Tim Kroeger

May 18, 2004

GCIA Practical
Version 3.4

Security Information Management Systems
What are they? Who makes them? Do I need one?

Summary

This practical is submitted for the SANS GCIA certification process. The first portion of this paper discusses Security Information Management Systems (SIM's). I discuss what SIM's are, who makes them and whether or not you need one.

The second portion of this paper analyzes three network detects. They are:

1. Scan NMAP TCP
2. WebDav: Search-Overflow
3. IIS:CMD.EXE

The last portion of this practical analyzes 5 days worth of IDS logs (Scan, Alert and OOS) from a University. Each log is analyzed for Top Talkers and any other specific information that can be obtained from the logs. The third section is wrapped up with some defensive recommendations based on the analyzed logs.

Section 1

What is a SIM?

IDS logs, firewall logs, router logs, system logs, anti-virus logs…. Oh my! For a security analyst to perform his or her job properly, he has to be very intimate with all these different logs, all of the different formats, what information is found in what log, the list could go on for awhile. So, what are we to do…? Tralalalaaaa…. SIM's to the rescue, right? Well, the jury is still out on that one. SIM's or Security Event Management tools are relatively new to the information security field. In this portion of my practical, I will tell you what a SIM is and what they attempt to do. I will also tell you which SIM's are out there and a little about how they differ. And lastly, I will ask the question, do I need one? It sure would be nice to solve all of my network security problems, but are SIM's the answer?

SIM's are also known as Security Event Management (SEM) systems. For the purpose of this paper, I will refer to all of them as SIM's.

So, what exactly is a Security Information Management system? A SIM is defined in a survey by Open Service Inc. (www.open.com) as a:

system that integrates threat data from security and network devices in order to filter out false alarms, link events from multiple data sources and identify false negatives to reduce unmanaged risks and improve operational efficiency. (Open)

Wow! That's a tall order… Sounds like SEM's are the answer to all of our problems. They will "identify false negatives"?? Well, according to the SIM makers, they ARE the answers to our problems. My definition of a SIM is: they are data correlation engines that take inputs from many different network devices, database the data and present it to users. Pretty simple, huh? Well… sort of.

For a SIM to be effective they have to perform four basic functions as shown in Figure 1 from netForensics (netForensics)

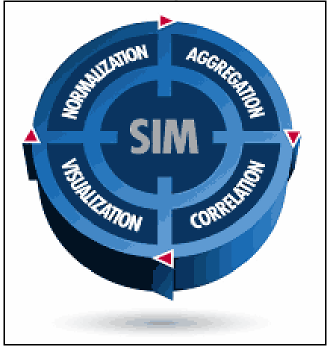


Figure 1

- Aggregation – a process of accepting inputs from several different network devices and combining them into one information flow.
- Normalization – logs and events gathered in the aggregation phase are transformed into one common format for insertion into the database and for further use in the SIM.
- Correlation – takes normalized events and uses different methods (rule, algorithm and statistical) to “find” incidents.
- Visualization – displays the incidents and events to the user in many different formats and allows the user to manipulate the data. ArcSight’s (ArcSight) visualization is shown below in Figure 2.

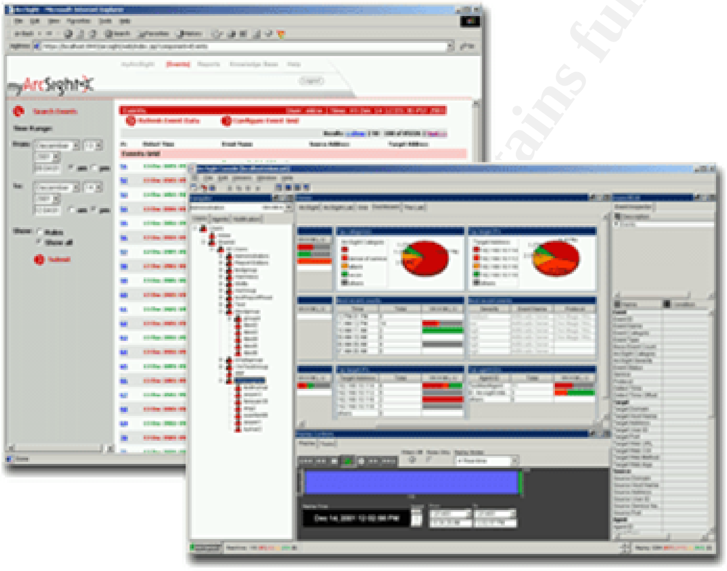


Figure 2

Who makes them?

There are several SEM’s on the market today. They all pretty much perform the same functions and they all have their own catch phrases to describe the same capabilities. Below is a list of several SIM’s on the market:

- ArcSight - www.arcsight.com
- Intellitactics - www.intellitactics.com
- netForensics – www.netforensics.com

- netIQ – www.netiq.com
- Open – www.open.com
- Network Intelligence – www.opensystems.com
- e-Security – www.esecurityinc.com
- neuSecure – www.guarded.net
- Open Source Security Information Management – www.ossim.net

As I stated above, most of the SIM's above all have similar capabilities. So, what is different about them? I found a review of several SIM's written by Tom Oel and Greg Shipley in September 2003 (Oel and Shipley). This review is the second conducted by Shipley and Internet Week which assessed SIM's. The first review was written in April 2002 (Shipley). Both reviews are definitely worth reading if you're looking into purchasing or learning more about SIM's. They show a nice progression in different products and let you see which products improved and what areas they improved in. This could be a sign of where specific products are headed and if they are worth your time and money.

During the most recent review of SIMS, Oel and Shipley compared the SIM's based on how they performed in the following categories (Oel and Shipley):

- Correlation Capabilities
- Architecture
- Reporting and Analysis
- Price
- Integration and Device Support

The above list is a nice representation of the areas that are important when researching SIM's. One area, I wish the review would have discussed in greater depth is customization and rule creation. Sure, it's nice to have default rules, but what about the new attacks and threats out there. How hard is it for me to take a new attack and create a rule for it? One SIM, neuSecure offers a rule testing option in their product. This allows you to test a newly created rule by testing it internally to the SIM.

Do I need a SIM?

At this point in my short presentation about SIMS, I'm sure you're asking; do I need a SIM? If you spend 8 hours a day sifting through log files looking for events, or are constantly producing reports for management of security related events, a SIM might be a good investment. Once up and running they will definitely help reduce some of the workload when searching through log files and many of the SIM's I've looked at have very extensive reporting features. I recommend that you sit down with management and figure out if you're getting the right information from your security devices. If you are satisfied with your current solution, then a SIM will probably just complicate things for you. Also, if your security devices are misconfigured or not being used properly, then no SIM is going to help you get those devices working properly.

No matter how nice a SIM sounds right now, they still have a lot of maturing to do. One of the drawbacks that I noticed is the complexity that comes with a product like a SIM. Many of the vendors claim that they can have their product up and running in a few hours. Well, in my opinion that's all they will have, is their product up and running. It will take many weeks and maybe months, depending on your network size, to get your SIM properly configured and tuned before it is actually useful. Once you have your SIM tuned and running smoothly, have fun, because SIM's will take your security team to the next level.

References:

Open Service Inc. "2003 Security Event Survey." URL: http://www.open.com/pdf/2003_Security_Event_Survey.pdf (17 May 2004).

NetForensics Inc. "Security Information Management." URL: http://www.netforensics.com/documents/pr_sim.asp (17 May 2004).

ArcSight. "ArcSight Console and myArcSight." URL: http://www.arcsight.com/product_info05.htm (17 May 2004).

Oel, Tom and Shipley, Greg. "Security Information Management Product Review." InternetWeek.com. URL: http://www.internetweek.com/shared/printableArticle.jhtml?articleID=14706765 (17 May 2004).

Shipley, Greg. "Security Information Management Tools: NetForensics Leads a Weary Fleet." Network Computing. URL: http://www.nwc.com/1307/1307f22.html (17 May 2004).

Martin, Michael. "Keys to Implementing a Successful Security Information Management Solution (or Centralized Security Monitoring)." SANS Institute. URL: http://www.giac.org/practical/GSEC/Michael_Martin_GSEC.pdf (17 May 2004).

Godfrey, Michael. "netForensics® – A Security Information Management Solution." Sans Institute. URL: http://www.sans.org/rr/papers/index.php?id=408 (17 May 2004).

Section 2

#1 Detect: SCAN nmap TCP

1. **Source of Trace:**

   This trace was downloaded from the http://www.incidents.org/logs/Raw. I downloaded and analyzed the 2002.10.14 file.

2. **Detect was generated by:**

   I used Snort 2.1.0 (Snort) running on Windows XP Professional with a rule set dated 26 February, 2004. I ran snort with the following command:

   ```
   snort –r 2002.10.14 –b –l c:\snort\log –c c:\snort\etc\snort.conf –A full
   ```

   Here are what the options mean according to the snort help file:

-r	<tf> Read and process tcpdump file <tf>
-b	Log packets in tcpdump format (much faster!)
-c	<rules> Use Rules File <rules>
-A	Set alert mode: fast, full, console, or none (alert file alerts only)

   There were a total of 42 alerts logged when I ran snort against the downloaded file.

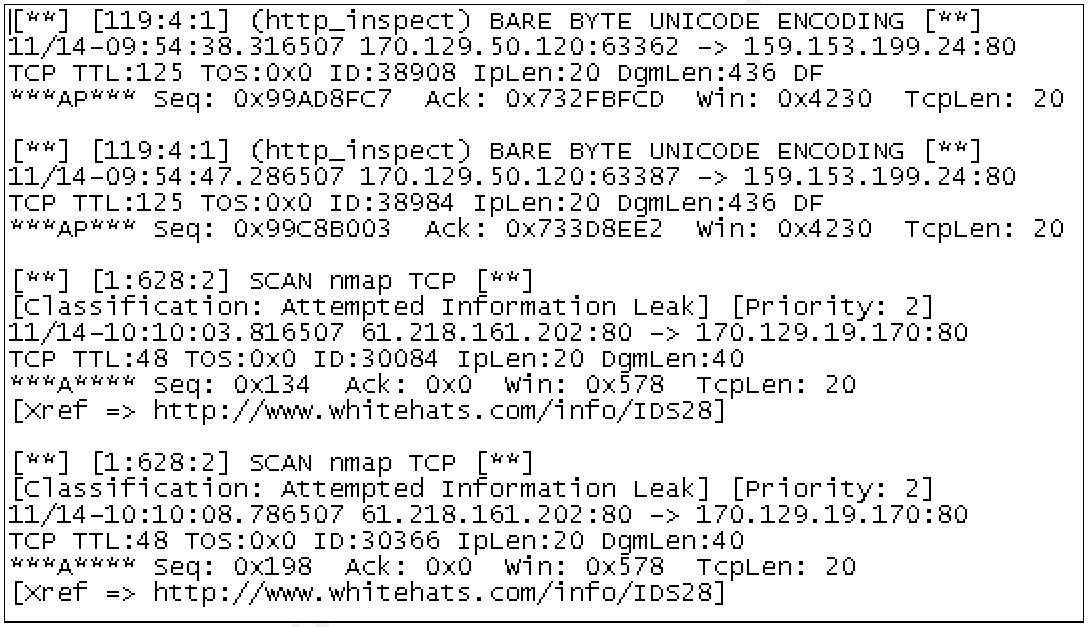
Alert	**Count**
BARE BYTE UNICODE ENCODING	3
BAD-TRAFFIC ip reserved bit set	4
SCAN Proxy Port 8080 attempt	1
SCAN Squid Proxy attempt	3
SCAN SOCKS Proxy attempt	2

SCAN nmap TCP	29

Table 1

I decided to analyze the SCAN nmap TCP alerts because there were an overwhelming number of them compared to the other five types of alerts.

A small portion of the SCAN nmap TCP alerts from the snort alert file are displayed in Figure 3.



```
[**] [119:4:1] (http_inspect) BARE BYTE UNICODE ENCODING [**]
11/14-09:54:38.316507 170.129.50.120:63362 -> 159.153.199.24:80
TCP TTL:125 TOS:0x0 ID:38908 IpLen:20 DgmLen:436 DF
***AP*** Seq: 0x99AD8FC7  Ack: 0x732FBFCD  Win: 0x4230  TcpLen: 20

[**] [119:4:1] (http_inspect) BARE BYTE UNICODE ENCODING [**]
11/14-09:54:47.286507 170.129.50.120:63387 -> 159.153.199.24:80
TCP TTL:125 TOS:0x0 ID:38984 IpLen:20 DgmLen:436 DF
***AP*** Seq: 0x99C8B003  Ack: 0x733D8EE2  Win: 0x4230  TcpLen: 20

[**] [1:628:2] SCAN nmap TCP [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/14-10:10:03.816507 61.218.161.202:80 -> 170.129.19.170:80
TCP TTL:48 TOS:0x0 ID:30084 IpLen:20 DgmLen:40
***A**** Seq: 0x134  Ack: 0x0  Win: 0x578  TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS28]

[**] [1:628:2] SCAN nmap TCP [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/14-10:10:08.786507 61.218.161.202:80 -> 170.129.19.170:80
TCP TTL:48 TOS:0x0 ID:30366 IpLen:20 DgmLen:40
***A**** Seq: 0x198  Ack: 0x0  Win: 0x578  TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS28]
```

Figure 3

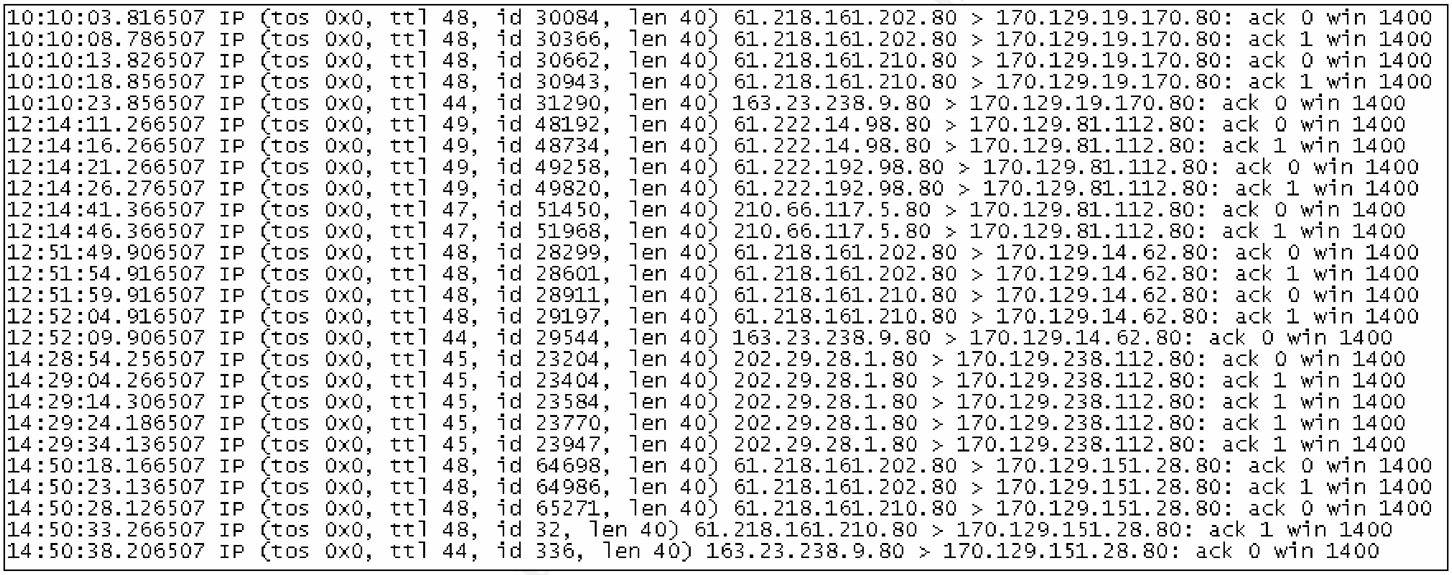
The snort rule that triggered the alert is:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN nmap TCP"; stateless; flags:A,12; ack:0; reference:arachnids,28; classtype:attempted-recon; sid:628

3. Probability the source address was spoofed:

To determine if the source address was spoofed, we need to look at some characteristics of the raw traffic that triggered the snort alerts. All of the SCAN nmap TCP alerts triggered by snort had a source port and destination of 80 and all had a window size of 1400. So, I ran windump with the following options to pull the packets out that I wanted to look at in greater detail. I could have used the snort binary log that was created when I ran snort against this file, but that file only contains the actual packets that triggered the alerts in the alert file. In this case, I wanted to see all packets with a source port of 80 to a destination net of 170.129 on dst port 80 with the window size of 1400. The results are displayed in Figure2.

windump –r 2002.10.14 –nv “src port 80 and dst net 170.129 and dst port 80 and tcp[14:2] = 0x578”



```
10:10:03.816507 IP (tos 0x0, ttl 48, id 30084, len 40) 61.218.161.202.80 > 170.129.19.170.80: ack 0 win 1400
10:10:08.786507 IP (tos 0x0, ttl 48, id 30366, len 40) 61.218.161.202.80 > 170.129.19.170.80: ack 1 win 1400
10:10:13.826507 IP (tos 0x0, ttl 48, id 30662, len 40) 61.218.161.210.80 > 170.129.19.170.80: ack 0 win 1400
10:10:18.856507 IP (tos 0x0, ttl 48, id 30943, len 40) 61.218.161.210.80 > 170.129.19.170.80: ack 1 win 1400
10:10:23.856507 IP (tos 0x0, ttl 44, id 31290, len 40) 163.23.238.9.80 > 170.129.19.170.80: ack 0 win 1400
12:14:11.266507 IP (tos 0x0, ttl 49, id 48192, len 40) 61.222.14.98.80 > 170.129.81.112.80: ack 0 win 1400
12:14:16.266507 IP (tos 0x0, ttl 49, id 48734, len 40) 61.222.14.98.80 > 170.129.81.112.80: ack 1 win 1400
12:14:21.266507 IP (tos 0x0, ttl 49, id 49258, len 40) 61.222.192.98.80 > 170.129.81.112.80: ack 0 win 1400
12:14:26.276507 IP (tos 0x0, ttl 49, id 49820, len 40) 61.222.192.98.80 > 170.129.81.112.80: ack 1 win 1400
12:14:41.366507 IP (tos 0x0, ttl 47, id 51450, len 40) 210.66.117.5.80 > 170.129.81.112.80: ack 0 win 1400
12:14:46.366507 IP (tos 0x0, ttl 47, id 51968, len 40) 210.66.117.5.80 > 170.129.81.112.80: ack 1 win 1400
12:51:49.906507 IP (tos 0x0, ttl 48, id 28299, len 40) 61.218.161.202.80 > 170.129.14.62.80: ack 0 win 1400
12:51:54.916507 IP (tos 0x0, ttl 48, id 28601, len 40) 61.218.161.202.80 > 170.129.14.62.80: ack 1 win 1400
12:51:59.916507 IP (tos 0x0, ttl 48, id 28911, len 40) 61.218.161.210.80 > 170.129.14.62.80: ack 0 win 1400
12:52:04.916507 IP (tos 0x0, ttl 48, id 29197, len 40) 61.218.161.210.80 > 170.129.14.62.80: ack 1 win 1400
12:52:09.906507 IP (tos 0x0, ttl 44, id 29544, len 40) 163.23.238.9.80 > 170.129.14.62.80: ack 0 win 1400
14:28:54.256507 IP (tos 0x0, ttl 45, id 23204, len 40) 202.29.28.1.80 > 170.129.238.112.80: ack 0 win 1400
14:29:04.266507 IP (tos 0x0, ttl 45, id 23404, len 40) 202.29.28.1.80 > 170.129.238.112.80: ack 1 win 1400
14:29:14.306507 IP (tos 0x0, ttl 45, id 23584, len 40) 202.29.28.1.80 > 170.129.238.112.80: ack 1 win 1400
14:29:24.186507 IP (tos 0x0, ttl 45, id 23770, len 40) 202.29.28.1.80 > 170.129.238.112.80: ack 1 win 1400
14:29:34.136507 IP (tos 0x0, ttl 45, id 23947, len 40) 202.29.28.1.80 > 170.129.238.112.80: ack 1 win 1400
14:50:18.166507 IP (tos 0x0, ttl 48, id 64698, len 40) 61.218.161.202.80 > 170.129.151.28.80: ack 0 win 1400
14:50:23.136507 IP (tos 0x0, ttl 48, id 64986, len 40) 61.218.161.202.80 > 170.129.151.28.80: ack 1 win 1400
14:50:28.126507 IP (tos 0x0, ttl 48, id 65271, len 40) 61.218.161.210.80 > 170.129.151.28.80: ack 0 win 1400
14:50:33.266507 IP (tos 0x0, ttl 48, id 32, len 40) 61.218.161.210.80 > 170.129.151.28.80: ack 1 win 1400
14:50:38.206507 IP (tos 0x0, ttl 44, id 336, len 40) 163.23.238.9.80 > 170.129.151.28.80: ack 0 win 1400
```

Figure 4

A couple of things jumped out at me right away to make me think that the source addresses are spoofed.

TTL – The TTL field indicates the maximum time the datagram is allowed to remain in the internet system (Northcutt and Novak). Every time the datagram passes through a router, the router decrements the ttl field by one and sends it one. The ttl field is set to a specific value by the operating system when the ip header is added the datagram. Of course the operating system vendors couldn’t make this easy on us by agreeing on a ttl, so each operating system sets the ttl to a different value. Below are a couple of initial ttl values for some more popular operating systems plus nmap according to my SANS Passive O/S fingerprinting t-shirt (SANS).

Operating System	**Initial TTL**

Novell	128
Cisco IOS	255
Linux	64
Mac OS	255
Windows (all versions)	128
Nmap	64

Table 2

The TTL (time to live) of all of the packets is between 44 and 49. That would mean that all seven of the source addresses should all be running the same operating system and all be about the same number of hops away from the destination network. This is strange, because when I looked up the IP addresses on ARIN and APNIC, they resolved to Taiwan, Thailand, Colorado, USA and New York, NY. It could be possible, but pretty rare that all would have similar hop counts.

Window size – The window size tells the transmitting host how much data it can send before it has to stop and wait for an acknowledgement from the destination host. Using my handy dandy SANS Passive OS t-shirt again (SANS), I looked up the default window size for the same operating systems and nmap. (it sure would be nice if the passive O/S matrix was added to the tcp/ip and tcpdump pocket reference…)

Operating System	**Default Window Size**
Novell	don't care
Cisco IOS	4128
Linux	32120 or 5840 (depending on version)
Mac OS	don't care
Windows (all versions)	16384, don't care, 8192 (depending on

	version)
Nmap	don’t care

Table 3

I’ve already stated that the window sizes for all the packets were set to 1400. This is possible if all of the sources were talking to the same destination host or if all if all of packets were crafted using the same tool.

IP Length – The IP length indicates the total size of the IP datagram. I broke out the default packet sizes according to my well worn SANS t-shirt (SANS) the same way I did with window sizes and ttl.

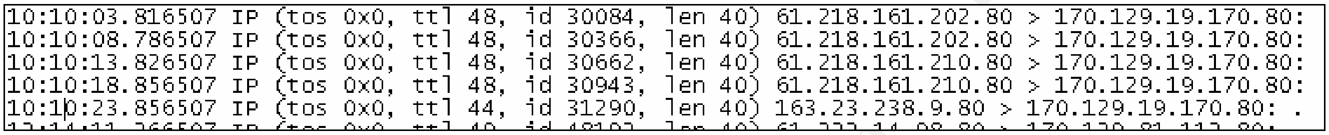
Operating System	**Packet size**
Novell	don’t care
Cisco IOS	don’t care
Linux	don’t care
Mac OS	48
Windows (all versions)	44,48 or 60 (depending on version)
Nmap	40 or 60

Table 4

If you refer back up to figure 2, you can see that the IP total length is 40 bytes. The only O/S or tool that sets the packet size to 40 by default is nmap. That makes sense since the snort alert was SCAN nmap TCP.

Source & Destination Port – The source and destination port is set to 80 for all of the packets in snort alert log. This indicates that the packet is crafted. Nmap (Fyodor) allows the user to specify a source port to use when conducting scans.

Time – I have displayed all of the offending packets that were transmitted in the 10 o’clock hour in Figure 3. There are three unique source addresses and one destination address. The first source transmits at 10:10:03 and 10:10:08. The second source transmits at 10:10:13 and 10:10:18. And the last source transmits at 10:10:23. The first and second source hosts have a 5 millisecond walk-off. That means they wait .05 seconds for a response from the destination address, if there is no response, they send the initial packet again. There is nothing unusual about that, but if you look at all of the packets, you see a .05 second interval between all of the packets. There is even a .05 second interval between packets from different source addresses. This is a huge indication that all the packets were sent from the same source.



```
10:10:03.816507 IP (tos 0x0, ttl 48, id 30084, len 40) 61.218.161.202.80 > 170.129.19.170.80:
10:10:08.786507 IP (tos 0x0, ttl 48, id 30366, len 40) 61.218.161.202.80 > 170.129.19.170.80:
10:10:13.826507 IP (tos 0x0, ttl 48, id 30662, len 40) 61.218.161.210.80 > 170.129.19.170.80:
10:10:18.856507 IP (tos 0x0, ttl 48, id 30943, len 40) 61.218.161.210.80 > 170.129.19.170.80:
10:10:23.856507 IP (tos 0x0, ttl 44, id 31290, len 40) 163.23.238.9.80 > 170.129.19.170.80: .
```

Figure 5

Another option that should be considered is that this could be a distributed scan. A distributed scan is scan conducted by several scanning sources all targeting one destination computer. It can be conducted by one person with several remote scanning computers or by a team of attackers. A distributed scan is usually hard to detect due to the varying TTL and times. I don't think that the above scan is a distributed scan due to similar TTL's and other similarities noted above.

One of the preceding factors (TTL, window size, IP length, source/destination ports and time) by itself is suspicious of IP spoofing, but when you put all of them together, it really solidifies the idea that all of offending packets were sent by one host spoofing several other addresses.

4. Description of attack:

Nmap is a network-scanning tool developed by Fyodor (Fyodor). It is probably the most popular network-scanning tool available. One of the options available when scanning a host or network is the "ack scan". An ack scan is used to map out firewall rulesets. The scan sends an ack packet to the ports that are specified by the user. If a rst comes back, then nmap knows that the port is open on the firewall and the ack packet made it to the host. If nothing comes back, then the port is closed (Fyodor). In all of the alerts that were logged by snort, the ack was the only flag set.

A search of the CVE website at http://cve.mitre.org for CVE's pertaining to nmap returned 11 results (10 are shown in Table 5, 1 didn't relate to nmap). None of the CVE’s directly pertained to ack scans, but they all relate to nmap or scanning that could be performed by nmap (Mitre).

Name	Description
CVE-2000-0324	pcAnywhere 8.x and 9.0 allows remote attackers to cause a denial of service via a TCP SYN scan, e.g. by nmap.
CVE-2000-0962	The IPSEC implementation in OpenBSD 2.7 does not properly handle empty AH/ESP packets, which allows remote attackers to cause a denial of service.
CVE-2001-0773	Cayman 3220-H DSL Router 1.0 allows remote attacker to cause a denial of service (crash) via a series of SYN or TCP connect requests.
CVE-2001-0896	Inetd in OpenServer 5.0.5 allows remote attackers to cause a denial of service (crash) via a port scan, e.g. with nmap –PO.
CAN-1999-0454	A remote attacker can sometimes identify the operating system of a host based on how it reacts to some IP or ICMP packets, using a tool such as nmap or queso.
CAN-1999-1373	FORE PowerHub before 5.0.1 allows remote attackers to cause a denial of service (hang) via a TCP SYN scan with TCP/IP OS fingerprinting, e.g. via nmap.
CAN-2002-0116	Palm OS 3.5h and possibly other versions, as used in Handspring Visor and Xircom products, allows remote attackers to cause a denial of service via a TCP connect scan, e.g. from nmap.
CAN-2002-0119	Alcatel Speed Touch Home ADSL Modem allows remote attackers to cause a denial of service (reboot) via a network scan with unusual packets, such as nmap with OS detection.
CAN-2002-0127	Netgear RP114 Cable/DSL Web Safe Router Firmware 3.26, when configured to block traffic below port 1024, allows remote attackers to cause a denial of service (hang) via a port scan of the WAN port.
CAN-2002-1322	Rational ClearCase 4.1, 2002.05, and possibly other versions allows remote attackers to cause a denial of service (crash) via certain packets to port 371, e.g. via nmap.

Table 5

5. Attack mechanism:

I was having trouble figuring out what to put in this section, so I referred to the Intrusion Signatures and Analysis Book (Northcutt, Cooper, Fearnow and Frederick). According to them, there are four basic questions that need to be answered here, so that’s exactly what I’ll do.

Is this a stimulus or response?

-The activity logged by snort is the stimulus. We are looking at nmap actively conducting an ack scan against a network.

What service is being targeted?

-All of the packets were sent to machines within the 170.129 network at port 80. The ack scan was specifically targeting web servers which are served up on port 80.

Does this service have known vulnerabilities or exposures?

-Yes, there are many know vulnerabilities for web servers. A search of the CVE database for "http" returned 436 records. Web servers vulnerabilities also top the SANS Top Vulnerabilities to Windows and is #3 on the Vulnerabilities to Unix list. The list can be found at http://www.sans.org/top20/.

Is this benign, an exploit, denial or service, or reconnaissance?

- The nmap ack scan that I have been analyzing appears to be reconnaissance. The scanner is looking for a host that will answer on port 80.

6. Correlations:

There are plenty of correlations for a nmap scan. I have already listed the CVE relating to nmap and scanning in section 4. Probably the best source of information relating to nmap is the nmap website at www.insecure.org. The website contains a wealth of knowledge with the documentation and nmap mailing lists. A definite must see for those interested in nmap or scanning.

A quick google of "SCAN nmap tcp" returned numerous results. One in particular is http://cert.uni-stuttgart.de/archive/intrusions/2003/01/msg00017.html. This mailing list post is very similar

7. Evidence of active targeting:

There is no indication that the scanner was specifically targeting hosts, but the 170.129 network was a definite target.

8. **Severity:**

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

Calculating the severity of the scan is difficult because of the source of the trace and the lack of knowledge of the network and related machines. Each value should be ranked on a scale from 1 (lowest) to 5 (highest). Make sure you discuss the threat on the criticality and lethality side of the equation and the defensive portions on the system countermeasures and network countermeasures side).

Criticality = 4 - The scanner was specifically looking for web servers. I don't know how important the web servers are to this origination, but they are pretty important in most, so I will assume the same here.

Lethality = 2 – Scanning is a reconnaissance action which may lead to more concentrated attacks later which is why I gave it a 2 vs. 1.

System countermeasures = 3 – Due to the source of this trace, I don't have much information of system countermeasures, so I gave it a middle of the road score. There were no responses to the ack scan, so either no web servers were scanned or they weren't allowed to answer.

Network countermeasures = 2 – Again, due to the source of the trace, I'm not sure of the layout of the network. I have to rate this area a 2, because I'm assuming that the scan is making its way past the perimeter defenses.

Severity = (4 + 3) – (3 + 2) or 2

9. Defensive recommendation:

Without detailed knowledge of the network and the organization, it's hard to make a recommendation. I would recommend a stateful firewall and an ACL on the router that only lets allows inbound web traffic to specified web servers.

A stateful firewall would keep track of outbound and inbound traffic only allowing in ack's only in response to a syn that originated from the internal network or on a case by case basis as defined in the ruleset.

10. Multiple choice test question:

According to Fyodor, an nmap ack scan is usually used for what specific purpose?

A) map out an organization's web servers
B) map out an organization's firewall ruleset
C) map out an organization's mail servers
D) map out an organization's IDS signatures

Answer: B

Results of posting to intrusions@incidents.org mailing list:

This detect was posted to the mailing list on February 27th., 2004. I had one reply from Ronny Rietveld.

Response received on February 28th, 2004:

Hi Tim,

You wrote that when 'rst comes back ... the port is open and accepting connections'. This is true in most cases, but when I send an 'ack' packet to a closed port I also get a 'rst' packet back. How can that be?

Can you also be in more detail on the part 'When an icmp unreachable message or nothing comes back, then port is closed'. ICMP unreachable messages tell you more about reachability of network, host, port, protocol, etc. With regards to TCP can the attacker expect information on port state from ICMP unreachable messages?

Ronny

I responded back to Ronny on March 3rd, 2004:

Ronny,

Thanks for your comments.

-->snip<--
... but when I send an 'ack' packet to a closed port I also get a 'rst' packet back. How can that be?
-->snip<--

You get a rst packet back for both a closed port and an open port. In the case of an open port, it is expecting a syn to start the three way handshake. When it sees an ack, it send back a rst, indicating that it has

*not established a proper three handshake. When an ack is sent to a
closed
port, the port sends back a rst indicating that it is not accepting any
connections.*

*-->snip<--
Can you also be in more detail on the part 'When an icmp unreachable
message
or nothing comes back, then port is closed'.... With regards to TCP can the
attacker expect information on port state from ICMP unreachable
messages?
-->snip<--*

Ronny’s final response was received on March 3rd, 2004:

Hi Tim,

*Thanks for the answers. First one is right on the spot. With the second
one I left a little comment. Good luck with your exams.*

Ronny

> -->snip<--
> Can you also be in more detail on the part 'When an icmp unreachable
> message
> or nothing comes back, then port is closed'.... With regards to TCP
can
> the
> attacker expect information on port state from ICMP unreachable
messages?
> -->snip<--
>
> There are several types of icmp unreachable messages according to
IANA
> (http://www.iana.org/assignments/icmp-parameters).
>
>   0 Net Unreachable
>   1 Host Unreachable
>   2 Protocol Unreachable
>   3 Port Unreachable
>   4 Fragmentation Needed and Don't Fragment was Set
>   5 Source Route Failed
>   6 Destination Network Unknown
>   7 Destination Host Unknown
>   8 Source Host Isolated
>   9 Communication with Destination Network is

Administratively
> Prohibited
>   10 Communication with Destination Host is Administratively
> Prohibited
>   11 Destination Network Unreachable for Type of Service
>   12 Destination Host Unreachable for Type of Service
>   13 Communication Administratively Prohibited
>   14 Host Precedence Violation
>   15 Precedence cutoff in effect
>
> If I sent an ack to a closed port, I could expect an imcp port unreachable
> message back. This information could help an attacker map out a router
> acl
> or firewall ruleset. By receiving an icmp port unreachable message back,
> the attacker knows that the port he sent traffic to is allowed through the
> router acl and the firewall.

True, ICMP port unreachable may help you to map out ACLs and rulesets, or may be useful to inverse map hosts on a network, but you will only see ICMP port unreachable messages as a response on UDP packets sent to
a closed UDP port. If you send an ACK to a closed TCP port, you get a RST back. Just as you described in your first answer.

Thanks for your response and help Ronny!

#2 Detect: WEBDAV:SEARCH-OVERFLOW

1. Source of Trace:

This trace was obtained from an intrusion detection sensor where I work on April 2, 2004. The sensor sniffs a subnet in the DMZ where we have some web servers. The line is tapped using a NetOptics 10/100 Tap. The configuration is displayed below.

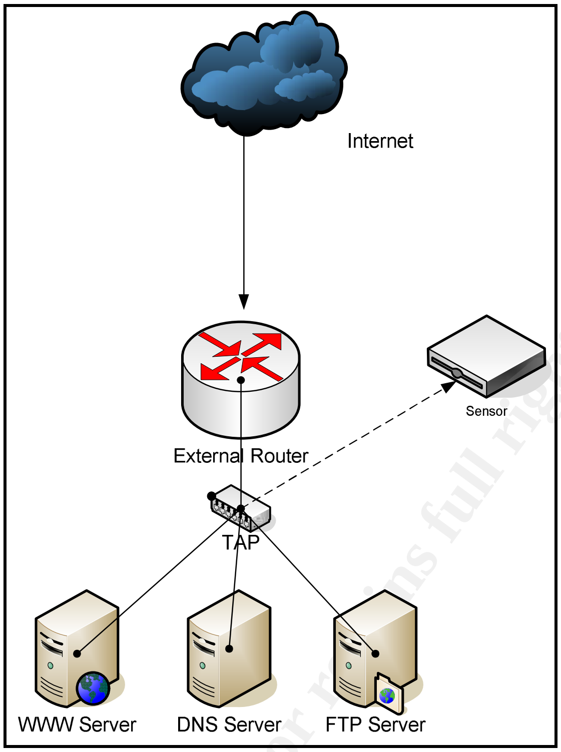
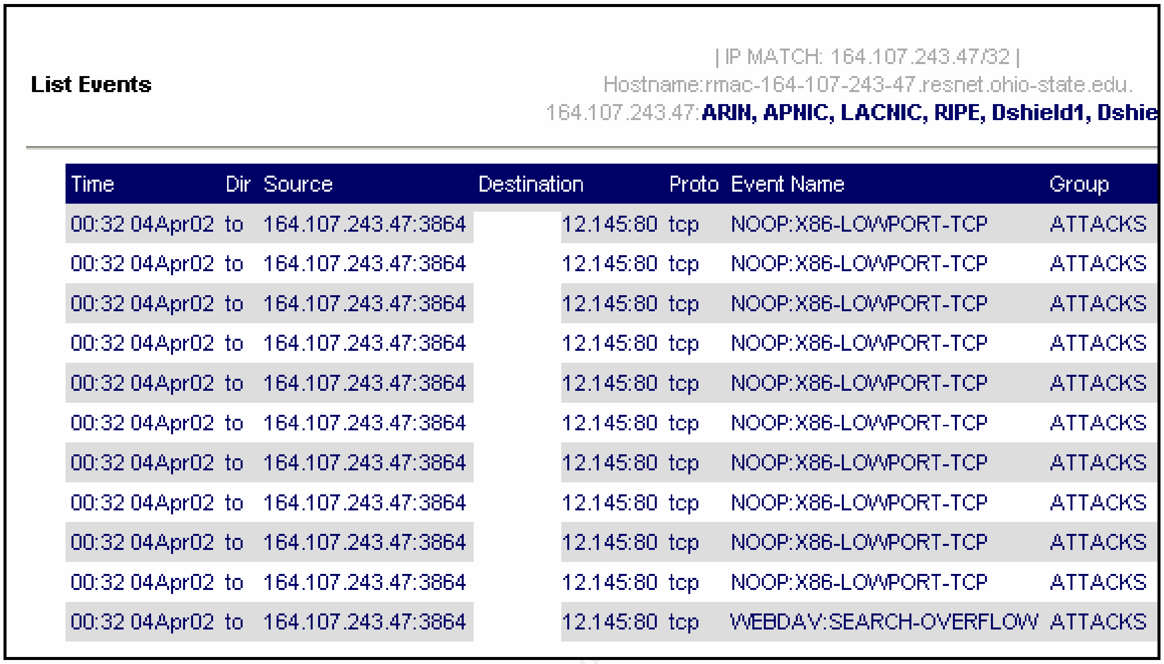


Figure 6

1. **Detect was generated by:**

   This detect was generated by Dragon IDS running version 6.2 with a signature set dated March 30th, 2004. The event that attracted my interest was the WEBDAV: SEARCH-OVERFLOW.



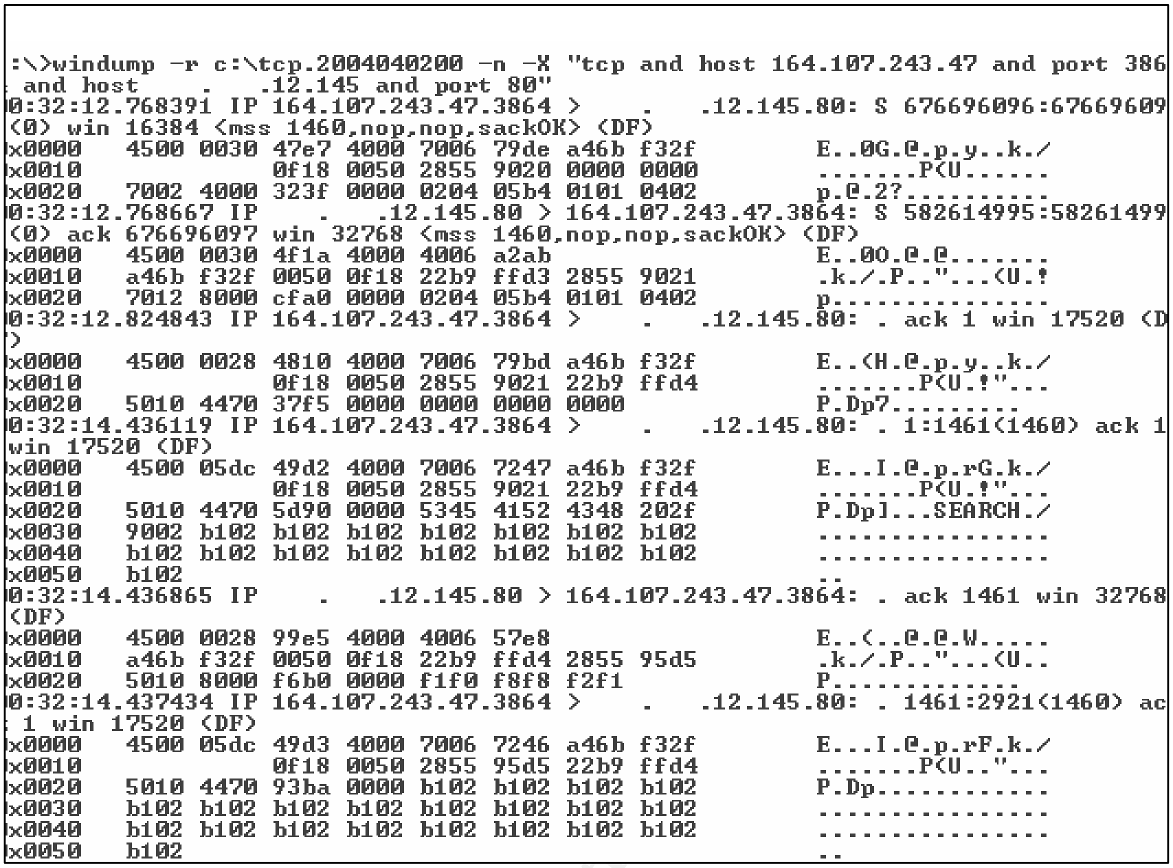
List Events

| IP MATCH: 164.107.243.47/32 |
Hostname:rmac-164-107-243-47.resnet.ohio-state.edu.
164.107.243.47:**ARIN, APNIC, LACNIC, RIPE, Dshield1, Dshie**

Time	Dir	Source	Destination	Proto	Event Name	Group
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	NOOP:X86-LOWPORT-TCP	ATTACKS
00:32 04Apr02	to	164.107.243.47:3864	12.145:80	tcp	WEBDAV:SEARCH-OVERFLOW	ATTACKS

Figure 7

Below is a sample of the tcpdump of the network traffic that generated these alerts.



```
:\>windump -r c:\tcp.2004040200 -n -X "tcp and host 164.107.243.47 and port 386
 and host     .    .12.145 and port 80"
0:32:12.768391 IP 164.107.243.47.3864 >     .    .12.145.80: S 676696096:67669609
(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
x0000   4500 0030 47e7 4000 7006 79de a46b f32f        E..0G.@.p.y..k./
x0010             0f18 0050 2855 9020 0000 0000        .......P(U......
x0020   7002 4000 323f 0000 0204 05b4 0101 0402        p.@.2?..........
0:32:12.768667 IP     .    .12.145.80 > 164.107.243.47.3864: S 58261499 5:5826149 9
(0) ack 676696097 win 32768 <mss 1460,nop,nop,sackOK> (DF)
x0000   4500 0030 4f1a 4000 4006 a2ab                  E..0O.@.@.......
x0010   a46b f32f 0050 0f18 22b9 ffd3 2855 9021        .k./.P.."...(U.!
x0020   7012 8000 cfa0 0000 0204 05b4 0101 0402        p...............
0:32:12.824843 IP 164.107.243.47.3864 >     .    .12.145.80: . ack 1 win 17520 (D
)
x0000   4500 0028 4810 4000 7006 79bd a46b f32f        E..(H.@.p.y..k./
x0010             0f18 0050 2855 9021 22b9 ffd4        .......P(U.!"...
x0020   5010 4470 37f5 0000 0000 0000 0000             P.Dp7.........
0:32:14.436119 IP 164.107.243.47.3864 >     .    .12.145.80: . 1:1461(1460) ack 1
win 17520 (DF)
x0000   4500 05dc 49d2 4000 7006 7247 a46b f32f        E...I.@.p.rG.k./
x0010             0f18 0050 2855 9021 22b9 ffd4        .......P(U.!"...
x0020   5010 4470 5d90 0000 5345 4152 4348 202f        P.Dp]...SEARCH./
x0030   9002 b102 b102 b102 b102 b102 b102 b102        ................
x0040   b102 b102 b102 b102 b102 b102 b102 b102        ................
x0050   b102                                           ..
0:32:14.436865 IP     .    .12.145.80 > 164.107.243.47.3864: . ack 1461 win 32768
(DF)
x0000   4500 0028 99e5 4000 4006 57e8                  E..(..@.@.W.....
x0010   a46b f32f 0050 0f18 22b9 ffd4 2855 95d5        .k./.P.."...(U..
x0020   5010 8000 f6b0 0000 f1f0 f8f8 f2f1             P.............
0:32:14.437434 IP 164.107.243.47.3864 >     .    .12.145.80: . 1461:2921(1460) ac
 1 win 17520 (DF)
x0000   4500 05dc 49d3 4000 7006 7246 a46b f32f        E...I.@.p.rF.k./
x0010             0f18 0050 2855 95d5 22b9 ffd4        .......P(U.."...
x0020   5010 4470 93ba 0000 b102 b102 b102 b102        P.Dp............
x0030   b102 b102 b102 b102 b102 b102 b102 b102        ................
x0040   b102 b102 b102 b102 b102 b102 b102 b102        ................
x0050   b102                                           ..
```

Figure 8

The rest of the trace contained more of the NOOP's.

The signature that triggered the alert is:

T D A S 10 10 W WEBDAV:SEARCH-OVERFLOW search/20 > 200

T – Protocol Field: TCP

D – Direction: Destination

A – Protected Networks: All traffic

S – Binary or Case Insensitive Search: Case Insensitive Search

10 – Dynamic Log: This tells Dragon how many session packets to log after the rule is triggered.

10 – Bytes to compare: How far into the packet to search for a signature match

W – Port: Usually a port number would be placed here, but in this case there are multiple port numbers, so they are defined in another configuration file. The "W" indicates which line in the configuration file to refer to when looking for port information. The ports that this signatures looks at are: 80, 3128 and 8080.

WEBDAV:SEARCH-OVERFLOW – Signature Name

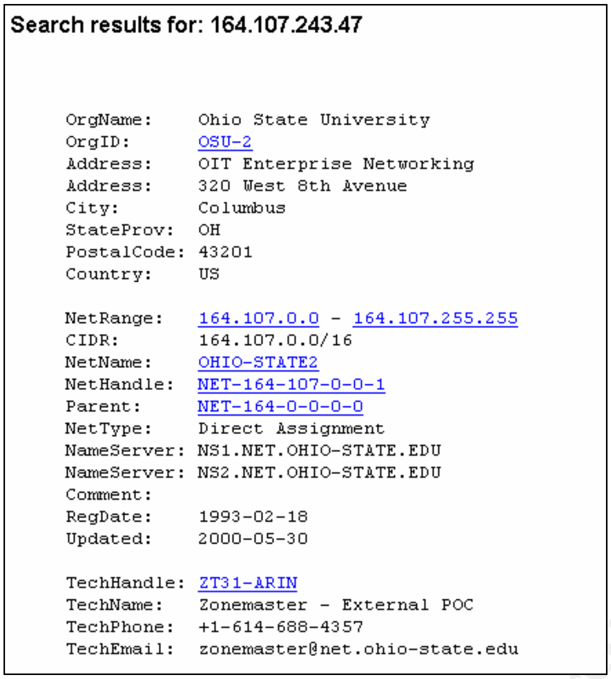
search/20 > 200 – Search String: This searches for the string "search/20". Then it searches the remaining data for a new line. If more than 200 characters are found an event is triggered. This signature can generate false positives, but looking for a search string with more than 200 characters filters out quite a bit of the possible false positives.

2. **Probability the source address was spoofed:**

The possibility that the source address is spoofed is pretty low. If you look at Figure 3 above, you can see that the attacker completes a tcp three-way handshake with the web server before he sends the exploit.

To help confirm my suspicion that the IP address is not spoofed, I trace routed to the source address to see if the TTL's matched. For this trace route I was unable to run the trace route from the same location that the network trace is taken from, but I'm in the same general location using the same ISP. With that in mind, I was expecting to see a small variation in the results, but nothing too unusual. The TTL value for offending packets was 112 and when I trace routed back to the host, I received a value of 106. As I stated above this variation is due to my change in location and normal Internet routing fluctuations.

I queried ARIN and the record that was returned is displayed below in Figure 4, which matches the reverse dns record returned to me what I ran traceroute (rmac-164-107-243-47.resnet.ohio-state.edu).



Search results for: 164.107.243.47

```
OrgName:    Ohio State University
OrgID:      OSU-2
Address:    OIT Enterprise Networking
Address:    320 West 8th Avenue
City:       Columbus
StateProv:  OH
PostalCode: 43201
Country:    US

NetRange:   164.107.0.0 - 164.107.255.255
CIDR:       164.107.0.0/16
NetName:    OHIO-STATE2
NetHandle:  NET-164-107-0-0-1
Parent:     NET-164-0-0-0-0
NetType:    Direct Assignment
NameServer: NS1.NET.OHIO-STATE.EDU
NameServer: NS2.NET.OHIO-STATE.EDU
Comment:
RegDate:    1993-02-18
Updated:    2000-05-30

TechHandle: ZT31-ARIN
TechName:   Zonemaster - External POC
TechPhone:  +1-614-688-4357
TechEmail:  zonemaster@net.ohio-state.edu
```

Figure 9

3. Description of attack:

The World Wide Web Distributed Authoring and Versioning (WebDav) protocol is an extension to the HTTP/1.1 protocol that allows clients to perform remote web content authoring operations (RFC 2518). In other WebDav is a protocol for manipulating documents via the Internet (WebDav FAQ).

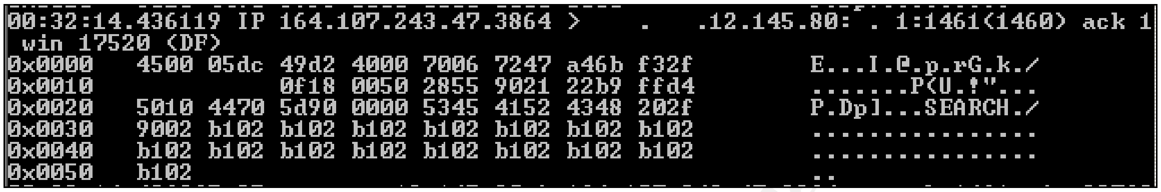
Some of the major features of WebDav are (WebDav FAQ):

- Locking - long-duration exclusive and shared write locks prevent overwrite problems.
- Properties - XML properties provide storage for arbitrary metadata, such as a list of authors on Web resources.
- Namespace Manipulation – support for copy and move operations as a web site evolve.

Microsoft implemented WebDav in Windows 2000, Windows XP and Windows NT 4.0. WebDav is automatically loaded with IIS 5.0 on Windows 2000. This attack exploits an unchecked buffer in a component of the WebDav implementation, ntdll.dll.

An attacker could exploit this vulnerability by sending an unusually long request to the web server. If the attack is successful, the attacker could execute commands on the server with the same privileges as IIS (Microsoft).

In this case, the attacker sends a SEARCH request followed by about 16,000 bytes worth on NOOP's in an attempt to overrun the buffer.



```
00:32:14.436119 IP 164.107.243.47.3864 >    .   .12.145.80: . 1:1461(1460) ack 1
 win 17520 (DF)
0x0000   4500 05dc 49d2 4000 7006 7247 a46b f32f        E...I.@.p.rG.k./
0x0010             0f18 0050 2855 9021 22b9 ffd4        ........P(U.!"...
0x0020   5010 4470 5d90 0000 5345 4152 4348 202f        P.Dp]...SEARCH./
0x0030   9002 b102 b102 b102 b102 b102 b102 b102        ................
0x0040   b102 b102 b102 b102 b102 b102 b102 b102        ................
0x0050   b102                                           ..
```

Figure 10

The web server is running apache 2.x, so it is not susceptible to this attack.

4. Attack mechanism:

As with the Detect #1, I decided to answer the four questions from the Intrusion Signatures and Analysis Book (Northcutt, Cooper, Fearnow and Frederick) again.

Is this a stimulus or response?

-The traffic that triggered the Dragon alert WEBDAV: SEARCH-OVERLFOW is a stimulus. The signature looks for the word "SEARCH" followed by more than 200 characters.

What service is being targeted?

- Web servers are being targeted. More specifically, Windows 2000 servers running IIS 5.0 are being targeted.

Does this service have known vulnerabilities or exposures?

-Yes, there are many know vulnerabilities for web servers. A search of the CVE database for "http" returned 436 records. Web servers vulnerabilities also top the SANS Top Vulnerabilities to Windows and is #3 on the Vulnerabilities to Unix list. The list can be found at http://www.sans.org/top20/ .

Is this benign, an exploit, denial or service, or reconnaissance?

- This is an exploit. As stated in section 3, this is an exploit of a vulnerable windows file found in Windows 2000 IIS 5.0 WebDav code.

5. Correlations:

This attack is pretty old, so there was no shortage of correlation when I searched for some. Here is a link to a posting on the intrusions mailing list http://www.dshield.org/pipermail/intrusions/2003-May/007624.php. While searching for correlations, I ran across a nice analysis of the exploit by Eric Hines. http://www.fatelabs.com/library/fatelabs-ntdll-analysis.pdf

More information on the WebDav buffer overflow vulnerability can be found at the following websites:

Microsoft's description of the vulnerability as well as links to patches and suggested "workarounds" (Microsoft). http://www.microsoft.com/technet/security/bulletin/MS03-007.mspx

Bugtraq's analysis of the vulnerable as well links to some exploits and suggested solutions (Bugtraq). http://securityfocus.com/bid/7116/solution/

Here is the CVE for the WebDav search-overflow. The CVE also lists a ton more references to look into if needed (Mitre). http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0109

6. Evidence of active targeting:

There is no evidence that the attacker targeted our network, besides the WebDav alerts in the IDS. There is also no evidence that the attacker performed any reconnaissance before launching this attack, because he did not just target IIS. The web server that this exploit was targeted against is running Apache 2.x, which indicates that the attacker only looked for servers that answered up on port 80 and it didn't look specifically for IIS 5.0 web servers.

7. Severity:

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

Criticality = 5 – The targeted system is a very critical server. It is primary used for getting safety information out to a specific audience.

Lethality = 5 – If the exploit would have been successful, the attacker would have been able to run “code of choice”

System countermeasures = 5 – The system that was attacked is a very secure system. All patches and updates are applied, a HIDS is installed and monitored and logs are reviewed daily.

Network countermeasures = 4 – All the necessary network defensives are in place. I gave this a 4 instead of a 5, because there is now way to positively secure all network connections.

Severity = (5 + 5) – (5 + 4) or 1

8. Defensive recommendation:

There are a couple of defensive measures that I would recommend for a web server that needs to withstand similar buffer overflows.

1. Remove all unnecessary software and disable all unnecessary services.
2. Apply all security patches and operating system updates
3. Properly secure applications that are running on the server. If you are running IIS, use URLscan and similar tools to help lockdown the service.

These are just a couple recommendations. There are plenty of more. Some resources for you to use when securing you web server are listed below.

-Microsoft’s new security initiative includes a nice new security website. If you haven’t checked it out, I suggest going there, especially if you administer Windows. They have some checklists listed there to help you secure some of their products. http://www.microsoft.com/security/guidance/checklists/default.mspx

-The SANS SCORE website is another nice resource for you. They list recommended security settings a several different operating systems. There are security benchmarking tools available for download too. http://www.sans.org/score/.

-The SANS Reading Room has a section dedicated to Windows, Windows 2000, web servers and many other topics for you. Another must check out for you. http://www.sans.org/rr/

If those don't have enough information for you, you can always Google for IIS security or web server security.

9. **Multiple choice test question:**

Which versions of Windows contain the vulnerable version of ntdll.dll

a) Win98, WinME and WinXP
b) WinNT, Win2k and WinXP
c) Win95, Win98 and WinME
d) WinNT and Win2K

The correct answer is B. All three versions contain the vulnerable version of ntdll.dll. The only known attack vector is through IIS 5.0 with WebDav, but that doesn't mean there won't be more.

#3 Detect: IIS:CMD.EXE

1. **Source of Trace:**

This trace was obtained from an intrusion detection sensor where I work on April 2, 2004. The sensor sniffs a subnet in the DMZ where we have some web servers. The line is tapped using a NetOptics 10/100 Tap. The configuration is displayed below.

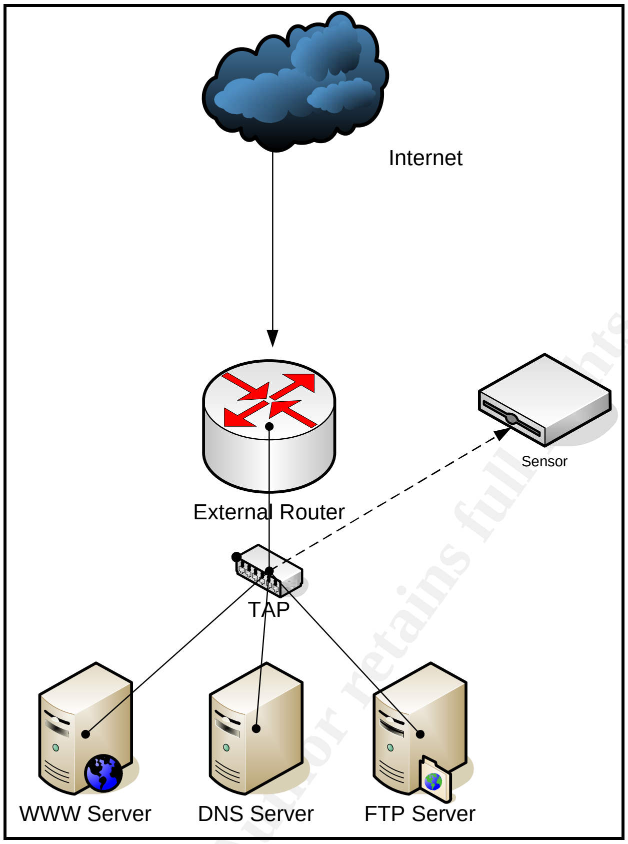
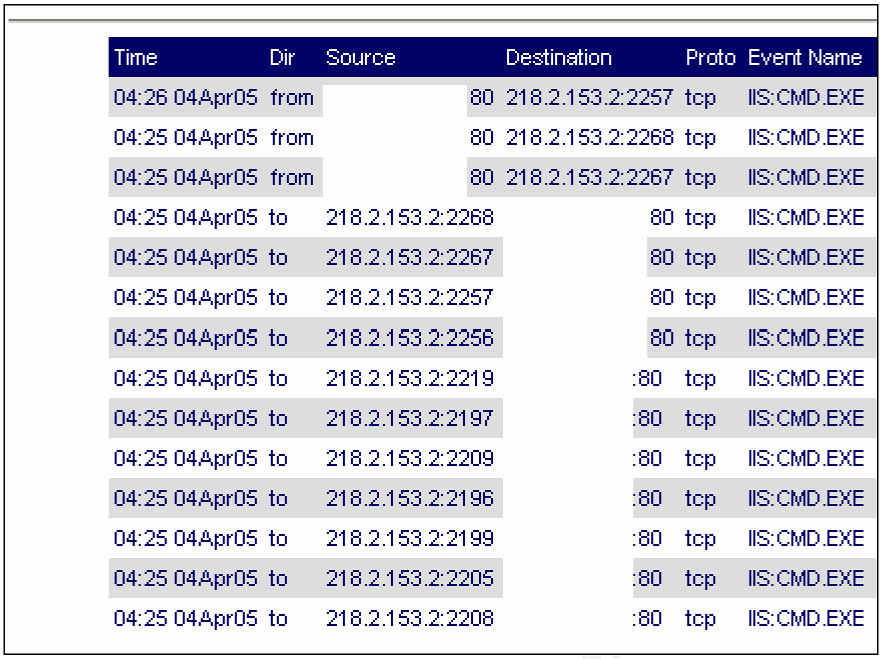


Figure 11

2. Detect was generated by:

This detect was generated by Dragon IDS running version 6.2 with a signature set dated March 30th, 2004. The event that attracted my interest was the IIS:CMD.EXE.



Time	Dir	Source	Destination	Proto	Event Name
04:26 04Apr05	from	80	218.2.153.2:2257	tcp	IIS:CMD.EXE
04:25 04Apr05	from	80	218.2.153.2:2268	tcp	IIS:CMD.EXE
04:25 04Apr05	from	80	218.2.153.2:2267	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2268	80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2267	80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2257	80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2256	80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2219	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2197	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2209	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2196	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2199	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2205	:80	tcp	IIS:CMD.EXE
04:25 04Apr05	to	218.2.153.2:2208	:80	tcp	IIS:CMD.EXE

Figure 12

The signature and explanation that triggered the alerts is listed below.

T D A S 15 300 W IIS:CMD.EXE /2fwinnt/2fsystem32/2fcmd.exe/3f

T – Protocol Field: TCP

D – Direction: Destination

A – Protected Networks: All traffic

S – Binary or Case Insensitive Search: Case Insensitive Search

15 – Dynamic Log: This tells Dragon how many session packets to log after the rule is triggered.

300 – Bytes to compare: How far into the packet to search for a signature match

W – Port: Usually a port number would be placed here, but in this case there are multiple port numbers, so they are defined in another configuration file. The "W" indicates which line in the configuration file to refer to when looking for port information. The ports that this signatures looks at are: 80, 3128 and 8080.

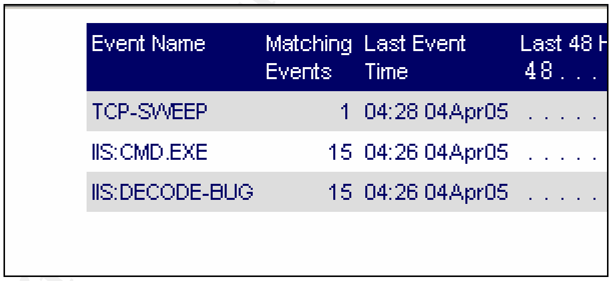
IIS:CMD.EXE – Signature Name

/2fwinnt/2fsystem32/2fcmd.exe/3f – Search String: This searches for the string “/winnt/system32/cmd.exe?”.

The offending packet is displayed below:

```
04:25:52.816524 218.2.153.2.2267 > xxx.xxx.2.121.http: P 1:60(59) ack 1 win 17520 (DF)

0x0000   4500 0063 02b9 4000 6f06 ee87 da02 9902       E..c..@.o.......

0x0010   xxxx xxxx  08db 0050 3122 2c84 854c 7b85       ...y...P1",..L{.

0x0020   5018 4470 8c51 0000 4745 5420 2f73 6372       P.Dp.Q..GET./scr

0x0030   6970 7473 2f2e 2e25 3235 3563 2532 3535       ipts/..%255c%255

0x0040   632e 2e2f 7769 6e6e 742f 7379 7374 656d      c../winnt/system

0x0050   3332                                         32
```

This event is just one of many from the same source address. Figure 3 is a shot of all the alerts that correlate to 218.2.153.2.



Event Name	Matching Events	Last Event Time	Last 48 H 48 . . .
TCP-SWEEP	1	04:28 04Apr05	. . . . .
IIS:CMD.EXE	15	04:26 04Apr05	. . . . .
IIS:DECODE-BUG	15	04:26 04Apr05	. . . . .

Figure 13

TCP-SWEEP is the alert triggered when one source host attempts connections to multiple destination hosts on the same port. In this case our attacker has attempted connections with 15 live servers on port 80.

IIS:DECODE-BUG is another alert that was triggered by the same network traffic sent from the attacker. The signature for this alert is:

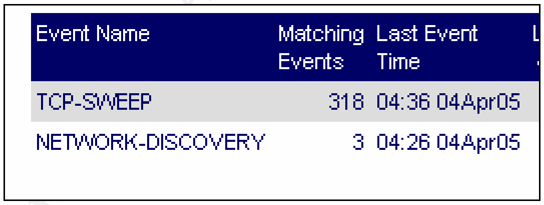
T D A B 15 0 W IIS:DECODE-BUG %255c

“%255c” is what triggered this alert. Here is the offending packet that triggered the IIS:DECODE-BUG alert.

```
04:25:52.816524 218.2.153.2.2267 > xxx.xxx.2.121.http: P 1:60(59) ack 1 win 17520 (DF)

0x0000   4500 0063 02b9 4000 6f06 ee87 da02 9902       E..c..@.o.......

0x0010   xxxx xxxx  08db 0050 3122 2c84 854c 7b85      ...y...P1",..L{.

0x0020   5018 4470 8c51 0000 4745 5420 2f73 6372       P.Dp.Q..GET./scr

0x0030   6970 7473 2f2e 2e25 3235 3563 2532 3535       ipts/..%255c%255

0x0040   632e 2e2f 7769 6e6e 742f 7379 7374 656d      c../winnt/system

0x0050   3332                                         32
```

As you can tell, the offending packet for both IIS:CMD.EXE and IIS:DECODE-BUG are the same. That's the reason for both alerts.

Review of an external sensor shows even more activity from the source address. Figure 4 shows all the alerts that were blocked by the border router.



Event Name	Matching Events	Last Event Time
TCP-SWEEP	318	04:36 04Apr05
NETWORK-DISCOVERY	3	04:26 04Apr05

Figure 14

NETWORK-DISCOVERY is an alert for ICMP Ping Sweeps. The attacker attempted to sweep 3 different subnets in our entire Class B address space, before he figured out that the border router blocks all ICMP traffic.

TCP-SWEEP is triggered when Dragon sees one source host make connections to multiple destination hosts on the same destination port. In this case our attacker attempted connections to hundreds of hosts in our Class B address space on port 80. The border router blocks network traffic to non-existent subnets within our organization's Class B address space.

If we break down the timeline, we see some typical patterns emerge.

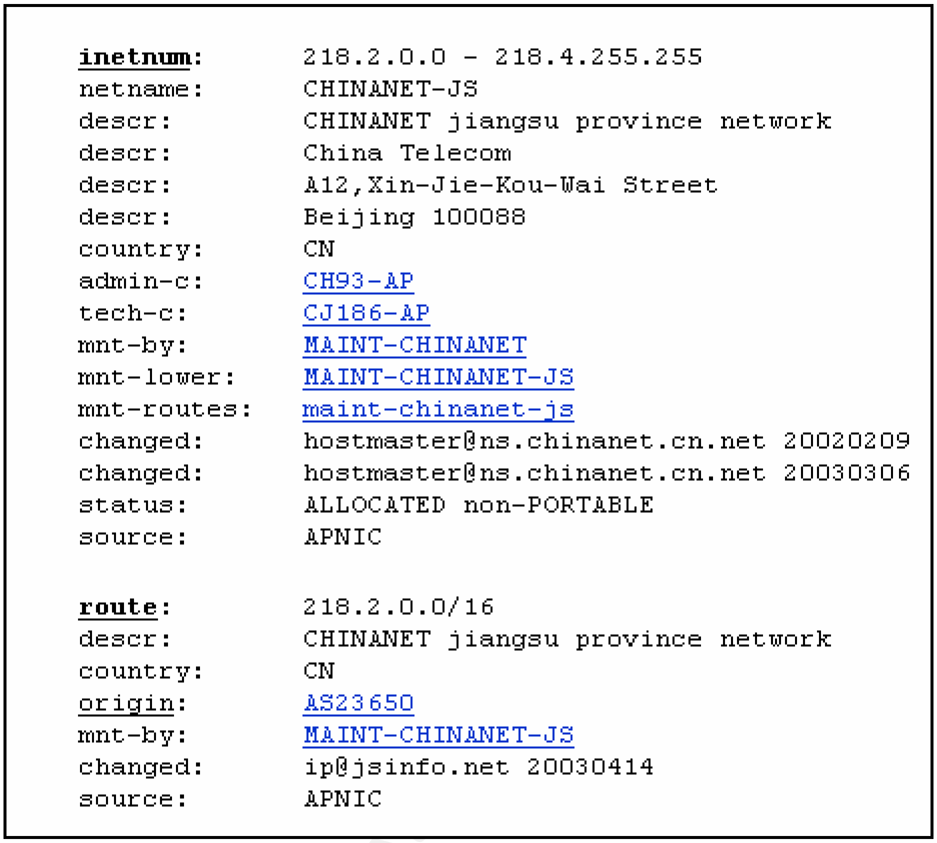
3. 04:24 – Network Discovery – ICMP ping sweeps (blocked by router)
4. 04:25-04:36 – TCP-SWEEP – TCP connect scans attempting connections to Port 80
5. 04:26-04:28 – IIS:CMD.EXE – Connections were made to live servers on port 80 and exploit sent

In step a. the attacker is doing some basic reconnaissance looking for live hosts. There are only 3 alerts logged for this, because the attacker quickly realized that ICMP was being blocked. Step b. shows the attacker conducting some TCP connect scanning for port 80. Lastly in step c. the attacker has found live hosts that respond on port 80, so he attempts his exploit.

3. **Probability the source address was spoofed:**

The possibility that the source address is spoofed is low. The attacker completes the three-way handshake with all of the web servers before the exploit is sent.

A query of ARIN showed that the source address is handled by APNIC. I then queried APNIC, which returned the following record displayed in Figure 5.



```
inetnum:      218.2.0.0 - 218.4.255.255
netname:      CHINANET-JS
descr:        CHINANET jiangsu province network
descr:        China Telecom
descr:        A12,Xin-Jie-Kou-Wai Street
descr:        Beijing 100088
country:      CN
admin-c:      CH93-AP
tech-c:       CJ186-AP
mnt-by:       MAINT-CHINANET
mnt-lower:    MAINT-CHINANET-JS
mnt-routes:   maint-chinanet-js
changed:      hostmaster@ns.chinanet.cn.net 20020209
changed:      hostmaster@ns.chinanet.cn.net 20030306
status:       ALLOCATED non-PORTABLE
source:       APNIC

route:        218.2.0.0/16
descr:        CHINANET jiangsu province network
country:      CN
origin:       AS23650
mnt-by:       MAINT-CHINANET-JS
changed:      ip@jsinfo.net 20030414
source:       APNIC
```

Figure 15

4. Description of attack:

IIS:CMD.EXE is a general signature that triggers whenever someone tries to access “cmd.exe”. Not too much information there, for us to have a bigger picture of the attack, we have to look at the IIS:DECODE-BUG alert too. This alert triggers whenever it sees “%255c”. Again, not too much, but put that together with the first alert and we get:

“255c%255c../winnt/system32/cmd.exe?/”.

If you look at the offending packet shown above, you will see the whole command that was sent to our web server, which was:

“GET /scripts/..%255c%255c../winnt/system32/cmd.exe?/c+dir”

Now, we have enough information to figure exactly what attack is being used against us.

This attack is targeting web servers and more specifically it is exploiting the web server folder traversal vulnerability found in IIS 4/5 web servers. This vulnerability allows attackers to execute arbitrary commands by encoding the “dot dot” and “\” characters twice. If an attacker succeeds in running their code of choice, they will be able to run it with the IUSR_machinename account privileges. This account is in the Everyone group by default. Some well-known worms that have used this vulnerability were Code Red and Nimda. Since this is an older attack, there are a ton of resources out there. Some of the better web sites I found are listed below.

-Security Focus

-ISS:X-Force Alerts and Advisories

-CVE-2001-0333

-US-CERT

-CERT

-Microsoft Security Bulletin

-SANS.org

Unicode is used to encode and decode the “dot dot” and “\” characters. This allows the malformed URL to pass IIS security checks. In this example, IIS runs the following URL through its security checks to check for directory traversals to traverse outside the normal inetpub folder.

```
GET /scripts/..%255c%255c../winnt/system32/cmd.exe?/c+dir
```

The above URL passes the security checks, because they are looking for “../” not “/..\%255c../”. When IIS decodes Unicode from the above URL, it becomes:

```
GET /scripts/..\%255c../winnt/system32/cmd.exe?/c+dir
```

The URL is sent through security checks again where it is decoded into:

```
 GET /scripts/..\\../winnt/system32/cmd.exe?/c+dir
```

Again this passes the security checks and the attacker gets a directory listing of the c drive and life is good, right? If you’re the attacker, if not, then your troubles have just started.

To manually decode Unicode characters, everything after the % character is hexadecimal. In the above example, “%255c” becomes “%5c”, then that becomes “\”.

Microsoft has released a patch to fix this problem; which can be found at:

http://www.microsoft.com/technet/security/bulletin/MS01-026.mspx

5. Attack mechanism:

As with the Detect #1 and #2, I decided to answer the four questions from the Intrusion Signatures and Analysis Book again (Northcutt, Cooper, Fearnow and Frederick).

Is this a stimulus or response?

-The network traffic that triggered our two alerts is a stimulus. The response can be seen if you look at the complete session. See below for the one server's response. This response is from Apache 2.x running on Windows 2000. Notice the second part of the response, it tells the attacker exactly what version of Apache it is and what version on OpenSSL is running too. The exploit didn't work, but our server gave up some nice information to the attacker.

< p > The requested URL /scripts/..%5c%5c../winnt/system32/cmd.exe was not found on this server. < /p >

< address > Apache/2.0.47 (Win32) mod_ssl/2.0.47 OpenSSL/0.9.7c Server at www.mycompany.coml Port 80 < /address >

Since there were several types of web servers that this exploit was ran against, there were several different types of responses, but all of the servers returned error pages.

What service is being targeted?

- Web servers are being targeted. More specifically, Windows 2000 servers running IIS 4/5 are being targeted.

Does this service have known vulnerabilities or exposures?

-Yes, there are many know vulnerabilities for web servers. A search of the CVE database for "http" returned 436 records. Web servers vulnerabilities also top the SANS Top Vulnerabilities to Windows and is #3 on the Vulnerabilities to Unix list. The list can be found at http://www.sans.org/top20/ .

Is this benign, an exploit, denial or service, or reconnaissance?

- This is an exploit. This exploit is targeting Windows servers running an unpatched version of IIS 4 and 5.

6. **Correlations:**

   Information on the web directory transversal vulnerability can be found on many web sites. The better ones are listed in section 3. The web site that I found the most useful was the SANS IDS FAQ (SANS FAQ).

   The SANS IDS FAQ broke down the directory traversal attack to a very basic level. It described in depth how IIS security checks worked and how they are defeated using this technique. This FAQ also helped me write the multiple-choice question (hint, hint).

7. **Evidence of active targeting:**

   The attacker specifically targeted web servers. The attack was not limited to web servers running IIS 4/5, but all servers that responded on port 80. As I stated in section 2, the attacker tried to ping sweep our network, but all attempts were blocked by the border router. There is also no evidence that the attacker attempting any earlier reconnaissance against our networks.

8. **Severity:**

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

   Criticality = 5 – Web servers were targeted and they perform some critical services for our organization.

   Lethality = 5 – Even though the attack specifically targeted IIS servers and we only had 2 out of 15 running IIS, the end result if successful is root access to a web server.

   System countermeasures = 5 – All web servers had all of the latest patches and updates installed. Logs are reviewed daily and HIDS are installed on all publicly accessed servers.

   Network countermeasures = 4 – All the necessary network defensives are in place. I gave this a 4 instead of a 5, because there is now way to positively secure all network connections.

   Severity = (5 + 5) – (5 + 4) or 1

9. **Defensive recommendation:**

During this attack, all attempts to exploit the web servers failed. This is due to properly patched systems. Some general recommendations to secure your web server are listed below:

1. Remove all unnecessary software and disable all unnecessary services.

2. Apply all security patches and operating system updates

3. Properly secure applications that are running on the server. If you are running IIS, use URLscan and similar tools to help lockdown the service.

One thing I specifically noticed with this alert is that all of the Apache 2.x web servers gave out a lot more information then we want known when an error page was returned.

< address > Apache/2.0.47 (Win32) mod_ssl/2.0.47 OpenSSL/0.9.7c Server at www.mycompany.coml Port 80 < /address >

I recommend that you change the default error message that is returned and create your own custom error message. Detailed instructions can be found at Apache's website. Some more links to secure your apache web server can be found at the following websites:

-http://www.securityfocus.com/infocus/1694

-http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/security-guide/s1-server-http.html

-http://www.sans.org/rr/

10. **Multiple choice test question:**

Which answer decodes the URL correctly?

/msadc/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe?/c+dir+c:\

a) /scripts/..cd../..cd../..cd../winnt/system32/cmd.exe?/c+dir+c:\
b) /scripts/……/……/……/winnt/system32/cmd.exe/c+dir+c:\
c) /scripts/../../../../../../winnt/system32/cmd.exe/c+dir+c:\

d) /scripts/..\../..\../..\../winnt/system32/cmd.exe/c+dir+c:\

The correct answer is D. You have to decode the %35 and %63 first. It then becomes %5c which then becomes \.

References:

Snort.org. "The Open Source Network Intrusion Detection System." URL: www.snort.org (18 May 2004).

Passive OS Fingerprinting T-Shirt. SANS School Store. URL: https://store.sans.org/store_item.php?item=79 (18 May 2004)

Northcutt, Stephen and Judy Novak. Network Intrusion Detection Third Edition. New Riders Publishing, 2002.

Fyodor. "Insecure.org." URL: www.insecure.org (18 May 2004).

Mitre. "Common Vulnerabilities and Exposures." URL: http://cve.mitre.org (18 May 2004).

Northcutt, Stephen, Mark Cooper, Matt Fearnow and Karen Frederick. Intrusions Signatures and Analysis. New Riders Publishing, 2001.

ARIN. "American Registry for Internet Numbers." URL: www.arin.net (18 May 2004).

Dragon IDS. "Enterasys Intrusion Defense." URL: www.enterasys.com/products/ids (18 May 2004).

Microsoft Corporation. "Microsoft Security Bulletin MS03-007." URL: http://www.microsoft.com/technet/security/bulletin/MS03-007.mspx (18 May 2004).

Bugtraq. "Security Focus - Microsoft Windows ntdll.dll Buffer Overflow Vulnerability." URL: http://securityfocus.com/bid/7116/solution/ (18 May 2004).

APNIC. "Asia Pacific Network Information Centre." URL: www.apnic.net (18 May 2004).

Brannen, Andrew. "Unicode Vulnerability" – How & Why?" SANS Institute. http://www.sans.org/rr/papers/60/458.pdf (18 May 2004).

RFC 2518. HTTP Extensions for Distributed Authoring -- WEBDAV

Section 3

Analyze this

Executive Summary

Introduction

The following section is an analysis of five days worth of logs from an unknown University. The log files were obtained from http://www.incidents.org/logs. A Snort Intrusion Detection Sensor (IDS) generated the logs. Snort is an open source IDS that was created by Marty Roesch (Snort). More information on Snort can be found at www.snort.org. There are three different types of log files that will be discussed here. They are:

- Alert Logs
- Out of Specifications (OOS) Logs
- Scan Logs

The Alert log is a listing of all alerts generated by Snort. Alerts are triggered by network traffic that meets a certain pattern. For example if we told Snort to look for the pattern "ABC", every time "ABC" passed through the sensor, an alert would be generated. Keep in mind, the signatures that Snort has loaded can and usually are a lot more complex than "ABC". The OOS Log contains a listing of network traffic that is abnormal. Abnormal traffic is network traffic that does not meet certain criteria. Most entries in the OOS logs were generated due to invalid flag combinations. More information on valid flag combinations can be found in RFC 793 (RFC) It also appears that any traffic the University staff wanted to keep an eye on is logged here. The scan logs contain a listing of scanning that was intercepted by the Snort sensor. Scanning is one computer making connections to multiple computers on one port or one computer making connections to one computer on many ports.

Overall Network Health

Overall, the network has some major problems that demand immediate attention. I noticed all types of events that would cause many security analysts to grimace. There appeared to be machines participating in botnets, worms, viruses and Trojans running rampant and all types of network scans. This activity originated both from inside the University's network and from outside. These problems might seem overwhelming, but I think a couple of quick changes to the existing security architecture would help clean up the network a lot.

IDS tuning, Firewall rules, in-line virus scanners and patch management will go a long way to cleaning up this network.

IDS tuning will allow security analysts to "weed" through all of the network traffic to find illegal activity on the networks more efficiently. Currently the IDS's are clogged with irrelevant alerts; for example, MY.NET.30.4 and MY.NET.30.3 activity. If these are relevant alerts, I recommend using Snort's threshold feature on these and similar alerts to help unclutter the IDS data.

Firewall rules need to be tightened down. If there is no valid reason to allow connections to port 515 from outside of the network, then this port and others like it should be blocked at the firewall. I recommend a firewall review board to discuss University requirements and help determine the best policy for the firewall rules.

There are several alerts that are related to different worms and viruses. By adding some in-line anti-virus scanners, this would help cut-down on the virus problem within the network. You should set up the network to force all of a specific subnet or traffic type to be checked by the virus scanner without any user interaction.

And lastly patch management will help close a lot of the vulnerabilities that many worms and viruses target these days. My recommendation is to treat computers not managed by the University as untrusted. Force these computers to use some type of end point security where they must logon to some type of server where their machine is checked for current patches and A/V definitions. If the computer is not up to date, force the user to patch the machine before access is granted to University resources, in most cases, the University's Internet connection.

Report Specifics

In each section, I will break out the “Top Talkers”. These talkers will be broken out as follows:

- Alerts Log
    - Top 10 source IP addresses
    - Top 10 destination IP addresses
    - Top 10 destination ports
    - Top 5 alerts
- Scans Log
    - Top 10 scanning sources
    - Top 10 scanned destinations
    - Top 10 scanned ports
- OOS Log
    - Top 10 source IP addresses
    - Top 10 destination IP address
    - Top 10 destination ports

Listing of log files used:

Alert Log	Scan Log	OOS Log
Alert.040318.gz	scans.040318.gz	oos_report_040318.txt
Alert.040319.gz	scans.040319.gz	oos_report_040319.txt
Alert.040320.gz	scans.040320.gz	oos_report_040320.txt
Alert.040321.gz	scans.040322.gz	oos_report_040321.txt
Alert.040322.gz	scans.040325.gz	oos_report_040322.txt

The files scans.040321.gz and scans.040324.gz were useable due to corrupted gzipped files.

MY.NET was substituted with 123.456 to allow SnortSnarf processing.

Alert Log

In this section, the Alert logs are analyzed. The primary tool used to perform this analysis was SnortSnarf available from Silicon Defense at http://www.silicondefense.com/software/snortsnarf/ (Silicon Defense). Perl scripts found in Peter Van Oosterom's GCIA practical were also used (Oosterom).

There were 821,452 alerts logged in between March 18th and March 22. Of those alerts, 713,710 were spp_portscan alerts. SnortSnarf doesn't process the spp_portscan alerts due to their formatting, so those alerts have been stripped out and the remaining alerts were processed by SnortSnarf. (Port Scans are analyzed in the Port Scan Section) The results are displayed below in descending order of number of alerts.

Signature	# Alerts	# Sources	# Dest.
MY.NET.30.4 activity	60738	292	1
MY.NET.30.3 activity	24920	182	1
EXPLOIT x86 NOOP	7970	723	543
SMB Name Wildcard	4266	1	1
Connect to 515 from outside	4255	4	3
High port 65535 tcp - possible Red Worm - traffic	2060	66	34
Null scan!	1780	115	84
[UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.	1136	57	64
[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC	1039	1	1
NMAP TCP ping!	753	201	70
High port 65535 udp - possible Red Worm - traffic	507	82	18
Possible trojan server activity	347	29	21
Incomplete Packet Fragments Discarded	323	111	98
TFTP - Internal TCP connection to external tftp server	286	5	9
SUNRPC highport access!	166	26	27
ICMP SRC and DST outside network	165	29	49
External RPC call	156	2	121
TCP SRC and DST outside network	128	30	43
IRC evil - running XDCC	112	1	1
[UMBC NIDS] External MiMail alert	106	19	1
SMB C access	85	17	5
Connect to 515 from outside	83	3	3
FTP passwd attempt	78	64	1

[UMBC NIDS IRC Alert] Possible drone command detected.	58	2	9
TFTP - Internal UDP connection to external tftp server	48	6	3
TFTP - External TCP connection to internal tftp server	48	10	18
EXPLOIT x86 setuid 0	41	33	24
Tiny Fragments - Possible Hostile Activity	34	1	1
[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.	31	5	3
RFB - Possible WinVNC - 010708-1	31	6	6
EXPLOIT x86 setgid 0	24	19	17
Attempted Sun RPC high port access	16	5	2
EXPLOIT x86 NOPS	16	2	15
DDOS shaft client to handler	14	5	3
External FTP to HelpDesk MY.NET.70.50	12	6	1
EXPLOIT NTPDX buffer overflow	10	4	4
Probable NMAP fingerprint attempt	9	6	6
TCP SMTP Source Port traffic	9	3	3
EXPLOIT x86 stealth noop	8	6	6
[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC	8	1	1
SYN-FIN scan!	7	5	5
External FTP to HelpDesk MY.NET.70.49	6	5	1
External FTP to HelpDesk MY.NET.53.29	6	3	1
[UMBC NIDS IRC Alert] User joining XDCC channel detected. Possible XDCC bot	4	4	3
Back Orifice	2	1	1
DDOS mstream client to handler	2	2	2
NIMDA - Attempt to execute cmd from campus host	2	1	1
TFTP - External UDP connection to internal tftp server	2	2	2
[UMBC NIDS] Internal MiMail alert	2	1	1
Incomplete Packet Fragments Discarded [**] 68.94.221.5703/21-22:08:52.838514 [**] SMB Name Wildcard	1	1	1
[UMBC NIDS IRC Alert] K\:line'd user detected, possible trojan.	1	1	1
[UMBC NIDS IRC Alert] User joining Warez channel detected. Possible XDCC bot	1	1	1
Fragmentation Overflow Attack	1	1	1

Table 6

Top Ten Source IP addresses

Source IP address	# of Alert's	Alert(s)
68.55.155.26	21312	MY.NET.30.3 Activity MY.NET.30.4 Activity
68.48.90.101	16706	MY.NET.30.4 Activity
68.55.156.102	6192	MY.NET.30.3 Activity MY.NET.30.4 Activity

68.34.27.67	5141	MY.NET.30.3 Activity
68.55.205.180	4007	MY.NET.30.4 Activity
68.50.102.64	3301	MY.NET.30.4 Activity
61.129.45.60	2515	EXPLOIT x86 NOOP MY.NET.30.3 Activity MY.NET.30.4 Activity SMB Wildcard
68.33.138.193	2489	MY.NET.30.4 Activity
68.55.191.197	2226	MY.NET.30.4 Activity
63.13.142.215	1874	MY.NET.30.3 Activity MY.NET.30.4 Activity

Table 7

The top ten sources are almost all related to custom University rules, logging network traffic bound for MY.NET.30.3 and MY.NET.30.4 (both reviewed below). Due to the huge amount of alerts these rules generated, I decided to look at the top ten sources without any of the MY.NET.30.3 and MY.NET.30.4 alerts. Results are displayed below.

Top Ten Sources *(minus MY.NET.30.3 and MY.NET.30.4 alerts)*

Source IP address	**# of Alert's**	**Alert(s)**
68.32.127.158	4186	connect to 515 from outside
61.129.45.60	2184	EXPLOIT x86 NOOP
MY.NET.11.7	2018	SMB Name Wildcard
61.48.11.22	968	High port 65535 tcp - possible Red Worm - traffic Probable NMAP fingerprint attempt Null scan!
63.251.52.75	901	SYN-FIN scan! Probable NMAP fingerprint attempt

		Null scan!
MY.NET.75.13	862	SMB Name Wildcard
139.165.206.128	787	[UMBC NIDS IRC Alert] Possible drone command detected. [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.
MY.NET.150.198	665	SMB Name Wildcard
136.224.229.125	574	[UMBC NIDS IRC Alert] Possible drone command detected. [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.
MY.NET.112.174	542	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC

Table 8

Due to the same large number of alerts (MY.NET.30.3 and MY.NET.30.4), I decided to query for the top ten destinations without the custom university rules.

Top Ten Destination IP addresses

Source IP address	# of Alert's	Alert(s)
MY.NET.24.15	4269	TFTP - External TCP connection to internal tftp server connect to 515 from outside
169.254.25.129	2052	SMB Name Wildcard
MY.NET.71.240	1530	EXPLOIT x86 NOOP
169.254.45.176	1083	SMB Name Wildcard
MY.NET.153.76	974	High port 65535 tcp - possible Red Worm - traffic Probable NMAP fingerprint attempt EXPLOIT x86 NOOP Null scan!
MY.NET.153.173	866	Probable NMAP fingerprint attempt

		SYN-FIN scan! [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan. EXPLOIT NTPDX buffer overflow Back Orifice High port 65535 udp - possible Red Worm - traffic Null scan!
139.165.206.128	715	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC
136.224.229.125	660	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC
MY.NET.12.6	515	EXPLOIT identd overflow Happy 99 Virus Incomplete Packet Fragments Discarded Probable NMAP fingerprint attempt TCP SMTP Source Port traffic NMAP TCP ping! Tiny Fragments - Possible Hostile Activity [UMBC NIDS] External MiMail alert Null scan!
MY.NET.112.174	465	[UMBC NIDS IRC Alert] Possible drone command detected. [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.

Table 9

The top 3 source and destination IP's are all seen in the same three Snort events; connect to 515 from outside, SMB Name Wildcard and EXPLOIT x86 NOOP. All three of these alerts are reviewed below in the TOP 5 Alerts.

TOP 5 Alerts

MY.NET.30.4 activity

This alert appears to be a signature that triggers whenever network traffic is bound to the computer with an ip address of MY.NET.30.4. The rule was placed into the snort signatures list probably due to some unusual activity originating from or destined to MY.NET.30.4. This could also be a critical server that University personnel would like to monitor. Without actually getting a statement from them, we don't know exactly why this signature was added to the IDS.

The alert MY.NET.30.4 had 60,738 alerts from 292 sources all going to one destination. The top ten source ip addresses for this alert are displayed in Table 2.

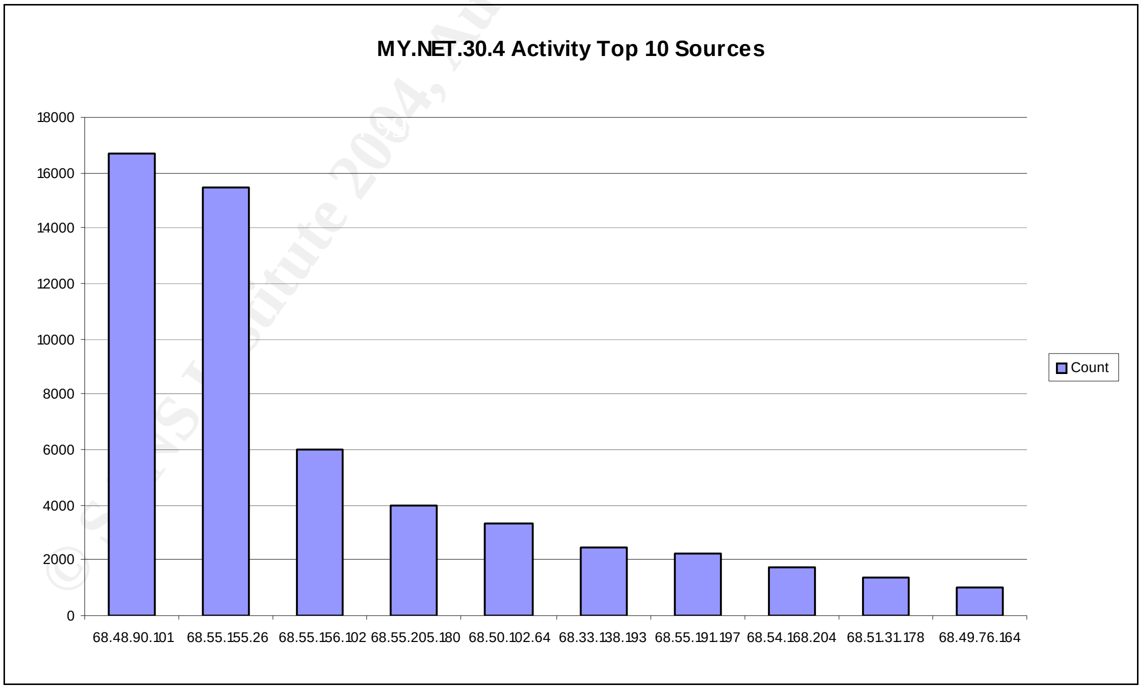


Chart 1

Of the 60,738 MY.NET.30.4 alerts, there were 43 different destination ports, but the overwhelming majority of the traffic was bound for port 51443 and about a 25% of the traffic was bound for port 8009. (All the ports that had less than 5 hits were grouped into the "other" category). See Chart 2 below.

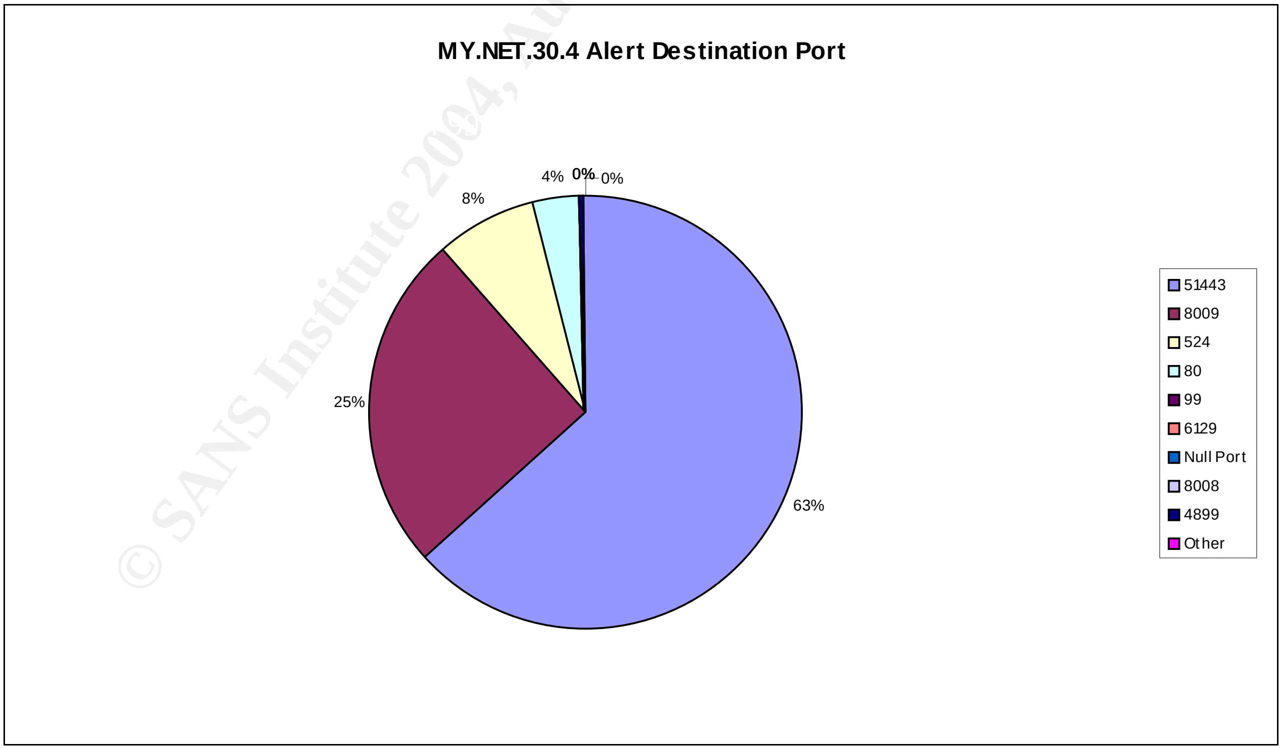


Chart 2

A search of the Neohapsis port list did not return any results for port 51443 and neither did a search of IANA's port list (Neohapsis)(IANA). A Google of port

51443 didn’t return much, but it did give me the name “NetStorage”. Some searching into NetStorage took me to Novell, which led me to iChain. You can read about iChain and NetStorage at Novell’s website. From what I could find on Novell’s website, port 51443 is the default port associated with NetStorage which is part of the iChain product (Novell).

TCP Port 8009 was a little bit easier to find information on. A search of the Neohapsis website told me that port 8009 is registered to the Novell Netware Remote Manager (Neohapsis).

With port 51443 being used for Novell’s NetStorage and port 8009 used for Novell Netware Remote Manager, I can make a very rough guess, that MY.NET.30.4 is running a Novell O/S. I can also assume (with caution) that this server is being used as a remote file storage server. A quick Google of “UMBC NetStorage” took me to this page (http://www.umbc.edu/oit/sans/helpdesk/Netstorage/netstorage.html). It stated:

“Novell NetStorage is a secure internet based application using web/internet browser. It will allow a user to copy, move, rename, delete, read, and write files between a home computer and their departments shared drive or personal space on the UMBC Novell servers.”.

If this is the case, most of the source IP addresses should be facility or students working from home in the Baltimore area and their IP’s should resolve to that geographical area. To confirm my suspicions I decided to see where the top 10 sources are from:

68.48.90.101 – Comcast Cable Communications, Inc.
68.55.155.26 – Comcast Cable Communications, Inc.
68.55.156.102 – Comcast Cable Communications, Inc.
68.55.205.180 – Comcast Cable Communications, Inc.
68.50.102.64 – Comcast Cable Communications, Inc.
68.33.138.193 – Comcast Cable Communications, Inc.
68.55.191.197 – Comcast Cable Communications, Inc.
68.54.168.204 – Comcast Cable Communications, Inc.
68.51.31.178 – Comcast Cable Communications, Inc.
68.49.76.164 – Comcast Cable Communications, Inc.

Comcast Cable Communications, Inc. sounds like it matches up with my suspicion. They are a local cable ISP for the Baltimore area.

Correlations

After I spent a good amount of time researching this, I'm kicking myself for not looking for correlations to start with. I googled "port 51443 GCIA". There were two practicals that covered the same alert.

- Tom King (www.giac.org/practical/GCIA/Tom_King_GCIA.pdf)

- David Barrosa (www.giac.org/practical/GCIA/David_Barroso_GCIA.pdf)

Tom King stated that port 51443 was used for secure iFolder part of Novell 6 web services. He also found confirmation that the University was trying out this product. David also stated the port 51443 was used in Novell Netware 6 file storage.

Alert Summary

As I stated above, this alert is a custom signature developed by University personnel. From the information I gathered, I determined that the MY.NET.30.4 server is some type of Novell server that allows facility and students to remotely store files. My review of the top users IP address confirmed my assumption that facility and students are accessing the file server from the same geographic location as the University.

I also wrote a couple of scripts to help parse out other alerts that were triggered by the same source IP's as the MY.NET.30.4 alert. Due to the nature of the MY.NET.30.4 signature this included almost all of the alerts for the entire 5 days. To perform correlation like this, the signature needs to be more specific.

MY.NET.30.3 activity

This alert is another custom signature created by University personnel to monitor traffic destined for a specific computer. In this case, it looks for traffic bound for MY.NET.30.3. I used the same techniques while analyzing this alert as I did for the MY.NET.30.4 alert. I took the top 10 source ip addresses and the destination ports to try to determine what was going on. Chart 3 displays the top ten source IP addresses.

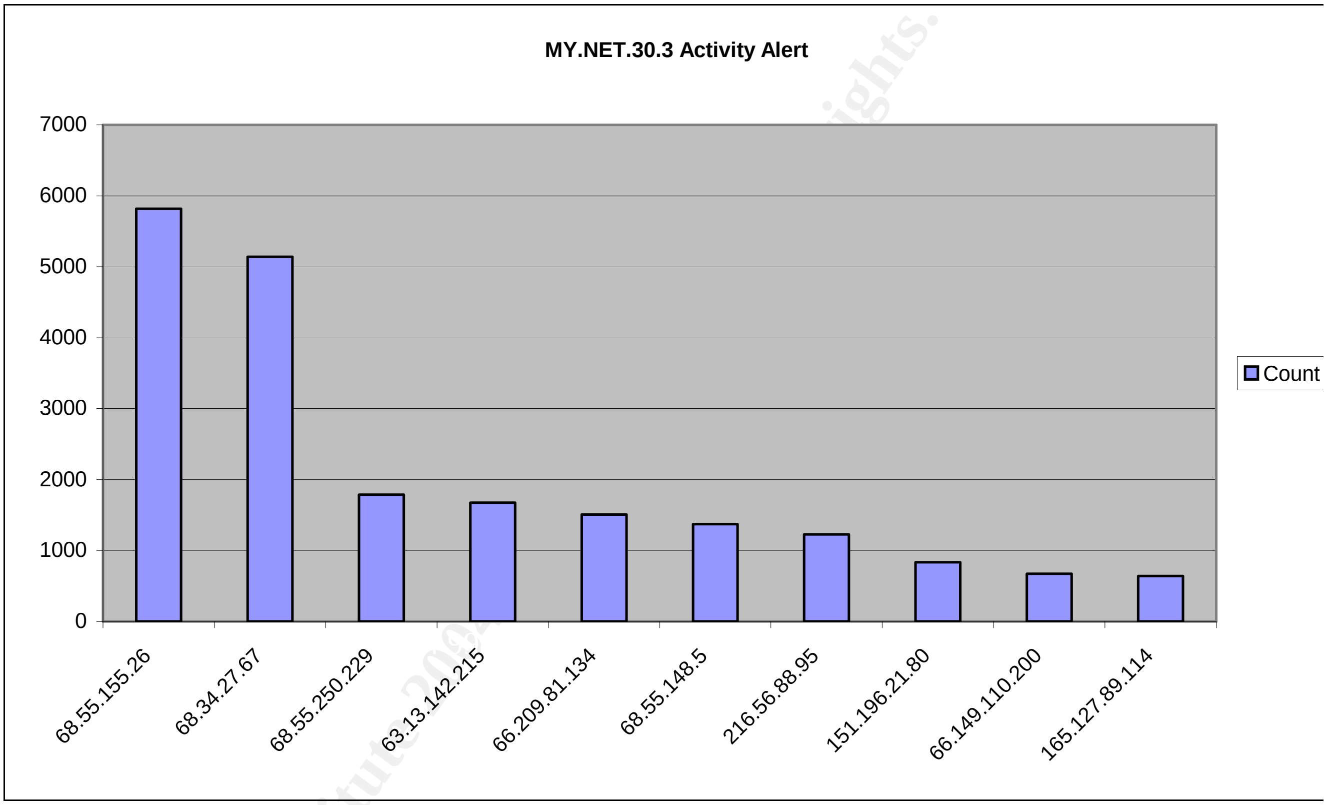


Chart 3

The top ten destination ports for MY.NET.30.3 are displayed below in Chart 4. Port 8009 is listed again at #2 and the #1 destination port is 524. Just to refresh our memory, port 8009 is the Novell Netware Remote Manager (Neohapsis).

Port 524 is registered as NCP or Novell Netware Core Protocol (IANA). Here is a link to more information about NCP (http://www.protocols.com/pbook/novel.htm#NCP).

I'm assuming (yes, again..) that MY.NET.30.3 is a Novell Server serving up files to remote users, similar to MY.NET.30.4.

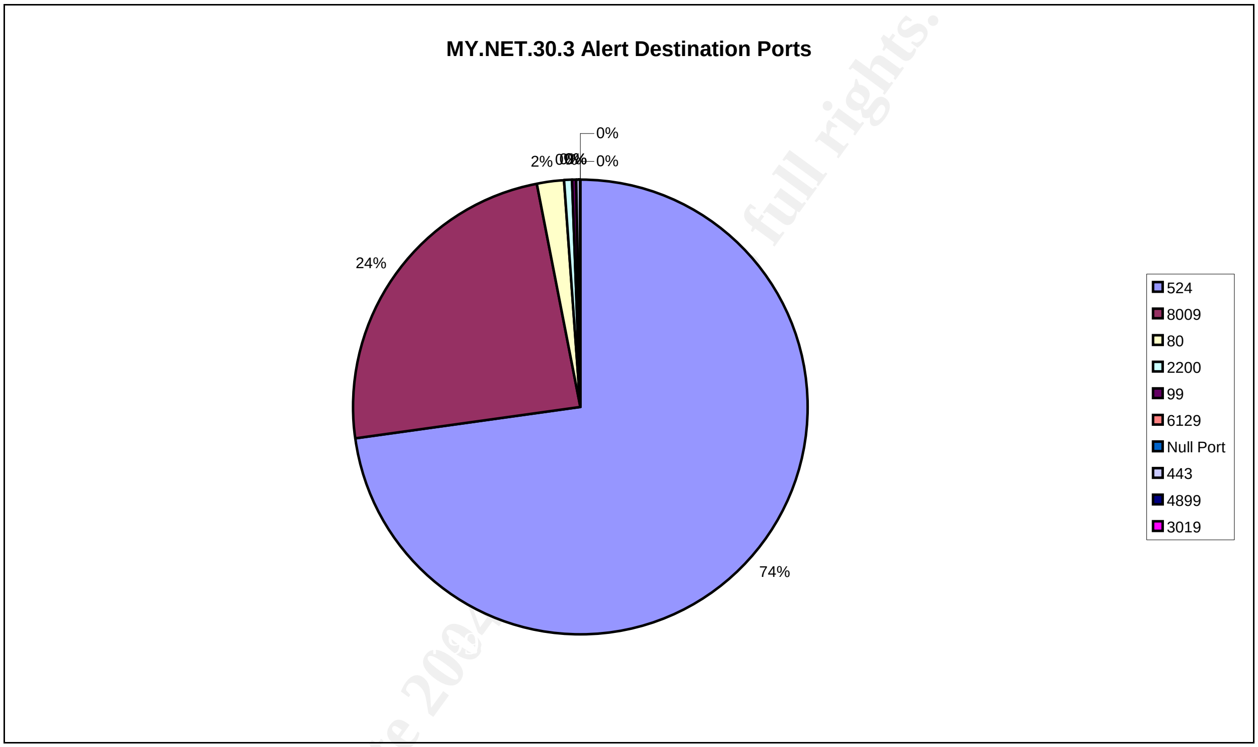


Chart 4

To help confirm this, I again looked up the top 10 sources to see if they are in the same geographic region as our university.

68.55.155.26 – Comcast Cable Communications, Inc.

68.34.27.67 – Comcast Cable Communications, Inc.

68.55.250.229 – Comcast Cable Communications, Inc.

63.13.142.215 –UUNET Technologies

66.209.81.134 – Power Pulse

68.55.148.5 – Comcast Cable Communications, Inc.

216.56.88.95 – WiscNet

151.196.21.80 - Verizon

66.149.110.200 – Earthlink

165.127.89.114 – State of Colorado

With the exception of the Power Pulse and the State of Colorado IP's all of the other IP's resolved to the same geographic region as our university. As for Power Pulse, there were no unusual alerts triggered. It was only observed in the MY.NET.30.3 alerts. The State of Colorado IP address was also logged in external TFTP alerts with three other internal IP addresses.

> -MY.NET.6.7 – Also involved in HIGH Port 65535 tcp – possible Red Worm – traffic and NMAP TCP Ping! Alerts. It looks like there may be a web server running on this machine, which the Red Worm and NMAP alerts targeted. All of the alerts showed MY.NET.6.7 as the destination.
>
> -MY.NET.24.44 – Also involved in NMAP TCP Ping!, Possible Trojan activity, Exploit X86 NOOP, HIGH Port 65535 tcp – possible Red Worm – traffic, Null Scan! and Incomplete Packet Fragments Discarded. As with the host above, this machine appears to be running a web server on port 80 and is the destination address in all of the alerts.
>
> -MY.NET.1.3 – Also involved in NMAP TCP Ping!, Exploit X86 NOOP, HIGH Port 65535 tcp – possible Red Worm – traffic, SUNRPC highport access!, Probable NMAP fingerprint attempt and Null Scan!. This server appears to be running a DNS server. Port 53 is the destination of all of the listed alerts.

It does not appear that any of the "other" alerts have anything to do with the TFTP alerts triggered by the source IP from the State of Colorado.

Correlations

I used Google again to find practical correlations. I find it easiest to use the site search when looking for correlations among practicals. The only drawback is that I usually get older practicals returned in the results.

The practicals that mention the same alerts are listed below:

- Bernard Kan (www.giac.org/practical/GCIA/Bernard_Kan_GCIA.doc)
- Tom King (www.giac.org/practical/GCIA/Tom_King_GCIA.pdf)
- Barbara Morgan (www.giac.org/practical/GCIA/Barbara_Morgan_GCIA.doc)

- Ronnie Clark (www.giac.org/practical/GCIA/Ronnie_Clark_GCIA.doc)

- Antonia Rona (www.giac.org/practical/GCIA/Antonia_Rana_GCIA.pdf)

All of these practicals noted the MY.NET.30.3 activity alert, but none of them actually analyzed this alert. The only one that discussed the MY.NET.30.X activity is Tom King's, which I discussed in the MY.NET.30.4 section.

EXPLOIT x86 NOOP

This alert is triggered when an attempted buffer overflow is detected. According to the Snort website (http://www.snort.org/snort-db/sid.html?sid=1394), "the NOOP warning occurs when a series of NOOP (no operation) are found in stream" (Snort).

This is one of my favorite alerts, NOT! Anyone that has looked at IDS's for a while knows that this alert is prone to a high-false positive rate. This alert is known for a high false positive rate in environments where there are a lot of file transfers and web surfing. Now that we now that, let's look at a breakdown of destination ports to see which ports were involved in these alerts.

Here is a breakdown of all of the Top 10 destination ports for the Exploit x86 NOOP alert:

Port	Count	Port Assignment
80	5290	World Wide Web HTTP (iana)
6881	1528	unassigned (iana)
135	413	DCE endpoint resolution (iana)
1025	390	network blackjack (iana)
119	53	Network News Transfer Protocol (iana)
445	47	Microsoft-DS (www.iana.org)
5000	21	Universal Plug and Play (neohapsis)
4140	19	unassigned (iana)
6129	18	DameWare remote control agent (neohapsis)
2889	10	RSOM (iana)
1944	10	close-combat (iana)

It is not unexpected to see port 80 (www) as the #1 port for this alert. JPEG pictures typically trigger this alert a lot. The #2 port, 6881, does stick out a little. The port is unassigned according to IANA (IANA). This could be malicious traffic aimed towards a custom application that the University is running or it could just be normal traffic from a custom traffic or it could be file an FTP file transfer that triggered this alert. I searched through the alerts for more clues, but I couldn't find any. We would need more information to properly analyze what is going on with port 6881 and the NOOP alerts.

There were 723 different sources and 543 different destinations. My experience tells me that most of these are most likely false positives. I chose to look closely at the first two ports for this alert, mainly due to their high numbers, but in the field it could be the NOOP with one alert that is the real malicious activity. The best course of action here is figure out your definite false positives first and then from there.

Correlations

This attack was discussed by Erik Montcalm (Montcalm). He came to the same conclusions that this attack is a false positive prone alert and he gave the following link: http://www.derkeiler.com/Mailing-Lists/securityfocus/focus-ids/2002-04/0041.html

The above URL links to a mailing list post where a couple of security professionals discuss the EXPLOIT x86 NOOP signature and attack.

SMB Name Wildcard

This alert is triggered when a standard NETBIOS table retrieval query is observed. This is a common occurrence in Windows networks, when machines exchange queries to discover NETBIOS names (Neohapsis Archives).

Below is a listing of the top source addresses (less than 20 alerts were omitted for space).

Count	IP address
1551	MY.NET.11.7
737	MY.NET.75.13
507	MY.NET.150.198
267	MY.NET.150.44
67	MY.NET.29.30
33	MY.NET.190.92
27	MY.NET.190.93
27	MY.NET.190.102
24	MY.NET.5.34
24	MY.NET.153.85
24	MY.NET.152.17
23	MY.NET.190.95
23	MY.NET.112.152

It appears as if all of the alerts were destined for external addresses. Here is a list of the top destination addresses (less than 20 alerts were omitted for space).

Count	IP Address
1582	169.254.25.129
807	169.254.45.176
87	199.239.137.216

```
50    63.163.24.78
36    63.218.84.115
30    63.218.84.28
30    63.218.84.26
28    216.74.144.15
26    64.211.50.56
26    63.218.84.5
24    216.74.144.14
24    216.145.5.196
23    131.211.213.63
20    63.218.84.21
```

It seems to be almost a one for one ratio. There do not seem to be any scanners or one computer does not appear to be generating all of the traffic. That said, the MY.NET.11.7 -> 169.254.25.129 pair does have a huge amount of alerts. The destination address appears to be external, but if you look up 169.254.0.0, you'll find out that this range is used by Microsoft as its Automatic Private IP Addressing (APIPA) net block. This net block is used whenever a windows DHCP client can't find a DHCP server. As far as I could tell, APIPA addresses are not routable, so this leads me to the conclusion that this machine is local (Microsoft).

I would investigate all of the other destination addresses though; they could be external hosts conducting probing or worse, exploiting common NETBIOS vulnerabilities.

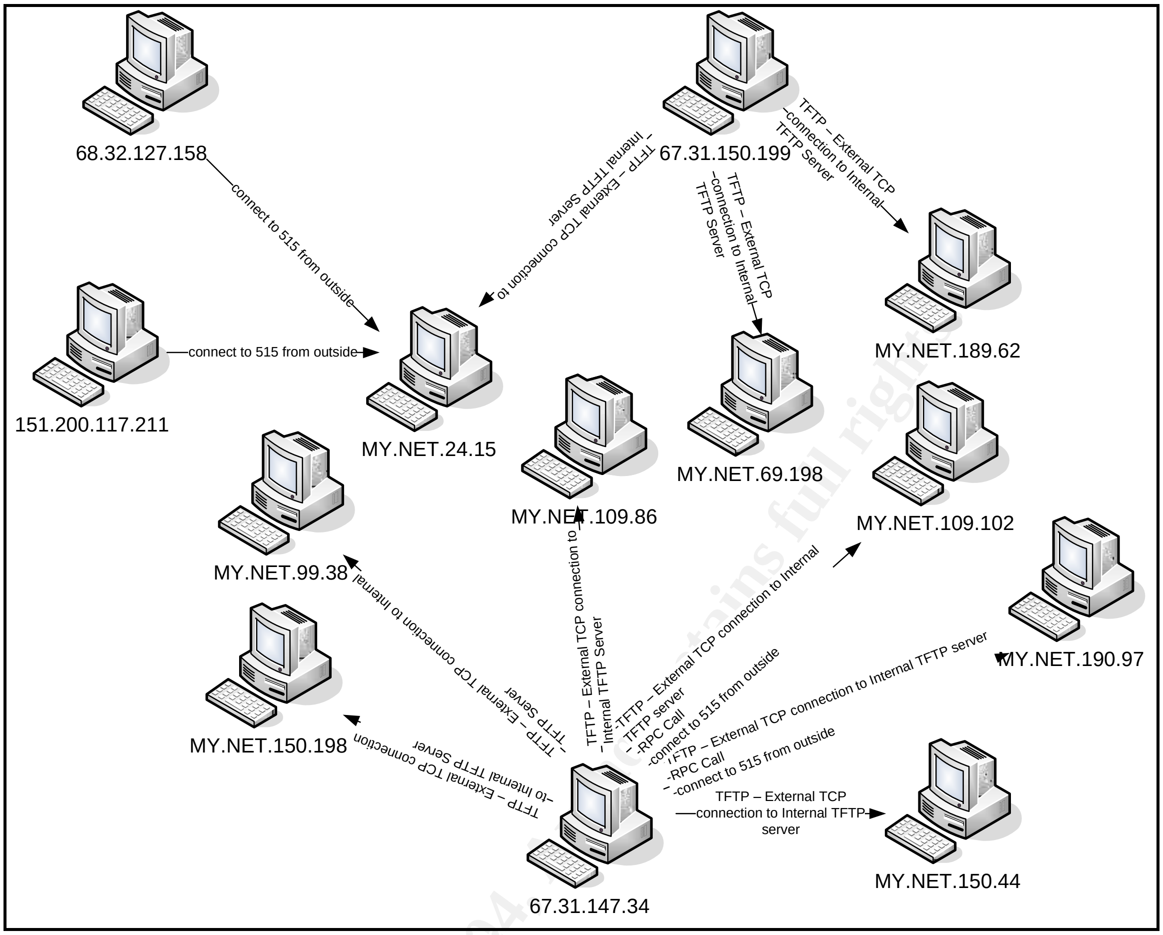
Correlations

Brian Cahoon also discussed this alert in his practical (Cahoon). He referenced the SANS Top Twenty (www.sans.org/top20.htm) for NETBIOS vulnerabilities. The SANS Top Twenty discussed NETBIOS vulnerabilities under W5 – Windows Remote Access Services.

connect to 515 from outside

This alert triggers when an attempt to connect to port 515 on a local computer is observed. TCP Port 515 is the Line Printer Daemon Port (IANA). This port has been associated with a couple of worms (Ramen and lpdw0rm) according to Internet Storm Center (ISC).

There were 4 different sources all connecting to port 515 from outside of the local network. See the link graph of all the connections below:



68.32.127.158
67.31.150.199
151.200.117.211
MY.NET.24.15
MY.NET.189.62
MY.NET.69.198
MY.NET.109.86
MY.NET.109.102
MY.NET.99.38
MY.NET.150.198
MY.NET.190.97
MY.NET.150.44
67.31.147.34
connect to 515 from outside
connect to 515 from outside
TFTP – External TCP connection to Internal TFTP Server
TFTP – External TCP connection to Internal TFTP Server
TFTP – External TCP connection to Internal TFTP Server
TFTP – External TCP connection to Internal TFTP Server
TFTP – External TCP connection to Internal TFTP Server
TFTP – External TCP connection to Internal TFTP Server
-TFTP – External TCP connection to Internal TFTP server
-RPC Call
-connect to 515 from outside
TFTP – External TCP connection to Internal TFTP server
-RPC Call
-connect to 515 from outside
TFTP – External TCP connection to Internal TFTP server

Figure 16

All of these alerts are pretty troubling, but in particular are the alerts from 67.31.150.199 and 67.31.147.34. These two sources had a couple more alerts associated with them as you can see from the Illustration above.

What bothers me about these alerts is that we have a port with known vulnerabilities being accessed, combined with TFTP and RPC traffic. Not good! I recommend that the destination machines be investigated right away and the source machines be blocked. It turns out that both source machines have IP's within a range owned by the same organization, Level 3 communications, so chances are that one person is controlling both machines.

Correlations

- CERT Incident Note IN-2001-01 (http://www.cert.org/incident_notes/IN-2001-01.html)
- CERT Advisory CA-2000-22 Input Validation Problems in LPRng (http://www.cert.org/advisories/CA-2000-22.html)

Out of Specification Logs

The out of specification logs are useful because they help us see "abnormal" traffic. They don't tell us exactly what is happening, but I like to use them to focus my searches of the Snort Alert logs. Below are the top talkers for the OOS logs:

Top Ten Sources

Count	Address	Alert Log Correlation
1306	68.54.84.49	Scanning Activity
144	66.225.198.20	Scanning Activity
114	67.72.78.212	Scanning Activity
108	80.243.1.199	Scanning Activity
101	67.114.19.186	Scanning Activity
66	151.196.178.128	none
65	68.122.128.1	Scanning Activity
65	35.8.2.252	Scanning Activity
46	68.115.197.90	none

Top Ten Destinations

Count	Address	Alert Log Correlation
1399	MY.NET.6.7	High port 65535 tcp - possible Red Worm - traffic NMAP TCP ping! TFTP - External TCP connection to internal tftp server
862	MY.NET.12.6	[UMBC NIDS] External MiMail alert Null scan! Tiny Fragments - Possible Hostile Activity NMAP TCP ping! Scanning Activity TCP SMTP Source Port traffic Incomplete Packet Fragments Discarded
394	MY.NET.24.44	NMAP TCP ping! Possible trojan server activity EXPLOIT x86 NOOP High port 65535 tcp - possible Red Worm - traffic TFTP - External TCP connection to internal tftp server Scanning Activity Null scan! Incomplete Packet Fragments Discarded
166	MY.NET.24.34	High port 65535 tcp - possible Red Worm - traffic Possible trojan server activity

```
                        EXPLOIT x86 NOOP
                        NMAP TCP ping!
95    MY.NET.12.4
                        Null scan!
                        Scanning Activity
                        NMAP TCP ping!
                        Incomplete Packet Fragments Discarded
                        High port 65535 tcp - possible Red Worm –
                        traffic
76    MY.NET.34.11
                        EXPLOIT x86 NOOP
                        NMAP TCP ping!
                        Possible trojan server activity
                        Incomplete Packet Fragments Discarded

46    MY.NET.112.152
                        SMB Name Wildcard
                        Scanning Activity
                        Incomplete Packet Fragments Discarded
                        EXPLOIT x86 NOOP
                        Null scan!
40    MY.NET.84.203
                        High port 65535 tcp - possible Red Worm -
                        traffic
                        Scanning Activity
                        Incomplete Packet Fragments Discarded
                        Null scan!
                        EXPLOIT x86 NOOP
                        EXPLOIT x86 setuid 0
37    MY.NET.34.14
                        Scanning Activity
                        High port 65535 tcp - possible Red Worm -
                        traffic
                        SUNRPC highport access!
                        NMAP TCP ping!
                        MY.NET.30.4 activity
36    MY.NET.70.164
                        Scanning Activity
                        Null scan!
                        Incomplete Packet Fragments Discarded
                        High port 65535 tcp - possible Red Worm -
                        traffic
                        SYN-FIN scan!
                        EXPLOIT x86 setuid 0
```

Top Ten Destination Ports

Count	Port	Service
1371	110	POP3
895	25	SMTP
784	80	HTTP
122	4662	Overnet P2P Server

89	113	IDENT
27	6346	gnutella (BearShare)
19	2234	DirectPlay
19	22	SSH
18	143	IMAP
14	443	SSL

Using the OOS logs, we are able to quickly zero in on suspicious traffic. And by searching our Alert logs, we are able to get more information on some of the top talkers as shown in the tables above.

Scan Logs

Of all three logs, the Scan logs were the largest. There were over 14 million scans logged by the Snort sensor. Scans are important to monitor, because attackers often scan networks for the holes or vulnerabilities they are attempting to exploit. If we know what attackers are targeting, hopefully we can stay one step ahead of them.

Below are the top talkers broken out by Top Ten Sources, Destinations and Destination Ports:

Top Ten Sources

Count	Address
4678240	130.85.153.195
3166044	130.85.1.3
2013957	130.85.190.92
600317	130.85.97.209
514282	130.85.1.4
342529	130.85.18.27
224416	130.85.84.187
213648	130.85.153.174
182065	130.85.150.199
163049	130.85.110.72

Top Ten Destinations

Count	Address
87829	69.6.57.9
87655	69.6.57.8
87505	69.6.57.10
83705	69.6.57.7
67866	130.85.97.85
64668	192.26.92.30
55983	130.85.25.69
50021	192.48.79.30
48277	203.20.52.5
41528	192.5.6.30

Top Ten Destination Ports

Count	Port	Service
3662369	53	DNS
2162597	135	NETBIOS Session Service
1953758	445	W2K SMB
1323957	2745	URBISNET (Bagle/Beagle Worm Backdoor)
937779	1025	Network Blackjack
856211	6129	DameWare
717390	80	http
711044	3127	none (Mydoom Worm Backdoor)
458740	139	DCE Endpoint Resolution
305348	25	SMTP

It appears that the Top Ten Scanners are local machines. This could be due to web, DNS and Mail servers appearing to be scanning machines by snort. To verify my theory, I searched the scan logs to see what was the source port being accessed. If these machines really are servers, then the scanning source port should be a well known port (i.e. 80, 53 or 25). If the source address is conducting scanning for vulnerabilities then the destination ports will be our well known ports (i.e. 80, 53, or 25). See the illustrations below:

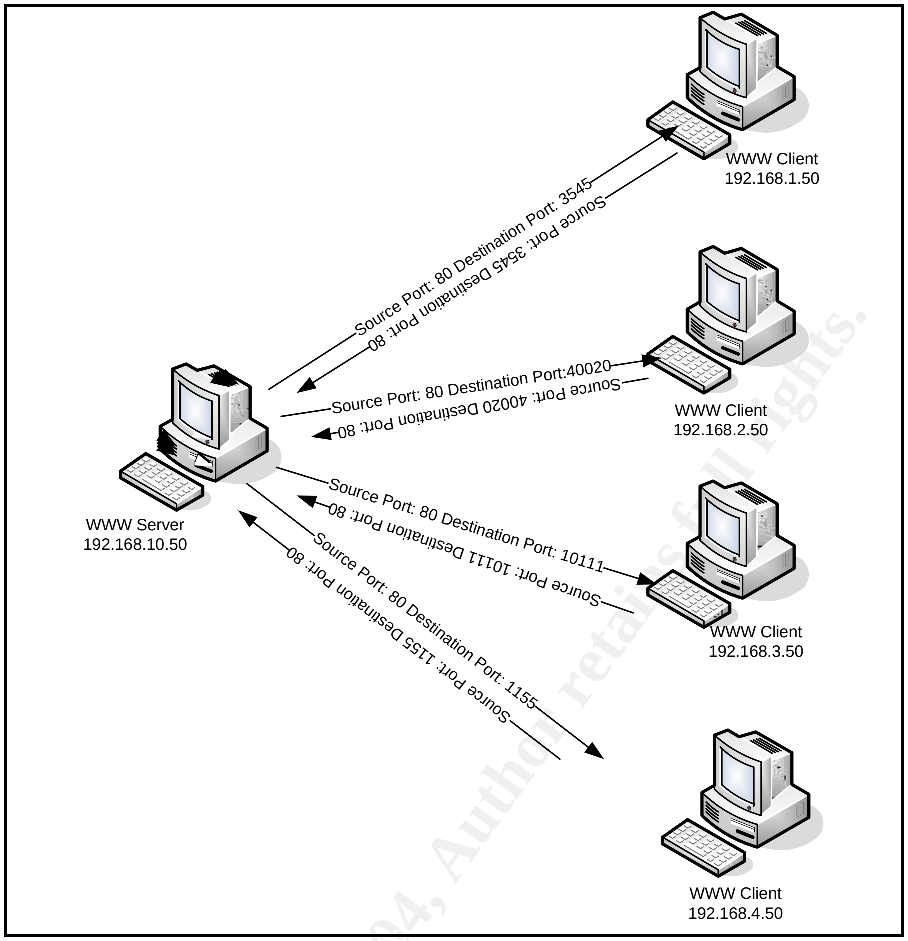

WWW Server
192.168.10.50
WWW Client
192.168.1.50
WWW Client
192.168.2.50
WWW Client
192.168.3.50
WWW Client
192.168.4.50
Source Port: 80 Destination Port: 3545
Source Port: 3545 Destination Port: 80
Source Port: 80 Destination Port:40020
Source Port: 40020 Destination Port: 80
Source Port: 80 Destination Port: 10111
Source Port: 10111 Destination Port: 80
Source Port: 80 Destination Port: 1155
Source Port: 1155 Destination Port: 80

Figure 17

This activity appears to Snort as scanning activity with the source being the web server. This is what I was hoping to see when I searched the logs, but what I actually saw is illustrated below:

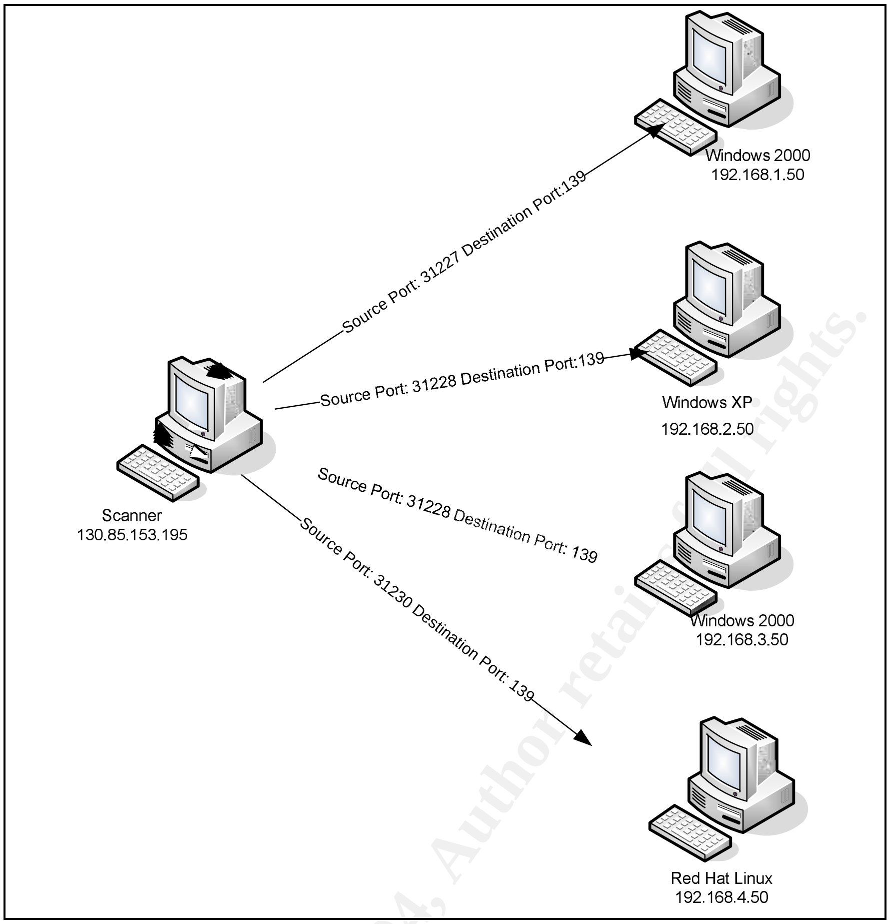

Windows 2000
192.168.1.50
Source Port: 31227 Destination Port:139
Source Port: 31228 Destination Port:139
Windows XP
192.168.2.50
Scanner
130.85.153.195
Source Port: 31228 Destination Port: 139
Windows 2000
192.168.3.50
Source Port: 31230 Destination Port: 139
Red Hat Linux
192.168.4.50

Figure 18

Below are the ports our Top Ten Scanners were scanning for the most.

```
Count       Port  Service
3660228     53    DNS
3112214     135   NETBIOS Session Service
2513537     445   W2K SMB
2177684     2745  URBISNET (Bagle/Beagle Worm Backdoor)
1556749     1025  Network BlackJack
1185137     3127  none (Mydoom Worm Backdoor)
1026232     6129  DameWare
 836502     139   DCE Endpoint Resolution
 745495     80    http
  16537     123   Network Time Protocol
```

Some of the Top Scanners are exhibiting characteristics similar to the Phatbot or Agobot Worms. According to LURHQ, the Phatbot/Agobot worms conduct scanning to exploit the following vulnerabilities (LURHQ):

- DCOM
- DCOM2
- MyDoom backdoor

- DameWare
- Locator Service (Update: This exploit appears to be non-functional)
- Shares with weak passwords
- WebDav
- WKS - Windows Workstation Service
- Bagle virus backdoor
- CPanel resetpass vulnerability
- UPnP (MS01-059)
- MSSQL weak administrator passwords

WHOIS Lookups

Below are the external IP addresses picked for lookup. I used a combination of the regional whois websites and the Dshield database to perform my queries (Dshield). I picked the top 2 external IP addresses from each log, Alert, Scan and OOS. The top 2 external addresses from the Alert log with the MY.NET.30.3 and MY.NET.30.4 activity alerts filtered out.

- Alert Log
  - 68.32.127.158
  - 61.129.45.60
- OOS Log
  - 68.54.84.49
  - 66.225.198.20
- Scan Log
  - 64.136.199.197
  - 213.180.193.68

Alert Log

IP Address: 68.32.127.158

HostName: pcp01823879pcs.howard01.md.comcast.net

Dshield Profile:

Country:	🇺🇸 US
Contact E-mail:	
AS Number:	22909
Total Records against IP:	not processed
Number of targets:	select update below
Date Range:	to

Update Summary

Whois:

CustName: Comcast Cable Communications, Inc.

Address: 3 Executive Campus

```
Address:    5th Floor
City:       Cherry Hill
StateProv:  NJ
PostalCode: 08002
Country:    US
RegDate:    2003-03-18
Updated:    2003-03-18

NetRange:   68.32.112.0 - 68.32.127.255
CIDR:       68.32.112.0/20
NetName:    BALTIMORE-A-2
NetHandle:  NET-68-32-112-0-1
Parent:     NET-68-32-0-0-1
NetType:    Reassigned
Comment:    NONE
RegDate:    2003-03-18
Updated:    2003-03-18

TechHandle: IC161-ARIN
TechName:   Comcast Cable Communications Inc
TechPhone:  +1-856-317-7200
TechEmail:  cips_ip-registration@cable.comcast.com

OrgAbuseHandle: NAPO-ARIN
OrgAbuseName:   Network Abuse and Policy Observance
OrgAbusePhone:  +1-856-317-7272
OrgAbuseEmail:  abuse@comcast.net

OrgTechHandle: IC161-ARIN
OrgTechName:   Comcast Cable Communications Inc
OrgTechPhone:  +1-856-317-7200
OrgTechEmail:  cips_ip-registration@cable.comcast.com
```

IP Address: 61.129.45.60

HostName: 61.129.45.60

Dshield Profile:

Country:	🇨🇳 CN
Contact E-mail:	ip-admin@mail.online.sh.cn
AS Number:	4134
Total Records against IP:	56
Number of targets:	7
Date Range:	2004-03-09 to 2004-04-28

Update Summary

Last Fightback Sent: not sent

```
% [whois.apnic.net node-2]
% Whois data copyright terms    http://www.apnic.net/db/dbcopyright.html
inetnum:     61.129.45.48 - 61.129.45.83
netname:      null
descr:          null
country:     CN
admin-c:     WQ58-AP
tech-c:         WL371-AP
mnt-by:         MAINT-CHINANET-SH
changed:      wanglin@shaidc.com 20040413
status:          ASSIGNED NON-PORTABLE
source:          APNIC
person:      Wang Qing
address:      6F,380 Fushan Road,Shanghai   200122
country:     CN
phone:       +86-21-68761255-807
fax-no:     +86-21-68761255-805
e-mail:     wanglin@shaidc.com
nic-hdl:     WQ58-AP
mnt-by:      MAINT-CN-SHTELE-XINCHAN
changed:      wanglin@shaidc.com 20021007
source:      APNIC
person:      Wang Lin
address:      6F,380 Fushan Road,Shanghai   200122
country:     CN
phone:       +86-21-68761255-807
```

```
fax-no:     +86-21-68761255-805
e-mail:     wanglin@shaidc.com
nic-hdl:    WL371-AP
mnt-by:     MAINT-CN-SHTELE-XINCHAN
changed:    wanglin@shaidc.com 20021007
source:     APNIC
```

OOS Log

IP Address: 68.54.84.49

HostName: pcp01741335pcs.howard01.md.comcast.net

Dshield Profile:

Country:	🇺🇸 US
Contact E-mail:	abuse@comcastpc.com
AS Number:	22909
Total Records against IP:	not processed
Number of targets:	select update below
Date Range:	to

Update Summary

Whois:

```
CustName:   Comcast Cable Communications, Inc.
Address:    3 Executive Campus
Address:    5th Floor
City:       Cherry Hill
StateProv:  NJ
PostalCode: 08002
Country:    US
RegDate:    2003-03-19
Updated:    2003-03-19

NetRange:   68.54.80.0 - 68.54.95.255
CIDR:       68.54.80.0/20
NetName:    BALTIMORE-A-4
NetHandle:  NET-68-54-80-0-1
Parent:     NET-68-32-0-0-1
NetType:    Reassigned
Comment:    NONE
```

```
RegDate:    2003-03-19
Updated:    2003-03-19

TechHandle: IC161-ARIN
TechName:   Comcast Cable Communications Inc
TechPhone:  +1-856-317-7200
TechEmail:  cips_ip-registration@cable.comcast.com

OrgAbuseHandle: NAPO-ARIN
OrgAbuseName:   Network Abuse and Policy Observance
OrgAbusePhone:  +1-856-317-7272
OrgAbuseEmail:  abuse@comcast.net

OrgTechHandle: IC161-ARIN
OrgTechName:   Comcast Cable Communications Inc
OrgTechPhone:  +1-856-317-7200
OrgTechEmail:  cips_ip-registration@cable.comcast.com

# ARIN WHOIS database, last updated 2004-05-12 19:15
# Enter ? for additional hints on searching ARIN's WHOIS database.


OrgName:    Comcast Cable Communications, Inc.
OrgID:      CMCS
Address:    1800 Bishops Gate Blvd
City:       Mt Laurel
StateProv:  NJ
PostalCode: 08054
Country:    US

NetRange:   68.32.0.0 - 68.63.255.255
CIDR:       68.32.0.0/11
NetName:    JUMPSTART-1
NetHandle:  NET-68-32-0-0-1
Parent:     NET-68-0-0-0-0
NetType:    Direct Allocation
```

```
NameServer: DNS01.JDC01.PA.COMCAST.NET
NameServer: DNS02.JDC01.PA.COMCAST.NET
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2001-11-29
Updated:    2003-11-05

TechHandle: IC161-ARIN
TechName:   Comcast Cable Communications Inc
TechPhone:  +1-856-317-7200
TechEmail:  cips_ip-registration@cable.comcast.com

OrgAbuseHandle: NAPO-ARIN
OrgAbuseName:   Network Abuse and Policy Observance
OrgAbusePhone:  +1-856-317-7272
OrgAbuseEmail:  abuse@comcast.net

OrgTechHandle: IC161-ARIN
OrgTechName:   Comcast Cable Communications Inc
OrgTechPhone:  +1-856-317-7200
OrgTechEmail:  cips_ip-registration@cable.comcast.com
```

IP Address: 66.225.198.20

HostName: unknown.splashhost.net

Dshield Profile:

Country:	🇺🇸 US
Contact E-mail:	
AS Number:	23352
Total Records against IP:	not processed
Number of targets:	select update below
Date Range:	to

Update Summary

Whois:

```
OrgName:    Server Central Network
OrgID:      SCN-18
Address:    2002 W Chicago
Address:    PMB 101
```

```
City:       Chicago
StateProv:  IL
PostalCode: 60622
Country:    US

NetRange:   66.225.192.0 - 66.225.255.255
CIDR:       66.225.192.0/18
NetName:    SCN-2
NetHandle:  NET-66-225-192-0-1
Parent:     NET-66-0-0-0-0
NetType:    Direct Allocation
NameServer: NS1.SCSERVERS.COM
NameServer: NS2.SCSERVERS.COM
Comment:
RegDate:    2003-06-10
Updated:    2004-04-29

TechHandle: JL1890-ARIN
TechName:   Server Central, Jordan
TechPhone:  +1-312-829-1111
TechEmail:  scsupport@servercentral.net

OrgTechHandle: JL1890-ARIN
OrgTechName:   Server Central, Jordan
OrgTechPhone:  +1-312-829-1111
OrgTechEmail:  scsupport@servercentral.net
```

Scan Log

IP Address: 64.136.199.197

HostName: 64-136-199-197-dhcp-kc.everestkc.net

Dshield Profile:

Country:	🇺🇸 US
Contact E-mail:	abuse@everestkc.net
AS Number:	18712
Total Records against IP:	not processed
Number of targets:	select update below

Date Range:	to

Update Summary

Whois:

OrgName: Everest Broadband

OrgID: EVERES-14

Address: 9669 Lackman Road

City: Lenexa

StateProv: KS

PostalCode: 66219

Country: US

NetRange: 64.136.192.0 - 64.136.207.255

CIDR: 64.136.192.0/20

NetName: EVEREST-KSLECMTS-2

NetHandle: NET-64-136-192-0-2

Parent: NET-64-136-192-0-1

NetType: Reassigned

NameServer: NS1.EVERESTKC.NET

NameServer: NS2.EVERESTKC.NET

Comment: Service provided by Everest Connections

Comment: (http://everestgt.com)

RegDate: 2002-09-25

Updated: 2002-10-03

IP Address: 213.180.193.68

HostName: proxychecker.yandex.net

Dshield Profile:

Country:	🇷🇺 RU
Contact E-mail:	kostik@comptek.ru
AS Number:	13238
Total Records against IP:	5221
Number of targets:	74
Date Range:	2004-04-29 to 2004-05-12

Update Summary

Last Fightback Sent: sent to kostik@comptek.ru on 2003-12-29 13:27:36

Whois: % This is the RIPE Whois server.

% The objects are in RPSL format.

```
%
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      213.180.192.0 - 213.180.193.255
netname:      COMPTEK-NET1
descr:        CompTek International
descr:        3, Gubkina str., Moscow, 117809
country:      RU
admin-c:      YNDX1-RIPE
tech-c:       YNDX1-RIPE
status:       ASSIGNED PA
notify:       noc@yandex.net
mnt-by:       COMPTEK-MNT-RIPE
changed:      wawa@comptek.ru 20020607
source:       RIPE

route:        213.180.192.0/20
descr:        CompTek network / special
origin:       AS13238
notify:       noc@comptek.ru
mnt-by:       COMPTEK-MNT-RIPE
changed:      wawa@comptek.ru 20010123
source:       RIPE

role:         Yandex LLC Network Operations
address:      Yandex LLC
address:      40A Vavilova st.
address:      117333, Moscow, Russia
phone:        +7 095 9743555
fax-no:       +7 095 9743565
e-mail:       noc@yandex.net
trouble:      ------------------------------------------------------
trouble:      Points of contact for Yandex LLC Network Operations
trouble:      ------------------------------------------------------
trouble:      Routing and peering issues:  noc@yandex.net
```

```
trouble:     SPAM issues:              abuse@yandex.ru
trouble:     Network security issues:     abuse@yandex.ru
trouble:     Mail issues:              postmaster@yandex.ru
trouble:     General information:        info@yandex.ru
trouble:     ----------------------------------------------------
admin-c:     VLI1-RIPE
tech-c:      KBG2-RIPE
notify:      noc@yandex.net
nic-hdl:     YNDX1-RIPE
mnt-by:      COMPTEK-MNT-RIPE
changed:     wawa@comptek.ru 20020607
source:      RIPE
```

Defensive Recommendations

I discussed some general University Defensive Recommendations in the opening part of this section. In addition to firewall rules, IDS tuning and patch management, here are some specific recommendations for the alerts that I analyzed.

- MY.NET.30.4 and MY.NET.30.3 alerts do not need to be set to alert every time, network traffic is observed going to those two destinations. By changing to snort rule from:

```
any any -> MY.NET.30.4 any alert
```

to

```
any any -> MY.NET.30.4 any log
```

This would log the alert in the snort logs, but not in the alert log. This will help cut down on all of the alerts that the IDS analysts has to sift through and might help him/her catch some more obscure alerts.

- SMB Name Wildcard – This alert is triggered by netbios traffic associated with Windows Networks. This alert is a perfectly normal alert to see among internal network machines, the concern comes in when you have external addresses triggering this alert. That means that you are allowing netbios ports through the firewall. As stated in the first section, firewall rules need to be evaluated. If netbios ports are deemed necessary, then it should only be allowed to specific machines that are known to be well maintained and monitored.

- connect to 515 from outside – This is another alert where the firewall rules need to be reviewed. If port 515 is deemed necessary, then specific internal machines should be allowed access, not the entire internal network.

Analysis Method

When I started this portion of the practical, I was blown away by the sheer amount of data that I had to analyze. The first step was to review as many practical as I could. I read over other methods that other people used. I found a lot of great ideas for manipulating the data. After I scoured the posted GCIA practicals, I found some useful perl scripts in Peter Van Oosterom's practical. His scripts allowed me to sort through the Alert Log. In addition to Peter's script to sort Alert data, I ran the logs through SnortSnarf. In order to get SnortSnarf to take the logs without "choking" too much, I stripped out the spp_portscan entries in the log with the following command within vi.
g/spp_portscan/d. SnortSnarf took about 30-45minutes to chew through the data, so be patient.

To get more detailed in certain events, I wrote some scripts to parse through the data for me. They are not pretty, but they worked for me. Here is the script I used to see what else the source IP's in MY.NET.30.4 Activity alert did.

```
#!/bin/bash
grep 'MY.NET.30.4'  all_alert |  awk '{print $6}' | awk -F: '{print $1}' | uniq -u > temp1
echo "Done getting MY.NET.30.4 alerts"
echo "I'm looking for other alerts now"
grep -f temp1 all_alert > temp2
echo "gotta get rid of those stupid scan alerts"
grep -v 'spp_portscan' temp2 > temp3
echo "Let me clean the results up a little"
cat temp3 | awk -F**] '{print $2}' | awk -F[ '{print $1}' | sort -ru  > temp4
echo "grepping out MY.NET's"
grep -v 'MY.NET.30.' temp4 > temp5
echo "I'm Done"
```

After I had the results in a file, I manually inputted them into another script. Due to my limited abilities, I couldn't automate that process, though I'm sure someone else could. Here is the second script I ran. This script generated a list of IP's from the list generated above. I omitted most of the script to limit space, but the rest follows the same pattern.

```
#!/bin/bash
echo "MY.NET.30.4 related activity" > temp9
echo "connect to 515 from outside sources" >> temp9
echo >> temp9
echo  "connect to 515 from outside"
```

```
grep   'connect to 515 from outside' all_alert | awk '{print $9,$11}' | awk -F: '{print $1}' | sort -r |
uniq -c | sort -rn >> temp9
echo "Tiny Fragments - Possible Hostile Activity" >> temp9
echo >> temp9
echo " Tiny Fragments - Possible Hostile Activity"
grep 'Tiny Fragments - Possible Hostile Activity' all_alert | awk '{print $10,$12}' | awk -F: '{print
$1}' | sort -r | uniq -c | sort -rn >> temp9
echo "TFTP - Internal UDP connection to external tftp server"
echo >> temp9
grep 'TFTP - Internal UDP connection to external tftp server' all_alert | awk '{print $13,$15}' | awk
-F: '{print $1}' | sort -r | uniq -c | sort -rn >> temp
```

I used this list of IP's and Alerts to see what else the source IP's in MY.NET.30.4 Activity alert did. I'm sure there are easier and quicker ways but this is what my little brain came up with. :)

Peter wrote a script to parse out the scan log information, but I didn't have much luck with it, so I wrote the following script to do some real primitive data sorting.

```
cat all_scans | awk '{print $4}' | awk -F: '{print $1}' | sort | uniq -
c | sort -rn > scan_source
cat all_scans | awk '{print $6}' | awk -F: '{print $1}' | sort | uniq -
c | sort -rn > scan_dest
cat all_scans | awk '{print $6}' | awk -F: '{print $2}' | sort | uniq -
c | sort -rn > scan_destports
echo "Top Scanners" > scan_report
echo "------------" >> scan_report
echo >> scan_report
echo "Sources" >> scan_report
echo "count   address" >> scan_report
echo "---------------" >> scan_report
head scan_source >> scan_report
echo >> scan_report
echo "Destinations" >> scan_report
echo "count   address" >> scan_report
echo "---------------" >> scan_report
head scan_dest >> scan_report
echo >> scan_report
echo "Destination Ports" >> scan_report
echo "count   address" >> scan_report
echo "---------------" >> scan_report
echo >> scan_report
head scan_destports >> scan_report
```

I manipulated the scan and oos logs in the same manner. I found that it was easiest for me to grep for specific pieces of the data and pull what I wanted out. No grand scripts from me, just easy tips that will hopefully help someone out.

References:

Snort.org. "The Open Source Network Intrusion Detection System." URL: www.snort.org (18 May 2004).

RFC. “Request for Comments.” URL: http://www.faqs.org/rfcs/rfc793.html (18 May 2004).

SnortSnarf. “Silicon Defense.” URL: http://www.silicondefense.com/software/snortsnarf/ (18 May 2004).

Van Oosterom, Peter. “GCIA Practical.” SANS Institute. URL: http://www.giac.org/practical/GCIA/Peter_Van_Oosterom.pdf (18 May 2004).

Neohapsis. “Port List.” URL: http://www.neohapsis.com/neolabs/neo-ports/ (18 May 2004).

IANA. “Port List.” URL: http://www.iana.org/assignments/port-numbers (18 May 2004).

Novell Inc. “iChain 2.3 Technical White Paper.” URL: http://www.novell.com/coolsolutions/icmag/features/a_ic23_whitepaper_ic.html#3 (18 May 2004).

King, Tom. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Tom_King_GCIA.pdf (18 May 2004)

Barrosa, David. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/David_Barroso_GCIA.pdf (18 May 2004).

Kan, Bernard. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Bernard_Kan_GCIA.doc (18 May 2004).

King, Tom. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Tom_King_GCIA.pdf (18 May 2004).

Morgan, Barbara. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Barbara_Morgan_GCIA.doc (18 May 2004).

Clark, Ronnie. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Ronnie_Clark_GCIA.doc (18 May 2004).

Rana, Antonia. “GCIA Practical.” SANS Institute. URL: www.giac.org/practical/GCIA/Antonia_Rana_GCIA.pdf (18 May 2004).

Montcalm, Erik. “GCIA Practical.” SANS Institute. URL: http://www.giac.org/practical/GCIA/Erik_Montcalm_GCIA.pdf (18 May 2004)

Neohapsis archives “SMB Name Wildcard.” URL: http://archives.neohapsis.com/archives/snort/2000-01/0222.html (18 May 2004).

Microsoft Corporation. "Configuring APIPA." URL: http://www.microsoft.com/resources/documentation/Windows/XP/all/reskit/en-us/Default.asp?url=/resources/documentation/Windows/XP/all/reskit/en-us/prjj_ipa_eiih.asp (18 May 2004).

Cahoon, Brian. "GCIA Practical." SANS Institute. URL: http://www.giac.org/practical/GCIA/Brian_Cahoon_GCIA.pdf (18 May 2004).

ISC. "SANS Internet Storm Center." URL: http://isc.sans.org (18 May 2004).

LURHQ. "LURHQ Threat Intelligence Group – Phatbot Trojan Analysis." URL: http://www.lurhq.com/phatbot.html (18 May 2004).

Dshield. "Dshield.org – Distributed Intrusion Detection System." URL: www.dshield.org (18 May 2004).