauthorized access was logged at the the sensor that sits in front of the gateway router.

8. Severity
   Critical: 5—The scanned host is a core router
   Lethal: 5 – the attack could gain control of our whole network
   System countermeasure: 5—The router is using the latest IOS version.
   Net countermeasure: 5 –The firewall did not permit packets to enter our network.

   (critical + lethal) – (system + net countermeasures) = severity
   (5 + 5) – (5 + 5) = 0

9. Defensive recommendation

1.) Protect your SNMP enabled hosts by firewalling those hosts from unauthorized hosts. This includes blocking SNMP access at your gateway router (SNMP uses UDP port 161 and 162).
2.) Change the community name to something that is very difficult to guess.

10. Multiple choice test question
    What is the default SNMP community name:
    A. private
    B. public
    C. snmp
    D. none

Detect 8:

Hostname of 207.55.175.8 is Non-existent.

```
22:37:48.941038 207.55.175.8.1081 > my.net.34.111: S 2120750621:2120750621(0
) win 16060 <mss 1460,sackOK,timestamp 260354414 0,nop,wscale 0> (DF) (ttl 51, i
d 15177)
22:37:48.943510 207.55.175.8.1080 > my.net.33.111: S 2112952250:2112952250(0
) win 16060 <mss 1460,sackOK,timestamp 260354414 0,nop,wscale 0> (DF) (ttl 51, i
d 15176)
22:37:48.953855 207.55.175.8.1082 > my.net.35.111: S 2114716325:2114716325(0
) win 16060 <mss 1460,sackOK,timestamp 260354414 0,nop,wscale 0> (DF) (ttl 51, i
d 15178)
22:37:48.970114 207.55.175.8.1083 > my.net.36.111: S 2109026901:2109026901(0
) win 16060 <mss 1460,sackOK,timestamp 260354416 0,nop,wscale 0> (DF) (ttl 51, i
d 15180)
22:37:49.030203 207.55.175.8.1085 > my.net.38.111: S 2116890900:2116890900(0
) win 16060 <mss 1460,sackOK,timestamp 260354422 0,nop,wscale 0> (DF) (ttl 51, i
d 15193)
22:37:49.051377 207.55.175.8.1086 > my.net.39.111: S 2110357015:2110357015(0
) win 16060 <mss 1460,sackOK,timestamp 260354424 0,nop,wscale 0> (DF) (ttl 51, i
d 15198)
22:37:49.090861 207.55.175.8.1087 > my.net.40.111: S 2125071262:2125071262(0
) win 16060 <mss 1460,sackOK,timestamp 260354428 0,nop,wscale 0> (DF) (ttl 51, i
d 15209)
```

...

```
22:37:53.811897 207.55.175.8.1281 > my.net.234.111: S 2113301603:2113301603(
0) win 16060 <mss 1460,sackOK,timestamp 260354900 0,nop,wscale 0> (DF) (ttl 51,
id 15650)
```

...

22:37:53.890893 207.55.175.8.1297 > my.net.250.111: S 2121636172:2121636172(0) win 16060 <mss 1460,sackOK,timestamp 260354900 0,nop,wscale 0> (DF) (ttl 51, id 15666)

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, with –vv option

3. Probability the source address was spoofed
   The probability of a spoofed source IP is fairly low, unless the machine the did the scanning was compromised by the attacker, in which case, the true origin of the attack may not be known without log files from that host.

4. Description of attack
   The attacker was probably looking for exploits in RPC programs.

5. Attack mechanism
   Since various RPC services register their port information with Portmapper, the attacker tried to determine if he/she could connect to Portmapper by sending SYNs to TCP port 111 of many hosts in our network (host 31 to 254). If the attacker received SYN/ACKs back from a host, then he/she would be able to launch attack against any services that exhibit vulnerabilities.

6. Correlation
   Attacks against various RPC services are documented at many sites. The following sites document some of the RPC services vulnerabilities:

   rpc.statd Vulnerability
   http://ciac.llnl.gov/ciac/bulletins/g-22.shtml

   rpc.pcnfsd Vulnerability
   http://www.stanford.edu/~security/Advisories/99-0908.html

   rpc.ypupdated Vulnerability
   http://www.stanford.edu/~security/Advisories/96-067.html

   more vulnerabilities listed at the following sites:
   http://www.cert.org/current/current_activity.html#scans
   http://www.sans.org/y2k/trouble_RPCs.htm

7. Evidence of active targeting

The attack is logged at our external sensor, and can be viewed as evidence of searching for RPC exploits against our network.

8. Severity
   Critical: 4—Many of the attacked hosts are servers.
   Lethal: 5 – The attack can gain root access if he/she is allowed to connect to portmapper and any of the registered RPC services exhibits vulnerabilities.
   System countermeasure: 5—Hosts that have portmapper port open are all running the most recent version of operating system, and all registered RPCs are well-patched.
   Net countermeasure: 5 – The firewall blocked all inbound connections to port 111.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 5) – (5 + 5) = -1

9. Defensive recommendation
   Block port 111 (both TCP and UDP) at the gateway router, or at the very least, allow accesses by those authorized hosts only.

10. Multiple choice test question
    Which program do most RPC services register to?
    A) mountd
    B) portmapper
    C) in.rpcd
    D) nfsd

    Answer: B

Detect 9:

Hostname of 168.191.72.23 is sdn-ar-001tnmempP015.dialsprint.net.

```
4:03:54.142465 168.191.72.23 > my.net.30: icmp: echo request (ttl 50, id 22
184)
04:03:54.156193 168.191.72.23 > my.net.31: icmp: echo request (ttl 50, id 22
440)
04:03:54.158903 168.191.72.23 > my.net.32: icmp: echo request (ttl 50, id 22
696)
04:03:54.198560 168.191.72.23 > my.net.33: icmp: echo request (ttl 50, id 22
952)
04:03:54.200778 168.191.72.23 > my.net.34: icmp: echo request (ttl 50, id 23
208)
04:03:54.203486 168.191.72.23 > my.net.35: icmp: echo request (ttl 50, id 23
464)
04:03:54.298069 168.191.72.23 > my.net.38: icmp: echo request (ttl 50, id 24
232)
```

```
04:03:54.325655 168.191.72.23 > my.net.39: icmp: echo request (ttl 50, id 24
488)
04:03:54.330828 168.191.72.23 > my.net.40: icmp: echo request (ttl 50, id 24
744)
04:03:54.607188 168.191.72.23 > my.net.67: icmp: echo request (ttl 50, id 31
912)

04:03:55.773220 168.191.72.23.1336 > my.net.30.8080: S 11853214:11853214(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 44968)
04:03:55.776432 168.191.72.23.1337 > my.net.31.8080: S 11853247:11853247(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 45224)
04:03:55.820551 168.191.72.23.1338 > my.net.32.8080: S 11853280:11853280(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 45736)
04:03:55.866090 168.191.72.23.1339 > my.net.33.8080: S 11853318:11853318(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 46248)
04:03:55.881844 168.191.72.23.1340 > my.net.34.8080: S 11853350:11853350(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 46504)
04:03:55.924212 168.191.72.23.1341 > my.net.35.8080: S 11853382:11853382(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 46760)
04:03:55.929136 168.191.72.23.1342 > my.net.38.8080: S 11853413:11853413(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 47016)
04:03:55.968130 168.191.72.23.1343 > my.net.39.8080: S 11853444:11853444(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 47272)
04:03:55.972550 168.191.72.23.1344 > my.net.40.8080: S 11853483:11853483(0)
win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 47528)
04:03:56.093182 168.191.72.23.1345 > my.net.67.8080: S 11853517:11853517(0)
                                    ...
04:04:02.231730 168.191.72.23.1361 > my.net.246.8080: S 11857650:11857650(0)
 win 6096 <mss 536,nop,nop,sackOK> (ttl 50, id 53930)
```

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, with –vv option

3. Probability the source address was spoofed
   The source IP does not appear to be spoofed. The attacker should be interested in getting results back. The above statements assume that the source machine was not compromised.

4. Description of attack
   The attacker scanned our network for http proxy service.

5. Attack mechanism

The attacker first sent echo requests to hosts in our network. Once determined that a particular host existed, the attacker attempted scanning of that host for http proxy service. If successful, the attacker may be able to attack other sites via proxy service without revealing his/her true origin--if the proxy server does not forward attacker's IP address to the site that he/she is trying to connect to.

However, all his/her attempted connections were dropped at our gateway router. In addition, no hosts were running http proxy in our network.

6. Correlation
   This is the first attempted connection from this source IP address. However, similar scans can be found at http://www.sans.org/y2k/022900.htm.

7. Evidence of active targeting
   Yes, the attack was logged at our external sensor and our gateway router.

8. Severity
   Critical: 4—The attacked hosts are either web servers or mail servers
   Lethal: 4 – The attack could launch net-based attackers against other networks.
   System countermeasure: 5—All hosts are running the latest operating system with all available patches installed. In addition, no proxy service was running in any of the scanned hosts.
   Net countermeasure: 3 – The firewall did allow these packets to go through.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 4) – (5 + 3) = 0

9. Defensive recommendation
   If you do not run proxy service, block all traffic to port 8080. If you do, at least log all the connections to your proxy service, or use a proxy server that forwards a user's real IP address to the site he/she wants to connect to.

10. Multiple choice test question
    A) http proxy service scanning
    B) ftp scanning
    C) RPC services scanning
    D) POP2 scanning

    Answer: A

Detect 10:

Hostname of 161.142.104.194 is j60.kgr2.jaring.my.

Pattern 1:

07:30:30.287438 161.142.104.194.**42777** > my.net.30.80: **R** 23670022:23670022(0) win 0 (ttl **111,** id 18147)

4500 0028 46e3 0000 6f06 3f4a a18e 68c2
xxxx xxxx a719 0050 0169 2d06 2f91 47f9
5004 0000 9cda 0000 0101 080a 057c

07:30:31.242699 161.142.104.194.**42777** > my.net.30.80: **R** 0:1460(1460) ack 1 win 8760 (ttl **40**, id 28407)

4500 05dc 6ef7 0000 2806 5882 a18e 68c2
xxxx xxxx a719 0050 0169 2d06 68d4 5cf7
5014 2238 5253 0000 8492 4489 ba6e 5b56
c1b2 4919 95d6 00d2 3fea 3297 7764 d015
02d3 18c9 eacf 3f99 0fcc 94f9 0bf0 5fe5
0752 c3b8 60af c866 7ac6 b23d bf45 555d
0538 c9e5 b1d5 f8b9 29e8 e07a a9d2 04aa
a8a5 8656 4566 504a e906 e403 963f 7b6f
69ae 761e 58f6 9f9a

07:31:04.607305 161.142.104.194.**42876** > my.net.30.80: **R** 23781666:23781666(0) win 0 (ttl **111**, id 26852)

4500 0028 68e4 0000 6f06 1d49 a18e 68c2
xxxx xxxx a77c 0050 016a e122 2f59 e2a9
5004 0000 4de1 0000 0869 6672 6565

07:31:04.616918 161.142.104.194.**42876** > my.net.30.80: **FR** 0:0(0) ack 0 win 8760 (DF) (ttl **40**, id 60500)

4500 0028 ec54 4000 2806 a0d8 a18e 68c2
xxxx xxxx a77c 0050 016a e122 82b9 ebab
5015 2238 cf35 0000 0869 6672 6565

07:31:33.977088 161.142.104.194.42777 > my.net.30.80: **R** 23670022:23670022(0) win 0 (ttl **111**, id 34789)
07:31:34.932032 161.142.104.194.42777 > my.net.30.80: **R** 0:1460(1460) ack 1 win 8760 (ttl **40**, id 33490)
07:32:06.406527 161.142.104.194.42876 > my.net.30.80: **R** 23781666:23781666(0) win 0 (ttl **111**, id 21478)
07:32:06.426484 161.142.104.194.42876 > my.net.30.80: **FR** 0:0(0) ack 0 win 8760 (ttl **40**, id 3364)

07:32:43.055782 161.142.104.194.42777 > my.net.30.80: **R** 23670022:23670022(0) win 0 (ttl **111**, id 13287)
07:32:44.027941 161.142.104.194.42777 > my.net.30.80: **R** 0:1460(1460) ack 1 win 8760 (ttl **40**, id 45222)
07:33:10.107260 161.142.104.194.42876 > my.net.30.80: **R** 23781666:23781666(0) win 0 (ttl **111**, id 53991)

07:33:10.166145 161.142.104.194.42876 > my.net.30.80: **FR** 0:0(0) ack 0 win 87
60 (ttl **40**, id 17080)

… The above pattern continues for 2 more minutes …

07:36:22.654133 161.142.104.194.**42876** > my.net.30.80: **R** 23781666:23781666(0)
win 0 (ttl **111**, id 38636)
4500 0028 96ec 0000 6f06 ef40 a18e 68c2
xxxx xxxx a77c 0050 016a e122 e996 4481
5004 0000 31cc 0000 0101 080a 1d3f

07:36:22.728521 161.142.104.194.**42876** > my.net.30.80: **FR** 0:0(0) ack 0 win 87
60 (ttl **40**, id 2938)
4500 0028 0b7a 0000 2806 c1b3 a18e 68c2
xxxx xxxx a77c 0050 016a e122 82b9 ebab
5015 2238 cf35 0000 0101 080a 057d

07:37:27.214920 161.142.104.194.**42876** > my.net.30.80: **R** 23781666:23781666(0)
win 0 (ttl **111**, id 30701)
07:37:27.224814 161.142.104.194.**42876** > my.net.30.80: **FR** 0:0(0) ack 0 win 87
60 (ttl **40**, id 16139)

…The above pattern continues for a little more than 2 minutes

1. Source of trace
   Detected from our network

2. Detect was generated by
   Tcpdump, with –vv and –x option

3. Probability the source address was spoofed
   The probability of a spoofed source IP is low. The attacker should be interested in getting responses back.

4. Description of attack
   Possibly OS fingerprinting

5. Attack mechanism
   All packets formed nice patterns.

   The first pattern:
   1.) 3 RESETs followed by a FRA
   2.) Alternating TTL values (111 and 40)
   3.) Alternating RESET sequence number
   4.) Alternating source ports every other two packets (42777 and 42876)

The second pattern:
1) Source port stays the same (42876)
2) Alternating RESET and FRA packets
3) Alternating TTL values

All of the above may well indicate that those packets were crafted by a program. The purpose might be to fingerprinting OS running at the target host.

6. Correlation
   None, a search over the Internet did not yield any result.

7. Evidence of active targeting
   Yes, the attacker did a scanning of our whole network in the previous day.

8. Severity
   Critical: 4—The attacked host is a core host in our network.
   Lethal: 3 – The attack can collect some useful information about the host, such as the operating system running at the target host.
   System countermeasure: 5—The attacked host is running the latest version of operating system with all patched installed.
   Net countermeasure: 4 – The firewall did allow these packets to go through.

   (critical + lethal) – (system + net countermeasures) = severity
   (4 + 3) – (5 + 4) = -2

9. Defensive recommendation
   Add the attacker's IP address to the router deny list.

10. Multiple choice test question

    4500 0028 0b7a 0000 2806 c1b3 a18e 68c2
    xxxx xxxx a77c 0050 016a e122 82b9 ebab
    5015 2238 cf35 0000 0101 080a 057d

    A) This is a TCP packet
    B) This is a UDP packet
    C) This is an ICMP packet
    D). This is an IGMP packet

    Answer: A