



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 permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Intrusion Detection In-Depth (Security 503)"
at <http://www.giac.org/registration/gcia>

SANS 2000 San Jose

GIAC Intrusion Detection Practical Assignment

Michael Wee

Detect 1:

Jun 8 21:10:44 bali portsentry[437]: attackalert: SYN/Normal scan from host: c604394-a.altn1.tx.home.com/24.14.93.7 to TCP port: 1080

Jun 8 21:10:45 bali kernel: Packet log: input DENY eth0 PROTO=6 24.14.93.7:2529 24.92.71.224:1080 L=48 S=0x00 I=3040 F=0x4000 T=113 SYN (#1)

1	Source of trace: Linux host residing on cable network
2	Detect was generated by: a) Portsentry from Psionic software Jun 8 21:10:44 {date/time} bali {hostname} portsentry[437] {service:PID} : attackalert: SYN/Normal scan from host {portsentry scan warning}: c604394- a.altn1.tx.home.com/24.14.93.7 {source address} to TCP port: 1080 {target port & protocol} b) Linux IPChains firewall Jun 8 21:10:45 {date/time} bali {hostname} kernel: Packet log: input {ipchains name} DENY {action} eth0 {interface} PROTO=6 {protocol} 24.14.93.7:2529 {source address and port} 24.92.71.224:1080 {destination address and port}L=48 {packet length (bytes)} S=0x00 {TOS} I=3040 {IP ID} F=0x4000 {16 bit fragment offset} T=113 {TTL} SYN {TCP flag} (#1) {ipchains rule number}
3	Probability the source address was spoofed: Low probability of a spoofed source address. There is not enough evidence in the packet log to suggest packet header manipulation. The address space belongs to @home cable network provider.
4	Description of attack: This attack is targeted at exploiting the existence of any SOCKS proxy service (e.g. Wingate).
5	Attack mechanism: The intruder was attempting to exploit any listening SOCKS server that was *improperly *configured to allow connections from its external network. (Many home users use Wingate or similar SOCKS server to proxy Internet connections to their home LAN.) Once the intruder has successfully found such a system, they might proceed to launch an attack on the internal hosts (or the proxy/gateway hosts itself), or attack other Internet hosts via the SOCKS server, thereby masking their

	<p>true source address. These probes might also be prevalent from IRC servers that are checking for the existence of “anonymous” IRC users who might be proxying behind another host.</p>
6	Correlations:
	<p>Socks scans were discussed in Stephen Northcutt’s SANS class (Network based intrusion detection analysis). CVE-1999-0290 - The WinGate telnet proxy allows remote attackers to cause a denial of service via a large number of connections to localhost. CVE-1999-0291 - The WinGate proxy is installed without a password, which allows remote attackers to redirect connections without authentication. CVE-1999-0494 - Denial of service in WinGate proxy through a buffer overflow in POP3.</p>
7	Evidence of active targeting:
	<p>This attack was logged on a specific host. It is most likely that this attack is just one in a series of SOCKS probe of the RR’s cable LAN.</p>
8	Severity:
	<p>(Critical + Lethal) – (System + Network countermeasures) = Severity $(2 + 1) - (5 + 4) = -6$</p>
9	Defensive recommendations:
	<p>The current defenses are sufficient. There is no SOCKS server to exploit, and host access from the intruder has been blocked.</p>
10	Multiple choice test question based on trace and analysis:
	<p>Some IRC servers use the following service to determine source origination of chat users:</p> <ul style="list-style-type: none"> a) icmp b) socks c) ftp d) http <p>Answer: b</p>

Detect 2:

```

Jun  4 05:05:13 bali portsentry[439]: attackalert: UDP scan from host: cx348938-
b.msnv1.occa.home.com/24.1.143.225 to UDP port: 137
Jun  4 05:05:14 bali kernel: Packet log: input DENY eth0 PROTO=17 24.1.143.225:137
24.92.71.224:137 L=78 S=0x00 I=37154 F=0x0000 T=111 (#1)
Jun  4 05:05:15 bali kernel: Packet log: input DENY eth0 PROTO=17 24.1.143.225:137
24.92.71.224:137 L=78 S=0x00 I=37410 F=0x0000 T=111 (#1)
  
```

1	Source of trace:
	Linux host residing on cable network

2	Detect was generated by:
	<p>a) Portsentry from Psionic software Jun 8 21:10:44 {date/time} bali {hostname} portsentry[437] {service:PID} : attackalert: SYN/Normal scan from host {portsentry scan warning}: c604394- a.altn1.tx.home.com/24.14.93.7 {source address} to TCP port: 1080 {target port & protocol}</p> <p>b) Linux IPChains firewall Jun 8 21:10:45 {date/time} bali {hostname} kernel: Packet log: input {ipchains name} DENY {action} eth0 {interface} PROTO=6 { protocol} 24.14.93.7:2529 {source address and port} 24.92.71.224:1080 {destination address and port} L=48 {packet length (bytes)} S=0x00 {TOS} I=3040 {IP ID} F=0x4000 {16 bit fragment offset} T=113 {TTL} SYN {TCP flag} (#1) {ipchains rule number}</p>
3	Probability the source address was spoofed:
	Low probability of spoofed address. This is a reconnaissance scan used primarily to gather Netbios host information. The intruder's IP belongs to the address space own by @Home cable network provider.
4	Description of attack:
	The intruder is attempting to initiate a "Node Status" query to a local Netbios naming service. Although the existence of this probe by itself might not be hostile in nature, depending on the information that is gathered from the request, it might pose a security concern. The windows naming service might disclose private system information (user, workgroup,etc) that could be used to mount subsequent attacks. The use of UDP source port 137 usually indicates that the probe is originating from a windows machine.
5	Attack mechanism:
	The Netbios NS request (137/udp) is usually the prelude to other more hostile attempts to exploit improperly configured windows file sharing service. The acknowledgement of a Netbios NS request might prompt subsequent probes for globally accessible shares, file shares with weak password or Scope ID, and other file sharing exposures. Any of these exposures could potentially led to a file system compromise. One of the first Netbios open share scanner was <i>legion</i> from Rhino9. In the meantime, there have been emergencies of Netbios worms (e.g. ExploreZip virus/worm, Network.VBS VisualBasic script, and the 911 worm), which propagates through the network via windows file shares.
6	Correlations:
	The risk identified with Netbios and windows file sharing has been documented extensively (CAN-1999-0520, CAN-1999-0554) and has even made SANS' Top 10 Internet Security Threats list.
7	Evidence of active targeting:
	This attack was logged on a specific host. It is most likely that this attack is just one in a series of probe of the RR's cable LAN.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (2 + 1) – (5 + 4) = -6

9	Defensive recommendations:
	The current network and system countermeasures are effective. Considering the volume of Netbios scans, it might be advisable to just block all Netbios traffic (139/tcp, 137/udp).
10	Multiple choice test question based on trace and analysis:
	<p>What are the possible reasons for windows to legitimately send a Netbios query?</p> <ul style="list-style-type: none"> a) Your DNS servers are slow; or your link is unreliable/congested, causing the DNS queries to be dropped. b) You haven't configured the PTR entries within your DNS server. c) Your ISP doesn't forward the PTR queries to your DNS server. d) The client's ISP cannot handle CNAME -> PTR indirection for CIDR addresses. e) All of the above <p>Answer: e)</p>

Detect 3:

Jun 10 15:34:16 bali portsentry[437]: attackalert: SYN/Normal scan from host: a-ve11-8.tin.it/212.216.39.7 to TCP port: 12345

Jun 10 15:34:17 bali kernel: Packet log: input DENY eth0 PROTO=6 212.216.39.7:1848 24.92.71.224:12345 L=48 S=0x00 I=54796 F=0x4000 T=108 SYN (#1)

1	Source of trace:
	Linux host residing on cable network
2	Detect was generated by:
	<p>a) Portsentry from Psionic software Jun 8 21:10:44 {date/time} bali {hostname} portsentry[437] {service:PID} : attackalert: SYN/Normal scan from host {portsentry scan warning}: c604394-a.altn1.tx.home.com/24.14.93.7 {source address} to TCP port: 1080 {target port & protocol}</p> <p>b) Linux IPChains firewall Jun 8 21:10:45 {date/time} bali {hostname} kernel: Packet log: input {ipchains name} DENY {action} eth0 {interface} PROTO=6 {protocol} 24.14.93.7:2529 {source address and port} 24.92.71.224:1080 {destination address and port} L=48 {packet length (bytes)} S=0x00 {TOS} I=3040 {IP ID} F=0x4000 {16 bit fragment offset} T=113 {TTL} SYN {TCP flag} (#1) {ipchains rule number}</p>
3	Probability the source address was spoofed:
	Low probability of spoofed address. IP belongs to an (dailup and ADSL) ISP in Italy.

4	Description of attack:
	Netbus Trojan horse probe. A quick search through the logs did not turn up any historical information for that particular domain, or Netbus scan.
5	Attack mechanism:
	Netbus is one of the earlier Trojan horse targeted at the win32 platform. Its origin dates back to September 1998, where it was the earliest backdoor program that worked on the NT platform.
6	Correlations:
	CAN-1999-0660 ** CANDIDATE ** A hacker utility or Trojan Horse is installed on a system, e.g. NetBus, Back Orifice, Rootkit, etc. ISS Vulnerability Alert (September 10, 1998) - Windows Backdoors Update; win-netbus-installed (1228)
7	Evidence of active targeting:
	This attack was logged on a specific host. There is no evidence of active targeting from the information gathered from logs.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (2 + 1) – (5 + 4) = -6
9	Defensive recommendations:
	None needed. *nix boxes are not susceptible to Netbus.
10	Multiple choice test question based on trace and analysis:
	Identify the above scan: a) Back orifice (BO) b) Netbus 2.0 c) SubSeven d) Netbus Answer: d)

Detect 4:

```

Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=1 24.92.82.40:0 24.92.71.224:0 L=28 S=0x00 I=26461 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:80 24.92.71.224:36235 L=40 S=0x00 I=26717 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:320 24.92.71.224:36215 L=40 S=0x00 I=26973 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:2784 24.92.71.224:36215 L=40 S=0x00 I=27229 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:338 24.92.71.224:36215 L=40 S=0x00 I=27485 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:377 24.92.71.224:36215 L=40 S=0x00 I=27741 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:900 24.92.71.224:36215 L=40 S=0x00 I=27997 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:6666 24.92.71.224:36215 L=40 S=0x00 I=28253 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:843 24.92.71.224:36215 L=40 S=0x00 I=28509 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1437 24.92.71.224:36215 L=40 S=0x00 I=28765 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:29 24.92.71.224:36215 L=40 S=0x00 I=29021 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1990 24.92.71.224:36215 L=40 S=0x00 I=29277 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:718 24.92.71.224:36215 L=40 S=0x00 I=29533 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:4 24.92.71.224:36215 L=40 S=0x00 I=29789 F=0x0000 T=123 (#24)
Additional logs truncated for brevity. It is available in the footnotei
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1490 24.92.71.224:36215 L=40 S=0x00 I=21859 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:725 24.92.71.224:36215 L=40 S=0x00 I=22115 F=0x0000 T=123 (#24)

```

```

Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:504 24.92.71.224:36215 L=40 S=0x00 I=22371 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:9535 24.92.71.224:36215 L=40 S=0x00 I=22627 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:807 24.92.71.224:36215 L=40 S=0x00 I=22883 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:848 24.92.71.224:36215 L=40 S=0x00 I=23139 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:2003 24.92.71.224:36215 L=40 S=0x00 I=23395 F=0x0000 T=123 (#24)
Jun 6 08:01:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:986 24.92.71.224:36215 L=40 S=0x00 I=23651 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36222 L=44 S=0x00 I=23907 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36223 L=40 S=0x00 I=24163 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36224 L=44 S=0x00 I=24419 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36225 L=40 S=0x00 I=24675 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1 24.92.71.224:36226 L=40 S=0x00 I=24931 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1 24.92.71.224:36227 L=40 S=0x00 I=25187 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1 24.92.71.224:36228 L=40 S=0x00 I=25443 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=1 24.92.82.40:3 24.92.71.224:3 L=56 S=0x00 I=25699 F=0x0000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36216 L=44 S=0x00 I=25955 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36217 L=44 S=0x00 I=26211 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36218 L=44 S=0x00 I=26467 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36219 L=44 S=0x00 I=26723 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36220 L=44 S=0x00 I=26979 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36221 L=44 S=0x00 I=27235 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36222 L=44 S=0x00 I=27491 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36224 L=44 S=0x00 I=27747 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36216 L=44 S=0x00 I=28003 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36217 L=44 S=0x00 I=28259 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36218 L=44 S=0x00 I=28515 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36219 L=44 S=0x00 I=28771 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36220 L=44 S=0x00 I=29027 F=0x4000 T=123 (#24)
Jun 6 08:01:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36221 L=44 S=0x00 I=29283 F=0x4000 T=123 (#24)
Jun 6 08:01:11 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36215 L=44 S=0x00 I=29539 F=0x4000 T=123 (#24)
Jun 6 08:01:14 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36222 L=44 S=0x00 I=29795 F=0x4000 T=123 (#24)
Jun 6 08:01:14 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36224 L=44 S=0x00 I=30051 F=0x4000 T=123 (#24)
Jun 6 08:01:14 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36216 L=44 S=0x00 I=30307 F=0x4000 T=123 (#24)
Jun 6 08:01:14 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36217 L=44 S=0x00 I=30563 F=0x4000 T=123 (#24)
Jun 6 08:01:14 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36218 L=44 S=0x00 I=30819 F=0x4000 T=123 (#24)
Jun 6 08:01:15 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36219 L=44 S=0x00 I=31075 F=0x4000 T=123 (#24)
Jun 6 08:01:15 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36220 L=44 S=0x00 I=31331 F=0x4000 T=123 (#24)
Jun 6 08:01:15 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36221 L=44 S=0x00 I=31587 F=0x4000 T=123 (#24)
Jun 6 08:01:23 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36215 L=44 S=0x00 I=31843 F=0x4000 T=123 (#24)
Jun 6 08:01:26 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36222 L=44 S=0x00 I=32099 F=0x4000 T=123 (#24)
Jun 6 08:01:26 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36224 L=44 S=0x00 I=32355 F=0x4000 T=123 (#24)
Jun 6 08:01:26 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36216 L=44 S=0x00 I=32611 F=0x4000 T=123 (#24)
Jun 6 08:01:26 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36217 L=44 S=0x00 I=32867 F=0x4000 T=123 (#24)
Jun 6 08:01:26 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36218 L=44 S=0x00 I=33123 F=0x4000 T=123 (#24)
Jun 6 08:01:27 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36219 L=44 S=0x00 I=33379 F=0x4000 T=123 (#24)
Jun 6 08:01:27 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36220 L=44 S=0x00 I=33635 F=0x4000 T=123 (#24)
Jun 6 08:01:27 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:139 24.92.71.224:36221 L=44 S=0x00 I=33891 F=0x4000 T=123 (#24)

```

1	Source of trace:
	Linux host residing on cable network
2	Detect was generated by:
	<p>a) Portsentry from Psionic software Jun 8 21:10:44 {date/time} bali {hostname} portsentry[437] {service:PID} : attackalert: SYN/Normal scan from host {portsentry scan warning}: c604394-a.altn1.tx.home.com/24.14.93.7 {source address} to TCP port: 1080 {target port & protocol}</p> <p>b) Linux IPChains firewall Jun 8 21:10:45 {date/time} bali {hostname} kernel: Packet log: input {ipchains name} DENY {action} eth0 {interface} PROTO=6 {protocol} 24.14.93.7:2529 {source address and port} 24.92.71.224:1080 {destination address and port} L=48 {packet length (bytes)} S=0x00 {TOS} I=3040 {IP ID} F=0x4000 {16 bit fragment offset} T=113 {TTL} SYN {TCP flag} (#1) {ipchains rule}</p>

	number}
3	Probability the source address was spoofed:
	The use of low source ports suggests that the packets might be crafted, and the probability that the source address have been spoofed. Someone is probably spoofing our address.
4	Description of attack:
	This attack has the characteristic of a Denial of Service attempt using IP address spoofing. The source address captured in the logs is just probably an intermediate host being used by the real attacker. The real attacker is spoofing our address (i.e., 24.92.71.224) and directing those packets to the victims (i.e., 24.92.82.40). One of the evidence of IP spoofing is the presence of the uninitiated ICMP echo reply (ICMP type 0), and ICMP destination unreachable packets (ICMP type 3) response. The low TTL of the packets suggest that the source host might be a windows box (TTL = 128) ⁱⁱⁱ . The IP ID sequence might possibly suggests an x86/Intel box, with its erratic progression cause by the reverse byte order of incrementing the IP ID (i.e., characteristic of little endian machines).
5	Attack mechanism:
	Spoofing attacks occur when the real attacker masquerades as the victim A's IP address. They can then flood an intermediate victim B with the source address of A. This will allow the real attacker to anonymously mount a DoS attack on both victims A and B.
6	Correlations:
	IP spoofing was discusses in the Network Intrusion detection analysis class in SANS 2000 by Stephen Northcutt. DoS attacks are documented in the GIAC 12/22/99 detects, as well as CVE-1999-0074 Listening TCP ports are sequentially allocated, allowing spoofing attacks.
7	Evidence of active targeting:
	Yes, the attacker actively targeted our host for this attack.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (2 + 4) – (5 + 2) = -1
9	Defensive recommendations:
	The countermeasures are adequate. The incident might also be reported to the ISP for investigation.
10	Multiple choice test question based on trace and analysis:
	a) Host scan b) Port scan c) RPC scans d) Third party DoS flood
	Answer: d)

Detect 5:

Jun 10 22:31:38 bali portsentry[437]: attackalert: SYN/Normal scan from host: mlhAsIn144.midsouth.rr.com/24.92.69.144 to TCP port: 7
Jun 10 22:31:38 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3001 24.92.71.224:7 L=48 S=0x00 I=46901 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:39 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3021 24.92.71.224:21 L=48 S=0x00 I=58933 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:41 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3040 24.92.71.224:23 L=48 S=0x00 I=12086 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:42 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3021 24.92.71.224:21 L=48 S=0x00 I=29494 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:42 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3058 24.92.71.224:25 L=48 S=0x00 I=29750 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:43 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3040 24.92.71.224:23 L=48 S=0x00 I=48182 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:44 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3080 24.92.71.224:37 L=48 S=0x00 I=50486 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:44 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3001 24.92.71.224:7 L=48 S=0x00 I=56374 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:45 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3058 24.92.71.224:25 L=48 S=0x00 I=3639 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:45 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3100 24.92.71.224:53 L=48 S=0x00 I=4919 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:47 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3080 24.92.71.224:37 L=48 S=0x00 I=22839 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:47 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3120 24.92.71.224:80 L=48 S=0x00 I=23351 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:48 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3021 24.92.71.224:21 L=48 S=0x00 I=39991 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:48 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3100 24.92.71.224:53 L=48 S=0x00 I=40759 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:48 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3137 24.92.71.224:110 L=48 S=0x00 I=41271 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:49 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3040 24.92.71.224:23 L=48 S=0x00 I=50743 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:50 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3120 24.92.71.224:80 L=48 S=0x00 I=52279 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:50 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3146 24.92.71.224:119 L=48 S=0x00 I=53559 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:51 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3058 24.92.71.224:25 L=48 S=0x00 I=59959 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:51 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3137 24.92.71.224:110 L=48 S=0x00 I=60727 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:53 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3080 24.92.71.224:37 L=48 S=0x00 I=1080 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:53 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3146 24.92.71.224:119 L=48 S=0x00 I=2104 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:54 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3100 24.92.71.224:53 L=48 S=0x00 I=7480 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:56 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3120 24.92.71.224:80 L=48 S=0x00 I=12088 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:56 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3001 24.92.71.224:7 L=48 S=0x00 I=13368 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:57 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3137 24.92.71.224:110 L=48 S=0x00 I=17720 F=0x4000 T=125 SYN (#1)
Jun 10 22:31:59 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3146 24.92.71.224:119 L=48 S=0x00 I=22072 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:00 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3021 24.92.71.224:21 L=48 S=0x00 I=25656 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3040 24.92.71.224:23 L=48 S=0x00 I=28472 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:03 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3058 24.92.71.224:25 L=48 S=0x00 I=28984 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:05 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3080 24.92.71.224:37 L=48 S=0x00 I=29496 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:06 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3100 24.92.71.224:53 L=48 S=0x00 I=30008 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:08 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3120 24.92.71.224:80 L=48 S=0x00 I=30520 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:09 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3137 24.92.71.224:110 L=48 S=0x00 I=31032 F=0x4000 T=125 SYN (#1)
Jun 10 22:32:11 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.69.144:3146 24.92.71.224:119 L=48 S=0x00 I=31288 F=0x4000 T=125 SYN (#1)

1 Source of trace:

Linux host residing on cable network	
2	Detect was generated by:
	<p>a) Portsentry from Psionic software Jun 8 21:10:44 {date/time} bali {hostname} portsentry[437] {service:PID} : attackalert: SYN/Normal scan from host {portsentry scan warning}: c604394- a.altn1.tx.home.com/24.14.93.7 {source address} to TCP port: 1080 {target port & protocol}</p> <p>b) Linux IPChains firewall Jun 8 21:10:45 {date/time} bali {hostname} kernel: Packet log: input {ipchains name} DENY {action} eth0 {interface} PROTO=6 { protocol} 24.14.93.7:2529 {source address and port} 24.92.71.224:1080 {destination address and port}L=48 {packet length (bytes)} S=0x00 {TOS} I=3040 {IP ID} F=0x4000 {16 bit fragment offset} T=113 {TTL} SYN {TCP flag} (#1) {ipchains rule number}</p>
3	Probability the source address was spoofed:
	<p>Low probability of spoof address. A port scan is usually an information-gathering maneuver, and the fact that the intruder is using a half open scan implies their determination to collect that information without triggering most IDS. The TTL (if we assume that it started with TTL 128) is also consistent with the proximity of the source IP (i.e. source IP is from the same ISP network as the monitoring host).</p>
4	Description of attack:
	<p>Half open (SYN) port scan targeting ports 7(Echo), 21(FTP), 23(Telnet), 25(SMTP), 37(Time), 53(Domain), 80(HTTP), 110(Pop), 119(NNTP). A static source port is used for each destination port that is scanned. An unsuccessful port probe is repeated at least 3 times before it finally gives up. The low frequency of probes might suggest that source machine might be busy scanning other hosts concurrently.</p>
5	Attack mechanism:
	<p>The objective of a port scan is to obtain recon information on the targeted services on a host. This information can be used to formulate further attacks (e.g. DOS using the echo port and ftp bounces, CVE-1999-0103; remote exploits via SMTP, CVE -1999-0096, CVE -1999-0096; DNS zone transfer, popd, and NNTP)</p>
6	Correlations:
	<p>Stealthy port scans were discussed in Network based Intrusion detection analysis by Stephen Northcutt in SANS2000 San Jose. Port scans using “stealthy” methods (SYN, FIN scans) are also extensively reviewed by the Global Incident Analyst Center. This attack shares characteristics with the “msscan” and “sscan” exploit script.</p>
7	Evidence of active targeting:
	<p>Yes, the port scan was actively targeted at the specific host.</p>
8	Severity:
	<p>(Critical + Lethal) – (System + Network countermeasures) = Severity (2 + 2) – (5 + 4) = -5</p>

9	Defensive recommendations:
	The current defensives are adequate.
10	Multiple choice test question based on trace and analysis:
	<ul style="list-style-type: none"> a) SYN flood b) Port scans c) Portmapper d) Stealth/Half open port scans
	Answer: d)

Detect 6:

```

4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.0.255;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.2.255;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.4.255; 8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.0.0;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.1.255;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.3.255;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.2.0;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.4.0;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.3.0;8;0
4Jun2000;11:42:13; drop;;nf2;inbound;icmp;24.234.102.37;x.x.1.0;8;0

```

1	Source of trace:
	<p>My network</p> <p>The 1st two octets of the destination networks have been intentionally removed.</p>
2	Detect was generated by:
	<p>Firewall-1</p> <p>Date;time;action;i/f_name;i/f_dir;proto;src;dst;service;icmp-type;icmp-code</p>
3	Probability the source address was spoofed:
	<p>Low probability of the source address been spoofed. The source address belongs to COX cable network. According to ARIN, that network block is registered to someone geographically located near Las Vegas.</p>
4	Description of attack:
	<p>Recon scan for directed broadcast network</p>
5	Attack mechanism:
	<p>The intruder is attempting to locate any network that has an improperly configured router that allows directed broadcast. This is done by pinging 2 special network address (x.x.x.0 and x.x.x.255). Pinging a directed broadcast enabled router is equivalent to pinging every host in that network. A vulnerable network will become “amplifiers” for mounting DoS</p>

	attacks. A common exploit that for IP directed broadcast is the “smurf” and “papasmurf” script.
6	Correlations:
	CVE-1999-0513 - ICMP messages to broadcast addresses are allowed, allowing for a Smurf attack that can cause a denial of service.
7	Evidence of active targeting:
	This is a recon scan. There is no evidence of active targeting involved.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (5 + 4) – (4 + 5) = 0
9	Defensive recommendations:
	The firewall provides effective defense against these scans. But it is recommended that directed broadcast must be disabled on all routers.
10	Multiple choice test question based on trace and analysis:
	a) Smurf DoS attack b) Host scans c) Port scans d) Recon scan for directed broadcast network
	Answer: d)

Detect 7:

```
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.119;sunrpc;1749;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.120;sunrpc;1750;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.121;sunrpc;1751;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.122;sunrpc;1752;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.123;sunrpc;1753;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.124;sunrpc;1754;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.125;sunrpc;1755;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.126;sunrpc;1756;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.127;sunrpc;1757;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.128;sunrpc;1758;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.129;sunrpc;1759;60
5Jun2000; 1:02:02;drop;;nf2;inbound;tcp;159.226.63.139;x.x.153.130;sunrpc;1760;60
```

1	Source of trace:
	My network The 1 st two octets of the destination networks have been intentionally removed.
2	Detect was generated by:
	Firewall-1 Date;time;action;i/f name;i/f dir;proto;src;dst;service;s_port;len
3	Probability the source address was spoofed:

	Low probability of the source address was spoofed. The source address belongs to the Institute of Computing Technology Chinese Academy of Sciences campus.
4	Description of attack:
	Sun RPC portmapper attack The intruder is trying to probe the sun RPC portmapper service to gather information on RPC services. Judging by the frequency of probes, this is probably a scripted attack.
5	Attack mechanism:
	The Sun RPC portmapper (or rpcbind) is an interesting port because it provides critical information that all the RPC services such as, as rpc.mountd, NFS, rpc.statd, rpc.cmsd, rpc.ttybd, amd, etc (which are notoriously for being vulnerable to buffer overflows remote exploits). With this information, the intruder can proceed to mounts attack against those RPC services. Some of the commands that might be capture with a content sensitive IDS are : <ul style="list-style-type: none"> • Null • Set – Use internally between RPC services and the portmapper. Usually a good indicator of an intrusion. • Unset - Use internally between RPC services and the portmapper. Usually a good indicator of an intrusion. • Getport. • Dump – Most intrusion use this command to retrieve service information. Most commonly seen in recon scans. • Callit - The <i>callit</i> feature was created for RPC broadcasts. Sun boxes usually use ports between 32770 and 32900 for RPC services.
6	Correlations:
	RPC exploits are very common, and notorious for many host compromises. It is listed in the Top 10 most critical Internet security threat published by SANS. rpc.cmsd – CVE-1999-0696 rpc.statd - CVE-1999-0018, CVE-1999-0019 rpc.ttdbserverd - CVE-1999-0687, CVE-1999-0003, CVE-1999-0693
7	Evidence of active targeting:
	There is no evidence of active targeting. This is just a general scan of a class B network.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (3 + 5) – (4 + 5) = -1
9	Defensive recommendations:
	Network defenses are effective against RPC attacks. But it is recommended that unnecessary RPC services be disabled on hosts.
10	Multiple choice test question based on trace and analysis:
	How can you obtain information on RPC services on a hosts? a) Query the portmapper

- b) Scan all TCP high ports
- c) Scan all UDP high ports
- d) All of the above

Answer: d)

Detect 8:

[**] Possible SubSeven access [**]
 05/19-06:33:41.579445 24.95.140.215:3781 -> 24.95.107.137:1243
 TCP TTL:111 TOS:0x0 ID:30116 DF
 S*** Seq: 0x254A7A Ack: 0x0 Win: 0x2000
 TCP Options => MSS: 1460 NOP NOP SackOK

[**] Possible SubSeven access [**]
 05/19-06:33:42.328713 24.95.140.215:3781 -> 24.95.107.137:1243
 TCP TTL:111 TOS:0x0 ID:43428 DF
 S*** Seq: 0x254A7A Ack: 0x0 Win: 0x2000
 TCP Options => MSS: 1460 NOP NOP SackOK

[**] Possible SubSeven access [**]
 05/19-06:33:43.143867 24.95.140.215:3781 -> 24.95.107.137:1243
 TCP TTL:111 TOS:0x0 ID:47268 DF
 S*** Seq: 0x254A7A Ack: 0x0 Win: 0x2000
 TCP Options => MSS: 1460 NOP NOP SackOK

[**] Possible SubSeven access [**]
 05/19-06:33:43.846011 24.95.140.215:3781 -> 24.95.107.137:1243
 TCP TTL:111 TOS:0x0 ID:50340 DF
 S*** Seq: 0x254A7A Ack: 0x0 Win: 0x2000
 TCP Options => MSS: 1460 NOP NOP SackOK

1	Source of trace: Linux host residing on cable network
2	Detect was generated by: Snort IDS version 1.6
3	Probability the source address was spoofed: Low probability of spoofed source address.
4	Description of attack: Subseven Trojan attack The intruder is attempting to probe and connect to a Subseven server. After four unsuccessful requests to connect to port 1243, the client finally ceases its connection attempt.
5	Attack mechanism:

	Subseven is one of the many rogue Trojan programs for win32 platforms. A successful compromise occurs when the Subseven client connects to a server already installed beforehand. Although ports tcp/1243, tcp/6711, tcp/6712, tcp/6713, tcp/6776 are commonly used by the server, it can be configured to use any other ports.
6	Correlations:
	Subseven detects are widely seen and discussed in the GIAC. It has also been analyze in SANS Subseven Trojan special notice by George Wagner.
7	Evidence of active targeting:
	Yes, the intruder is actively targeting the host.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (2 + 1) – (5 + 2) = -4
9	Defensive recommendations:
	Current system and network defense are effective against an Subseven attack.
10	Multiple choice test question based on trace and analysis:
	What is the known port that Subseven 2.1 uses? a) tcp/1243 b) tcp/ 27374 c) udp/1243 d) udp/27374 Answer: b)

Detect 9:

10Jun2000;22:04:48;drop;;nf2;inbound;tcp;213.219.19.148;x.x.106.1;http;61405;48
10Jun2000;22:05:44;drop;;nf2;inbound;tcp;213.219.19.148;x.x.106.1;http;64671;48
10Jun2000;22:15:37;drop;;nf2;inbound;tcp;213.219.19.148;x.x.106.1;http;63216;48
10Jun2000;22:23:49;drop;;nf2;inbound;tcp;213.219.19.148;x.x.107.1;http;62925;48
10Jun2000;22:29:30;drop;;nf2;inbound;tcp;213.219.19.148;x.x.107.1;http;64972;48
10Jun2000;22:29:35;drop;;nf2;inbound;tcp;213.219.19.148;x.x.107.1;http;61098;48
10Jun2000;22:33:32;drop;;nf2;inbound;tcp;213.219.19.148;x.x.107.1;http;64093;48
10Jun2000;22:35:29;drop;;nf2;inbound;tcp;213.219.19.148;x.x.107.1;http;63686;48
10Jun2000;22:43:51;drop;;nf2;inbound;tcp;213.219.19.148;x.x.108.1;http;63818;48
10Jun2000;22:48:33;drop;;nf2;inbound;tcp;213.219.19.148;x.x.108.1;http;61908;48
10Jun2000;22:52:34;drop;;nf2;inbound;tcp;213.219.19.148;x.x.108.1;http;61233;48
10Jun2000;22:53:42;drop;;nf2;inbound;tcp;213.219.19.148;x.x.108.1;http;62319;48
10Jun2000;22:54:31;drop;;nf2;inbound;tcp;213.219.19.148;x.x.108.1;http;64784;48

1	Source of trace:
	My network The 1 st two octets of the destination networks have been intentionally

	removed.
2	Detect was generated by:
	Firewall-1 Date;time;action;i/f_name;i/f_dir;proto;src;dst;service;s_port;len
3	Probability the source address was spoofed:
	Low probability of a spoofed source address because the detect suggests the intruder was on a information gathering maneuver.
4	Description of attack:
	This looked like an uninteresting host scans for http server except that intruder was targeting just one specific IP out of each network. The host scan was targeted at the 1 st IP of each /24 slice network, which coincidentally is where most network administrator uses for core network gear such as routers, and switches.
5	Attack mechanism:
	Some of those modern network equipment also support administration via a web interface (e.g. Cisco IOS routers/switches). The intruder is scanning the network for an improperly configured routers with a active administrative web interface. These system could be susceptible to brute password cracking and denial of service attack.
6	Correlations:
	The IOS HTTP service in Cisco routers and switches running IOS 11.1 through 12.1 allows remote attackers to cause a denial of service by requesting a URL that contains a %% string - CAN-2000-0380 (under review).
7	Evidence of active targeting:
	The intruder is attempting a blanket scan of http servers. There is no evident of active targeting.
8	Severity:
	(Critical + Lethal) – (System + Network countermeasures) = Severity (5 + 4) – (4 + 3) = 1
9	Defensive recommendations:
	The network defenses are effective against such attacks.
10	Multiple choice test question based on trace and analysis:
	a) Host scan b) CGI exploit c) Recon scan of core infrastructure d) HTTP flood
	Answer: c)

Detect 10:

29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.87;domain;65535;40

29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.89;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.91;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.92;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.93;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.94;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.95;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.98;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.99;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.101;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.103;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.105;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.109;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.110;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.112;domain;65535;40
 29May2000; 3:30:53;drop;;nf2;inbound;tcp;195.76.27.44;x.x.18.113;domain;65535;40

1	Source of trace: My network The 1 st two octets of the destination networks have been intentionally removed.
2	Detect was generated by: Firewall-1 Date;time;action;i/f_name;i/f_dir;proto;src;dst;service;s_port;len
3	Probability the source address was spoofed: Low probability of spoof source address. The source address is registered to an ISP in Spain.
4	Description of attack: BIND/named attack The intruder is attempting to exploit any hosts running vulnerable versions of BIND. Judging from the source port (port number 65535), it is likely that the packets were crafted and may contain the exploit code to specially exploit one of the many buffer overflow issues in BIND. The attacks were very swift and furious.
5	Attack mechanism: There are numerous bugs (“nxt bug”, “sig bug”, “so_linger bug”, “fdmax bug”, “maxdname bug”, “naptr buf”) in bind, which results in varying degree of security vulnerability. They might result in slowdown in performance, denial of service, or complete remote root compromise of the DNS server. Alternatively, the above detect could also be indicative of an intruder trying to execute a DNS zone transfer. Zone files provide an attacker with a lot of information on the network architecture, and design of the victim’s network.
6	Correlations: Bind remote exploits and denial of service have been the bane of many security administrators, and it’s no wonder that this threat made it to the list of Top 10 most critical Internet threat.

nxt CVE-1999-0833
qinv CVE-1999-0009
Other related entries: CVE-1999-0835, CVE-1999-0848, CVE-1999-0849,
CVE-1999-0851
CERT® Advisory CA-99-14 Multiple Vulnerabilities in BIND

7 Evidence of active targeting:

The intruder is attempting a bulk scan of all hosts in our network. There is no evidence of the intruder targeting a specific DNS server.

8 Severity:

(Critical + Lethal) – (System + Network countermeasures) = Severity
(5 + 5) – (5 + 4) = 1

9 Defensive recommendations:

Network defense are effective against a DNS scan. It is recommended that the production DNS servers be audited on a regular basis to verify that all of them are securely configured.

10 Multiple choice test question based on trace and analysis:

Which of the following is not true?

- a) DNS DoS attack
- b) Named remote root exploit
- c) DNS zone transfer
- d) DNS name lookup

Answer: d)

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

References:

ⁱ Detail explanation of ipchains

The kernel logs this information looking like:

```
Packet log: input DENY eth0 PROTO=17 192.168.2.1:53 192.168.1.1:1025
L=34 S=0x00 I=18 F=0x0000 T=254
```

This log message is designed to be terse, and contain technical information useful only to networking gurus, but it can be useful to the rest of us. It breaks down like so:

1. `'input'` is the chain which contained the rule which matched the packet, causing the log message.
2. `'DENY'` is what the rule said to do to the packet. If this is `'I'` then the rule didn't effect the packet at all (an accounting rule).
3. `'eth0'` is the interface name. Because this was the input chain, it means that the packet came in `'eth0'`.
4. `'PROTO=17'` means that the packet was protocol 17. A list of protocol numbers is given in `'/etc/protocols'`. The most common are 1 (ICMP), 6 (TCP) and 17 (UDP).
5. `'192.168.2.1'` means that the packet's source IP address was 192.168.2.1.
6. `'53'` means that the source port was port 53. Looking in `'/etc/services'` shows that this is the `'domain'` port (ie. this is probably an DNS reply). For UDP and TCP, this number is the source port. For ICMP, it's the ICMP type. For others, it will be 65535.
7. `'192.168.1.1'` is the destination IP address.
8. `'1025'` means that the destination port was 1025. For UDP and TCP, this number is the destination port. For ICMP, it's the ICMP code. For others, it will be 65535.
9. `'L=34'` means that packet was a total of 34 bytes long.
10. `'S=0x00'` means the Type of Service field (divide by 4 to get the Type of Service as used by ipchains).
11. `'I=18'` is the IP ID.
12. `'F=0x0000'` is the 16-bit fragment offset plus flags. A value starting with `'0x4'` or `'0x5'` means that the Don't Fragment bit is set. `'0x2'` or `'0x3'` means the 'More Fragments' bit is set; expect more fragments after this. The rest of the number is the offset of this fragment, divided by 8.
13. `'T=254'` is the Time To Live of the packet. One is subtracted from this value for every hop, and it usually starts at 15 or 255. `'(5)'` there may be a final number in brackets on more recent kernels (perhaps after 2.2.9). This is the rule number which caused the packet log.

ⁱⁱ Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:5977 24.92.71.224:36215 L=40 S=0x00 I=30045 F=0x0000 T=123 (#24)

```
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:944 24.92.71.224:36215 L=40 S=0x00 I=30301 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:802 24.92.71.224:36215 L=40 S=0x00 I=30557 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1532 24.92.71.224:36215 L=40 S=0x00 I=30813 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:938 24.92.71.224:36215 L=40 S=0x00 I=31069 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:295 24.92.71.224:36215 L=40 S=0x00 I=31325 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:888 24.92.71.224:36215 L=40 S=0x00 I=31581 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1380 24.92.71.224:36215 L=40 S=0x00 I=31837 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:50 24.92.71.224:36215 L=40 S=0x00 I=32093 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:975 24.92.71.224:36215 L=40 S=0x00 I=32349 F=0x0000 T=123 (#24)
Jun 6 08:01:01 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:400 24.92.71.224:36215 L=40 S=0x00 I=32605 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1416 24.92.71.224:36215 L=40 S=0x00 I=32861 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:776 24.92.71.224:36215 L=40 S=0x00 I=33117 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1103 24.92.71.224:36215 L=40 S=0x00 I=33373 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:198 24.92.71.224:36215 L=40 S=0x00 I=33629 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:193 24.92.71.224:36215 L=40 S=0x00 I=33885 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:336 24.92.71.224:36215 L=40 S=0x00 I=34141 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:1442 24.92.71.224:36215 L=40 S=0x00 I=34397 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:460 24.92.71.224:36215 L=40 S=0x00 I=34653 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:208 24.92.71.224:36215 L=40 S=0x00 I=34909 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:628 24.92.71.224:36215 L=40 S=0x00 I=35165 F=0x0000 T=123 (#24)
Jun 6 08:01:02 bali kernel: Packet log: input DENY eth0 PROTO=6 24.92.82.40:2108 24.92.71.224:36215 L=40 S=0x00 I=35421 F=0x0000 T=123 (#24)
```


Upcoming Training

Click Here to
{Get CERTIFIED!}



SANS Northern VA Spring - Tysons 2018	McLean, VA	Mar 17, 2018 - Mar 24, 2018	Live Event
SANS Secure Canberra 2018	Canberra, Australia	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS 2018	Orlando, FL	Apr 03, 2018 - Apr 10, 2018	Live Event
SANS Abu Dhabi 2018	Abu Dhabi, United Arab Emirates	Apr 07, 2018 - Apr 12, 2018	Live Event
SANS London April 2018	London, United Kingdom	Apr 16, 2018 - Apr 21, 2018	Live Event
SANS Baltimore Spring 2018	Baltimore, MD	Apr 21, 2018 - Apr 28, 2018	Live Event
Baltimore Spring 2018 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Apr 23, 2018 - Apr 28, 2018	vLive
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201805,	May 02, 2018 - Jun 14, 2018	vLive
Community SANS Virginia Beach SEC503	Virginia Beach, VA	May 07, 2018 - May 12, 2018	Community SANS
SANS Security West 2018	San Diego, CA	May 11, 2018 - May 18, 2018	Live Event
Mentor Session - SEC503	Dulles, VA	May 24, 2018 - Jun 28, 2018	Mentor
SANS Oslo June 2018	Oslo, Norway	Jun 18, 2018 - Jun 23, 2018	Live Event
Minneapolis 2018 - SEC503: Intrusion Detection In-Depth	Minneapolis, MN	Jun 25, 2018 - Jun 30, 2018	vLive
SANS Minneapolis 2018	Minneapolis, MN	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS London July 2018	London, United Kingdom	Jul 02, 2018 - Jul 07, 2018	Live Event
SANSFIRE 2018	Washington, DC	Jul 14, 2018 - Jul 21, 2018	Live Event
Security Operations Summit & Training 2018	New Orleans, LA	Jul 30, 2018 - Aug 06, 2018	Live Event
SANS San Antonio 2018	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	Live Event
Community SANS Columbia SEC503	Columbia, MD	Aug 13, 2018 - Aug 18, 2018	Community SANS
SANS Virginia Beach 2018	Virginia Beach, VA	Aug 20, 2018 - Aug 31, 2018	Live Event
SANS Tokyo Autumn 2018	Tokyo, Japan	Sep 03, 2018 - Sep 15, 2018	Live Event
SANS Amsterdam September 2018	Amsterdam, Netherlands	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Network Security 2018	Las Vegas, NV	Sep 23, 2018 - Sep 28, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced