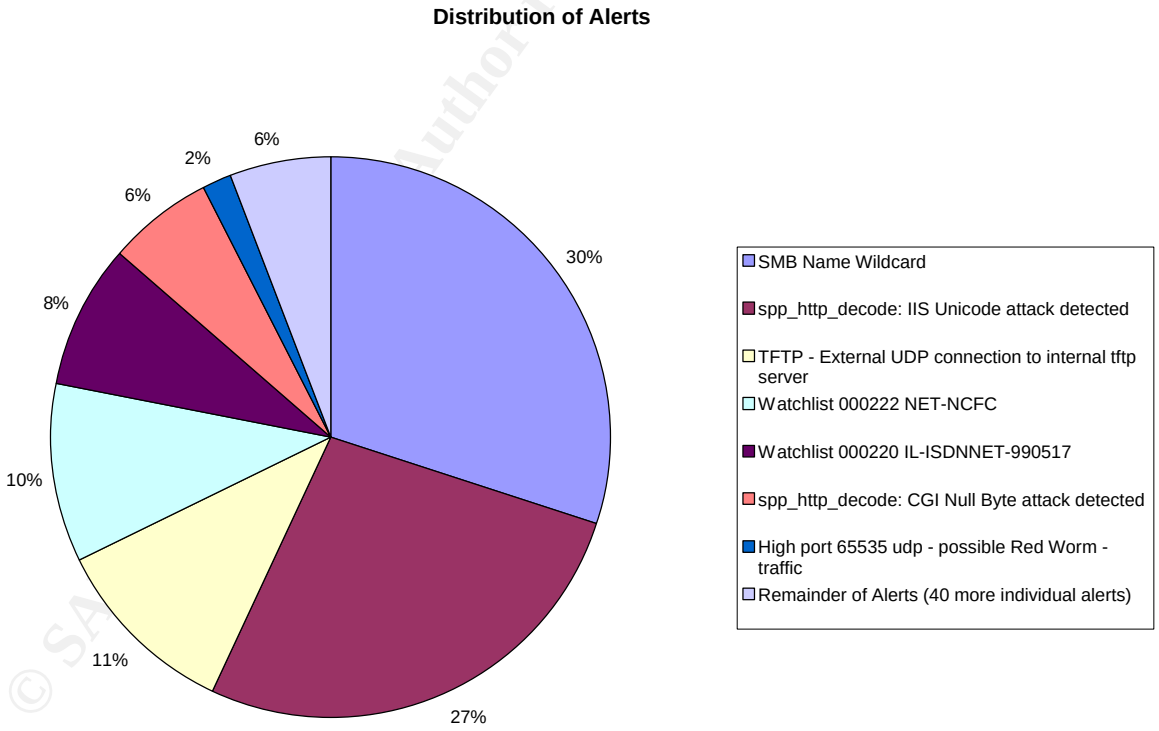
Global Information Assurance Certification Paper

GIAC Certified Intrusion Analyst (GCIA)
Practical Assignment
Version 3.3

Al Maslowski-Yerges
Rocky Mountain SANS 2002
Jan. 5 2003



PART 1 – THE STATE OF INTRUSION DETECTION

Analysis of the 3com embedded Firewall NIC's security and Intrusion Detection capabilities.

Abstract

In this paper I will cover a number of topics related to the embedded firewall technology introduced by 3Com and how it can/should fit into intrusion detection and information security in general.

One of the first topics I cover is an explanation of what an embedded firewall is in its current iteration.

My hope was that I would find that these network cards could serve as a distributed IDS system that could constantly watch the entire network if they were distributed strategically across the enterprise. In order to find out if this was the case I evaluated several things: the feature set, speed/performance, manageability, and logging/alerting.

After this evaluation I have come to the conclusion that in its current state the embedded Firewall NIC from 3Com only has limited usefulness for intrusion detection. But it does offer some very nice features that make it an attractive solution for protecting certain highly sensitive systems.

What is a 3com embedded firewall NIC?

In short it is a traffic inspection engine and encryption engine embedded in the hardware so that it can run at hardware speeds.

When it comes down to it all packet filtering firewalls must read in the every packet destined for it and make a decision, based on a rule set, of what action to take with the packet. This can be a very processor intensive task depending on the rule set and the amount of traffic. This NIC takes that load away from the CPU and processes the packet before it ever reaches the upper layer applications.

Another process that takes a lot of CPU cycles is encrypting and decrypting packets that are involved in an IPSEC tunnel. This hardware/software combination takes that load off of the CPU as well.

This solution also offers central management and logging capabilities so that all or a portion of the NIC's installed can be managed by group or individual policies without user intervention.

Tests and Analysis

Feature Set

The first thing I looked at was the FW feature set. One thing I noticed right away was that this is not a stateful firewall. There is not enough memory embedded on the card for it to keep state tables for inspection. Therefore, you are limited to strict packet filtering decisions as you would be with normal Cisco access lists many are familiar with.

Some nice features were included though. There were built in basic rule sets that would cover many situations and definitely give a good starting point. The logging was pretty complete as discussed further below. Also, the firewall is quite hard for anyone except the designated administrator to remove or alter which makes a good security solution. Software based firewalls can usually be disabled or altered by a user fairly easily.

Performance

Another thing that was of interest to me was the performance of the NIC. On software firewalls and routers, performance depends directly on how complex the rule set is or how many VPN connections you have. One of the benefits of this NIC was supposed to be that it offloaded the CPU intensive operations so that speed is not affected very much at all.

To test this I performed two basic sets of tests based on the transfer speed of a large file via FTP from one computer to another. I tested the transfer speeds with a restrictive firewall policy and then with a 3DES tunnel set up between the hosts. Throughout the tests I used the same file (W2ksp3.exe, a large patch file for Windows 2000), the same single switch and the same FTP server software to keep the number of other variables as low as possible.

First I established a baseline for file transfer speed with the standard, built-in network cards and standard OS's with the latest patches. I found that the transfers ranged from 33 seconds to a high of 266 seconds (normally up to only 39 seconds this was an anomaly, I think) See the following table

	Transfer Speed Standard Card NT4	Transfer Speed Standard Card 2000 Srv
From 2000 Server Standard Card	33.12Seconds 3954.91Kbytes/sec 36.78Seconds 3560.94Kbytes/sec 35.12Seconds 3729.46Kbytes/sec	Not Tested
From 2000 Pro Standard Card	37.03Seconds 3536.81Kbytes/sec 36.47Seconds 3591.21Kbytes/sec 37.36Seconds 3505.48Kbytes/sec	38.31Seconds 3419.36Kbytes/sec 42.90Seconds 3053.04Kbytes/sec 39.43Seconds 3322.06Kbytes/sec
From Win98 Standard Card	266.39Seconds 491.68Kbytes/sec 36.47Seconds 3591.21Kbytes/sec	36.78Seconds 3560.94Kbytes/sec 35.51Seconds 3688.40Kbytes/sec

	36.46Seconds 3592.10Kbytes/sec	37.03Seconds 3536.81Kbytes/sec

Next, I installed a Firewall NIC in the NT4 machine, configured the duplex and speed settings the same as the old NIC and installed the driver and software as described in the instructions. I ran a test set of file transfers with no FW policy at all installed and again with a very long and restrictive policy installed. The restrictive policy allowed only FTP and that was at the very bottom of a rule set with over 65 rules in it. The transfer speeds got a little slower but stayed quite high. For some reason it seemed to unequally affect the Windows 98 client even without the policy (last row of data)

Here is the data with and without the Policy enabled on the FW card

Transfer Speed FW card No policy NT4	Transfer Speed FW card Restrictive policy NT4	Transfer Speed FW Card No policy 2000 Srv
35.51Seconds 3688.40Kbytes/sec	39.94Seconds 3279.63Kbytes/sec	40.32Seconds 3248.64Kbytes/sec
32.93Seconds 3977.85Kbytes/sec	39.15Seconds 3345.82Kbytes/sec	40.87Seconds 3204.84Kbytes/sec
34.13Seconds 3837.75Kbytes/sec	39.36Seconds 3328.05Kbytes/sec	39.61Seconds 3306.96Kbytes/sec
36.26Seconds 3612.01Kbytes/sec	44.23Seconds 2961.11Kbytes/sec	41.55Seconds 3152.31Kbytes/sec
37.96Seconds 3450.08Kbytes/sec	43.11Seconds 3038.10Kbytes/sec	42.76Seconds 3063.04Kbytes/sec
36.46Seconds 3592.10Kbytes/sec	42.74Seconds 3064.47Kbytes/sec	45.84Seconds 2857.61Kbytes/sec
91.67Seconds 1428.81Kbytes/sec	150.27Seconds 871.62Kbytes/sec	140.55Seconds 931.90Kbytes/sec
91.72Seconds 1428.03Kbytes/sec	144.56Seconds 906.05Kbytes/sec	147.58Seconds 887.51Kbytes/sec
223.71Seconds 585.48Kbytes/sec	149.57Seconds 875.70Kbytes/sec	145.88Seconds 897.85Kbytes/sec

Next, I added one more FW card to my configuration and set up a standard Windows 2000 IPSEC association between two Windows 2000 servers (same hardware and memory as the Windows NT 4.0 server, just dual booted) and tested the throughput between these two machines with and without the FW card. See the following data table. As you can see, the card made a huge difference and actually brought the speeds back to what they were without 3DES encryption. With 3DES encryption, the transfer times almost tripled, but with the EFW card the speeds were brought back down to the 36-40 second range.

	Transfer Speed Normal Card IPSec 3DES Encryption 2000 Srv	Transfer Speed FW card IPSec 3DES Encryption 2000 Srv
From 2000 Server Normal Card IPSEC 3DES Encryption	111.54Seconds1174.26Kbytes/sec 118.08Seconds1109.24Kbytes/sec 118.34Seconds1106.80Kbytes/sec	
From 2000 Server FW card IPSEC 3DES Encryption		36.64Seconds 3574.45Kbytes/sec 43.01Seconds 3045.17Kbytes/sec 37.98Seconds 3449.08Kbytes/sec

Manageability

The software was easy to install and relatively easy to use. I felt the interface was a little difficult to navigate and not always intuitive, but once I got used to it, it was easy to get around.

The software allows a moderate amount of customization. The firewall rules are as flexible as a non-stateful firewall can be. The standard rule sets come in an XML file and new rule sets are saved in the same way. This will allow further flexibility in the future as XML is, by design, very portable. Being able to use groups of computers and groupings of rules helps the manageability significantly. One thing that could be improved to make it more flexible would be to allow the NIC's to be members of more than one group and adjust the policies to be cumulative. This would allow more effective "graded" security policies

The NIC's can't be configured without the all three of the components of the system being installed. The system consists of a Policy server, a Management Console, and the Embedded Firewall Device (EFW). Each EFW must be associated with one policy server as a primary server but can be associated with up to 3 policy servers with 2 operating as secondary servers. There is no local workstation utility with which to manage a NIC that is installed there unless you install all 3 components on a single machine.

The secondary servers increase the fault tolerance of the system because a secondary server can take over the management of the EFW if the primary fails. Also, multiple management consoles can manage the policies on the policy servers, but only one console can connect to any policy server at a time for editing. Communications between the policy server and each EFW device as well as the management server are encrypted and authenticated with a private/public key pair. This key set is unique to each EFW domain and must be backed up and protected to ensure that the system could be recovered if the policy server failed. By design, no other installation of the policy server would be able to control/manage the EFW devices without these keys. Each EFW device also caches a "fallback" policy in the event that it can't contact one of its policy servers. This can be configured in a restrictive or permissive manner depending on your requirements.

Logging

The logging in this product can be pretty extensive but it is a little difficult to use. It is kept in database files on the policy server and zipped up based on size. A query tool must be used to extract data from the log files. The log files can contain packet data, data from policy installs, and other debugging information.

The fidelity of the logs was impressive. Logging was able to capture and display the hex as well as to give the standard decodes of source and destination etc... I was unable to find out if there was a limit to the "snap-length" or amount of hex data kept, but I doubt

the default is greater than 64 bytes and I saw no documentation of any way to change that. Below is a sample log entry:

```
0     11/5/02 15:53:28:589 MST      Policy Rule Match       Warning     al-
nt4 10.150.9.159 Restricted   Restricted to FTP 11 Default 00:b0:d0:e4:2d:e3
ff:ff:ff:ff:ff:ff IP UDP 10.150.107.140 10.150.255.255      137   137 FALSE
      Deny
ffffffffffff00b0d0e42de30800450000606d01000080114cd40a966b8c0a96ffff008900890
04cc6ac8030281000010000000000012045444542454544424341434143414341434143414341
434143414341434149480000200001c00c00200001000493e0000600000a966b8c
```

The log fields are in the following order starting from the first line: Unique ID number, date, time, Audit code, Category, Device Name, Policy Server (IP address), Device Set, Policy Name, Policy Version, Rule Number (default), Src MAC Addr, Dst MAC Addr, MAC Type (IP), IP Protocol, Src Addr, Dst Addr, Src Port, Dst Port, Testing Mode (false), Action, Packet Data.

The logs are centralized on the primary policy server, but there was no facility for sending the logs to a syslog server or other central log analysis server.

Alerting

This was the most disappointing part of the product. There really was no mechanism at all for alerting or reporting on violations of security policy. Since the log files are proprietary and don't allow for syslog or other output, there is no way to monitor them effectively in real time or near real time. Therefore the usefulness of these devices as IDS sensors/nodes is severely limited. Certainly they could be used for forensic analysis or for capturing packet data during events that are known. It could also be used for collecting historical data on the activity of a specific machine, but they are not ready to be part of a real IDS system.

Conclusions

This is a good candidate for a distributed Host based firewall solution.

Because most of the work is done in hardware this solution has very few conflicts with any other software and it is remarkably fast. It is also a good choice because it has been designed so that it would be very hard to tamper with the system or disable any policies enforced from the policy server. For most functions the software driver doesn't even have to load for the policy to be enforced. This is a good addition to a "defense in depth" strategy because it makes it feasible and scalable to add managed host based firewalls to your infrastructure. Along with perimeter defenses and good patching policy, etc… this solution would be a big help in securing the enterprise.

Where it fits best for deployment is for sensitive workstations or servers with sensitive information and/or mission critical applications. Another great fit for it is in applications where encryption is required such as SSL, or VPN situations.

It is not ready for extensive use as an IDS device. I think the potential is there and the idea of having relatively cheap distributed sensors throughout your network is a compelling one, but these do not fit in that role yet. The best strength the EFW device offers as a part of an IDS strategy is in logging and capturing packet data. This is where many commercial IDS systems fail to a large degree. It is hard to get packet data with any decent fidelity out of them. These EFW devices can capture significant amounts of packet data but they are hard to manage and the logs are difficult to use.

References

"Features and Benefits." 3Com® Embedded Firewall Policy Server. Product #: 3CR010PS-1-97B URL: http://www.3com.com/products/en_US/detail.jsp?tab=features&pathtype=purchase&sku=3CR010PS-1-97B (Jan. 3, 2003)

"Features and Benefits." 3Com® 10/100 Secure NIC. Oct. 31, 2002. 3CR990-TX-97. URL: http://www.3com.com/products/en_US/detail.jsp?tab=features&pathtype=purchase&sku=3CR990-TX-97 (Jan. 3, 2003)

"EtherLink 10/100 PCI Network Interface Card with 3XP Processor User Guide." v1.1. June 5, 2000. URL: http://support.3com.com/infodeli/tools/nic/3cr990/UsrGd_11.pdf. (Jan 3, 2003)

"How to Configure a L2TP/IPSec Connection Using Pre-shared Key Authentication." Oct. 10 2002. URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;240262. (Jan 3, 2003)

"Step-by-Step Guide to Internet Protocol Security (IPSec)" Feb. 17, 2000. URL: http://www.microsoft.com/windows2000/techinfo/planning/security/ipsecsteps.asp . (Jan 3, 2003)

PART 2 - NETWORK DETECTS

Detect #1 BACKDOOR Q access

1. Source of Trace:

This trace was obtained from http://www.incidents.org/logs/Raw. It is from the 2002.11.13 file but I analyzed similar traffic from 11/1 - 11/14 files. In these traces the obfuscated IP range is a class B range of 207.129.x.x. I can't be sure about the placement of the sensor. But, I must assume that the sensor is between an Internet facing router and an Internal network router because of the variety of source addresses seen in these files (not the packets for this detect but other traffic seen by the sensor) and because the MAC addresses found in virtually every packet are always consistent and belong to ranges assigned to Cisco Systems.

2. Detect Generated by:

Snort 1.9.0 running on a Linux 2.4.18 kernel using the "snortrules-stable" standard rule set from 12/6/2002 with no modifications. The command used extract the data of interest was: snort -r 2002.10.13 -A console -q port 31337 -h MY.NET.0.0/16 -d. With the "-A console" option alerts were sent to the console screen but further application layer data was sent to the standard log output as shown below.

The Snort alert shown below gives the name of the alert, it's assigned classification and priority followed by the Protocol and the source and destination IP's and port numbers. These fields can all be easily seen within the structure of the rule that generated this alert. The rule states that it will match on any TCP traffic from a broadcast address on any port to the home network on any port with TCP flags of at least ACK set and a data payload size of at least 1 byte. The rule also includes a reference for where you can find out more information on the attack.

Snort Rule:

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;)

Example Alert:

```
[**] [1:184:3] BACKDOOR Q access [**] [Classification: Misc
activity] [Priority: 3] {TCP} 255.255.255.255:31337 ->
MY.NET.225.96:515
```

Example Application layer Dump:

```
11/13-15:55:13.576507 255.255.255.255:31337 -> MY.NET.225.96:515
TCP TTL:15 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
63 6B 6F                                          cko
```

=+

3. Probability the Source Address was spoofed:

Very High - The source address is a broadcast address and at first glance that would seem very normal except that what is normal is for the destination address to be a broadcast address. I'm not aware of any normal process that would create packets whose source is a broadcast address.

Other aspects of the packets indicate that they are crafted in some way. Since we see the ACK and RST bits set, we might think this is part of some broken TCP connection, but the sequence numbers and acknowledgment numbers are all set to zero. Having both the ACK and RST bits set together in this packet is also interesting. It probably indicates crafting but not necessarily. More discussion below.

4. Description of Attack:

Q is a program originally written by "Mixter" as a

> "client / server backdoor which features remote shell access with strong encryption for root and normal users, and a encrypted on-demand TCP relay/bouncer that supports encrypted sessions with normal clients using the included tunneling daemon. Also has stealth features like activation via raw packets, syslog spoofing, and single on-demand sessions with variable ports. " (http://packetstormsecurity.org/groups/mixter/)

This Snort alert indicates activity that is consistent with the existence of a “Q backdoor” on your network or scanning for/controlling such a backdoor. The packets that set off this alert all came from a broadcast source address (255.255.255.255) on port 31337 (a favorite port of hackers spelling "eleet") to internal addresses on port 515 which is normally the LPD daemon port.

5. Attack Mechanism:

I did quite a bit of research to try to understand this attack and there doesn't seem to be much consensus on exactly what is going on here.

Some have analyzed this traffic and concluded that it was some sort of broken reconnaissance tool looking for holes in the LPD daemon. (Trenton Riddell - http://www.giac.org/practical/Trenton_Riddell_GCIA.doc) Some have declared the traffic harmless (http://online.securityfocus.com/archive/75/194288) A few different sources have linked this activity to IRC servers and/or possible worm activity. (http://online.securityfocus.com/archive/75/182244/2002-11-04/2002-11-10/1)

From my understanding of the "Q" tool and this traffic, I propose a different explanation.

This traffic is probably from a tool based on the randomness of the target IP addresses. It may even be a try at a worm looking for a previously infected "Q" server, but it is not harmless. "Q" and specifically the "libmix" library that it is based upon can activate code on the target system with raw packets or "command" packets that control the daemon running on the target system. This target system could then send a shell prompt to a specific location, relay encrypted traffic, or perform other tasks.

I believe that the packets are crafted the way they are in order to bypass perimeter security and firewalls. Many firewalls will allow ACK and/or RST packets through as part of an established TCP session. Firewalls may also allow the source address of broadcast through since it isn't expected behavior on the network. The attacker needs no established session because the trojaned "Q" system will do his/her bidding based on the command string embedded in the packet. Therefore it makes sense to hide your identity if you are the attacker by spoofing the broadcast address as the source address.

Another interesting aspect of these packets is that they all (over a period of 2 weeks in the traces) have an IP TTL value of 15 with one packet on 10-12 with a TTL of 14. Since the rest of the packet has pretty obviously been crafted the initial TTL may have been crafted as well but having it this consistent over 2 weeks seems to point to a couple of things. The source of this attack is almost certainly one lone machine. Either the packets are crafted with an initially low TTL value and the attacker is on or very near the target network or they have a very consistent path (not dial-up) to the target network. Since the MAC addresses in the Ethernet headers seem to belong to Cisco's and are also consistent in virtually every packet I must assume that the attacker is probably not on the target network, but is very close and/or has a consistent network connection.

Another possible advantage of using this type of packet as a "trigger" or communication method for the "Q" trojan running on a remote machine is that it will be ignored by many Intrusion analysts. Snort and other IDS systems will detect the initial packet, but they will probably not trigger on any response to this control packet because it would be seen as normal IP activity coming from an internal host. The stateful firewalls will happily add the connection information into their tables and allow the return traffic to go through as if the internal system were surfing the web or initiating on an SSL session with a remote server. According to the documentation that comes with the "Q" source code the connection can be encrypted for "anonymity."

Since the source packet is a non-standard packet, it should not elicit even an ICMP error message from the host and even if it did, the message would be addressed to the broadcast address and never make it past the router on that segment. It could be that the attacker wants to elicit a response to the packet destined for the broadcast address in order to scan for specific types of hosts (OS etc...) or to efficiently scan for other vulnerable systems from an already compromised host.

Not having a full tcpdump trace, I can't ascertain whether or not the target systems "answered" in any way. They may have returned the packet with some other specific

code destined for the broadcast address that activated or shut down every trojan on the same LAN segment. The sensor would not have seen it since it appears to be on a segment between two routers and not on the same segment as the target machines. Or since the target machines might have "answered" by establishing an entirely new TCP/UDP session with the remote host an analyst would have to know what is "normal" traffic for any host studied.

These packets are probably stimulus packets. They couldn't really be responses because the source address is the broadcast address. They are also most likely reconnaissance or active exploit. The packets all have a very small payload -- "cko" . Therefore it can't really be encrypted but is probably a command that an installed "Q" trojan would respond to by starting a new connection to a pre-configured host, or possibly triggering a DOS response or some other action. Therefore I would categorize this as either a reconnaissance attack or an active exploit. My best guess would be reconnaissance since each IP is targeted only once and most aren't repeated even once throughout the 2 week range of data. Looking into the documentation that came with the "Q" source code, there isn't any built in "cko" command but it seems pretty likely that it would be easy enough to add this command string to the code before compiling it.

6. Correlations:

Reference for the Snort signature:
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids203&view=signatures

CVE number. General CVE candidate to cover multiple trojans/backdoors.
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0660

Security Focus Postings about similar traffic:
http://online.securityfocus.com/archive/75/182244/2002-11-04/2002-11-10/1

Insecure.org mailing list:
http://lists.insecure.org/lists/incidents/2001/Jun/0265.html

GCIA paper with same traffic:
http://www.giac.org/practical/Trenton_Riddell_GCIA.doc

Source code for "Q" application:
http://mixter.warrior2k.com/

7. Evidence of Active Targeting:

I don't think the attacker targeted this specific IP. The fact that many different IP addresses were attacked over a period of 2 weeks with IP addresses rarely being repeated and never in any sequential order seems to indicate that it was a scan of IP

addresses done somewhat randomly. I would guess it is a script or a tool that the attacker may have fed some IP ranges into after some initial reconnaissance. The fact that the packets are normally spread in time throughout the day indicates that if it is a script or a tool, the machine running it is also scanning a large number of other IP ranges not in this class B network. Another possibility is that this traffic is triggered after activity on an IRC channel as suggested in some of the Security Focus postings (http://online.securityfocus.com/archive/75/182244/2002-11-04/2002-11-10/1)

8. Severity:

The severity will be calculated using the following formula on a scale from 1 to 5, where 1 is lowest and 5 highest:

severity = (criticality + lethality) - (system countermeasures + network countermeasures)

Criticality = 3 - The purpose of this specific system is unknown. Therefore a moderate to high value is most appropriate as the criticality of the system.

Lethality =4 - It is unknown what a successful compromise of a system might bring. It could range from simple remote access to possibly root access or DOS against internal or external entities.

System Countermeasures = 2 - Unknown so I am assuming they could be minimal.

Network Countermeasures = 2 - Unknown so I am assuming they could be minimal as stated above. It seems plausible that this class B network has been targeted because the attacker thinks it already contains or is likely to contain compromised "Q" trojan machines. It is only one machine attacking and if it were truly scanning across the Internet and using all addresses, I doubt there would be this many hits per day in this class B network.

(3 + 4) - (2 + 2) = 3

9. Defense Recommendations:

a. Review current router ACL's and firewall policy to ensure that traffic destined for port 515 is not allowed.

b. Monitor traffic inside of the current firewall (possibly with another snort sensor) to ensure that these crafted packets are not traversing the firewall inadvertently.

c. Employ egress filtering if at all possible to prevent any unwanted connections both now and in the future from any compromised systems.

d. Deny access to IRC servers if it is not necessary for the business or organization.

e. Look for traces of the "Q" trojan on machines in your infrastructure. Also, deploy a product such as "tripwire" to monitor the integrity of key systems to detect when trojans or other unauthorized software have been installed.

10. Multiple Choice Question:

Unsolicited TCP packets arriving at your perimeter firewall with invalid addresses and/or improper TCP flags set can be used?

a) to bypass the firewall by mimicking established TCP sessions.

b) for port scanning by deciphering the ICMP messages or lack thereof that get returned to the attacker.

c) to "remote control" active trojans, triggering other connections, starting DDOS tools, or opening up other backdoors into your network.

d) all of the above

Answer: d) all of the above

This really emphasizes the importance of the analysis of whether or not the source address is being spoofed. Too often when the protocol is TCP analysts can make the mistake of assuming that the source address could not be spoofed because of the TCP 3-way handshake. Here is yet another reason to question that assumption before jumping to a conclusion.

Detect #2 Malicious DNS?

1. Source of Trace:

This trace comes from a client network. It was first noticed on a PIX firewall. The traffic was then also confirmed on a Check Point firewall and later analyzed from tcpdump logs collected by a Shadow sensor on the network. The basic stick drawing is below. There are no publicly accessible DNS servers in this network. The DNS server in this trace sits on the internal network and is NATed to a public address on the way out.

Internet ---Router ---Sensor ---Ckpt FW---screened subnet ---PIX ---Internal Net

2. Detect Generated by:

Cisco PIX 515 set up in a fail-over configuration with 6 interfaces. 4 interfaces are currently used, 1 - screened subnet, 2 - Internal network, 3 - Internal screened subnet, 4 - unused, 5 - VPN segment, 6 - state sync.

The Check Point firewall is also set up in a fail-over configuration running NG FP2 on Nokia appliances running a 3.5 version of the IPSO code. The Check Point machines have 3 interfaces, 1 - Internet, 2 - screened subnet, 3 - state sync.

The Shadow sensor is running on Red Hat 8.0 with tcpdump version 3.6.2 which comes with the Shadow 1.7 distribution. After the first detect of the strange DNS traffic, I started another instance of tcpdump with a snap length of 256 in order to capture the payload of the DNS traffic. I also used Ethereal 0.9.6 to further analyze the traffic.

The PIX firewall is fairly standard in it's rule set. It has most default fix-ups so it keeps state information on most traffic passing through, including UDP. However, fix-up of DNS is not enabled so DNS is currently following the default UDP stateful inspection. It is using access-lists along with the default PIX rules of denying any traffic not specifically allowed from a less trusted interface to a more trusted interface.

The Check Point firewall is pretty tight in its rules. It employs both ingress and egress filtering. Client machines are generally not permitted to access any resources on the Internet but use internal proxy servers and internal DNS servers to access Internet resources.

Both firewalls allow the internal DNS server in this trace to make DNS requests outbound and statefully accept DNS return traffic.

PIX example Alert:

Dec 12 09:39:17 [PIX.INTERFACE.IP.ADDR.2.2] %PIX-4-106023: Deny udp src outside:128.242.107.15/53 dst lan:client.internal.DNS.IP/53 by access-group "from-outside"

At the beginning of the alert you can see the time-stamp portion. This client is using all local time zone settings. Next in brackets is the IP identifier of the PIX with the actual alert following. The number after the %PIX portion is the severity level assigned by the PIX and the message number following that. For this alert it is a severity 4 with message 106023 which means "An IP packet was denied by the access-list" (http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_62/syslog/pixemsgs.htm#42801). In this case a UDP packet coming into the interface named "outside" from 128.242.107.15 on port 53 (DNS) destined for the interface named "lan" and the IP address of the internal DNS server was denied by the access-group named "from-outside."

Check Point example Alert:

"5101" "12Dec2002" " 9:39:35" "VPN-1 & FireWall-1" "eth-s5p1c0" "internal.fw.ip.addr" "log" "drop" "domain-udp" "bs1.sjc.mirrorimage.net" "client.internal.DNS.IP" "udp" "52" "" "" "" ""

The Check Point log entry takes a little more explanation. For this I will refer to an excellent paper describing these fields quite well. John Ryan in his paper "Security Logs and Checkpoint Firewall-1" at http://rr.sans.org/firewall/logs.php describes the logging format with:

> "The first record of the output file will contain the field names to simplify importing this file into other programs. It will be similar to the following though the order may be different.
>
> num, date, time, orig, type, action, alert, i/f_name, i/f_dir, proto, src, dst, service, s_port, len, rule, icmp-type, icmp-code, xlatesrc, xlatedst, xlatesport, xlatedport, message, user, reason, scheme:, methods:, srckeyid, dstkeyid, sys_msgs
>
> The following table defines the fields.

num	Record number
date	Date record was written
time	Time record was written
orig	Which firewall is writing the record
type	Log entry or Alert
action	Accept or Drop
alert	Kind of alert generated, if any
i/f_name	Firewall interface that the traffic was seen on
i/f_dir	In relation to the firewall, inbound or outbound
proto	TCP, UDP, ICMP
src	Source IP address
dst	Destination IP address
service	Destination port
s_port	Source port
len	Packet length
rule	Firewall rule that triggered the log Entry
icmp-type	ICMP Type
icmp-code	ICMP Code
xlatesrc	NAT, the source IP that was translated
xlatedst	NAT, the destination IP that was translated
xlatesport	NAT, the translated source port
xlatedport	NAT, the translated destination port
message	Firewall message to explain an action
user	User authenticated by the Firewall
reason	Is Encryption happening, authentication?
scheme:	Encryption Scheme being used

methods:	Encryption protocol being used
srckeyid	Key scheme used by source IP
dstkeyid	Key scheme used by destination IP
sys_msgs	System messages”

My example log contains most of the fields at the beginning of this list but is missing most of the fields at the end because they either were not included or not applicable to this packet.

Example TCP dump output for DNS exchange

```
client.internal.DNS.IP.53 > 128.242.107.15.53:  12543 A? cserver.mii.instacontent.net.
(46) (DF)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 128.241.221.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 192.147.176.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 204.0.99.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 168.143.179.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 168.143.179.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 66.250.213.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 66.250.213.115 (62)
128.242.107.15.53 > client.internal.DNS.IP.53:  12543*- 1/0/0 A 128.242.107.190 (62)
128.242.107.15.55555 > client.internal.DNS.IP.53:  12543 A?
cserver.mii.instacontent.net. (46)
```

This trace shows the DNS packets going in and out of the network. Notice the seemingly normal traffic on port 53. TCPDUMP recognizes it as normal DNS traffic and decodes the DNS requests and answers. Also notice the "*" which indicates it is an authoritative answer, the "-" which indicates that recursion is not available, and notice the "1/0/0" which indicates that this packet has 1 answer 0 referrals to other authoritative servers and 0 additional records.

Example Application layer Dump for the last 2 packets:

```
128.242.107.15.53 > client.internal.DNS.IP.53:  [udp sum ok] 12543*- q: A? cserver.
mii.instacontent.net. 1/0/0 cserver.mii.instacontent.net. A 128.242.107.1
90 (62) (ttl 242, id 52305, len 90)
0x0000   4500 005a cc51 0000 f211 2cb5 80f2 6b0f        E..Z.Q....,...k.
0x0010   d084 1306 0035 0035 0046 8ecb 30ff 8400        .....5.5.F..0...
0x0020   0001 0001 0000 0000 0763 7365 7276 6572        .........cserver
0x0030   036d 6969 0c69 6e73 7461 636f 6e74 656e        .mii.instaconten
0x0040   7403 6e65 7400 0001 0001 c00c 0001 0001        t.net...........
0x0050   0000 0258 0004 80f2 6bbe                                ...X....k.

128.242.107.15.55555 > client.internal.DNS.IP.53:  [udp sum ok] 12543 A? cserver.mi
i.instacontent.net. [|domain] (ttl 242, id 52779, len 74)
0x0000   4500 004a ce2b 0000 f211 2aeb 80f2 6b0f        E..J.+....*...k.
0x0010   d084 1306 d903 0035 0036 e939 30ff 0000        .......5.6.90...
0x0020   0001 0000 0000 0000 0763 7365 7276 6572        .........cserver
0x0030   036d 6969 0c69 6e73 7461 636f 6e74 656e        .mii.instaconten
0x0040   7403 6e65 7400 0001 0001                                t.net.....
```

3. Probability the Source Address was was spoofed:

Very very low - This is a DNS server that is actually supplying valid addresses and names. The communication appears to be non-standard though and points to some well known issues with stateful firewalls and UDP. Nothing in the content of the packets indicates crafting of any kind. More discussion below.

4. Description of Attack:

What is happening at the firewalls is that they are opening stateful connections for outbound DNS traffic as they are supposed to and therefore allowing returning DNS traffic from the target DNS server back into the network for a pre-determined time before closing that "hole." This traffic keeps returning past that time-out and then is dropped by the firewalls. The stateful "hole" in the firewall allows some non-standard DNS traffic to enter the network that could just as easily be malicious.

5. Attack Mechanism:

I first noticed this traffic when I saw almost simultaneous drops in the firewall logs of both the PIX and the Check Point for inbound DNS traffic from the same source. At first it appeared that there was a problem with the Check Point firewall and that it was actually passing UDP traffic even though it logged it as being dropped. Not knowing if the times of all of the log sources were synchronized, I couldn't distinguish between the packets because there were no other packet identifiers in the logs. Therefore I couldn't be sure if it was really the same packet or not.

Looking at tcpdump raw data from the Shadow sensor I saw packets with the same pattern as above. What appears to be happening is that the internal DNS server is making a DNS request to the target server. The target server then responds with a whole series of DNS packets, from 9 to 16 of them, each with only one authoritative answer. Lastly the target server sends a DNS query of it's own from port 55555 to the normal DNS port of the source server. The query contains one question that is the same as the original one asked by the internal DNS server. This is not normal behavior for DNS. Normally the answer would return in one packet with multiple entries if required. If there are too many entries to return the data in a standard DNS packet, then a TCP connection is established.

What appears to be happening at the firewalls is that because they have slightly different time-outs for UDP/DNS entries in their respective state tables, a few more response packets make it through to the PIX than the PIX allows through to the internal network. They are both at the defaults so the Check Point firewall allows the DNS responses for 40 seconds and the PIX allows the responses for 30 seconds. With these time-outs, any packets returning after 30 seconds would be blocked on the PIX and show up in the logs for 10 seconds before any packets would be blocked at the Check Point and show up in those logs. Depending on the latency of the logging and the timing

of the return packets it might look like the drops hit the logs at almost the same time. Without having the data in the IP header the two packets might also look like identical packets.

Why would a DNS server do this?

One suggestion made by analysts on the Intrusions mailing list at incidents.org was that since mirror-image.net is a distributed web content provider, they are probably doing some strange things with their DNS servers in order to balance content across their entire network. I have trouble with this explanation since the TTL's in all of the packets are between 245 and 242. If it were truly distributed DNS the TTL's should vary by quite a bit.

Another possibility would be that the remote server is using non-standard DNS in order to elicit responses from querying DNS servers and collect data about the responses. I find it very odd that the exchange always ends with a DNS query from the remote DNS server with a source port of 55555. Are they trying to map which DNS answers got put into the cache of the requesting server? If they keep a table of which answers they gave, they could potentially map characteristics of the firewall and the DNS servers they are responding to. I envision a DNS specific version of Nmap's OS detection capabilities. Although I think this activity is probably non-malicious it points to the problems of stateful firewalls especially in dealing with UDP and ICMP. Even if no scanning is being done as I suggest it might be it is certainly possible for a company or hacker to write a program to mine data in this way. Taking advantage of the stateful way in which most firewalls must handle DNS traffic --with timers, this application could covertly be attempting to gain more data about the client DNS servers over the same stateful "conduit."

I just found the answer on mynetwatchman.com while writing and researching this! Apparently the company uses Cisco's "Boomerang" product so several servers answer the query as fast as they can. The first server that answers "wins" in order to send the client to the closest server for the content. They also admit to querying the DNS server to see if they can find out which of their servers "won." (http://www.mynetwatchman.com/LID.asp?IID=1550509)

DNS is, however, historically a favorite target of hackers that can yield very important compromises. If there were another tool to mine data about BIND versions and possibly firewall behavior at the same time, it could put an additional advantage in the hands of attackers.

6. Correlations:

E-mail thread on the topic on incidents.org mailing list

From: "James C Slora Jr"
To: Al Maslowski-Yerges, GATEWAY:incidents.org:intrusions,
Date: Wednesday - December 4, 2002 12:35 PMSubject: RE: Has anyone seen this odd(to me) DNS traffic[Attachments] Mime.822 (3490 bytes) [View] [Save As]<> This looks like a dynamic cache server system. Look at instaContent servicesat <>http://www.mirror-image.com/services/contentdelivery.html
It's typical (for me anyway) to get state problems from cache servers.DNSspoof alerts don't sound too farfetched either, given that their system isconstantly moving content and redirecting queries.

- Jim

-----Original Message-----From: Phil Brossman Sent: Wednesday, December 04, 2002 11:20 AMTo: 'Al Maslowski-Yerges'Cc: 'intrusions@incidents.org'Subject: RE: Has anyone seen this odd(to me) DNS traffic

I'm glad you asked, I've been seeing these on both of my DNS servers formonths. BlackIce reports these as Successful DNS Spoofs.

Anyone have any ideas?

-----Original Message-----From: Al Maslowski-Yerges Sent: Wednesday, December 04, 2002 9:37 AMTo: intrusions@incidents.orgSubject: Has anyone seen this odd(to me) DNS traffic

I have been seeing some odd DNS behavior on my network. As far as Icantell so far, it doesn't seem particularly malicious, but it shows up in myfirewall logs because the state table times out before all of the trafficcomes in. Any help would be appreciated. Below is a representative tcpdumpof the traffic taken outside the perimeter firewall.

```
07:31:11.255940 MY.DNS.19.6.53 > 128.242.107.15.53:  [udp sum ok] 29727
A?instacontent.mirror-image.net. [|domain] (DF) (ttl 254, id 18813, len
75)0x0000   4500 004b 497d 4000 fe11 6398 d084 1306
E..KI}@...c.....0x0010   80f2 6b0f 0035 0035 0037 646c 741f 0000
..k..5.5.7dlt...0x0020   0001 0000 0000 0000 0c69 6e73 7461 636f
.........instaco0x0030   6e74 656e 740c 6d69 7272 6f72 2d69 6d61
ntent.mirror-ima0x0040   6765 036e 6574 0000 0100 01
ge.net.....

07:31:11.414498 128.242.107.15.53 > MY.DNS.19.6.53:  [udp sum ok] 29727*- q:A?
<>instacontent.mirror-image.net. 1/0/0 instacontent.mirror-image.net.
A128.242.107.114 (63) (ttl 243, id 3319, len 91)0x0000   4500 005b 0cf7 0000
f311 eb0e 80f2 6b0f        E..[.........k.0x0010   d084 1306 0035 0035 0047
109c 741f 8400        .....5.5.G..t...0x0020   0001 0001 0000 0000 0c69 6e73
7461 636f        .........instaco0x0030   6e74 656e 740c 6d69 7272 6f72 2d69
6d61        ntent.mirror-ima0x0040   6765 036e 6574 0000 0100 01c0 0c00 0100
```

```
ge.net.........0x0050   0100 0002 5800 0480 f26b 72
....X....kr
```

CVE listing of BIND vulnerabilities:

http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=BIND

Another Posting asking about the same traffic

http://groups.google.com/groups?q=55555+DNS&hl=en&lr=&ie=UTF-8&selm=ae4i8o%24bk43%241%40isrv4.isc.org&rnum=1

Found the answer!!!!
http://www.mynetwatchman.com/LID.asp?IID=15505093

Cisco Documentation that verifies that the "Boomerang" feature works as described in the netwatchman.com article:
http://www.cisco.com/en/US/products/sw/iosswrel/ps1839/products_feature_guide09186a0080087d3f.html#xtocid1

7. Evidence of Active Targeting:

In this case, I would have to say this is active targeting. It is not malicious, but the traffic does take advantage of stateful firewall problems and the fact that many DNS servers are open to the Internet even though they serve internal clients. They are trying to actively glean information from the DNS servers that query their systems.

8. Severity:

The severity will be calculated using the following formula on a scale from 1 to 5, where 1 is lowest and 5 highest:

severity = (criticality + lethality) - (system countermeasures + network countermeasures)

Criticality = 5 - This server is the primary DNS server for all internal domains. If it was compromised or poisoned it could bring down entire internal systems including ERP, Intranet, etc...

Lethality =2 - This specific traffic is not a harm to the system at all but it shows the vulnerability of this and other DNS servers. It also shows the vulnerabilities of UDP with stateful firewalls.

System Countermeasures = 1 - This particular server is running a very old version of

BIND and is not properly patched at the OS level.

Network Countermeasures = 3 - The system is behind 2 firewalls and both have stateful inspection of UDP. Also, no access is granted to the DNS server from the outside. However, fix-up for DNS was not enabled on the PIX firewall and the Check Point doesn't have this feature so more traffic than necessary is allowed into the network.

(5 + 2) - (1 + 3) = 3

9. Defense Recommendations:

a. Turn on the DNS fix-up feature on the PIX firewall.

b. Patch the server with the latest patches for the OS and BIND. Many vulnerabilities have been found and patching them is extremely important.

c. Tune down the UDP time-outs if applications will allow this. Certainly keep them in synch between the firewalls.

d. After upgrading to BIND 9.x, restrict queries to the DNS server to only source addresses within the internal network to prevent unauthorized access by enabling this new feature of BIND on the server.

e. Consider using an architecture to separate DNS services for internal hosts from DNS services to resolve public Internet addresses.

f. Install a host-based intrusion detection system and a host-based firewall to help protect this important system.

10. Multiple Choice Question:

If all parts of a string of incoming UDP packets are exactly the same (source and dest. IP's, ports, etc...) and the time-stamps of the packets might be suspect, how can you track the packet stream definitively across several network devices?

a) The IP header ID field.

b) The sequence numbers in the packets.

c) They will all have the same TTL value.

d) The Ethernet MAC addresses will be the same.

Answer: a) The IP header ID field.

This became a problem for me because I couldn't discern whether or not the packet dropped on the Check Point firewall was the same as the packet also dropped on the PIX firewall. I'm not aware of any standard logging on commercial firewall and IDS systems that include the IP ID. Here is one instance where having the raw packet data as available with Snort and Shadow is important in analyzing what is really happening on a network. In fact the analysis of the raw data prevented me from spending hours rebuilding the Check Point firewalls for this client as was suggested by Check Point support.

Posting to Incidents.org and responses

I posted 2 of my 3 detects to the website and Anton A. Chuvakin, Ph.D. was kind enough to ask some questions on this one. Below is the e-mail exchange with his questions and my responses. (http://cert.uni-stuttgart.de/archive/intrusions/2002/12/msg00210.html)

> Thanks for the questions! Here are my answers:
>
> **1. What examination was done to conclude there was no crafting?** I looked through each of the DNS return packets and they seemed to fit a standard response other than the fact that they each had only one authoritative answer in them. The IP ID's seem normal and change as expected. The UDP ports match the request. The DNS response is formatted in a standard way, and has an answer to the questioned asked (nothing extra for poisoning the cache etc...) The last packet in the string comes from port 55555 but it also includes standard UDP and DNS formats although it is a query. I wondered about the consistency of the source port and did wonder about crafting but since it always followed this same pattern I was leaning toward an app. written to query the DNS servers that queried the host. When I finally found the mynetwatchman.com reference it just confirmed my suspicions.
>
> **2. How would a host-based fw help in this case?** For this particular client, their DNS server only servers internal clients so my thought was that they could limit inbound DNS requests to only internal hosts and only allow just the responses to return.
>
> >>> "Anton Chuvakin, Ph.D., GCIA" <anton@chuvakin.org> 12/16/02 08:24AM
> >>>
> Al,
>
> >3. Probability the Source Address was was spoofed:
> ...
> >Nothing in the content of the packets indicates crafting of any kind.
> >More discussion below.
> Just curious, what examination was done to conlude this? I understand that

you discovered what was going on and it kind of makes crafting unlikely, but it is still interesting what was examined?

>f. Install a host-based intrusion detection system and a host-based
>firewall to help protect this important system.
How would a host-based fw help in this case? DNS server will alsways allow DNS traffic, right?

Best,
--
Anton A. Chuvakin, Ph.D.
GCIA Advisory Board Member
http://www.chuvakin.org
http://www.info-secure.org

Detect #3 Spoofed yahoo proxy scan

1. Source of Trace:

This trace comes from another client network. It comes from a Shadow sensor just inside the Internet router and before the firewall. The target hosts in this trace sit on the internal network and are NATed to a public address on the way out.

Internet ---Router ---Sensor ---CheckPoint FW---Internal Net

2. Detect Generated by:

A Shadow sensor running on Red Hat 8.0 with tcpdump version 3.6.2 which comes with the Shadow 1.7 distribution generated this detect. The Shadow system is set up in a fairly standard configuration with a "stealth" interface running on the sensor which feeds raw tcpdump files on an hourly basis to the analysis station. The analysis station runs several perl scripts against the data using customized tcpdump filters to generate the hourly and daily html files showing "traffic of interest". This "traffic of interest" includes port scans, traffic to or from known trojan ports, fragmented traffic, TCP options being set, among other non-standard or non-expected traffic. These html files are reviewed by security staff daily.

The Check Point firewall is set up in a fail-over configuration running NG FP2 on Nokia appliances running a 3.5 version of the IPSO code. Logs are sent to the management server for analysis and review. The management server is running Solaris 9 with all of the latest service packs and hot-fixes.

Shadow Alert:

```
209.139.204.164 > my.c.net.230
03:27:04.815668 srhst19.yahoo.com.0 > my.c.net.230.3128: S 267916:267916(0) win 512 (DF)
03:27:04.846622 srhst19.yahoo.com.0 > my.c.net.230.80: S 267917:267917(0) win 512 (DF)
03:27:04.878132 srhst19.yahoo.com.0 > my.c.net.230.8080: S 267918:267918(0) win 512 (DF)
03:27:04.914623 srhst19.yahoo.com.0 > my.c.net.230.1080: S 267919:267919(0) win 512 (DF)

209.139.204.164 > my.c.net.160
08:10:16.719687 srhst19.yahoo.com.0 > my.c.net.160.3128: S 318448:318448(0) win 512 (DF)
08:10:16.750399 srhst19.yahoo.com.0 > my.c.net.160.80: S 318449:318449(0) win 512 (DF)
08:10:16.780863 srhst19.yahoo.com.0 > my.c.net.160.8080: S 318450:318450(0) win 512 (DF)
08:10:16.812434 srhst19.yahoo.com.0 > my.c.net.160.1080: S 318451:318451(0) win 512 (DF)
08:12:22.255619 srhst19.yahoo.com.0 > my.c.net.125.3128: S 319004:319004(0) win 512 (DF)
08:12:22.286177 srhst19.yahoo.com.0 > my.c.net.125.80: S 319005:319005(0) win 512 (DF)
08:12:22.318284 srhst19.yahoo.com.0 > my.c.net.125.8080: S 319006:319006(0) win 512 (DF)
08:12:22.350238 srhst19.yahoo.com.0 > my.c.net.125.1080: S 319007:319007(0) win 512 (DF)
08:32:29.056479 srhst19.yahoo.com.0 > my.c.net.150.3128: S 322776:322776(0) win 512 (DF)
08:32:29.092935 srhst19.yahoo.com.0 > my.c.net.150.80: S 322777:322777(0) win 512 (DF)
08:32:29.119427 srhst19.yahoo.com.0 > my.c.net.150.8080: S 322778:322778(0) win 512 (DF)
08:32:29.183939 srhst19.yahoo.com.0 > my.c.net.150.1080: S 322779:322779(0) win 512 (DF)
```

In this trace, you can first see the time, down to 6 decimal places. Next is the source address as resolved by DNS on the Shadow analyzer station along with the source port. Following is the direction indicator ">" and then the destination address with the destination port. After the colon are the TCP flags that were set. Next comes the TCP sequence numbers with the amount of data indicated in the "()" marks. Then comes the window size and then finally the IP flags if any. In this case the don't fragment bit was set (DF).

Check Point logs of the traffic:

```
"526"  "10Dec2002"  " 1:36:47"  "VPN-1 & FireWall-1"  "eth-s5p1c0"
"internal.fw.ip.addr"  "log"  "drop"  "3128"  "srhst19.yahoo.com"  "my.c.net.50"
"tcp"  ""  ""  ""  ""  " message_info Invalid TCP packet - source / destination port
0"
"957"  "10Dec2002"  " 3:26:39"  "VPN-1 & FireWall-1"  "eth-s5p1c0"
"internal.fw.ip.addr"  "log"  "drop"  "3128"  "srhst19.yahoo.com"  "my.c.net.230"
"tcp"  ""  ""  ""  ""  " message_info Invalid TCP packet - source / destination port
0"
"4535"  "10Dec2002"  " 8:09:50"  "VPN-1 & FireWall-1"  "eth-s5p1c0"
"internal.fw.ip.addr"  "log"  "drop"  "3128"  "srhst19.yahoo.com"  "my.c.net.160"
"tcp"  ""  ""  ""  ""  " message_info Invalid TCP packet - source / destination port
0"
"4586"  "10Dec2002"  " 8:11:56"  "VPN-1 & FireWall-1"  "eth-s5p1c0"
"internal.fw.ip.addr"  "log"  "drop"  "3128"  "srhst19.yahoo.com"  "my.c.net.125"
"tcp"  ""  ""  ""  ""  " message_info Invalid TCP packet - source / destination port
0"
"5178"  "10Dec2002"  " 8:32:03"  "VPN-1 & FireWall-1"  "eth-s5p1c0"
"internal.fw.ip.addr"  "log"  "drop"  "3128"  "srhst19.yahoo.com"  "my.c.net.150"
"tcp"  ""  ""  ""  ""  " message_info Invalid TCP packet - source / destination port
0"
```

Rather than explain the format of the Check Point logs again, I will refer you to my earlier detect above for that. However, notice a couple of things about these log entries.

The first thing to notice is that they start at 1:36 whereas the logs above only start at 3:26. This is because the Shadow logs for that hour are missing.

The second thing to notice is that it would seem that only the first packet of each scan was detected. This is an artifact of the log settings on the Check Point firewalls. They are set to ignore logging of consecutive alerts from the same IP address hitting the same rule. In this case it would be rule "0" which is part of the internal checkpoint settings. Therefore the additional packets to the other destination ports are not shown in this log.

3. Probability the Source Address was spoofed:

Very High (with explanation) - The numeric IP address is probably not spoofed because the person scanning probably wants to see the replies and they wouldn't see them if they spoofed the IP address unless they could ensure that they were somehow sniffing all traffic in the path between the source and the destination. What is definitely spoofed here is the reverse lookup of this IP within DNS. This makes it look like the attack was from a yahoo server and potentially throws off the analyst for a time. This behavior of sending incorrect information for the reverse DNS address can be very misleading and is against standards, but is certainly possible and not against the law as far as I know. It would also suggest that the ISP in this case is extremely loose or complicit in this deception.

Another aspect of the packets cause them to look crafted also. The IP ID is consistent across packets within a scan. This is not normal behavior. The IP ID should increment with each IP datagram sent.

4. Description of Attack:

This appears, at first, to be a classic RingZero scan. There are some differences as discussed below. It is unclear what the attacker is trying to do. It doesn't seem to be a random scan and I would say that it isn't the result of a normal RingZero trojan infection. Their other actions are very deliberate. They may be trying to evade some firewall devices by using a source port of "0" and they may be trying to throw off security personnel by registering their IP address(es) in their netblock with yahoo.com host names. Another possibility is that they are just trying to give yahoo some pain by creating an easily recognizable trojan scan and entering a reverse DNS entry that looks like it comes from a yahoo server so that yahoo will be inundated with calls claiming their server is attacking other networks.

My best guess is that they have modified the RingZero code and have been deliberately pointing it to a previously tabulated list of hosts on the Internet for several of the reasons listed above. They may indeed be trying to cause yahoo some pain, but the seemingly targeted nature of the scan (see discussion below) points to a further purpose. They are likely tabulating open proxies or verifying an old list that was created long ago in an attempt to do reconnaissance for later attacks.

Based on other posts and traffic I have seen, I think this is from a new RingZero tool out there that isn't really designed to use scripts but is probably fed data manually. The particular attacker in this trace is just also adding the yahoo.com reverse DNS entry to try to further conceal their activity.

5. Attack Mechanism:

The original RingZero trojan had many of the same characteristics of this scan. It scanned for open known proxy ports (3128, 1080, 8080) and http (80) and when it found and open machine, it attempted to install itself and send back confirmation to a central website that presumably collected the data and put it into a database for later use. The RingZero trojan also opened up the machine for later instruction, or reconfiguration by the attacker. (http://www.sans.org/resources/idfaq/ring_zero.php)

This attack appears to be a similar scan but it has some interesting changes. This scan is coming in with a source port of "0". Also, there appears to be only one or two source addresses from the same netblock. Other interesting behavior was noted by James Slora in a posting to the intrusions mailing list at incidents.org. He noted two separate scan events, one with port "0" as the source and the next event with variable source ports. In the port "0" source scans, he saw the scanning IP send a RST if the target answered with a SYN/ACK. No payload was seen. However in the second scan, he saw payloads with CONNECT strings to mail.ultrawebhost.com and mainin-02.mx.aol.com. Apparently the second scan was attempting to connect back to a remote site and either download data or put data back up to the server. The ultrawebhost.com domain refers back to splitinfinity.net and is ultimately owned by Verio. I assume this server is "owned" in one way or another by the attacker and is collecting data from scanned machines that answer. Another post from Michael Scheidell correlates the data sent by James as follows:

```
--------------------------------------------------------------------------------
#(2 - 163) [2002-12-09 20:58:30] [url/help.undernet.org/proxyscan/]
[snort/615]  SCAN SOCKS Proxy attempt
IPv4: 209.139.204.164 -> 208.237.120.134
     hlen=5 TOS=0 dlen=40 ID=772 flags=2 offset=0 TTL=115 chksum=7464
TCP:  port=0 -> dport: 1080  flags=******S* seq=139143
     ack=0 off=5 res=0 win=512 urp=0 chksum=41597
Payload: none
--------------------------------------------------------------------------------
#(2 - 162) [2002-12-09 20:58:30] [snort/620]  SCAN Proxy (8080) attempt
IPv4: 209.139.204.164 -> 208.237.120.134
     hlen=5 TOS=0 dlen=40 ID=772 flags=2 offset=0 TTL=115 chksum=7464
TCP:  port=0 -> dport: 8080  flags=******S* seq=139142
     ack=0 off=5 res=0 win=512 urp=0 chksum=34598
Payload: none
--------------------------------------------------------------------------------
#(2 - 161) [2002-12-09 20:58:30] [snort/618]  SCAN Squid Proxy attempt
IPv4: 209.139.204.164 -> 208.237.120.134
     hlen=5 TOS=0 dlen=40 ID=772 flags=2 offset=0 TTL=115 chksum=7464
TCP:  port=0 -> dport: 3128  flags=******S* seq=139140
     ack=0 off=5 res=0 win=512 urp=0 chksum=39552
Payload: none
```

The scan appears to be targeted because there are too many addresses scanned in

any one range to be truly random and because the scan doesn't cover a whole netblock. Instead it appears to normally scan 2 to 10 hosts for each network. The scans are also somewhat spread out in time and have TCP sequence numbers that increment fairly slowly showing that this box is not busy running a scripted scan all across the Internet. In fact, it doesn't seem to be too busy with other tasks at all unless the TCP sequence numbers are crafted as well. This scan behavior seems to be correlated in the incidents.org postings as well as the scans listed on mynetwatchman.com referred to below. They are also slow, targeted and from a single source to a limited number of hosts in the subnet. From this behavior I assume the attacker has some previously constructed list of IP addresses from prior reconnaissance. (Possibly web server logs or mail server logs?) I also think that the attacker is definitely compiling responses based on James' post referred to above. The attacker seems to have come back for a slightly different scan (with an HTTP CONNECT) on hosts that answered the first time.

6. Correlations:

A snippet of the E-mail thread on the topic on incidents.org mailing list:

> From: "James C Slora Jr"
> To: Al Maslowski-Yerges
> Date: Tuesday - December 10, 2002 10:47 AM
> Subject: RE: Yahoo server compromised?
> [Attachments] Mime.822 (4648 bytes) [View] [Save As]
>
> I don't think we're looking at root server compromise, or even DNS poisoning. It appears to be Johannes' option 3.
>
> >- they provide only reverse, but no 'regular' DNS for that IP/hostname
> > (odd, but things are not always RFC conform)
>
> The reason there is no forward lookup is because they don't have authority for yahoo.com.
>
> This is what I see: the spammers are running their own DNS servers (which are authoritative for reverse lookups on their netblocks) and providing false host name information that steps all over the honor codes of the DNS system.
>
> -----Original Message-----
> From: Al Maslowski-Yerges
> Sent: Tuesday, December 10, 2002 12:35 PM
> To: intrusions@incidents.org
> Subject: Re: Yahoo server compromised?
>
> The IP in my logs is 209.139.204.164.

Here is a tracert from network-tools.com:

```
TraceRoute to 209.139.204.164 [srhst19.yahoo.com]
Hop (ms) (ms) (ms) IP Address Host name
1 0 16 0 66.46.176.3 -
2 16 0 0 216.191.97.45 pos5-2.core2-mtl.bb.attcanada.ca
3 0 0 0 216.191.65.217 srp2-0.core1-mtl.bb.attcanada.ca
4 31 16 0 216.191.65.173 pos8-1.core2-tor.bb.attcanada.ca
5 0 16 0 216.191.65.244 srp2-0.gwy2-tor.bb.attcanada.ca
6 15 0 16 216.191.65.138 attca.peer.tor.gt.ca
7 16 31 0 66.59.191.5 ge3-0.wana-toroon.ip.grouptelecom.net
8 94 63 62 66.59.190.21 ge5-0.peera-vancbc.ip.grouptelecom.net
9 62 94 109 216.18.31.130 -
10 62 63 78 216.18.13.138 -
11 62 63 109 209.139.204.164 srhst19.yahoo.com
```

It appears they resolve it the same way, so if DNS is poisoned somewhere, it is probably at the root server level???

>>> "Johannes Ullrich" 12/10/02 10:28 AM >>>

hm. srhst19.yahoo.com does no longer resolve. So there are two options:
- they figured out something was bad and took the machine off the net. (funny that they even take the DNS entry out, but maybe thats part of their procedure/script)
- someone poisoned your DNS server :-/ (whats the IP address?)
- they provide only reverse, but no 'regular' DNS for that IP/hostname (odd, but things are not always RFC conform)

On Tue, 10 Dec 2002 10:08:12 -0700
"Al Maslowski-Yerges" wrote:

> Are any of you seeing this traffic? At first glance it looks pretty
> targeted. Anyone got a "good" contact for Yahoo?
>

```
> 08:10:16.719687 srhst19.yahoo.com.0 > my.classc.net.160.3128: S
> 318448:318448(0) win 512 (DF) 08:10:16.750399 srhst19.yahoo.com.0 >
> my.classc.net.160.80: S 318449:318449(0) win 512 (DF) 08:10:16.780863
> srhst19.yahoo.com0 > my.classc.net.160.8080: S 318450:318450(0) win 512
> (DF) 08:10:16.812434 srhst19.yahoo.com.0 > my.classc.net.160.1080: S
> 318451:318451(0) win 512 (DF) 08:12:22.255619 srhst19.yahoo.com.0 >
> my.classc.net.125.3128: S 319004:319004(0) win 512 (DF) 08:12:22.286177
> srhst19.yahoo.com.0 > my.classc.net.125.80: S 319005:319005(0) win 512
> (DF) 08:12:22.318284 srhst19.yahoo.com.0 > my.classc.net.125.8080: S
> 319006:319006(0) win 512 (DF) 08:12:22.350238 srhst19.yahoo.com.0 >
> my.classc.net.125.1080: S 319007:319007(0) win 512 (DF) 08:32:29.056479
> srhst19.yahoo.com.0 > my.classc.net.150.3128: S 322776:322776(0) win 512
> (DF) 08:32:29.092935 srhst19.yahoo.com.0 > my.classc.net.150.80: S
> 322777:322777(0) win 512 (DF) 08:32:29.119427 srhst19.yahoo.com.0 >
> my.classc.net.150.8080: S 322778:322778(0) win 512 (DF) 08:32:29.183939
```

```
> srhst19.yahoo.com.0 > my.classc.net.150.1080: S 322779:322779(0) win 512
> (DF)
>
>
>


--
---------------------------------------------------------------------
        Collaborative Intrusion Detection
                        join http://www.dshield.org
```

Incident details at mynetwatchman.com:
http://www.mynetwatchman.com/LID.asp?IID=15505093
http://www.mynetwatchman.com/LID.asp?IID=15305708

7. Evidence of Active Targeting:

As discussed above, I would say this is most likely active targeting. Again, it doesn't have a truly random nature and it is relatively slow executing. It also doesn't scan a whole range of addresses at one time.

8. Severity:

The severity will be calculated using the following formula on a scale from 1 to 5, where 1 is lowest and 5 highest:

severity = (criticality + lethality) - (system countermeasures + network countermeasures)

Criticality = 5 - The target addresses are mission critical systems for a variety of purposes. Some are e-business platforms or proxy servers.

Lethality =3 - Unknown exactly, but this activity could be the precursor to additional attacks if the attacker was able to get the information they were looking for.

System Countermeasures = 3 - These servers are all well patched and maintained, but with one of them being a public web server there are openings and potentially exploitable holes. None of the systems in question have host based IDS or integrity checkers.

Network Countermeasures = 4 - These systems are behind a firewall that allows very

limited access to them with the necessary exception of the web server. A NIDS system is also in place with a sensor both on the Internet segment and the internal segment

(5 + 3) - (3 + 4) = 1

9. Defense Recommendations:

a. Double-check the patch level and possible CVE exposures of all publicly accessible servers.

b. Consider "black-holing" the netblock ranges that the attacks came from.

c. Review firewall policy to be sure it effectively filters on ingress AND egress. Consider more active use of a proxy style firewall.

d. Install a host-based intrusion detection system and a host-based integrity checker on the web server to help guard against and warn of potential compromise.

e. Conduct periodic intrusion tests to identify weaknesses in the defensive posture.

10. Multiple Choice Question:

When traffic appears at your perimeter that seems to resolve to an address like mail.xyz.com in your logs but you can't get nslookup to resolve mail.xyz.com to an IP address, the best reason for this from the list below is _____.

a) DNS is being blocked at your perimeter firewall.

b) the reverse DNS entry for that netblock doesn't have a corresponding forward DNS entry.

c) the Internet root DNS servers have been hacked.

d) your DNS cache has been poisoned.

Answer: b) the reverse DNS entry for that netblock doesn't have a corresponding forward DNS entry.

As discussed above, it looks like this attacker purposely set up reverse DNS entries that seem to point to the yahoo.com domain but the host was not resolvable in forward DNS because it didn't really exist in yahoo.com's domain.

PART 3 - "ANALYZE THIS"

Executive Summary

The following analysis of 5 days of data from a "University environment" attempts to group the data into useable formats and extract patterns that point to specific vulnerabilities and/or exploits. Recommendations will also be included. Beside the raw data analysis and recommendations for defending against the attacks that were found, other observations and recommendations for improving the IDS system deployed or improving the overall security posture of the organization will be given.

A couple of general observations come immediately to light from my analysis. First, it appears that the Snort IDS system that is in place contains a very outdated set of rules and probably also is an old version of the software. Many improvements to the software including pre and post processors have been made over the past couple of years. Also, improvements in the signature set are constantly made. Therefore I recommend that this software be updated and monitored on a regular (monthly) basis to reduce the number of false positives that an analyst must wade through and more effectively protect the environment. Secondly, it appears that peer-to-peer file sharing programs are allowed relatively free reign on the network. This is a dangerous position, but may be necessary in a University environment. If this traffic must be allowed on the network, then I strongly suggest tuning the snort alerts to ignore most of this traffic so that other malicious traffic is not missed simply because of the volume of alerts that must be processed.

Files Analyzed

Alert Files Analyzed	Scan Files Analyzed	OOS Files Analyzed
Alert.021213	Scans.021213	OOS_Report_2002_12_14_28570.txt
Alert.021214	Scans.021214	OOS_Report_2002_12_15_884.txt
Alert.021215	Scans.021215	OOS_Report_2002_12_16_11235.txt
Alert.021216	Scans.021216	OOS_Report_2002_12_17_17468.txt
Alert.021217	Scans.021217	OOS_Report_2002_12_18_22751.txt

All of these files appear to be consistent as well as sequential so I concatenated each file type into one large file of that type for overall analysis. The OOS files each had actual data from the previous day so I used the sequence of logs starting one day after the logs used for alerts and scans.

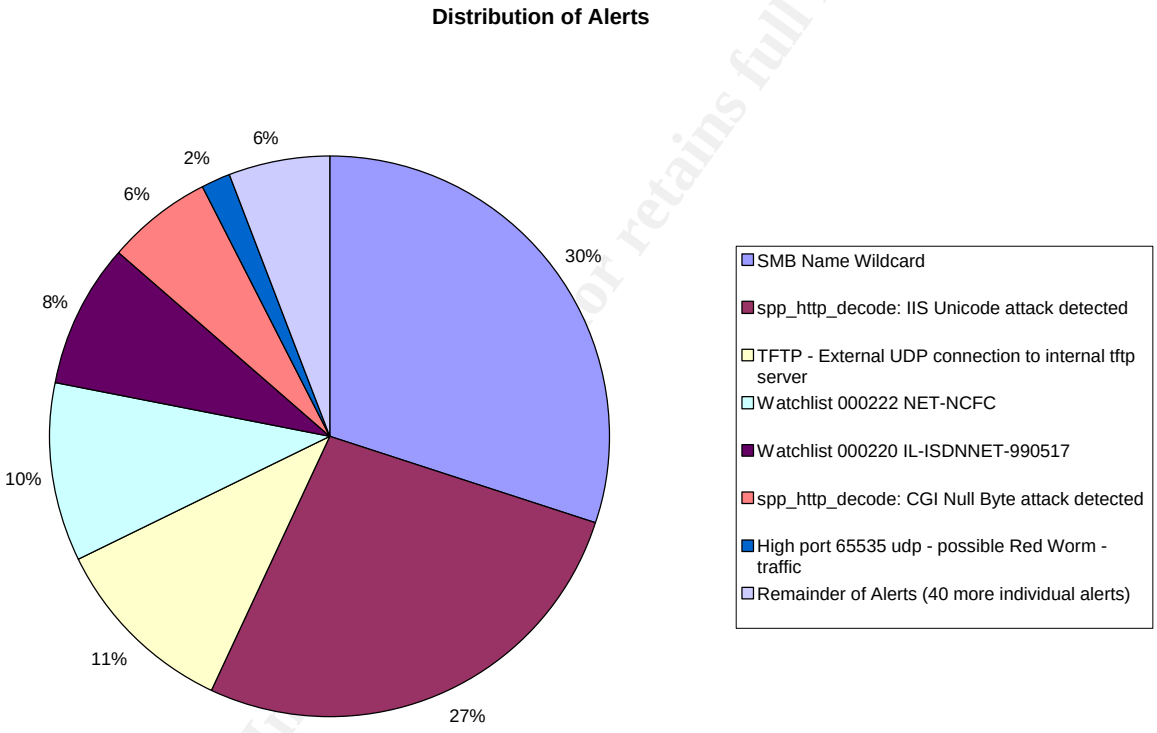
Analysis

For the analysis of the data I chose to start by not ignoring any data for analysis. I felt I might miss something if I pared down the data too quickly. I treat each of the alerts, scans, and OOS categories starting from the greatest frequency to least frequent assigning them a relative severity or just "noise" value from the data observed. After

analysis of the largest frequency items, I concentrate on those that might represent the greatest danger to the organization. All items are not exhaustively analyzed simply due to the volume of data. I also make recommendations for each category or sub-category.

Placement of the Sensor: From what I could tell, I believe the IDS sensor that this data came from was probably placed between the exterior perimeter router and the internal network, probably before any firewall if one exists. The network also seems to have "screened subnet" of MY.NET.1.x that is also in the path of this IDS system. My analyses that follow are based on these assumptions.

Summary of Alerts



Alert Summary Data

Alert Name	Number of Alerts
SMB Name Wildcard	47545
spp_http_decode: IIS Unicode attack detected	42440
TFTP - External UDP connection to internal tftp server	17392
Watchlist 000222 NET-NCFC	16329
Watchlist 000220 IL-ISDNNET-990517	12896
spp_http_decode: CGI Null Byte attack detected	9927
High port 65535 udp - possible Red Worm - traffic	2685

IDS552/web-iis_IIS ISAPI Overflow ida nosize	2252
Possible trojan server activity	1266
Queso fingerprint	1131
Incomplete Packet Fragments Discarded	952
IRC evil - running XDCC	832
EXPLOIT x86 NOOP	640
SUNRPC highport access!	548
High port 65535 tcp - possible Red Worm - traffic	255
Port 55850 tcp - Possible myserver activity - ref. 010313-1	243
SMB C access	174
Null scan!	155
External RPC call	89
NMAP TCP ping!	76
Port 55850 udp - Possible myserver activity - ref. 010313-1	65
EXPLOIT x86 setuid 0	58
FTP DoS ftpd globbing	36
EXPLOIT x86 setgid 0	35
TFTP - Internal UDP connection to external tftp server	34
EXPLOIT x86 stealth noop	33
RFB - Possible WinVNC – 010708-1	30
TCP SRC and DST outside network	27
MY.NET.30.4 activity	22
DDOS mstream client to handler	16
Tiny Fragments - Possible Hostile Activity	14
EXPLOIT NTPDX buffer overflow	11
ICMP SRC and DST outside network	9
Attempted Sun RPC high port access	9
Back Orifice	8
connect to 515 from inside	6
External FTP to HelpDesk MY.NET.70.50	5
TFTP - Internal TCP connection to external tftp server	4
Bugbear@MM virus in SMTP	2
MY.NET.30.3 activity	1
External FTP to HelpDesk MY.NET.83.197	1
Fragmentation Overflow Attack	1
Probable NMAP fingerprint attempt	1
NIMDA - Attempt to execute cmd from campus host	1
SNMP public access	1
EXPLOIT identd overflow	1
SYN-FIN scan!	1
HelpDesk MY.NET.83.197 to External FTP	1
	158260

Most Frequent Alerts (a lot of noise, some important detects)

I split the alerts in to two sections to handle them. First, this section deals with many of the alerts that generated the most traffic. I felt it was important to address these in order to emphasize/document the need to tune the IDS rules so that the most important alerts are easily seen and the excessive noise that just distracts is ignored.

SMB Name Wildcard

Alert Name	# seen (%)	Classif.	Arachnids/CVE
SMB Name Wildcard	47545 Alerts (30%)	Noise (2)	IDS177/CAN-1999-0621

Discussion: Windows and Samba clients typically use this type of NetBIOS traffic to find hosts even if they are involved in other communications with the host if they can't resolve the name with DNS (i.e. IIS servers trying to resolve the names of the hosts connecting to them). The ArachNIDS listing IDS177 suggests that this is normal traffic with internal hosts but may be a scan if the sources are external hosts. Most of the traffic is from external hosts and much of it is not from a source port of 137. (Therefore probably not Windows machines)

Some of this traffic is certainly active scanning from Internet hosts such as with "Legion" (http://www.pdaconsulting.com/Cracker%20Tools/legionv21.zip). I base this on both the packet data in which I see single Internet hosts scanning through whole ranges of addresses and the scan logs further analyzed in the section on scan alerts below.

```
ALERT   Dec   13 21:10.6 SMB Name Wildcard    66.136.178.22    61239 MY.NET.135.112      137
ALERT   Dec   13 21:10.8 SMB Name Wildcard    66.136.178.22    61242 MY.NET.135.113      137
ALERT   Dec   13 21:11.1 SMB Name Wildcard    66.136.178.22    61248 MY.NET.135.115      137
ALERT   Dec   13 21:11.8 SMB Name Wildcard    66.136.178.22    61263 MY.NET.135.120      137
ALERT   Dec   13 21:12.6 SMB Name Wildcard    66.136.178.22    61278 MY.NET.135.125      137
ALERT   Dec   13 21:12.7 SMB Name Wildcard    66.136.178.22    61281 MY.NET.135.126      137
ALERT   Dec   13 21:12.9 SMB Name Wildcard    66.136.178.22    61284 MY.NET.135.127      137
ALERT   Dec   13 21:13.8 SMB Name Wildcard    66.136.178.22    61302 MY.NET.135.133      137
```

Many of the source addresses are from private RFC 1918 addresses. At first this might seem like some mis-routing or a "leaking" NAT firewall, but this appears to be somewhat normal behavior for Windows machines with multiple interfaces defined. (http://cert.uni-stuttgart.de/archive/incidents/2001/05/msg00041.html)

Since this activity is so common both on internal networks and on the Internet, it is still considered "noise" except when it correlates with other malicious activity.

Correlation: This first one is from jsage@finchhaven.com http://www.finchhaven.com/pages/incidents/030102_udp_137.html and it refers to several other links that are useful in understanding this alert.

The following is from SANS and Stephen Northcutt:
http://www.sans.org/y2k/051300.htm

Another good discussion of NetBIOS and these alerts can be found on Robert Graham's site: http://www.robertgraham.com/pubs/firewall-seen.html#netbios

Recommendations: I recommend that all traffic destined for UDP 137 (actually 135-139 and 445 TCP and UDP) inbound and outbound be dropped at the perimeter router and internal routers before it ever reaches the IDS sensor or the firewall. Routers are particularly well suited to drop this kind of traffic and blocking it there will take a load off of the firewall, the IDS, and the analyst trying to keep the network secure. Today on the Internet, this traffic is ubiquitous and it really does no good in almost all circumstances to log it and alert on it with the IDS. If logging of this traffic is ever necessary then the router should be set to send syslog output to a centralized syslog server for later analysis.

IIS related Alerts

Alert Name	# seen (%)	Classif.	Arachnids/CVE
IIS Unicode attack	42440 Alerts (27%)	Noise (3)	IDS432/CAN-2000-0884
IIS CGI Null Byte attack	9927 Alerts (6%)	noise (3)	
IIS isapi ida nosize	2252 Alerts (1 %)	Noise (3)	IDS552/CVE-2001-0500

Discussion: The first 2 detects in this set, while useful, often give "false positives." The Snort FAQ even has a section discussing this and explaining how to disable the detects (http://www.snort.org/docs/faq.html#4.17). The http_decode preprocessor is looking for "…for Unicode-encoded "\" "/" and "."characters" (Tod Beardsley - http://www.giac.org/practical/Tod_Beardsley_GCIA.doc) Many dynamically created URL's and search engines will also trigger this alert. For instance, looking through the destination addresses in this scan data when the source addresses were internal, I saw that a large number of the address ranges were owned by AOL or Netscape. Certainly these are probably support servers for AOL or Netscape services such as mail or search engines.

However, as with other alerts that generate so much traffic, there are probably some events buried in the first two alerts that indicate malicious activity. In this case probably code red or Nimda. I did not run across any set of data that looked like either a scripted or manual scan for Unicode vulnerabilities.

The 3rd alert in the set rarely gives a false positive. When I looked at the alerts for these, all of them came from external hosts attempting to connect to internal hosts. These are almost certainly a result of a code red variant still trying to infect new servers. From the scans I can't tell if there are also internal servers infected with code red. Further investigation of the Snort rule shows that by default it only looks for inbound traffic.

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS552/web-iis_IIS ISAPI
Overflow ida"; dsize: >239; flags: A+; uricontent: ".ida?"; classtype:
system-or-info-attempt; reference: arachnids,552;)
```

In order to find the truly malicious activity some work needs to be done on the defenses here. See the recommendations below for more details.

Correlation: Tod Beardsley's practical contains two sections dealing with Unicode alerts like this and the Nimda and Code Red worms that are likely causing much of it. http://www.giac.org/practical/Tod_Beardsley_GCIA.doc

The snort FAQ also has insight here and I have used it extensively.
http://www.snort.org/docs/faq.html#4.17

Here is another link for correlation with actual tools to do directed attacks with unicode
http://www.geocrawler.com/mail/msg.php3?msg_id=6521002&list=4890

Recommendations: First of all, there should be ingress and egress filtering aimed at specifically stopping the code red and the Nimda variants. Also, it might be a good idea to add the Berkley Packet Filter code to the Snort systems suggested in the Snort FAQ which will stop alerts on Unicode and CGI-Null's when the packets originate from inside the network. This should help stop many of the false alerts, but I would suggest implementing it in this environment only if there are other means of detecting internally infected Nimda or code red machines such as the egress filtering on internal firewalls and routers mentioned above. Otherwise in a University environment you wouldn't know when you were in danger from the inside.

After further protecting the interior network from code red and nimda or similar activity, it might be useful to "turn around" the Snort rule for ISAPI so that the analyst would be alerted of internally infected code red computers. (This may, in fact, be similar to what was done to create the "NIMDA - Attempt to execute cmd from campus host" rule that triggered one alert in this sample. Without seeing the active rule-set I can't be sure, but the rule probably looks for the cmd.exe command in the payload. This alert will not be analyzed further.)

TFTP - External UDP connection to internal tftp server/Internal TCP/UDP connection to external tftp server

Alert Name	# seen (%)	Classif.	Arachnids/CVE
TFTP - External UDP connection to internal tftp server	17392 Alerts (11%)	Noise (3)	CAN-1999-0616
TCP/UDP connection to external tftp server	38 Alerts (<1%)	High (5)	NA/Local Rule/Old Rule

Discussion: The first alert just looks like a misconfigured/broken router or switch trying to find a TFTP server to load its code from. All except 2 connections are between MY.NET.111.219 (Src.) and 192.168.0.253 (Dst.). I think the MY.NET.111.219 server is indeed a TFTP server and it is accepting the connection from the router and trying to respond with data but it doesn't have a route to get to this address. Often switches or other network equipment that get their configurations from a TFTP server will default to some private IP address if they lose their configuration or can't boot from an internal

configuration. This internal address is apparently not routed correctly on the University network.

The second set of Alerts looks a little more troubling. I can definitely see active connections. The table below shows TCP connections but there are pairs of UDP connections as well:

ALERT	Dec	15	18:24:58.55	TFTP - Internal TCP connection to external tftp server	66.93.54.212	69	MY.NET.83.116	59322
ALERT	Dec	15	18:23:47.35	TFTP - Internal TCP connection to external tftp server	66.93.54.212	69	MY.NET.83.116	43336
ALERT	Dec	15	18:23:47.24	TFTP - Internal TCP connection to external tftp server	MY.NET.83.116	43336	66.93.54.212	69
ALERT	Dec	15	18:24:58.41	TFTP - Internal TCP connection to external tftp server	MY.NET.83.116	59322	66.93.54.212	69

The University is apparently allowing inbound and outbound TFTP traffic. This is dangerous because there are multiple vulnerabilities found in various TFTP servers that will allow the attacker to gain access to password and other files. (http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=TFTP) If TFTP servers are enabled on network equipment also then it is a simple matter to download the configuration files and/or upload changes to them.

Below is the contact information for 3 of the hosts seen in these TFTP alerts since TFTP downloads are often a way of installing Trojan code. We might want to know who some of the addresses belong to and possibly contact them to see if they are seeing the same traffic or have some explanation.

66.93.54.255	whois whois.arin.net 66.93.54.255: Speakeasy Network SPEAKEASY-5 (NET-66-92-0-0-1) 66.92.0.0 - 66.93.255.255 BLT BRIDGED CIRCUITS SPEK-BLT-BR-1 (NET-66-93-54-1-1) 66.93.54.1 - 66.93.54.255 Whois Server Version 1.3 Domain names in the .com, .net, and .org domains can now be registered with many different competing registrars. Go to http://www.internic.net for detailed information. Domain Name: SPEAKEASY.NET Registrar: NETWORK SOLUTIONS, INC. Whois Server: whois.networksolutions.com Referral URL: http://www.networksolutions.com Name Server: NS2.SPEAKEASY.NET Name Server: NS1.SPEAKEASY.NET Updated Date: 05-nov-2001 Registrant: Speakeasy, Inc. (SPEAKEASY3-DOM) 2304 2nd Ave Seattle

	WA,98121 US Domain Name: SPEAKEASY.NET Administrative Contact: Executive, Speakeasy (SEN144) h Speakeasy Network 2222 - 2nd Ave Suite 222 Seattle, WA 98121 +1 206 728 9770 (FAX) +1 206 728 1500 Technical Contact: Hostmaster, Speakeasy (HS1672-ORG) 2222 - 2nd Ave Suite 222 Seattle , WA 98121 US +1 206 728 9770 Fax- - - - - - +1 206 728 1500 Record expires on 06-Jun-2008. Record created on 05-Jun-1995. Database last updated on 3-Jan-2003 01:47:51 EST. Domain servers in listed order: NS1.SPEAKEASY.NET 216.254.0.9 NS2.SPEAKEASY.NET 216.231.41.19
134.126.10.162	whois whois.arin.net 134.126.10.162: OrgName: James Madison University OrgID: JMU NetRange: 134.126.0.0 - 134.126.255.255 CIDR: 134.126.0.0/16 NetName: JMU NetHandle: NET-134-126-0-0-1 Parent: NET-134-0-0-0-0 NetType: Direct Assignment NameServer: DOC.JMU.EDU NameServer: FALCON.JMU.EDU Comment: RegDate: 1989-06-02 Updated: 1995-11-10 TechHandle: CM50-ARIN TechName: Minnick, Connie TechPhone: +1-703-568-6711 TechEmail: minniccr@jmu.edu
209.249.64.202	whois whois.arin.net 209.249.64.202: OrgName: Abovenet Communications, Inc OrgID: ABVE

	NetRange: 209.249.0.0 - 209.249.255.255 CIDR: 209.249.0.0/16 NetName: ABOVENET-4 NetHandle: NET-209-249-0-0-1 Parent: NET-209-0-0-0-0 NetType: Direct Allocation NameServer: NS.ABOVE.NET NameServer: NS3.ABOVE.NET Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE RegDate: 1998-06-15 Updated: 2001-04-27 TechHandle: NOC41-ORG-ARIN TechName: Metromedia Fiber Networks AboveNet TechPhone: +1-877-479-7378 TechEmail: noc@mfnx.net OrgTechHandle: NOC41-ORG-ARIN OrgTechName: Metromedia Fiber Networks AboveNet OrgTechPhone: +1-877-479-7378 OrgTechEmail: noc@mfnx.net

Correlation: Joe Ellis also reported on this activity in his recent practical http://www.giac.org/practical/Joe_Ellis_GCIA.doc

Montgomery Toren also referes to TFTP traffic in his practical describing the traffic he saw as malicious http://www.giac.org/practical/Montgomery_Toren_GCIA.doc

Recommendations: Simply fixing the broken router or switch in this case will fix the 1st problem and greatly reduce the number of alerts here. A product like ipchains or other host based firewall could also help. It is obvious here that the TFTP server is responding to this host even though the host is probably unauthorized based on the fact that the TFTP server doesn't seem to be able to route traffic to it correctly. Some sort of access control should be implemented if it is necessary to keep the TFTP server operational for network support. Otherwise any internal or external host that can find the TFTP server could potentially compromise it or pull off important router and switch configuration information from it. There are also a number of vulnerabilities in TFTP servers that could allow arbitrary access to other files on the server such as /etc/passwd. So the best course of action is not to use a TFTP server if that is possible.

For the second set of alerts in this section, the best course of action is to block access to TFTP at both ingress and egress points. Also, each of the internal hosts found in these alerts should be closely examined for trojans, etc… and probably reformatted once protections are in place.

Alerts related to P2P File Sharing

Alert Name	# seen (%)	Classif.	Arachnids/CVE
Watchlist 000222 NET-NCFC	16329 (10%)	Noise (2)	NA/Local Rule/Old Rule
Watchlist 000220 IL-ISDNNET-990517	12896 (8%)	Noise (2)	NA/Local Rule/Old Rule
High port 65535 udp/tcp - possible Red Worm - traffic	2940 (2%)	Noise (2)	NA/Local Rule/Old Rule
Possible trojan server activity	1266 (1%)	Noise (3)	IDS279/CAN-1999-0660
Queso fingerprint	1131 (1%)	Noise (3)	IDS29/CAN-1999-0454
IRC evil - running XDCC	832 (<1%)	High (3)	NA/Local Rule/Old Rule

All of these alerts appear to be somewhat related to P2P file sharing software such as KaZaA, Morpheus, WinMX, and XDCC. These can certainly be dangerous programs, but they are generating way too many alerts in the IDS system. It drowns out other important traffic that may not be seen otherwise. Together these 6 alerts almost equal the number of alerts generated by the NetBIOS alert described above. If local policy allows this traffic, then the traffic should be ignored. If the policy doesn't allow it, then the traffic should be blocked before it ever makes it to the perimeter IDS sensor.

Discussion: I will briefly discuss each alert and why I think they are/are not simply noise that should be tuned out when looking for real IDS events of interest.

Watchlist 000222 is apparently a netblock that the personnel running this Snort instance have had trouble with in the past and they are watching activity from that netblock closely. The vast majority of the traffic was between a host in the watchlist range (159.226.221.127) and one internal host (MY.NET.153.178) on port 2320. The registered use of port 2320 is "Siebel Name Service" (http://www.iana.org/assignments/port-numbers) but I also found a number of references to ICQ using this port. (http://www.martijnjongen.com/portnumbers.htm, http://www.users.qwest.net/~rlutton/ADSL/NAT_Settings.html) This is the more likely use of the port in this situation. It might also be another service using that port. Much of the other traffic from this netblock was either web traffic or KaZaA traffic.

Watchlist 000220 is another netblock the University is watching. This traffic was a little more spread out among hosts and ports. I looked closely at a few of the top destination ports to try to identify what was going on. Some of them I had to identify by the other side of the conversation.

```
__/ EOIs by Destination Port (External Only) \_____________________________
|                                                                          |
| 3250     2917        Elvin                                               |
| 1220     3011        trusted web                                         |
| 954      4242        80                                                  |
| 602      4662        EDonkey                                             |
| 535      3292        80                                                  |
| 457      1083        80                                                  |
| 447      3551        80                                                  |
| 305      25          smtp                                                |
| 145      3898         other side of Elvin                                |
```

```
| 133       1214          KaZAa                                          |
| 100       80                                                          |
| 89        1226        Exchange server                                 |
| 87        1068        80                                              |
```

Elvin is a messaging protocol used by some chat and gaming programs. Edonkey and KaZaA are also near the top with browser traffic and SMTP rounding it out. It also looks as though there might be some MS exchange servers installed on both the local network and a remote host.

The "Possible Red Worm – traffic" alerts appear to be based on rules that solely look for the signature port of the worm. The Adore or Red worm listens on this port on infected machines. However, I have my doubts about whether or not this is actually Adore Traffic. Nearly all of the traffic is between ports 65535 and port 6257 for the UDP alert set. UDP 6257 is the default UDP port for WinMX which is a P2P file sharing program primarily used for exchanging music files. For the TCP alert set many of the alerts show traffic between 65535 and the KaZaA port 1214. Many of the instruction sets for these programs propose getting around firewalls by using a port in some high range such as WinMX documentation which suggests ports 5001 – 65535 (http://winmx.2038.net/winmx/fr-blocked.html). I suggest that we see many more packets on 65535 than would normally be expected with these programs because many users will automatically pick this high number in the range when they are configuring the program settings.

The Possible trojan server activity alerts also look like mostly false positives for the Sub Seven trojan. Almost all of this traffic is either to or from port 1214 which is the port for KaZaA (http://www.incidents.org/archives/intrusions/thrd118.html). Other traffic is to or from 4662 which is Edonkey's default port. Certainly there are other alerts in this set that are real but the large number of alerts here make it hard to sort them out of the noise. I expect that the following couple of lines are real scans for a Sub Seven host. They don't appear to have been answered though.

```
ALERT  Dec  17  51:26.6  Possible trojan server activity  211.155.246.218  2383  MY.NET.134.108  27374
ALERT  Dec  17  51:44.6  Possible trojan server activity  211.155.246.218  2383  MY.NET.134.108  27374
```

The Queso fingerprint set of alerts again look like they are probably mostly false positives. From what I have seen concerning the rest of the Snort rule-set, it seems quite likely that this rule is also an older version of the rule. This has become subject to false positives since the advent of ECN (Explicit Congestion Notification) in IP. The rule was just looking for these specific reserved IP flags being set as they typically are in Queso syn packets. A good resource explaining this is at http://www.sans.org/y2k/ecn.htm. More recently the rule has been adjusted to also check for a high TTL which is also a signature of Queso so it results in many fewer false positives. I suggest that this signature be updated on the snort installation.

IRC evil running XDCC: Unlike the other alerts in this category, this one is immediately cause for concern. There are probably very few, if any, false positives here. XDCC is an "IRC bot" that has many capabilities. Its main purpose is file sharing, but it acts like a

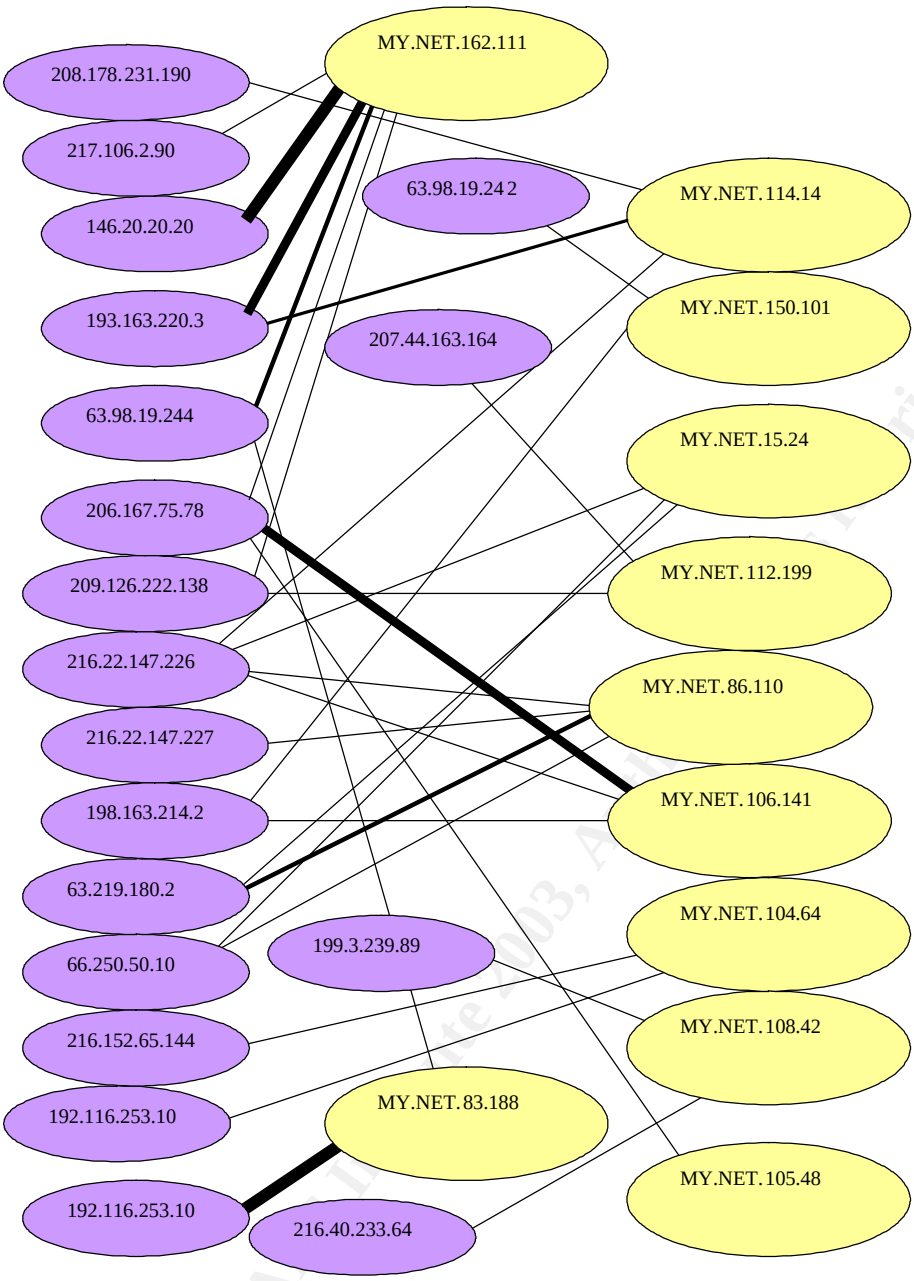
worm by infecting any windows machines with open shares. Dave Dittrich gives a very detailed analysis here: http://staff.washington.edu/dittrich/talks/core02/xdcc-analysis.txt. Another aspect of this XDCC bot is that it has the ability to conduct DDOS attacks. Here is another posting that lists known infected computers and suggests they could easily be used to run a DDOS attack against a major network. (http://www.theorygroup.com/Archive/Unisog/2002/msg00618.html) It appears that there are 11 hosts internally that are infected with this IRC bot. See the following table for the details on the hosts.

```
_____________________________________
 __/ EOIs by Source IP (Internal Only) \_____________________________
|                                                                     |
| 262      MY.NET.162.111                                             |
| 115      MY.NET.86.110                                              |
| 108      MY.NET.83.188                                              |
| 98       MY.NET.106.141                                             |
| 66       MY.NET.105.48                                              |
| 63       MY.NET.150.101                                             |
| 46       MY.NET.114.14                                              |
| 34       MY.NET.104.64                                              |
| 19       MY.NET.15.24                                               |
| 15       MY.NET.108.42                                              |
| 6        MY.NET.112.199                                             |
|                                                                     |
| Total Uniques:           11                 Total EOIs:      832 |
```

In Dave Dittrich's paper, he discusses one step in the infection process as using TFTP to download IRC bot configuration files. This may correspond to some of the TFTP traffic also discussed earlier. Analysis of relationships between alerts is done below at LINK. It also may correspond to some of the packet fragmentation activity seen in a later alert.

Link Graph

I was interested to see if there were any relationships between the clients and servers for this alert so I constructed a Link Graph of this alert traffic. From this graph, it is apparent that the activity is definitely not just random use of the port 6666 or 6667. There seems to be definite client/server activity going on here or possibly a P2P type network. Several of the public IP addresses (in purple) have connections with multiple internal hosts, and the internal hosts (in tan) often connect to multiple public hosts but only a small number of them. This limited set of hosts indicates a coordinated, non-distributed application. I can't see any internal connections between the hosts because that would not have been logged at the IDS.



Correlation: Many other GCIA papers show similar traffic. Here are some links to a few of them. Other correlations are included inline above where appropriate.

John Garis also discusses the “Watchlist” alerts (http://www.giac.org/practical/John_Garris_GCIA.doc)

James Hoover writes about the Adore or Red Worm (http://www.giac.org/practical/James_Hoover_GCIA.doc)

Joe Ellis includes a section on the “possible Trojan server” alert in his paper

(http://www.giac.org/practical/Joe_Ellis_GCIA.doc)

Recommendations: The rule set and the mindset appear to both be out of date. The rules can and should be tuned so that they reflect policy. If the policy is to allow TFTP traffic and P2P traffic, then the snort rules should be constructed to ignore this traffic. If the policy is to NOT allow this traffic, then it should be filtered by firewalls and routers so that it never hits the IDS system. The rules also appear to be a little old. Keeping the rules and the Snort code up to date will also help to reduce the number of false positives. Again, having this many alerts makes it nearly impossible to react quickly to real threats or real breaches of policy.

For the XDCC alert, I suggest immediate action be taken. The machines that have been identified must be checked closely and cleaned carefully to prevent their use in DDOS attacks or other malicious activity.

Generally speaking, again I strongly urge that ingress and egress filtering be employed to the fullest extent allowed by University policy.

Packet Fragment Alerts

Alert Name	# seen (%)	Classif.	Arachnids/CVE
Incomplete Packet Fragments Discarded	952 (<1%)	Noise (2)	NA/Local Rule/Old Rule
Tiny Fragments - Possible Hostile Activity	14 (<1%)	Noise (2)	NA/Local Rule/Old Rule

Discussion: These alerts are potentially interesting because sometimes fragmentation is generated simply to bypass firewalls and fool IDS systems that don't do stream reasembly before making decisions on whether to pass traffic or generate an alert. Fragmentation can also be used to overwrite portions of the packet upon reassembly on the receiving side so that traffic not otherwise allowed through the defenses can be forced through. The packets that get past the firewall can be used for reconnaissance purposes or for direct attack.

The packets in the first alert group do not seem to represent a scan. They nearly uniformly correspond to specific conversations between 2 hosts consisting of multiple packets. They could, however, be part of an attack or possibly an MTU problem or very small link somewhere in the path between the hosts. Most of the top 10 hosts in this alert are from somewhere in China. This adds to the likelihood that there are small MTU links in the path. The only way to rule out malicious activity would be to do extensive analysis of the actual packet data.

Correlation: Michael Wilkinson discusses these packet types in his practical as well. (http://www.giac.org/practical/michael_wilkinson_gcia.doc)

Recommendations: If the packet fragments consistently come from the same source and it can be determined that they are not malicious, I suggest removing these addresses from consideration by this rule. This brings me to the next recommendation

which is to analyze some of the raw packet data further to determine the intent of the the packets and the reason for the fragmentation.

Less Frequent but significant alerts

This is where many of the important alerts are. After the noise alerts are taken out, it is much easier to concentrate on the important activity that is left. This section addresses some of the highest severity alerts found but not every alert that was found over the 5 day period. I felt it was more important to concentrate on some of the most dangerous activity than to analyze every alert.

Possible myserver activity

Alert Name	Total # seen	Classif.	Arachnids/CVE
Port 55850 tcp/udp - Possible myserver activity	308	High (5)	CAN-2000-0138

Discussion: Myserver is a DDOS tool that listens on UDP port 55580 for commands from the master. It can be installed as part of a root kit or in other ways. I found no evidence that the tool listens on TCP so I'm not sure why there is both a TCP and a UDP version of the rule on this server. The rule appears to be somewhat susceptible to false positives because it seems to only fire alerts when traffic is from or to this port. This can happen quite often on a busy network throughout any given week. As a result, it appears that there are a number of false positives. I would completely rule out the TCP traffic as it appears that it is just regular HTTP and SMTP traffic, but some of the UDP traffic does appear that it is likely "myserver" activity. After using grep to remove only the UDP activity and then remove the traceroute activity, I'm left with the following packets:

MY.NET.87.172,55850,10.0.1.1,192
MY.NET.188.24,55850,10.0.1.1,192
MY.NET.188.24,55850,10.0.1.1,192
63.250.205.20,4691,MY.NET.153.169,55850
MY.NET.188.24,55850,10.0.1.1,192
63.250.219.157,55850,MY.NET.152.162,19349
3.250.205.102,55850,MY.NET.110.57,10924
MY.NET.188.24,55850,10.0.1.1,192
210.115.150.103,6164,MY.NET.84.216,55850
MY.NET.188.24,55850,10.0.1.1,192
63.250.219.157,55850,MY.NET.152.186,25507

Next I take out the traffic destined for port 192 because it is probably a probe for a local wireless router (http://www.net.princeton.edu/software/osunms_probe/osunms_probe.8.html) Then I'm left with:

63.250.205.20,4691,MY.NET.153.169,55850
63.250.219.157,55850,MY.NET.152.162,19349
63.250.205.102,55850,MY.NET.110.57,10924
210.115.150.103,6164,MY.NET.84.216,55850
63.250.219.157,55850,MY.NET.152.186,25507

These packets don't correspond to any other known activity so it is likely that they are actual myserver packets.

Since this seems to be malicious activity, I have researched and included WHOIS information for these ranges in the table below

63.250.205.20, 219.157, 205.102	Final results obtained from whois.arin.net. Results: OrgName: Yahoo! Broadcast Services, Inc. OrgID: YAHO NetRange: 63.250.192.0 - 63.250.223.255 CIDR: 63.250.192.0/19 NetName: NETBLK2-YAHOOBS NetHandle: NET-63-250-192-0-1 Parent: NET-63-0-0-0-0 NetType: Direct Allocation NameServer: NS1.YAHOO.COM NameServer: NS2.YAHOO.COM NameServer: NS3.YAHOO.COM NameServer: NS4.YAHOO.COM NameServer: NS5.YAHOO.COM Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE RegDate: 1999-11-24 Updated: 2002-03-27 TechHandle: NA258-ARIN TechName: Netblock Admin, Netblock TechPhone: +1-408-349-7183 TechEmail: netblockadmin@yahoo-inc.com
210.115.150.103	IP Address : 210.115.128.0-210.115.159.255 Network Name : DONGA-NET Connect ISP Name : ISP-1 Connect Date : 19980916 Registration Date : 19980916 [Organization Information] Orgnization ID : ORG6695 Org Name : Midas Dong-A Ilbo State : SEOUL Address : 139 Sejong-ro chongro-ku Seoul Zip Code : 110-050 [Admin Contact Information] Name : Ki-Choon Ha Org Name : Midas Dong-A Ilbo State : SEOUL Address : 139, Sejong-ro, chongro-ku, Seoul, Korea

	Zip Code : 110-050 Phone : +82-2-721-7690 Fax : (02) 721-7676 E-Mail : matthias@mail.dongailbo.co.kr [Technical Contact Information] Name : Ki-Choon Ha Org Name : Midas Dong-A Ilbo State : SEOUL Address : 139, Sejong-ro, chongro-ku, Seoul, Korea Zip Code : 110-050 Phone : +82-2-721-7690 Fax : (02) 721-7676 E-Mail : matthias@mail.dongailbo.co.kr

Correlation: The first correlation is from the University of Massachusetts "Brian" (http://216.239.53.100/search?q=cache:n8VTuga3_A4C:www-net.cs.umass.edu/~brian/cs515-S02/515-incident.ppt+myserver+DDOS+umass&hl=en&ie=UTF-8) and it explains in detail how myserver works.

The next correlation is from SANS and shows more myserver activity (http://www.**sans**.org/y2k/082200.htm)

Jason Lam also writes about this alert in his practical: (http://www.giac.org/practical/Jason_Lam_GCIA.doc)

Recommendations: All of the internal hosts left after parsing down the logs should be carefully scrutinized for this DDOS tool and any rootkit that came along with it. Special care should be taken to use "known good" binaries (possibly on a CD) when searching for traces of the tool. "Replacement" OS binaries are known to accompany installations of this tool in order to hide its existence.

SMB C access

Alert Name	Total # seen	Classif.	Arachnids/CVE
SMB C access	174	High (5)	IDS339/CAN-1999-0621

Discussion: This alert fires when someone attempts to access the C$ on a Windows machine. This is the administrative share for the entire C:\ drive on the computer and should not be shared out or accessed. Unfortunately many versions of windows install by default with file sharing turned on, the root of all drives shared, and generally without a password assigned. These internal hosts almost certainly have their computers open to the world and almost as certainly have trojan's, viruses, and other

malicious software installed on them. The rule that generated this alert appears to be an accurate rule that rarely misfires, therefore action should be taken to secure and clean these target hosts. The following is a table generated by the summarize.pl script that shows the target hosts.

```
 __/ EOIs by Destination IP (Internal Only) \_____________________________
|                                                                         |
| 30       MY.NET.132.43                                                  |
| 22       MY.NET.190.102                                                 |
| 20       MY.NET.132.42                                                  |
| 20       MY.NET.190.100                                                 |
| 17       MY.NET.137.35                                                  |
| 16       MY.NET.137.46                                                  |
| 12       MY.NET.137.36                                                  |
| 9        MY.NET.137.34                                                  |
| 8        MY.NET.190.17                                                  |
| 6        MY.NET.190.26                                                  |
| 3        MY.NET.190.19                                                  |
| 3        MY.NET.190.34                                                  |
| 2        MY.NET.132.24                                                  |
| 2        MY.NET.190.38                                                  |
| 2        MY.NET.132.23                                                  |
| 1        MY.NET.132.25                                                  |
| 1        MY.NET.132.27                                                  |
|                                                                         |
| Total Uniques:            17                  Total EOIs:      174 |
 -------------------------------------------------------------------------
```

Correlation: The first correlation comes from some posted pages out of a Snort Snarf run: (http://www.geniussystems.net/stats/snortsnarf/sig/sigsid-533.html)

The next correlation comes from another GCIA paper by Hee So (http://www.giac.org/practical/Hee_So_GCIA.doc)

Recommendations: As stated above, these 17 machines should be secured and cleaned of any malware. Also, once again I make the plea for even minimal ingress and egress filtering to prevent this traffic. Educating users on the dangers of the Windows default sharing activity could also help in this environment.

FTP DoS ftpd globbing

Alert Name	Total # seen	Classif.	Arachnids/CVE
FTP DoS ftpd globbing	36	High (5)	IDS487/CAN-2001-0247

Discussion: This alert is caused when a user connected to an FTP server sends a specific set of characters such as a wildcard request for a very long directory path or a "~" or a "{" in the right manner. This doesn't normally happen in FTP sessions and is usually done to try to crash the FTP server and cause a Denial of Service situation or gain access to other local files. There are only two external hosts in this alert who are targeting one internal machine that I assume is probably an FTP server.

Correlation: The first correlation entry I'm adding is a CERT advisory (http://www.cert.org/advisories/CA-2001-07.html)

Here is another posting on incident.org: (http://www.incidents.org/diary/diary.php?id=95)

And here is another one from another GCIA paper by David R Williams (http://www.giac.org/practical/david_r_williams_GCIA.doc)

Recommendations: Patch any UNIX based ftp servers since this has been fixed long ago with patches for most distributions. Edit/alter the rule as necessary as discussed in the incidents.org posting above. Deploy ingress and egress filtering again.

EXPLOIT x86 stealth noop

Alert Name	Total # seen	Classif.	Arachnids/CVE
EXPLOIT x86 stealth noop	33	High (5)	ID291

Discussion: This signature is a pretty specific signature. It is possible that application code would use this string of "`eb 02 eb 02 eb 02`" that the rule is looking for, but it is very unlikely. This means that all of the events that caused the alert to fire are probably actual attempts at system compromises. From this data, we can't tell definitively if they were successful or not, but I believe they probably were. Looking at the alert data, there are no broad scans and very little repetition. This leads me to believe that the attacker was very focused on their target(s) and was able to successfully launch the attack with only one or two packets.

Correlation: Here is a link to Carlin Carpenter's very good GCIA paper listing this exploit (http://www.giac.org/practical/Carlin_Carpenter_GCIA.doc)

Edward Peck also includes this alert in his GCIA practical (http://www.giac.org/practical/Edward_Peck_GCIA.doc)

Recommendations: Secure and scan the 10 internal machines involved. Help/Educate users on staying up with the latest patch levels. Deploy ingress and egress filtering again. Here is a table listing the machines involved.

```
 __________________________________________
  __/ EOIs by Destination IP (External Only) \_____________________________
|                                                                           |
| 17       "MY.NET.83.247"                                                  |
| 6        "MY.NET.70.210"                                                  |
| 3        "MY.NET.70.101"                                                  |
| 1        "MY.NET.87.68"                                                   |
| 1        "MY.NET.153.203"                                                 |
| 1        "MY.NET.112.173"                                                 |
| 1        "MY.NET.182.68"                                                  |
| 1        "MY.NET.84.182"                                                  |
| 1        "MY.NET.153.151"                                                 |
| 1        "MY.NET.87.161"                                                  |
```

```
|                                                                    |
| Total Uniques:            10                  Total EOIs:       33 |
 --------------------------------------------------------------------
```

EXPLOIT NTPDX buffer overflow

Alert Name	Total # seen	Classif.	Arachnids/CVE
EXPLOIT NTPDX buffer overflow	11	High (5)	ID492

Discussion: This indicates buffer overflow attempts against the NTP daemon. It is done by sending data in the packet above 128 bytes. If the buffer overflow is successful, root access can be gained. Eight Internal hosts were targeted.

```
__/ EOIs by Destination IP (External Only) \_____________________
|                                                                  |
| 2        "MY.NET.110.57"                                         |
| 2        "MY.NET.110.139"                                        |
| 2        "MY.NET.168.181"                                        |
| 1        "MY.NET.53.40"                                          |
| 1        "MY.NET.109.25"                                         |
| 1        "MY.NET.152.162"                                        |
| 1        "MY.NET.53.50"                                          |
| 1        "MY.NET.112.193"                                        |
|                                                                  |
| Total Uniques:            8                  Total EOIs:       11 |
 ------------------------------------------------------------------
```

Again, each host was only hit one or two times. This indicates that either these are false positives (unlikely according to the information at whitehats.com http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids492) or that they were successful in only one or two packets.

Correlation: First, here is a post about the vulnerability on Linuxsecurity.com (http://www.linuxsecurity.com/advisories/netbsd_advisory-1255.html)

Here is a post on securityfocus.com that confirms that Cisco routers are also vulnerable. (http://online.securityfocus.com/archive/1/175701)

Recommendations: Make sure that all NTP servers and routers are patched with the latest code. Unless it is necessary, NTP should only be allowed in to specific public NTP servers and the rest should be blocked. If NTP servers for the public exist, then they should be placed in a DMZ area so that if they were compromised, they wouldn't endanger the rest of the network.

Back Orifice

Alert Name	Total # seen	Classif.	Arachnids/CVE
Back Orifice	8	High (5)	ID397/CAN-1999-0660

Discussion: Five hosts made connection attempts to only 7 internal hosts. This again is not an undirected scan. This appears to be very targeted and efficient because

each host was not connected to more than 2 times. I believe the attackers own these boxes. What's more, 2 of the hosts are connecting from 27999 which is the port for the "master server" for the "Tribes 2 game" I doubt this is actual gaming traffic since the signature looks for a specific string in the data stream as well. (That is unless the signature is very out of date) These machines are certainly launching points for attackers to carry out other exploits and reconnaissance.

Correlation: Here is a page with an explanation of Back Orifice (http://www.irchelp.org/irchelp/security/bo.html)

Here is Shawn Beatty's practical which contains a detect analysis of Back Orifice (http://www.giac.org/practical/Shawn_Beatty.doc)

Recommendations: The Trojan should be checked for and removed from the internal hosts. All recent virus scanners detect and defend against this Trojan. Users should be educated/helped to keep their virus definitions up to date. Possibly a firewall with virus protection could be deployed at the perimeter.

Summary of Scans

Following is an analysis of the Scans.* log files over the five days chosen. Some of these are a repeat of previously seen activity. Some of it sheds new light on the alerts analyzed above. I split it into 3 components, UDP scans, SYN scans, and the rest of the scans. UDP and SYN scan alerting is notorious for false alarms and are therefore the hardest to deal with. However, I didn't want to completely throw the data out because there are probably useful items in there and if nothing else they should be analyzed in order to provide a basis for removing the harmless alerts from the IDS analysis process in the future.

UDP Scans

Discussion: There were over 1.96 million entries for UDP scans over this 5 day period. That was even ignoring any scan alerts in the log where there were fewer than 5. Obviously there is something wrong here. There is no way to effectively deal with this number of alerts on a weekly basis. Here is a table listing all of the internal hosts conducting more than 10 thousand scans.

# Scans	SRC Address
788282	MY.NET.70.176
164180	MY.NET.83.153
138769	MY.NET.84.178
135209	MY.NET.114.45
119556	MY.NET.88.228
113696	MY.NET.84.244
75575	MY.NET.91.252
47047	MY.NET.118.6
45470	MY.NET.88.220
43477	MY.NET.153.153

31556	MY.NET.153.142
27302	MY.NET.88.194
25872	MY.NET.86.106
19378	MY.NET.153.199
17299	MY.NET.108.42
17077	MY.NET.189.61
15701	MY.NET.88.69
14274	MY.NET.70.200
13452	MY.NET.137.7
11988	MY.NET.88.182
10774	MY.NET.153.157

By far the largest IP address as the source for these scans was from MY.NET.70.176. (788,282 entries) The vast majority of that traffic was a destined for port 6257 which is the default port for the P2P file sharing program "WinMX". Therefore this is an active WinMX client/server looking for other connections or actively sharing files. This host was also very active on some other interesting ports, 65535, 9000, 6348, 6699, 10000, 6000, 5556 and others. These are all either P2P file sharing ports or known Trojan ports. This host is completely compromised and looks to be actively targeting other hosts. I can't believe anyone can actually use it for anything else with this much activity going on. Perhaps it is a Trojan Honey Pot?

The second largest source IP address was MY.NET.83.153 with 164,180 entries. This host is interesting since, by contrast, it is talking mostly on lower range port numbers such as 1214, 1394, 1966, 1313, 1991. Although there is much less activity than the previous host, this host should also be carefully scrutinized for the presence of Trojans. It is also likely heavily infected.

The rest of the traffic from the hosts in the list above is very similar. There appears to be a real problem on this network with UDP Trojans!

Correlation: Here is the page that I used to identify many of the ports listed as Trojans (http://www.hitekredneck.net/ports.asp)

Here is a good SANS page listing known Trojan ports as well. This was contributed by Joakim von Braun (http://www.sans.org/newlook/resources/IDFAQ/oddports.htm)

Recommendations: If this host is not a Honey Pot, it should be taken offline immediately and reformatted. Or, perhaps it could be saved and used as an example of how bad things can get in an uncontrolled environment. At first I thought that the UDP scan alert was firing way too much, but I don't think so any longer. It could possibly be tweaked so that the threshold is a little higher so that the security department can concentrate on the worst offenders first, but there is definitely a big problem here and it shouldn't be ignored or masked.

SYN Scans

Discussion: There were far fewer SYN scan alerts (241254) but still a lot of traffic to analyze in any detail. Much of the traffic looks like it is probably regular network traffic for HTTP, SMTP, NetBIOS along with P2P traffic like Edonkey, Gnutella, and KaZaA. There is some definite active scanning for Trojans though. Some internal hosts in particular were doing some very obvious scanning (MY.NET.140.47, MY.NET.83.153) on many Trojan ports for external hosts. There are also a few external hosts (i.e. 209.11.36.196) that are actively scanning across many different internal hosts. This particular host was scanning for port 445 (Windows 2000 CIFS or MicrosoftDS) specifically. This is not the normal 445 and 135 traffic that is seen from Microsoft boxes while web surfing or making other connections.

Correlation: Here are several posts to intrusions@incidents.org that discuss SYN scan activity (http://cert.uni-stuttgart.de/archive/intrusions/2002/05/msg00007.html)

There are many correlations for Trojan scans that can be found (http://www.sans.org/y2k/051200.htm)

Recommendations: Again ingress and egress filtering is a must for stopping this activity. Up to date virus scanning software should also be deployed if it isn't already put into place to help protect from Trojan infections. The scan pre-processor should be adjusted so that it doesn't alert quite as easily. Instead of the default 4 packets in 3 seconds, it should be increased to 6-10 and adjusted as needed either up or down to reduce the amount of "noise" that must be waded through.

Other Scans Non-SYN Non-UDP

Discussion: The rest of the scans caused alerts because of non-standard or non-existent TCP flags or reserved bits set. These are usually used for reconnaissance such as OS fingerprinting or mapping out a network by the specific responses garnered from the network hosts.

The following table summarizes the activity seen for these scan alerts for instances where more than 5 alerts were generated.

106	NULL
103	VECNA
99	INVALIDACK
39	NOACK
27	UNKNOWN
18	FIN

NULL scans are used for reconnaissance because hosts with the port open won't respond at all but hosts with the port closed will send a reset packet when they receive a packet without any flags set. These 106 scans are mostly from 65.60.155.113, 209.193.36.1, and 63.204.132.240.

VECNA scans typically use packets without the ACK bit set. Any of the other bits can be set. Tod Beardsley pulls together a nice explanation of this in his practical (http://www.giac.org/practical/Tod_Beardsley_GCIA.doc). Again, except for KaZaA, as Tod pointed out, these packets are used for reconnaissance. This is really just background noise on the network.

INVALIDACK scans are explained by William Stearns in (http://www.incidents.org/archives/intrusions/msg14734.html) as packets with the ACK bit set but no other valid combination of flags and no other combination that signifies another scan type. Again for this scan type there is no glaring pattern here. Quite a few packets have sources or destinations of 6346 - 6348 – Gnutella. Perhaps some Gnutella code causes these non-standard packets.

NOACK scans are scans with packets that might have any of the other flags set or all of the other flags set but which do not have the ACK flag set. These are probably real scans. Again, this is really background noise on the network unless it can be correlated to other activity. These should be blocked at the perimeter.

FIN scans are scans which only have the FIN flag set. This is almost certainly a false positive since almost every alert is related to either a Gnutella or a KaZaA port. I found a Snort Snarf log that shows similar activity also related to Gnutella ports (http://www.nd.edu/~dmehlber/ids/html3/html/129/252/127/dest129.252.127.65.html) but no indication of malicious activity.

Correlation: Included in-line above. Scans such as these are very common. NMAP and Queso were designed to produce non-standard scans on purpose in order to get around firewalls and also do OS fingerprinting.

Recommendations: These scans are done for reconnaissance purposes and are mostly background noise on networks that are protected by firewalls and/or filtering routers. If this traffic can be filtered, then this noise should be ignored and filtered out. If the traffic can't be filtered out, then the only hope is to sufficiently patch and protect the individual hosts.

Out Of Spec Packets

4079 OOS packets logged over the five days. The following table shows summary data for the top source addresses and their targets for these packets.

Count	SRC Add	DST Add	DST Ports	
391	MY.NET.70.183	MY.NET.1.4	37	rdate
357	MY.NET.53.10	MY.NET.1.4	37	rdate
274	194.106.96.8	MY.NET.70.231	80	http
261	MY.NET.53.84	MY.NET.1.4	37	rdate
193	202.156.128.218	MY.NET.117.10	21, high ports	FTP session
103	63.98.19.244	MY.NET.27.210	113	ident
91	80.223.198.153	MY.NET.71.164	4662	Edonkey

82	209.47.251.30	MY.NET.6.40	25	smtp
77	209.47.251.14	MY.NET.6.40	25	smtp
56	209.47.251.23	MY.NET.6.40	25	smtp
55	209.47.251.24	MY.NET.6.40	25	smtp
53	209.47.251.18	MY.NET.6.40	25	smtp
52	209.167.239.27	MY.NET.6.40	25	smtp
51	209.47.251.22	MY.NET.6.40	25	smtp
49	209.47.251.13	MY.NET.6.40	25	smtp
49	209.47.251.16	MY.NET.6.40	25	smtp
49	209.47.251.20	MY.NET.6.40	25	smtp
47	209.167.239.29	MY.NET.6.40	25	smtp
45	217.84.3.74	MY.NET.71.164		smtp

Discussion: Generally speaking OOS packets are somehow crafted or have been transmitted by faulty IP stacks. They can be used for OS fingerprinting, reconnaissance, or active exploits. In short, they are usually evidence of malicious activity of some kind.

Three of the top 4 entries in the table above all have the same destination address and the packets are all very similar. They are all from internal hosts and have a destination address of one specific host. They are attempting to reach the "time" port of this machine, port 37. After sorting through the data, this traffic looks especially strange. Packets in the exchanges often increment their source ports in coordination with each other. The TCP flags are all unset and the IP ID's look unusually low and consistent. The Seq numbers look crafted as well since they are identical between 2 of the 3 hosts and not that much different on the 3rd. The sequence numbers are also nowhere near random for any of these packets. I know there have been vulnerabilities in inetd for Linux in the past that could cause the service to crash if enough SYN packets were sent to port 37, but this is different. Also the TTL for these packets is 64 suggesting that the source is either a Linux or a BSD box. I suspect all of these machines are compromised.

12/13	6	5	36.631832	MY.NET.70.183	61780	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:226	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	6	5	45.61632	MY.NET.70.183	61780	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:230	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	6	6	5.586449	MY.NET.70.183	61780	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:233	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	6	6	21.562635	MY.NET.70.183	61780	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:234	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	6	6	51.515777	MY.NET.70.183	61780	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:235	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	7	0	17.875887	MY.NET.53.84	61781	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:189	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	7	0	21.872762	MY.NET.53.84	61781	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:190	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	7	0	37.848669	MY.NET.53.84	61781	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:192	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	7	0	53.82384	MY.NET.53.84	61781	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:193	IpLen:20	DgmLen:40	********	Seq:0x89C00000
12/13	7	0	56.597635	MY.NET.53.10	61782	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:241	IpLen:20	DgmLen:40	********	Seq:0xC2000000
12/13	7	1	5.587489	MY.NET.53.10	61782	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:244	IpLen:20	DgmLen:40	********	Seq:0xC2000000
12/13	7	1	11.578115	MY.NET.53.10	61782	MY.NET.1.4	37	TCP	TTL:64	TOS:0x0	ID:245	IpLen:20	DgmLen:40	********	Seq:0xC2000000

The 3rd entry in the table is again a set of packets that only occur between these two hosts. Unlike the last set discussed, these look to be relatively normal packets except that they have the two ECN bits set. This could be a Queso (or other tool) scan. Indeed, this same address set shows up in the alerts discussed above.

The next one with 193 alerts at first reminds me a lot like a normal FTP session except that the external client doesn't stick to one higher level port for the data connection as it

should in passive FTP. It jumps all around and the traffic is spread out over multiple days and is going rather slowly. I still think this is normal FTP traffic however, I think we are seeing multiple FTP sessions over many hours and many days. Entries only show up in the OOS logs when the ECN bits are set and these are only set during the initial handshake to try to determine if the other end is ECN aware.(http://www.sans.org/y2k/ecn.htm) Since it is not in this case, we see only the first SYN packet sent on the data connection.

In fact 2,689 of the OOS alert packets have the ECN and SYN bits set alone. These are either OS fingerprinting scans or hosts trying to negotiate ECN communications. I tend to think most of them are the later. The rest of the top OOS alert generators are of this same type.

The remainder of the OOS packets consisted of relatively few instances and they don't appear to be widespread scans. There are targeted to just a couple of hosts from a specific IP set of IP addresses address. These were evident in the discussion and analysis of the scan logs above. There are XMAS and FULLXMAS packets in here but only a few of each. Not a widespread scan. That is probably good news, but it could also indicate that the attacker has narrowed down his target.

Top 10 Talkers

The following are the top 10 talkers in terms of numbers of alerts in each category for both source and destination IP addresses. I plan to use these addresses to do some combined analysis across all of the alert files to see if there are any other patterns or other important information that was missed by looking at only the specific alerts one at a time. It seems most reasonable to look for these patterns among the hosts that are doing the most talking on the network.

Scans Top Talkers

Source IPs	Frequency
MY.NET.70.176	789354
MY.NET.83.153	167522
MY.NET.84.178	139174
MY.NET.114.45	135679
MY.NET.88.228	119710
MY.NET.84.244	114192
MY.NET.86.110	102617
MY.NET.91.252	76786
MY.NET.118.6	47507
MY.NET.88.220	45618

Destination IPs	Frequency
216.207.229.67	38530
68.57.239.69	34492
24.24.23.6	8369
24.239.158.197	7469
204.183.84.240	7434
66.186.79.50	4366
67.34.8.178	3531
64.156.139.131	2943
66.28.140.133	2893
66.93.54.212	2133

Alerts Top Talkers

Source IPs	Frequency
159.226.221.127	14733
192.168.5.2	5631
MY.NET.111.232	3492
MY.NET.111.230	3488
MY.NET.111.235	3483
MY.NET.111.231	3476
MY.NET.111.219	3451
212.179.83.121	2257
66.183.191.107	1806
212.179.107.228	1506

Destination IPs	Frequency
192.168.0.253	17391
MY.NET.153.178	14705
MY.NET.24.16	5575
MY.NET.114.45	3265
MY.NET.86.106	1388
MY.NET.118.6	1226
MY.NET.6.40	959
213.243.0.2	761
MY.NET.139.46	738
MY.NET.90.217	711

OOS Packets Top Talkers

Source IPs	Frequency
MY.NET.70.183	391
MY.NET.53.10	357
194.106.96.8	274
MY.NET.53.84	261
202.156.128.218	193
63.98.19.244	134
80.223.198.153	91
209.47.251.30	82
209.47.251.14	80
209.47.251.23	59

Destination IPs	Frequency
MY.NET.6.40	1296
MY.NET.1.4	1009
MY.NET.185.48	276
MY.NET.70.231	274
MY.NET.71.164	248
MY.NET.117.10	193
MY.NET.27.210	103
MY.NET.113.4	87
MY.NET.84.144	60
MY.NET.114.45	39

Cross-file analysis

Scans to others

MY.NET.70.176 is involved in heavy P2P file sharing activity. It is not in the top 10 of the Alerts, but it was number 13. The vast majority of the alerts were for port 65535 activity that could be associated with the Adore or Red worm, but the port on the other side of the conversation is 6257 which is the default port for WinMX.

MY.NET.83.153 generated many alerts with watchlist member 212.179.87.188. This watchlist member is connecting to some service running on the internal machine at port 3806 or there is something crafting packets on MY.NET.83.153 with a source port of 3806. There are also OOS packets showing up directed at this machine on port 3806. This internal machine was also involved in some ISAPI and NMAP alerts as well so it should definitely be closely checked out.

MY.NET.84.178 also appears to be running WinMX because it is causing many Adore/Red Worm alerts as well. It is also active with a member of Watchlist 000220. It has also been the recipient of a set of OOS packets destined for port 6699 (the TCP side of WinMX)

MY.NET.114.45 is doing P2P also with Watchlist 0002200 (and certainly others) but they are using KaZaA. KaZaA also generates OOS packets as discussed above. Code red types of traffic also show up for this host.

MY.NET.88.228 doesn't have any alerts or OOS packets at all, but it appears to be scanning for or responding to scans for "Dobol" a remote access Trojan (http://lists.insecure.org/lists/incidents/2002/Sep/0055.html). This machine should be taken offline and cleaned.

MY.NET.84.244 appears to be very active on ICQ with some Watchlist 0002200 members and others as evidenced by the large number of scans in the scan alerts files. Alerts also exist for some IIS alerts but they may be false positives.

MY.NET.86.110 appears to be scanning a handful of hosts on the internet very quickly but continuously with some automated tool. The destination ports don't make it seem like it is a Trojan scan. They seem to vary randomly instead of increasing sequentially or in an orderly pattern as one would expect in a scan. On the internal host, however, the packets show an orderly increase of ports as you would expect. The internal host is probably running a DOS attack against these external hosts. If the external host was scanning inbound and choosing random source ports, we should only see responses on the handful of ports that were open and we would see the scan alerts from the inbound traffic but it doesn't look that way. From the alert, scan, and OOS logs, I don't see any evidence that it is acting like a Trojan. This machine should be taken offline and cleaned or rebuilt.

MY.NET.91.252 appears to have an active Trojan listening on it, possibly on port 1237. I haven't found any documentation of known Trojans on this port, but someone could have easily altered existing code to work on this port. From what I have been able to piece together, the scenario works like this. Inbound connections destined to port 1237 are coming in from Watchlist 0002200 (and possibly others, I wouldn't see the connections) along with OOS packets destined for 1237. These are all TCP. Curiously MY.NET.91.252 is then scanning other external hosts with a source port of 1237 at 5-20 packets per second. All of these are UDP and the destination ports and possibly addresses appear to be somewhat random. It looks as though the TCP connections might be control connections. More research on actual packet data, not just alert files needs to be done to verify this activity.

MY.NET.118.6 is probably offering some service on port 3011. I can't tell simply from these log files, but it appears to be carrying on UDP sessions with multiple hosts at the same time. One possibility is a multi-user game called "SubSpace." They list instructions on their website for setting up a proxy listening on this port for others to use to connect. (http://games.igateway.net/subspace/proxy.html) I doubt it is the assigned purpose of this port: "trusted-web."

MY.NET.88.220 is generating all of this traffic with the WinMX P2P file sharing software.

204.183.84.240 is doing a lot of DNS transactions with one internal host. This is probably something like a log parsing tool or something that is trying to do name resolution for all of the entries it processes.

66.93.54.212 is carrying on a conversation with MY.NET.83.116. Alerts show an active TFTP session between the two along with some activity on 515. Also, MY.NET.83.116 is scanning 66.93.54.212 repeatedly for open low ports. The source port changes very little so it looks like a script that doesn't spawn a new process for each UDP packet that is sent out or a tool that crafts packets. It sent 2133 packets in just under 5 minutes

Alerts to others

This section will only contain incidents of significance that have not already been reported on.

As alluded to above, 66.183.191.107 is scanning a large range of Internal Address space for reconnaissance. Specifically he is targeting large class C blocks of the MY.NET.x.x network.

MY.NET.86.106 appears to be answering a lot of traffic that could be for a Smart Card. If this is a Smart Card server, then this traffic should be ignored and filtered out. If it isn't then some other service is doing quite a bit of work on this port whether it is a good service or not is unknown.

OOS to others

Again these are only correlations or incidents not already covered.

Nearly all of these are related to the ECN bit being set as discussed above. The rest have already been covered.

Analysis Process

In general my analysis process was to take each type of alert file for each day and join each respective type of file together with "cat." Then I used UNIX command line tools including grep, sed, uniq, wc and others to parse out the data. I also used several perl scripts and an awk script that I found from other GCIA papers to parse the data into meaningful patterns. I found that Excel was also a wonderful tool using the sort and filter functions to help me isolate and look at the pertinent data. Below is a list of the scripts that I used and a link to the source of each. I sure am glad there are so many good perl programmers out there. It made my job much easier than it would have been otherwise.

csv.pl, summarize.pl	http://www.giac.org/practical/Tod_Beardsley_GCIA.doc
Alertcount.pl, ipsort.pl, scanalyze.pl, scancount.pl	http://www.giac.org/practical/chris_kuethe_gcia.html
Top_talkers.pl,	http://www.giac.org/practical/Mike_Bell_GCIA.doc

top_talkers_oos.pl,	
Scanstat.awk	http://www.sans.org/practical/Crist_Clark_GCIA.html

In order to find relationships across the alert files, I took the output of the top_talkers.pl scripts and used grep with several pipes, etc… to find relationships.

REFERENCES/BIBLIOGRAPHY

Beardsley, Tod. "Intrusion Detection and Analysis: Theory, Techniques, and Tools." May 8, 2002. URL: http://www.giac.org/practical/Tod_Beardsley_GCIA.doc. (Jan. 4, 2003)

Bell, Mike. "GCIA Practical." Jan. 2001. URL: http://www.giac.org/practical/Mike_Bell_GCIA.doc. (Jan. 4, 2003)

Clark, Crist. "Crist Clark - GCIA Practical Assignment." Oct. 20, 2000. URL: http://www.giac.org/practical/Crist_Clark_GCIA.html. (Jan. 4, 2003

Clark, Crist. "Re: Strange Broadcasts to Printer Port." Security Focus Online. Jun 29 2001. URL: http://online.securityfocus.com/archive/75/194288. (Jan. 3, 2003)

"defcon" <ion.storm@verizon.net>. "More XDCC/DDOS bots found on efnet #x-dcc!" May 7, 2002. URL: http://www.theorygroup.com/Archive/Unisog/2002/msg00618.html. (Jan. 4, 2003)

Dittrich, Dave. "World-wide distributed DoS and "warez" bot networks." May 3, 2002. URL: http://staff.washington.edu/dittrich/talks/core02/xdcc-analysis.txt. (Jan. 4, 2003)

Ellis, Joe. "GCIA Practical Assignment, v3.0 Intrusion Detection In Depth." May 14, 2002. URL: http://www.giac.org/practical/Joe_Ellis_GCIA.doc. (Jan. 4, 2003)

Garris, John. "Level II Intrusion Detection GCIA Practical Assignment – Version 2.7." URL: http://www.giac.org/practical/John_Garris_GCIA.doc. (Jan. 4, 2003)

Hoover, James. "GCIA Practical Version 3.0." Dec. 23, 2000. URL: http://www.giac.org/practical/James_Hoover_GCIA.doc. (Jan. 4, 2003)

"How do I setup a proxy server to work with SubSpace?" URL: http://games.igateway.net/subspace/proxy.html. (Jan. 4, 2003)

Jongen, Martijn. "Portnumbers." 2002. URL: http://www.martijnjongen.com/portnumbers.htm. (Jan. 4, 2003)

Junginger, Jeremy. "Security Incidents: RE: UDP port 22321." Incidents@insecure.org Sept. 9, 2002. URL: http://lists.insecure.org/lists/incidents/2002/Sep/0055.html.

(Jan. 4, 2003)

Kuethe, Chris. “Chris Kuethe: GCIA Practical Assignment.” URL: http://www.giac.org/practical/chris_kuethe_gcia.html. (Jan. 4, 2003)

Lutton, R. “Cisco 675 Setup for QWEST.net - PPP mode.” URL: http://www.users.qwest.net/~rlutton/ADSL/NAT_Settings.html. (Jan. 4, 2003)

Martin, Daniel. “Re: Spoofed SMB name wildcard probes.” Incidents @ Security Focus.com mailing list. May 4, 2001. URL: http://cert.uni-stuttgart.de/archive/incidents/2001/05/msg00041.html. (Jan. 3, 2003)

Melhber, D. “SnortSnarf Alert Page.” March 20,2002. URL: http://www.nd.edu/~dmehlber/ids/html3/html/129/252/127/dest129.252.127.65.html. (Jan. 4, 2003)

Miller, Toby. “ECN and it’s impact on Intrusion Detection.” Global Incident Analysis Center - Special Notice -. 2000. URL: http://www.sans.org/y2k/ecn.htm. (Jan. 4, 2003)

Mixter. “Q-2.4” April 15, 2001. URL: http://packetstormsecurity.org/groups/mixter/ (Jan. 3, 2003)

Multiple Authors. “1214 Scans” Incidents.org Intrusions Index (by Thread). Feb. 4, 2002. URL: http://www.incidents.org/archives/intrusions/thrd118.html. (Jan. 4, 2003)

Multiple Authors. “SNORT FAQ” v 1.14. March 25, 2002. URL: http://www.snort.org/docs/4.17. (Jan. 3, 2003)

Northcutt, Stephen. “What was the Ring Zero scan?” Intrusion Detection FAQ. Oct. 11, 1999. URL: http://www.sans.org/resources/idfaq/ring_zero.php. (Jan. 3, 2003)

Online Discussion. “Back Door Q Access?.” Security Focus Online. May 4, 2001. URL: http://online.securityfocus.com/archive/75/182244/2002-11-04/2002-11-10/1. (Jan. 3, 2003)

Riddell, Trenton. “GCIA Certification Practical Assignment.” March 6, 2002. URL: http://www.giac.org/practical/Trenton_Riddell_GCIA.doc. (Jan. 3, 2003)

Ryan, John. "Security Logs and Checkpoint Firewall-1." Firewalls & Perimeter Protection. June 4, 2001. URL: http://www.sans.org/rr/firewall/logs.php. (Jan. 3, 2003)

Stearns, William. “Re: Scan Analysis” Intrusions@incidents.org May 31, 2002. URL: http://www.incidents.org/archives/intrusions/msg14734.html. (Jan. 4, 2003)

“System Log Messages.” Cisco PIX Firewall System Log Messages, Version 6.2. Dec. 30 2002. URL:

http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_62/syslog/pixemsgs.htm#42801. (Jan. 3, 2003)

Toren, Montgomery. “Intrusion Detection In Depth.” V 3.0. Feb. 11,2002. URL: http://www.giac.org/practical/Montgomery_Toren_GCIA.doc. (Jan. 4, 2003)

Wilkinson, Michael. “GCIA Practical for SANS Darling Harbour.” URL: http://www.giac.org/practical/michael_wilkinson_gcia.doc. (Jan. 4, 2003)

“Working Around ISP Port Blocks.” URL: http://winmx.2038.net/winmx/fr-blocked.html. (Jan. 4, 2003)

General Resources/Tools Used

Dshield. URL: http://www.dshield.org

GeekTools URL: http://www.geektools.com

Google. URL: http://www.google.com

IANA Port numbers: URL: http://www.iana.org/assignments/port-numbers

myNetWatchman. URL: http://www.mynetwatchman.com

Network-Tools: URL: http://www.network-tools.com

Last Updated: July 25th, 2017

Upcoming Training

SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
Baltimore Fall 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
Community SANS Pensacola SEC503	Pensacola, FL	Nov 27, 2017 - Dec 02, 2017	Community SANS
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Las Vegas 2018	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	Live Event
SANS Dallas 2018	Dallas, TX	Feb 19, 2018 - Feb 24, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced