Global Information Assurance Certification Paper

GIAC Intrusion Detection Practical Assignment

John Vajda

Submitted: June 15, 2000
Conference: SANS2000 San Jose, May 8-12, 2000

Contents

Detect 1 - Web / Proxy Server Probe

Jun 11 02:14:01 rtr1 560103: Jun 11 10:14:00: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2317) -> xxx.yyy.152.3(80), 1 packet
Jun 11 02:14:01 rtr2 493893: Jun 11 10:14:00: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2358) -> xxx.yyy.152.24(8080), 1 packet
Jun 11 02:14:04 rtr1 560106: Jun 11 10:14:03: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2366) -> xxx.yyy.152.28(8080), 1 packet
Jun 11 02:14:06 rtr1 560109: Jun 11 10:14:06: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2370) -> xxx.yyy.152.30(8080), 1 packet
Jun 11 02:14:06 rtr1 560108: Jun 11 10:14:05: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2364) -> xxx.yyy.152.27(8080), 1 packet
Jun 11 02:14:09 rtr1 560110: Jun 11 10:14:08: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2313) -> xxx.yyy.152.1(80), 1 packet
Jun 11 02:14:10 rtr1 560111: Jun 11 10:14:09: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2393) -> xxx.yyy.152.41(80), 1 packet
Jun 11 02:14:12 rtr1 560112: Jun 11 10:14:11: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2362) -> xxx.yyy.152.26(8080), 1 packet
Jun 11 02:14:13 rtr1 560113: Jun 11 10:14:12: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2364) -> xxx.yyy.152.27(8080), 1 packet
Jun 11 02:14:16 rtr1 560115: Jun 11 10:14:15: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2382) -> xxx.yyy.152.36(8080), 1 packet
Jun 11 02:14:16 rtr1 560116: Jun 11 10:14:16: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2397) -> xxx.yyy.152.43(80), 1 packet
Jun 11 02:14:18 rtr1 560117: Jun 11 10:14:17: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2543) -> xxx.yyy.152.116(80), 1 packet
Jun 11 02:14:19 rtr1 560118: Jun 11 10:14:18: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2529) -> xxx.yyy.152.109(80), 1 packet
Jun 11 02:14:21 rtr1 560119: Jun 11 10:14:20: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2441) -> xxx.yyy.152.65(80), 1 packet
Jun 11 02:14:22 rtr1 560120: Jun 11 10:14:21: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2617) -> xxx.yyy.152.153(80), 1 packet
Jun 11 02:14:23 rtr1 560121: Jun 11 10:14:22: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2598) -> xxx.yyy.152.144(8080), 1 packet
Jun 11 02:14:25 rtr1 560122: Jun 11 10:14:24: %SEC-6-IPACCESSLOGP: list 199 denied tcp 61.142.73.247(2666) -> xxx.yyy.152.178(8080), 1 packet
Jun 11 02:14:27 rtr1 560123: Jun 11 10:14:26: %SEC-6-IPACCESSLOGP: list 199 denied

(Approx. 300 attempts to connect to tcp port 80 and 8080 on different addresses continue on and on, with very close time stamps. Source address is always the same)

1. Source of Trace

My network

2. Detect was generated by:

Cisco router ACL logs (syslog)

Explanation of fields:
Jun 11 02:14:25 *[date/time entry syslog'd, GMT*] **rtr1** *[router name]* **560122:** *[entry ID]* **Jun 11 10:14:24**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog*

level 6] **list 199** *[ACL]* **denied** *[action taken]* **tcp** *[protocol]* **61.142.73.247(2666)** *[src address/port]* **-> xxx.yyy.152.178(8080)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Unlikely. The protocol used was tcp (i.e. requiring 3-way handshake), and as indicated below, the attacker appears to be gathering information. The source address can be pinged. There was no DNS entry for the specific source address, but thanks to the whois proxy at http://www.geektools.com/cgi-bin/proxy.cgi, investigation showed that the address space belongs to *Chinanet, Guangdong Province Network.*

4. Description of attack

Reconnaissance. Attempting to map out internal web or proxy servers by running port scans across company's address space on ports 80 and 8080.

5. Attack Mechanism

The attacker is attempting to establish tcp connections on port 80 and 8080. Interestingly, the order of dest addresses scanned does not follow a specific pattern within the last octet. However, the attacker seems to be running the scans (within the third octet) in sequential order (x.y.226.z, then x.y.227.z, then x.y.228.z, and so on). The full trace is too large to include it here, but it sure is fun to import detects into excel and sort various columns to identify patterns!

6. Correlations

SANS has described that a large number of scans being detected are for proxy servers. See http://www.sans.org/y2k/proxy.htm

7. Evidence of active targeting

The attack appears to be a general scan of entire networks. It's worthwhile mentioning that two completely different address ranges, corresponding to two different domain names (yet both assigned to our company for different purposes) were scanned back to back with no time delay between scanning non-contiguous network ranges. The attacker seemed to be targeting our company specifically, rather than just a haphazard scan of internet address ranges.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(3 + 3) - (4 + 4) = -2

Criticality:	Internal web or proxy server
Lethality:	Subjective; Hard to know the attacker's ultimate goal. (Gave it a 3)
System:	Very hardened boxes in general, administered via ssh, patches kept current
Cntr-Meas	Routers have highly restrictive ACLs and block incoming connections on those ports.

9. Defensive Recommendation

Defenses are fine; attack was blocked by the router for this attack. However, router logs should be monitored for activity from this address for other types of attacks.

10. Multiple Choice Question

The detect indicates:

a) cgi-bin exploit
b) stealthy ping mapping
c) web/proxy server mapping
d) Invalid source ports

Answer is c.

Detect 2 - DNS Zone Transfer

Jun 11 00:30:19 rtr4 7913453: Jun 10 14:30:18: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(44442) -> xxx.yyy.zzz.36(53), 1 packet
Jun 11 00:31:18 rtr3 387879: Jun 10 14:31:17: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(44447) -> xxx.yyy.zzz.35(53), 1 packet
Jun 11 00:35:26 rtr4 7913462: Jun 10 14:35:25: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(44442) -> xxx.yyy.zzz.36(53), 6 packets
Jun 11 00:37:09 rtr3 387892: Jun 10 14:37:08: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(44447) -> xxx.yyy.zzz.35(53), 6 packets
Jun 12 00:35:50 rtr3 391751: Jun 11 14:35:49: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(52019) -> xxx.yyy.zzz.36(53), 1 packet
Jun 12 00:41:49 rtr3 391767: Jun 11 14:41:48: %SEC-6-IPACCESSLOGP: list 102 denied tcp 193.189.160.11(52019) -> xxx.yyy.zzz.36(53), 6 packets

1. Source of Trace

My network

2. Detect was generated by:

Cisco router ACL logs (syslog)

Explanation of fields:
Jun 11 00:30:52 *[date/time entry syslog'd, GMT*] **rtr3** *[router name]* **52019:** *[entry ID]* **Jun 11 14:30:18**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog level 6]* **list 102** *[ACL]* **denied** *[action taken]* **tcp** *[protocol]* **193.189.160.11(44442)** *[src address/port]* **-> xxx.yyy.zzz.35(53)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Low probability. The connections used tcp, requiring a three-way hand-shake. Since this appears to be a dns zone transfer attempt, the attacker would require that the information be able to get back to him/her.

An nslookup yields:
Name: taurus-1.siol.net
Address: 193.189.160.11

"Siol.net" suggest it might be a network service provider. A whois search yields:

inetnum: 193.189.160.0 - 193.189.163.255
netname: SIOL
descr: SiOL d.o.o. (Slovenia OnLine)
descr: **Biggest ISP in Slovenia**
descr: Slovenia
country: SI
admin-c: SM90-RIPE
admin-c: VM439-RIPE
tech-c: JZ23-RIPE

tech-c: TC124-RIPE

So now that we know the address range belongs to an ISP, we can investigate further to see if the address appears to be simply from a pool assigned to customers, or perhaps to the ISP itself.

More investigation using nslookup provides the following:

Authoritative answers can be found from:
siol.net nameserver = TAURUS-1.siol.net (it's the ISP's nameserver!)
siol.net nameserver = TAURUS-2.siol.net
TAURUS-1.siol.net internet address = 193.189.160.11
TAURUS-2.siol.net internet address = 193.189.160.12

4. Description of attack

Examining the log reveals an attempt by a dns server in Slovenia connect to TCP port 53 on two dns servers to initiate a DNS zone transfer. The risk is that if an unauthorized zone transfer succeeds, an attacker has an extensive mapping about one's network. TCP is used when a remote name server detects that its response to a server to server query would be greater than 512 bytes. When this occurs, the remote name server sends back a UDP packet telling the local name server to retry the query using TCP. The transfer attempts occurred on two successive nights against each name server.

5. Attack Mechanism

An ISP in Slovenia is attempting to initiate DNS zone transfers as part of a three way TCP handshake. This could be a due to a misconfiguation on the part of their DNS server, or possibly because the ISP wants to optimize DNS response time for there customers. Or there could be ulterior motives involved. In any case, the user is attempting to capture all DNS records for the targeted domain, allowing the user to have unauthorized knowledge about the targeted domain.

6. Correlations

There are some postings at
http://www.sans.org/y2k/053000-1000.htm (L. Christopher Paul submitted zone transfer scan.)
http://www.sans.org/y2k/052100.htm (Lisa Yeo submitted log file entries with source port 65535.)
http://advice.networkice.com/advice/Intrusions/2000401/default.htm

7. Evidence of active targeting

The zone transfers appear to be directed toward true dns servers rather than a network scan for DNS servers.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(5 + 3) - (4 + 5) = -1

Criticality: Targets are DNS servers
Lethality: The zone transfer itself is not lethal, but could lead to lethal attacks against other servers.
System: Hardened OS, and DNS is configured to not allow zone transfer requests
Cntr-Meas The routers denied connection attempts

9. Defensive Recommendation

Ensure that DNS zone transfer requests are limited to trusted servers, and use a split DNS strategy where internal hosts are not listed in Internet-accessible DNS servers.

10. Multiple Choice Question

A DNS zone transfer request can be initiated

a) only by another dns server
b) nslookup's ls command
c) dig's axfr command
d) all of the above

Answer is d.

Detect 3 - Back Orifice

Jun 10 08:02:17 rtr4 7911456: Jun 9 22:02:16: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.183(31338) -> xxx.yyy.128.5(31337), 1 packet
Jun 10 08:02:18 rtr4 7911457: Jun 9 22:02:17: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.183(31338) -> xxx.yyy.128.102(31337), 1 packet
Jun 10 08:02:19 rtr4 7911458: Jun 9 22:02:18: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.183(31338) -> xxx.yyy.128.241(31337), 1 packet
Jun 10 08:12:30 rtr4 7911476: Jun 9 22:12:29: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.183(31338) -> xxx.yyy.128.1(31337), 1 packet
Jun 10 08:12:31 rtr4 7911477: Jun 9 22:12:30: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.183(31338) -> xxx.yyy.128.166(31337), 1 packet
[snip]
Jun 10 19:05:18 rtr4 7912885: Jun 10 09:05:17: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.57(31338) -> xxx.yyy.128.180(31337), 1 packet
[snip]
Jun 10 19:35:33 rtr4 7912945: Jun 10 09:35:32: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.75(31338) -> xxx.yyy.128.5(31337), 1 packet
Jun 10 19:35:34 rtr4 7912946: Jun 10 09:35:33: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.42.75(31338) -> xxx.yyy.128.153(31337), 1 packet
[snip]
Jun 11 00:02:18 rtr4 7913412: Jun 10 14:02:17: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.44.27(31338) -> xxx.yyy.128.1(31337), 1 packet
Jun 11 00:02:18 rtr5 387815: Jun 10 14:02:17: %SEC-6-IPACCESSLOGP: list 102 denied udp 212.71.44.27(31338) -> xxx.yyy.128.180(31337), 1 packet

1. Source of Trace

My Network

2. Detect was generated by:

Cisco router ACL logs (syslog)

Explanation of fields:
Jun 11 00:02:18 *[date/time entry syslog'd, GMT*] **rtr4** *[router name]* **7913412:** *[entry ID]* **Jun 10 14:02:17**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog level 6]* **list 102** *[ACL]* **denied** *[action taken]* **udp** *[protocol]* **212.71.44.27(31338)** *[src address/port]* **-> xxx.yyy.128.1(31337)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Low probability. Since this is a Back Orifice attack, the hacker would want to know where he has been successful so that he can take control of the systems.

However, the attacker may be "borrowing" inactive addresses to throw off detection or forensic activity. To illustrate, the source addresses used within a two-day period are:

212.71.42.183 (5 hosts scanned on subnet xxx.yyy.128.for udp port 31337)
212.71.42.57 (1 host scanned on subnet xxx.yyy.128.for udp port 31337)
212.71.42.75 (1 hosts scanned on subnet xxx.yyy.128.for udp port 31337)

212.71.44.27 (2 hosts scanned on subnet xxx.yyy.128.for udp port 31337)

There is no DNS info for the specific addresses

Using the whois server, whois.ripe.net, the following information was provided for the owner of the addresses.

National Engineering Services and Marketing Company Ltd. (Saudi Arabia backbone and local registry address space)

4. Description of attack

This appears to be a Back Orifice attack. By default, Back Orifice runs on udp port 31337. The attacker checks only a few machines at a time (with very close time stamps) to see if this port is responding. Then the attacker waits a few hours, and checks a few more machines, this time using a slightly different source IP address, but one that is registered to the same provider.

5. Attack Mechanism

Back Orifice is a client/server application that can gather information, perform system commands, reconfigure machines, and redirect network traffic. By executing the Back Orifice server program on a machine (e.g., a victim of a Trojan Horse), a user can connect remotely to that specific IP address and perform any of the above actions. Although Back Orifice can be used as a simple monitoring tool, its main purpose is to maintain control over another machine for reconfiguration and data collection.

http://www.bo2k.com/indexwhatis.html

6. Correlations

http://cve.mitre.org -- CAN-1999-0660 (under review)
http://www.sans.org/y2k/060900.htm. (Detect posted by Daniel Holzman)
http://xforce.iss.net/alerts/advise31.php
http://www.cert.org/vul_notes/VN-98.07.backorifice.html
http://www.cert.org/summaries/CS-99-01.html,

7. Evidence of active targeting

The attacker is checking specific IP addresses in short bursts, and then waits before returning. Although the attacker might be returning to machines on which he tried to plant the BO server, this is more likely a "low and slow" reconnaissance to try to identify where BO is running.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(4 + 3) - (4 + 2) = 1

Criticality:	The dest addresses in the detect weren't critical, but some machines on that subnet are.
Lethality:	An attacker could steal sensitve data are destroy a system if BO comprimised
System:	Very hardened NT servers. Many systems weren't even the right OS for Back Orifice
Cntr-Meas	Routers have highly restrictive ACLs and block incoming connections on Port 31337. However, newer versions of BO could bypass this rule.

9. Defensive Recommendation

Although the router caught and denied connections to udp port 31337, this is insufficient with the newest release of Back Orifice: Back Orifice 2000 (bo2k). The latest version can run under udp or tcp, and the port under which it runs is user-configurable, and the data stream can be encrypted. The xforce advisory recommends searching for the following pattern to identify (and decrypt) BO2K packets:

> By looking for a series of packets that contain a 4 byte length (in little-endian byte order), followed by that length of data, you can detect all BO2k packets, regardless of the encryption used. This format is used on both the TCP and UDP transports.
>
> To decrypt the packets using the XOR encryption, XOR the 4 bytes starting at offset 4 with the value 0x3713C3CD (0xCDC31337 in little-endian order). This will give you the XOR encryption key, which is generated from the XOR key configured by the user. You can then XOR that 4 byte key with the rest of the packet -- XOR it with the 4 bytes at offset 8, 12, 16, etc.

10. Multiple Choice Question

Which is true?

a) Back Orifice always uses udp port 31337
b) Back Orifice always uses tcp port 31337
c) Back Orifice hides it's payload in icmp
d) Using src port 31337 for DNS queries can trigger false alarms

Answer: d

Detect 4 - Sunrpc

Jun 10 03:18:41 rtr5 5833: Jun 10 12:18:39: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(967) -> xxx.yyy.208.227(111), 1 packet
Jun 10 03:45:14 rtr5 5838: Jun 10 12:45:13: %SEC-6-IPACCESSLOGP: list 103 denied tcp 140.109.140.30(989) -> xxx.yyy.210.246(111), 1 packet
Jun 10 03:48:53 rtr6 343590: Jun 10 04:48:52: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(724) -> xxx.yyy.137.230(111), 1 packet
Jun 10 05:08:15 rtr5 5851: Jun 10 14:08:14: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(644) -> xxx.yyy.209.227(111), 1 packet
Jun 10 05:11:41 rtr7 2472383: Jun 10 01:11:40: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(992) -> xxx.yyy.249.235(111), 1 packet
Jun 10 05:26:00 rtr5 5864: Jun 10 14:25:59: %SEC-6-IPACCESSLOGP: list 103 denied tcp 140.109.140.30(663) -> xxx.yyy.211.246(111), 1 packet
Jun 10 05:40:15 rtr6 343991: Jun 10 06:40:14: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(820) -> xxx.yyy.138.230(111), 1 packet
Jun 10 06:53:39 rtr5 5870: Jun 10 15:53:38: %SEC-6-IPACCESSLOGP: list 103 denied tcp 140.109.140.30(742) -> xxx.yyy.210.227(111), 1 packet
Jun 10 06:56:38 rtr5 5871: Jun 10 15:56:37: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(756) -> xxx.yyy.212.246(111), 1 packet
Jun 10 07:29:18 rtr7 2473106: Jun 10 03:29:17: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(935) -> xxx.yyy.191.242(111), 1 packet
Jun 10 07:31:46 rtr6 344427: Jun 10 08:31:45: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(918) -> xxx.yyy.139.230(111), 1 packet
Jun 10 07:53:43 lo0-rt75sc03.digisle.net 1887270: Jun 10 00:53:42: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(903) -> xxx.yyy.230.212(111), 1 packet
Jun 10 08:34:45 rtr5 5892: Jun 10 17:34:44: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(856) -> xxx.yyy.213.246(111), 1 packet
Jun 10 08:36:03 rtr5 5893: Jun 10 17:36:02: %SEC-6-IPACCESSLOGP: list 103 denied tcp 140.109.140.30(840) -> xxx.yyy.211.227(111), 1 packet
Jun 10 08:38:14 rtr7 2473444: Jun 10 04:38:13: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(743) -> xxx.yyy.251.235(111), 1 packet
Jun 10 09:22:08 lo0-rt75sc03.digisle.net 1888058: Jun 10 02:22:07: %SEC-6-IPACCESSLOGP: list 103 denied tcp 140.109.140.30(1008) -> xxx.yyy.231.212(111), 1 packet
Jun 10 09:55:44 rtr5 5901: Jun 10 18:55:43: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(931) -> xxx.yyy.212.227(111), 1 packet
Jun 10 11:21:13 lo0-skil-ny.digisle.net 1632779: Jun 10 07:21:12: %SEC-6-IPACCESSLOGP: list 102 denied tcp 140.109.140.30(931) -> xxx.yyy.251.216(111), 1 packet
Jun 10 11:29:45 rtr5 5902: Jun 10 20:29:44: %SEC-6-IPACCESSLOGP: list 105 denied tcp 140.109.140.30(605) -> xxx.yyy.213.227(111), 1 packet

[snip]

1. Source of Trace

My Network

2. Detect was generated by:

Cisco router ACL logs (syslog)

Explanation of fields:
Jun 10 03:18:41 *[date/time entry syslog'd, GMT*] **rtr5** *[router name]* **5833:** *[entry ID]* **Jun 10 12:18:39**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog level 6]* **list 102** *[ACL]* **denied** *[action taken]* **tcp** *[protocol]* **140.109.140.30(967))** *[src address/port]* **-> xxx.yyy.208.227(111)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Low Probability. This appears to a be a reconnaissance scan over several subnets, and the attacker requires feedback from the scan. Additionally, tcp was used to establish a connection to the ports (111) scanned across subnets, relying on setting up a 3-way handshake.

The address is pingable, and the address range belongs to Academia Sinica in Taiwan. The machine is entered in DNS, which might possibly indicate it's a machine owned by the university, vs. just an ip address assigned to a student.

nslookup 140.109.140.30
Name: lampedusa.ihp.sinica.edu.tw
Address: 140.109.140.30

4. Description of attack

The attacker is performing reconnaissance to identify which servers are running the sunrpc service (port 111). The attack has a pattern of searching machines on *different* subnets, but favors machines whose IP address ends in .227, .230, 235, and .246. The time stamps between each query ranges between 5 minutes and 30 minutes. This time gaps and randomized pattern of scanned subnets may be a (poor) attempt to be stealthy, or indicate that the attacker is scanning machines outside of my address space during the time gaps. In either case, the attack is probably scripted.

5. Attack Mechanism

If the attacker discovers sunrpc running on a server, then the attacker can use one of several known sunrpc exploits. The Remote Procedure Call (RPC) service was developed by Sun Microsystems and is common among many UNIX systems. rpcbind and portmapper and two programs which rely on the sunrpc service.

The rpcbind program is a server that converts RPC program numbers into universal addresses. When an RPC service is started, it tells rpcbind the address at which it is listening, and the RPC program numbers it is prepared to serve. When a client wishes to make an RPC call to a given program number, it first contacts rpcbind on the server machine to determine the address where RPC requests should be sent.

By identifying if the sunrpc service is running, the attacker can delve deeper into launching specific attacks to which a system might be vulnerable.

. Correlations

The are many rpc vulnerabilities and exploits:

CVE-1999-0003 Execute commands as root via buffer overflow in Tooltalk database server (rpc.ttdbserverd

CVE-1999-0008 Buffer overflow in NIS+, in Sun's rpc.nisd program

CVE-1999-0208 rpc.ypupdated (NIS) allows remote users to execute arbitrary commands.

CVE-1999-0212 Solaris rpc.mountd generates error messages that allow a remote attacker to determine what files are on the server.

CVE-1999-0353 rpc.pcnfsd in HP gives remote root access by changing the permissions on the main printer spool directory.

CVE-1999-0493 rpc.statd allows remote attackers to forward RPC calls to the local operatin system via the SM_MON and SM_NOTIFY commands, which in turn could be used to remotely exploit other bugs such as in automountd.

CVE-1999-0687 The ToolTalk ttsession daemon uses weak RPC authentication, which allows a remote attacker to execute commands

CVE-1999-0696 Buffer overflow in CDE Calendar Manager Service Daemon (rpc.cmsd)

CVE-1999-0900 Buffer overflow in rpc.yppasswdd allows a local user to gain privileges via MD5 hash generation

CVE-1999-0974 Buffer overflow in Solaris snoop allows remote attackers to gain root privileges via GETQUOTA requests to the rpc.rquotad service.

Other helpful information can be found at:

http://www.nai.com/nai_labs/asp_set/advisory/15_solaris_rpcbind_adv.asp
http://www.cert.org/advisories/CA-99-08-cmsd.html
http://www.cert.org/advisories/CA-98.06.nisd.html
http://sunsolve.sun.com/pub-cgi/retrieve.pl?doctype=coll&doc=secbull/142&type=0&nav=sec.sba

7. Evidence of active targeting

The pattern of targeting IP address which end in .227, .230, 235, and .246. across different subnets indicates that the perpetrator is probably conducting a scripted scan across many subnets.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(3 + 5) - (4 + 5) = -1

Criticality:	Subnets rather than specific servers are scanned
Lethality:	Many dangerous rpc exploits exist
System:	Most systems are running recent OS versions. Generally rpc isn't running
Cntr-Meas	Routers have highly restrictive ACLs and block incoming connections on those port 111.

9. Defensive Recommendation

The router is currently denying connections to tcp/udp port 111. However, under Solaris 2.x, rpcbind listens not only on TCP port 111 and UDP port 111, but also on a UDP port

numbers greater than 32770. The exact number depends on the OS release and architecture.

This results in a large number of packet filters which intend to block access to rpcbind/portmapper being ineffective. Instead of sending requests to TCP or UDP port 111, the attacker simply sends them to the other UDP port. This vulnerability allows an attacker to obtain remote RPC program information even if TCP or UDP port 111 is being filtered. It can also aid an attacker to gain unauthorized access to hosts running vulnerable versions of the software. To ensure that an attacker does not attempt to exploit such vulnerablities, the following Sun patches should be applied

OS Version	Patch ID
SunOS 5.5.1	104331-02
SunOS 5.5.1_x86	104332-02
SunOS 5.5	104357-02
SunOS 5.5_x86	104358-02
SunOS 5.4	102070-03
SunOS 5.4_x86	102071-03
SunOS 5.3	102034-02

10. Multiple Choice Question

On Solaris, rpc is a service which runs on

a) TCP Port 111
b) UDP Port 111
c) TCP/UDP Ports 111 and > 32770
d) All of the above.

Answer is d.

Detect 5 - Linuxconf Probe

Jun 1 15:42:40 : Deny inbound tcp src 210.112.192.74/2222 dst xxx.xxx.xxx.0/98
Jun 1 15:42:40 : Deny inbound tcp src 210.112.192.74/2234 dst xxx.xxx.xxx.3/98
Jun 1 15:42:40 : Deny inbound tcp src 210.112.192.74/2236 dst xxx.xxx.xxx.4/98
Jun 1 15:42:40 : Deny inbound tcp src 210.112.192.74/2243 dst xxx.xxx.xxx.5/98

1. Source of Trace

http://www.sans.org/y2k/060300.htm (Daniel Holzman)

2. Detect was generated by:

The detect appears to be a firewall log, where the fields are
Jun 1 15:42:40 *[date/time entry]***: Deny** *[action taken]* **inbound** *[direction of traffic]***tcp** *[protocol]*
src 210.112.192.74/2222 *[src address/port* **dst xxx.xxx.xxx.0/98** *[dst address/port*

3. Probability the source address was spoofed

Unlikely that the source address was spoofed, as the attacker would need to know which of his targeted hosts responded to the TCP port 98 scan. Using traceroute, the last registered router shows the path through lginternet.channeli.com. An nslookup does not yield a dns entry for this specific address. The address range to which it belongs appears to be an ISP in Korea, as indicated in the following from whois.apnic.net:

inetnum:	210.112.192.0 - 210.112.192.255
netname:	CHANNELI
descr:	LG InterNet Inc.
descr:	15-28, 4th floor, Korea Investors Service Bldg.
descr:	150-010
country:	KR
admin-c:	YO4-AP
tech-c:	CM18-AP
remarks:	**ISP in Korea**
changed:	hostmast@rs.krnic.net 980714
source:	APNIC
person:	Youngdo An
address:	LG InterNet Inc.
address:	15-28, 4th floor, Korea Investors Service Building
address:	Yoido-Dong, Youngdungpo-Gu, Seoul, Korea
address:	150-010
phone:	+82 2 3773 1888
fax-no:	+82 2 3773 1814
country:	KR
e-mail:	mgr@channeli.net
nic-hdl:	YO4-AP
mnt-by:	MAINT-NULL
changed:	hostmast@krnic.net 19980702
source:	APNIC

4. Description of attack

This is a Linuxconf scan from a single source IP address to four IP address on the same Class C subnet. The close timestamps indicate this was a very rapid scan against port 98.

Linuxconf is an administration tool for the Linux operating system. The tool can be used to determine when daemons should be started, killed, etc., as well as executing many other common configuration tasks. For example, it can be used to reconfigure network interfaces, mount volumes, and modify network routes.

5. Attack Mechanism

Systems on the subnet are being scanned to see if the Linux remote configuration service is available, indicative of connection attempts to TCP port 98. Linuxconf can be used to manage remote systems and runs with root privileges. If an attacker can gain access to the Linuxconf utility on a system, he can compromise it to make powerful configuration changes.

6. Correlations

Sean Brown posted a scan originating from the same source address at http://www.sans.org/y2k/060100-1400.htm

BUGTRAQ:19991221 (Possible) Linuxconf Remote Buffer Overflow Vulnerability

More information about linuxconf and configuring securely can be found at:
http://www.redhat.com/support/manuals/RHL-6.1-Manual/ref-guide/ch-sysconfig.html
http://www.solucorp.qc.ca/linuxconf/

7. Evidence of active targeting

The intruder appears to be targeting a range of hosts on subnet in order to identify linuxconf-vulnerable systems. The close timestamps imply that this may be an initial sweep to identify vulnerable systems, rather than a return visit to actually exploit a specific system.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(3 + 4) - (3 + 5) = -1

Criticality: Subnets rather than specific servers are scanned.
Lethality: If exploited, a remote user could can root access for config changes.
System: Don't know if all system countermeasure (patches, OS version) has been applied.
Cntr-Meas The firewall is denying connection attempts.

9. Defensive Recommendation

Disable linuxconf network access using its native utility if it's not required Keep informed of latest patch levels and apply promptly. Since we've seen scans from other detects with the same source address (a very intent intruder), possibly filter on that address to block all traffic or at least send alerts.

10. Multiple Choice Question

Linuxconf is normally identified remotely by

a) udp
b) tcp
c) icmp echo request
d) icmp echo reply

Answer: b

Detect 6 - Smurf

06/08/2000 15:23:23.624 - Smurf Amplification Attack Dropped -
Source:151.4.157.1, 8, WAN - Destination:255.255.255.255, 8, LAN - -

1. Source of Trace

http://www.sans.org/y2k/061100.htm (Bill Stewart)

2. Detect was generated by:

The detect was generated by a firewall log. Although I'm not familiar with this specific log format, the relevant fields are:

06/08/2000 15:23:23.624 *[date/time entry]***- Smurf Amplification Attack** *[attack description]* **Dropped** *[action taken]***- Source:151.4.157.1,** *[src address]* **8, WAN - Destination:255.255.255.255,** *[dst address]* **8, LAN - -**

3. Probability the source address was spoofed

Likely spoofed. The source address is used by the destination in responding back. Since smurf is a denial of service attack, the attacker certainly does not want the responses directed toward his real IP address. It's interesting to note that the source address ends in ".1", which is often used for a router.

4. Description of attack

Spoofed ICMP echo request packets (commonly known as "ping" packets) are sent to IP broadcast addresses. These attacks can result in large amounts of ICMP echo reply packets being sent from an intermediary site to a victim, which can cause network congestion or outages. In the detect above, we can see that the destination address is a broadcast address

5. Attack Mechanism

As described in CERT Advisory CA-98.01, the Smurf attack is a denial of service that impacts a target network and an intermediary network. The attacker spoofs an IP address and floods the intermediary network with broadcast echo requests.

The intermediary receives an ICMP echo request packet directed to the IP broadcast address of their network. If the intermediary does not filter ICMP traffic directed to IP broadcast addresses, many of the machines on the network will receive this ICMP echo request packet and send an ICMP echo reply packet back. When (potentially) all the machines on a network respond to this ICMP echo request, the result can be severe network congestion or outages.

When the attackers create these packets, they do not use the IP address of their own machine as the source address. Instead, they create forged packets that contain the spoofed source address of the attacker's intended victim. The result is that when all the machines at the intermediary's site respond to the ICMP echo requests, they send replies to the victim's machine. The victim is subjected to network congestion that could potentially make the network unusable. Even though we have not labeled the intermediary as a "victim," the intermediary can be victimized by suffering the same bombardment that the "victim" does in these attacks.

6. Correlations

There is plenty written about smurf. Good points of reference include

CVE-1999-0513, (http://cve.mitre.org)
http://www.cert.org/advisories/CA-98.01.smurf.html
http://www.codetalker.com/whitepapers/dos-smurf.html
http://www.cisco.com/warp/public/707/22.html

7. Evidence of active targeting

By its very nature, launching a smurf attack involves actively targeting a victim and intermediary networks

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(5 + 5) - (4 + 5) = -1

Criticality:	Critical systems, including routers are affected.
Lethality:	Can bring a machine and interfere with an entire network if successful.
System:	Don't know if all system countermeasure (patches, OS countermeasures) has been applied, but the user is clearly mindful of attacks and monitoring them actively.
Cntr-Meas	The firewall is identifying and dropping smurf attempts = 5

9. Defensive Recommendation

Create icmp echo filters in the screening router or firewall and filter incoming broadcasts. On Cisco routers, disable IP directed broadcast for all interfaces on which it is not needed. This should be done on all routers in the network, not just on the border routers. The command "no ip directed-broadcast" should be applied to each interface on which directed broadcasts are to be disabled.

Preventive measure can also be taken at the system level. To prevent incoming broadcast packets from entering your network on Solaris 2.6, 2.5.1, 2.5, 2.4, and 2.3, use the command: ndd -set /dev/ip ip_forward_directed_broadcasts 0

On SunOS 4.1.3_U1 and 4.1.4:
Add ``options DIRECTED_BROADCAST=0" to system configuration file and rebuild kernel

To prevent systems from responding to broadcast ICMP packets on Solaris 2.6, 2.5.1, 2.5, 2.4, and 2.3, use the command: ndd -set /dev/ip ip_respond_to_echo_broadcast 0

10. Multiple Choice Question

Smurf attacks can be blocked
a) At border routers
b) Firewalls
c) Operating Systems
d) All of the above

Answer is d.

Detect 7 - SYN-FIN Scan

June 1 08:34:25.118877 210.196.222.18.53 > MyNet.3.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.140392 210.196.222.18.53 > MyNet.4.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.164549 210.196.222.18.53 > MyNet.5.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.174844 210.196.222.18.53 > MyNet.6.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.193856 210.196.222.18.53 > MyNet.7.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.218688 210.196.222.18.53 > MyNet.8.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.242696 210.196.222.18.53 > MyNet.9.53:
SF 907639663:907639663(0) win 1028

1. Source of Trace

Judith Ostroot's posting: http://www.sans.org/y2k/060300.htm

2. Detect was generated by:

This detect came from a windump log. The fields shown or assumed from the log are identified below.

June 1 08:34:25.118877 *[timestamp]* **210.196.222.18.53** *[src address and port]* **> MyNet.3.53:** *[dest address and prot]* **SF** *[SYN-FYN flags set]* **907639663:907639663** *[seq no.?]* **(0)** *[Number of Bytes]* **win 1028** *[TCP Window Size]*

3. Probability the source address was spoofed

Low probability that the address was spoofed. The attacker appears to be searching for responses to these unusual packets. The packets are tcp, requiring a three way handshake (although the the unusual SF flags may indicate that the attacker does not intend on really setting up a connection!)

Some whois research indicates that the address belongs to the Housing and Urban Development Corporation:

```
$ whois -h whois.nic.ad.jp 210.196.222.18/e
[ JPNIC database provides information on network administration. Its use is   ]
[ restricted to network administration purposes. For further information, use ]
[ 'whois -h whois.nic.ad.jp help'. To suppress Japanese output, add '/e' at   ]
[ the end of command, e.g. 'whois -h whois.nic.ad.jp xxx/e'.                  ]

DION (DDI CORPORATION)
        SUBA-161-NGY [Sub Allocation]                          210.196.222.0
Housing And Urban Development Corporation
             CHUBSYS [210.196.222.16 <-> 210.196.222.31]     210.196.222.16/28
```

There is a DNS entry for that specific address:
Name: dns1.udc-c.dion.ne.jp
Address: 210.196.222.18

Aliases: 18.222.196.210.in-addr.arpa

The name implies that it *might* be a nameserver, but I couldn't verify it.

4. Description of attack

In this detect, the attacker has two flags set in the packets (SYN and FIN) that would never occur naturally. Therefore the packets appear to be crafted. The sequence numbers and source port remains the same, and the timestamps are milliseconds apart. The destination hosts within the subnet are sequentially probed for destination port 53. This clearly looks like a scripted scan.

5. Attack Mechanism

This SYN-FIN attack may have occurred for two reasons.

1. The attacker may believe that this combination of flag settings will circumvent the firewall.

2. More likely, the attacker may be hoping to generate a response from the illegal packet, thereby identifying the OS. For example, Linux machines respond to SYN-FIN with SYN-FIN-ACK. This is TCP/IP stack fingerprinting. Once the attacker discovers the OS, he is one step closer to compromising the system. It's kind of like killing two birds with one stone. That is, the attacker is looking specifically for a DNS server running on a particular OS.

6. Correlations

Both QueSO and nmap may be used to send packets to the target and compare the responses with a known set of responses by OS. Also, the links below provide additional information.

http://www.insecure.org/nmap/nmap-fingerprinting-article.html
http://www.securityfocus.com/templates/archive.pike?list=1&msg=Pine.LNX.3.96.980710165820.1382C-100000@think.kung.foo

7. Evidence of active targeting

The attacker is targeting hosts on a specific subnet sequentially. This is infomation gathering, and the attacker might try to return later to exploit vulnerabilities.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(5 + 3) - (1 + 1) = 6

Criticality:	DNS servers
Lethality:	If OS is disovered, exploits might be tried. But we haven't reached this stage yet.
System:	There is no indication that the packets were stopped. Systems might have responded (this isn't my detect, so I can't accurately gage this)

Cntr-Meas Again, there is no indication that the packets were stopped. Systems might have responded.

9. Defensive Recommendation

Ensure that a firewall is blocking tcp access to port 53 regardless of the flag settings, as this will help prevent a zone transfer on a misconfigured DNS server. Packets with illegal flag settings (to any port) should be denied to prevent OS finger printing.

10. Multiple Choice Question

Which is legal for a tcp packet?

a) PSH is set and ACK is not set
b) URG is set and ACK is not set
c) ACK is set and a reserved bit is set
d) FIN is set and ACK is set

Answer is d.

Detect 8 - Deep Throat Scan

May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.1/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.2/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.3/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.4/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.5/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.6/2140
May 31 00:36:51 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.7/2140
May 31 00:37:02 Deny inbound udp src 213.6.30.204/60000 dst xxx.xxx.xxx.255/2140

1. Source of Trace

Daniel Holzman's posting at http://www.sans.org/y2k/060100.htm

2. Detect was generated by:

A firewall log, although I'm sure exactly which brand of firewall. The relevant fields are

May 31 00:36:51 *[timestamp]* **Deny** *[action taken]* **inbound** *[direction of packet]* **udp** *[protocol]* **src 213.6.30.204/60000** *[src address/port]* **dst xxx.xxx.xxx.1/2140** *[dst address/port]*

3. Probability the source address was spoofed

Unlikely that the source was spoofed. The pattern suggests that the attacker is on a reconnaissance mission to identify machines running deep throat, so the attacker needs the information to get back to him/her. The address is registered in Germany.

An nslookup on the source address yields:
Name: A1ecc.pppool.de
Address: 213.6.30.204

whois.ripe.net provides the following, indicating that the address might have been assigned to an access server (as) / dialpool (making it even more unlikely that the address was spoofed.)

inetnum: 213.6.0.0 - 213.6.251.255
netname: MOBILCOM-CITYLINE-NET
descr: MobilCom City LINE GmbH
country: DE
admin-c: DRR11-RIPE
tech-c: DRR11-RIPE
tech-c: DRR11-RIPE
status: ASSIGNED PA
remarks: please report spam/abuse to: abuse@pppool.de
mnt-by: ROKA-MNT
changed: tech-c@pppool.de 20000112
changed: tech-c@pppool.de 20000313
source: RIPE

route: 213.6.0.0/16

descr: MobilCom City LINE GmbH **Dialpool**
origin: AS5430
notify: **as**-guardian@roka.net
mnt-by: ROKA-MNT
changed: jens@roka.net 19991027
source: RIPE

4. Description of attack

The attacker is searching for machines responding on udp port 2140. The close time stamps and sequential order of addresses scanned suggest that this is a scripted/automated task. The source address and source port are the same for each packet. The unchanging source port (60000) is the normal source port for the deep throught client.

5. Attack Mechanism

Deep Throat is a Trojan Horse program that consists of a client program called "Deep Throat Remote Control" which is run on a remote computer to gain access to any computer connected to a TCP/IP network. An executable server program is required to be installed on the victim's computer to permit the remote site access to the victim's computer in a manner similar to Netbus, Back Orifice and other internet "Remote administration" trojan horses. This program exploits security vulnerabilities in the Windows95 and Windows98 platforms. There are reports that an NT version of Deep Throat is being worked on.

Because the Deep Throat trojan has an FTP server built in, once it does find its way onto a computer, the perpetrator is able to download files from any portion of the compromised computer. The Deep Throat trojan allows the perpetrator to "ping" a subnetwork looking for other machines which may contain this trojan and Deep Throat also provides the means to invoke a program remotely on the victim's machine and then extract files which may have been created or modified by the remotely controlled program.

6. Correlations

http://www.sans.org/y2k/DT.htm
http://www.commodon.com/threat/threat-dt2.htm
http://www.nsclean.com/psc-dt.html
http://advice.networkice.com/advice/phauna/rats/deep_throat/default.htm

7. Evidence of active targeting

The attacker is targeting hosts on a specific subnet sequentially. This is infomation gathering, and the attacker might try to return later to remotely access computers found to be running the Deep Throat server..

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(1 + 3) - (3 + 5) = -4

Criticality: Both versions 1 and 2 run only on Win9x platforms (so far)
Lethality: Attacker could use compromised machine to remotely scan the subnet, or transfer sensitive files off the machine.

System: Identify the Trojan is heavily dependent on running anti-virus or checksum software.
Cntr-Meas The firewall successfully denied udp packets to port 2140.

9. Defensive Recommendation

Deep Throat V2 has been released, and uses TCP port 6670, UDP Port 2140 and UDP port 3150 to establish its connection between the "client" and "server". These ports should be blocked by the firewall or border router.

Additionally, anti-virus or checksum software should pay special attention to the systray.exe program in Windows. Trojan deletes the existing "systray.exe," which is normally 36kb in size, and replaces it with the Deep Throat v2 "server", which is approximately 301kb in size.

10. Multiple Choice Question

Another port that Deep Throat uses is:

a) TCP 5631
b) UDP 5632
c) TCP 6670
d) UDP 22

Answer is c.

Detect 9 - pcAnywhere Probes

Jun 13 13:48:43 rtr6 1660608: Jun 13 09:48:42: %SEC-6-IPACCESSLOGP: list 102 denied udp 24.18. 212.119(1113) -> xxx.yyy.248.153(5632), 1 packet
Jun 13 13:57:56 rtr6 1660710: Jun 13 09:57:55: %SEC-6-IPACCESSLOGP: list 102 denied udp 24.18. 212.119(1117) -> xxx.yyy.248.153(5632), 1 packet
Jun 13 15:11:35 rtr7 2513457: Jun 13 11:11:34: %SEC-6-IPACCESSLOGP: list 102 denied udp 2 4.18.212.119(1136) -> xxx.yyy.248.154(5632), 1 packet
Jun 13 15:18:26 rtr7 2513586: Jun 13 11:18:25: %SEC-6-IPACCESSLOGP: list 102 denied udp 2 4.18.212.119(1140) -> xxx.yyy.248.154(5632), 1 packet
Jun 13 15:20:48 rtr7 2513609: Jun 13 11:20:46: %SEC-6-IPACCESSLOGP: list 102 denied udp 2 4.18.212.119(1142) -> xxx.yyy.248.154(5632), 1 packet
Jun 13 15:49:39 rtr7 2514043: Jun 13 11:49:38: %SEC-6-IPACCESSLOGP: list 102 denied udp 2 4.18.212.119(1149) -> xxx.yyy.248.78(5632), 1 packet
Jun 13 15:53:07 rtr7 2514108: Jun 13 11:53:06: %SEC-6-IPACCESSLOGP: list 102 denied udp 2 4.18.212.119(1153) -> xxx.yyy.248.152(5632), 1 packet
Jun 13 16:07:22 rtr6 1662116: Jun 13 12:07:21: %SEC-6-IPACCESSLOGP: list 102 denied udp 24.18. 212.119(1159) -> xxx.yyy.248.153(5632), 1 packet
Jun 13 17:45:40 rtr7 2515827: Jun 13 13:45:39: %SEC-6-IPACCESSLOGP: list 102 denied udp 24.18.212.119(1068) -> xxx.yyy.248.152(5632), 1 packet

1. Source of Trace

My Network

2. Detect was generated by:

Cisco router ACL logs (syslog)

Explanation of fields:

Jun 13 13:48:43 *[date/time entry syslog'd, GMT*] **rtr6** *[router name]* **1660608:** *[entry ID]* **Jun 13 09:48:42**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog level 6]* **list 102** *[ACL]* **denied** *[action taken]* **udp** *[protocol]* **24.18.212.119(1113)** *[src address/port]* **-> xxx.yyy.248.153(5631)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Unlikely that the source address was spoofed. The intruder was a cable modem user who was either trying to establish a connection via pcAnywhere (at which point tcp must be used), or was searching for pcAnywhere hosts, in which case he/she would want the information to get back to him/her.

An nslookup on the source address yields:

Name: cc712460-c.union1.nj.home.com
Address: 24.18.212.119

4. Description of attack

Over a period of approx. four hours, several connection attempts were made from the intruder's address to udp port 5632 on four different internal hosts. Udp port 5632 is used by Symantec's pcAnywhere ver. 7.52 and higher. pcAnywhere is a Windows-based remote control software package that gives a remote user GUI console access.

5. Attack Mechanism

PCAnywhere sets up a connection by first contacting the target machine using UDP with a random source port and a destination port of 5632. (For backwards compatibility, if 5632 doesn't work, it will attempt udp port 22).

Once that has been established, it will then attempt an outbound connection to port 5631 via TCP.

Attackers scan the Internet looking for machines supporting this product. Many users use empty passwords or passwords that are easy to guess (like the word "password"). This will provide easy-entry for the hacker. If hackers gain control over the machine, not only can they steal information on that machine, they can use it to attack yet other machines on the Internet.

6. Correlations

This topic was discussed in Steve Northcutt's GIAC Network Intrustion Class in San Jose (May 11-12, 2000). References on page 260 illustrate this type of attack.

Other references can be found at:

http://www.symantec.com
http://advice.networkice.com/advice/intrusions/2001507/default.htm

7. Evidence of active targeting

According to the symantec literature, the pcAnywhere client will attempt connections to udp port 5632 on it's local subnet in order to identify pcAnywhere hosts. This can lead to false positives.

In this detect, however, the intruder certainly is not on the local subnet. Specific hosts have been targeted.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(4 + 4) - (4 + 5) = -1

Criticality: These machines are web servers affecting customer business
Lethality: At least one has the software loaded.
System: Extensive measures to prevent unauthorized remote access are taken.
Cntr-Meas The firewall successfully denied udp packets to port 5632.

9. Defensive Recommendation

Don't run the service unless it's necessary. For example, it's possible to ssh into an NT server and run a "net start" command so that the service isn't always running. Disable it entirely if not needed.

In addition to restricting udp port 5632 at the firewall and router, restrict the alternate ports used by pcAnywhere (tcp 5631, udp 22).

As an added measure, you can alter the default ports that pcAnywhere uses by modifying the following registry keys on the

[HKEY_LOCAL_MACHINE\SOFTWARE\Symantec\pcAnywhere\CurrentVersion\System]
"TCPIPDataPort"=dword:000015ff (hex value of tcp port)
"TCPIPStatusPort"=dword:00001600 (hex value of udp port)

10. Multiple Choice Question

Ports used by pcAnywhere include

a) tcp 22
b) udp 22
c) udp 5631
d) udp 37331

Answer is B.

Detect 10 - Remote Shell Connection Attempts

Jun 7 22:56:16 rtr8 289908: Jun 7 15:56:15: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209. 184.88.49(603) -> xxx.yyy.193.160(514), 1 packet
Jun 7 22:56:28 rtr8 289919: Jun 7 15:56:27: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(603) -> xxx.yyy.193.160(514), 1 packet
Jun 7 22:56:52 rtr8 289934: Jun 7 15:56:51: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209. 184.88.49(603) -> xxx.yyy.193.160(514), 1 packet
Jun 7 22:57:13 rtr8 289950: Jun 7 15:57:12: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(603) -> xxx.yyy.193.160(514), 1 packet
Jun 7 23:25:09 rtr8 290295: Jun 7 16:25:08: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209. 184.88.49(651) -> xxx.yyy.193.160(514), 1 packet
Jun 7 23:25:55 rtr8 290331: Jun 7 16:25:54: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(651) -> xxx.yyy.193.160(514), 1 packet
Jun 7 23:30:03 rtr9 3088002: Jun 7 16:30:01: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(676) -> xxx.yyy.193.163(514), 1 packet
Jun 7 23:31:34 rtr9 3088085: Jun 7 16:31:33: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(689) -> xxx.yyy.193.181(514), 1 packet
Jun 7 23:32:40 rtr8 290732: Jun 7 16:32:39: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(53232) -> xxx.yyy.193.174(514), 1 packet
Jun 7 23:34:21 rtr8 290844: Jun 7 16:34:20: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(721) -> xxx.yyy.193.184(514), 1 packet
Jun 7 23:34:30 rtr8 290850: Jun 7 16:34:29: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(715) -> xxx.yyy.193.184(514), 1 packet
Jun 7 23:34:39 rtr8 290856: Jun 7 16:34:38: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(715) -> xxx.yyy.193.184(514), 1 packet
Jun 7 23:34:42 rtr8 290857: Jun 7 16:34:41: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(721) -> xxx.yyy.193.184(514), 1 packet
Jun 7 23:36:23 rtr8 290892: Jun 7 16:36:22: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(731) -> xxx.yyy.193.176(514), 1 packet
Jun 7 23:36:41 rtr8 290900: Jun 7 16:36:40: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(731) -> xxx.yyy.193.176(514), 2 packets
Jun 7 23:37:17 rtr9 3088407: Jun 7 16:37:16: %SEC-6-IPACCESSLOGP: list 102 denied tcp 209.184.88.49(737) -> xxx.yyy.193.161(514), 1 packet

1. Source of Trace

My Network

2. Detect was generated by:

Jun 7 22:56:16 *[date/time entry syslog'd, GMT*] **rtr8** *[router name]* **289908:** *[entry ID]* **Jun 7 15:56:15**: *[date/time of router action, local TZ]* **%SEC-6-IPACCESSLOGP**: *[securty violation, syslog level 6]* **list 102** *[ACL]* **denied** *[action taken]* **tcp** *[protocol]* **209.184.88.49(603)** *[src address/port]* **-> xxx.yyy.193.160(514)**, *[dst address/port]* **1 packet**

3. Probability the source address was spoofed

Unlikely. This was a tcp connection attempt. An nslookup yields:

Name: 209-184-88-49.alignsc.com

Address: 209.184.88.49

Checking the arin whois database indicates that this address is owned by Alignment Solutions, which appears to be either a customer or division of Southwestern Bell

4. Description of attack

Over a period of approx. 40 minutes, 14 connections attempts to tcp port 514 were identified. They originated from a single address, and 7 computers with non-contiguous addresses were the target. Connection attempts ranged from several seconds to several minutes.

5. Attack Mechanism

TCP port 514 is associated with the remote shell (rsh) command. (not to be confused with UDP 514, associated with syslogd). The remote shell command is known as one of the Berkeley "r" commands that enable remote execution of a program. Improperly configured UNIX systems can have trust relationship settings that allow an unauthorized user to remotely execute a command on that system. Programmers and system administrators often set trust relationships too lax in order to facilitate their work. .rhosts and hosts.equiv are files that dictate which machines and remote users may execute commands remotely, But they can easily be spoofed and their use is discouraged.

Reviewing the logs, it's impossible to tell if the intruder was simply "checking" if the service rsh was running, or if he/she was actually trying to execute commands remotely. At least two attempts were made before moving on to the next address.

6. Correlations

CVE-1999-0180 (http://cve.mitre.org), in.rshd allows users to login with a NULL username and execute commands

7. Evidence of active targeting

Specific, non-sequential addresses were targeted.

8. Severity

(Critical + Lethal) – (System + Net Countermeasures) = Severity
(4 + 3) - (5 + 5) = -3

Criticality: These machines are web servers affecting customer business
Lethality: It's possible to exploit rsh vulnerabilities and damage a system, but difficult.
System: The systems have new OS's, patches, and are not running the rsh daemon
Cntr-Meas The firewall successfully denied tcp packets to port 514.

9. Defensive Recommendation

The router was successfully denying packets, and the rsh service isn't even running on the hosts. Measures are currently satisfactory

10. Multiple Choice Question

UDP 514 is associated with

a) rlogin
b) syslog
c) remote shell
d) imap

Answer is b.