Global Information Assurance Certification Paper

*** Northcutt, I suspect this student had fun with this. Hard to grade since some are lab created, some are from GIAC, some are from local site. Solid use of a process, this document has a lot of teaching value, some solid research. Good show! 85 *

GCIA Certification Practical

10 Detects with Analyses

Jason Steckler

April 20, 2000

Submitted as practical for SANS 2000 written exam (March 25, 2000).

DETECT # 1

09:46:22.770824 SCANNER.NET.63410 > NT.SERVER.ORG.5300: F 0:0(0) win 4096
09:46:22.771070 SCANNER.NET.63410 > NT.SERVER.ORG.416: F 0:0(0) win 4096
09:46:22.771154 SCANNER.NET.63410 > NT.SERVER.ORG.7003: F 0:0(0) win 4096
09:46:22.771233 SCANNER.NET.63410 > NT.SERVER.ORG.783: F 0:0(0) win 4096
09:46:22.771290 SCANNER.NET.63410 > NT.SERVER.ORG.1380: F 0:0(0) win 4096
09:46:22.771382 SCANNER.NET.63410 > NT.SERVER.ORG.207: F 0:0(0) win 4096
09:46:22.771452 SCANNER.NET.63410 > NT.SERVER.ORG.960: F 0:0(0) win 4096
09:46:22.771520 SCANNER.NET.63410 > NT.SERVER.ORG.10082: F 0:0(0) win 4096
09:46:22.771940 NT.SERVER.ORG.288 > SCANNER.NET.63410: R 0:0(0) ack 1 win 0
09:46:22.772033 NT.SERVER.ORG.5300 > SCANNER.NET.63410: R 0:0(0) ack 1 win 0
09:46:22.772134 NT.SERVER.ORG.416 > SCANNER.NET.63410: R 0:0(0) ack 1 win 0
09:46:22.772231 NT.SERVER.ORG.7003 > SCANNER.NET.63410: R 0:0(0) ack 1 win 0

Targeting: Yes
History: No other history of this host on our net.
Techniques: This is a FIN scan of just one particular host. The source port is remaining the same for the entire scan.
Intent: The host is scanning one particular host looking for a certain response/non-response.
Analysis: This is a very fast attack. The source port remains the same throughout the scan indicating that a automated tool such as nmap was used. Scary part would be just one particular host was targeted. May show the attacker already has some mapping experience with this network. The fact that the FIN flag is set while scanning this particular host indicates that TCP/IP fingerprinting is probably taking place. The scanner is trying to distinguish what operating system is present by using the responses/non-responses from the target. Since not all machine follow the RFC 793 it would be more logical to see a SYN scan first to find open ports and then a FIN scan to determine the OS.

Components	Score	Comments
Criticality	5	Only one machine is targeted. May have some previous info.
Lethality	4	Trying to determine what OS is present for a future attack.
System Countermeasures	3	This particular host is not up to date with patches.
Network Countermeasures	4	Firewall is in place.
Severity Total	2	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 2 **(some lines removed for brevity)**

03:33:02.294354 syn.scanner.54591 > target.net.212: S 2226815369:2226815369(0) win 2048
03:33:02.294424 syn.scanner.54591 > target.net.331: S 2226815369:2226815369(0) win 2048
03:33:02.294493 syn.scanner.54591 > target.net.749: S 2226815369:2226815369(0) win 2048
03:33:02.294562 syn.scanner.54591 > target.net.2042: S 2226815369:2226815369(0) win 2048
03:33:02.294630 syn.scanner.54591 > target.net.811: S 2226815369:2226815369(0) win 2048
03:33:02.294701 syn.scanner.54591 > target.net.415: S 2226815369:2226815369(0) win 2048
03:33:02.295035 syn.scanner.54591 > target.net.135: S 2226815369:2226815369(0) win 2048
03:33:02.301413 target.net.135 > syn.scanner.54591: S 109380:109380(0) ack 2226815370 win 8576 <mss 1460> (DF)
03:33:02.301580 syn.scanner.54591 > target.net.135: R 2226815370:2226815370(0) win 0

Targeting: Yes
History: No other history of this host on our net.
Techniques: This is an example of half open scanning (TCP Scanning). SYN scan of just one particular host. The source port is remaining the same for the entire scan.
Intent: The host is scanning one particular host looking for open ports and possible vulnerabilities associated with it.
Analysis: This is a very fast attack happening in the early morning. The source port remains the same throughout the scan indicating that an automated tool such as nmap was used. Since the SYN flag is set while scanning this particular host, it indicates that fingerprinting is probably taking place. Notice once the scanner encounters an open port on the target and the target responds back with a SYN-ACK. The connection is immediately terminated by the scanner with a RST. This is due to the fact that the scanners OS did not initiate the connection and knows nothing about it. The packets were crafted by the scanning program and the OS generates the RST. This is important, not all targets will log this activity because the 3-way handshake is never completed.

Components	Score	Comments
Criticality	4	Only one machine is targeted. Non-critical system.
Lethality	2	Just a scan at present. OS determination
System Countermeasures	4	This particular host is up to date with patches.
Network Countermeasures	3	Firewall is in place. But not all targets pick this type of scan up.
Severity Total	-1	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 3 **(conducted in a lab)**

```
12:23:38.089191 BAD.GUYS.COM.1335 > POOR.SAP.NET.12346: S 110186:110186(0) win 8192 <mss 1460> (DF) [tos 0x10]
12:23:38.089484 POOR.SAP.NET.12346 > BAD.GUYS.COM.1335: S 40163:40163(0) ack 110187 win 8760 <mss 1460> (DF)
12:23:38.089640 BAD.GUYS.COM.1335 > POOR.SAP.NET.12346: . ack 1 win 8760 (DF) [tos 0x10]
12:23:38.097553 POOR.SAP.NET.12346 > BAD.GUYS.COM.1335: . 1:1461(1460) ack 1 win 8760 (DF)
12:23:38.098784 POOR.SAP.NET.12346 > BAD.GUYS.COM.1335: . 1461:2921(1460) ack 1 win 8760 (DF)
12:23:38.098909 BAD.GUYS.COM.1335 > POOR.SAP.NET.12346: . ack 2921 win 8760 (DF) [tos 0x10]
12:23:38.100367 POOR.SAP.NET.12346 > BAD.GUYS.COM.1335: P 2921:4381(1460) ack 1 win 8760 (DF)
12:23:38.101600 POOR.SAP.NET.12346 > BAD.GUYS.COM.1335: . 4381:5841(1460) ack 1 win 8760 (DF)
12:23:38.101905 BAD.GUYS.COM.1335 > POOR.SAP.NET.12346: . ack 4381 win 8760 (DF) [tos 0x10]
```

Targeting: Yes
History: Have had problems with somewhat savvy users trying to mail/install Netbus on fellow workers machines (I guess they think it is funny).
Techniques: This is the classic Netbus traffic (port 12346). Connection is already made.
Intent: This is not a scan. The connection and use of the Trojan is apparent. Unsure whether harmful intent is at hand or someone just trying to have fun and gain info. Either way, must consider it very hostile and a definite breech.
Analysis: This is a old Trojan, but still very much a problem. This detect was captured in a lab after our client AV and mail server AV were reporting several users trying to mail the Patch.exe file to fellow workers. We have not seen the "savvy users" scanning for the infected machines only the initial attempt to induce the Trojan. For the sake of the above detect, this would be considered a definite security problem as shown in the Severity scale below.

Components	Score	Comments
Criticality	3	Machine is already infected, but non-critical workstation.
Lethality	5	Attacker has all the access he needs.
System Countermeasures	2	This particular host is not up to date with AV Def. Files.
Network Countermeasures	3	AV is installed on mail servers to protect recipients. Filtering router in lab did not stop this traffic.
Severity Total	3	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 4

14:32:43.464892 192.1.1.10.1269 > HOST.OUR.NET.139: S 109664:109664(0) win 8192 <mss 1460> (DF) [tos 0x10]
14:32:43.465149 HOST.OUR.NET.139 > 192.1.1.10.1269: S 39682:39682(0) ack 109665 win 8760 <mss 1460> (DF)
14:32:43.465343 192.1.1.10.1269 > HOST.OUR.NET.139: . ack 1 win 8760 (DF) [tos 0x10]
14:32:43.466977 192.1.1.10.1269 > HOST.OUR.NET.139: P 1:5(4) ack 1 win 8760 urg 4 (DF) [tos 0x10]
14:32:43.467201 HOST.OUR.NET.139 > 192.1.1.10.1269: FP 1:6(5) ack 5 win 8757 (DF)
14:32:43.467392 192.1.1.10.1269 > HOST.OUR.NET.139: . ack 7 win 8755 (DF) [tos 0x10]
14:32:43.467961 192.1.1.10.1269 > HOST.OUR.NET.139: R 109669:109669(0) win 0 (DF) [tos 0x10]
14:32:51.656210 192.1.1.10.1270 > HOST.OUR.NET.139: S 109670:109670(0) win 8192 <mss 1460> (DF) [tos 0x10]

Targeting: Yes
History: No previous history of this host on our net.
Techniques: This is a Winnuke attack against a Windows machine (port 139).
Intent: This is not a scan. This is an attempted DOS attack.
Analysis: This type of DOS attack is also called an Out of Band (OOB) nuke. The NetBIOS port (139) is being used on the target to send OOB traffic. If the target does not have the latest patches, lock-ups or Blue Screens can occur. The first 3 lines show the three-way handshake. Then traffic is sent with the Urgent bit set. In this case the client must be patched since the connection is discontinued. The attacker comes right back with his next port of 1270 and tries the same port on the target again. This probably indicates someone with a low level of knowledge or needs some more confirmation his tool isn't working. Another interesting point is that port 139 traffic from outside nets are not very common. On this particular net there should be no NetBIOS traffic coming from the outside. I can probably understand this type of traffic for a Windows NT trust or something that is established across two networks.

Components	Score	Comments
Criticality	2	Non-critical windows machine.
Lethality	4	If successful, a complete DOS.
System Countermeasures	5	This particular host is up to date with patches.
Network Countermeasures	2	Firewall present, but allowing port 139 traffic at the time of the incident.
Severity Total	-1	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 5 (conducted in a lab)

(THREE-WAY HANDSHAKE)
08:11:59.499777 BO.NET.1046 > INFECTED.NET.54320: S 94006:94006(0) win 8192 <mss 1460> (DF) [tos 0x10]
08:11:59.500022 INFECTED.NET.54320 > BO.NET.1046: S 42442:42442(0) ack 94007 win 8760 <mss 1460> (DF)
08:11:59.500184 BO.NET.1046 > INFECTED.NET.54320: . ack 1 win 8760 (DF) [tos 0x10]

(CLIENT STIMULUS)
08:11:59.500753 BO.NET.1046 > INFECTED.NET.54320: P 1:226(225) ack 1 win 8760 (DF) [tos 0x10]

(SERVER RESPONSE)
08:11:59.514864 INFECTED.NET.54320 > BO.NET.1046: P 1:29(28) ack 226 win 8535 (DF)
08:11:59.527833 INFECTED.NET.54320 > BO.NET.1046: P 29:118(89) ack 270 win 8491 (DF)
08:11:59.528008 INFECTED.NET.54320 > BO.NET.1046: P 118:187(69) ack 270 win 8491 (DF)
08:11:59.528424 INFECTED.NET.54320 > BO.NET.1046: P 187:260(73) ack 270 win 8491 (DF)
08:11:59.528605 INFECTED.NET.54320 > BO.NET.1046: P 260:322(62) ack 270 win 8491 (DF)

Targeting: Yes
History: Conducted in a lab environment.
Techniques: This is a Back Orifice 2000 session against a Windows NT machine.
Intent: This is remote administration utility, but mostly used as a Trojan program.
Analysis: Back Orifice 2000 is completely different that the old BO. In the detect above, TCP is used rather than UDP (the old BO could only use UDP). The new BO2K also allows you to use any TCP port. As you can see above, a connection is established. Then we notice that a single packet from BO.NET produces many response packets, such as in a client server relationship. This was conducted in a lab environment. But for the sake of determining severity the traffic was viewed as real.

Components	Score	Comments
Criticality	2	Non-critical windows machine.
Lethality	4	If successful, an attacker has full remote administration.
System Countermeasures	2	This particular host is not up to date with AV definitions.
Network Countermeasures	2	Permissive firewall present.
Severity Total	2	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 6 (conducted in a lab)

09:54:23.658584 NET.MAPPER.COM > 192.10.80.255: icmp: echo request
09:54:23.876032 NET.MAPPER.COM > 192.10.80.0: icmp: echo request
09:54:23.882483 NET.MAPPER.COM > 192.10.80.255: icmp: echo request
09:54:23.899721 NET.MAPPER.COM > 192.10.80.0: icmp: echo request

Targeting: Yes
History: Conducted in a lab environment.
Techniques: Sending ping packet to entire subnet of 192.10.80 using nmap.
Intent: Port mapping used for reconnaissance of network.
Analysis: Automated attack using nmap. This type of attack is used to send a ping packet to entire subnet to received information about hosts that may be active. We have all used ping to see if a host is alive on our net, but this type of activity can be a precursor to an attack. Yes, this scan was coming fast and furious and the system could become overloaded. In actuality, I am sure that some "buffer time" would be used in between each echo request. Stopping icmp echo replies to the Internet would be a good idea in this case. After I did this in the lab, we checked our Cisco router ACL's for this vulnerability. For the sake of determining severity the traffic was viewed as real.

Components	Score	Comments
Criticality	4	Not a system, but a subnet so I vote 4
Lethality	4	If successful, an attacker has great recon info on you.
System Countermeasures	2	Not much can be done at system level.
Network Countermeasures	5	Can disable icmp replies to Internet
Severity Total	1	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 7 (from GIAC site)

Apr 8 11:05:48 192.116.7.35:2434 -> a.b.e.52:53 UDP
Apr 8 11:05:48 192.116.7.35:2825 -> a.b.e.58:53 UDP
Apr 8 11:05:49 192.116.7.35:2973 -> a.b.e.63:53 UDP
...
Apr 8 11:05:51 192.116.7.35:2184 -> a.b.e.201:53 UDP
Apr 8 11:05:55 192.116.7.35:4141 -> a.b.e.176:53 UDP
Apr 8 11:05:55 192.116.7.35:3800 -> a.b.e.135:53 UDP
Apr 8 11:05:55 192.116.7.35:3367 -> a.b.e.216:53 UDP

Targeting: Yes
History: Obtained from GIAC site. Noticed same source on 12 April detects.
Techniques: Sending UDP packets to various hosts (port 53) on the a.b.e. network.
Intent: Looking for active DNS servers. (DNS scan)
Analysis: This scan is coming fast. The scanner is using UDP port scanning to determine which hosts are active with UDP port 53. UDP port 53 is used for server to server DNS queries. Many programs including nmap are able to perform this type of scan. The concept is simple, closed ports respond with an icmp "port unreachable" error message. Open ports do not respond at all. UDP is an unreliable protocol. This unreliability makes it difficult for the scanner when he is far away from the target (packets get lost along the way). DNS filtering rules should be included in your router's ACL's. If you have an external secondary name server, make sure you are specific about tcp 53. This will help prevent outsiders from trying to download your zone file. Attackers address resolves as Bethlehem Bible College, Jerusalem, Israel. Noticed activity from this host also on 12 April 2000 scanning another net. Based on the information at hand, it probably requires a capture of all traffic coming from the source ip.

C:\nslookup 192.116.7.35
Server: xxxxxxxxx
Address: xxxxxxxxx

Name: linux.bethlehembiblecollege.EDU
Address: 192.116.7.35

Kinda have to guess on Severity since I don't know a lot about the MY.NET hosts.

Components	Score	Comments
Criticality	5	Specifically looking for DNS services.
Lethality	4	If successful, an attacker can gain info or exploit DNS vulnerabilities.
System Countermeasures	4	Modern OS with current patches.
Network Countermeasures	3	No info about Firewall. Router ACL's can help though.
Severity Total	2	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 8 **(conducted in a lab)** **(some lines removed for brevity)**

```
19:11:49.312959 ATTACKER.COM > TARGET.ORG: icmp: echo request (frag 22016:1480@0+)
19:11:49.313023 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@1480+)
19:11:49.313036 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@2960+)
19:11:49.313047 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@4440+)
19:11:49.313058 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@5920+)
19:11:49.313071 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@7400+)
19:11:49.313082 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@8880+)
19:11:49.313094 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@10360+)
19:11:49.313107 ATTACKER.COM > TARGET.ORG: (frag 22016:1480@11840+)
……..
19:11:52.348049 ATTACKER.COM > TARGET.ORG: (frag 23296:1480@57720+)
19:11:52.348060 ATTACKER.COM > TARGET.ORG: (frag 23296:808@59200)
```

Targeting: Yes
History: No history available. Conducted in a Lab.
Techniques: Crafting an oversized ICMP datagram (ping packet).
Intent: The attacker is trying to crash the target machines OS. Denial of service attack.
Analysis: I used an old NT 3.51 machine to execute the ping of death directed toward an NT 4 machine. The packet is very easily crafted with out the use of any tools (older OS needed). The basis behind this type of attack is simple; send a large ping packet to the target machine. Since most OS's do not know how to handle this packet, which is larger than the maximum size, a crash or system hang will usually occur. This occurs since the TCP/IP specification only allows for a packet size of up to 65536 octets (1 octet=8 bits of data), containing 20 octets of IP header information and 0 or more octets of additional information. The rest of the packet is considered data. A lot of OS's like NT and routers with older IOS images can be vulnerable to this type of attack. NT 4 is vulnerable to sending large packets such as this (no hot fixes applied and low service pack), but does not crash when it receives a large packet.

Components	**Score**	**Comments**
Criticality	3	Only one machine is targeted. Non-critical windows system.
Lethality	4	Can cause a denial of service.
System Countermeasures	5	Modern Os and is up to date with patches.
Network Countermeasures	2	Current router IOS does not stop these packets.
Severity Total	0	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 9 **(Conducted in a lab)** **(Some lines removed for brevity)**

127.0.0.50 telnets to router 127.0.0.1.
127.0.0.55 conducts a session hijacking of the telnet session using Hunt.

Phase 1
Steal the session. Hijacker arp spoofs 127.0.0.1 by telling it that 127.0.0.50 is at the spoofed mac of ea:1a:de:ad:be:3. Later, the hijacker arp spoofs 127.0.0.50 by telling it that 127.0.0.1 is at the spoofed mac of ea:1a:de:ad:be:4. Notice the arps that occur.

06:40:36.503284 arp reply 127.0.0.50 is-at ea:1a:de:ad:be:3
06:40:36.503330 arp reply 127.0.0.50 is-at ea:1a:de:ad:be:3
06:40:36.612366 arp reply 127.0.0.50 is-at ea:1a:de:ad:be:3
06:40:36.612422 arp reply 127.0.0.50 is-at ea:1a:de:ad:be:3

06:40:42.342922 arp reply 127.0.0.1 is-at ea:1a:de:ad:be:4
06:40:42.342967 arp reply 127.0.0.1 is-at ea:1a:de:ad:be:4
06:40:42.343038 arp who-has 127.0.0.55 tell 127.0.0.1
06:40:42.343103 arp who-has 127.0.0.55 tell 127.0.0.1
06:40:42.345042 arp who-has 127.0.0.55 tell 127.0.0.50
06:40:42.345219 arp who-has 127.0.0.55 tell 127.0.0.50
06:40:42.345280 arp who-has 127.0.0.55 tell 127.0.0.50
06:40:42.452386 arp reply 127.0.0.1 is-at ea:1a:de:ad:be:4
06:40:42.452445 arp reply 127.0.0.1 is-at ea:1a:de:ad:be:4
06:40:42.452514 arp who-has 127.0.0.55 tell 127.0.0.1
06:40:42.452583 arp who-has 127.0.0.55 tell 127.0.0.1
06:40:42.452689 127.0.0.1 > 127.0.0.50: icmp: echo request (DF)
06:40:42.452775 127.0.0.50 > 127.0.0.1: icmp: echo reply (DF)

Phase 2
The hijacker conducts his business. He has the same control of the router that the user who initiated the connection had….Maybe more if he has been watching various sessions to this router in the past. With the attacker in the middle, in promiscuous mode, all traffic between 127.0.0.1 and 127.0.0.50 can be captured. The real hosts of 127.0.0.1 and 127.0.0.50 will not gather the data because it is destined for the spoofed mac address. 127.0.0.55 becomes the relay between the two hosts.

06:43:06.315303 127.0.0.50.1095 > 127.0.0.1.23: P 63:64(1) ack 798 win 8664 (DF) [tos 0x10]
06:43:06.315552 127.0.0.1.23 > 127.0.0.50.1095: P 798:799(1) ack 64 win 4176 (DF)
06:43:06.459142 127.0.0.50.1095 > 127.0.0.1.23: . ack 799 win 8663 (DF) [tos 0x10]
06:43:06.479061 127.0.0.50.1095 > 127.0.0.1.23: P 64:65(1) ack 799 win 8663 (DF) [tos 0x10]
06:43:06.479320 127.0.0.1.23 > 127.0.0.50.1095: P 799:800(1) ack 65 win 4176 (DF)

Phase 3
This is where the session was synching back up. Notice the Hijacker sends a gratuitous arp letting 127.0.0.1 know that 127.0.0.50 is at its original mac address of 0:50:4:5e:4f:ef. He sends an arp informing 127.0.0.50 that 127.0.0.1 is at its original mac address of 0:10:7b:12:58:96.

06:43:07.661175 arp reply 127.0.0.50 (0:50:4:5e:4f:ef) is-at 0:50:4:5e:4f:ef
06:43:07.661231 arp reply 127.0.0.50 (0:50:4:5e:4f:ef) is-at 0:50:4:5e:4f:ef

06:43:07.661559 arp reply 127.0.0.1 (0:10:7b:12:58:96) is-at 0:10:7b:12:58:96
06:43:07.661627 arp reply 127.0.0.1 (0:10:7b:12:58:96) is-at 0:10:7b:12:58:96

Targeting: Yes
History: No history available. Conducted in a Lab.
Techniques: Conducted a session hijacking with a successful re-synch.
Intent: The attacker stole the telnet session of an unsuspecting user. Would consider malicious activity.
Analysis: I used the hunt program on a Linux box to steal a telnet session between a windows machine and a router. There are a couple of interesting parts to this detect. The first would be the initial hijacking. There is an increased amount of arp traffic on the network. This is be very hard to detect on an actual network and was even in a lab. The second is that it took about 5 attempts to successfully synch the session back to the original participants. This is because the program has a pretty high success rate of stealing the session, but a low rate of actually synching the session back up so as not to tip off the user. If the hijacking was successful, but the re-synch was not, big deal. How many times have we all had a telnet session fail and just re-connected. This was a very interesting study, but very scary. This is a good one to show management because it adds a little realism to what the security staff is up against.

Components	Score	Comments
Criticality	5	Could be directed against a core router or critical system.
Lethality	5	Attacker can gain high privilege (15 on this particular Cisco router).
System Countermeasures	2	Can initiate VPN's or SSH.
Network Countermeasures	2	Very hard to detect except for the increased number of arps at steal and re-synch.
Severity Total	6	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

DETECT # 10 **(Conducted in a lab)** **(Some lines removed for brevity)**

Phase 1
192.168.1.59 has an open telnet session to 192.168.1.1.

06:25:49.321494 192.168.1.1.23 > 192.168.1.59.1089: P 1:2(1) ack 1 win 4160
06:25:49.468195 192.168.1.59.1089 > 192.168.1.1.23: . ack 2 win 8152 (DF) [tos 0x10]
06:25:49.574002 192.168.1.59.1089 > 192.168.1.1.23: P 1:2(1) ack 2 win 8152 (DF) [tos 0x10]
06:25:49.577887 192.168.1.1.23 > 192.168.1.59.1089: P 2:3(1) ack 2 win 4159
06:25:49.768539 192.168.1.59.1089 > 192.168.1.1.23: . ack 3 win 8151 (DF) [tos 0x10]
06:25:49.873036 192.168.1.59.1089 > 192.168.1.1.23: P 2:3(1) ack 3 win 8151 (DF) [tos 0x10]
06:25:49.878053 192.168.1.1.23 > 192.168.1.59.1089: P 3:4(1) ack 3 win 4158
06:25:50.017313 192.168.1.59.1089 > 192.168.1.1.23: P 3:4(1) ack 4 win 8150 (DF) [tos 0x10]
06:25:50.021702 192.168.1.1.23 > 192.168.1.59.1089: P 4:5(1) ack 4 win 4157

Phase 2
The hijacker steals the session from 192.168.1.59, but does not conduct arp spoofing. Since the hijacker simply takes over the session, the sequence numbers between 192.168.1.59 and 192.168.1.1 become out of synch. As the two hosts try to resynchronize, they will send SYNs and ACKs back and forth trying to fix the problem. This behavior results in the following ACK storm.

06:26:05.992980 192.168.1.59.1089 > 192.168.1.1.23: P 15:23(8) ack 615 win 7434 (DF)
06:26:06.025354 192.168.1.1.23 > 192.168.1.59.1089: P 720:760(40) ack 23 win 4138
06:26:06.025475 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:06.025521 192.168.1.59.1089 > 192.168.1.1.23: . ack 760 win 7434 (DF)
06:26:06.028931 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:06.029014 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:06.031150 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:06.031233 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:06.033354 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:06.033443 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:06.035573 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:06.035655 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]

Phase 3
Hundreds of ACKs later, 192.168.1.59 sends RST's to 192.168.1.1 trying to regain communication.

06:26:10.090598 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:10.090678 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:10.092745 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:10.092825 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:10.094972 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:10.095053 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:10.097127 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138
06:26:10.097208 192.168.1.59.1089 > 192.168.1.1.23: . ack 720 win 7434 (DF) [tos 0x10]
06:26:10.097276 192.168.1.59.1089 > 192.168.1.1.23: R 23:23(0) ack 762 win 7434 (DF)
06:26:10.097454 192.168.1.59.1089 > 192.168.1.1.23: R 24:24(0) ack 763 win 7434 (DF)
06:26:10.097519 192.168.1.59.1089 > 192.168.1.1.23: R 25:25(0) ack 764 win 7434 (DF)
06:26:10.097589 192.168.1.59.1089 > 192.168.1.1.23: R 26:26(0) ack 765 win 7434 (DF)
06:26:10.097658 192.168.1.59.1089 > 192.168.1.1.23: R 27:27(0) ack 766 win 7434 (DF)
06:26:10.100920 192.168.1.1.23 > 192.168.1.59.1089: . ack 23 win 4138

Targeting: Yes
History: No history available. Conducted in a Lab.
Techniques: Simple hijacking with no arp spoofing.
Intent: Gain control of a current telnet session.
Analysis: Again, I used the hunt program on a Linux box to steal a telnet session between two clients. The hijacker watches the session happening between the two clients. Then when the attacker is ready, the session hijacking tool simply jumps in and continues the session with 192.168.1.1. The Hunt program uses its built in sniffer to receive responses from 192.168.1.1. The ack storm is a very discernable feature of this detect. We have watched our production net, and have never quite seen traffic like this. This traffic is a good contrast to detect 9. There we conducted arp spoofing to avoid the ack storm. This could have also been accomplished by a DOS attack against 192.168.1.59 while hijacking takes place. The best way to prevent this type of activity is to encrypt sessions, use strong authentication, initiate VPNs and don't telnet to firewalls.

Components	Score	Comments
Criticality	5	Could be directed against a core router or critical system.
Lethality	5	Attacker takes over current session.
System Countermeasures	3	Can initiate VPN's or SSH.
Network Countermeasures	3	Large ack storm occurs.
Severity Total	4	(Criticality + Lethality) – (System Countermeasures + Net countermeasures)

Last Updated: May 28th, 2017

Upcoming Training

Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
Baltimore September 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Boston SEC503	Boston, MA	Oct 09, 2017 - Oct 14, 2017	Community SANS
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced