AirNIDS: The Need for Intrusion Detection on the Wireless Ether

Thomas Hoffecker

GIAC GCIA Practical Assignment

Version 3.2

SANSFIRE Boston 2002

June 25 - July 2 2002

Thomas Hoffecker

AirNIDS: The Need for Intrusion Detection on the Wireless Ether

Introduction

The inherent insecurities and vulnerabilities of wireless 802.11b networks are well known. The benefit of being wireless is the greatest drawback. In the past two years there has been an explosion of wireless networks installed. Many of these are legitimate wireless access points deployed by network administrators. However, there is also a fairly large number that are installed by employees that desire an untethered LAN connection. They trade the security of a hardwired connection for the convenience a wireless attachment to the corporate network. The concern arises of whom or what will perform intrusion detection on that wireless connection?

Wireless Network Insecurities

The security vulnerabilities that wireless networks bring to organizations that employ them are well established in the network security arena. With the proliferation of inexpensive wireless cards and high gain external antennas, a new generation of war dialers has emerged. These enthusiasts don't dial phone numbers, they drive around business parks and industrial areas looking for open wireless networks. They collect their wardriving[1] gathered data and publish it to the web. Tools such as NetStumber, Kismet, and Wellenreiter[2] are well known and are used to find wireless access points. These tools range from easy to use (NetStumbler) to the more sophisticated (Kismet). A tool such as NetStumbler is active and loudly probes for access points, both Kismet and Wellenreiter will passively listen for the presence of 802.11b network activity. These tools will detect wireless networks on which administrators have taken steps to secure them by disabling the broadcast of the SSID.

After a wireless access point has been found, the intruder will determine if the built in Wired Equivalent Privacy (WEP) has been enabled. Even though the wireless network may be encrypted, the intruder can use such tools as AirSnort, wepcrack, or dwepcrack[3] to break the encryption through weaknesses in the RC4 cryptographic algorithm key exchange[4] that was implemented by the IEEE for the 802.11b standard.

Mitigating the Wireless Network Risk

Securing wireless networks involves designing security into the deployment plan from the beginning. The Department of Defense uses the term of controlled space[5] to determine what safeguards must be put in place to prevent signal leakage beyond the intended users. Controlling how far the signal emanates is a key to wireless security. A

[1] Wardriving - The searching for wireless networks by means of a roaming (driving, walking, busing?) wireless client. Sometimes accompanied by a high gain antenna and GPS. As per definition by Seattle Wireless. URL: http://www.seattlewireless.net/index.cgi/WarDriving

[2] Netstumbler. URL: http://www.netstumbler.org/. Kismet. URL: http://www.kismet-wireless.org, Wellenreiter. URL: http://www.remote-exploit.org.

[3] AirSnort URL: http://airsnort.shmoo.com , wepcrack. URL: http://sourceforge.net/projects/wepcrack, dwepcrack - http://www.dachb0den.com/projects/dwepcrack.html

[4] Fluhrer, Scott; Mantin, Itsik; Shamir, Adi; *Weakness in the Key Scheduling Algorithim of RC4*. URL http://www.drizzle.com/~aboba/IEEE/rc4_ksaproc.pdf

[5] NIST. National Information System Security Glossary. September 2000. Page 17. URL: http://www.nstissc.gov/Assets/pdf/4009.pdf

tool such as Wireless Valley's LANPlanner[6] can be used to both optimize your access point placement and signal leakage.

Once control of the signal emanation has taken place, the owner of the wireless network needs to control how the access points react to wireless client's probes. Many access points allow the broadcast of the SSID to be disabled. If the access point has this feature it should be enabled.

Another measure that can be taken is to enable WEP with 128bit encryption (actually uses a 104bit key). Although with enough time and intruder can break WEP, it will increase the amount of time required for a successful intrusion. The time it will take can also depend on the brand of equipment and amount of traffic traversing the wireless segment. During informal testing at my place of employment, using a Linksys access point with 64bit WEP enabled and a continuous ping on the wireless segment, AirSnort was able to crack the key as quickly as 27 minutes. Using a Cisco Aironet access point with the same testing parameters AirSnort was not able to crack the key after 24 hours. The difference was the number of interesting packets[7] that the Linksys access point produced. Using WEP will dissuade most wardrivers that are looking for "low hanging fruit."

In addition to encrypting traffic, an extra layer of security can be achieved by restricting what wireless clients can communicate with an access point. This can be done with restriction by a client's MAC (media access layer) address. This six character hexadecimal uniquely identifies a network card. However, it is possible to spoof a MAC address, so restricting by MAC address does not offer a great deal of additional security.

For many organizations the built-in security to 802.11b is insufficient. They elect to use technologies such as IPSec or SSH to further encrypt their wireless traffic as an additional layer of encryption. They connect their wireless access points to a wireless firewall gateway[8] that enforces a security policy upon wireless users. The use of VPN concentrators[9] that service remote access users can also double as wireless access control devices.

These measures should be implemented on closed networks that provide service to specific users. However, these measures are not possible for public hotspots from which wireless Internet connectivity is offered since users could not easily find the service or connect to it without additional software or configuration changes. For example, many Starbucks[10] coffee houses offer T-Mobile wireless access. Many other public places such

[6] LANPlanner. URL: http://www.wirelessvalley.com/Products/LANPlanner/LANPlannerSuite.asp
[7] AirSnort FAQ. "About how long would it take to get the password for a network with AirSnort?" URL: http://airsnort.sourceforge.net/faq.html
[8] Boscia, Nichole K.;Shaw, Derek G. "NASA Wireless Firewall Gateway White Paper" 3.0.3. URL: http://www.nas.nasa.gov/Groups/Networks/Projects/Wireless/. 9 September 2002.
[9] Snyder, Joel. "Network World Fusion – Wireless IPSec" 9 September 2002. URL: http://www.nwfusion.com/research/2002/0909ilabipsec.html
[10] Griffith, Eric. "Starbucks offers wireless Internet access" 20 August 2002. URL: http://www.80211-planet.com/news/article.php/1449661

as airport terminals and hotels offer similar services. The question arises, who performs intrusion detection on these networks? Furthermore, who performs intrusion detection on wireless networks if the intruder does not attack the wired infrastructure of the network beyond the access point? If the intruder only attacks other wireless clients the only measure of defense is a host based IDS (HIDS), if installed? What if the intruder compromised another wireless client and exploits it as a stepping-stone to attack hosts on the wired infrastructure? How will denial of service (DoS) attacks against the wireless clients and access points be detected? How will BestBuy[11] stores realize that someone in the parking lot has hijacked sessions from the wireless point-of-sale terminals and is now stealing customer's credit card numbers? How will an organization's network security team find a rogue network access point before a wardriver does? A recent vulnerability in some access points allows an intruder to steal a copy of the configuration via TFTP[12] including the WEP key and sanctioned client MAC addresses, how will this be detected?

Wireless Network Attacks: Beyond the Hobbyist Wardriver

Intrusion detection on wireless networks is in its infancy. However, with security incidents such as a recent incident in Atlanta[13] by Internet Security Systems (ISS), attacks have escalated beyond the casual wardriver. The attacker configured an access point close enough to an organization so that wireless clients could connect to it in addition to the organization's legitimate access points. This access appeared to the casual user as the corporate wireless infrastructure. This decoy access point represents the very unsophisticated man-in-the-middle attack using wireless technology. Unfortunately, not much information of this attack has been made available. However, it goes far beyond the accidental connecting to a corporate network by an unaware laptop or PDA user.

In a similar manner, an attacker could render a public hotspot inoperable. If the attacker sends out continuous 802.11b management frames that tell clients by either MAC address or broadcast to deauthenticate, all clients will deassociate with the access point and will be unable to reconnect. At public hotspot, such as a Starbucks users might complain to the staff. However, if the hotspot is in a public airport terminal, there is no staff present to receive paying customer's complaints. Calling tech support of the provider may result in a field engineer coming out and checking the equipment, however, if the DoS attack is periodic the true cause may never be discovered nor the problem remedied.

Although the attack in Atlanta used an access point, one is not required to accomplish this type of attack. The key to this attack and others that directly attack 802.11b is that the management frames between the access points and the clients are not authenticated. These management frames are at layer two of the OSI model, the data link layer. These frames operate where normally the Ethernet frames such as Ethernet_II or Ethernet802.2 are on traditional hard wired Ethernet networks. These frames control the speed and link

[11] "BestBuy uses 802.11b without WEP on cash registers" URL: http://www.geocrawler.com/lists/3/Security/90/0/8551181/

[12] Fungus. "Access points disclose wep keys, password and mac filter" 5 November 2002. URL: http://www.netstumbler.org/article.php?sid=496&mode=thread&order=0

[13] Cox, John. "Wireless LAN attacks grow in sophistication" 28 October 2002. URL: http://www.idg.net/ic_959359_6149_1-3427.html

properties of the wireless connection. In this case, the attacker sends out a management frame that deauthenticates and deassociates the client with the access point. This forces the client's wireless network card to scan all the available channels for another access point with which to associate. The attacker's simulated access point, on a different channel, is then chosen by the client. Under normal circumstances, the user might not realize that he is associated with a different access point. The attacker is simulating the original access point by spoofing the MAC address and ESSID. Unless the change in channel is noticed, the user will not realize that he is now connected to a different access point. After the traffic has passed "through" the attacker's machine it is forwarded to the original access point. The attacker's machine has now become the man-in-the-middle and is able capture any traffic sent from the client.

Although these attacks might seem to require a sophisticated arsenal of tools understandable and available only to serious SIGINT (signals intelligence) professionals, the tools necessary are readily available on the Internet. The described attacks are easily reproduced with a family of tools built upon the Air-Jack driver and programs like wlan-jack and monkey-jack. Air-Jack and its family of tools were presented by Mike Lynn and Robert Baird at the Black Hat Briefings in Las Vegas, Nevada in July 2002. These attacks directly target the 802.11b protocol and cannot be detected by traditional IDS products such as Snort or RealSecure. The result or side effects of them can be detected once the attacker turns his crosshairs to the hosts that are located on the wired network side of the access point. Therefore, the need arises to put a NIDS that can understand the 802.11b protocol and can keep track of access points and their clients. What is needed is a NIDS that can watch the wireless airspace of our networks – we need an AirNIDS!

Wireless Network Intrusion

A small company near Atlanta has become a pioneer in the wireless intrusion detection arena. AirDefense Incorporated has released a product that aims to protect against such threats as the attacks described above. The IDS consists of two parts: remote sensors strategically position near access points and a central server appliance from which the sensors are managed. The remote sensors continuously watch and analyze all wireless traffic between clients and access points. In order to differentiate between friend and foe the central server keeps an inventory of clients and access points.

The information stored on the central server appliance is used to watch for rogue access points and new clients by comparing them to a list of authorized clients and access points. A wireless network security policy can be enforced since network managers can determine which access points are communicating on which specific channels, broadcast specific SSIDs, and are using of WEP. The sensors can also detect the presence of ad-hoc networks between clients. Ad-hoc networks function without the use of an access point in a peer-to-peer network arrangement. The network administrators enforce the established security policy from the central console which is monitored and managed over an SSL connection with any standard web browser.

AirDefense can detect active probes looking for access points emitted by such tools as Netstumber. More sophisticated tools aimed at layer 2 of the 802.11b protocol such as

Air-Jack can be detected. Currently AirDefense proceeds to shutdown wireless network access points instead of combating the attack. AirDefense is considering including an offensive "shoot-back" capability for future versions. However, care must be taken when implementing such technologies because "hacking" or attacking back may violate local and Federal statutes. The current version supports disconnecting the intruder from the access point and preventing him from reassociating with the access point thus preventing man-in-the-middle type attacks.

AirDefense is also capable of detecting MAC spoofing by matching the MAC address to vendor specific idiosyncrasies in the network cards. However, this could be rendered ineffective if the hacker determines the make and model of a wireless client by the manufacture's code within the MAC address.

AirDefense provided the security industry's response to the BlackHat briefing on Air-Jack. It monitored the wireless network at DefCon X over a two hour period. During that period the IDS generated over 13,000 alarms. The system found the 8 authorized access points, 35 rogue access points, and more than 800 wireless clients. AirDefense estimates that 200-300 of these clients were spoofed since the number of people involved was about 350.

During the two hour period 807 attacks were detected[14]:

490 Wireless probes by tools such as Netstumbler.
190 Identify thefts by impersonating the MAC address or SSID.
100 Various DoS attacks consisting of creating network noise to shutdown access points, targeting specific clients and deassociating them, targeting clients with man-in-the-middle type attacks.
27 Out-of-specification 802.11b management frames to take over clients and control the network.

At his website (http://802.11ninja.net/) the author of Air-Jack tool denies that he used Air-Jack to corral all wireless users on to a specific access point causing a network outage on Saturday evening at DefCon X. In addition to forcing users onto one access point he also made all DNS queries resolve to one specific host. Although he denies this, he states that he will be releasing the source code with which this feat was performed.

AirDefense can also monitor the performance and health of wireless networks. Reports generated by the central server can be used for performance analysis of individual access points and capacity planning for growth of the wireless network. Other diagnostic information such as excessive CRC error rates, bandwidth hogs, failed connections, or interference from access points can be tracked. The server maintains a log of such errors to provide a complete network health history. The catalog of access points and clients can also be utilized to maintain an inventory of wireless equipment.

[14] Moran, Brian. "AirDefense Press Release" 5 August 2002. URL: http://www.airdefense.net/newsandpress/08_05_02.shtm

If an intrusion should occur, reports can be generated with detailed information to ascertain from which access point the intruder gained access and what network hosts were targeted. This information can be used during forensic analysis of an incident or intrusion.

Although AirDefense is the first vendor with such a wireless intrusion detection product many other vendors are probably preparing similar products. As wireless networks become even more prevalent the need for such products is already becoming a requirement. An ideal product would harness the IDS product that network security professionals already know and extend it into the arena of wireless intrusion detection.

Homegrown AirNIDS: The piggy we already know & love!

By harnessing the open source Snort NIDS and adding the capability of promiscuously sniffing all wireless packets on a given 802.11b channel we can attain a wireless NIDS – the Snort AirNIDS.[15] Although this NIDS will not detect attacks against layers 1 and 2, it will detect intruders that attack other wireless clients or access points themselves.

As proof of this concept, I built such an IDS system using two laptops, Orinoco Silver & D-Link DWL-650 802.11b wireless cards, several desktop computers, and an SMC2655W access point. The IDS' ability to recognize and generate alerts was verified with popular security tools NMap, Nessus, and Whisker.[16] The network used to perform this test is connected to an ISP via a DSL PPPoE connection. The network is separated into two segments using a Pentium II 266 running OpenBSD's PF[17] firewall software. The firewall ruleset allows any outbound connection to the Internet from either the wireless or internal network and keeps statefulness on each connection. The only open port to the Internet is for SSH allowing remote access. The ruleset did not restrict packets between the internal network (10.0.0.0/24) and the wireless network (172.16.0.0/24) segments. IP addressing for both segments is performed via DHCP running on the OpenBSD box. A detailed network drawing is provided at the end of the white paper.

For my testing I used Red Hat Linux 8.0 and Snort RPM[18], however, the complete installation instructions for this operating system and Snort is beyond the scope of this paper. In his paper, William Metcalf[19] gives a very detailed description of the process of installing Red Hat Linux 7.2 and Snort. It is recommended that the wireless network card not be inserted into laptop until the installation of Linux and Snort has been test and completed. The installation steps should be very similar between Red Hat Linux 7.2 and

[15] It is unfortunate that the term "airsnort" has already been used since it would be a perfect description in my opinion. Because of this, I have chosen the term AirNIDS as a generic description of any NIDS that uses an 802.11b card in monitor mode. A search for the term at Google.com resulted in no hits as of early November 2002.

[16] NMap. URL: http://www.insecure.org/nmap, Nessus. URL: http://www.nessus.org, Whisker. URL: http://www.wiretrip.net/rfp/2/index.asp, NMapWin. URL: http://www.nmapwin.org

[17] OpenBSD PF. URL: http://www.openbsd.org & http://www.benzedrine.cx/pf.html

[18] Roesch, Marty. The basic Snort RPM (snort-1.9.0-1snort.i386.rpm) was used. It does not include any of the advanced features.

[19] William Metcalf. "A Practical Guide to Running SNORT on Red Hat Linux 7.2 and Management Using IDS Policy Manger MySQL+IIS+ACID from your Workstation" 2 April 2002. URL: http://rr.sans.org/intrusion/practical_guide.php

8.0.[20] Since this is a test bed, it is advisable to install everything and configure the built-in firewall to allow only inbound SSH. This will avoid having to install any additional libraries or RPMs later. Use of this system with "everything" is not recommended for a production IDS!

In order for AirNIDS box to see all the packets on the wireless network it must be able to enter the wireless equivalent of promiscuous mode called monitor mode. This enables the card to receive all the wireless traffic on that specific channel from all nodes associated with the access point. This is accomplished by replacing the standard PCMCIA drivers with the Linux wlan-ng PCMCIA drivers. This can be done via RPMs or compiled directly from the source code.[21]

After wlan-ng drivers have been successfully installed, insert a Prism2 based PC Card. I used a D-Link DWL-650 because it can easily be modified[22] for connecting an external antenna. The card should be recognized by the laptop. Performing an "`ifconfig -a`" in a terminal window should show that a new interface wlan0 is now available and is in an up status. Your output should look similar this:

```
eth0      Link encap:Ethernet  HWaddr 00:20:E0:66:63:11
          inet addr:10.0.0.152  Bcast:10.0.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:1026 (1.0 Kb)  TX bytes:954 (954.0 b)
          Interrupt:10

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:72 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4682 (4.5 Kb)  TX bytes:4682 (4.5 Kb)

wlan0     Link encap:Ethernet  HWaddr 00:05:5D:A7:06:C9
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:704 (704.0 b)  TX bytes:672 (672.0 b)
          Interrupt:3 Base address:0x100
```

Using another wireless client generate a continuous stream of traffic by opening a streaming audio session or something similar. From a terminal window start the Ethereal sniffer by issuing "`ethereal &`" and start listening on the wlan0 interface. To see packets scroll in real time be sure to select that option under Display Options.[23] It is also

[20] Red Hat, Inc. "'The Official Red Hat Linux x86 Installation Guide" URL: http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/install-guide/
[21] Miller, Tim. "WLAN-NG RPMs" URL: http://prism2.unixguru.raleigh.nc.us/rh80/index.html; Ritchie@tipsybottle.com. "Red Hat Linux 8.0 + Orinoco + Kismet". 3 November 2002. URL: http://www.tipsybottle.com/technology/wireless/RedHat8-Orinoco-Kismet-HOWTO.html.
[22] will@c0rtex.com. "DWL-650 Antenna Connector " 25 July 2002. URL: http://c0rtex.com/~will/antenna/
[23] Sharpe,Richard;Warnicke, Ed. "Ethereal User's Guide" Version 1.1 URL: http://www.ethereal.com/docs/user-guide/ch03capturestart.html

recommended to disable name resolution of hosts since it will speed up the process of displaying the packet data in the capture window.

If packets of the other wireless client are being displayed, you're ready to start Snort on the wireless network card. I configured my `/etc/snort/snort.conf` to use 172.16.0.0/24 as my `HOME_NET` variable, `/etc/snort` as my rule path, and enabled all the `.rules` files for maximum level of detection. I used the following command to start Snort:

```
/usr/sbin/snort -vdUeXDl /var/log/snort -i wlan0 -c /etc/snort/snort.conf &
```

After checking `/var/log/messages` file with the tail to ensure that Snort was running, I performed an NMap ping sweep from the intruder laptop on the wireless network. Snort detected the scan and recorded it in the `/var/log/snort/scan.log` file:

```
[Edited for brevity – event ID removed]
19:36:41.899841  ICMP src: 172.16.0.27 dst: 10.0.0.7 type: 8 code: 0 tgts: 7
19:36:41.900488  ICMP src: 172.16.0.27 dst: 10.0.0.8 type: 8 code: 0 tgts: 8
19:36:41.902529  ICMP src: 172.16.0.27 dst: 10.0.0.9 type: 8 code: 0 tgts: 9
19:36:41.903499  ICMP src: 172.16.0.27 dst: 10.0.0.10 type: 8 code: 0 tgts: 10
19:36:41.904604  ICMP src: 172.16.0.27 dst: 10.0.0.11 type: 8 code: 0 tgts: 11
19:36:41.906003  ICMP src: 172.16.0.27 dst: 10.0.0.12 type: 8 code: 0 tgts: 12
19:36:41.906731  ICMP src: 172.16.0.27 dst: 10.0.0.13 type: 8 code: 0 tgts: 13
19:36:41.907383  ICMP src: 172.16.0.27 dst: 10.0.0.14 type: 8 code: 0 tgts: 14
19:36:41.908089  ICMP src: 172.16.0.27 dst: 10.0.0.15 type: 8 code: 0 tgts: 15
[snip]
19:36:48.246905  ICMP src: 172.16.0.27 dst: 10.0.0.57 type: 8 code: 0 tgts: 57
19:36:48.247914  ICMP src: 172.16.0.27 dst: 10.0.0.58 type: 8 code: 0 tgts: 58
19:36:48.516873  ICMP src: 172.16.0.27 dst: 10.0.0.59 type: 8 code: 0 tgts: 59
19:36:48.517493  ICMP src: 172.16.0.27 dst: 10.0.0.60 type: 8 code: 0 tgts: 60
19:36:48.518520  ICMP src: 172.16.0.27 dst: 10.0.0.61 type: 8 code: 0 tgts: 61
19:36:48.519431  ICMP src: 172.16.0.27 dst: 10.0.0.62 type: 8 code: 0 tgts: 62
[snip]
19:36:49.762754  ICMP src: 172.16.0.27 dst: 10.0.0.194 type: 8 code: 0 tgts: 194
19:36:49.765329  ICMP src: 172.16.0.27 dst: 10.0.0.195 type: 8 code: 0 tgts: 195
19:36:49.766455  ICMP src: 172.16.0.27 dst: 10.0.0.196 type: 8 code: 0 tgts: 196
19:36:49.767021  ICMP src: 172.16.0.27 dst: 10.0.0.197 type: 8 code: 0 tgts: 197
```

In addition to testing Snort AirNIDS capability of detecting traffic originating from the wireless network, I wanted to ensure that Snort was able to watch for attacks originating from the wired network back across the wireless network. In essence, the attacker became the victim. Snort was able to detect the Nessus scan I performed against the intruder's laptop. Snort detected and recorded these events in `/var/log/snort/alert.log`

```
[Edited for brevity – some fields & events removed]
22:05:24 [1:524:4] BAD TRAFFIC tcp port 0 traffic [Classification: Misc activity]
[Priority: 3]: {TCP} 10.0.0.152:12808 -> 172.16.0.27:0
22:05:24 [111:8:1] (spp_stream4) STEALTH ACTIVITY (FIN scan) detection {TCP}
10.0.0.152:12809 -> 172.16.0.27:0
22:05:24 [1:524:4] BAD TRAFFIC tcp port 0 traffic [Classification: Misc activity]
[Priority: 3]: {TCP} 10.0.0.152:12810 -> 172.16.0.27:0
22:05:24 [111:13:1] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection {TCP}
10.0.0.152:12811 -> 172.16.0.27:0
22:05:24 [111:11:1] (spp_stream4) STEALTH ACTIVITY (Vecna scan) detection {TCP}
10.0.0.152:12812 -> 172.16.0.27:0
22:05:30 [1:236:1] DDOS Stacheldraht client-check-gag [Classification: Attempted
Denial of Service] [Priority: 2]: {ICMP} 10.0.0.152 -> 172.16.0.27
22:05:58 [1:1443:1] TFTP GET passwd [Classification: Successful Administrator
Privilege Gain] [Priority: 1]: {UDP} 10.0.0.152:32771 -> 172.16.0.27:69
```

```
22:17:07 [1:239:1] DDOS shaft handler to agent [Classification: Attempted Denial
of Service] [Priority: 2]: {UDP} 10.0.0.152:1024 -> 172.16.0.27:18753
22:17:10 [1:236:1] DDOS Stacheldraht client-check-gag [Classification: Attempted
Denial of Service] [Priority: 2]: {ICMP} 10.0.0.152 -> 172.16.0.27
22:17:11 [1:384:4] ICMP PING [Classification: Misc activity] [Priority: 3]: {ICMP}
10.0.0.152 -> 172.16.0.27
22:17:11 [1:566:3] POLICY PCAnywhere server response [Classification: Misc
activity] [Priority: 3]: {UDP} 10.0.0.152:32857 -> 172.16.0.27:5632
22:17:11 [1:1417:2] SNMP request udp [Classification: Attempted Information Leak]
[Priority: 2]: {UDP} 10.0.0.152:32861 -> 172.16.0.27:161
22:17:11 [1:237:1] DDOS Trin00:MastertoDaemon(defaultpassdetected!)
[Classification: Attempted Denial of Service] [Priority: 2]: {UDP} 10.0.0.152:1024
-> 172.16.0.27:27444
```

Viewing the Snort alert and scan logs demonstrated that it had indeed been able to watch the wireless network and detect malicious activity. Based upon this proof of concept this implementation can be expanded to use any of the popular add-in programs to Snort such as SnortSnarf,[24] ACID[25], and many others.

Conclusion

Although our homegrown Snort AirNIDS cannot detect the sophisticated attacks that specifically target the 802.11b protocol, these might be added in the future in the form of a plug-in or preprocessor. Commercial products that perform wireless intrusion detection will become increasingly prevalent as wireless networks continue to expand their penetration in both the corporate enterprise and service provider networks. The risk of profit margin loss or degradation of service is too great.

General References:

Is Your Wireless Network Secure? Ken Hodges
http://rr.sans.org/wireless/wireless_net2.php

Wireless Risks and Defenses White Paper, AirDefense Inc. smartin@airdefense.net

AirDefense Products, http://www.airdefense.net/products/index.shtm

Information Security Magazine, Hot Picks July 2002,
http://www.infosecuritymag.com/2002/jul/hotpick.shtml

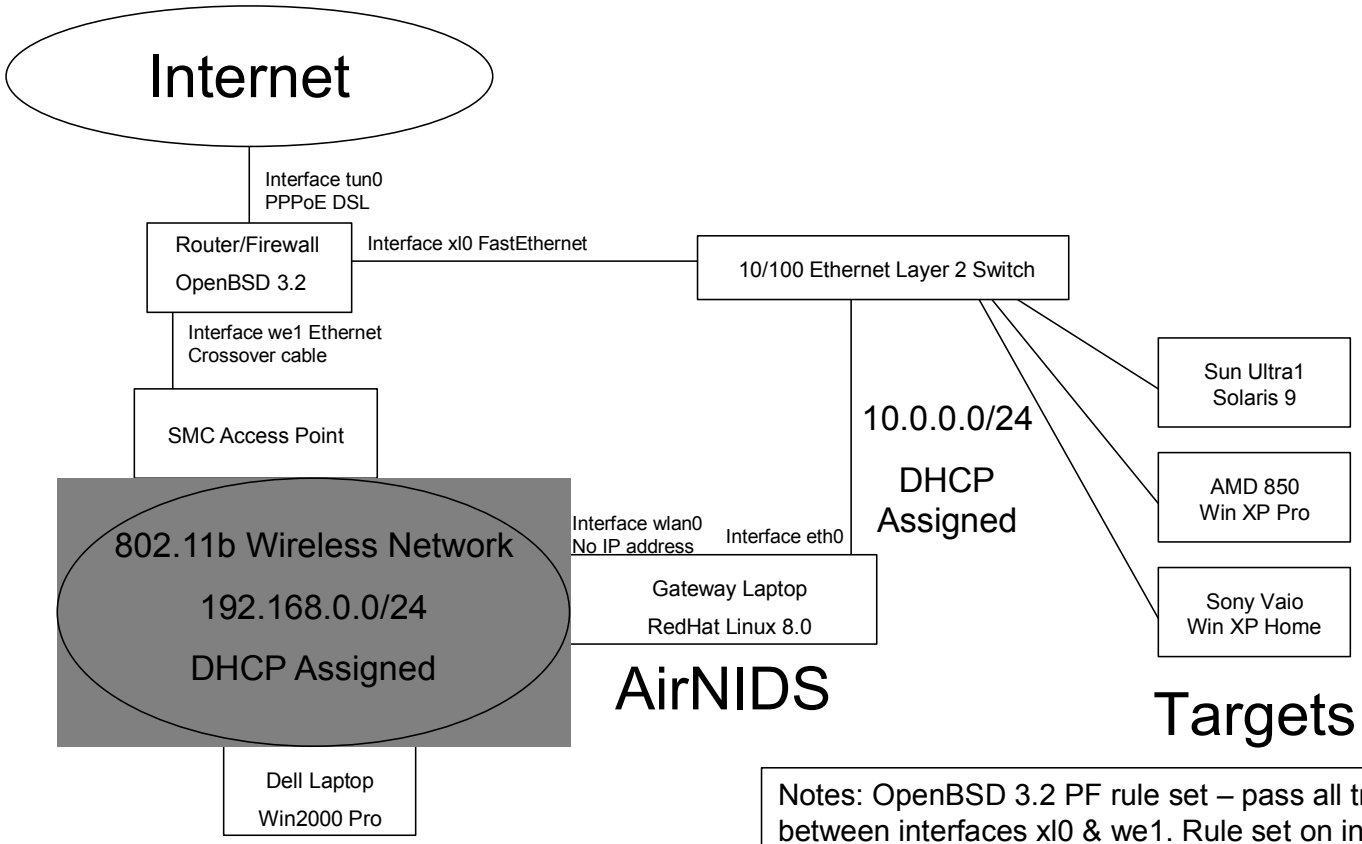
Advanced 802.11 Attack, Black Hat Briefings 2002 Las Vegas, NV, Mike Lynn & Robert Baird, http://802.11ninja.net/

Using Snort as an IDS and Network Monitor in Linux, James Kipp,
http://rr.sans.org/intrusion/monitor.php

[24] Hoagland,James;Staniford,Stuart; McAlerney,Joe. SnortSnarf. URL: http://www.silicondefense.com/software/snortsnarf/index.htm

[25] Danyliw, Roman. ACID URL: http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html

AirNIDS Proof of Concept Network Diagram



Notes: OpenBSD 3.2 PF rule set – pass all traffic between interfaces xl0 & we1. Rule set on interface tun0 allows inbound tcp/22 and keeps statefulness on all outbound IP connections. This box also provides DHCP services to both network segments.

Network

Detect #1

1.1 Source of trace:

Trace was obtained from http://www.incidents.org/logs/Raw/2002.6.6. Network layout of the source network is unknown.

```
Snort packet decodes:
07/06-01:14:13.694488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x3C
255.255.255.255:31337 -> 46.5.64.15:515 TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 00 2B 00 00 00 00 0E 06 45 C1 FF FF FF FF 2E 05  .+......E.......
0x0020: 40 0F 7A 69 02 03 00 00 00 00 00 00 00 00 50 14  @.zi..........P.
0x0030: 00 00 F9 E8 00 00 63 6B 6F 00 00 00              ......cko...

Packets with similar source IP/port, destination port, and payload:
Time                        Source IP
07/06-02:50:01.694488       46.5.102.189
07/06-03:28:49.714488       46.5.149.184
07/06-05:57:01.754488       46.5.200.194
07/06-18:28:19.134488       46.5.223.69

07/06-21:38:01.124488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x3C
255.255.255.255:31337 -> 46.5.135.239:515 TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 00 2B 00 00 00 00 0E 06 FC DF FF FF FF FF 2E 05  .+..............
0x0020: 87 EF 7A 69 02 03 00 00 00 00 00 00 00 00 50 14  ..zi..........P.
0x0030: 00 00 B1 07 00 00 63 6B 6F 00 00 00              ......cko...
```

1.2 Detect was generated by:

Snort 1.9.0 with all rules enabled. Specific detect was generated by the backdoor.rules file. The triggering rule is for BACKDOOR Q access.

alert tcp 255.255.255.0/24 any -> $HOME_NET any (msg:"BACKDOOR Q access"; flags:A+; dsize: >1; reference:arachnids,203; sid:184; classtype:misc-activity; rev:3;)

1.3 Probability the source address was spoofed:

The source IP address is all ones (255.255.255.255 – decimal representation) which is the network broadcast IP address. This is not a valid source address. These packets are very likely spoofed and crafted. Since the network layout where this detect occur is not known, it can not be determined if these packets came from the Internet or were created by an insider.

1.4 Description of attack:

The source port of this attack is 31137. This port is popular with hackers and script kiddies. Port 31337 commonly referred to as the "elite" port. Many backdoor programs such as BackOrifice use this as a default port. The TCP sequence number is consistently zero, which is a very strong indication that the packet is crafted. The intent of these packets is to target network printers offering LPR (UNIX line printer remote) daemon. This may be a remote Denial of Service (DoS) attack against a certain brand of network printers or Unix print servers. This attack may cause the printer to go offline and require a power cycle before being operational for legitimate users print jobs. On a UNIX print server, it may cause the LPR daemon to crash. Even though Snort suggests that is backdoor access, the source IP address of network broadcast would repudiate that. The payload does not reveal any apparent imbedded commands for a trojan.

1.5 Attack mechanism:

The attack uses a TCP packet that has both the ACK and RESET flags set. There is no evidence of an attempted or completed TCP 3-way handshake. Since the attack is using a broadcast source IP address, no 3-way handshake could have been established. It is not possible by the very nature of TCP connections, which must be between two distinct hosts. TCP connections are like "virtual circuits" and are produced between two hosts by the TCP implementation in the IP stack. Crafted packets such as these bypass that mechanism and directly assemble the packet before it is passed down to the OSI data link layer.

1.6 Correlations:

This attack can be correlated with the GCIA Practical of Trenton Riddell (http://www.giac.org/practical/Trenton_Riddell_GCIA.doc).

In his analysis, he describes his network layout indirectly. He describes that the targeted hosts are valid IP addresses and that NAT is being used on the firewall. His firewall logs confirmed that no outbound connections had been attempted, nor would they have been permitted by the firewall rules. In this network configuration, it can be stated that this traffic is indeed a stimulus.

A Google search revealed more correlation: (Hyperlink to Google query) –

http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00288.html - Detect with similar payload and the same TCP flags coming for a source address of 255.255.255.255.

http://online.securityfocus.com/archive/19/187958 - A network administrator from Australia reported that he is seeing packets from 255.255.255.255 against various target IPs in his networks. Source and destination ports match our detect. He configured his router to drop inbound packets.

http://maclux-rz.uibk.ac.at/~maillists/focus-ms/msg01549.shtml - joem@nist.gov reports that he has seen these packets for the last week and about twenty in an eight hour period.

1.7 Evidence of active targeting:

The attack was directed at six different hosts within twenty hours. Other sites experiencing this attack received this stimulus against multiple targets.

1.8 Severity:

Criticality = 2 – Network printer servers are needed in daily business operations.
Lethality = 1 – Correlation does not demonstrate the use of buffer overflow exploit.
System Countermeasures = 2 – Host should only accept connections from authorized networks.
Network Countermeasures = 1 – Packets with broadcast source IP should be dropped by the perimeter firewall or router ACL.

Severity = (2 + 1) – (2+1) = 0

1.9 Defensive Recommendations:

If these packets are entering the network perimeter from the Internet, these should be blocked using an access control list (ACL) on a router or a firewall. Ideally lpr traffic should not be permitted to pass through the network perimeter. If lpr is a necessary business processes, a tight firewall rule restricting lpr by source and destination IP address will mitigate the risk of this attack against the need of the business process.

Contact the system/network administrators of these six IP address. Request that administrator provide make, model, and firmware version of these network printers, if they are indeed network printers. Contact the vendor or look at their website to determine if there is any vulnerability associated with the equipment and/or specific firmware revisions.

1.10 Multiple choice question:

IP packets coming from the Internet with a source IP address of 255.255.255.255 are:

A. Coming from someone's workstation at IANA.
B. Most likely crafted and should be stopped at the network perimeter.
C. Streaming multi-media TCP return traffic multicast packets.
D. Most likely a side effect of an improperly configured host

Answer: B – Packets with a source IP address of all ones are most like crafted and should not be permitted to pass through the network perimeter security devices such as a firewall.

1.11 Incidents.org Feedback (My responses in italics)

From: Robert Wagner <rwagner@eruces.com>
To: "'Thomas Hoffecker'" <thomas@hoffecker.com>, intrusions@incidents.org
Subject: RE: LOGS: GIAC GCIA Version 3.2 Practical Detect(s)
Date: Mon, 11 Nov 2002 08:13:29 -0600

Is allowing traffic going to LPR or allowing traffic with a source IP of 255.255.255.255 more dangerous? I think you missed something.

My Response:
Robert,

Traffic from 255.255.255.255 should not be permitted to enter through the network perimeter from the Internet at all, regardless of destination port. Of the two, that is more dangerous. However, I would still stand firm with my defensive recommendation that 515 should not be allowed in from just anywhere on the Internet.

Thomas

From: Robert Wagner rwagner@eruces.com
To: "'Thomas Hoffecker'" thomas@hoffecker.com
Subject: RE: LOGS: GIAC GCIA Version 3.2 Practical Detect(s)
Date: Thu, 14 Nov 2002 15:53:47 -0600

I would agree.

From: Bryce_Alexander@Vanguard.com
To: Thomas Hoffecker thomas@hoffecker.com
Date: Mon, 11 Nov 2002 09:11:48 -0700
Subject: Re: LOGS: GIAC GCIA Version 3.2 Practical Detect(s) Hoffecker

At one point, in the description of attack you state that this may be a DOS of a UNIX LPR, then later you mention in the correlations section that this appears to be a stimulus packet. For certification you should take a stance and tell us which one you believe it is. If this is a DOS can you name the version of LPR that is vulnerable to this type of attack? Or what tool is generating this?

I am also going to challenge you to think outside the box, if it is a stimulus packet, does it have to be directed to a

specific service? Just because LPR uses port 515 in a well behaved world, could a person who is not well behaved write something to listen on that port that isn't an LPR service?

You seem to reject Q as a possibility, have you researched Q to see what it's characteristics are? Is it possible that Q may respond in some way when it sees this type of packet?

You stated that "Crafted packets such as these bypass that mechanism and directly assemble the packet before it is passed down to the OSI data link layer." It may just be me, but I am missing something here, could you expand a little on what you mean here? How about a description of what you believe a normal IP stack would do with this packet, step by step.

My Response:
Bryce,
See my responses in line. Appreciate any comments.

Thomas

At 05:11 PM 11/11/2002, you wrote:

At one point, in the description of attack you state that this may be a DOS of a UNIX LPR, then later you mention in the correlations section that this appears to be a stimulus packet. For certification you should take a stance and tell us which one you believe it is. If this is a DOS can you name the version of LPR that is vulnerable to this type of attack? Or what tool is generating this?

If the attacker is performing a DoS it is not at the level of creating the desired effect. In correlation with another GCIA paper, it was shown that these packets were unsolicited, so this is someone just throwing junk packets out at various IP networks.

I am also going to challenge you to think outside the box, if it is a stimulus packet, does it have to be directed to a specific service? Just because LPR uses port 515 in a well behaved world, could a person who is not well behaved write something to listen on that port that isn't an LPR service?

Yes, absolutely, someone could write something and have it listen on tcp/515. Any port can have any daemon listening, it's matter of configuration. TCP/515 is just the IANA standardized port for lpr.

You seem to reject Q as a possibility, have you researched Q to see what it's characteristics are? Is it possible that Q may respond in some way when it sees this type of packet? You stated that "Crafted packets such as these bypass that mechanism and directly assemble the packet before it is passed down to the OSI data link layer." It may just be me, but I am missing something here, could you expand a little on what you mean here? How about a description of what you believe a normal IP stack would do with this packet, step by step.

*I was referring to using something like the tools Nemesis (http://jeff.wwti.com/nemesis/) on the unix platforms or RafaleX (**http://www.packx.net/packx/html/en/index-en.htm**) for some Windows platforms. Packets such as these are not produced via the normal OS API, but via the use of RawSockets. I'm not a programmer, so this isn't an area of expertise. My point being is that these packets were not created by someone with an improperly configured machine trying to print somewhere.*

The IP stack in the targets will not send a response to these packets. However, I would argue that any traffic directed at a target can be viewed as a stimulus. The stimulus may not elicit the machine to respond across the network. Thinking out of the box, one could use packets as these for a covert channel to direct a machine to do something.

Network Detect #2

2.1 Source of trace:

Trace was obtained from http://www.incidents.org/logs/Raw/2002.6.5 . Network layout of the source network is unknown.

```
Snort packet decodes:
07/05-03:31:17.244488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A6
64.216.218.242:4038 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12210 IpLen:20 DgmLen:408 DF
***AP*** Seq: 0xF6DC817D  Ack: 0x46D3F449  Win: 0x2238  TcpLen: 20
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 01 98 2F B2 40 00 6F 06 E2 5E 40 D8 DA F2 2E 05  ../.@.o..^@.....
0x0020: B4 85 0F C6 00 50 F6 DC 81 7D 46 D3 F4 49 50 18  .....P...}F..IP.
0x0030: 22 38 6D 00 00 00 47 45 54 20 2F 63 67 69 2D 62  "8m...GET /cgi-b
0x0040: 69 6E 2F 66 6F 72 6D 6D 61 69 6C 2E 63 67 69 3F  in/formmail.cgi?
0x0050: 72 65 63 69 70 69 65 6E 74 3D 67 65 74 72 61 66  recipient=getraf
0x0060: 72 61 6E 6B 65 6C 40 66 6C 61 73 68 6D 61 69 6C  rankel@flashmail
0x0070: 2E 63 6F 6D 26 73 75 62 6A 65 63 74 3D 70 6C 65  .com&subject=ple
0x0080: 61 73 65 20 74 61 6B 65 20 61 20 6C 6F 6F 6B 20  ase take a look
0x0090: 61 74 20 79 6F 75 72 20 6F 70 65 6E 20 66 6F 72  at your open for
0x00A0: 6D 6D 61 69 6C 26 65 6D 61 69 6C 3D 67 65 74 72  mmail&email=getr
0x00B0: 61 67 72 74 65 40 66 6C 61 73 68 6D 61 69 6C 2E  agrte@flashmail.
0x00C0: 63 6F 6D 26 3D 68 74 74 70 3A 2F 2F 77 77 77 2E  com&=http://www.
0x00D0: 58 58 58 58 2E 63 6F 6D 2F 63 67 69 2D 62 69 6E  XXXX.com/cgi-bin
0x00E0: 2F 66 6F 72 6D 6D 61 69 6C 2E 63 67 69 20 2F 66  /formmail.cgi /f
0x00F0: 6F 72 6D 6D 61 69 6C 2E 63 67 69 20 48 54 54 50  ormmail.cgi HTTP
0x0100: 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 69 6D  /1.1..Accept: im
0x0110: 61 67 65 2F 67 69 66 2C 20 69 6D 61 67 65 2F 78  age/gif, image/x
0x0120: 2D 78 62 69 74 6D 61 70 2C 20 69 6D 61 67 65 2F  -xbitmap, image/
0x0130: 6A 70 65 67 2C 20 69 6D 61 67 65 2F 70 6A 70 65  jpeg, image/pjpe
0x0140: 67 2C 20 2A 2F 2A 0D 0A 55 73 65 72 2D 41 67 65  g, */*..User-Age
0x0150: 6E 74 3A 20 4D 69 63 72 6F 73 6F 66 74 20 55 52  nt: Microsoft UR
0x0160: 4C 20 43 6F 6E 74 72 6F 6C 20 2D 20 36 2E 30 30  L Control - 6.00
0x0170: 2E 38 38 36 32 0D 0A 48 6F 73 74 3A 20 77 77 77  .8862..Host: www
0x0180: 2E 58 58 58 58 2E 63 6F 6D 0D 0A 43 61 63 68 65  .XXXX.com..Cache
0x0190: 2D 43 6F 6E 74 72 6F 6C 3A 20 6E 6F 2D 63 61 63  -Control: no-cac
0x01A0: 68 65 0D 0A 0D 0A                                he....

Packets with same payload:
07/05-03:31:17.284488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A4
64.216.218.242:4039 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12213 IpLen:20 DgmLen:406 DF
***AP*** Seq: 0xF6DD300E  Ack: 0x470904F8  Win: 0x2238  TcpLen: 20

07/05-03:31:26.274488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A8
64.216.218.242:4057 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12455 IpLen:20 DgmLen:410 DF
***AP*** Seq: 0xF6F90AF9  Ack: 0x476E55E2  Win: 0x2238  TcpLen: 20
```

```
07/05-03:31:35.374488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1AA
64.216.218.242:4083 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12650 IpLen:20 DgmLen:412 DF
***AP*** Seq: 0xF7328CE3  Ack: 0x48A68AAA  Win: 0x2238  TcpLen: 20

07/05-03:31:35.454488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A2
64.216.218.242:4085 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12654 IpLen:20 DgmLen:404 DF
***AP*** Seq: 0xF733B62E  Ack: 0x47F2283A  Win: 0x2238  TcpLen: 20

07/05-03:31:38.654488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A2
64.216.218.242:4085 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12714 IpLen:20 DgmLen:404 DF
***AP*** Seq: 0xF733B62E  Ack: 0x47F2283A  Win: 0x2238  TcpLen: 20

07/05-03:31:45.644488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A4
64.216.218.242:4115 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12913 IpLen:20 DgmLen:406 DF
***AP*** Seq: 0xF76D7EED  Ack: 0x48778EA8  Win: 0x2238  TcpLen: 20

07/06-00:41:50.794488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x16B
64.252.130.205:3821 -> 46.5.180.133:80 TCP TTL:46 TOS:0x0 ID:38704 IpLen:20 DgmLen:349 DF
***AP*** Seq: 0x4234C66  Ack: 0x5C706F4  Win: 0xFAF0  TcpLen: 20
```

2.2 Detect was generated by:

Snort 1.9.0 with all rules enabled. This detect was found indirectly, while analyzing raw packets of a BIND version attack and nmap TCP scan in this trace. Analyst curiosity was trigger because of similar payload in the http data stream.

2.3 Probability the source address was spoofed:

These packets were not spoofed. Although the network trace does not show the initiation of a 3-way handshake, it appears that a valid http connection was made because the attack is attempting to execute a CGI script on the targeted web server. This could not occur unless a valid TCP connection had been established.

2.4 Description of attack:

The target of the attack is a web server running a Perl based CGI script that allows the user to fill-out an online form. The resulting data of the form is then emailed to the form's intended recipient. This script was originally written by Matt Wright and it is part of a suite of CGI scripts at http://www.scriptarchive.com/. Several newer variations and code forks of these scripts exist that have addressed past security vulnerabilities.

2.5 Attack mechanism:

This attack can be performed with any web browser. However, by automating the process with a script this vulnerability can allow an unauthorized user to send email via a third party (the targeted web server). The SMTP headers would not reveal the IP address or SMTP header data of the true originator of the spam. If a recipient reports this spam to the website operator's ISP, the web server will appear to be the originator of the spam. The executing origin IP address of the Perl CGI script can be determined by examination of the web server access logs.

The user agent that is executing this attack is not a normal web browser such as Internet Explorer or Netscape. To make spamming using this attack scalable, a script is necessary. Research into the "Microsoft URL Control Agent" did not provide any conclusive information as to what functionality this agent provides. There were many references to this http agent appearing in web server logs. A list serve archived message did speculate that this might be a Microsoft Proxy server. A search of the Microsoft Knowledge Base did not provide conclusive information.

The use of this attack though a 3rd party proxy server would further hide the identity of the spammer. The Microsoft Proxy Server logs would reveal the true IP address of the spammer. The spammer has done this with the intention to make tracing the origin much more difficult. Tracing this spam would now require the assistance of the abused web server owner, the owner of the Microsoft Proxy Server, and the abuse department of the ISP whose customer made use of the open proxy server and executed this attack.

2.6 Correlations:

This specific attack can be correlated with similar "well intentioned" payload to alert the system administrator that the server is vulnerable to spamming. http://www.securityfocusonline.com/archive/119/280123/2002-07-01/2002-07-07/0

A Google search revealed more correlation: (Hyperlink to Google query)

http://www.webmasterworld.com/forum11/1005.htm - Webmaster World discussion forum - This correlation specifically mentions that it appears that the URL Control Agent is being used to search for web servers running vulnerable versions of formmail. It also gives some basic steps to better secure your CGI scripts.

http://www.geocrawler.com/archives/3/4890/2002/2/0/7971692/ - Snort users mailing list archive – This correlation specifically states that spamming using this vulnerability is on the rise "almost like Sanford Wallace is back."

http://www.impressions.com.my/watch/annual/2002/01/01/cgi.html - This URL shows attempted abuses of a website's formmail script. Several variations of the "well intentioned" payload exist.

http://cert.uni-stuttgart.de/archive/honeypots/2002/07/msg00009.html - SecurityFocus Honeypot mailing list – More correlation of the "well intentioned" payload.

Public vulnerability alerts of formmail being susceptible to spamming:
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0357
http://online.securityfocus.com/bid/3955

2.7 Evidence of active targeting:

The attacker has already determined that a vulnerable version of this script is available for abuse on this web server. The attacker is abusing the CGI script to send email to getrafrankel@flashmail.com advising that the version of formmail on their web server is vulnerable to exploitation. Although this may seem to be a public service, it could be possibly interpreted as abuse & theft of computing resources. In the past, open SMTP relays were often identified by testing them for abuse and then making this data available to system administrators so that they could reject or filter email from those specific SMTP servers. The practice of using this exploit for such purposes would probably be viewed in a similar manner by the Internet community.

2.8 Severity:

Criticality = 5 – The web server is probably part of critical business infrastructure.
Lethality = 3 – Although not much damage can be done to the web server at very low volumes of this attack, at a high volume rate the server may experience high CPU and network utilization.
System Countermeasures = 2 – CGI scripts can lead to potential compromises since they are executed by the web server and not the client's web browser. The system administrator should check frequently for any new published vulnerabilities on any third party CGI scripts that are running on the web server.

Network Countermeasures = 3 - Since the exact configuration of the network and web server is unknown, determination is not easily made. HTTP packets must reach this web server so that it can fulfill its function.

Severity = (5 + 3) – (2 + 3) = 3

2.9 Defensive Recommendations:

Frequently check for new vulnerabilities of any CGI scripts used on the web server. Have the web server relay all SMTP via another mail server. This mail server can be restricted to only allow internal recipients to receive email generated by the formmail script. This would not prevent abuse of this script, however, it would prevent the spam from reaching the intended victim.

Consider replacing the original Matt Wright script with one from http://sourceforge.net/projects/nms-cgi/. On his own website Matt Wright recommends the nms versions of his scripts (http://www.scriptarchive.com/nms.html). He states the original scripts were written by him in 1995 while still in high school and do not reflect current CGI programming practices.

2.10 Multiple choice question:

Spam originating from the web server (46.5.180.133) could have been proactively prevented by:

```
07/05-03:31:17.244488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x1A6
64.216.218.242:4038 -> 46.5.180.133:80 TCP TTL:111 TOS:0x0 ID:12210 IpLen:20 DgmLen:408 DF
***AP*** Seq: 0xF6DC817D  Ack: 0x46D3F449  Win: 0x2238  TcpLen: 20
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 01 98 2F B2 40 00 6F 06 E2 5E 40 D8 DA F2 2E 05  ../.@.o..^@.....
0x0020: B4 85 0F C6 00 50 F6 DC 81 7D 46 D3 F4 49 50 18  .....P...}F..IP.
0x0030: 22 38 6D 00 00 00 47 45 54 20 2F 63 67 69 2D 62  "8m...GET /cgi-b
0x0040: 69 6E 2F 66 6F 72 6D 6D 61 69 6C 2E 63 67 69 3F  in/formmail.cgi?
0x0050: 72 65 63 69 70 69 65 6E 74 3D 67 65 74 72 61 66  recipient=getraf
0x0060: 72 61 6E 6B 65 6C 40 66 6C 61 73 68 6D 61 69 6C  rankel@flashmail
0x0070: 2E 63 6F 6D 26 73 75 62 6A 65 63 74 3D 70 6C 65  .com&subject=ple
0x0080: 61 73 65 20 74 61 6B 65 20 61 20 6C 6F 6F 6B 20  ase take a look
0x0090: 61 74 20 79 6F 75 72 20 6F 70 65 6E 20 66 6F 72  at your open for
0x00A0: 6D 6D 61 69 6C 26 65 6D 61 69 6C 3D 67 65 74 72  mmail&email=getr
0x00B0: 61 67 72 74 65 40 66 6C 61 73 68 6D 61 69 6C 2E  agrte@flashmail.
0x00C0: 63 6F 6D 26 3D 68 74 74 70 3A 2F 2F 77 77 77 2E  com&=http://www.
0x00D0: 58 58 58 58 2E 63 6F 6D 2F 63 67 69 2D 62 69 6E  XXXX.com/cgi-bin
0x00E0: 2F 66 6F 72 6D 6D 61 69 6C 2E 63 67 69 20 2F 66  /formmail.cgi /f
0x00F0: 6F 72 6D 6D 61 69 6C 2E 63 67 69 20 48 54 54 50  ormmail.cgi HTTP
0x0100: 2F 31 2E 31 0D 0A 41 63 63 65 70 74 3A 20 69 6D  /1.1..Accept: im
```

```
0x0110: 61 67 65 2F 67 69 66 2C 20 69 6D 61 67 65 2F 78  age/gif, image/x
0x0120: 2D 78 62 69 74 6D 61 70 2C 20 69 6D 61 67 65 2F  -xbitmap, image/
0x0130: 6A 70 65 67 2C 20 69 6D 61 67 65 2F 70 6A 70 65  jpeg, image/pjpe
0x0140: 67 2C 20 2A 2F 2A 0D 0A 55 73 65 72 2D 41 67 65  g, */*..User-Age
0x0150: 6E 74 3A 20 4D 69 63 72 6F 73 6F 66 74 20 55 52  nt: Microsoft UR
0x0160: 4C 20 43 6F 6E 74 72 6F 6C 20 2D 20 36 2E 30 30  L Control - 6.00
0x0170: 2E 38 38 36 32 0D 0A 48 6F 73 74 3A 20 77 77 77  .8862..Host: www
0x0180: 2E 58 58 58 58 2E 63 6F 6D 0D 0A 43 61 63 68 65  .XXXX.com..Cache
0x0190: 2D 43 6F 6E 74 72 6F 6C 3A 20 6E 6F 2D 63 61 63  -Control: no-cac
0x01A0: 68 65 0D 0A 0D 0A                                he....
```

A. Setting proper file permissions on cgi-bin directory.
B. Using a SMTP relay that only accepts mail for local users.
C. Changing the GET http command to POST in the script.
D. Insuring that only the latest version of the script is used.

Answer: B – If the email is sent to an SMTP relay, the relay can be restricted to only deliver email to local recipients. Any email destined for external email addresses could be deleted. This would prevent spamming from occurring, even if a different exploit is found in the future. This would not prevent spamming against internal users, it would keep the web server from being a spam relay and spare the organization public embarrassment.

2.11 Incidents.org Feedback (My responses in italics)

From: Robert Wagner rwagner@eruces.com
To: "'Thomas Hoffecker'" <thomas@hoffecker.com>, intrusions@incidents.org
Subject: RE: LOGS: GIAC GCIA Version 3.2 Practical Detect(s)
Date: Mon, 11 Nov 2002 08:10:28 -0600

Do you have any evidence that the attack worked? Lethality - This can be damaging if this is also a mail server: About the time the server sends 100,000 spam messages and you find your server blacklisted. Not to mention the complaints. Formmail has some features that allow you to prevent this sort of thing. It may be a good time to review the configuration of formmail. Another thought is to prevent outbound mail from this server (if it is not also the mail server).

My response:
Robert,

Thanks for your reply to my detect posting.

There was no evidence that the attack was successful. It could have been, but I would make the case that it was not because of the volume of these attempts. This was someone attempting to look for a vulnerable formmail. If it had been vulnerable, I believe more attempts would have been seen (i.e. spammers exploiting this box). As a counterpoint, if this email was successfully relayed via formmail, I highly doubt that the Snort box on this network would have detected and logged it since the IDS would have no real way of determining if it was spam or legit email being sent out. A firewall or router ACL might have prevented it if this server wasn't permitted to send SMTP beyond the network perimeter.

I totally agree this could be a Lethality of 5 if this is also a mail server. There was no evidence of any inbound or outbound SMTP activity to that IP address in the trace. It would definitely not be a best practice to put your mail and web server on the same box if either are mission critical.

The review of the formmail config would need to be performed in conjunction with the web server admin and/or webmaster. I consider that a given as part of a follow-up with server admin along with a review of the http logs. I saw defensive recommendations as generally specific things that can be done to prevent these type of attacks from happening. However, point well taken, proper configuration of the CGI script and verification during/after an incident such as this is a must and my analysis should have stated so.

The web server will need to have outbound mail at some level, at least to other internal mail servers so that formmail can fulfill its intended purpose. Restricting the web server's ability to open SMTP connections beyond the network perimeter is definitely recommended. Hence, my defense recommendation to relay all SMTP via another server. As you stated, that won't work well if this server is also being used as a mail server.

Appreciate any further comments and I'll reply to your other emails tomorrow since I'm UTC+1 and it's getting late.

Thomas Hoffecker

Network Detect #3

3.1 Source of trace:

Trace was obtained from http://www.incidents.org/logs/Raw/2002.6.6. Network layout of the source network is unknown.

```
07/06-02:49:25.874488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x4A
64.228.107.189:56697 -> 46.5.165.181:1080 TCP TTL:50 TOS:0x0 ID:53552 IpLen:20 DgmLen:60 DF
******S* Seq: 0xAA33C853  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 125245795 0 NOP WS: 0
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 00 3C D1 30 40 00 32 06 FD 35 40 E4 6B BD 2E 05  .<.0@.2..5@.k...
0x0020: A5 B5 DD 79 04 38 AA 33 C8 53 00 00 00 00 A0 02  ...y.8.3.S......
0x0030: 16 D0 41 CE 00 00 02 04 05 B4 04 02 08 0A 07 77  ..A............w
0x0040: 19 63 00 00 00 00 01 03 03 00                    .c........

Further packets with similar payloads & alerts (edited for brevity).
07/06-02:49:28 64.228.107.189:56697 -> 46.5.165.181:1080 TCP ID:53553 Seq: 0xAA33C853
07/06-02:49:35 64.228.107.189:56697 -> 46.5.165.181:1080 TCP ID:53554 Seq: 0xAA33C853
07/06-02:49:36 64.228.107.189:58085 -> 46.5.165.181:1080 TCP ID:30598 Seq: 0xAAF36ACD
07/06-02:49:39 64.228.107.189:58085 -> 46.5.165.181:1080 TCP ID:30599 Seq: 0xAAF36ACD
07/06-02:49:44 64.228.107.189:58085 -> 46.5.165.181:1080 TCP ID:30600 Seq: 0xAAF36ACD
07/06-02:49:46 64.228.107.189:59380 -> 46.5.165.181:1080 TCP ID:7605  Seq: 0xAB523C86
07/06-02:49:48 64.228.107.189:59380 -> 46.5.165.181:1080 TCP ID:7606  Seq: 0xAB523C86
07/06-02:49:55 64.228.107.189:59380 -> 46.5.165.181:1080 TCP ID:7607  Seq: 0xAB523C86
07/06-02:49:55 64.228.107.189:32793 -> 46.5.165.181:1080 TCP ID:32536 Seq: 0xABFB5A03
07/06-02:49:58 64.228.107.189:32793 -> 46.5.165.181:1080 TCP ID:32537 Seq: 0xABFB5A03
07/06-02:50:04 64.228.107.189:32793 -> 46.5.165.181:1080 TCP ID:32538 Seq: 0xABFB5A03
07/06-02:50:05 64.228.107.189:34084 -> 46.5.165.181:1080 TCP ID:16742 Seq: 0xAC366865
07/06-02:50:08 64.228.107.189:34084 -> 46.5.165.181:1080 TCP ID:16743 Seq: 0xAC366865
07/06-02:50:15 64.228.107.189:34084 -> 46.5.165.181:1080 TCP ID:16744 Seq: 0xAC366865
07/06-02:50:15 64.228.107.189:35203 -> 46.5.165.181:1080 TCP ID:52599 Seq: 0xACBBE19C
07/06-02:50:18 64.228.107.189:35203 -> 46.5.165.181:1080 TCP ID:52600 Seq: 0xACBBE19C
07/06-02:50:24 64.228.107.189:35203 -> 46.5.165.181:1080 TCP ID:52601 Seq: 0xACBBE19C

Alert created by Snort.
[**] [1:615:3] SCAN SOCKS Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
07/06-02:49:25.874488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x4A
64.228.107.189:56697 -> 46.5.165.181:1080 TCP TTL:50 TOS:0x0 ID:53552 IpLen:20 DgmLen:60 DF ******S* Seq: 0xAA33C853  Ack: 0x0
Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 125245795 0 NOP WS: 0
[Xref => url help.undernet.org/proxyscan/]
```

3.2 Detect was generated by:

Snort 1.9.0 with all rules enabled. Specific detect was generated by the scans.rules file. The triggering rule is for detecting scans for SOCKS proxy servers.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 1080 (msg:"SCAN SOCKS Proxy attempt"; flags:S; reference:url,help.undernet.org/proxyscan/;
classtype:attempted-recon; sid:615; rev:3;)
```

3.3 Probability the source address was spoofed:

Since the attacker is looking for open proxies the probability that the source is spoofed is very low. The volume of the scan rate in this trace file does not demonstrate intent to create a Denial of Service (DoS) against the targeted IP. This attack has only a SYN flag set, so the attacker is attempting to initiate the 3-way handshake to determine if tcp/1080 is listening.

3.4 Description of attack:

This attack is a scan for hosts with tcp/1080 listening. This appears to be a scan using a simple TCP port scanner. It appears that each connection attempt is being created naturally by the IP stack and the packets are not crafted, although this cannot be conclusively determined by the sequence number. Six separate connection attempts are made and three retransmissions are made for each one. The retransmit timings fall within the TCP retransmission parameters. The first three connection attempts are using source ports in the mid-high fifty thousand range. The second three attempts are from source ports in the low-mid thirty thousand range. The attacker may be conducting other scans that cause the source port to change into the thirty thousand range. This does not facilitate any conclusion, but it may be useful if scans can be correlated with other targeted organizations.

3.5 Attack mechanism:

It appears from the scan that a simple TCP port scanner is being used since this is not a very stealthy port scan. The attacker is simply doing SYN scans to find hosts that respond with a SYN-ACK in order to complete the second step of the TCP connection process. This may be someone using a tool such as ProxyHunter (ProxyHunter Website).

3.6 Correlations:

The source IP address report at DShield.org reveals that there have been reports of this IP targeting six hosts. However, this is fairly inconclusive since the IP address resolves in DNS to a PPP dialup. http://www.dshield.org/ipinfo.php?ip=064.228.107.189

A Google search for correlation of other scan victims(Hyperlink to Google search).

http://www.sans.org/y2k/062200.htm - Correlation of scanning for tcp/1080 during June 2000.

http://cert.uni-stuttgart.de/archive/incidents/2000/01/msg00172.html - Explanation of why someone would want to connect and abuse an open proxy.

http://rdweb.cns.vt.edu/~lat/log_archives/020423.txt - Correlation of scanning for SOCKS proxies being detected by Snort and PortSentry.

http://help.undernet.org/proxyscan.htm - Undernet.org description of scanning a user's host when login into their IRC server.

3.7 Evidence of active targeting:

The system was targeted six times within one minute. Although this could be a simple wrong number, it is unlikely. The trace file covers a time period of nearly twenty-four hours. No other evidence of SOCKS proxy scans was found. This may be a low-and-slow scan that would only be detectable over a longer period of time. In most cases, scans for tcp/1080 are done to find unprotected SOCKS proxies that hackers can use as open proxy servers for cloaking their activity.

3.8 Severity:

Criticality = 1 – Not easily determined with available data. Many IRC servers scan users shortly after login for open proxies to prevent proxy servers from being abused. The source IP is a PPP dialup in Canada according to reverse DNS (Toronto-ppp221626.sympatico.ca).
Lethality = 2 – System probably cannot be compromised unless the SOCKS proxy is actually on the host and is vulnerable to an exploit.

System Countermeasures = 3 – Not easily determined. There is no evidence in the trace file that the targeted host was abused as a proxy.
Network Countermeasures = 5 – Based upon the lack of a response by the target host and retransmissions of each connection attempt a firewall or router is most likely preventing inbound tcp/1080 connections.

Severity = (1 + 2) – (3 + 5) = -5

3.9 Defensive Recommendations:

Inbound tcp/1080 connections should not be allowed to pass through the perimeter network security devices. It appears that such a defense measure may already be in place based up on TCP retransmission of the connection attempts.

3.10 Multiple choice question:

This Snort rule would detect:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 1080 (msg:"SCAN SOCKS Proxy attempt"; flags:S; reference:url,help.undernet.org/proxyscan/;
classtype:attempted-recon; sid:615; rev:3;)
```

A. The SYN packet of user accessing a public http server.
B. An inbound scan for a SOCKS proxy.
C. A TCP connection with the source port of 1080.
D. An attack against a host on the $EXTERNAL_NET.

Answer B – This rule detects inbound scans against tcp/1080. This port is used for the SOCKS proxy daemon.

3.11 Incidents.org Feedback (My responses in italics)

From: Robert Wagner <rwagner@eruces.com>
To: "'Thomas Hoffecker'" <thomas@hoffecker.com>, intrusions@incidents.org
Subject: RE: LOGS: GIAC GCIA Version 3.2 Practical Detect(s)

You mention that you think this is already being blocked (1080). How can you determine this without the flags on the packets? Maybe a really short test session?

My response:
Robert,

I'm basing this upon the lack of ICMP port unreachables being sent back to the source IP address (they're are not in the trace). The Snort IDS at this site may not be configured to watch for packets that match the signatures in the ICMP-INFO.rules file. If a simple TCP port scanner was used, it should not have retransmitted the SYN packet after the first ICMP port unreachable was received. The scans that fall outside the TCP retransmission times were not reinitiated as a function of the IP stack (i.e. it was the user or his script).

Just to clarify, all these packets had the SYN flag set. I omitted the TCP flags line for brevity, but didn't specifically state that the TCP flags were the same, point well taken. But I did state that the alerts produced were similar. I do not consider a Snort alert caused by a SYN and a SYN-ACK scan similar.

Thomas

Executive Summary

After extensive analysis of the IDS log files provided, it appears that the network is not protected with a firewall or packet filtering router. There is also strong evidence of the use of peer-to-peer file sharing activity involving the trading of copyrighted software. If the current acceptable use policy (AUP) does not address such activities, a section concerning duplication and sharing of copyrighted material should be created. There also seems to be a large number of Microsoft based web servers on the network. Some of these appear to have been compromised by external users or worms. Several Windows based computers were accessed by what appears to be an "administrator" level file share. Several FTP servers were subjected to a possible Denial of Service (DoS) attack. It is highly recommended that more specific information about the network infrastructure and list of authorized email & web servers be made available for further analysis.

Files Used

Five consecutive days of Snort IDS logs were analyzed for the time period of 4 November 2002 thru 8 November 2002.

Alert	Scans	Out of Spec
alert.021104.gz	scans.021104.gz	OOS_Report_2002_11_04_10436.txt
alert.021105.gz	scans.021105.gz	OOS_Report_2002_11_05_23122.txt
alert.021106.gz	scans.021106.gz	OOS_Report_2002_11_06_5934.txt
alert.021107.gz	scans.021107.gz	OOS_Report_2002_11_07_21575.txt
alert.021108.gz	scans.021108.gz	OOS_Report_2002_11_08_4950.txt

The alert files contained 703,836 alerts, with the first alert at 00:00:09 on 4 November 2002 and the last alert occurring at 00:06:38 on 9 November 2002.

There were some improperly formatted alert and scan entries that had appeared to be corrupted. Most of these appeared to be lines of alerts that had been combined. Some lines were missing ports and others were missing IP addresses. Any lines showing the 130.85.0.0/16 IP address space were assumed to be MY.NET.0.0/16, this was discovered after extensive analysis. All analysis scripts of the scans files were rerun in order to prevent any faulty analysis. The packet payload portion of the out-of-spec files also had some references to 130.85.0.0/16, however, these were only present in the payload.

Analysis Process and Tools Used

The amount of data was too much to analyze by hand. For basic file manipulation Windows ports of the popular Unix utilities of cat, grep, sed, wc, and md5 were used. Replacement of MY.NET with numeric values was required by the software to achieve accurate results. The files were grep'd for RFC1918 IP networks, these did occur in the logs files so these were not used in order to avoid tainting the log entries. In each of the files the "MY.NET" of each IP address was replaced with the numeric values of "192.170." These octets were selected because a previous GCIA analyst used them on his practical, they fell outside the RFC1918 ranges, and did not occur within the provide log files.

The alert and scans logs were analyzed with the popular tools Snort_sort.pl[26] and Snort_stat.pl.[27] I did attempt to use SnortSnarf, however, I was not able to fully process the entire 5 days of alert logs cat'd together with an AMD 2400+ and 1 GB RAM running Windows XP Professional. During the analysis period this machine was dedicated to running SnortSnarf. After 5 days of letting the process run I aborted it. This was after upgrading this machine from an AMD Duron 850 with 512MB RAM in an attempt to process the alert logs with SnortSnarf. Based upon performance analysis after the hardware upgrade, the bottleneck of the system became the ATA/100 7,200 RPM 80GB hard drive.

In his practical Kyle Haugsness describes that he attempted to use a laptop with SnortSnarf but quickly ran out of memory. To remedy this situation he ran SnortSnarf on a "larger Sun machine[28]" with 1 GB RAM, the hardware was not further described. Unfortunately, I do not have such hardware at my disposal.

I was able to use the SnortSnarf signature count table since it was completed before I aborted the process. However, the top twenty source and destination tables had not been completed yet. Based on a quick look at the files already created in the "sig" directory SnortSnarf had completed 600 of 16,000 alerts. In order to achieve the top ten talkers table I ran SnortSnarf on each individual day's alert logs and merged the data in Excel based upon the number of alerts generated by the host.

Most of the analysis was conducted with the use of grep, sed, cut, and wc. Once the data for specific signatures or hosts was separated from the cat'd alert files it was imported into Microsoft Excel and Access for analysis.

Two Windows XP based machines were used for analysis. The AMD system was used primarily for SnortSnarf. The other system was used for creation of this report and data manipulation in Microsoft Access and Excel. The other machine used is a Sony VAIO Pentium 4 2.0Ghz with 512MB RAM.

The link graph was generated using the udp/137 SMB wildcard scans and the visual analysis Advizor[29] by Visual Insights. The color graphic was exported as GIF file and pasted into this document. The data was taken from the report produced by SnortSort. The alert entries for SMB wildcard alerts were converted into an acceptable CSV format using Microsoft Excel.

Top 20 Scanners

[26] andrewb@snort.org. "Snort_sort.pl" 2000.03.17. URL: http://www.dpo.uab.edu/~andrewb/snort/snort_sort.html
[27] Chen, Yen-Ming. "Snort_stat.pl" 2001/08/24 URL: http://www.snort.org/dl/contrib/data_analysis/snort_stat.pl
[28] Haugsness, Kyle "SANS GIAC GCIA Practical" URL: http://www.giac.org/practical/Kyle_Haugsness_GCIA.zip
[29] Visual Insights, Inc. "Advizor" URL: http://www.visualinsights.com/products/data_visualization_solutions.asp

The table below shows the top twenty scanning source hosts. It is interesting to note that of these twenty hosts only five are from external source IP addresses. Although some of these may be false positives, the Snort scan pre-processor is reliable at detecting port scans. If these offending scans are SMTP, DNS, or web servers, the administrator of the Snort IDS should configure the respective server variables in the snort.conf file to prevent these false positives. These machines should be examined further to determine what is generating the port scans. If this traffic is legitimate, it can be ignored in future logs.

Rank	Scan Source IP	Count	External Scan
1	192.170.150.220	1390	
2	192.170.114.25	999	
3	209.116.70.75	460	✔
4	192.170.151.128	383	
5	192.170.111.213	293	
6	192.170.83.146	257	
7	192.170.104.155	251	
8	148.64.12.3	235	✔
9	61.252.141.22	204	✔
10	192.170.104.64	177	
11	192.170.70.176	175	
12	211.72.151.85	164	✔
13	140.112.42.18	155	✔
14	192.170.83.146	150	
15	192.170.137.18	144	
16	192.170.150.213	129	
17	192.170.168.159	121	
18	192.170.70.176	118	
19	192.170.71.164	113	
20	192.170.83.146	113	
Note: 192.170.0.0/16 IPs reflect MY.NET.0.0/16			

Top 5 External Scanners

These five external scanners are nominated for inclusion into the top ten talkers requirement of the analysis.

Scanning Source IP # 1 – 209.116.70.75

Registrant information (http://www.geektools.com/cgi-bin/proxy.cgi?query=net-209-116-70-64-1):

CustName: Red Hat, Inc.
Address: 4518 South Miami Blvd. Suite #100 Durham NC 27703
Country: US
RegDate: 2002-09-23
Updated: 2002-09-23

NetRange: 209.116.70.64 - 209.116.70.95
CIDR: 209.116.70.64/27
NetName: INFLOW-18773-5591
NetHandle: NET-209-116-70-64-1
Parent: NET-209-116-68-0-1
NetType: Reassigned
Comment:
RegDate: 2002-09-23
Updated: 2002-09-23

This source conducted scans on port tcp/25 against several different hosts in the network. Snort also triggered on the Queso OS fingerprinting signature. Each of the scans has the ECN bits turned on. This triggered the false positive in the Snort ruleset, resulting in entries in the alert, scans, and out-of-spec files. The host has a valid reverse DNS entry of *vger.kernel.org*. A visit using a web browser to this IP address states that "TCP/ECN is ON" leading me to believe that Red Hat frequently receives complaints about this false positive. The web page also states that this host provides email list services for Linux kernel developers. There is correlation of this detect in the SANS practicals of Robert Turner[30] and Hee So[31]. It is interesting to note that the hostname is the same, but the IP address is actually different in the Robert Turner practical, indicating that this host's services as now being provided by Red Hat Inc.

This host has a total of 121 records against 41 targets listed at DShield.org. One of these attacks was against port 62481 and nine attacks against port 25.[32]

Scanning Source IP # 2 – 148.64.12.3

Registrant Information (http://www.geektools.com/cgi-bin/proxy.cgi?query=148.64.12.3)

OrgName: Spacenet, Inc.
OrgID: SPAN

NetRange: 148.62.0.0 - 148.78.255.255
CIDR: 148.62.0.0/15, 148.64.0.0/13, 148.72.0.0/14, 148.76.0.0/15, 148.78.0.0/16
NetName: SPACENET-SPAN
NetHandle: NET-148-62-0-0-1
Parent: NET-148-0-0-0-0
NetType: Direct Allocation
NameServer: NS1-MCL.STARBAND.COM
NameServer: NS2-MCL.STARBAND.COM
NameServer: NS1-MAR.STARBAND.COM
NameServer: NS2-MAR.STARBAND.COM

[30] Turner, Robert. "SANS GIAC GCIA Practical" URL: http://www.giac.org/practical/Robert_Turner_GCIA.doc
[31] So, Hee. "SANS GIAC GCIA Practical" URL: http://www.giac.org/practical/Hee_So_GCIA.doc
[32] DShield Reports. URL: http://www.dshield.org/ipinfo.php?ip=209.116.70.75&Submit=Submit

Comment:
RegDate: 2000-05-31
Updated: 2001-07-26

Name: Miller, Fred
Handle: FM173-ARIN
Company:
Address: Spacenet Inc.
1750 Old Meadow Rd. McLean VA 22102
Country: US
Comment:
RegDate: 2000-09-21
Updated: 2002-07-29
Phone: +1-703-848-1108 (Office)
Phone: +1-703-848-1504 (Fax)
Email: fred.miller@spacenet.com

This IP address was conducting VECNA and SYN scans against 192.170.137.18. The VECNA scans have only the PUSH TCP flag set. In addition, several of the scans were SYN scans. There were no other hosts on the network targeted by this scanner. Since the scan activity of this host was out-of-spec, the payload of the packet was available for examination.

```
11/05-00:16:25.246393 148.64.12.3:3853 -> 192.170.137.18:6346
TCP TTL:110 TOS:0x0 ID:44327 IpLen:20 DgmLen:259 DF
****P*** Seq: 0x774F7E0A  Ack: 0x0  Win: 0x2000  TcpLen: 20
47 45 54 20 2F 67 65 74 2F 37 2F 41 64 6F 62 65  GET /get/7/Adobe
28 52 29 20 50 68 6F 74 6F 73 68 6F 70 28 52 29  (R) Photoshop(R)
20 37 2E 30 2E 73 69 74 20 48 54 54 50 2F 31 2E   7.0.sit HTTP/1.
31 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 4B  1..Connection: K
65 65 70 2D 41 6C 69 76 65 0D 0A 55 73 65 72 2D  eep-Alive..User-
41 67 65 6E 74 3A 20 42 65 61 72 53 68 61 72 65  Agent: BearShare
20 34 2E 30 2E 37 0D 0A 52 61 6E 67 65 3A 20 62   4.0.7..Range: b
79 74 65 73 3D 31 31 34 38 35 39 35 34 2D 31 37  ytes=11485954-17
32 32 38 39 33 30 0D 0A 58 2D 51 75 65 75 65 3A  228930..X-Queue:
20 30 2E 31 0D 0A 58 2D 47 6E 75 74 65 6C 6C 61   0.1..X-Gnutella
2D 43 6F 6E 74 65 6E 74 2D 55 52 4E 3A 20 75 72  -Content-URN: ur
6E 3A 73 68 61 31 3A 52 49 4D 4C 51 46 51 53 48  n:sha1:RIMLQFQSH
47 4D 50 37 50 52 51 45 33 48 4B 37 4B 32 58 49  GMP7PRQE3HK7K2XI
4B 35 32 4B 5A 4A 54 0D 0A 0D 0A                 K52KZJT....
```

It appears from the payload that this is the popular P2P file sharing utility BearShare. Also of interest is the GET request for a file. Based upon the extension it is a StuffIt compressed file. This compression software is mostly used on Apple Mac computers. This may be someone illegally distributing the Adobe Photoshop software using the university network as transmission medium.

I do not believe the transfer of the file was successful. The repeated packets with only PSH set are identical to what I have seen on my own DSL connection when I received the IP address of a previously P2P enabled machine. The global gnutella index still reflects

the availability of the Photoshop software at this IP host. Eventually these scans will stop once the gnutella network realizes that the host is no longer participating in the P2P gnutella network. Although the packet cited was generated by the gnutella client Morpheus, there is a correlation to this behavior on Neohapsis archives[33] of the Snort list server.

This host has a total of 11 records against 2 targets listed at DShield.org.[34]

Scanning Source IP # 3 – 61.252.141.22

Registrant Information (http://whois.nic.or.kr/)

IP Address : 61.252.141.0-61.252.141.63
Network Name : HANINTERNET-LLINE-IOK
Connect ISP Name : HANINTERNET
Connect Date : 20021009
Registration Date : 20021010

[Organization Information]
Orgnization ID : ORG257722
Org Name : IOK
State : SEOUL
Address : HANNAM YONGSAN
Zip Code : 140-210

[Admin Contact Information]
Name : SEOJIN OH
Org Name : IOK
State : SEOUL
Address : HANNAM YONGSAN
Zip Code : 140-210
Phone : +82-2-797-9347
E-Mail : ip@haninternet.co.kr

[Technical Contact Information]
Name : SEOJIN OH
Org Name : IOK
State : SEOUL
Address : HANNAM YONGSAN
Zip Code : 140-210
Phone : +82-2-797-9347
E-Mail : ip@haninternet.co.kr

[33] francisv@dagupan.com. "Morpheus traffic classified as Vecna scan" 4 September 2002. URL: http://archives.neohapsis.com/archives/snort/2002-09/0104.html
[34] DShield.org Reports. URL: http://www.dshield.org/ipinfo.php?ip=148.64.12.3&Submit=Submit

This attacker performed scans against 2,555 unique hosts on the university network. The attacker targeted tcp/445 which is used for Microsoft Active Directory. This attacker was seeking Microsoft Windows 2000 or XP machines. The scan was a standard SYN type scan. This appears to be a standard SYN against all the targets; however, host 192.170.132.42 elicited a scan against tcp/139 as indicated in the log excerpt below:

```
Nov  5 06:31:48 61.252.141.22:3716 -> 192.170.132.42:445 SYN ******S*
Nov  5 06:31:48 61.252.141.22:3717 -> 192.170.132.42:139 SYN ******S*
```

The scan log doesn't have time granularity beyond one second. However, based upon the source port of the scan it can be deduced that tcp/139 was scanned after the scanner determined that tcp/445 was open and therefore is a Windows based computer.

There were no reports of any activity by this host at DShield.org.[35]

A check of other targeted activity against host 192.170.132.42 shows that Snort triggered on the SMB C access signature for the following hosts:

200.70.46.237	211.38.70.68	220.82.182.106
24.92.26.222	212.237.5.51	61.156.206.34
203.223.35.130	61.216.18.203	203.73.239.235
217.125.133.72	62.150.25.91	

Based upon the number of external accesses against this Windows based host, it is highly recommended that this host be examined for being compromised. This host has either open shares or a weak password on the default accounts of guest or administrator. It is interesting to note that the scanning host did not trigger any other Snort rules during the five day period.

Scanning Source IP # 4 – 211.72.151.85

Registrant Information (http://whois.twnic.net)

IP 代理發放單位網段：211.72.128.0 - 211.72.255.255

Netname HINET-NET
Organization Name CHTD, Chunghwa Telecom Co.,Ltd.
Address 1 Data-Bldg.6F, No.21, Sec.21, Hsin-Yi Rd.
Address 2 Taipei Taiwan 100
Admin-c HN184-TW
Tech-c HN185-TW

用戶網段：211.72.151.80/29

Organization Name Ford Lio Ho Motor Comdany Ltd.
Street Address No.705, Sec.1, Chung Huo Rd., Taoyuan
City Taoyuan

[35] DShield.org Reports. URL: http://www.dshield.org/ipinfo.php?ip=61.252.141.22&Submit=Submit

State Taiwan
Country Code TW
IP Network 211.72.151.80/29
Network Name FORD-LIO-HO--TY-NET
Admin Contact E-Mailbox 1 jhsus@fordtw.com.tw
Tech Contact E-Mailbox 1 jhsus@fordtw.com.tw
Host Number 3/4/5
Subnet Number 1/1/1
Registered Date 2000/02/24

This attacker used the same scan technique as the previous host targeting tcp/445. He targeted 748 unique hosts on the university network. Only one host was scanned with an additional port. Snort triggered on the SMB Name Wildcard rule, this rule is triggered when an attacker elicits a list of Netbios services a host offers.

11/05-01:01:31.537334 SMB Name Wildcard 211.72.151.85:137 -> 192.170.137.7:137

This targeted host was also the subject of other attacks as shown below. Based upon this machine being a target of tcp/445 scan and then an udp/137, I do not believe it is a Unix based machine running Samba. The host was the subject of a Sun RPC scan, however, that scan covered a large number of hosts and appears to be a general reconnaissance scan.

11/04-09:03:41 NMAP TCP ping! 63.243.68.2:80 -> 192.170.137.7:53
11/05-14:52:53 EXPLOIT x86 setgid 0 207.46.227.174:80 -> 192.170.137.7:29104
11/06-17:24:38 NMAP TCP ping! 65.36.72.4:80 -> 192.170.137.7:53
11/06-23:06:53 Port 55850 udp - Possible myserver activity - ref. 010313-1 204.183.84.240:55850 -> 192.170.137.7:53
11/06-23:06:57 Port 55850 udp - Possible myserver activity - ref. 010313-1 192.170.137.7:53 -> 204.183.84.240:55850
11/07-20:53:07 IDS552/web-iis_IIS ISAPI Overflow ida nosize 80.206.168.161:3598 -> 192.170.137.7:80
11/08-03:27:06 NMAP TCP ping! 159.237.4.2:80 -> 192.170.137.7:53
11/08-06:25:01 NMAP TCP ping! 66.208.15.65:80 -> 192.170.137.7:137
11/08-06:25:06 NMAP TCP ping! 66.208.15.65:80 -> 192.170.137.7:137
11/08-06:25:11 NMAP TCP ping! 207.19.80.40:80 -> 192.170.137.7:137
11/08-06:25:16 NMAP TCP ping! 207.19.80.40:80 -> 192.170.137.7:137
11/08-06:50:29 Port 55850 udp - Possible myserver activity - ref. 010313-1 204.183.84.240:55850 -> 192.170.137.7:53
11/08-06:50:29 Port 55850 udp - Possible myserver activity - ref. 010313-1 192.170.137.7:53 -> 204.183.84.240:55850
11/08-21:13:32 IDS552/web-iis_IIS ISAPI Overflow ida nosize 67.119.108.239:2261 -> 192.170.137.7:80
11/08-23:14:05 External RPC call 211.254.220.230:2739 -> 192.170.137.7:111
Note: This is a snipped Snort log – removed time beyond 1 second granularity and [**] spacers.

The NMAP pings are attempts to determine if the host is listening on that port. It appears that this host may also serve as a DNS server based upon the signatures triggered indicating that the port udp/53 was used. The logs show no evidence of an attempted DNS BIND exploit.

The last alert targets Sun RPC on tcp/111. Windows hosts do not use that port unless specific software has been installed to provide services such as NFS to Unix machines, therefore this appears to be part of a generalized scan looking for machines offering Sun RPC services. The host 211.254.220.230 conducted a scan triggering the Snort alert for rule “External RPC call” a total of 176 times.

This host has a total of 7 records against 7 targets listed at DShield.org.[36]

Two hosts launched the "web-iis_IIS ISAPI Overflow ida nosize" attack against this host. This attack attempts to exploit a buffer overflow in the Microsoft Index Service, which indexes the content of a website. This attack is described as:

> This event indicates that a remote attacker has attempted to exploit a vulnerability in Microsoft IIS. An unchecked buffer in the Microsoft IIS Index Server ISAPI Extension could enable a remote intruder to gain SYSTEM access to the web server.[37]

It is recommended that this host be examined to see if it is indeed running Microsoft IIS. If this host is not patched in accordance with Microsoft Security Bulletin MS01-033[38], the server has been compromised and should be rebuilt. If the host is an authorized web server the use of the IIS Lockdown[39] tool and the latest service packs/hotfixes should be applied after being rebuilt. Ideally these service packs & hotfixes should be applied before connecting the system to the network.

Scanning Source IP # 5 – 140.112.42.18

Registrant Information:

OrgName: Ministry of Education Computer Center
OrgID: MOEC

NetRange: 140.112.0.0 - 140.112.255.255
CIDR: 140.112.0.0/16
NetName: TANET1
NetHandle: NET-140-112-0-0-1
Parent: NET-140-0-0-0-0
NetType: Direct Assignment
NameServer: DNS.NTU.EDU.TW
NameServer: NTU3.NTU.EDU.TW
NameServer: NEWS.NTU.EDU.TW
Comment:
RegDate: 1990-03-24
Updated: 1999-06-01

TechHandle: AT122-ARIN
TechName: Administrator TANet, Administrator
TechPhone: 886-2-27377010-305
TechEmail: tanetadm@moe.edu.tw

[36] DShield.org Reports. URL: http://www.dshield.org/ipinfo.php?ip=211.72.151.85&Submit=Submit
[37] arachNIDS. URL: http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids552&view=event
[38] Microsoft Corp. "Microsoft Security Bulletin MS01-033" 18 June 2001. URL: http://www.microsoft.com/technet/security/bulletin/MS01-033.asp
[39] Microsoft Corp. "IIS Lockdown Tool" URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/tools/locktool.asp

This attacker performed a simple TCP SYN scan targeting port 80 listening hosts on the university network. A total of 2,877 unique hosts were scanned during the time period examined.

Day	Detected Scan Count
4 November	0
5 November	0
6 November	0
7 November	348
8 November	5,709

This host did not produce any other alerts or appear in the out-of-spec files. As long as the host does not continue to scan the university networks, this scanner can be ignored. If the scans persist it would be advisable to contact the technical point of contact listed for this host. If there is no response, it is recommended that the upstream ISP of Ministry of Education be contacted. Their upstream ISP is Global Crossing, which lists the email address of security@gblx.net for complaints concerning cracking/hacking activities.

This host has a total of 360 records against 254 targets listed at DShield.org.[40]

The remainder of this /16 address space did conduct some other activity against the university network. The target host 192.170.21.43 was subjected to 154 Unicode attacks and one IIS ISAPI buffer overflow. The remainder of the activity was triggered by the attacker 140.112.110.89 conducting scans to determine the Netbios services that various hosts provide.

```
11/04-17:54:26 spp_http_decode: IIS Unicode attack detected 140.112.214.61:4908 -> 192.170.21.43:80
11/05-16:06:37 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.24:137
11/05-16:06:38 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.29:137
11/05-16:06:38 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.32:137
11/05-16:06:39 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.34:137
11/05-16:06:39 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.35:137
11/05-16:06:40 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.44:137
11/05-16:06:41 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.52:137
11/05-16:06:42 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.55:137
11/05-16:06:43 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.64:137
11/05-16:06:45 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.74:137
11/05-16:06:48 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.95:137
11/05-16:06:49 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.101:137
11/05-16:06:49 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.102:137
11/05-16:06:49 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.104:137
11/05-16:06:49 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.106:137
11/05-16:06:51 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.115:137
11/05-16:06:51 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.117:137
11/05-16:06:51 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.119:137
11/05-16:06:52 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.125:137
11/05-16:06:52 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.126:137
11/05-16:06:53 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.132:137
```

[40] DShield.org Reports. URL: http://www.dshield.org/ipinfo.php?ip=140.112.42.18&Submit=Submit

11/05-16:06:54 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.135:137
11/05-16:06:54 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.138:137
11/05-16:06:55 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.145:137
11/05-16:06:56 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.153:137
11/05-16:06:57 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.157:137
11/05-16:06:57 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.159:137
11/05-16:06:58 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.162:137
11/05-16:06:58 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.165:137
11/05-16:06:59 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.170:137
11/05-16:06:59 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.173:137
11/05-16:07:00 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.179:137
11/05-16:07:01 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.181:137
11/05-16:07:02 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.192:137
11/05-16:07:03 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.195:137
11/05-16:07:03 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.199:137
11/05-16:07:04 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.205:137
11/05-16:07:06 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.215:137
11/05-16:07:06 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.219:137
11/05-16:07:07 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.220:137
11/05-16:07:08 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.232:137
11/05-16:07:09 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.237:137
11/05-16:07:10 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.244:137
11/05-16:07:11 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.248:137
11/05-16:07:11 SMB Name Wildcard 140.112.110.89:1028 -> 192.170.133.251:137
Note: This is a snipped Snort log – removed time beyond 1 second granularity and [**] spacers.

There is a tremendous amount of reported activity of this address space at http://isc.incidents.org/source_report.html?order=&subnet=140.112.

Top 20 Alerting Hosts

This table represents the top twenty hosts producing the most alerts on the Snort IDS. It is interesting to note that only four of these hosts are from outside the university's network. Although some of these alerts may be false positives, it gives the overall impression that there are serious security related issues that need to be addressed on the university's network.

Rank	Alert Source IP	Count	External Alert
1	192.170.106.109	7,109	
2	192.170.162.91	4,957	
3	192.170.111.232	4,047	
4	192.170.111.219	4,043	
5	192.170.111.235	4,039	
6	192.170.111.230	4,012	
7	192.170.53.55	3,537	
8	192.170.106.105	3,249	
9	192.170.106.108	3,193	
10	140.142.19.69	2,997	✔
11	192.170.106.109	2,917	
12	192.170.183.59	2,910	
13	192.170.111.231	2,502	
14	212.179.35.8	2,454	✔
15	212.179.35.39	2,112	✔

16	192.170.153.175	1,951	
17	68.97.54.51	1,783	✔
18	192.170.88.242	1,781	
19	192.170.91.103	1,764	
20	192.170.153.179	1,603	

The top five of these hosts account for just over two-thirds of the alerts with the top ranked host accounting for 11% of the overall total. Therefore, these five top alert generating hosts are the remaining five hosts nominated for the top ten talkers requirement of the analysis. A table showing the top ten talkers is at the end of this report

Alerting Host #1 – 192.170.106.109

Twice this host performed the CGI Null Attack against one host (66.28.47.166). I believe this to be a false positive because a traceroute to the host shows the last hop host before the target as Speedera.demarc.cogentco.com. Speedera is a web content delivery provider.

```
Tracing route to 66.28.47.166 over a maximum of 30 hops

  1    <1 ms    <1 ms    <1 ms  router [10.0.0.1]
  2    31 ms    33 ms    30 ms  217.5.98.151
  3    30 ms    58 ms    39 ms  217.237.156.170
  4   117 ms   119 ms   119 ms  NYC-gw19.USA.net.DTAG.DE [62.156.131.134]
  5   632 ms   632 ms   631 ms  p1-0.pr1.nyc4.netrail.net [205.215.1.9]
  6   645 ms   644 ms   634 ms  p14-0.core02.jfk02.atlas.cogentco.com [205.215.61.173]
  7   621 ms   619 ms   620 ms  p15-0.core01.jfk02.atlas.cogentco.com [66.28.4.13]
  8   628 ms   655 ms   640 ms  p4-0.core02.dca01.atlas.cogentco.com [66.28.4.81]
  9   615 ms   619 ms   613 ms  g50.ba01.b003364-1.dca01.atlas.cogentco.com
[66.250.9.254]
 10   611 ms   611 ms   613 ms  Speedera.demarc.cogentco.com [66.250.7.166]
 11   612 ms   623 ms   619 ms  66.28.47.166
```

There have been some problems with Speedra services triggering some ICMP Snort rules. More information concerning these ICMP Snort rules can be found at http://www.whitehats.com/info/IDS152. However, these rules were not triggered.

The host also triggered 7,107 instances of the unicode attack signature against three unique hosts 210.219.201.2, 218.50.2.40, and 218.50.2.41.

The first targeted host does not appear to be online anymore. A traceroute to this host results in a routing loop between 203.239.183.77 and 203.239.183.78. A Google.com and DShield.org search for the IP address resulted in no hits.

```
Tracing route to 210.219.201.2 over a maximum of 30 hops

  1    <1 ms    <1 ms    <1 ms  router [10.0.0.1]
  2    36 ms    31 ms    31 ms  217.5.98.151
  3    31 ms    29 ms    29 ms  217.237.156.170
  4    46 ms    51 ms    45 ms  LON-SA1.LON.GB.NET.DTAG.DE [62.154.5.30]
  5    53 ms    52 ms    53 ms  64.215.195.173
  6    53 ms    55 ms    52 ms  pos0-0-622M.br1.LON3.gblx.net [195.8.96.186]
  7    53 ms    64 ms    62 ms  pos8-0-2488M.cr1.LON3.gblx.net [64.212.107.145]
  8   344 ms   345 ms   342 ms  pos1-1-155M.cr2.SEL2.gblx.net [203.192.165.118]
  9   356 ms   342 ms   343 ms  so1-0-0-622M.ar1.SEL2.gblx.net [203.192.165.86]
 10   359 ms   347 ms   345 ms  ELIMNET.t3-0-2-3.ar1.SEL2.gblx.net [203.192.164.246]
 11   352 ms   344 ms   360 ms  203.239.183.73
 12   345 ms   346 ms   352 ms  203.239.183.78
```

```
13   345 ms   377 ms   375 ms  203.239.183.77
14   348 ms   354 ms   352 ms  203.239.183.78
15   350 ms   348 ms   350 ms  203.239.183.77
16   366 ms   355 ms   352 ms  203.239.183.78
17   348 ms   350 ms   381 ms  203.239.183.77
18   374 ms   370 ms     *     203.239.183.78
19   350 ms   352 ms   364 ms  203.239.183.77
20   355 ms   356 ms   351 ms  203.239.183.78
21   350 ms   353 ms   349 ms  203.239.183.77
22   358 ms   358 ms   354 ms  203.239.183.78
23   352 ms   354 ms   351 ms  203.239.183.77
24   353 ms   361 ms   360 ms  203.239.183.78
25   353 ms   355 ms   374 ms  203.239.183.77
26   372 ms   365 ms   360 ms  203.239.183.78
27   355 ms   362 ms   355 ms  203.239.183.77
28   362 ms   357 ms   357 ms  203.239.183.78
29   360 ms   365 ms   361 ms  203.239.183.77
30   364 ms   359 ms   373 ms  203.239.183.78
```

Both 218.50.2.40 and 218.50.2.41 are listening on port 80. However, there is no HTML that is rendered. Both hosts are Windows 2000 machines running IIS 5.0 based upon the HEAD request that was issued against these hosts. These hosts were accessed using the Unix text based web browser Lynx.

No correlation on these hosts could be found at DShield.org or Google.com. It appears that these hosts may at one time have been infected with a worm such as Code Red or Nimda.

Alerting Host #2 – 192.170.162.91

This host launched the *web-iis_IIS ISAPI Overflow ida nosiz*e attack against 4,959 unique hosts. Snort detected 4,423 instances of these attacks as SYN port scans. This attack attempts to exploit a buffer overflow in the Microsoft Index Service, which indexes the content of a website. This attack is described as:

> This event indicates that a remote attacker has attempted to exploit a vulnerability in Microsoft IIS. An unchecked buffer in the Microsoft IIS Index Server ISAPI Extension could enable a remote intruder to gain SYSTEM access to the web server.[41]

Based on the available log information this host was compromised by 160.253.153.66 (no reverse DNS). The attacking host used the IIS ISAPI buffer overflow against the university host. Within less than 1 second, host 192.170.162.91 started triggering the same attack signature on the Snort IDS. The triggered signature is similar to the stock Snort signature, except that it requires an internal university IP address must be the source of the attack.

```
11/08-10:38:29.851976 IDS552/web-iis_IIS ISAPI Overflow ida nosize 160.253.153.66:36752 -> 192.170.162.91:80
11/08-10:38:30.744134 IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize 192.170.162.91:3458 -> 165.175.104.149:80
```

Note: This is a snipped Snort log – removed the [**] spacers.

This attacking host has a total of 1951 records against 737 targets listed at DShield.org[42]

[41] arachNIDS. URL: http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids552&view=event
[42] DShield.org Reports. URL: http://www.dshield.org/ipinfo.php?ip=160.253.153.66&Submit=Submit

Registrant Information: (http://www.geektools.com/cgi-bin/proxy.cgi?query=160.253.153.66)

```
OrgName: Montgomery College OrgID: MONTGO-11

NetRange: 160.253.0.0 - 160.253.255.255
CIDR: 160.253.0.0/16
NetName: MC
NetHandle: NET-160-253-0-0-1
Parent: NET-160-0-0-0-0
NetType: Direct Assignment
NameServer: SPARKY.MC.CC.MD.US
NameServer: AUTH02.NS.UU.NET
Comment:
RegDate: 1992-06-24
Updated: 1999-04-23

TechHandle: RM5456-ARIN
TechName: Morrow, Rick
TechPhone: +1-301-650-1322
TechEmail: rmorrow@mc.cc.md.us
```

It is highly recommended that this host be taken offline and examined further for being compromised.

Alerting Host #3 – 192.170.111.232

This host executed 4,047 attempts that triggered the *TFTP - External UDP connection to internal tftp server* rule. All these attempts were against the host 192.168.0.253. This IP falls within the RFC1918 range. These IP addresses cannot be routed, so these packets never reached the intended destination. I searched all the files for more indications of what this attack was trying to accomplish. I did find reference to a Red Worm attack also directed against host 192.168.0.253.

I have a theory about how this alert was caused; however, I do not have any correlation of my theory with any other SANS practicals. I believe the host might have been compromised by the Code Red worm or a variant before the time period that was analyzed. The attacking host is located behind a NAT device. This NAT device changes the source IP address in the IP header of the packets sent to 192.170.111.232, however, the IP address reflected within the actual http payload is that of web server itself. This could be any RFC1918 IP address. Therefore when the university host attempts to TFTP the worm code it uses the RFC1918 IP address instead of the routable NAT'd IP address. The host was scanned 28 times with a SYN scan against port 80, which makes the theory more creditable.

Regardless of the theories correctness, this host needs to be examined for being compromised and to determine why these TFTP requests were being made.

Alerting Host #4 – 192.170.111.219

This host exhibits the same alerts and behavior as 192.170.111.232. Snort triggered 4,044 attempts to perform a TFTP transfer against host 192.168.0.253. It was the victim

of 27 port 80 SYN scans. This host should be examined for being compromised and to determine what caused the TFTP requests to be made.

Alerting Host #5 – 192.170.111.235

This host exhibits the same alerts and behavior as 192.170.111.232 and 192.170.111.219. Snort triggered 4,040 attempts to perform a TFTP transfer against host 192.168.0.253. It was the victim of 27 port 80 SYN scans. This host should be examined for compromise and to determine the cause of the TFTP requests.

These three hosts are located on the same subnet and have been subjected to the same level of attacks and have exhibited almost identical alert counts. These three hosts have probably been compromised using the same technique and tactics.

Analysis by Alert Type

The previous section focused on analysis by source and destination IP address. This section seeks to give analysis based up on the most significant alerts. While all these alerts may not all have a high occurrence count, they are still significant in the security of the university's network. Where significant, the host IP addresses of internal university hosts are identified for examination by respective system administrators. A list ranking all alerts based upon number of occurrences is attached at the end of this document.

[TCP/IMCP] SRC and DST outside network & TFTP connection to outside server

> The alerts were generated by hosts that had source or destination IP addresses outside the 192.170.0.0/16 IP address range. Most of these systems are using IP addresses from the RFC1918 ranges. The alerts may be the result of misconfigured NAT devices on the network. In addition to the RFC1918 IP addresses, the IPAA range of 169.254.0.0/16 also frequently appeared. This address range is utilized when a DHCP enabled host is unable to reach a DHCP server. Eliminating the use of these addresses will lower the false positive rate and make actual alerts caused by these hosts easier to track down.

Back Orifice

> This alert triggered only once during the analyzed time period. This is most likely a false positive. However, if the Back Orifice preprocessor is enabled on the Snort IDS the host 192.170.85.114 should be examined.

SMB C access

> This alert is triggered when an external host accesses the SMB administrative share ($ADMIN) or administrative root share of a drive (C$) on a Windows

host.[43] In the referenced list serve message Max Vision describes that if the HOME network variable is properly defined in the snort.conf file, there should be very few false positives of this signature. It is recommended that the hosts in the table below be examined or a network scanner such as ISS Scanner or Nessus be used to assess these hosts.

192.170.105.103	192.170.152.157	192.170.177.21	192.170.84.185
192.170.105.94	192.170.152.160	192.170.177.47	192.170.84.193
192.170.106.103	192.170.152.161	192.170.177.50	192.170.84.196
192.170.106.105	192.170.152.162	192.170.177.60	192.170.84.202
192.170.107.191	192.170.152.163	192.170.178.101	192.170.84.219
192.170.107.76	192.170.152.166	192.170.178.140	192.170.84.238
192.170.108.241	192.170.152.170	192.170.178.159	192.170.84.253
192.170.108.42	192.170.152.171	192.170.178.169	192.170.87.177
192.170.108.43	192.170.152.178	192.170.178.172	192.170.88.191
192.170.108.45	192.170.152.185	192.170.178.184	192.170.88.209
192.170.109.25	192.170.152.246	192.170.178.19	192.170.88.210
192.170.109.59	192.170.152.250	192.170.178.218	192.170.88.215
192.170.109.64	192.170.152.48	192.170.178.221	192.170.88.231
192.170.110.202	192.170.153.108	192.170.178.53	192.170.88.238
192.170.110.205	192.170.153.110	192.170.178.55	192.170.88.241
192.170.110.211	192.170.153.116	192.170.179.11	192.170.88.246
192.170.110.226	192.170.153.117	192.170.179.68	192.170.90.202
192.170.110.229	192.170.153.126	192.170.179.79	192.170.90.210
192.170.110.230	192.170.153.146	192.170.18.28	192.170.90.217
192.170.110.52	192.170.153.148	192.170.180.185	192.170.90.223
192.170.111.128	192.170.153.150	192.170.180.57	192.170.90.233
192.170.111.145	192.170.153.151	192.170.82.241	192.170.90.234
192.170.111.178	192.170.153.157	192.170.83.162	192.170.90.237
192.170.111.188	192.170.153.159	192.170.83.175	192.170.90.250
192.170.111.213	192.170.153.161	192.170.83.176	192.170.90.253
192.170.111.218	192.170.153.166	192.170.83.177	192.170.90.254
192.170.111.233	192.170.153.168	192.170.83.196	192.170.91.183
192.170.112.171	192.170.153.174	192.170.83.213	192.170.91.243
192.170.112.183	192.170.153.175	192.170.83.232	192.170.91.245
192.170.112.208	192.170.153.185	192.170.83.235	192.170.91.246
192.170.112.211	192.170.153.195	192.170.83.243	192.170.91.251
192.170.112.23	192.170.153.203	192.170.84.151	192.170.91.253
192.170.112.28	192.170.153.204	192.170.84.152	192.170.99.177
192.170.112.30	192.170.153.211	192.170.84.155	192.170.99.179
192.170.113.203	192.170.163.235	192.170.84.158	
192.170.152.143	192.170.177.13	192.170.84.170	

[43] Butler, Max (Max Vision). "Re: [snort] 'SMB Name Wildcard'" 17 January 2000. URL: http://archives.neohapsis.com/archives/snort/2000-01/0220.html

EXPLOIT NTPDX buffer overflow

According to the event information located at Whitehats.com this signature is triggered when a NTP payload exceeds 128 bytes.[44] Under normal operations this length should not be exceeded. There are several versions of ntpd and xntpd that are exploitable to achieve a root level compromise of a host. If any of these hosts are running ntpd or xntpd they should be examined. The system administrators of the hosts below need to verify with the software vendor that these versions are not be susceptible to any buffer overflow attacks.

192.170.10.20	192.170.15.184	192.170.153.45	192.170.88.164
192.170.112.193	192.170.153.171	192.170.80.71	192.170.90.236

IIS Unicode attack detected

A total of 73 unique hosts on the university network were subjected to the Unicode attack. This family of attacks uses Unicode character representations to evade IDS detection and normal restrictions imposed by the web server. A popular form of the Unicode attack is the directory traversal. This method uses the Unicode characters to represent the backslash character used on Windows computers to indicate the root directory. Each of these attempts required an established TCP session against the host. Although these hostd may not be running Microsoft IIS, if they are not authorized web servers they should be shutdown.

192.170.100.133	192.170.110.113	192.170.113.220	192.170.151.128
192.170.100.143	192.170.110.114	192.170.113.224	192.170.153.178
192.170.100.145	192.170.110.165	192.170.114.116	192.170.153.219
192.170.100.158	192.170.111.126	192.170.116.101	192.170.162.109
192.170.100.187	192.170.111.140	192.170.116.110	192.170.162.119
192.170.100.217	192.170.111.145	192.170.122.127	192.170.162.123
192.170.100.221	192.170.111.155	192.170.130.122	192.170.162.203
192.170.100.223	192.170.111.159	192.170.130.182	192.170.162.241
192.170.104.104	192.170.111.206	192.170.130.187	192.170.163.131
192.170.104.113	192.170.111.222	192.170.130.190	192.170.163.132
192.170.104.128	192.170.112.164	192.170.130.212	192.170.163.134
192.170.104.139	192.170.113.202	192.170.138.202	192.170.163.138
192.170.104.145	192.170.113.206	192.170.140.188	192.170.167.200
192.170.104.155	192.170.113.207	192.170.145.139	192.170.168.236
192.170.104.177	192.170.113.208	192.170.145.148	192.170.178.254
192.170.104.180	192.170.113.211	192.170.150.139	192.170.198.214
192.170.104.211	192.170.113.212	192.170.150.228	
192.170.106.191	192.170.113.213	192.170.150.231	
192.170.106.222	192.170.113.214	192.170.151.114	

FTP DoS ftpd globbing

[44] arachNIDS. URL: http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids492&view=event

This is a DoS attack against ftp servers.[45] According to the arachnids database this attack against vulnerable servers is executed when a wildcard is used with the LIST command. This attack was directed at two hosts. If these are not authorized ftp servers, they should be removed from the network. A check with the vendor or author of the ftp daemon should be done in order to verify that the software is not vulnerable to this attack. If the software is vulnerable to this attack, the ftp software should be upgraded to the most current version to prevent future DoS attacks.

192.170.100.158	192.170.114.116

External RPC call

The table below reflects the external access attempts against the SunRPC port tcp/111. Although these may hosts may have been the subject of a port scan against tcp/111, the scans should be noted. If these are machines that are running Unix RPC services, they should be examined and removed from the network if they are not authorized. Ideally all access from the Internet to tcp/111 should be blocked. If tcp/111 is required to be open to the Internet, a watch list reflecting only the authorized servers should be created. These hosts should be closely monitored after this network reconnaissance.

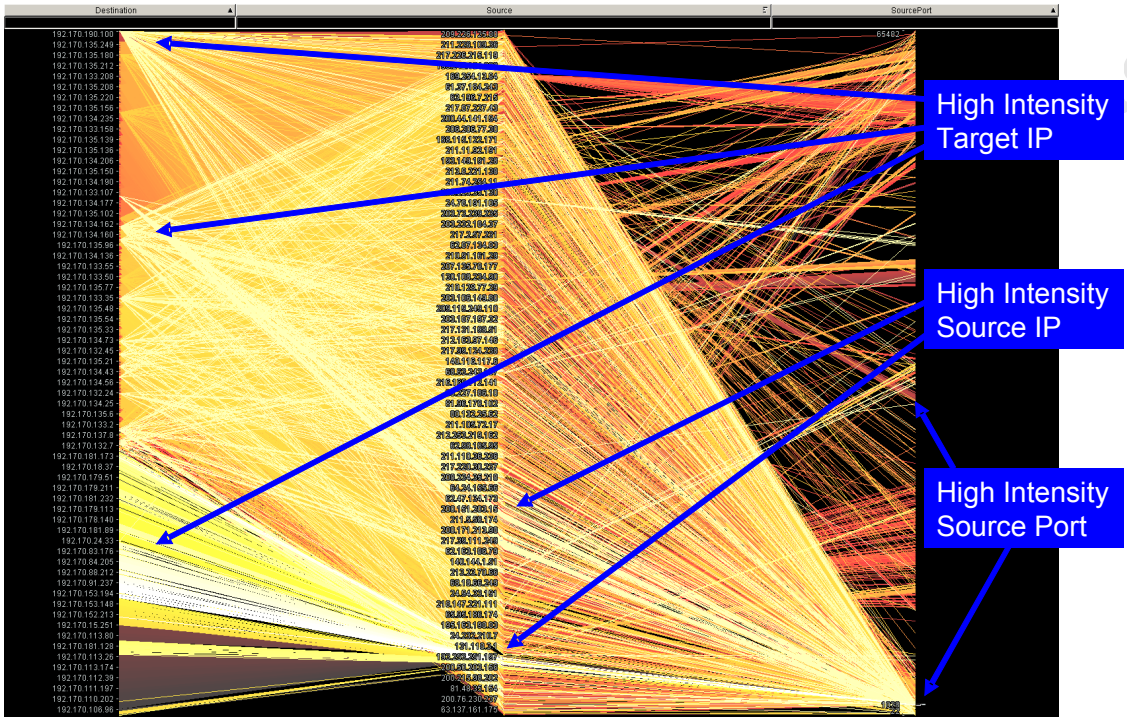
192.170.133.10	192.170.133.218	192.170.133.58	192.170.134.51
192.170.133.100	192.170.133.22	192.170.133.6	192.170.134.52
192.170.133.103	192.170.133.220	192.170.133.60	192.170.134.54
192.170.133.105	192.170.133.222	192.170.133.61	192.170.134.55
192.170.133.106	192.170.133.223	192.170.133.62	192.170.134.57
192.170.133.107	192.170.133.224	192.170.133.63	192.170.134.58
192.170.133.108	192.170.133.225	192.170.133.64	192.170.134.6
192.170.133.109	192.170.133.226	192.170.133.65	192.170.134.63
192.170.133.11	192.170.133.227	192.170.133.66	192.170.134.69
192.170.133.110	192.170.133.228	192.170.133.67	192.170.134.70
192.170.133.111	192.170.133.23	192.170.133.7	192.170.134.79
192.170.133.113	192.170.133.238	192.170.133.70	192.170.134.93
192.170.133.115	192.170.133.239	192.170.133.71	192.170.134.95
192.170.133.119	192.170.133.24	192.170.133.74	192.170.134.98
192.170.133.12	192.170.133.240	192.170.133.76	192.170.135.36
192.170.133.126	192.170.133.241	192.170.133.77	192.170.135.51
192.170.133.128	192.170.133.243	192.170.133.79	192.170.135.6
192.170.133.129	192.170.133.244	192.170.133.8	192.170.135.68
192.170.133.13	192.170.133.245	192.170.133.80	192.170.135.81
192.170.133.131	192.170.133.246	192.170.133.81	192.170.135.84
192.170.133.132	192.170.133.251	192.170.133.82	192.170.135.85
192.170.133.133	192.170.133.252	192.170.133.9	192.170.135.90
192.170.133.134	192.170.133.253	192.170.133.90	192.170.137.1
192.170.133.137	192.170.133.254	192.170.133.93	192.170.137.10
192.170.133.144	192.170.133.26	192.170.134.1	192.170.137.11

[45] arachNIDS. URL: http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids487&view=event

192.170.133.145	192.170.133.27	192.170.134.100	192.170.137.12
192.170.133.146	192.170.133.28	192.170.134.104	192.170.137.13
192.170.133.147	192.170.133.29	192.170.134.14	192.170.137.14
192.170.133.149	192.170.133.3	192.170.134.15	192.170.137.15
192.170.133.15	192.170.133.30	192.170.134.16	192.170.137.16
192.170.133.155	192.170.133.31	192.170.134.17	192.170.137.17
192.170.133.16	192.170.133.32	192.170.134.19	192.170.137.18
192.170.133.17	192.170.133.33	192.170.134.190	192.170.137.19
192.170.133.175	192.170.133.34	192.170.134.20	192.170.137.20
192.170.133.176	192.170.133.35	192.170.134.21	192.170.137.21
192.170.133.18	192.170.133.36	192.170.134.219	192.170.137.23
192.170.133.180	192.170.133.37	192.170.134.22	192.170.137.37
192.170.133.181	192.170.133.4	192.170.134.220	192.170.137.4
192.170.133.182	192.170.133.40	192.170.134.23	192.170.137.44
192.170.133.183	192.170.133.41	192.170.134.24	192.170.137.5
192.170.133.184	192.170.133.42	192.170.134.244	192.170.137.6
192.170.133.185	192.170.133.43	192.170.134.25	192.170.137.65
192.170.133.19	192.170.133.44	192.170.134.26	192.170.137.66
192.170.133.190	192.170.133.45	192.170.134.29	192.170.137.67
192.170.133.193	192.170.133.46	192.170.134.3	192.170.137.69
192.170.133.197	192.170.133.47	192.170.134.30	192.170.137.7
192.170.133.198	192.170.133.49	192.170.134.32	192.170.137.72
192.170.133.20	192.170.133.5	192.170.134.33	192.170.137.73
192.170.133.207	192.170.133.50	192.170.134.39	192.170.137.74
192.170.133.208	192.170.133.52	192.170.134.40	192.170.137.8
192.170.133.209	192.170.133.53	192.170.134.41	192.170.190.100
192.170.133.21	192.170.133.54	192.170.134.43	192.170.190.36
192.170.133.211	192.170.133.55	192.170.134.46	
192.170.133.213	192.170.133.56	192.170.134.48	
192.170.133.215	192.170.133.57	192.170.134.49	

Link Graph

The link graph below shows the destination (target) and source IP address with source port over the analyzed time period. The coloring of the graph was achieved by using the thermal color setting in Advizor. The more intense the heat is the whiter the color. Red represents the coolest setting. The heat is represented by the intensity of a scan. Most of the scans originated on fairly low ephemeral ports as evidenced by the white coloring near the bottom right hand of the graph. Several source IP addresses are fairly white and indicate intense scanning activity showing that these IP address repeatedly performed scans. The left column represents the destination IP addresses. This data is limited to only those targets that were scanned on port udp/137. A more accurate spread of the scan across the whole university network could be achieved by using all the IP address space in use. Using the entire /16 address is possible; however, use of only the specific address space that is actually in use would create the most accurate representation of this scan. The graph can be recreated if this data is made available.



Defensive Recommendations

In addition to the defensive recommendations concerning specific hosts in the above portion of this document, the following measures should be taken to increase the overall security of the network.

Based upon the attacks and the lack of information about the network infrastructure deployed, it appears that there is no firewall or packet filtering device protecting the university network. If at all possible, such a device should be deployed. This device should filter RFC1918 IP ranges bi-directionally and perform ingress & egress filtering as described in RFC2827. Services such as LPR, SunRPC, CIFS, HTTP/HTTPS, and those ports/protocols only required locally on the campus network should be restricted from external source IP addresses. If such access is required by administration, faculty, or students from outside the campus LAN the use of a VPN (Virtual Private Network) should be considered. It should be noted that the perimeter of the university network then extends to all VPN clients. This will need to be taken into account when considering all aspects of the network security policy.

In addition to the official university IP address space there is extensive use of the RFC1918 and IPAA (169.254.0.0/16) address space on the network. This may be due to the proliferation of NAT (Network Address Translation) devices on the network which may be misconfigured and therefore leaking packets with incorrect source IP addresses. It

is highly recommended that these be sought out and removed if they are prohibited by the network security policy. Locations of such hosts can be determined by obtaining copies of the network routers and switches' ARP tables with the corresponding physical interface. This will allow the localization of the host to the Ethernet RJ-45 connection. Depending on what further equipment is connected to the port, localization of the host may still require extensive physical searching. If a wireless access point is attached to the port, it may be impossible to find the offending host.

There seems to be extensive use of P2P file sharing software. In at least one out-of-spec packet payload there seems to be evidence of sharing copyrighted software. If this is against the acceptable use policy, the ports[46] commonly used for such software should be blocked using a firewall or packet filtering router. If the current acceptable use policy (AUP) does not address the redistribution of copyrighted materials, a new section needs to be added reflecting the official policy regarding such matters.

There is also a large number of Windows based machines that are running the web server software IIS. Many times this software is installed accidentally. There are numerous worms that target such servers. Any unneeded installations of IIS should be removed and those hosts carefully examined for being compromised. Such attacks can be mitigated by blocking inbound tcp/80 to all but the official authorized web servers. Although this will not prevent polymorphic worms, such as Nimda, from compromising these servers once it has enter through the perimeter of the network. It will prevent external hosts from compromising these unauthorized web servers.

[46] Australian CERT. "AA-2002.02 -- File-Sharing Activity Part 1 of 2 - Security implications of using peer-to-peer file sharing software" 20 May 2002. URL: http://www.auscert.org.au/render.html?it=2228

Top Ten Talkers

	IP Address	*Fully Qualified Domain Name*	*Country*	*Organization*	*Inclusion Reason*
1	209.116.70.75	vger.kernel.org	USA	Red Hat, Inc.	Within top five ranked external IP addresses in terms of scans
2	148.64.12.3	vsat-148-64-12-3.c050.t7.mrt.starband.net	USA	Spacenet/Speedera	Within top five ranked external IP addresses in terms of scans
3	61.252.141.22	(none)	South Korea	Han Internet (ISP)	Within top five ranked external IP addresses in terms of scans
4	211.72.151.85	(none)	China	Ford Lio Ho Motor Company	Within top five ranked external IP addresses in terms of scans
5	140.112.42.18	dcserver.ee.ntu.edu.tw	Taiwan (China)	Ministry of Education Computer Center	Within top five ranked external IP addresses in terms of scans
6	192.170.106.109	(unknown)	USA	University Internal IP Address	Within top five of the alert producing hosts
7	192.170.162.91	(unknown)	USA	University Internal IP Address	Within top five of the alert producing hosts
8	192.170.111.232	(unknown)	USA	University Internal IP Address	Within top five of the alert producing hosts
9	192.170.111.219	(unknown)	USA	University Internal IP Address	Within top five of the alert producing hosts
10	192.170.111.235	(unknown)	USA	University Internal IP Address	Within top five of the alert producing hosts

Note: Numerical ranking is not significant in this table.

SnortSnarf Alert Report 4 – 8 November 2002

Rank	Triggering Signature	Alert Count	IP Source Count	IP Destination Count
1	spp_http_decode: IIS Unicode attack detected	87,679	731	1,442
2	SMB Name Wildcard	65,323	1,533	1,479
3	TFTP - External UDP connection to internal tftp server	20,141	13	9
4	Watchlist 000220 IL-ISDNNET-990517	17,209	115	76
5	High port 65535 tcp - possible Red Worm – traffic	6,706	32	33
6	spp_http_decode: CGI Null Byte attack detected	6,031	85	96
7	IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize [arachNIDS]	4,957	1	4,949
8	Incomplete Packet Fragments Discarded	3,281	27	18
9	High port 65535 udp - possible Red Worm – traffic	1,932	137	131
10	Port 55850 tcp - Possible myserver activity - ref. 010313-1	1,666	41	41
11	Queso fingerprint	1,476	129	38
12	SUNRPC highport access!	1,438	39	45
13	Watchlist 000222 NET-NCFC	1,330	33	37
14	SMB C access	503	168	188
15	Tiny Fragments - Possible Hostile Activity	482	11	8
16	IDS552/web-iis_IIS ISAPI Overflow ida nosize [arachNIDS]	479	461	311
17	Null scan!	308	43	23
18	FTP DoS ftpd globbing	298	10	2
19	External RPC call	272	4	222
20	TCP SRC and DST outside network	194	11	63
21	EXPLOIT x86 NOOP	186	45	54
22	NMAP TCP ping!	172	45	38
23	EXPLOIT x86 setuid 0	160	72	34
24	TFTP - Internal TCP connection to external tftp server	155	3	3
25	TFTP - Internal UDP connection to external tftp server	136	16	16
26	Port 55850 udp - Possible myserver activity - ref. 010313-1	105	14	10
27	Possible trojan server activity	91	24	25
28	RFB - Possible WinVNC - 010708-1	45	20	21
29	EXPLOIT x86 setgid 0	40	28	21
30	IRC evil – running XDCC	24	2	3
31	EXPLOIT NTPDX buffer overflow	13	13	8

32	ICMP SRC and DST outside network	13	7	8
33	Attempted Sun RPC high port access	8	6	5
34	EXPLOIT x86 stealth noop	6	4	4
35	HelpDesk 192.170.70.50 to External FTP	6	1	2
36	SYN-FIN scan!	5	4	4
37	TFTP - External TCP connection to internal tftp server	5	2	2
38	External FTP to HelpDesk 192.170.70.50	3	3	1
39	External FTP to HelpDesk 192.170.70.49	3	2	1
40	connect to 515 from outside	3	2	1
41	FTP passwd attempt	2	2	2
42	Bugbear@MM virus in SMTP	2	2	1
43	NIMDA – Attempt to execute cmd from campus host	1	1	1
44	Probable NMAP fingerprint attempt	1	1	1
45	HelpDesk 192.170.70.49 to External FTP	1	1	1
46	SNMP public access	1	1	1
47	Back Orifice	1	1	1

Last Updated: May 21st, 2019

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS San Antonio 2019	San Antonio, TXUS	May 28, 2019 - Jun 02, 2019	Live Event
SANS Atlanta 2019	Atlanta, GAUS	May 28, 2019 - Jun 02, 2019	Live Event
Security Writing NYC: SEC402 Beta 2	New York, NYUS	Jun 01, 2019 - Jun 02, 2019	Live Event
Enterprise Defense Summit & Training 2019	Redondo Beach, CAUS	Jun 03, 2019 - Jun 10, 2019	Live Event
SANS Zurich June 2019	Zurich, CH	Jun 03, 2019 - Jun 08, 2019	Live Event
SANS London June 2019	London, GB	Jun 03, 2019 - Jun 08, 2019	Live Event
SANS Kansas City 2019	Kansas City, MOUS	Jun 10, 2019 - Jun 15, 2019	Live Event
SANS SEC440 Oslo June 2019	Oslo, NO	Jun 11, 2019 - Jun 12, 2019	Live Event
SANSFIRE 2019	Washington, DCUS	Jun 15, 2019 - Jun 22, 2019	Live Event
SANS Cyber Defence Canberra 2019	Canberra, AU	Jun 24, 2019 - Jul 13, 2019	Live Event
Security Operations Summit & Training 2019	New Orleans, LAUS	Jun 24, 2019 - Jul 01, 2019	Live Event
SANS ICS Europe 2019	Munich, DE	Jun 24, 2019 - Jun 29, 2019	Live Event
SANS Cyber Defence Japan 2019	Tokyo, JP	Jul 01, 2019 - Jul 13, 2019	Live Event
SANS Paris July 2019	Paris, FR	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS Munich July 2019	Munich, DE	Jul 01, 2019 - Jul 06, 2019	Live Event
SANS London July 2019	London, GB	Jul 08, 2019 - Jul 13, 2019	Live Event
SEC450 Security Ops-Analysis Beta 1	Crystal City, VAUS	Jul 08, 2019 - Jul 13, 2019	Live Event
SANS Cyber Defence Singapore 2019	Singapore, SG	Jul 08, 2019 - Jul 20, 2019	Live Event
SANS Charlotte 2019	Charlotte, NCUS	Jul 08, 2019 - Jul 13, 2019	Live Event
SANS Pittsburgh 2019	Pittsburgh, PAUS	Jul 08, 2019 - Jul 13, 2019	Live Event
SANS Rocky Mountain 2019	Denver, COUS	Jul 15, 2019 - Jul 20, 2019	Live Event
SANS Columbia 2019	Columbia, MDUS	Jul 15, 2019 - Jul 20, 2019	Live Event
SANS Pen Test Hackfest Europe 2019	Berlin, DE	Jul 22, 2019 - Jul 28, 2019	Live Event
SANS San Francisco Summer 2019	San Francisco, CAUS	Jul 22, 2019 - Jul 27, 2019	Live Event
DFIR Summit & Training 2019	Austin, TXUS	Jul 25, 2019 - Aug 01, 2019	Live Event
SANS Riyadh July 2019	Riyadh, SA	Jul 28, 2019 - Aug 01, 2019	Live Event
SANS July Malaysia 2019	Kuala Lumpur, MY	Jul 29, 2019 - Aug 03, 2019	Live Event
SANS Boston Summer 2019	Boston, MAUS	Jul 29, 2019 - Aug 03, 2019	Live Event
Security Awareness Summit & Training 2019	San Diego, CAUS	Aug 05, 2019 - Aug 14, 2019	Live Event
SANS Melbourne 2019	Melbourne, AU	Aug 05, 2019 - Aug 10, 2019	Live Event
SANS London August 2019	London, GB	Aug 05, 2019 - Aug 10, 2019	Live Event
SANS Crystal City 2019	Arlington, VAUS	Aug 05, 2019 - Aug 10, 2019	Live Event
SANS Krakow May 2019	OnlinePL	May 27, 2019 - Jun 01, 2019	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced