GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Interested in learning more?

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

GIAC Certified Intrusion Analyst PRACTICAL COMPONENT

*** Northcutt, I like the write up on capture 3 and the subnet mask in capture 7 would make a great test question, love it, overall this could benefit from a formal analysis process. Might want to take another look at capture 2 and 9 again. 75 **

Ray Johnston

Capture 1

Time	Source	SrcPort	Dest	DestPort	Flg	Begin seq#	end seq#	bytes	options
00:06.4	foreignhost	3002	> myclassc.2	111:	S	2112342080:	2112342080	(0)	win 32120 (DF)
00:06.4	foreignhost	3001	> myclassc.1	111:	S	2107900681:	2107900681	(0)	win 32120 (DF)
00:06.4	foreignhost	3004	> myclassc.4	111:	S	2109761710:	2109761710	(0)	win 32120 (DF)
00:06.4	foreignhost	3003	> myclassc.3	111:	S	2102386733:	2102386733	(0)	win 32120 (DF)
00:06.4	foreignhost	3003	> myclassc.3	111:	S	2102386733:	2102386733	(0)	win 32120 (DF)
00:06.4	foreignhost	3002	> myclassc.2	111:	S	2112342080:	2112342080	(0)	win 32120 (DF)
00:06.4	foreignhost	3001	> myclassc.1	111:	S	2107900681:	2107900681	(0)	win 32120 (DF)
00:06.4	foreignhost	3004	> myclassc.4	111:	S	2109761710:	2109761710	(0)	win 32120 (DF)
00:06.4	foreignhost	3005	> myclassc.5	111:	S	2102077764:	2102077764	(0)	win 32120 (DF)
00:06.4	foreignhost	3006	> myclassc.6	111:	S	2116656821:	2116656821	(0)	win 32120 (DF)
00:06.4	foreignhost	3005	> myclassc.5	111:	S	2102077764:	2102077764	(0)	win 32120 (DF)
00:06.4	foreignhost	3006	> myclassc.6	111:	S	2116656821:	2116656821	(0)	win 32120 (DF)
00:06.4	foreignhost	3007	> myclassc.7	111:	S	2112801082:	2112801082	(0)	win 32120 (DF)
00:06.4	foreignhost	3007	> myclassc.7	111:	S	2112801082:	2112801082	(0)	win 32120 (DF)
00:06.4	foreignhost	3008	> myclassc.8	111:	S	2110966650:	2110966650	(0)	win 32120 (DF)
00:06.4	foreignhost	3008	> myclassc.8	111:	S	2110966650:	2110966650	(0)	win 32120 (DF)
00:06.4	foreignhost	3010	> myclassc.10	111:	S	2117810620:	2117810620	(0)	win 32120 (DF)
00:06.4	foreignhost	3009	> myclassc.9	111:	S	2112599285:	2112599285	(0)	win 32120 (DF)
00:06.4	foreignhost	3010	> myclassc.10	111:	S	2117810620:	2117810620	(0)	win 32120 (DF)
00:06.4	foreignhost	3009	> myclassc.9	111:	S	2112599285:	2112599285	(0)	win 32120 (DF)

This detect appears to be automated SYN scan of a class C - myclassc.x on port 111. Port 111 is the RPC port and this is probably an attacker looking for machines that respond on port 111 to identify them as machines which may be open to various RPC attacks. Due to the speed of this attack it is most likely an automated scan and one in which the attacker is not concerned with Stealth (or doesn't know any better).

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

Initially, I felt that this was not a crafted packet, but after closer review I feel it is a crafted packet. If you notice the source port numbers increment in relation to the destination address - 3001 to myclassc.1, 3002 to myclassc.2, 3003 to myclassc.3, etc. These items are definitely a signature of the tool being used in the scan.

Capture 2

Time	Source	SrcPort	Dest	DestPort	Flg	Begin seq#	end seq#			
11:06.7	foreignhost.auth	>	1.2.3.25.1098:		S	14726994:14726994	(0)	ack	674711610	win 32736
11:21.3	foreignhost.auth	>	1.2.3.3.1243:		S	4987450:4987450	(0)	ack	674711610	win 32736
11:54.1	foreignhost.auth	>	172.16.1.9.1473:		S	5224050:5224050	(0)	ack	674711610	win 32736
11:56.1	foreignhost.auth	>	192.168.1.40.1098:		S	7997074:7997074	(0)	ack	674711610	win 32736
12:11.3	foreignhost.auth	>	192.168.2.18.1243:		S	10470410:10470410	(0)	ack	674711610	win 32736
12:22.8	foreignhost.auth	>	10.1.1.37.1142:		S	9958906:9958906	(0)	ack	674711610	win 32736
12:37.5	foreignhost.auth	>	10.1.2.15.1287:		S	11821098:11821098	(0)	ack	674711610	win 32736
14:00.1	foreignhost.auth	>	10.1.3.68.1142:		S	2216938:2216938	(0)	ack	674711610	win 32736
15:36.6	foreignhost.auth	>	192.168.3.19.1098:		S	8960346:8960346	(0)	ack	674711610	win 32736
16:57.8	foreignhost.auth	>	10.1.4.39.1287:		S	14043050:14043050	(0)	ack	674711610	win 32736
17:07.7	foreignhost.auth	>	192.168.4.57.1243:		S	5809858:5809858	(0)	ack	674711610	win 32736
17:32.2	foreignhost.auth	>	192.168.4.95.1098:		S	11842658:11842658	(0)	ack	674711610	win 32736
18:20.0	foreignhost.auth	>	192.168.5.110.1098:		S	16654178:16654178	(0)	ack	674711610	win 32736
19:06.9	foreignhost.auth	>	172.16.2.93.1473:		S	8678450:8678450	(0)	ack	674711610	win 32736
19:09.0	foreignhost.auth	>	192.168.6.125.1098:		S	9689562:9689562	(0)	ack	674711610	win 32736
19:55.9	foreignhost.auth	>	172.16.3.109.1473:		S	11594466:11594466	(0)	ack	674711610	win 32736

This could be a scan for the Trojan SUB7.2. I believe this to be true due to the destination port of 1243 on host machines 1.2.3.3, 192.168.2.18, 192.168.4.57. Port 1243 is the default port for SUB7.2. It is also abnormal to see a SYN/ACK without first seeing a SYN that would have elicited the response ("ALL RESPONSE and NO STIMULI"). This points to a Stealthy host discovery in which the packets have been crafted to attempt to pass through router ACLs and to avoid detection by analysts.

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

Capture 3

```
Time                Source          SrcPort          Dest            DestPort          Bytes
13:25:21.182824 192.168.111.164.1027 > SVRLOC.MCAST.NET.427: udp 138
13:25:21.306509 192.168.111.164.1027 > SVRLOC.MCAST.NET.427: udp 90
13:25:21.479862 192.168.111.111.1027 > SVRLOC.MCAST.NET.427: udp 90
13:25:21.795723 192.168.111.101.1027 > SVRLOC.MCAST.NET.427: udp 138
13:25:21.989945 192.168.111.101.1027 > SVRLOC.MCAST.NET.427: udp 90
13:25:22.183933 192.168.111.164.1027 > SVRLOC.MCAST.NET.427: udp 90
13:25:22.345969 192.168.139.38.1027 > SVRLOC.MCAST.NET.427: udp 90
```

I pulled this one off the GIAC website. Something about it caught my eye. It appears that several internal clients are trying to reach the host SVRLOC.MCAST.NET on port 427. Based on this I have a question, is SVRLOC.MCAST.NET a SCO Unix machine? If so this may very well be an example of a distributed attack using his local machines (192.168.x.x) to exploit a symlink vulnerability that exist in SCO OpenServer 5.0 - 5.0.5. This vulnerability can create instances where any files that are group 'auth' writeable may be overwritten.

Capture 4

```
Time      Source                  SrcPort    Dest          DestPort    Flg    Begin seq#  End seq#
51:04.7 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.164.100.1142:          S  4079879590:4079879590  (0)  ack 674711610 win 16384 (DF)
51:07.5 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.164.100.1142:          S  4079879590:4079879590  (0)  ack 674711610 win 16384 (DF)
51:15.9 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.164.100.1142:          R  4079879591:4079879591  (0)  ack 674711610 win 16384 (DF)
52:59.8 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.168.48.1901: S  1635931563:1635931563  (0)  ack 674711610 win 16384 (DF)
54:39.5 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.133.20.1119: S  436688995:436688995    (0)  ack 674711610 win 16384 (DF)
54:39.9 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.133.20.1119: R  436688996:436688996    (0)  ack 674711610 win 16384 (DF)
00:44.9 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.76.10.1607:            S  2475552389:2475552389  (0)  ack 674711610 win 16384 (DF)
00:45.1 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.76.10.1607:            R  2475552390:2475552390 (0)  ack 674711610 win 16384 (DF)
05:38.0 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.206.66.1273: S  2620989873:2620989873  (0)  ack 674711610 win 16384 (DF)
05:38.1 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.206.66.1273:           R  2620989874:2620989874 (0)  ack 674711610 win 16384 (DF)
06:33.0 dont.tell.anyone.im.an.xconvict.com.telnet > 1.2.152.83.1612:            S  4290171879:4290171879 (0)  ack 674711610 win 16384 (DF)
06:33.1 dont.tell.anyone.im.an.xconvict.com.telnet > 1.2.152.83.1612:            R  4290171880:4290171880 (0)  ack 674711610 win 16384 (DF)
07:12.5 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.224.109.1890:          S  1228683617:1228683617 (0)  ack 674711610 win 16384 (DF)
07:12.7 dont.tell.anyone.im.an.xconvict.com.telnet > 10.1.224.109.1890:          R  1228683618:1228683618 (0)  ack 674711610 win 16384 (DF)
```

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

My first impression is that this is a stealth SYN-ACK scan and a RESET scan combined. I am lead to believe a RESET scan is occurring due to the fixed acknowledgment number 674711610. The use of the RESET package may cause a router to reply with an unreachable. Through this the attacker could assume that addresses that did not result in an unreachable exist. The SYN-ACK package should cause live hosts to respond with a RESET of their own.

Capture 5

```
Time            Source     SrcPort     Dest   DestPort  udp
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33465      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33466      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33467      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33468      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33469      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33470      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33471      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33472      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33473      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33474      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33475      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33476      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33477      UDP
Mar 31 11:27:43 208.185.54.22:33161 -> a.b.c.34:33478      UDP
```

Here's another one I pulled from the GIAC website. This is a udp scan of Traceroute ports from a single source machine. This could be an example of some type of load balancing but, I honestly can't say for sure.

Capture 6

Mar 31 12:34:44 myhost portsentry[173]: attackalert: Connect from host: user-33qs1hs.dialup.mindspring.com/199.174.6.60 to UDP port: 31337
Mar 31 12:35:10 myhost2 portsentry[8311]: attackalert: Connect from host: user-33qs1hs.dialup.mindspring.com/199.174.6.60 to UDP port: 31337

Again, pulled from the GIAC website. Here we see the destination port of 31337. Port 31337 is the default port for the Back Orifice Trojan. This detect could be an attempt to exploit the BO Trojan. However, port 31337 could also be a valid ephemeral port so alerts based on the attempt to connect to 31337 should be reviewed with some caution. In this instance, I feel that it is most likely an attempt to find a machine that has the BO Trojan running on it. The attacker attempted to connect to two machines within a second of each other, trying to reach port 31337 on both.

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

Capture 7

15:21.2 yoda.formysite.com > 1.2.190.24:	icmp: address mask is 0xfffffe00
21:14.6 yoda.formysite.com > 1.2.149.110:	icmp: address mask is 0xfffffe00
24:00.1 yoda.formysite.com > 134.78.9.57:	icmp: address mask is 0xfffffe00
27:04.3 yoda.formysite.com > 1.2.194.3:	icmp: address mask is 0xfffffe00
34:22.2 yoda.formysite.com > 134.78.228.45:	icmp: address mask is 0xfffffe00
36:15.7 yoda.formysite.com > 1.2.181.48:	icmp: address mask is 0xfffffe00
37:04.7 yoda.formysite.com > 1.2.185.32:	icmp: address mask is 0xfffffe00
53:40.6 yoda.formysite.com > 1.2.231.114:	icmp: address mask is 0xfffffe00
57:58.7 yoda.formysite.com > 10.1.8.54:	icmp: address mask is 0xfffffe00
59:11.4 yoda.formysite.com > myhost:	icmp: address mask is 0xfffffe00

This detect appears to be a broadcast scan based on the mask of 23 bits. The attacker could be looking to determine the broadcast for these targets or he is aware of the target's subnet mask and is throwing out a broadcast to several of the subnets. If he is attempting to determine the broadcast I would expect to see additional traces with different masks in the near future.

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

Capture 8

40:00.3 foreignhost1.1095 > 1.2.235.148.20932:	S 2802696:2802696	(0)	win 8192 (DF)
40:03.2 foreignhost1.1095 > 1.2.235.148.20932:	S 2802696:2802696	(0)	win 8192 (DF)
40:09.3 foreignhost1.1095 > 1.2.235.148.20932:	S 2802696:2802696	(0)	win 8192 (DF)
40:21.3 foreignhost1.1095 > 1.2.235.148.20932:	S 2802696:2802696	(0)	win 8192 (DF)
40:49.6 foreignhost1.1096 > 1.2.235.148.20932:	S 2852091:2852091	(0)	win 8192 (DF)
40:52.6 foreignhost1.1096 > 1.2.235.148.20932:	S 2852091:2852091	(0)	win 8192 (DF)
40:58.6 foreignhost1.1096 > 1.2.235.148.20932:	S 2852091:2852091	(0)	win 8192 (DF)
41:10.6 foreignhost1.1096 > 1.2.235.148.20932:	S 2852091:2852091	(0)	win 8192 (DF)
41:22.7 foreignhost1.1097 > 1.2.235.148.20932:	S 2885186:2885186	(0)	win 8192 (DF)
41:25.7 foreignhost1.1097 > 1.2.235.148.20932:	S 2885186:2885186	(0)	win 8192 (DF)
41:31.7 foreignhost1.1097 > 1.2.235.148.20932:	S 2885186:2885186	(0)	win 8192 (DF)
41:43.7 foreignhost1.1097 > 1.2.235.148.20932:	S 2885186:2885186	(0)	win 8192 (DF)
26:09.9 foreignhost2.1048 > 1.2.235.148.14874:	S 180334854:180334854	(0)	win 8192 (DF)
26:13.2 foreignhost2.1048 > 1.2.235.148.14874:	S 180334854:180334854	(0)	win 8192 (DF)
26:19.8 foreignhost2.1048 > 1.2.235.148.14874:	S 180334854:180334854	(0)	win 8192 (DF)
26:32.9 foreignhost2.1048 > 1.2.235.148.14874:	S 180334854:180334854	(0)	win 8192 (DF)
26:58.5 foreignhost2.1049 > 1.2.235.148.14874:	S 902354562:902354562	(0)	win 8192 (DF)
27:01.6 foreignhost2.1049 > 1.2.235.148.14874:	S 902354562:902354562	(0)	win 8192 (DF)
27:07.9 foreignhost2.1049 > 1.2.235.148.14874:	S 902354562:902354562	(0)	win 8192 (DF)
27:15.3 foreignhost2.1052 > 1.2.235.148.14874:	S 649914605:649914605	(0)	win 8192 (DF)
27:18.6 foreignhost2.1052 > 1.2.235.148.14874:	S 649914605:649914605	(0)	win 8192 (DF)
27:21.0 foreignhost2.1049 > 1.2.235.148.14874:	S 902354562:902354562	(0)	win 8192 (DF)
27:25.2 foreignhost2.1052 > 1.2.235.148.14874:	S 649914605:649914605	(0)	win 8192 (DF)
27:35.9 foreignhost2.1053 > 1.2.235.148.14874:	S 1728257859:1728257859	(0)	win 8192 (DF)

This detect could be a coordinated SYN flood attempt. Both source machines are targeting the same destination machine. Each source machine istargeting a different destination port, but both are sending several packets a minute at the targeted destination port.

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

Capture 9

04:16.9 bigcompanyserver.com.2400 > myns1.53: S 1839919940:1839920004(64) win 2048
04:16.9 bigcompanyserver.com.2400 > myns1.53: S 1839919940:1839920004(64) win 2048
04:16.9 bigcompanyserver.com.2402 > myns1.53: S 561709080:561709144(64) win 2048
04:16.9 bigcompanyserver.com.2402 > myns1.53: S 561709080:561709144(64) win 2048
04:16.9 bigcompanyserver.com.2401 > myns1.53: S 753842643:753842707(64) win 2048
04:16.9 bigcompanyserver.com.2401 > myns1.53: S 753842643:753842707(64) win 2048
05:08.6 bigcompanyserver.com.2400 > myns2.53: S 142380666:142380730(64) win 2048
05:08.6 bigcompanyserver.com.2400 > myns2.53: S 142380666:142380730(64) win 2048
05:08.6 bigcompanyserver.com.2402 > myns2.53: S 1430682108:1430682172(64) win 2048
05:08.6 bigcompanyserver.com.2402 > myns2.53: S 1430682108:1430682172(64) win 2048
05:08.6 bigcompanyserver.com.2401 > myns2.53: S 1607270437:1607270501(64) win 2048
05:08.6 bigcompanyserver.com.2401 > myns2.53: S 1607270437:1607270501(64) win 2048
45:09.3 bigcompanyserver2.com.2300 > myns1.53: S 1840811591:1840811655(64) win 2048
45:09.3 bigcompanyserver2.com.2300 > myns1.53: S 1840811591:1840811655(64) win 2048
45:09.3 bigcompanyserver2.com.2301 > myns1.53: S 947326486:947326550(64) win 2048
45:09.3 bigcompanyserver2.com.2301 > myns1.53: S 947326486:947326550(64) win 2048
45:09.3 bigcompanyserver2.com.2302 > myns1.53: S 1917739764:1917739828(64) win 2048
45:09.3 bigcompanyserver2.com.2302 > myns1.53: S 1917739764:1917739828(64) win 2048
45:45.3 bigcompanyserver2.com.2100 > myns2.53: S 1166290998:1166291062(64) win 2048
45:45.3 bigcompanyserver2.com.2100 > myns2.53: S 1166290998:1166291062(64) win 2048
45:45.3 bigcompanyserver2.com.2102 > myns2.53: S 754822080:754822144(64) win 2048

The full detect runs for over 5 hours with the same two machines within bigcompany's network sending 6 SYN packets each. This is a detect that I would like to have the full dump of. It is a case where two source machines from the same network are both hitting the same two DNS server servers. At first glance I thought this might be a case of load balancing but there are a few items that caused me to discredit this. The first is that these are TCP connections and the only cases I have seen of load balancing involve UDP. While this does not rule out the possibility, it does create a reasonable doubt in my mind. The second is if you notice the SYN packets contain a payload of 64 bytes - very interesting/alarming. I feel that this is more likely a case of a Denial of Service attack and that the payload could be designed to create a buffer overflow. Then there is the duration of time that the six packets arrive in. All six hit within a second which is very similar to some DoS attack signatures I have seen.

Capture 10

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

25:42.5 notmyhost1.net.44080 > 10.1.60.101.33466: udp 12
25:47.5 notmyhost1.net.44080 > 10.1.60.101.33467: udp 12
25:51.8 notmyhost2.com.43871 > 10.1.60.101.33468: udp 10 [ttl 1]
25:52.5 notmyhost1.net.44080 > 10.1.60.101.33468: udp 12
25:56.8 notmyhost2.com.43871 > 10.1.60.101.33469: udp 10 [ttl 1]
25:57.5 notmyhost1.net.44080 > 10.1.60.101.33469: udp 12
26:00.0 notmyhost4.net.39346 > 10.1.60.101.33466: udp 12 [ttl 1]
26:01.9 notmyhost2.com.43871 > 10.1.60.101.33470: udp 10 [ttl 1]
26:02.5 notmyhost1.net.44080 > 10.1.60.101.33470: udp 12
26:02.7 notmyhost3.com.62179 > 10.1.60.101.33483: udp 14 [ttl 1]
26:05.1 notmyhost4.net.39346 > 10.1.60.101.33467: udp 12 [ttl 1]
26:07.0 notmyhost2.com.43871 > 10.1.60.101.33471: udp 10
26:07.5 notmyhost1.net.44080 > 10.1.60.101.33471: udp 12
26:07.7 notmyhost3.com.62179 > 10.1.60.101.33484: udp 14 [ttl 1]
26:10.0 notmyhost4.net.39346 > 10.1.60.101.33468: udp 12

This is most likely a coordinated traceroute. This technique can be used to find points outside of the targets control/protection, that when attacked could effect the targets service thereby creating a DoS condition.

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