SANS/GIAC Intrusion Detection In Depth GCIA Practical Assignment

Chris Calabrese
December 2001

Table of Contents

Assignment 1 - Describe the State of Intrusion Detection Network IDS Probe Placement in Redundant Networks

Sections

Background and Introduction

The IDS world has finally gotten its arms around switched LAN's, but what about networks with redundant components? How do you know you're seeing all the attacks?

This paper examines common forms of network redundancy, strategies for placing network IDS probes in redundant networks, and the effectiveness of those strategies.

The goal here is to allow a single IDS probe to see all the traffic associated with a particular attack from one host to another regardless of how many packets the attack takes, how many network-level "sessions" the attack takes, or what network paths the packets traverse..

Redundant Network Elements

The following table gives a taxonomy of redundant network elements we'll consider in this paper:

Redundant Network Elements			
	Failover of network access	**Multiple points of network access**	**Multiple access, single address**
System/Node Availability	System with failover between between network interface cards (NIC's)	NIC multi-pathing through route advertisements	NIC multi-pathing through switch link-aggregation advertisements
Application Availability	Cluster with failover of IP address	Cluster with independent IP addresses (e.g., DNS secondaries)	Cluster with shared IP address (e.g, using a web server load balancer)
Network Availability	Multiple network switches	Parallel network routes	Cluster of transparent, stateful routing elements (i.e., load balanced firewalls)

Note that these elements are orthogonal and may be arbitrarily combined. However, it would be unusual to combine two elements from the same row, such as NIC failover and route-based NIC multi-pathing, since they're aimed at the same purpose.

The rest of this section serves as a brief introduction to how these elements operate, how they're useful, and, and their challenges for IDS sensor placement.

NIC failover

With NIC failover, an individual network node (typically a server or router) has multiple links to its LAN (or LAN's) through redundant NIC's that allow failover of the system's network address (es) from a primary NIC to a secondary NIC. This can ensure that the system continues operating in the face of a NIC failure.

In this scenario, failover software on the network node itself detects that the primary NIC is inoperative, de-configures it from use at the OS level, and configures the secondary NIC to take its place. In more sophisticated implementations, both the IP address and MAC address will fail over, allowing the secondary NIC to start talking on the network immediately. In less sophisticated implementations only the IP address will fail over, causing delays as ARP caches time out.

If we assume that it's highly unlikely for a NIC to fail in the middle of an attack, then the IDS placement challenge is to ensure that the traffic to/from both NIC's is covered, but not necessarily by the same IDS probe.

NIC multi-pathing through route advertisements

With NIC multi-pathing in general, an individual network node (typically a server or router) has multiple links to its LAN (or LAN's) through redundant NIC's operating in parallel. This allows higher network throughput compared to NIC failover.

With multi-pathing through route advertisements, the system has one IP address per NIC and advertises routes to its various IP addresses through each of its other IP addresses. Solaris 8, for example, does this by responding appropriately to Router Discovery Protocol requests[1].

The IDS challenge with multi-pathing through route advertisements is for one IDS probe to see all the network traffic regardless of which NIC the traffic is going to. Otherwise the IDS would have to do complex multi-probe correlations to construct a single attack.

Multi-pathing through switch link-aggregation advertisements

This is very similar to the multi-pathing through route advertisements, but here the system advertises paths to itself at the switching level instead of the routing level. The HP-UX implementation, for example, uses Cisco's Fast EtherChannel protocol to tell the switch how to send packets for the system's MAC address(es)[2].

The IDS challenge is the same as for multi-pathing through route advertisements.

Failover clustering

With failover clustering, network nodes (servers, routers, firewalls, etc.) can take over each other's network addresses when the primary address holder fails. This is similar to the failover NIC scenario discussed above, but with the failover happening at the entire-machine level rather than at the NIC level. As with NIC failover, just the IP address or the IP and MAC addresses can fail over depending on the sophistication of the software. This can ensure that a key service to continues operating in the face of a server failure.

If we assume that it's highly unlikely for a system to fail in the middle of an attack unless the result of the attack, then the IDS placement challenge is to ensure that the traffic to/from both systems is covered, but not that traffic to all cluster members must be covered by a single IDS probe.

Independent IP clustering

With independent IP clustering, each system in the cluster has its own unique IP address, and the client applications must decide how to distribute the load among the different servers. This is the basis of DNS clustering (the client choses among multiple DNS servers that are authoritative for the domain) and SMTP clustering (the client choses among multiple SMTP servers that have MX records in the domain)[3].

The IDS placement challenge with independent IP clustering is to ensure that a single probe sees all the traffic for a particular source/destination address pair. This may require multiple probes if the systems are geographically disparate.

Shared IP clustering

With shared IP clustering, an external load balancer is used to distribute requests sent to a shared/virtual network address to multiple servers. Each server in the cluster has their own unique/real IP address, and traffic between the load balancer and the real servers reflects these addresses rather than the shared address. This is how most popular web server load balancing systems work.

The IDS placement challenge with shared IP clustering is to ensure that a single probe sees all the traffic for a particular source/destination address pair (i.e., all traffic for a single attack). However, if we assume that the load balancer keeps all traffic from a particular source going to a particular destination as long as that destination is alive, and if we further assume that it's highly unlikely for a server to go down during an attack except as a result of the attack, than this collapses to all the traffic for a particular source regardless of which server the load balancer decided to send the traffic to.

Multiple switches

Merely having multiple switches handling the same network segment does not itself make the switches redundant or contribute to higher network availability. But multiple switches can be combined with redundant NIC's, failover routers, etc. so that the network will stay up in the event of a single switch failure.

The IDS placement challenge is to ensure that all data from all switches is seen by the IDS probes.

Parallel routes

In this scenario, a network may have redundant links internally (i.e., it is not a tree/hierarchy) or have multiple links to the outside world (i.e., multiple ISP links). This is typically used to ensure that network connectivity can survive a single link being down and also to increase network throughput.

The IDS placement challenge is to ensure that a single probe sees all the traffic that may flow between a particular source/destination pair so that multi-probe correlation is not necessary to construct a single attack.

Load-balanced firewalls

This is similar to the shared IP clustering scenario, but here the load balancers are placed both in front of and behind the firewall farm. This is used both to ensure that network connectivity will remain in the face of a firewall failure and to increase overall throughput through the firewall farm.

The IDS placement challenge is to ensure that a single probe sees all the traffic associated with a particular attack, and therefore a particular IP source/destination pair. However, similarly to the shared IP clustering scenario, load balancers keep all traffic for a particular source/destination pair going through the same firewall as long as the firewall is alive, and if we further assume that it's highly unlikely for a firewall to go down during an attack except as a result of the attack, than a single attack will be tied to a single firewall.

IDS Placement Strategies

In this section we'll consider the following placement strategies:

1. Single-segment IDS probe placed outside area of redundancy
2. Multiple single-segment probes
3. Multi-segment probe
4. Multiple multi-segment probes

Single-segment IDS probe placed outside area of redundancy

A single IDS probe is used, but carefully placed to avoid the problems introduced by redundant network elements. The probe could be a general-purpose system running a software IDS, a purpose-built IDS appliance, or a special purpose IDS card for a network switch such as Cisco's Catalyst 6000 IDS Module[4] or Intrusion, Inc.'s SecureCom 6000 series[5].

The challenge is finding a location where the IDS probe can see all the traffic. The table below shows where a probe might be located when used with different redundant network elements. Remember when reading the table, however, that the elements are orthogonal, so a location can only be used if it works for all the redundant elements in the network. For example, a switch SPAN port only works if you have a single switch or if the switches you have support multi-switch spanning.

Redundant Network Element	IDS Probe Location
Failover NIC's	Switch SPAN port
NIC multi-pathing	Switch SPAN port
Failover cluster	Switch SPAN port
Independent IP cluster	Switch SPAN port, if not geographically disparate
Shared IP cluster	In front of load balancer
Multiple switches	Multi-switch SPAN port
Parallel routes	Whatever part of the network is not parallel, if any
Load-balanced firewalls	In front of or behind entire cluster

A switch SPAN port is a network switch port setup so that that the switch copies all packets traveling through the swich (or a particular VLAN or set of VLAN's) to that port. This is sometimes referred to as port mirroring, though technically that refers to copying packets from one particular port to another, rather than copying all the packets from all the ports.

A multi-switch SPAN port refers to a SPAN port that sees data from multiple switches. In this way a single IDS probe can see span traffic from multiple switches. The poor man's version of this is to connect the SPAN port of one switch into another switch, but this can be cumbersome and cause performance problems and delays. Some high end switches such as the Cisco Catalyst 6000 family of switches have native support for inter-switch spanning over the normal inter-switch backbone connections[6].

Multiple single-segment probes

Next up the food chain is using multiple single-segment probes to cover multiple network segments. This gives us more flexibility to cover situations where failover network elements are used in conjunction with other redundant elements such as multiple switches. However, this technique still can not address issues where the traffic from a single attack may be sent along multiple paths that can't be spanned, such as multi-path NIC's on multiple switches that don't support multi-switch spanning. Having multiple probes also increases acquisition and management costs.

The following table summaries where multiple single-segment probes can offer an improvement over a single single-segment probe.

Redundant Network Element	Advantages/Disadvantages of Multiple Probes vs. Single Probe
Failover NIC's	Improves flexibility to work with other redundant elements
NIC multi-pathing	No difference
Failover cluster	Improves flexibility to work with other redundant elements
Independent IP cluster	Improves performance, ability to handle geographic diversity (locate by individual servers rather than in front of cluster)
Shared IP cluster	Improves performance (locate by individual servers rather than in front of cluster)
Multiple switches	No difference
Parallel routes	No difference
Load-balanced firewalls	Improves performance (locate by individual firewalls rather than in front of / behind cluster)

Multi-segment probe

An IDS probe that can listen on multiple network segments can cover arbitrarily complex redundant network scenarios as long as the whole network is in the same building. And less probes means lower management costs. But should we always jump to this design if a single single-segment IDS probe won't cut it, or are there situations where multiple single-segment probes would be more effective? Or even a combination of the single- and multi-segment probes. The issues are going to be:

- What is the acquisition cost vs. multiple single-segment probes?
- Can a single complex probe handle the traffic levels?

The answers to these questions depend greatly on exactly how the probe is constructed. We have two options:

- Probe with multiple NIC's
- Probe connected to a specialized switch

Let's look at these two options in more detail.

Probe with multiple NIC's

Not all NIDS software supports multiple NIC's. For example, SNORT can not look at more than one data stream at a time[7]. Only higher-end versions other systems may support this feature, especially on appliance-based probes, and the added cost may not be justified if other high-end features (such as probe performance) aren't required. Regarding performance, many IDS technologies are not able to make use of SMP technology and therefore are limited to roughly 100Mb/s total throughput [7]. Even systems that support SMP may have very high costs relative to performance due to non-linear scaling with the number of processors.

Probe connected to specialized switch

It is possible to connect IDS systems that do not support multiple NIC's to a specialized switch such as the Top Layer AppSwitch that feeds the IDS probe from multiple ports on the switch[8]. The cost of a Top Layer AppSwitch starts at around $10,000 at the time of this writing[9], so again, this may not be a cost effective solution. However, the AppSwitch is also capable of feeding multiple IDS probes and making sure that each network flow goes to one and only one probe, so this may be an excellent choice if very high performance is necessary.

To put this in our familiar format:

Redundant Network Element	Advantages/Disadvantages of Multi-Segment Probe vs. Multiple Single-Segment Probes
Failover NIC's	No difference
NIC multi-pathing	Increases effectiveness/flexibility
Failover cluster	No difference
Independent IP cluster	Decreases performance, ability to handle geographic diversity
Shared IP cluster	Decreases performance

Multiple switches	Increases effectiveness/flexibility
Parallel routes	Increases effectiveness/flexibility
Load-balanced firewalls	Decreases performance

Multiple multi-segment probes

Same advantages of performance as multiple single probes, but with the flexibility of multi-segment probes.

Redundant Network Element	Advantages/Disadvantages of Multiple Multi-Segment Probes vs. One Multi-Segment Probe
Failover NIC's	No difference
NIC multi-pathing	No difference
Failover cluster	No difference
Independent IP cluster	Increases performance, ability to handle geographic diversity
Shared IP cluster	Increases performance
Multiple switches	No difference
Parallel routes	No difference
Load-balanced firewalls	Increases performance

IDS Probe "Sweet Spots"

Application Redundancy (Clustering)	System Redundancy	Network Redundancy		
		Multiple Switches (no multi-switch port SPANning)	**Multiple Switches and Multiple Links / Segments**	**Multiple Switches, Multiple Links / Segments, and Firewall Load Balancer (high data rate)**
No Clustering or Failover Clustering	**Failover NIC's**	Simple probe	Multiple simple probes	Multiple simple probes

	NIC multi-pathing	Multi-segment probe	Multi-segment probe	Multiple multi-segment probes
Geographically Disparate Independent IP Cluster	**Failover NIC's**	Multiple simple probes	Multiple simple probes	Multiple simple probes
	NIC multi-pathing	Multiple multi-segment probes	Multiple multi-segment probes	Multiple multi-segment probes
Shared IP Cluster (high data rate)	**Failover NIC's**	Simple probe	Multiple simple probes	Multiple simple probes
	NIC multi-pathing	Multiple multi-segment probes	Multiple multi-segment probes	Multiple multi-segment probes

Conclusions

No matter what kind of bizarre network architecture you have, it's always possible to find some way of monitoring it with a network IDS. However more complex networks may require multiple single-segment probes, a multi-segment probe, or even multiple multi-segment probes.

References

1. Mark Garner, "IP Network Multipathing," Sun BulePrints OnLine, February 2001. Available from http://www.sun.com/blueprints/0201/Multipathing.pdf .
2. Hewlett Packard, " Cisco Systems and hp fast etherchannel and auto-port aggregation software ." Available from http://www.hp.com/products1/unixserverconnectivity/infolib/cisco.html .
3. Paul Albitz and Cricket Liu, *DNS and BIND, 4th Edition* , O'Reilly and Associates, 2001.
4. Cisco Systems, "Catalyst 6000 Intrusion Detection System Module Data Sheet." Available from http://www.cisco.com/warp/public/cc/pd/si/casi/ca6000/prodlit/6kids_ds.htm .
5. Intrusion.Com, "SecureCom Chassis Overview." Available from http://www.intrusion.com/products/productcategory.asp?lngCatId=21 .
6. Cisco Systems, "Configuring the Catalyst Switched Port Analyzer (SPAN) Feature." Available from http://www.cisco.com/warp/public/473/41.html .
7. Martin Roesch, comments made during SNORT BoF at SANS Network Security 2001 Conference, October 2001.
8. Gary Kessler, "IDS-in-Depth: Top Layer's AppSwitch filters a copy of traffic flows to downstream IDSeS," Information Security Magazine, August 2001.
9. Top Layer Networks, private correspondence, November 2001.

Assignment 2 - Network Detects

Detects

Detect 1 - Distributed FTP Scan

Trace Data:

```
10/26-06:59:12.559492 208.184.11.192:53290 -> www.xxx.yyy.4:21
TCP TTL:118 TOS:0x0 ID:629 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCCFD871B  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
10/26-06:59:12.572921 216.132.188.188:4275 -> www.xxx.yyy.10:21
TCP TTL:48 TOS:0x0 ID:8363 IpLen:20 DgmLen:44
******S* Seq: 0x2B9F9F0  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.653053 208.184.11.192:53291 -> www.xxx.yyy.14:21
TCP TTL:118 TOS:0x0 ID:632 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCCFE7532  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
10/26-06:59:12.735977 216.161.238.129:2740 -> www.xxx.yyy.12:21
TCP TTL:115 TOS:0x0 ID:35245 IpLen:20 DgmLen:44 DF
******S* Seq: 0x98C0740C  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.793004 204.101.16.163:2754 -> www.xxx.yyy.16:21
TCP TTL:49 TOS:0x0 ID:44376 IpLen:20 DgmLen:44
******S* Seq: 0x114C54B0  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.835849 216.158.34.123:1422 -> www.xxx.yyy.9:21
TCP TTL:117 TOS:0x0 ID:34740 IpLen:20 DgmLen:44 DF
******S* Seq: 0x513CFB0  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:15.059068 216.145.95.3:1171 -> www.xxx.yyy.8:21
TCP TTL:56 TOS:0x0 ID:21880 IpLen:20 DgmLen:44
******S* Seq: 0xA1DE0310  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:17.129919 208.184.11.192:53298 -> www.xxx.yyy.6:21
TCP TTL:118 TOS:0x0 ID:667 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCD1628E9  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

```
10/26-06:59:17.163196 216.124.39.10:3749 -> www.xxx.yyy.2:21
TCP TTL:113 TOS:0x0 ID:24128 IpLen:20 DgmLen:44 DF
******S* Seq: 0x118ADB90  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
```

1. Source of Trace:

http://www.incidents.org/archives/intrusions/msg02249.html

2. Detect was generated by:

Appears to be SNORT.

3. Probability the source address was spoofed:

Looking at the packets and grouping them by TTL, TCP Options, and Window size and then also looking at ISN's and IP ID's, it looks like there are probably a several source systems.

Plus the flags and values mentioned above don't look like crafted packets, and this type of scan is only useful if the attacker can see the return packets (routing attack, nearby system on broadcase media, etc.).

It's somewhat suspicious that all the source addresses are in 204.0.0.0, 208.0.0.0, and 216.0.0.0. But then these addresses are all in netblocks assigned to major web hosting companies, which are likely to be good "zombie" targets for attackers because they have lax security and high bandwidth.

10%

4. Description of attack:

Since we don't have a full packet dump, there's not much more we can say but that it's an FTP scan using fairly small payloads (24 and 28 bytes). Most likely the packets we're seeing are initial FTP login attempts.

5. Attack Mechanism:

The most likely scenario is that the attack tries to login to the FTP server and then exploits some direct FTP-based vulnerability that allows execution of arbitrary code such various well known FTP daemon buffer overflows (CVE-1999-0219, CVE-1999-0349, CVE-1999-0368, CVE-1999-0878 , CVE-1999-0879 , etc.), format string errors,"site exec" vulnerabilities, or shell-metacharacter vulnerabilities (CVE-1999-0080 , CVE-1999-0955 , CVE-1999-0097 , CVE-2000-0573, CVE-2001-0318, etc.). Or possibly drops files that can be used for subsequent attack through another service such as needed to exploit Microsoft FrontPage vulnerabilities (CVE-2001-0341, CAN-1999-1376, CAN-2000-0256).

Unfortunately not much more can be said without seeing the site's firewall logs and/or FTP server logs.

6. Correlations:

The attack I was able to find that was most similar was one identified by Sean Brown back in March in a posting to the Security Focus Incidents mailing list (http://archives.neohapsis.com/archives/incidents/2001-03/0266.htm l). This attack also appeared to come from a 20x.0.0.0 address (209.49.119.67), had a similarly spare number of packets involved, and had packets directed at the FTP service with 24-byte payloads (our attack uses 24- and 28-byte payloads). Perhaps we're now seeing a distributed version of this same attack. This particular attack attempts to login to the FTP server and create directory structures that can be used to exploit a Microsoft FrontPage Extensions vulnerability such as CVE-2001-0341 (not enough information is present to determine exactly what vulnerability it might be). Sean points out that one of the directory names, `/home/ftp/pub/313374159` , looks particularly hacker-ish (31337 is Elite in hacker-speak), though I'll also point out that 3.14159 is also the first few digits of PI, so perhaps this is meant to mean Eat Elite Pie or some such.

I was also able to find plenty of other attacks from these same net-blocks on http://www.dshield.org/subnet.php , giving credence to the idea that these netblocks are often used by attackers and that the IP addresses are not spoofed.

Finally, I'll point out that FTP (TCP/21) is the third most popular port to attack according to http://www.dshield.org/topports.html .

7. Evidence of active targeting:

The scan covered a fairly narrow range of addresses, and the attack has not been seen often, so it seems that this was likely actively targeted.

8. Severity:

Criticality - FTP servers are being targeted. I'll say 4.

Lethality - We don't know exactly what the purpose of the scan is, but given the number of buffer-overflow attacks on FTP servers and that FTP servers usually run with root/system priveleges, I'll guess 5.

System countermeasures - Again, we really don't know, but the site is running an IDS, so it must be at least somewhat sophisticated, but perhaps the systems are not owned by the same group that owns the IDS. I'll go with a conservative 3.

Network Coutnermeasures - We can't tell from the data whether there's an effective firewall block of this traffic. I'll guess 3.

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

9. Defensive measures:

Recommend audit of any systems that actually are running FTP to make sure they're properly patched against known FTP attacks. Implement better firewall rules to screen serves from receiving unnecessary traffic. Consider proxy-based firewall that can screen some attacks at the application-level (Raptor, CheckPoint with FTP Security Server turned on, etc.).

10. Multiple choice test question:

```
10/26-06:59:12.559492 208.184.11.192:53290 -> www.xxx.yyy.4:21
TCP TTL:118 TOS:0x0 ID:629 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCCFD871B  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
10/26-06:59:12.572921 216.132.188.188:4275 -> www.xxx.yyy.10:21
TCP TTL:48 TOS:0x0 ID:8363 IpLen:20 DgmLen:44
******S* Seq: 0x2B9F9F0  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.653053 208.184.11.192:53291 -> www.xxx.yyy.14:21
TCP TTL:118 TOS:0x0 ID:632 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCCFE7532  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
10/26-06:59:12.735977 216.161.238.129:2740 -> www.xxx.yyy.12:21
TCP TTL:115 TOS:0x0 ID:35245 IpLen:20 DgmLen:44 DF
******S* Seq: 0x98C0740C  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.793004 204.101.16.163:2754 -> www.xxx.yyy.16:21
TCP TTL:49 TOS:0x0 ID:44376 IpLen:20 DgmLen:44
******S* Seq: 0x114C54B0  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:12.835849 216.158.34.123:1422 -> www.xxx.yyy.9:21
TCP TTL:117 TOS:0x0 ID:34740 IpLen:20 DgmLen:44 DF
******S* Seq: 0x513CFB0  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:15.059068 216.145.95.3:1171 -> www.xxx.yyy.8:21
TCP TTL:56 TOS:0x0 ID:21880 IpLen:20 DgmLen:44
******S* Seq: 0xA1DE0310  Ack: 0x0  Win: 0x1C84  TcpLen: 24
TCP Options (1) => MSS: 1460
10/26-06:59:17.129919 208.184.11.192:53298 -> www.xxx.yyy.6:21
TCP TTL:118 TOS:0x0 ID:667 IpLen:20 DgmLen:48 DF
******S* Seq: 0xCD1628E9  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
10/26-06:59:17.163196 216.124.39.10:3749 -> www.xxx.yyy.2:21
TCP TTL:113 TOS:0x0 ID:24128 IpLen:20 DgmLen:44 DF
******S* Seq: 0x118ADB90  Ack: 0x0  Win: 0x2000  TcpLen: 24
TCP Options (1) => MSS: 146
```

How many distinct systems are MOST LIKELY to be launching the scan in the trace shown above?

a) One
b) Six
c) Eight
d) Ten

Answer: **c**

Note that when I examined this I first grouped the systems by TCP window size. Looking further, I realized that all the systems with size 0x4000 had the same IP address and had ID's and Seq's that looked like they really were the same system too, which gets us down to eight distinct systems. It looks like the systems with Win 0x2000 could also be the same system, which would make the number six, but I rejected this because the re-use of the same IP address for the 0x40000 systems makes non-reuse for the 0x2000 systems unlikely, and besides the ID's and Seq's don't line up quite as nicely either.

Detect 2 - Outbound on Port 20

Trace Data:

```
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:48:23.503613 0:60:9:C4:16:7A -> type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:63717 IpLen:
******S* Seq: 0x7C00A727  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 20 AF DC D7 D8 00 60 09 C4 16 7A 08 00 45 00  . .....`...z..E.
0x0010: 00 28 F8 E5 40 00 EE 06 DB A7 81 F7 BD F2 C7 4C  .(..@..........L
0x0020: B1 0B 00 14 00 15 7C 00 A7 27 00 00 00 00 50 02  ......|..'....P.
0x0030: 3F FF 94 50 00 00 00 00 00 00 00 00              ?..P........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:50:34.396615 type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:63485 IpLen:
******S* Seq: 0x7E3DF771  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 10 5A 0E 1A E9 00 60 09 C4 16 7A 08 00 45 00  ..Z....`...z..E.
0x0010: 00 28 F7 FD 40 00 EE 06 DC 86 81 F7 BD F2 C7 4C  .(..@..........L
0x0020: B1 14 00 14 00 15 7E 3D F7 71 00 00 00 00 50 02  ......~=.q....P.
0x0030: 3F FF 41 C0 00 00 00 00 B4 98 A0 BC              ?.A.........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:50:42.293008  type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xxx:21 TCP TTL:238 TOS:0x0 ID:5819 IpLen:
******S* Seq: 0x505702E7  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 A0 24 14 AF F0 00 60 09 C4 16 7A 08 00 45 00  ..$....`...z..E.
0x0010: 00 28 16 BB 40 00 EE 06 BD C8 81 F7 BD F2 C7 4C  .(..@..........L
0x0020: B1 15 00 14 00 15 50 57 02 E7 00 00 00 00 50 02  ......PW......P.
0x0030: 3F FF 64 30 00 00 00 00 00 00 00 00              ?.d0........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:51:06.654731 type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xxx.xx:21 TCP TTL:238 TOS:0x0 ID:30239 IpLen
******S* Seq: 0x8F270DE2  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 50 04 62 82 3E 00 60 09 C4 16 7A 08 00 45 00  .P.b.>.`...z..E.
0x0010: 00 28 76 1F 40 00 EE 06 5E 62 81 F7 BD F2 C7 4C  .(v.@...^b.....L
0x0020: B1 17 00 14 00 15 8F 27 0D E2 00 00 00 00 50 02  .......'......P.
0x0030: 3F FF 1A 63 00 00 00 00 00 00 00 00              ?..c........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:51:35.167840  type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:58759 IpLen:
```

```
******S* Seq: 0x4896A5B0  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 C0 4F 20 9B 00 00 60 09 C4 16 7A 08 00 45 00  ..O ...`...z..E.
0x0010: 00 28 E5 87 40 00 EE 06 EE F8 81 F7 BD F2 C7 4C  .(..@..........L
0x0020: B1 18 00 14 00 15 48 96 A5 B0 00 00 00 00 50 02  ......H.......P.
0x0030: 3F FF C9 24 00 00 00 00 00 00 00 00              ?..$........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:51:44.084966  type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xxx:21 TCP TTL:238 TOS:0x0 ID:2133 IpLen:
******S* Seq: 0x4E73DDB8  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 A0 24 05 EB 8B 00 60 09 C4 16 7A 08 00 45 00  ..$....`...z..E.
0x0010: 00 28 08 55 40 00 EE 06 CC 2A 81 F7 BD F2 C7 4C  .(.U@....*.....L
0x0020: B1 19 00 14 00 15 4E 73 DD B8 00 00 00 00 50 02  ......Ns......P.
0x0030: 3F FF 8B 3E 00 00 00 00 00 00 00 00              ?..>........
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:51:53.589853  type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:11653 IpLen:
******S* Seq: 0x921B6764  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 03 47 22 28 C5 00 60 09 C4 16 7A 08 00 45 00  ..G"(..`...z..E.
0x0010: 00 28 2D 85 40 00 EE 06 A6 F9 81 F7 BD F2 C7 4C  .(-.@..........L
0x0020: B1 1A 00 14 00 15 92 1B 67 64 00 00 00 00 50 02  ........gd....P.
0x0030: 3F FF BD E9 00 00 00 00 2A C4 5D 52              ?.......*.]R
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
[**] IDS6/misc_SourcePortTraffic-20-tcp [**]
10/18-12:52:04.997961  type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:23063 IpLen:
******S* Seq: 0x9AB71E4E  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 A0 24 1A 6F C4 00 60 09 C4 16 7A 08 00 45 00  ..$.o..`...z..E.
0x0010: 00 28 5A 17 40 00 EE 06 7A 66 81 F7 BD F2 C7 4C  .(Z.@...zf.....L
0x0020: B1 1B 00 14 00 15 9A B7 1E 4E 00 00 00 00 50 02  .........N....P.
0x0030: 3F FF FE 62 00 00 00 00 EA 0E C1 75              ?..b.......u
```

1. Source of Trace:

http://www.incidents.org/archives/intrusions/msg02176.html

2. Detect was generated by:

Appears to be SNORT.

3. Probability the source address was spoofed:

Packets do not appear to be crafted, and this type of scan won't work unless the attacker has some way of listening to the data for the spoofed address (routing attack, nearby system on broadcase media, etc.). 10%

4. Description of attack:

Attacking system sends binary data to victim's FTP port (TCP/21) from its own FTP-DATA port (TCP/20).

5. Attack Mechanism:

I considered the possibility that this could be a false postive where the IDS missed the intial FTP login, but rejected it because the FTP-DATA port should never talk to the FTP port in normal FTP usage.

So, this attack is most likely looking for FTP servers with buffer overflows or format string vulnerabilities on the initial login such as [CVE-1999-0028] or [CAN-2000-0843] . One good indication here is the number of null-bytes in the data. One puzzling aspect is that we see different data in each packet, but this is likely because it's trying signatures for a few different victim OS/hardware combinations or some such.

6. Correlations:

A search on the particular binary strings on Google revealed that there is some similarity to the payloads from `synscan2` (see for example http://www.incidents.org/archives/intrusions/msg01020.html). However other aspects of the packet are different enough to determine that it's definitely not the very same tool.

FTP (TCP/21) is the third most popular port to attack according to http://www.dshield.org/topports.html .

7. Evidence of active targeting:

The exact attack doesn't look like it's been seen before, so it seems that this was likely actively targeted.

8. Severity:

Criticality - Targets FTP servers. 4

Lethality - Looks like it's going for a buffer overflow or some such, and FTP runs as root. Assume 5

System countermeasures - Don't know. Assume 3.

Network Coutnermeasures - We're only seeing SYN's, so it's likely that a firewall is blocking the attack. 4

(4 + 5) - (3 + 4) = 2

9. Defensive measures:

Recommend audit of any systems that actually are running FTP to make sure they're properly patched against known FTP attacks. Also audit firewalls to make sure they effectively block traffic from FTP-DATA port that's not part of a legitimate FTP session.

10. Multiple choice test question:

```
10/18-12:48:23.503613 0:60:9:C4:16:7A -> type:0x800 len:0x3C
129.247.189.242:20 -> my.net.xx.xx:21 TCP TTL:238 TOS:0x0 ID:63717 IpLen:20 Dg
******S* Seq: 0x7C00A727  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
0x0000: 00 20 AF DC D7 D8 00 60 09 C4 16 7A 08 00 45 00  . .....`...z..E.
0x0010: 00 28 F8 E5 40 00 EE 06 DB A7 81 F7 BD F2 C7 4C  .(..@..........L
0x0020: B1 0B 00 14 00 15 7C 00 A7 27 00 00 00 00 50 02  ......|..'....P.
0x0030: 3F FF 94 50 00 00 00 00 00 00 00 00              ?..P........
```

The trace shown above is an example of:

a) Normal FTP login traffic
b) Normal FTP data traffic
c) A router access-list evasion scan
d) An IDS evasion scan

Answer: **c**

Detect 3 - Squid and `listen` Scan

Trace Data:

This trace contains nearly six thousand lines of trace data most of which looks roughly like:

```
Oct 29 00:44:03 hostmi snort: [ID 672207 auth.alert] INFO - Possible Squi
Oct 29 00:44:03 hostmi snort: [ID 672207 auth.alert] INFO - Possible Squi
Oct 29 00:44:04 hostmi snort: [ID 672207 auth.alert] INFO - Possible Squi
Oct 29 00:44:04 hostmi snort: [ID 672207 auth.alert] INFO - Possible Squi
```

And

```
Sep 23 15:34:42 hostj portsentry[737]: attackalert: Connect from host: 12
Sep 23 15:35:22 hostj portsentry[737]: attackalert: Connect from host: 12
Sep 23 15:54:37 hostj portsentry[737]: attackalert: Connect from host: 12
```

The folowing table summaries this data:

Date	Hits to Target Address			
	hostj		hosty	hostmi
	port 3128	port 2766	port 3128	port 3128
Sep 23	11	11	24	
Oct 1	4	5	4	
Oct 2	1	1	4	2561

Oct 4	6	6	20	
Oct 14	10	10	20	1
Oct 21	4	4	8	

All traces show the same source IP address.

1. Source of Trace:

I originally started with the http://www.incidents.org/archives/intrusions/msg02284.html section for the DRAGON INDUSTRIES netblock. However Laurie Zirkel of Virginia Tech was kind enough to supply me with the full trace.

2. Detect was generated by:

SNORT and Psionic Portsentry.

3. Probability the source address was spoofed:

We don't have enough information to see if the packets are likely crafted, but this type of scanning won't work unless the attacker has some way of listening to the data for the spoofed address (routing attack, nearby system on broadcase media, etc.). 20%

4. Description of attack:

Attacking system attempts to connect to TCP/3128 and TCP/2766, sends a large number of packets over a long period of time (>5000 connections over a three week period).

TCP/3128 is the default port for the Squid web proxy system.

TCP/2766 is assigned to the the Solaris `listen/nlps_server` , and is also a popular port for back-door Telnet servers (see http://www.cert.org/advisories/CA-2001-05.html).

5. Attack Mechanism:

We don't have much information on the individual packets, so there's not too much we can tell.

Possibilities include:

a. This Squid traffic is an attempt to root the Squid server via CVE-2001-0142 . However this does not fit the pattern of so many attempts to access the same target system.
b. This traffic is part of a Denial of Service. There is a well known Squid DoS using the Mkdir PUT command that fits the model here since it takes the proxy down for a few seconds (CAN-2001-0843). However, there is no similar DoS for the Solaris `listen/nlps_server` , and the Squid packets are not evenly distributed enough to look like a DoS.

c. The Squid proxy is being used to hide the identity of the attacking system in port-scanning other systems by taking advantage of the bug as described at http://www.securityfocus.com/archive/1/197727 , CAN-1999-1273 , and/or CAN-1999-1481 . This would explain the high number of Squid hits on `hostmi` on Oct 2.
d. The systems are being scanned for the TCP/2766 back-door described in (http://www.cert.org/advisories/CA-2001-05.html).
e. The systems are being scanned by something akin to `sscan`. As of the writeup in the January 1999 CERT advisory (http://www.cert.org/incident_notes/IN-99-01.html), this tool was able to scan for systems vulnerable to an expliotable buffer overflow in older versions of this service (http://packetstorm.decepticons.org/0008-exploits/nlps_server.c). However, this tool did not probe for Squid at all at the time of the CERT Advisory, and even if an extended version were written, this does not account for the high number of Squid hits on `hostmi` on Oct 2.

Most likely this traffic is a combination of some sort of scan that probes both these ports (or perhaps two scans running concurrently), and an exploit of an open Squid proxy for subsequent scanning that was stumbled upon in the Oct 2 scanning activity (and subsequently closed before the 14th).

6. Correlations:

A search on "Squid scan" on Google reveals that this is an extremely popular past-time. A small sampling of similar attack sitings include http://kang.sarang.net/snort/sig/sig2.html , http://www.incidents.org/archives/intrusions/msg01702.html , and http://archives.neohapsis.com/archives/incidents/2001-05/0198.html .

The TCP/2766 Trojan and and buffer overflow were popular enough to each garner their own CERT announcements (CA-2001-05 and IN-99-01 , respectively), but Google searches reveal that neither has been seen all that much in the wild. The Solaris `listen/nlps_server` buffer overflow one makes sense because it only affects particularly old and unpopular Solaris versions. But you would think that the Trojan resulting from the `snmpXdmid` buffer overflow attack would be pretty popular right now since it's only been around since March 2001 and there are references to it being in the wild in the CERT Advisory and an InfoSec News posting (http://lists.jammed.com/ISN/2001/04/0035.html).

Neither port is quite so popular as things like HTTP and FTP, however, according to http://www1.dshield.org/port_report.php .

7. Evidence of active targeting:

Considering that the same hosts were visited multiple times even though they were not vulnerable in most cases (the Portsentry tool blocks as well as reports), it seems pretty clear that this was an undirected scan.

8. Severity:

Squid Scan/Attack

Criticality - We know that one of the systems was a Squid proxy. We don't know what the others were. Assume 4.

Lethality - Allows subsequent attack on other systems, possibly bypassing firewall restrictions. Subsequent attacks could yield root. 5

System countermeasures - One of the systems appears to be vulnerable to this attack, which indicates it's probably not fully patched. Maybe 3.

Network Coutnermeasures - Network appears to allow the traffic. 2

(4 + 5) - (3 + 2) = 4

TCP/2766 Scan/Attack

Criticality - We know that one of the systems was a Squid proxy. We don't know what the others were. Assume 4.

Lethality - Allows root access. 5

System countermeasures - We said 3 in the Squid attack, but we also know that at least some of these systems are running Portsentry looking at this port, which makes system countermeasures for this attack somewhat better. 4

Network Coutnermeasures - Network appears to allow the traffic. 2

(4 + 5) - (4 + 2) = 3

9. Defensive measures:

Install appropriate Squid patches on Squid server. Install most recent patch bundles on Solaris systems. Setup firewall to block outside use of Squid proxy and TCP/2766.

10. Multiple choice test questions:

TCP/3128 Traffic

```
Sep 23 15:34:18  12.36.181.4:4374 -> z.y.x.34:3128
Sep 23 15:34:18  12.36.181.4:4374 -> z.y.x.34:3128
Sep 23 15:34:19  12.36.181.4:4374 -> z.y.x.34:3128
...
Oct 21 16:32:07  12.36.181.4:60668 -> z.y.x.66:3128
Oct 21 16:32:48  12.36.181.4:60668 -> z.y.x.66:3128
```

The trace shown above is MOST LIKELY an example of:

a) Normal web proxy traffic
b) A buffer overflow attack on a web proxy
c) A RingZero trojan searching for web proxy servers

d) Something other than RingZero searching for web proxy servers

Answer: **d**

TCP/2766 Traffic

```
Sep 23 15:34:42 hostj portsentry[737]: attackalert: Connect from host: 12.36.1
Sep 23 15:35:22 hostj portsentry[737]: attackalert: Connect from host: 12.36.1
Sep 23 15:54:37 hostj portsentry[737]: attackalert: Connect from host: 12.36.1
...
Oct 21 15:25:32 hostj portsentry[488]: attackalert: Connect from host: 12.36.1
Oct 21 16:32:07 hostj portsentry[488]: attackalert: Connect from host: 12.36.1
Oct 21 16:32:48 hostj portsentry[488]: attackalert: Connect from host: 12.36.1
```

Was the attack int the above trace successful?

a) Yes. The long term use of the same connection shows that the attacker is making use of the port.
b) Yes. The Portsentry tool has detected a buffer overflow.
c) No. Portsentry operates like TCP Wrappers and does not allow unauthorized access.
d) No. Portsentry replaces the real service running on the port, so there's nothing to attack.

Answer: **d**.

Detect 4 - Windows Registry Access

Trace Data:

```
4,1002660,1004732516,1004714516,2001/11/02,15:21:56,10008,8,\
7894,IN,OUT,5,3306,0,TCP/IP,aaa.bbb.cc0.101,aaa.bbb.cc1.101,\
2723,139,0.0.0.0,Windows Registry Access
```

1. Source of Trace:

Cisco Secure IDS (a.k.a., NetRanger) probe located in a DMZ network operated by the author's employeer.

2. Detect was generated by:

Cisco Secure IDS probe running Cisco IDS software release version 2.5. Specifically, this was triggered by signature 3306 for Windows Registry Access.

3. Probability the source address was spoofed:

The source addresses (`aaa.bbb.cc1.101`) listed here represents an application server that the target system talks to on a regular basis using Microsoft RPC's (such as those used for Windows registry access). Because the site employs firewalls that block similar access from other systems, it is highly likely that the access attempt came from the actual application server.

4. Description of attack:

Attempt to modify target systems' system configuration through Microsoft RPC calls to modify system registry values.

5. Attack Mechanism:

Our fear was that malicious software on the application server (i.e., a worm) was was attempting to modify remote registry settings to allow itself to propagate as implied may be possible in Microsoft Security Bulletin 00-008 .

A few well placed telephone calls, however, indicated that system administrators were using the Microsoft-supplied `regedit` tool to view/modify registry entries on one machine from another.

6. Correlations:

There are known vulnerabilities in network access to the Windows Registry (Microsoft Security Bulletin 00-008 , http://www.sans.org/newlook/alerts/NTE-bank.htm). And there are several well known attacks that use the Windows Registry locally (see for example http://www.incidents.org/cgi-bin/htsearch?method=and&config=htdig&words=registry). However, there don't seem to be many things out there that modify the registry remotely.

In fact, the only thing out there that does seem to try and access the registry remotely that is in any way malicious is the back-end code for those "Your internet connection isn't optimized" banner ads (see http://www.incidents.org/archives/intrusions/msg01163.html).

7. Evidence of active targeting:

Yes, it was actively targeted. By our own sysadmin's.

8. Severity:

Criticality - 4

Lethality - The sysadmin's had Administrator privelege, they conceivably could do a lot of damage, even if they didn't intend to. 5

System countermeasures - Slightly out-of-date though well-patched OS (NT 4) and remote registry access obviously allowed. 3

Network Coutnermeasures - Firewalls block access from other systems, but can't differentiate between the access needed to support the application and the access needed to remotely access the registry for the attacking system. 2

(4 + 5) - (3 + 2) = 4

9. Defensive measures:

Lock down remote registry access as per http://support.microsoft.com/support/kb/articles/Q153/1/83.ASP . Provide all NT system administrators with security awareness training. Develop or acquire NT lockdown programs to automate this type of lockdown in the future.

10. Multiple choice test question:

```
4,1002660,1004732516,1004714516,2001/11/02,15:21:56,10008,8,\
7894,IN,OUT,5,3306,0,TCP/IP,aaa.bbb.cc0.101,aaa.bbb.cc1.101,\
2723,139,0.0.0.0,Windows Registry Access
```

The target TCP port in the above trace appears to be

a) 1002660
b) 7894
c) 2723
d) 139

Answer: **d**

Detect 5 - NIMDA

Trace Data:

```
128.134.207.123 - - [04/Nov/2001:12:33:31 -0500] "GET /scripts/root.exe?/
128.134.207.123 - - [04/Nov/2001:12:33:35 -0500] "GET /MSADC/root.exe?/c+
128.134.207.123 - - [04/Nov/2001:12:33:35 -0500] "GET /c/winnt/system32/c
128.134.207.123 - - [04/Nov/2001:12:33:36 -0500] "GET /d/winnt/system32/c
128.134.207.123 - - [04/Nov/2001:12:33:36 -0500] "GET /scripts/..%255c../
128.134.207.123 - - [04/Nov/2001:12:33:37 -0500] "GET /_vti_bin/..%255c..
128.134.207.123 - - [04/Nov/2001:12:33:38 -0500] "GET /_mem_bin/..%255c..
128.134.207.123 - - [04/Nov/2001:12:33:38 -0500] "GET /msadc/..%255c../..
128.134.207.123 - - [04/Nov/2001:12:33:39 -0500] "GET /scripts/..%c1%1c..
128.134.207.123 - - [04/Nov/2001:12:33:40 -0500] "GET /scripts/..%c0%2f..
128.134.207.123 - - [04/Nov/2001:12:33:40 -0500] "GET /scripts/..%c0%af..
128.134.207.123 - - [04/Nov/2001:12:33:41 -0500] "GET /scripts/..%c1%9c..
128.134.207.123 - - [04/Nov/2001:12:33:41 -0500] "GET /scripts/..%%35%63.
128.134.207.123 - - [04/Nov/2001:12:33:42 -0500] "GET /scripts/..%%35c../
128.134.207.123 - - [04/Nov/2001:12:33:43 -0500] "GET /scripts/..%25%35%6
128.134.207.123 - - [04/Nov/2001:12:33:43 -0500] "GET /scripts/..%252f../
128.134.207.123 - - [04/Nov/2001:12:48:15 -0500] "GET /scripts/root.exe?/
128.134.207.123 - - [04/Nov/2001:12:48:16 -0500] "GET /MSADC/root.exe?/c+
128.134.207.123 - - [04/Nov/2001:12:48:19 -0500] "GET /c/winnt/system32/c
128.134.207.123 - - [04/Nov/2001:12:48:20 -0500] "GET /d/winnt/system32/c
128.134.207.123 - - [04/Nov/2001:12:48:21 -0500] "GET /scripts/..%255c../
128.134.207.123 - - [04/Nov/2001:12:48:21 -0500] "GET /_vti_bin/..%255c..
128.134.207.123 - - [04/Nov/2001:12:48:22 -0500] "GET /_mem_bin/..%255c..
128.134.207.123 - - [04/Nov/2001:12:48:23 -0500] "GET /msadc/..%255c../..
```

```
128.134.207.123 - - [04/Nov/2001:12:48:26 -0500] "GET /scripts/..%c1%1c..
128.134.207.123 - - [04/Nov/2001:12:48:33 -0500] "GET /scripts/..%c0%2f..
128.134.207.123 - - [04/Nov/2001:12:48:34 -0500] "GET /scripts/..%c0%af..
128.134.207.123 - - [04/Nov/2001:12:48:34 -0500] "GET /scripts/..%c1%9c..
128.134.207.123 - - [04/Nov/2001:12:48:38 -0500] "GET /scripts/..%%35%63.
128.134.207.123 - - [04/Nov/2001:12:48:39 -0500] "GET /scripts/..%%35c../
128.134.207.123 - - [04/Nov/2001:12:48:43 -0500] "GET /scripts/..%25%35%6
128.134.207.123 - - [04/Nov/2001:12:48:44 -0500] "GET /scripts/..%252f../
```

1. Source of Trace:

http://www.incidents.org/archives/intrusions/msg02358.html

2. Detect was generated by:

Nefarious behavior originally detected by SNORT. This trace comes from Apache web server access log.

3. Probability the source address was spoofed:

Near zero.

4. Description of attack:

Web server scan from system infected with NIMDA worm

5. Attack Mechanism:

From http://www.cert.org/advisories/CA-2001-26.html:

> The CERT/CC has received reports of new malicious code known as the "W32/Nimda worm" or the "Concept Virus (CV) v.5." This new worm appears to spread by multiple mechanisms:
>
> - from client to client via email
> - from client to client via open network shares
> - from web server to client via browsing of compromised web sites
> - from client to web server via active scanning for and exploitation of various Microsoft IIS 4.0 / 5.0 directory traversal vulnerabilities (VU#111677 and CA-2001-12)
> - from client to web server via scanning for the back doors left behind by the "Code Red II" (IN-2001-09), and "sadmind/IIS" (CA-2001-11 [Exploits) worms
>
> The worm modifies web documents (e.g., .htm, .html, and .asp files) and certain executable files found on the systems it infects, and creates numerous copies of itself under various file names.
>
> We have also received reports of denial of service as a result of network scanning and email propagation.

[The Microsoft IIS directory traversal vulnerabilities are CAN-2000-0884 and CAN-2001-0333, Code Red II exploits CAN-2001-0500, while the sadmind/IIS worm exploits CVE-1999-0977 and CAN-2000-0884.]

In this particular case, we're seeing evidence of the last two mechanisms.

6. Correlations:

Zillions of them. NIMDA is everywhere. See http://www.incidents.org/react/nimda.pdf . However, web searches revealed no other information on the 'net for the attacker's IP address.

7. Evidence of active targeting:

None.

8. Severity:

Criticality - Web servers being targeted. 4

Lethality - 5

System countermeasures - Site appears not to be running IIS, though that doesn't mean all the latest patches are applied, etc. 4

Network Coutnermeasures - Assume firewalls are in place, but do not appear to have application-level logic that can block web attacks. 2

(4 + 5) - (4 + 2) = 3

9. Defensive measures:

Consider the use of a firewall (Gauntlet, Raptor, CheckPoint with HTTP Security Service enabled) or add-on software (SecureIIS, AppSheild) that can filter HTTP requests at the application level.

10. Multiple choice test question:

```
128.134.207.123 - - [04/Nov/2001:12:33:31 -0500] "GET /scripts/root.exe?/c+dir
```

The above trace (part of a larger set) is MOST LIKELY to indicate:

a) A Code Red worm scanning for systems vulnerable to MS IIS directory traversal vulnerabilities.
b) That the target system is compromised and the attacker is coming back to use it as a zombie system.
c) A NIMDA worm scanning for systems infected by Code Red / Code Red II.
d) The target system is trying to infect an innocent web surfer with NIMDA.

Answer: **c**

Assignment 3 - "Analyze This" Scenario

Sections

Introduction and Executive Summary

The following represents my analysis of five days worth of Network IDS logs generated at the University during the period of Nov 15 - 19.

For the most part the analysis is presented in the order I actually performed it. The first few sections look at Scan, Alert, and Out-of-Spec logs in that order. In each of these sections I present overall counts (number of scanning hosts, number of alerts, etc.) as well as a more detailed analysis of various top-10 lists (top scanning hosts, top alerting hosts, etc.)

Next is an aggregate look at the IDS data, including total number of detects and top-10 talkers on MY.NET.

After that is a more in depth look at five hosts external to MY.NET, and we wrap up with a summary and recommendations.

Overall the University's security stance seems to have changed little since a recent analysis by Chris Baker, though things do vary in the details.

In particular, the Ramen worms that were running rampant a few months ago seem to be gone, they've been replaced wiht DoS activity and DHCP-based attacks.

Scan Analysis

Data Sources

The following files were used from http://www.research.umbc.edu/~andy:

scans.011115 scans.011118 scans.011120
scans.011116 scans.011119

Data Summary

Total Scans	`gzcat scan*.gz | grep -- "->" | wc -l`	501
Number of Source Hosts	`gzcat scan*.gz \` `| awk '$5 == "->" { print $4 }' \` `| cut -d : -f 1 | sort -u | wc -l`	
Number of Destination Hosts	`gzcat scan*.gz \` `| awk '$5 == "->" { print $6 }' \` `| cut -d : -f 1 | sort -u | wc -l`	21

Top Scan Destination Ports

```
gzcat scan*.gz \
| awk '$5 == "->" { print $6 }' \
| cut -d : -f 2 | sort | uniq -c | sort -rn | head
gzcat scan*.gz \
| awk '$5 == "->" { print $4 ":" $6 }' \
| cut -d : -f 1,4 > ports.out
grep ":$port" ports.out | cut -d : -f 1 \
| sort | uniq -c | sort -rn | head
```

Rank	Detects	Port	Use	Primary Source Systems	Comments
1	3919482	68	BOOTP/DHCP	MY.NET.5.75 MY.NET.5.76	These appear DHCP servers c but the amount of traffic they' quite surprising. Digging dee the traffic is also being sent v many of the "clients". Given malicious traffic identified fro below, they're likely "owning vulnerable to DHCP client bu CVE-1999-0814, CVE-2000- 2001-0181
2	208206	6970	RealAudio/ QuickTime streaming audio	Various in 205.188.0.0 netblock belonging to AOL.	Probably due to a really popu trailer or some such
3	197873	22	Secure Shell (SSH)	MY.NET.60.38 MY.NET.87.50 206.251.11.242	MY.NET.60.38 was identifie usage AFS system by Marc B which would account for the t

					MY.NET.87.50 shows massi traffic, but always from sourc and 999, which indicates that Database Protocol server (see http://www.freedb.org/). Thi look up meta-data about CD's copyright infringement. 206.251.11.242 is performing the entire MY.NET space. O universities have recently rep scale SSH scanning looking f vulnerabilities (see, for exam http://staff.washington.edu/di analysis.txt)
4	197873	27005	Half-Life game	MY.NET.87.50 MY.NET.160.114	MY.NET.160.114 appears to server as MY.NET.87.50 is.
5	58366	28800	MSN Gaming Zone	MY.NET.150.246 MY.NET.150.41 MY.NET.150.220 MY.NET.98.147 MY.NET.98.124	
6	41791	21	FTP	213.245.239.240 172.191.191.30 MY.NET.87.50	213.245.239.240 and 172.191 scanning MY.NET for FTP s
7	36391	1214	Unreal Tournament Server and KaZaA file sharing	MY.NET.98.136 MY.NET.98.197 MY.NET.98.165 MY.NET.98.175 MY.NET.98.196 MY.NET.97.248	Addressed Chris Baker's and analysis.
8	33958	6112	Battlenet game	Many systems	Gaming
9	24996	53	DNS	139.130.59.158 MY.NET.100.230	Also ranked 9 in Chris Baker' MY.NET.100.230 identified server in Chris Baker's analys 139.130.59.158 is scanning M DNS servers responding on T
10	24599	0	PING's and out-of-spec packets	209.190.237.123 64.50.168.74 211.106.159.132 61.134.9.121 61.150.5.18	These systems sent large num from their UDP/0 to the UDP MY.NET.70.134, MY.NET.1 MY.NET.190.30, MY.NET. MY.NET.153.194, MY.NET. MY.NET.152.160, MY.NET. MY.NET.111.221, MY.NET.

					MY.NET.153.177, MY.NET. MY.NET.153.187, and MY.N This could be evidence that t are "owned" by a Trojan oper In fact, at least one seems to b Back Orifice (see below).

Top Scan Source Hosts by Traffic

gzcat scan*.gz | awk '$5 == "->" { print $4 }' \
| cut -d : -f 1 | sort | uniq -c | sort -rn | head

Rank	Detects	Source Host	Comments
1	198152	MY.NET.87.50	Identified as CDDB server
2	194263	MY.NET.60.38	Heavy usage AFS system according to Marc Bayerkoh
3	41566	MY.NET.160.114	CDDB server
4	39705	205.188.233.185	Netblock owned by AOL according to ARIN and there likely web proxies
5	39418	205.188.233.121	
6	38672	205.188.233.153	
7	38572	205.188.244.57	
8	35437	213.245.239.240	Identified as scanning for FTP servers
9	30626	205.188.246.121	See AOL comment for ranks 4-7
10	23405	205.188.244.121	

Top Scan Source Hosts by Number of Destinations

```
gzcat scan*.gz | \
awk '$5 == "->" {
  gsub(/:.*$/, "", $4)
  gsub(/:.*$/, "", $6)
  print $4, $6 }' \
| sort -u | cut -d ' ' -f 1 \
| sort | uniq -c | sort -rn | head
```

Rank	Detects	Source Host	Comments
1	110136	MY.NET.60.38	Identified as AFS server
2	32275	MY.NET.87.50	Identified as CDDB server
3	14216	213.245.239.240	Identified as scanning for FTP servers
4	7927	139.130.59.158	Appears to be scanning for DNS servers on TCP/53

5	7215	MY.NET.160.114	Identified as CDDB server
6	5441	172.191.191.30	Identified as weeping MY.NET for FTP servers.
7	4287	MY.NET.100.230	Identified as DNS server by [Chris Baker]
8	3772	206.251.11.242	Identified as scanning for SSH servers
9	3471	MY.NET.111.157	Heavy traffic to TCP/3646 indicates Gnutella client
10	2710	MY.NET.5.75	Identified as DHCP attacker

Top Scan Destination Hosts by Traffic

```
gzcat scan*.gz | awk '$5 == "->"
   { print $6 }' \
| cut -d : -f 1 | sort \
| uniq -c | sort -rn | head
```

Rank	Detects	Destination Host	Comments
1	28958	24.164.45.163	Google search reveals this is a popular server for AG, Quake, and other games
2	18270	MY.NET.6.7	Identified as heavy AFS server by [Chris Baker]
3	15956	MY.NET.70.134	Identified as Port 0 destination
4	13870	202.130.4.1	Portscanned by MY.NET.98.240
5	11032	24.254.241.95	Looks like a CDDB client of MY.NET.87.50
6	10341	MY.NET.146.15	Heavy usage from 205.188.0.0 to UDP/6970 . UDP/69 identified as streaming media, and 205.188 is an AOL proxy, so probably benign. However, MY.NET.178.115 was identified as a Port 0 destination, and also shows heavy usage on UDP/1144
7	10038	MY.NET.184.23	
8	9527	MY.NET.109.62	
9	9367	MY.NET.178.115	
10	9327	MY.NET.145.197	

Alert Analysis

Data Sources

The following files were used from http://www.research.umbc.edu/~andy:

alert.011115 alert.011116 alert.011118 alert.011119 alert.011120

Data Summary

Total Alerts	`grep -E -- "(->|spp_portscan)" alert* \ | wc -l`	109
Number of Source Hosts	`grep '\[\*\*\]' alert* \ | awk ' /spp_portscan/ { if($7 ~ /from/) print $8 else print $7 next } { a=NF-2; print $a } ' \ | cut -d : -f 1 | sort \ | uniq -c | sort -rn > source.hosts wc -l source.hosts`	2
Number of Destination Hosts	`grep -- '->' alert* \ | awk '{ print $NF }' \ | cut -d : -f 1 | sort \ | uniq -c | sort -rn > dst.hosts wc -l dst.hosts`	1

Top Alerts

```
# portscan counts
grep spp_portscan alert* | wc -l
# non portscan counts
grep '\[\*\*\]' alert* \
| grep -v spp_portscan \
| cut -d \] -f 2 | cut -d \[ -f 1 \
| sort | uniq -c | sort -rn | head
# source hosts
grep "$alert_text" alert* \
| awk '{ a=NF-2; print $a}' \
| cut -d : -f 1 | sort \
| uniq -c | sort -rn | head
# dest hosts
grep "$alert_text" alert* \
| awk '{ print $NF}' \
| cut -d : -f 1 | sort \
| uniq -c | sort -rn | head
```

Rank	Detects	Alert	Primary Sources	Primary Destinations	Comments
1	413272	Portscan	Covered in Scan analysis above		
2	118974	MISC Large UDP Packet	209.190.237.123 61.150.5.18 61.153.17.24 61.150.5.19	MY.NET.70.134 MY.NET.111.221 MY.NET.153.187 MY.NET.53.40	These source addresses a sorts of bad things like P possible Red Worm activ See above and below. MY.NET.70.134, MY.N

					and MY.NET.153.187 i Port 0 destinations
3	93077	Tiny Fragments - Possible Hostile Activity	MY.NET.8.1	MY.NET.16.42	Likely some kind of DoS source also doing NMAI SMB Wildcard scans
4	81415	MISC source port 53 to <1024	194.90.1.5 134.93.19.12 192.115.189.100 199.203.1.20	MY.NET.1.3 MY.NET.1.5 MY.NET.1.4 MY.NET.88.88	Given that source port 5 < 1024 can be chosen by if no DNS server is runn systems are running thei as "root", and no other al been generated by these guessing this is not malic behavior.
5	62313	MISC traceroute	No issue in this environment		
6	54096	WEB-MISC prefix-get //	132.250.170.55	MY.NET.253.114	132.250.170.55 is in a ne owned by the Naval Res Laboratory, and there's n interesting traffic for this this is likely some kind o positive, especially giver amount of traffic to one since this type of attack v ususually be used only a against a single host to b
7	42463	INFO MSN IM Chat data			
8	40424	CS WEB-SERVER - external web traffic	No issue in this environment		
9	35367	SMB Name Wildcard	216.150.152.145 MY.NET.163.53 MY.NET.239.78 MY.NET.217.42	MY.NET.5.45 MY.NET.5.44	216.150.152.145 is also i portscanning, though mo SMB traffic is going to t destination addresses list which are also both Wat as identified below. The others appear to be t SMB systems.

					Also MY.NET.217.42 w as a primary destination : Watchlist traffic by Chri: now we're seeing it as a s address. It also seems to overly popular customer MY.NET.5.75 DHCP att system is probably "own being used as a jumping subsequent attacks.
10	32156	ICMP Echo Request BSDtype	No issue in this environment		

Most Severe Alerts

In chosing which alerts were the most severe, I went with the follwing logic:

- Watchlist alerts were given highest consideration because they indicate a history of malicious behavior
- Alerts indicating attacks on the network infrastructure (DNS, NTP) were given the next highest consideration
- Next I considered alerts indicating Trojan/Worm behavior, but dropped alerts that said "Probable"
- Finally, I added the "Virus" alerts, which rounded out the list

Alert Count	Alert	Primary Sources	Primary Destinations	Comments
6495	Watchlist 000220 IL-ISDNNET-990517	MY.NET.5.45 MY.NET.5.44 MY.NET.5.118	MY.NET.153.196 MY.NET.253.125 MY.NET.60.14	The source systems here all receiving heavy SMI Name Wildcard traffic above. They also seem t popular targets probes related to Windows syst Perhaps they are publicl known servers
4422	Watchlist 000222 NET-NCFC	159.226.61.72 159.226.45.204	MY.NET.253.114 MY.NET.6.7	159.226.61.72 appears a HTTP client of MY.NET.253.114. 159.226.45.204 was alsc

				identified for this same a by Chris Baker as a sour of Telnet attempts to MY.NET.6.7. That is exactly the traffic we see here. 1012 telnet connections in five days seems like a lot, though.
1044	BACKDOOR NetMetro Incoming Traffic	204.178.125.65	MY.NET.163.111	204.178.125.65 is chessclub.com, which se to operate various ches related services. Either chessclub.com is "owne one of its services opera whole lot like NetMetro
16	DDOS shaft client to handler	207.25.71.5	MY.NET.201.10	The source machine her www11.cnn.com, and th are no other detects for t source, so this is likely a false positive
2	IDS50/ trojan_trojan-active-subseven	MY.NET.70.148 MY.NET.130.86	216.167.107.65 204.152.184.75	MY.NET.70.148 appear be a well known (and scanned/attacked) anon server, here it's talking t 204.152.184.75, which i ftp.netbsd.org, so this is liklely a false positive. The MY.NET.130.86 -> 216.167.107.65 traffic c be real, but there's not m evidence one way or the other.
1	Virus - SnowWhite Trojan Incoming	MY.NET.6.59	12.78.116.254	This source address has off a bunch of other aler too, including Possible s Worm, Queso fingerprin Possible pif Worm, SMI Name Wildcard, Possibl MyRomeo Worm, and N Scan. This guy is bad n
1	Virus - Naked Wife	MY.NET.6.39	66.92.218.188	MY.NET.6.39 was ident as an AFS host by Chris Baker

				However we're now seei alerting for Synscan Portscan, Queso fingerprinting, Possible MyRomeo Worm, Possi shsWorm, Possible scr Worm. This doesn't seem norma activity for an AFS serv
1	EXPLOIT NTPDX buffer overflow	64.50.168.74	MY.NET.190.30	These pairs seem to be generating all kinds of a and such. These may be targeted attacks
1	DNS SPOOF query response with ttl	207.245.122.2	MY.NET.137.7	
1	Back Orifice	209.190.237.123	MY.NET.70.134	

Top Alerters

```
grep '\[\*\*\]' alert* \
| awk '/spp_portscan/ { print $7; next }
    { a=NF-2; print $a }' \
| cut -d : -f 1 | sort \
| uniq -c | sort -rn | head
```

Rank	Detects	Host	Comments
1	102251	MY.NET.5.75	Identified as DHCP attackers. Also seem to be doing portscans, sending a lot of ICMP Destination Unreachab (may be related to DHCP vulnerability CVE-1999-0875 and attempting various myserver, IIS, and CGI attacks
2	100293	MY.NET.5.76	
3	93069	MY.NET.8.1	Scanning with NMAP, light SMB wildcard scanning (ba on NMAP results?), and 93070 Tiny Fragments to MY.NET.16.42 MY.NET.16.41, in turn, is exhibiting its own portscanin SMB wildcard scanning, possible trojan activity, possibl Red Worm activity, web attacks, etc. Is nobody innocent?
4	49802	209.190.237.123	Identified as Port 0, large UDP, and Back Orifice source Also triggering 'possible Red Worm' alerts

5	49686	MY.NET.87.50	Identified as CDDB server. Looks like it's also running Anywhere
6	22840	61.150.5.18	Identified as Port 0 and large UDP source. Also seems t sending incomplete fragments.
7	20933	61.150.5.19	Large UDP source and sending incomplete fragments
8	19594	61.153.17.24	Identified as large UDP source
9	15867	61.134.9.88	Large UDP's and incomplete fragments.
10	14281	216.150.152.145	Identified as Watchlist host, portscanner, and SMB Wilc user

---

Out of Spec Analysis

Data Sources

The following files were used from http://www.research.umbc.edu/~andy:

oos_Nov.15.2001 oos_Nov.17.2001 oos_Nov.19.2001
oos_Nov.16.2001 oos_Nov.18.2001 oos_Nov.20.2001

Data Summary

Total Out of Spec Packets	`grep -- '->' oos* | wc -l`

Number of Source Hosts	`cat oos* \`
	 `| awk '$3 == "->" { print $2 }' \`
	 `| cut -d : -f 1 | sort -u | wc -l`

Number of Destination Hosts	`cat oos* \`
	 `| awk '$3 == "->" { print $4 }' \`
	 `| cut -d : -f 1 \`
	 `| sort -u | wc -l`

Top out of Spec Source Hosts

```
cat oos* \
| awk '$3 == "->" { print $2 }' \
| cut -d : -f 1 | sort \
| uniq -c | sort -rn | head
```

Rank	Detects	Source Host	Comments
1	154	199.183.24.194	vger.kernel.org - probably running Linux TCP stack wit experimental protocol support.

2	88	203.162.5.21	Web, DNS, and SNMP, etc. No other correlations. Probably running Linux stack.
3	61	141.99.131.88	
4	15	66.114.106.22	
5	14	212.51.220.121	
6	4	24.0.238.221	
7	4	213.249.157.121	
8	4	202.130.239.149	
9	3	64.76.131.4	
10	3	64.192.166.217	

Aggregate Analysis

Data Summary

```
gzcat scan*.gz | awk '$5 == "->"
{ print $4 }' \
| cut -d : -f 1 > tmp
grep '\[\*\*\]' alert* | grep -v spp_portscan \
| awk '{ a=NF-2; print $a }' \
| cut -d : -f 1 >> tmp
cat oos* \
| awk '$3 == "->" { print $2 }' \
| cut -d : -f 1 >> tmp
```

Total Snort Detects	`wc -l tmp`	1071
Number of Source Hosts	`sort -u tmp | wc -l`	2

Top Talkers on MY.NET

```
grep MY.NET tmp | sort | uniq -c | sort -rn | head
```

Rank	Number of Snort Entries	Host	Comments
1	2253465	MY.NET.5.75	Identified as portscanners, DHCP attackers, and server attackers. These two systems represent 2 of total Snort detects!
2	1699956	MY.NET.5.76	
3	198155	MY.NET.87.50	Identified as CDDB server
4	194397	MY.NET.60.38	Identified as AFS server
5	93069	MY.NET.8.1	Identified as Scanning with NMAP, light SMB wildcard scanning, and DoSing MY.NET.16.42

6	41567	MY.NET.160.114	Identified as CDDB server
7	21536	MY.NET.150.246	
8	16958	MY.NET.150.220	Identified as MSN Gaming Zone users
9	16285	MY.NET.150.41	
10	14054	MY.NET.100.230	Identified as DNS server

Relationships Between MY.NET Hosts Mentioned Elsewhere in this Document

```
tr ' '
' < this.doc.txt | tr -d ',' | grep MY.NET | sort -u > mynet.nodes
gzcat scan*.gz | fgrep -f mynet.nodes \
| awk '{print $4 ":" $6 }' | \
cut -d : -f 1,3 > tmp
cat alert* oos* | fgrep -f list \
| awk '{ a=NF-2; print $a ":" $NF }' \
| cut -d : -f 1,3 >> tmp
awk < tmp -F : '
 BEGIN {
    while(getline < "list")
        want[$0]++
 }
 want[$1] > 0 && want[$2] > 0 {
    print $0
 }' \
| sort tmp1 | uniq -c | sort -rn > mynet.edges
```

Detects	Source	Destination
3701	MY.NET.5.76	MY.NET.201.10
1871	MY.NET.5.75	MY.NET.217.42
1706	MY.NET.5.75	MY.NET.239.78
1059	MY.NET.16.42	MY.NET.1.3
182	MY.NET.16.42	MY.NET.1.4
126	MY.NET.16.42	MY.NET.1.6
4	MY.NET.53.40	MY.NET.53.40
2	MY.NET.239.78	MY.NET.253.125
1	MY.NET.53.39	MY.NET.239.78
1	MY.NET.239.78	MY.NET.53.39
1	MY.NET.217.42	MY.NET.97.248
1	MY.NET.16.42	MY.NET.137.7

Relationships Between MY.NET Hosts Mentioned Elsewhere in this Document:

Image of Relationships Graph

"Interesting" External Hosts

Host Selection

Hosts were selected by taking the first five unique non-MY.NET addresses from the Alerts Analysis section that appeared not to be false positives.

These are 204.178.125.65, 64.50.168.74, 207.245.122.2, 209.190.237.123, and 61.150.5.18.

Analysis Methods

As shown above, direct analysis of the data was performed using standard Unix tools such as `grep`, `awk`, `sort`, `uniq` , and `head`.

Analysis for 204.178.125.65

Internic Whois Record

The following netblock was sub-allocated from a netblock belonging to UUNET

Technologies

```
Sleator Games, Inc. (NETBLK-UU-204-178-125-64)
   5001 Baum Blvd. Suite 630
   Pittsburgh, PA 15213
   US
   Netname: UU-204-178-125-64
   Netblock: 204.178.125.64 - 204.178.125.79
   Coordinator:
      Luce, Doug A.  (DAL-ARIN)  doug@LM.NET
      412-688-3200 (FAX) 412-688-3211
   Record last updated on 26-Jul-1999.
   Database last updated on  2-Dec-2001 19:54:36 EDT.
```

DNS Records

```
www.chessclub.com       internet address = 204.178.125.65
*** No hostname information is available for "204.178.125.65"
```

Unique Alerts

1027 BACKDOOR NetMetro Incoming Traffic

Analysis

Earlier in the week, this IP address pointed to www.chessclub.com. Now the forward mapping exists, but the reverse mapping is gone. That's somewhat suspicious if you ask me.

On the other hand, Netcraft says this is a FreeBSD system, whereas NetMetro is Windows only according to http://www.glocksoft.com/trojan_list/Net_Metropolitan.htm. So this appears a false positive after all.

Analysis for 64.50.168.74

Internic Whois Record

```
CapuNet, LLC (NETBLK-CAPUNET-BLK-CIDR1)
   6000 Executive Blvd. Suite 600
   Rockville, MD 20852
   US
   Netname: CAPUNET-BLK-CIDR1
   Netblock: 64.50.128.0 - 64.50.223.255
   Maintainer: CAPU
   Coordinator:
      Dvorak, John  (JD707-ARIN)  noc@capu.net
      301-881-4900
   Domain System inverse mapping provided by:
   NS.CAPU.NET                     64.50.128.2
   NS2.CAPU.NET                    64.50.128.6
   NS3.CAPU.NET                    64.50.128.10
```

```
ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
Record last updated on 23-Jul-2001.
Database last updated on  2-Dec-2001 19:54:36 EDT.
```

DNS Record

```
Name:    cd-168-74.ra30.dc.capu.net
Address:  64.50.168.74
```

Unique Alerts

```
2362  portscan
  25  ICMP Fragment Reassembly Time Exceeded
  11  High port 65535 udp - possible Red Worm - traffic
   1  MISC source port 53 to <1024
   1  EXPLOIT NTPDX buffer overflow
```

Analysis

capu.net is a recently defunct web hosting provider according to www.capu.net. According to Netcraft, they ran IIS 5.0 on Win2K.

It's somewhat doubtful that that the problem here is a Code Red worm since this system was identified as sending a lot of Port 0 -> Port 0 traffic. In fact, it doesn't look like it's a worm at all since the so-called portscans are actually repeated traffic to two systems (MY.NET.178.115 and MY.NET.190.30), and neither of these systems is exhibiting similar alert behavior.

The top source/dest port pairs in this traffic (out of a total 653 pairs) are:

Count	Source:Dest Port	Comments
2329	0:0	Port 0 is reserved and not accessible through OS API's.
1809	1571:1144	1571 is Orbix
276	1383:1122	1383 is the Hanaway Network License Manager
184	0:34878	
35	8448:38001	38001 may be related to the game Mod Monkey
14	7000:7001	AFS or BBS related
9	67:68	DHCP
8	8448:37997	
3	8448:38000	
3	8448:79995	

This traffic definitely looks a bit suspicious given that port 0 is reserved and is generally not accessible through standard OS `socket` () routines. On the other hand, it doesn't seem to match any known malware, so I'm inclined to suspect it's not malicious, or at least that this isn't the "attack" phase of the malicious activity (it could be some kind of zombie that operates primarily on port 0 or some such I guess).

Given the amount of traffic, I'm going to guess this is some type of DoS. The presence of DHCP traffic also backs this up as, based on anecdotal evidence of recent DoS attacks against Ray Sundland, a collegue of mine, sending bogus/unrequested DHCP "replies" seems to be a favorite DoS in the wild right now.

Analysis for 207.245.122.2

Internic Whois Record

```
Consult Dynamics, Inc. (NETBLK-DCAN-000)
   1204 West Street
   Wilmington, DE 19801
   US
   Netname: DCAN-000
   Netblock: 207.245.64.0 - 207.245.127.255
   Maintainer: DCAN
   Coordinator:
      White, Andrew J.  (AW99-ARIN)  abuse@DCA.NET
      +1-302-654-1019 (FAX) +1-302-426-1568
   Domain System inverse mapping provided by:
   NS1.DCA.NET                    204.183.80.2
   NS2.DCA.NET                    207.245.82.2
   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
   Record last updated on 07-May-2001.
   Database last updated on  2-Dec-2001 19:54:36 EDT.
```

DNS Record

```
*** No hostname information is available for "207.245.122.2"
```

Going to this site via the web redirects the browser to www.emeron.com, which is 216.158.50.170.

According to Netcraft, the site www.emeron.com is running Microsoft-IIS/5.0 on Windows 2000.

Unique Alerts

```
57  SMB Name Wildcard
54  ICMP Echo Request L3retriever Ping
28  portscan
 5  ICMP Echo Request Nmap or HPING2
 1  DNS SPOOF query response with ttl
```

According to information in the ArachNIDS database on www.whitehats.com , L3 Retriever is a legitimate testing and scanning tool that produces large (1500 byte) ping packets.

Analysis

What's most interesting here is that alerts for this system started out with MY.NET.137.7 scanning this host for a few hours before it turned around and scanned back. After that MY.NET.137.7 stepped up its scanning activities, and this host finally shot back with the SMB wildcards almost two days later.

At least on the surface, this traffic from this system seems to be a tit-for-tat counter-attack, with neither system actually breaking into the other.

Again, since we don't have packet dumps, there's not much more we can say.

Analysis for 209.190.237.123

Internic Whois Record

```
     Atlantech Online, Inc. (NETBLK-AOI1999B)
        1010 Wayne Avenue, Suite 630
        Silver Spring, MD 20910
        US
        Netname: AOI1999B
        Netblock: 209.190.192.0 - 209.190.255.255
        Maintainer: ATON
        Coordinator:
           Center, Network Operations  (EF105-ARIN)  noc@atlantech.net
           301-589-3060 (FAX) 301-593-9897
        Domain System inverse mapping provided by:
        DNS1.ATLANTECH.NET           209.183.205.35
        DNS2.ATLANTECH.NET           209.183.192.65
        ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
        Record last updated on 22-May-2000.
        Database last updated on  3-Dec-2001 19:56:03 EDT.
```

DNS Record

```
Name:    7b.edbed1.client.atlantech.net
Address:  209.190.237.123
```

Unique Alerts

```
45076 MISC Large UDP Packet
4669  portscan
  55  High port 65535 udp - possible Red Worm - traffic
   1  ICMP Fragment Reassembly Time Exceeded
   1  Back Orifice
   1  Attempted Sun RPC high port access
```

All this is going to MY.NET.70.134.

Port Pairs

Top port pairs look very similar to the pairs for 64.50.168.74.

```
22841 0:0
 2141 0:34878
 1846 1571:1144
 1363 7000:7001
  612 1410:1606
  292 3902:1285
  204 4718:1560
  180 8448:38001
  168 67:68
  163 19:63980
```

Analysis

This looks almost identical to the traffic from 64.50.168.74, which we've said is probably a DoS.

Analysis for 61.150.5.18

Internic Whois Record

```
inetnum                    61.150.0.0 - 61.150.31.255
netname                    SNXIAN
descr                      xi'an data branch,XIAN CITY SHAANXI PROVINCE
country                    CN
admin-c                    WWN1-AP, inverse
tech-c                     WWN1-AP, inverse
mnt-by                     MAINT-CHINANET-SHAANXI, inverse
mnt-lower                  MAINT-CN-SNXIAN, inverse
changed                    ipadm@public.xa.sn.cn 20010309
source                     APNIC
person                     WANG WEI NA, inverse
address                    Xi Xin street 90# XIAN
country                    CN
phone                      +8629-724-1554
fax-no                     +8629-324-4305
e-mail                     xaipadm@public.xa.sn.cn, inverse
nic-hdl                    WWN1-AP, inverse
mnt-by                     MAINT-CN-SNXIAN, inverse
changed                    wwn@public.xa.sn.cn 20001127
source                     APNIC
```

DNS Record

```
*** No hostname information is available for "61.150.5.18"
```

Unique Alerts

```
21853 MISC Large UDP Packet
 850  portscan
```

```
 647  ICMP Fragment Reassembly Time Exceeded
 135  Incomplete Packet Fragments Discarded
   2  High port 65535 udp - possible Red Worm - traffic
```

Destinations

```
18055 MY.NET.111.221
 2250 MY.NET.153.177
  692 MY.NET.53.36
  647 61.150.5.18
  363 MY.NET.153.198
  245 MY.NET.152.44
  195 MY.NET.153.187
  109 MY.NET.53.37
   56 MY.NET.53.40
   25 MY.NET.152.19
```

Port Pairs

```
8711 1843:1446
3958 2261:2335
2601 0:0
2577 2670:4554
1976 2848:2061
 212 4491:2053
 210 4485:1232
 177 1168:1386
 152 2997:2418
```

Analysis

Yet another thing operating on port 0 with little other information to recommend it as some particular tool. More DoS traffic?

Summary and Recommendations

While the University seems to have solved the Ramen worm problem, there's still plenty of nasty stuff out there.

Compromised or Malicious Hosts

The following hosts have been identified in this analysis as malicious and/or likely compromised. These hosts require further investigation:

```
MY.NET.130.86
MY.NET.163.53
MY.NET.178.115
MY.NET.217.42
MY.NET.239.78
MY.NET.5.75
```

```
MY.NET.5.76
MY.NET.6.39
MY.NET.6.59
MY.NET.8.1
MY.NET.87.50
```

Recommendations

I'll start by reiterating some of the recommendations from Chris Baker's recent analysis, though a few with new twists.

- Maintain a known offenders list (it appears that this might actually be done given the Watchlist alerts)
- Implement more restrictive firewalling. Although it's not realistic for a University to implement a default-deny policy, it should at least be possible to block known offenders. It may also be possible to block protocols that should never come in from the outside. UDP/0 comes to mind here, as does DHCP.
- Keep patches up to date on University servers, implement centralized logging, etc.

I'll also add...

- Consider programs to make it easier for students and University departments to keep their systems secure, such as University-coordinated anti-virus updates and University-distributed CD-ROM's containing patch-bundles for popular operating systems.
- Consider security awareness programs for Students and University system administrators. On the student side, this can be part of the student orientation. On the system-admin side it can include such things as free-of-charge security classes, with a phased in requirement that all new servers connecting to the University network have identified a system-admin who has taken the University system-admin class. These ideas come from the programs that Randy Marchaney has told me they've imlemented at Virginia Tech.

Data Sources

Aside from data sources listed in the References section of Assignment 1, I also made heavy use of:

- Google for web searching - www.google.com
- Netcraft for determining OS types - www.netcraft.com
- The CVE site - cve.mitre.org/cve
- Incidents.org and Dshield.org for looking for attack correlations, port usage, etc. - www.incidents.org, www.dshield.org
- Inspiration was drawn from Chris Baker's excellent GCIA Practical - http://www.sans.org/y2k/practical/Chris_Baker_CGIA.zip

- Whois information came from ARIN and APNIC - www.arin.net/whois , www.apnic.net