Global Information Assurance Certification Paper

GIAC
Intrusion Detection Practical

By James G. McIntyre
(SD453294)
Capitol SANS

Contents:

Analysis format of detects:

- Source of trace
- Detect generated by
- Probability the source address was spoofed
- Description of attack
- Attack mechanism
- Correlations
- Evidence of active targeting
- Severity
- Defensive recommendation
- Multiple choice test question

Detect #1:

1) Source of trace:

GIAC - April 5, 2000 1230 - http://www.sans.org/y2k/040500-1230.htm

```
Apr 3 12:56:39 dns1 snort[4415]: IDS013 - RPC -
portmap-request-mountd: 216.160.38.58:761 -> a.b.c.34:111
--------
[**] IDS013 - RPC - portmap-request-mountd [**]
04/03-12:56:39.550530 216.160.38.58:761 -> a.b.c.34:111
UDP TTL:49 TOS:0x0 ID:47954
Len: 64
7A 62 57 13 00 00 00 00 00 00 00 02 00 01 86 A0 zbW.............
00 00 00 02 00 00 00 03 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 01 86 A5 00 00 00 01 ................
00 00 00 11 00 00 00 00 ........
--------
Apr 3 12:56:39 dns3 snort[9658]: IDS013 - RPC -
portmap-request-mountd: 216.160.38.58:750 -> a.b.c.98:111
--------
[**] IDS013 - RPC - portmap-request-mountd [**]
04/03-12:56:39.480862 216.160.38.58:750 -> a.b.c.98:111
UDP TTL:49 TOS:0x0 ID:47947
Len: 64
0B 3A 2F 6B 00 00 00 00 00 00 00 02 00 01 86 A0 .:/k............
00 00 00 02 00 00 00 03 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 01 86 A5 00 00 00 01 ................
00 00 00 11 00 00 00 00 ........
```

2) Detect generated by:

Snort intrusion detection software collected this trace. The version of snort or the ruleset version is not known. Snort rules are defined into 2 sections, the rule header and the rule options. For additional information please reference www.snort.org.

Snort rule that could generate this alert:

```
alert udp !$HOME_NET any -> $HOME_NET 111 (msg:"IDS13 - RPC - portmap-request-
mountd"; content:"|01 86 A5 00 00|";offset:40;depth:8;)
```

Snort message layout:

Rule Header: action, addresses, ports, direction (red high-light)
Rule options: detection modules to run and parameters (yellow background)

First trace record:

```
Apr 3 12:56:39 dns1 snort[4415]: IDS013 - RPC -
            portmap-request-mountd: 216.160.38.58:761 -> a.b.c.34:111
```

Description:

Date and time of trace :	Apr 3 12:56:39
Name of IDS machine:	dns1
IDS software and its PID:	snort[4415]
Snort detect message:	IDS013 - RPC - portmap-request-mountd
Source address & port:	216.160.38.58:761
Destination address & port:	a.b.c.34:111

Second trace record (header only):

```
[**] IDS013 - RPC - portmap-request-mountd [**]
04/03-12:56:39.550530 216.160.38.58:761 -> a.b.c.34:111
UDP TTL:49 TOS:0x0 ID:47954
Len: 64
```

Description:

```
Snort detect message:          [**] IDS013 - RPC - portmap-request-mountd [**]
Date and time of trace:        04/03-12:56:39.550530
Source address port:           216.160.38.58:761
Destination address & port:    a.b.c.34:111
Protocol used by this packet: UDP
Time to live for packet:       TTL:49
Type of Service:               TOS:0x0
Packet ID number:              ID:47954
Length of data payload:        Len: 64
actual data payload in packet:
7A 62 57 13 00 00 00 00 00 00 00 02 00 01 86 A0 zbW.............
00 00 00 02 00 00 00 03 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 01 86 A5 00 00 00 01 ................
00 00 00 11 00 00 00 00 ........
```

3) Probability the source address was spoofed:

The probability of the source address being spoofed is low. The attacker will need a response to ascertain success of the connection to portmapper and the query to the mountd service.

Information concerning source address, 216.160.38.58, was acquired via http://www.samspade.org/.

```
Registrant:
U S WEST Communication Services (USWEST2-DOM)
   600 Stinson Blvd.
   Minneapolis, MN 55413  US

   Domain Name: USWEST.NET

   Administrative Contact, Technical Contact:
      HOS48-ORGr  (HOS48-ORG)  dns-info@QWEST.NET
      Qwest Internet Solutions
      600 Stinson Blvd.
      Minneapolis, MN 55413  US
      800-672-8520 Fax- 123 123 1234
   Billing Contact:
      Lundgren, Paul  (PL84)  abuse@USWEST.NET
      U S WEST Interprise Networking
      600 Stinson Blvd
      Minneapolis, MN 55413
      (612) 664-3069 (FAX) (612) 664-4770

   Record last updated on 20-Nov-2000.
   Record expires on 22-Nov-2001.
   Record created on 21-Nov-1994.
   Database last updated on 29-Nov-2000 08:52:00 EST.

   Domain servers in listed order:

   NS1.USWEST.NET              204.147.80.5
```

NS2.DNVR.USWEST.NET	206.196.128.1
NS3.MN.USWEST.NET	204.147.80.1

4) Description of attack:

A query is sent to a specific host for port 111, portmapper, requesting information concerning an active rpc mountd service. Using this service, an attacker can acquire information concerning file systems. The mountd service is used for mounting NFS volumes.

CVE: CAN-1999-0632

Exploits for Rpc.mountd can be found at the following sites:
http://neworder.box.sk/search.php3?srch=mountd
ftp://ftp.pgci.ca/pub/pmap_tools/

5) Attack mechanism:

These transactions are benign from the aspect they will not negatively effect your network or a system. But it functions as "recon" for future attacks. The information gathering process using mountd involves:

- user issues a mount command for possible files on a remote system.
- mountd will return a "Permission denied error" if it can't access the specified file.
- mountd will return "No such files or directory" if the file does not exist.

Given the above capability, a user could map what files exist or which packages are installed on the remote system.

Mountd specific exploits and references:
http://xfortemce.iss.net/static/347.php
http://xforce.iss.net/static/967.php
http://ciac.llnl.gov/ciac/bulletins/i-048.shtml
http://xforce.iss.net/alerts/vol-2_num-6.php
http://xforce.iss.net/static/80.php

6) Correlations:

The following trace from Feb. 22nd is located at: http://www.sans.org/y2k/022300-2300.htm

```
[**] RPC - portmap-request-mountd [**]
02/22-15:02:53.171471 212.25.118.45:633 -> x.x.x.x:111
UDP TTL:49 TOS:0x0 ID:5046
Len: 64
39 BE 43 FB 00 00 00 00 00 00 00 02 00 01 86 A0 9.C.............
00 00 00 02 00 00 00 03 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 01 86 A5 00 00 00 01 ................
00 00 00 11 00 00 00 00 ........

[**] RPC - portmap-request-mountd [**]
02/22-15:02:58.162694 212.25.118.45:633 -> x.x.x.x:111
UDP TTL:49 TOS:0x0 ID:5088
Len: 64
39 BE 43 FB 00 00 00 00 00 00 00 02 00 01 86 A0 9.C.............
00 00 00 02 00 00 00 03 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 01 86 A5 00 00 00 01 ................
00 00 00 11 00 00 00 00 ........
```

Additional mountd attacks include:
http://www.sans.org/y2k/022100-1130.htm
http://lists.insecure.org/incidents/2000/Aug/0147.html

7) Evidence of active targeting:

The provided trace only contains 2 records. I will then assume that only these 2 systems have been addressed, i.e. targeted. If this is true, then attacker may know specific information as to the layout of the internal network and/or specific information as to interesting processes on these machines.

8) Severity:

It is not known what processes are running on these machines. Also, it is not known if a firewall exists or how it may be configured. Given these factors, correctly accessing the true severity will be impossible. Therefore, I will make up my own environment parameters.

```
Severity = (System criticality + Attack lethality) – (System countermeasures
           + Network Countermeasures)

System criticality:     5 – DNS Server
Attack lethality:       0 – recon
System countermeasures: 1 – rpc service is running, but not currently patched
Network countermeasures:2 – packets detected by IDS, but firewall not configured
                            for restricting access to port 111
(5 + 0 ) – ( 1 + 2 ) = 2
```

9) Defensive recommendation:

- If possible the rpc service on these machines should be deactivated.
- If a firewall exists, it could be configured to stop all traffic to port 111.
- The edge gateway should be configured to stop all traffic to port 111. *1
- If rpc is necessary, upgrade the system to a current version and apply any outstanding patches for rpc.

*1 Possible router acl:

```
access-list 110 deny udp any a.b.c.255 0.0.0.255 eq 111
```

10) Multiple choice test question:

Which port is utilized by portmapper?

- 111
- 761
- 479
- 110

Answer: 1

Detect #2:

1) Source of trace:

GIAC - April 5, 2000 1230 - http://www.sans.org/y2k/040500-1230.htm

```
Apr 3 14:33:46 dns1 snort[4415]: spp_portscan:
PORTSCAN DETECTED from 194.27.40.19
Apr 3 14:33:46 dns1 snort[4415]: IDS027 - SCAN-FIN:
194.27.40.19:47850 -> a.b.c.34:23
Apr 3 14:33:52 dns1 snort[4415]: spp_portscan: portscan status
from 194.27.40.19: 1 connections across 1 hosts: TCP(1), UDP(0) STEALTH
Apr 3 14:33:58 dns1 snort[4415]: spp_portscan: End of portscan
from 194.27.40.19
Apr 3 14:34:04 dns1 snort[4415]: spp_portscan: PORTSCAN DETECTED
from 194.27.40.19
Apr 3 14:34:04 dns1 snort[4415]: IDS027 - SCAN-FIN:
194.27.40.19:47850 -> a.b.c.34:23
Apr 3 14:34:10 dns1 snort[4415]: spp_portscan: portscan status
from 194.27.40.19: 1 connections across 1 hosts: TCP(1), UDP(0) STEALTH
Apr 3 14:34:16 dns1 snort[4415]: spp_portscan: End of portscan
from 194.27.40.19
--------
Apr 3 14:33:46 194.27.40.19:47850 -> a.b.c.34:23 FIN ***F****
Apr 3 14:34:04 194.27.40.19:47850 -> a.b.c.34:23 FIN ***F****
--------
[**] IDS027 - SCAN-FIN [**]
04/03-14:33:46.924487 194.27.40.19:47850 -> a.b.c.34:23
TCP TTL:229 TOS:0x0 ID:37380
***F**** Seq: 0x64780000 Ack: 0x0 Win: 0x200
00 00 00 00 00 00 ......
[**] IDS027 - SCAN-FIN [**]
04/03-14:34:04.467750 194.27.40.19:47850 -> a.b.c.34:23
TCP TTL:229 TOS:0x0 ID:37380
***F**** Seq: 0x96780000 Ack: 0x0 Win: 0x200
00 00 00 00 00 00 ......
```

2) Detect generated by:

Snort intrusion detection software. The version of snort or the ruleset version is not known. Two sections make up a snort rule, the rule header and the rule options. For additional information please reference www.snort.org.

Snort rule that could generate this alert:

```
Alert tcp !$HOME_NET any -> $HOME_NET any (flags: F; msg:"IDS027 SCAN-FIN";)
```

Rule Header: action, addresses, ports, direction (red high-light)
Rule options: detection modules to run and parameters (yellow high-light)

The following 3 records are all from the same detect each having a slightly different format.

Record as formatted by snort:

```
Apr 3 14:33:46 dns1 snort[4415]: IDS027 - SCAN-FIN:
194.27.40.19:47850 -> a.b.c.34:23
```

Description:

Date and time of trace: Apr 3 14:33:46

```
Name of IDS machine:          dns1
IDS software and its PID:     snort[4415]
Snort detect message:         IDS027 - SCAN-FIN:
Source address & port:        194.27.40.19:47850
Destination address & port:   a.b.c.34:23
```

Record in tcpdump format:

```
Apr 3 14:33:46 194.27.40.19:47850 -> a.b.c.34:23 FIN ***F****
```

Description:

```
Date and time of trace:       Apr 3 14:33:46
Source address & port:        194.27.40.19:47850
Destination address & port:   a.b.c.34:23
TCP protocol flags:           FIN ***F****
```

Record in snort format:

```
[**] IDS027 - SCAN-FIN [**]
04/03-14:33:46.924487 194.27.40.19:47850 -> a.b.c.34:23
TCP TTL:229 TOS:0x0 ID:37380
***F**** Seq: 0x64780000 Ack: 0x0 Win: 0x200
00 00 00 00 00 00 ......
```

Description:

```
Snort detect message:         [**] IDS027 - SCAN-FIN [**]
Date and time of trace:       04/03-14:33:46.924487
Source address & port:        194.27.40.19:47850
Destination address & port:   a.b.c.34:23
Protocol utilized:            TCP
Time to live for packet:      TTL:49
Type of Service:              TOS:0x0
Packet ID number:             ID:47954
TCP protocol flags:           FIN ***F****
Packet sequence number:       0x64780000
Acknowledgment number:        0x0
TCP Window size               0x200
The following is the actual data packet:
                        00 00 00 00 00 00 ......
```

3) Probability the source address was spoofed:

The probability of the source address being spoofed is low. The attacker will need a response to ascertain success of the connection to the telnet port, 23.

Information concerning source address 194.27.40.19 was acquired via http://www.samspade.org/.

```
inetnum:    194.27.40.0 - 194.27.40.255
netname:    ZKU-NET
descr:      Zonguldak Karaelmas Universitesi
country:    TR
admin-c:    OE75
tech-c:     OE75
status:     ASSIGNED PA
changed:    hostmaster@metu.edu.tr 19990303
source:     RIPE
```

4) Description of attack:

The attacker is trying to determine if the telnetd service is active on a specific machine. Since the packet is incorrectly built, it should elicit a particular response thereby indicating whether the service is active or not.

Exploit tool:

nmap - http://www.insecure.org/nmap/
http://saturnlink.com/articles/21700nmap.html

example of nmap command: nmap -sF -P0 -p1-100 193.189.XXX.YYY

5) Attack mechanism:

A record with only the FIN flag set is not a valid transaction. The primary purpose of the FIN is to close an open connection. A normal transaction would have the FIN-ACK flags set. The FIN-only record would have to be customized and it has only one use, recon. With a FIN-only flag set, the target machine should return a RST for closed ports and open ports should drop the packet, ie. send no response. Given this information, a scan would indicate whether the telnetd is active or not. This type of scan is only valid on a unix operating system, Microsoft decided not to follow the RFC, again.

This type of scan is used for recon only. In some cases this type of recon is not logged by a firewall. Please reference: http://lists.insecure.org/nmap-hackers/1999/Apr-Jun/0029.html

Exploits and references:

CVE-1999-0192
CVE-1999-0273
CAN-2000-0480

6) Correlations:

The following trace from Mar 24th is located at: http://www.sans.org/y2k/032400-2000.htm

Videon CableSystems Alberta Inc

```
Mar 24 12:08:25 dns1 snort[6970]: IDS027 -
SCAN-FIN: 24.108.45.77:47850 -> x.y.z.34:23
Mar 24 12:08:45 dns1 snort[6970]: IDS027 -
SCAN-FIN: 24.108.45.77:47850 -> x.y.z.34:23
--------
[**] IDS027 - SCAN-FIN [**]
03/24-12:08:25.749409 24.108.45.77:47850 -> x.y.z.34:23
TCP TTL:227 TOS:0x0 ID:37380
***F**** Seq: 0x5B770000 Ack: 0x0 Win: 0x200
00 00 00 00 00 00 ......

[**] IDS027 - SCAN-FIN [**]
03/24-12:08:45.625425 24.108.45.77:47850 -> x.y.z.34:23
TCP TTL:227 TOS:0x0 ID:37380
***F**** Seq: 0xCC770000 Ack: 0x0 Win: 0x200
00 00 00 00 00 00 ......
```

7) Evidence of active targeting:

The provided trace only contains 2 records. I will then assume that only this system was targeted. Assuming this is correct, the attacker may know specific information as to the layout of the internal network and/or specific information as to interesting processes on this machine.

8) Severity:

It is not known what processes are running on these machines. Also, it is not known if a firewall exists or how it may be configured. Given these factors, accessing the true severity will be impossible. Therefore, I will make up my own environment parameters.

```
Severity = (System criticality + Attack lethality) – (System countermeasures
           + Network Countermeasures)

System criticality:     5 – DNS Server
Attack lethality:       1 – recon
System countermeasures: 1 – telent service is running, but not currently patched
Network countermeasures:1 – records detected by IDS, but firewall not configured
                            to restrict port 23 access
( 5 + 1 ) – ( 1 + 1 ) = 3
```

9) Defensive recommendation:

- If possible the telnet service should be deactivated. If this is not possible, verify all outstanding patches have been applied. Also, consider using secure shell for all telnet traffic. *1
- If a firewall exists, it should be configured to stop port 23 traffic.
- The edge gateway router, could be configured to stop traffic to port 23. *2
- Install tcp wrappers on all unix systems. *3

*1 Secure shell reference:
ftp://coast.cs.purdue.edu/pub/tools/unix/netutils/openssh/
http://www.openssh.com/

*2 Cisco router acl:
access-list 110 deny tcp any 192.168.0.0 0.0.255.255 eq 23.

*3 Tcpwrapper reference:
ftp://coast.cs.purdue.edu/pub/tools/unix/netutils/tcp_wrappers/

10) Multiple choice test question:

Given the following trace, what is the proper response from the destination host with port 23 closed ?

```
Apr 3 14:33:46 194.27.40.19:47850 -> a.b.c.34:23 FIN ***F****
```

- nothing
- RST-ACK
- PUSH-ACK
- FIN-ACK

```
Correct answer: 2
```

Detect #3:

1) Source of trace:

GIAC - April 11, 2000 - http://www.sans.org/y2k/041100.htm

```
Apr 8 18:07:53 dns1 named[11759]:
       unapproved update from [208.243.251.37].65532 for .edu
Apr 8 18:07:54 dns1 named[11759]:
       unapproved update from [208.243.251.37].65531 for .edu
Apr 9 00:54:48 dns1 named[11759]:
       unapproved query from [130.39.190.28].3064 for "version.bind"
Apr 9 00:54:48 dns2 named[157]:
       unapproved query from [130.39.190.28].3062 for "version.bind"
```

2) Detect generated by:

It was not specified in the trace document, but the format of the messages would indicate they were generated by the named service. Depending on the system, these messages could be found in /var/log/messages or perhaps in syslog.

First trace record:

Apr 8 18:07:54 dns1 named[11759]: unapproved update from 208.243.251.37].65531 for .edu

Description:

Date and time of trace:	Apr 8 18:07:54
Destination machine:	dns1
Name of service:	named[11759]
Error message:	unapproved update
Source ip address & port:	[208.243.251.37].65531
Destination domain name:	.edu

Second trace record:

Apr 9 00:54:48 dns1 named[11759]: unapproved query from [130.39.190.28].3064 for "version.bind"

Description:

Date and time of trace:	Apr 9 00:54:48
Name of destination machine:	dns1
Name of service:	named[11759]:
Error message:	unapproved query for "version.bind"
Source ip address & port:	from [130.39.190.28].3064

3) Probability the source address was spoofed:

The probability of the source address being spoofed is low. The attacker will need a response to ascertain success of the connection for the DNS query. Per the DNS "unapproved update", the attacker would not require a response to determine whether the update worked or not.

Information concerning source address 208.243.251.37 was gathered from http://www.samspade.org/

Registrant:
HES Enterprises (TXCONNECT-DOM)

```
          7716 Rainfall Ridge
          San Antonio, TX 78239

          Domain Name: TXCONNECT.COM

          Administrative Contact, Technical Contact, Billing Contact:
             Hibdon, Steve  (SH1612)  admin@HES.NET
             Hot-Stuff
             7716 Rainfall Ridge
             San Antonio, TX 78239
             210-657-6590 (FAX) 210-654-3410

          Record last updated on 16-Sep-1998.
          Record expires on 13-Apr-2001.
          Record created on 13-Apr-1998.
          Database last updated on 21-Dec-2000 04:35:05 EST.

          Domain servers in listed order:

          NS1.TXCONNECT.COM           208.243.251.101
          NS2.TXCONNECT.COM           208.243.251.102

Information concerning source address 130.39.190.28 was provided by
http://whois.geektools.com/cgi-bin/proxy.cgi :

Query:     130.39.190.28
Registry:  whois.arin.net
Results:
Louisiana State University (NET-TIGERLAN)
   200 Computing Services Center
   Baton Rouge, LA  70803

   Netname: TIGERLAN
   Netnumber: 130.39.0.0

   Coordinator:
      Robbins, Sean  (SR935-ARIN)  sean@LSU.EDU
      (504) 388-5204 (FAX) (504) 388-6400

   Domain System inverse mapping provided by:

   TE6000.OTC.LSU.EDU          130.39.128.71
   TENET.OTC.LSU.EDU           130.39.130.175

   Record last updated on 15-May-1996.
   Database last updated on 21-Dec-2000 07:33:22 EDT.
```

4) Description of attack:

There are 2 different types of attacks occurring in this trace. The first is an attempt to modify the domain name tables. The modification could be the addition of a machine name/ip address, the removal of a name, or the changing of an ip address. The second trace is a query to the domain name server as to its current version. Given this information, an attacker can utilize known exploits. It is not known whether these DNS servers were located on internal or external lan.

The following are a number of known bind attacks:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0184
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0024

CERT® Advisories:

CA-98.05 – 3 vulnerabilities in bind
CA-99-14 Multiple Vulnerabilities in BIND

Netbus scanner:

http://neworder.box.sk/box.php3?gfx=neworder&prj=neworder&key=portsc&txt=Scanners

Exploit tool:
http://www.hack.co.za/daem0n/named/t666.c

Dig command inquiring for version information:
dig @domain.name version.bind txt chaos

5) Attack mechanism:

The first attack, updating DNS, utilizes standard DNS commands to dynamically update the DNS tables. This dynamic update process is integral to the DNS functionality. Tools that can be utilized to update the DNS are noted below. This is not a benign attack.

The second attack, is again using standard DNS commands, in this case "version.bind". This is an attempt to identify the version of bind and any other information concerning the domain name server. It is part of a recon process in an attempt to map out the local network. An overview of DNS and possible commands can be found at: http://www.isc.org/products/BIND/

Both traces noted have time stamps that are very close to each other. Also, they are utilizing the higher numbered ports, indicating they are probably running as a client process. Lastly, the port numbers are sequential. Given these characteristics, it would indicate a script of some type is being utilized.

Additional DNS attack techniques:
http://xforce.iss.net/static/1226.php
http://xforce.iss.net/static/206.php

6) Correlations:

http://www.sans.org/y2k/041100.htm

```
Apr 9 09:08:09 darkstar portsentry[69]: attackalert: SYN/Normal scan from
       host: cr520663-a.yec1.on.wave.home.com/24.114.44.85
       to TCP port: 53
Apr 9 09:08:09 darkstar portsentry[69]: attackalert: Host 24.114.44.85 has
       been blocked via dropped route using command: "/sbi
       n/ipchains -I input -s 24.114.44.85 -j DENY -l"
```

http://madhaus.utcs.utoronto.ca/bind8/bind-users-archive/1998/11/msg00151.html

```
12-Nov-1998 19:46:24.719 security: notice: unapproved query from
       [206.86.8.21].53 for "xx.xx.xx.xx.in-addr.arpa"
13-Nov-1998 02:32:42.868 security: notice: unapproved query from
       [204.152.166.73].3437 for "XX.com"
13-Nov-1998 09:43:13.763 security: notice: unapproved query from
       [206.184.139.147].4064 for "xx.xx.xx.xx.in-addr.arpa"
```

http://www.sans.org/y2k/041800.htm

```
Apr 16 10:34:52 morannon named[415]: unapproved query
       from [216.61.140.211].4716 for "aborasurfing.net"
Apr 16 10:55:02 morannon named[415]: unapproved query
       from [212.242.18.132].1033 for "21.240.21.208.in-addr.arpa"
Apr 16 12:31:58 morannon named[415]: unapproved query
       from [216.61.140.211].4765 for "acmefund.com"
```

Exploits and references:

CERT® Advisory: CA-98.05 – 3 vulnerabilities in bind
CERT® Advisory CA-99-14 Multiple Vulnerabilities in BIND -
http://www.sans.org/infosecFAQ/DNS_exploit.htm
http://archives.neohapsis.com/archives/freebsd/2000-08/0258.html

7) Evidence of active targeting:

The trace we have is very short. I will assume that these were the only 2 trace records encountered. Given this, these 2 machines were targeted from 2 different sources. The attackers new the ip addresses and probably new that each were DNS servers. Other interesting aspects of these packets include: the source port numbers are the same and the packet sequence numbers are the same. Both of these characteristics indicate a crafted packet.

8) Severity:

It is not known that only DNS is running on these machines. It is not known if a firewall exists or how it may be configured. Given these factors, accessing the true severity will be limiting. Therefore, I will make up my own environment parameters.

```
Severity = (System criticality + Attack lethality) – (System countermeasures
            + Network Countermeasures)

System criticality:     5 – DNS Server
Attack lethality:       5 – attack
System countermeasures: 1 – bind not currently patched or at a current version
Network countermeasures:1 – no IDS is in place, the firewall is not patched at a
                            current level.
(5 + 5 ) – ( 1 + 1 ) = 8
```

The sys-admin should consider updating their resume.

9) Defensive recommendation:

I will assume the 2 DNS servers in question are on the internal network. Given this premise, the following steps can be performed.

1) If a firewall is in place, upgrade it to the current release and any outstanding patches. Also, configure it so external DNS access is not permitted.
   - The external router can include an acl to restrict this access. *1
   - Upgrade to a more current version of bind on these 2 machines.
   - Utilize ip masquerading internally so the machines are not directly addressable for the outside. *2
   - Implement an external DNS server that contains only necessary internal machine addresses.
   - Implement an IDS to identify these transactions in real-time. It can also be setup to kill the packet. Utilize tools that can identify these packets, ie. anti-sniff for DNS queries. *3
7) Configure the DNS servers to not interact with external requests not utilizing restricted ports ie. <1024.

*1 Cisco router acl:

```
access-list 110 deny tcp any 192.168.0.0 0.0.255.255 eq 53.
```

*2 Linux ipchains command:

```
ipchains –A forward –j MASQ –s 192.168.0.0/24 –d 0.0.0.0/0
```

*3 Anti-sniff tools and IDS's:

http://www.l0pht.com/advisories/asniff_advisory.txt
http://www.nswc.navy.mil/ISSEC/CID/

http://www.snort.org/

10) Multiple choice test question:

```
Apr 9 00:54:48 dns2 named[157]:
unapproved query from [130.39.190.28].3062 for "version.bind"
```

Given the above trace, the query is unapproved due to the source port not being 53 ?

- T
- F

Answer: 2

Detect #4:

1) Source of trace:

GIAC – March 29, 2000 1200 - http://www.sans.org/y2k/032900.htm

```
02:26:31.574847 209.216.2.200 > morannon.kdi.com: (frag 30041:48@2960)
02:26:31.583572 209.216.2.200 > morannon.kdi.com: icmp: echo request (frag
                                                        30041:1480@0+)
02:26:31.583582 209.216.2.200 > morannon.kdi.com: (frag 30044:48@2960)
02:26:31.591760 209.216.2.200 > morannon.kdi.com: icmp: echo request (frag
                                                        30044:1480@0+)
02:26:31.591768 209.216.2.200 > morannon.kdi.com: (frag 30046:48@2960)
02:26:31.600166 209.216.2.200 > morannon.kdi.com: icmp: echo request (frag
                                                        30046:1480@0+)
```

2) Detect generated by:

The trace was collected by the tcpdump utility (*1). The trace consists of a large ping packet that has been fragmented into the standard ip packet size of 1480 bytes.

First trace record:

```
      02:26:31.5748:47 209.216.2.200 > morannon.kdi.com: (frag 30041:48@2960)
```

Description:

```
      Time stamp of trace:              02:26:31.574847
      Source ip address & port:         209.216.2.200
      Destination ip address:           morannon.kdi.com:
      Fragmentation description:        frag 30041:48@2960)
            Fragment ID:                      30041
            Length of data:                   48
            Data offset into packet:          2960
            Additional data indicator:        not set
```

Second trace record:

```
      02:26:31.583572 209.216.2.200 >
                  morannon.kdi.com: icmp: echo request (frag 30041:1480@0+)
```

Description:

```
      Time stamp of trace:              02:26:31.583572
      Source ip address & port:         209.216.2.200
      Destination ip address:           morannon.kdi.com:
      Protocol:                         icmp
      Command issued:                   echo request
      Fragmentation description:        (frag 30041:1480@0+)
            Fragment ID:                      30041
            Length of data:                   1480 (should be a multiple of 8 except last
                                              packet)
            Data offset into packet:          0
            Additional data indicator:        +
```

*1 Tcpdump utility:

ftp://coast.cs.purdue.edu/pub/tools/unix/netutils/tcpdump/
http://www.tcpdump.org/

3) Probability the source address was spoofed:

The probability the source address was spoofed is high. The purpose of these packets is to initiate a denial of service attack. Therefore, no useful response is expected from the destination machine.

Information concerning source address 209.216.2.200 was gathered from:
http://whois.geektools.com/cgi-bin/proxy.cgi

```
Query:     209.216.2.200
Registry:  whois.arin.net
Results:
AnaServe, Inc. (NETBLK-ANASERVE-BLK-1)
   1300 Bristol Street North, Suite 220
   Newport Beach, CA 92660
   US

   Netname: ANASERVE-BLK-1
   Netblock: 209.216.0.0 - 209.216.63.255

   Coordinator:
      Smith, Steve  (SS2039-ARIN)  ssmith@ANASERVE.COM
      949-250-7262

   Domain System inverse mapping provided by:

   NS1.NAMED.NET          208.197.88.4
   NS2.NAMED.NET          208.206.63.4

   Record last updated on 29-Apr-1998.
   Database last updated on 21-Dec-2000 18:38:05 EDT.
```

4) Description of attack:

The attacker is sending crafted packets that will create a denial of service situation. The transmitted packets have been marked as being fragmented thereby requiring the host to reassemble the whole packet. But a portion of the packet is never transmitted to the destination host.

Exploit tools:

hping2 - http://sourceforge.net/projects/hping2
Icmpenum v1.1.1, written by Simple Nomad (http://razor.bindview.com).
SING v1.0, written by Alfredo Andres Omella
http://www.sourceforge.net/projects/sing

CVE:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0305

5) Attack mechanism:

This particular attack utilizes a standard function of the TCP/IP stack, the reassembly of packets. If a packet is too large for the IP datagram, 1480, the packet is divided into multiple packets. Each packet contains a portion of the complete payload data, the offset of this piece into the whole payload, the length of the current data payload (which is in increments of 8 bytes, except for the last packet), and the packet ID number. Each of the divided packets have the same packet ID number. The destination machine will collect all of the packets until the complete data payload has been received. It will wait up till 90 seconds before

freeing the packets resources and then responding with a message of "ip reassembly time exceeded". If enough partial packets are sent, the destination machine will run out of resources and no longer receive any further packets, ie. DOS. There is also the possibility the machine will panic and die a horrible death.

In this particular trace, the first packet (1480 bytes) and the 3rd packet (48 bytes) are received. The 2nd packet is never received because it is never sent. The packets in this trace were coming quite quickly and in mass, over 24,000 packets were received. This definitely was a denial of service attack.

6) Correlations:

http://archives.neohapsis.com/archives/incidents/2000-06/0159.html

```
16:18:27.496564 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro: icmp: echo request
                                                        (frag 63124:1480@0+)
16:18:27.696848 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro: (frag
3124:1480@2960+)
16:18:27.761573 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro:(frag
63124:1480@1480+)
16:18:27.816649 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro:(frag

                                                  63124:1480@14800+)
16:18:27.876395 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro: (frag

                                                    63124:1480@31080+)
16:18:27.947211 agschool.FVSC.PeachNet.EDU > ns.geniusnet.ro: (frag
                                                         63124:1480@50320+
                                                    )
16:18:27.973133 ns.geniusnet.ro > agschool.FVSC.PeachNet.EDU: icmp: ip reassembly
                                                    time exceeded [tos
                                                  0xc0]
```

http://www.uwsg.iu.edu/hypermail/linux/net/9908.2/0039.html

```
01:55:56.414913 hydro.innsmouth > molybdenum.innsmouth: (frag 48163:1244@2960)
01:55:56.414913 hydro.innsmouth.nfs > molybdenum.innsmouth.1166040940: reply ok
1472
                                                  read (frag 48163:1480@0+)
01:56:02.414913 molybdenum.innsmouth > hydro.innsmouth: icmp: ip reassembly time
                                              exceeded [tos 0xc0]01:56:02.414913
```

7) Evidence of active targeting:

This specific machine was targeted. It was the only ip address in the trace and there were over 24,000 packets.

8) Severity:

It is not known what services are running on this machine. It is not known if a firewall exists or how it may be configured. Given these factors, accessing the true severity will be limiting. Therefore, I will make up my own environment parameters.

```
Severity = ( System criticality + Attack lethality ) – ( System countermeasures
           + Network Countermeasures )
```

```
System criticality:     3 - Office secretary's machine
Attack lethality:       5 - attack
System countermeasures: 2 - running current version of the o/s and patched
Network countermeasures:3 - IDS is in place, the firewall is patched at a
                            current level.
(3 + 5 ) - ( 2 + 3 ) = 3
```

9) Defensive recommendation:

I will assume the 2 DNS servers in question are on the internal lan. Given this premise, the following steps can be performed.

1) If a firewall is in place, upgrade it to the current release and any outstanding patches. Also, configure it so external echo requests are not passed through. If a firewall is not used, consider one.
2) The edge router can include additional restrictions on incoming and outgoing icmp commands. *1
3) Utilize ip masquerading internally so the machines are not directly addressable for the outside. *2
4) Implement an IDS to identify malformed icmp packets and possibly not allow them to reach the network. *3

*1 Block icmp echo attacks at the edge router for internal network 172.16.0.0:

```
block inbound echo requests and outbound echo replies
access-list 110 deny  icmp any 172.16.0.0 0.0.255.255 echo
access-list 111 deny  icmp 172.16.0.0 0.0.255.255 any echo-reply

allow outbound echo requests and inbound echo replies
access-list 111 permit icmp 172.16.0.0 0.0.255.255 any echo
access-list 110 permit icmp any 172.16.0.0 0.0.255.255 echo-reply
```

*2 Linux command to perform masquerading.

```
ipchains -A forward -j MASQ -s 192.168.0.0/24 -d 0.0.0.0/0
```

*3 Example filter for icmp packets that have data lengths not a multiple of 8 and have the MF, more fragments, bit set.

```
(ip[6:1]&0x20 != 0) and
( (ip[2:2] - ((ip[0:1]&0x0f)*4)) & 0x7 != 0)
```

10) Multiple choice test question:

Packets with the MF bit set can have any length payload between 0 and 1480 ?

- T
- F

Answer: F - Length of data payload must be a multiple of 8.

Assignment #2 Analyze This !

Data Overview:

The one months data provided spans the time frame of 9/26 to 11/23. Not all days are represented in the provided data logs. There are 3 types of logs, S-logs, A-logs, and OOS logs. File names for the logs appear to have no apparent relationship to the actual date of log creation. Each log represents 1 day. The logs contain snort scan messages (S-logs), snort alert messages (A-logs), and ascii trace data (OOS-logs).

All logs specify the home network of "MY.NET", a class B network. To assist the snortsnarf utility and myself, all the records were changed to a home network of "192.66". Each A & S log also contained 16 lines of header information that were removed. The script to modify these files is located in Appendix A.

A – log analysis:

110534 alerts found among the files:
21 identified signatures
Earliest alert at **00:00:52**.873106 *on 09/26*
Latest alert at **23:32:20**.988483 *on 11/22*

SnortA10.txt.m	SnortA23.txt.m	SnortA33.txt.m	SnortA43.txt.m	SnortA53.txt.m
SnortA11.txt.m	SnortA24.txt.m	SnortA34.txt.m	SnortA44.txt.m	SnortA54.txt.m
SnortA12.txt.m	SnortA25.txt.m	SnortA35.txt.m	SnortA45.txt.m	SnortA55.txt.m
SnortA13.txt.m	SnortA26.txt.m	SnortA36.txt.m	SnortA46.txt.m	SnortA57.txt.m
SnortA14.txt.m	SnortA27.txt.m	SnortA37.txt.m	SnortA47.txt.m	SnortA59.txt.m
SnortA15.txt.m	SnortA28.txt.m	SnortA38.txt.m	SnortA48.txt.m	SnortA6.txt.m
SnortA19.txt.m	SnortA29.txt.m	SnortA39.txt.m	SnortA49.txt.m	SnortA7.txt.m
SnortA2.txt.m	SnortA3.txt.m	SnortA4.txt.m	SnortA5.txt.m	SnortA8.txt.m
SnortA20.txt.m	SnortA30.txt.m	SnortA40.txt.m	SnortA50.txt.m	SnortA9.txt.m
SnortA21.txt.m	SnortA31.txt.m	SnortA41.txt.m	SnortA51.txt.m	SnortAle.txt.m
SnortA22.txt.m	SnortA32.txt.m	SnortA42.txt.m	SnortA52.txt.m	

Top 6 alert signatures of 21 identified signatures:

Signature (click for definition)	# Alerts	# Sources	# Destinations
Attempted Sun RPC high port access	2542	20	33
TCP SMTP Source Port traffic	2893	4	2836
WinGate 1080 Attempt	4802	570	2655
Watchlist 000222 NET-NCFC	8166	45	26
Watchlist 000220 IL-ISDNNET-990517	30998	61	108
SYN-FIN scan!	56250	30	25751

WinGate 1080 Attempt:

Signature (click for definition)	# Alerts	# Sources	# Destinations
WinGate 1080 Attempt	4802	570	2655

Earliest such alert at **00:00:52**.873106 *on 09/26*
Latest such alert at **23:32:20.**988483 *on 11/22*

WinGate functions as a windows proxy server for home networks. It can service connections to port 1080 that services socks requests. If improperly configured, WinGate server will function as a proxy server for anyone. A good description of the problem and configuration suggestions can be found at http://www.oz.org/help/wingate.html.

Correlations:
http://www.sans.org/y2k/062100-1030.htm
http://www.sans.org/y2k/analysts.htm - Bill Royds #0247

Example of SnortS record:
10/10-00:19:56.411913 [**] WinGate 1080 Attempt [**] 208.194.161.155:2258 -> 192.66.98.205:1080

Example of Snort alert command:
alert tcp $EXTERNAL_NET !53 -> $HOME_NET 1080 (msg:"MISC-WinGate-1080-Attempt";flags:S;)

WinGate 1080 – Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
63.193.210.208	1883	1883	1837	1837
208.194.161.155	222	222	104	104

Server used for this query: [whois.arin.net] - 63.193.210.208

```
Pacific Bell Internet Services,Inc. (NETBLK-PBI-NET-7)
   Marathon Plaza, North Tower
   303 Second St, Suite 830
   San Francisco, CA 94107

   Netname: PBI-NET-7
   Netblock: 63.192.0.0 - 63.207.255.255
   Maintainer: PACB
```

Server used for this query: [whois.arin.net] - 208.194.161.155

```
UUNET Technologies, Inc. (NETBLK-UUNET1996B)
   3060 Williams Drive, Suite 601
   Fairfax, VA 22031
   US

   Netname: UUNET1996B
   Netblock: 208.192.0.0 - 208.249.255.255
   Maintainer: UU
```

The scan from 63.193.210.208 lasted only 5 minutes utilizing a varying number of source ports. I did not notice any duplication in destination addresses which changed in an increasingly and in random increments.

10/05-18:58:22.389439 [**] WinGate 1080 Attempt [**] 63.193.210.208:1605->
192.66.1.10:1080
10/05-19:03:05.389213 [**] WinGate 1080 Attempt [**] 63.193.210.208:3591->
192.66.226.253:1080

WinGate 1080: Top 3 destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.206.118	372	374	7	9
192.66.225.154	126	127	6	7
192.66.60.11	76	79	44	47

Watchlist 000222 NET-NCFC – Alert:

Signature (click for definition)	# Alerts	# Sources	# Destinations
Watchlist 000222 NET-NCFC	8166	45	26

Earliest such alert at **01:43:43**.866602 *on 09/26*
Latest such alert at **21:27:46**.757337 *on 11/22*

The addresses for this network, The Computer Network Center Chinese Academy of Sciences, have been placed on the Watchlist. The destination port was primarily port 25, smtp, and to a lesser extent port 113, ident. Given the number of trace records at certain times, it could be a DOS attack. Other times they are scanning slowly. Reviewing the highest hit systems would be advisable.

Correlations:
http://www.sans.org/y2k/032600-2000.htm
http://www.zeltser.com/sans/practical/

Example SnortS record:
10/10-22:41:10.469671 [**] Watchlist 000222 NET-NCFC [**] 159.226.39.1:1729 ->
192.66.100.230:25

Watchlist 000222 - Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
159.226.45.3	6297	6297	8	8
159.226.91.20	1212	1212	4	4
159.226.41.166	123	123	2	2
159.226.5.77	96	96	1	1

Server used for this query: [whois.arin.net] – all of the above

```
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

   Netname: NCFC
```

Netblock: 159.226.0.0 - 159.226.255.255

The highest number of scans occurred during the range noted below. Most of the scans were to port 25 with a less number to port 113, ident. The highest hit address, 192.66.6.7, had 5665 hits during this timeframe.

10/04-02:09:13.681958 [**] Watchlist 000222 NET-NCFC [**] 159.226.45.3:3858-> 192.66.6.7:25
10/04-10:50:35.881311 [**] Watchlist 000222 NET-NCFC [**] 159.226.45.3:4124-> 192.66.253.43:25

Watchlist 000222 – Top destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.6.7	5801	5808	8	14
192.66.100.230	1299	1302	7	9
192.66.253.43	461	589	17	21
192.66.253.41	186	331	16	21
192.66.253.42	155	171	12	20

Watchlist 000222 IL-ISDNNET – Alert:

Signature (click for definition)	# Alerts	# Sources	# Destinations
Watchlist 000220 IL-ISDNNET-990517	30998	61	108

Earliest such alert at **01:14:52**.325234 *on 09/26*
Latest such alert at **14:58:55**.189582 *on 11/22*

The addresses for this network, a site in Israel, have been placed on the Watchlist. The destination port varied greatly, as 90 different ports were used. So it would be hard to identify just one security hole they may be searching for. It appears most of the scans were slow in nature. Reviewing the highest hit systems would be advisable.

Correlations:
http://www.sans.org/y2k/032500-2200.htm
http://www.sans.org/y2k/051900.htm
http://www.sans.org/y2k/analysts.htm - Matteo Nava #

Example SnortS record:
10/10-13:14:55.361220 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.1 91:3551 -> 192.66.207.158:6700

Watchlist 000222 IL-ISDNNET – Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
212.179.95.5	6117	6117	9	9
212.179.27.6	4011	4011	15	15
212.179.79.2	3950	3950	14	14
212.179.44.115	3938	3938	1	1

Server used for this query: [whois.ripe.net] – all of above

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

inetnum:      212.179.95.0 - 212.179.99.255
netname:      CABLE-XPRMNT
descr:        Cable-Modem-Experiment
country:      IL
admin-c:      NP469-RIPE
tech-c:       NP469-RIPE
status:       ASSIGNED PA
notify:       hostmaster@isdn.net.il
changed:      hostmaster@isdn.net.il 20000103
source:       RIPE

route:        212.179.0.0/17
descr:        ISDN Net Ltd.
origin:       AS8551
notify:       hostmaster@isdn.net.il
mnt-by:       AS8551-MNT
changed:      hostmaster@isdn.net.il 19990610
source:       RIPE

person:       Nati Pinko
address:      Bezeq International
address:      40 Hashacham St.
address:      Petach Tikvah  Israel
```

All of the alerts for the top destination address, 192.66.211.146, occurred between the times noted below. Also, only 1 destination port, 4922, was targeted.

```
11/05-04:47:27.303528 [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.95.5:1263-> 192.66.211.146:4922
11/05-07:05:43.012614 [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.95.5:1574-> 192.66.211.146:4922
```

Correlations:
http://www.sans.org/y2k/practical/Dale_Ross_GCIA.htm

Watchlist 000222 IL-ISDNNET – Top destinations for this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.211.146	4810	4814	1	3
192.66.223.98	3938	3940	1	3
192.66.206.90	3914	3918	2	6
192.66.203.142	1638	1640	1	3
192.66.218.142	1459	1463	1	5
192.66.214.170	1353	1371	1	8

SYN-FIN Scan Alert:

Signature (click for definition)	# Alerts	# Sources	# Destinations

Earliest such alert at **13:10:30**.153412 *on 09/30*
Latest such alert at **09:33:33**.732424 *on 11/22*

This type of scan is pretty typical. This is not a normal setting for TCP flags. Its purpose is one of intrusion. It could also be utilized to fingerprint a machine. The top source for this scan was port 53 and primary destination port of 53. It would appear they were looking for a DNS server. A utility to perform this type of scan is hping.

Correlations:
http://www.sans.org/y2k/analysts.htm - Bill Royds #0247
http://www.sans.org/y2k/032200-1700.htm

Example SnortS record:
10/10-14:23:20.357735 [**] SYN-FIN scan! [**] 212.0.107.107:53 -> 192.66.254.243:53

Example of Snort alert command:
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"IDS198 - SCAN-SYN FIN";flags:SF;)

SYN-FIN Scan – Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
160.78.49.191	7199	7199	7199	7199
208.61.4.207	6635	6635	6635	6635
209.92.40.32	4967	4967	4967	4967
63.195.56.20	3897	3897	3897	3897

Server used for this query: [whois.arin.net] - 160.78.49.191

```
Centro di Calcolo di Ateneo (NET-PARMANET1)
   Centro di Calcolo di Ateneo
   Universita` di Parma
   Viale Delle Scienze
   43100 PARMA - ITALIA

   Netname: PARMANET
   Netblock: 160.78.0.0 - 160.78.255.255
```

Server used for this query: [whois.arin.net] - 208.61.4.207

```
BellSouth.net Inc. (NETBLK-BELLSNET-BLK7)
   301 Perimeter Center North,  Suite 400
   Atlanta, GA 30346
   US

   Netname: BELLSNET-BLK7
   Netblock: 208.60.0.0 - 208.63.255.255
   Maintainer: BELL
```

All of the scans from the top source address, 160.78.49.191:53, occurred during the range of time specified below. The scan was continuous and scanned the range of subnets. The same source and destination port, 53 – DNS services, was utilized for all traffic.

```
09/30-13:10:30.153412 [**] SYN-FIN scan! [**] 160.78.49.191:53-> 192.66.1.9:53
09/30-13:32:06.932517 [**] SYN-FIN scan! [**] 160.78.49.191:53-> 192.66.254.253:53
```

SYN-FIN Scan – Top destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.223.251	10	12	10	11
192.66.253.82	8	8	8	8
192.66.104.90	8	9	8	9
192.66.201.126	8	12	3	7

S – log analysis:

314003 alert records
245 identified signatures
Earliest alert at **01:39:55** *on 9/27*
Latest alert at **21:15:34** *on 11/23*

SnortS10.txt.m	SnortS2.txt.m	SnortS31.txt.m	SnortS4.txt.m	SnortS58.txt.m
SnortS11.txt.m	SnortS20.txt.m	SnortS32.txt.m	SnortS41.txt.m	SnortS6.txt.m
SnortS12.txt.m	SnortS21.txt.m	SnortS33.txt.m	SnortS42.txt.m	SnortS7.txt.m
SnortS13.txt.m	SnortS22.txt.m	SnortS34.txt.m	SnortS45.txt.m	SnortS8.txt.m
SnortS14.txt.m	SnortS23.txt.m	SnortS35.txt.m	SnortS47.txt.m	SnortS9.txt.m
SnortS15.txt.m	SnortS24.txt.m	SnortS36.txt.m	SnortS48.txt.m	SnortSca.txt.m
SnortS16.txt.m	SnortS27.txt.m	SnortS37.txt.m	SnortS49.txt.m	
SnortS17.txt.m	SnortS3.txt.m	SnortS38.txt.m	SnortS5.txt.m	
SnortS18.txt.m	SnortS30.txt.m	SnortS39.txt.m	SnortS56.txt.m	

Top 4 scan signatures:

Signature (click for definition)	# Alerts	# Sources	# Destinations
TCP ***F**** scan	454	28	369
UDP scan	23954	84	1420
TCP **SF**** scan	51628	26	24919
TCP **S***** scan	235386	278	35788

Fin – Scan:

Signature (click for definition)	# Alerts	# Sources	# Destinations
TCP ***F**** scan	454	28	369

Earliest such alert at **06:34:02** *on 9/27*
Latest such alert at **14:05:04** *on 11/23*

This is not a normal flag setting. It is used for intrusion detection only. The proper response, according to RFC 793, is RST-ACK from a closed port and no response from an open port. Some operating systems do not follow this specification. This response characteristic can be used for finger printing systems. This type of scan can be generated with the use of NMAP.

Correlations:
http://www.sans.org/y2k/practical/Al_Evans_GCIA.doc

Example of SnortS log record:
Sep 30 21:52:44 65.33.16.3:1694 -> 192.66.204.30:6688 FIN ***F****

Example of Snort alert command:
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"IDS27 - SCAN-FIN"; flags: F;)

Fin-Scan: Top 2 sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
211.46.110.81	271	1342	271	1137
24.6.151.155	77	987	1	2

Server used for this query: [whois.nic.or.kr]

```
Korea Internet Information Service V1.0 ( created by KRNIC, 1999.6 )
query: 211.46.110.81

IP Address          : 211.46.110.0-211.46.111.255
Connect ISP Name    : PUBNET
Connect Date        : 9847
Registration Date   : 20000118
Network Name        : YONGIN-NET

[ Organization Information ]
Orgnization ID      : ORG90300
Name                : KYONGGIDO YONGIN OFFICE OF EDUCATION
State               : KYONGGI
Address             : 195 KIMRANGJANG-DONG YONGIN-SHI
Zip Code            : 449-020
```

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

```
@Home Network (NETBLK-ATHOME)
   450 Broadway Street
   Redwood City, CA 94063
   US

   Netname: ATHOME
   Netblock: 24.0.0.0 - 24.23.255.255
   Maintainer: HOME
```

Most of the trace records from the top source address, 211.46.110.81, came as one scan during the time frame noted below. It began at the first internal address and scanned for randomly increasing ip addresses. The destination port varied between primarily 23 and occasionally 1.
Nov 10 17:43:07 211.46.110.81:4-> 192.66.1.1:23 SYNFIN **SF****
Nov 11 05:23:04 211.46.110.81:5-> 192.66.253.255:23 VECNA *****P**

Fin-Scan: Top 2 destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.162.36	77	867	1	6
192.66.227.10	4	126	1	11

UDP – Scan:

Signature (click for definition)	# Alerts	# Sources	# Destinations
UDP scan	23954	84	1420

Earliest such alert at **01:57:45** *on 9/27*
Latest such alert at **21:15:34** *on 11/23*

The UDP scan can be utilized for both mapping a network and DOS attacks. For mapping, packets are sent to several ports for each address. The expected response is none if the port is open and an "ICMP port unreachable" message if closed. Given this information, you have a good idea whether a machine lives and what ports are available.

Correlations:
http://www.sans.org/y2k/practical/Joe_Church_GCIA.doc

Example of SnortS scan record:
Sep 30 14:27:12 195.149.21.65:27045 -> MY.NET.217.202:4875 UDP

UDP – Scan: Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
63.248.55.245	9073	9073	11	11
24.9.152.152	4702	4702	1	1
192.66.5.25	2311	2311	559	559
128.61.37.65	1535	1535	4	4
24.18.90.197	982	1911	2	3

Server used for this query: [whois.arin.net] – 63.248.55.245

```
Flashcom, Inc. (NETBLK-NETBLK-FLASHCOM-2)
   5312 Bolsa Ave.
   Huntington Beach, CA 92649
   US

   Netname: NETBLK-FLASHCOM-2
   Netblock: 63.248.0.0 - 63.248.255.255
   Maintainer: FLCM
```

Server used for this query: [whois.arin.net] - 24.9.152.152

```
@Home Network (NETBLK-ATHOME)
   450 Broadway Street
   Redwood City, CA 94063
   US

   Netname: ATHOME
```

```
Netblock: 24.0.0.0 - 24.23.255.255
Maintainer: HOME
```

Top source address generating this scan, 63.248.55.245 , performed 3 different scans, the last being the largest and longest. The scans were fast and furious at times. Most of the time there was a primary target, but at times it did very amongst a very small number of different addresses. The source port was always 7777, except once, and the destination ports varied among a limited number of ports.

The first and last scan records of the longest scan are noted below.

```
Oct 30 19:49:24 63.248.55.245:7778-> 192.66.215.210:2000 UDP
Oct 30 20:59:58 63.248.55.245:7777-> 192.66.205.246:2987 UDP
```

Also, there were appeared to be UDP scan initiated from an internal machine, 192.66.5.25:67. But it appears it was booting up and looking for a boot server.

```
Oct 18 12:15:08 192.66.5.25:67-> 192.66.217.45:67 UDP
Oct 18 12:20:51 192.66.5.25:67-> 192.66.218.57:67 UDP
```

UDP – Scan: Top destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.218.50	4702	4710	1	9
192.66.206.94	1784	1799	2	17
192.66.120.36	1586	1591	9	14

SYN-FIN Scan:

Signature (click for definition)	# Alerts	# Sources	# Destinations
TCP **SF**** scan	51628	26	24919

Earliest such alert at **13:10:30** on 9/30
Latest such alert at **19:10:47** on 11/23

This is not a normal setting of TCP flags. It can be utilized for mapping out an internal network. Given the unusual flag settings, some firewalls have allowed these packets to pass thru. For linux the usual response is RST-ACK for a closed port and some combination of SYN-ACK or SYN-FIN-ACK for an open port.

Correlations:
http://www.sans.org/y2k/analysts.htm - Bill Royds #247
http://www.sans.org/y2k/practical/Markus_DeShon.html

Example of SnortS record:
Sep 30 23:24:24 213.41.69.52:21 -> 192.66.254.252:21 SYNFIN **SF****

Example of Snort alert command:
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"IDS198 - SCAN-SYN FIN";flags:SF;)

SYN-FIN Scan: Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
160.78.49.191	7182	7192	7182	7189
208.61.4.207	6634	6634	6634	6634

209.92.40.32	4956	4956	4956	4956
130.89.229.48	3860	3860	3860	3860

Server used for this query: [whois.arin.net] - 160.78.49.191

```
Centro di Calcolo di Ateneo (NET-PARMANET1)
   Centro di Calcolo di Ateneo
   Universita` di Parma
   Viale Delle Scienze
   43100 PARMA - ITALIA

   Netname: PARMANET
   Netblock: 160.78.0.0 - 160.78.255.255
```

Server used for this query: [whois.arin.net] - 208.61.4.207

```
BellSouth.net Inc. (NETBLK-BELLSNET-BLK7)
   301 Perimeter Center North,  Suite 400
   Atlanta, GA 30346
   US

   Netname: BELLSNET-BLK7
   Netblock: 208.60.0.0 - 208.63.255.255
   Maintainer: BELL
```

The top source address, 160.78.49.191, scanned the full range of the internal network. The IP addresses were randomly incremented. All source and destination ports being 53. In between this scan one record was sent to 192.66.1.3:53, which could be an DNS server.

The first and last record of the scan are noted below.

```
Sep 30 13:10:30 160.78.49.191:53-> 192.66.1.9:53 SYNFIN **SF****
Sep 30 13:10:40 160.78.49.191:1327-> 192.66.1.3:53 UDP
Sep 30 13:32:06 160.78.49.191:53-> 192.66.254.253:53 SYNFIN **SF****
```

SYN-FIN Scan: Top destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.224.79	9	18	8	16
192.66.106.204	9	14	8	13
192.66.104.90	8	14	8	13

SYN-Scan:

Signature (click for definition)	# Alerts	# Sources	# Destinations
TCP **S***** scan	235386	278	35788

Earliest such alert at **01:39:55** *on 9/27*
Latest such alert at **19:42:37** *on 11/23*

The SYN only packet is a normal situation. It is the start of a connection with a given address and port. It can also be utilized in a DOS attack or just for network mapping. The normal response to a SYN is RST-ACK for a closed port and SYN-ACK for an open port. In a denial of service attack, no following responses are sent to the victim and thus forcing the original SYN request to time out.

Correlations:

http://www.sans.org/y2k/practical/Lenny_Zeltser.htm

Example of SnortS record:
Sep 30 23:21:42 213.41.69.52:4203 -> 192.66.223.30:21 SYN **S*****

Example of Snort alert command:
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"IDS236 - SCAN-IP Eye SYN Scan"; flags: S; seq: 1958810375;)

SYN-Scan: Top sources triggering this attack signature:

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
66.9.27.254	20649	20649	19322	19322
62.252.21.241	13057	13057	8267	8267
194.244.78.145	11904	11904	1	1
63.88.175.201	11717	11718	10646	10647

Server used for this query: [whois.arin.net] - 66.9.27.254

```
Intellispace Inc. (NETBLK-ISPACE-NET-2)
   1156 Avenue of the Americas
   New York, NY 10036
   US

   Netname: ISPACE-NET-2
   Netblock: 66.9.0.0 - 66.9.223.255
   Maintainer: ITLS
```

Server used for this query: [whois.ripe.net] - 62.252.21.241

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

inetnum:      62.252.0.0 - 62.252.31.255
netname:      NTL
descr:        NTL Internet
descr:        Guildford site
country:      GB
admin-c:      NNMC1-RIPE
tech-c:       COH1-RIPE
status:       ASSIGNED PA
changed:      hostmaster@ntli.net 20001219
source:       RIPE
```

The top source address, 66.9.27.254, scanned the full range of this network. The source port address was always incrementing by 1 and the destination port was 515, print spooling services. It lasted for the duration of 3 minutes generating 20,649 scan records.

```
Nov 23 19:39:33 66.9.27.254:4904-> 192.66.1.109:515 SYN **S*****
Nov 23 19:42:37 66.9.27.254:2717-> 192.66.253.8:515 SYN **S*****
```

SYN-Scan: Top destinations receiving this attack signature:

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
192.66.220.2	11916	11926	12	15
192.66.162.77	1756	1759	6	9
192.66.60.16	1304	1306	3	4
192.66.204.26	1166	1169	6	9

The top destination address, 192.66.220.2, incurred a DOS attack during the timeframe noted below. During this time period 11,916 SYN requests where sent utilizing numerous destination ports. All packets originated from the same source address.

```
Nov 4 04:02:46 194.244.78.145:36252-> 192.66.220.2:22173 SYN **S*****
Nov 4 03:53:06 194.244.78.145:14073-> 192.66.220.2:8 SYN **S*****
```

OOS – log analysis

Earliest trace record - Aug. 17th, 05:05
Latest trace record - Nov. 23rd, 20:40
63,398 trace records

OOSche10.txt OOSche20.txt OOSche3.txt OOSche45.txt OOSche6.txt
OOSche17.txt OOSche24.txt OOSche34.txt OOSche46.txt OOSche7.txt
OOSche19.txt OOSche25.txt OOSche4.txt OOSche5.txt OOScheck.txt
OOSche2.txt OOSche29.txt OOSche44.txt OOSche50.txt

The following table reflects the addresses generating the most traffic destined for the internal network.. The port is the highest used port and the record count reflects all traffic from that address. All statistics were generated from the script noted in the Appendix.

Top source IP addresses & port	Trace count
208.61.4.207:9704	8431
210.101.101.110:9704	6508
210.101.101.110:9704	5750
130.239.133.68:6699	5551
63.195.56.20:21	4749

Server used for this query: [whois.arin.net] – 208.61.4.207

```
BellSouth.net Inc. (NETBLK-BELLSNET-BLK7)
   301 Perimeter Center North,  Suite 400
   Atlanta, GA 30346
   US

   Netname: BELLSNET-BLK7
   Netblock: 208.60.0.0 - 208.63.255.255
   Maintainer: BELL
```

Server used for this query: [whois.apnic.net] – 210.101.101.110

```
% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html

inetnum:      210.101.64.0 - 210.101.127.255
netname:      KORNET
descr:        Korea Telecom
descr:        100 Sejong-no Chongno-gu Seoul, Korea
descr:        110-777
country:      KR
```

```
admin-c:      GC1-AP
tech-c:       JK14-AP
remarks:      ISP in Korea
changed:      hostmast@rs.krnic.net 980707
source:       APNIC

person:       Gisu Choi
address:      Korea Telecom
address:      100 Sejong-no Chongno-gu Seoul, Korea
```

The following table reflects the top internal addresses sending traffic to the external network.

Top source IP addresses & port	Trace count
192.66.218.106:1226	14
192.66.218.106:1094	10
192.66.218.106:34	9
192.66.218.106:0	7
192.66.203.150:8311	7
192.66.220.142:3043	6
192.66.219.2:4431	6
192.66.217.194:2420	6

The following table reflects outgoing traffic from internal addresses utilizing known trojans ports. These machines require some review.

Source IP addresses & Port	No. of records	Known Trojan for this port
192.66.203.150:1245	1	VooDoo Doll
192.66.217.186:113	1	Invisible Identd daemon, Kazimas
192.66.217.194:1082	1	WinHole
192.66.218.106:1090	3	Xtreme
192.66.218.106:50	1	DRAT
192.66.226.234:1212	1	Kaos
192.66.100.149:23	1	Tint Telnet Server, Truva Atl
192.66.150.139:21	1	Back Construction, Blade Runner, Doly Trojan, Fore, FTP Trojan, Invisible FTP, Larva, MBT, Motiv, Net Administrator, Senna Spy FTP Server, WebEx, WinCrash
192.66.181.131:21	4	Back Construction, Blade Runner, Doly Trojan, Fore, FTP Trojan, Invisible FTP, Larva, MBT, Motiv, Net Administrator, Senna Spy FTP Server, WebEx, WinCrash

The following table reflects the highest hit internal destinations by IP address & port.

IP Address & Port	No. records
192.66.217.46:994	186

192.66.211.146:4922	8
192.66.207.142:6688	8
192.66.201.130:6688	7

The following table reflects the highest hit destination IP addresses.

IP Address	No. records
192.66.217.46	245
207.172.3.46	211
192.66.207.142	16
192.66.211.146	13

The following table reflects internal addresses that appear to be scanning external addresses.
At the very least they are issuing packets with non-normal flag sequences. Not all addresses are listed.
All of these systems need to be reviewed.

IP Address
192.66.218.106
192.66.203.198
192.66.219.2
192.66.225.54
192.66.211.130
192.66.224.2
192.66.203.150
192.66.213.138
192.66.201.14

192.66.218.106 generated traffic:

```
10/14-05:10:15.074428 MY.NET.218.106:1094 -> 207.172.3.46:119
2*SFRP** Seq: 0xA6E890   Ack: 0x7EFD   Win: 0x5010
10/14-05:29:11.656309 MY.NET.218.106:1079 -> 207.172.3.46:119
**SFR*** Seq: 0x2   Ack: 0x61FB0F35   Win: 0x5010
10/14-05:31:29.286220 MY.NET.218.106:1079 -> 207.172.3.46:119
2*SFRP*U Seq: 0x26578   Ack: 0xFF1085   Win: 0x5010
10/14-05:46:07.359568 MY.NET.218.106:1086 -> 207.172.3.46:119
21SF**AU Seq: 0x160018   Ack: 0xD6F7358E   Win: 0x5010
10/14-05:50:48.987874 MY.NET.218.106:1086 -> 207.172.3.46:119
21SFRPAU Seq: 0x18EC04   Ack: 0x3B79BA23   Win: 0x5010
```

Not all incoming and outgoing traffic was included in the logs. Below is just such an example. The internal machine sends a request on port 0 and it is received on some port. The next traces sent from the internal machine uses the receivers same port. This happens many times in the logs. Please also note the un-normal TCP flags.

```
10/23-09:07:47.871105 MY.NET.217.194:0 -> 207.172.3.46:1560
**SF*PA* Seq: 0x770288   Ack: 0xBB9036BB   Win: 0x5010
10/23-09:19:03.084369 MY.NET.217.194:1560 -> 207.172.3.46:119
*1SF**A* Seq: 0x288D6F7   Ack: 0x5540CD   Win: 0x5010
10/23-09:19:35.263836 MY.NET.217.194:1560 -> 207.172.3.46:119
**SF*P*U Seq: 0x860288   Ack: 0xD8D34150   Win: 0x5010
10/23-09:22:42.446633 MY.NET.217.194:73 -> 207.172.3.46:1560
21SF*P** Seq: 0x770288   Ack: 0xE2B84407   Win: 0x5010
```

```
10/23-09:31:19.964696 MY.NET.217.194:1560 -> 207.172.3.46:119
**SFR*** Seq: 0x288   Ack: 0xFEBE4BB5   Win: 0x5010
```

Mapping the port usage:

internal port	external port
•	1560
73	
1560	119

Overall defensive recommendations –

It is proposed the customer purchase a firewall and an IDS system. Given the amount of attacks the customer is incurring, it will help their overall security. There are a number of internal systems that require immediate attention. Reasons include generating traffic from known Trojan ports to the scanning of external networks. Details have been noted in the above review. In reference to the internally generated scanning, review of the companies Computer Usage Policy would be in order.

Assignment #3 Analyze this Analysis:

The S & A logs were processed by snortsnarf, http://www.silicondefense.com/snortsnarf/ . The machine used to run this utility was 500Mh Pentium III with 256 M of memory. On the first attempt of running snortsnarf, the system ran out of memory. Next, an additional 128M of memory was added to the machine. The 2nd execution contained only the A-logs and completed in a relatively short time. The S-logs analysis ran for approximately 45 minutes, using upwards of 300M of memory. I believe a bit of tuning may help its run time. I was unable to locate a program that would process the OOS logs. The utility I wrote to process these trace records can , be found in Appendix A. It is not pretty, but it works.

The first step in performing assignment #2, was to understand what was being requested. From there I started reviewing each log understanding the contents and format used in each. A short script was written to remove the "MY.NET" and substitute "192.66". A current version of snortsnarf was downloaded and installed. Having no experience with the utility spent some time reviewing documentation and playing with it on a single log file. My experience with running it against the remainder of the logs is noted above.

The html output from snortsnarf provided the major portion of the statistics noted in this analysis. It allowed the review of masses amounts of data quickly and relatively easily. The provided drill down capability was from the perspective of a specific alert, or from a given ip address both source and destination traffic. Given a specific alert, the usual search sites, noted below, were utilized to find information about the alert and source of the alert.

The review of the OOS logs had to be accomplished with writing a script to parse the logs and provide something useful. Also, a number of grep's were used to find what I thought might be useful or interesting. After reviewing all the OOS logs, I was able to identify some signs that traffic was going between sites. These logs do not contain all trace records for either incoming or outgoing traffic.

Some of the web sites utilized:
http://www.snort.org/
http://www.google.com/
http://www.sans.org/giac.htm
http://whitehats.com/ids/index.html
http://www.geektools.com/cgi-bin/proxy.cgi
http://www.blackcode.com/
http://www.simovits.com/nyheter9902.html

The following partial screen shots is from the perspective of a specific ip address and incoming traffic during a specific timeframe.

SnortSnarf alert page
Source: *63.193.210.208*: overview
SnortSnarf v111500.1

1883 such alerts among the files:

- SnortA10.txt.m
- SnortA11.txt.m

Earliest: **18:58:22**.389439 *on 10/05*
Latest: **19:03:42**.376854 *on 10/05*

1 different signatures are present for *63.193.210.208* as a source

- 1883 instances of *WinGate 1080 Attempt*

There are 1837 distinct destination IPs in the alerts of the type on this page.

63.193.210.208	Whois lookup at:	ARIN	RIPE	APNIC	Geektools
	DNS lookup at:	Amenesi	TRIUMF	Riherds	Princeton

```
10/05-18:58:22.389439 [**] WinGate 1080 Attempt [**] 63.193.210.208:1605->
192.66.1.10:1080
10/05-18:58:22.475974 [**] WinGate 1080 Attempt [**] 63.193.210.208:1643->
192.66.1.48:1080
10/05-18:58:22.501827 [**] WinGate 1080 Attempt [**] 63.193.210.208:1647->
192.66.1.52:1080
10/05-18:58:22.581632 [**] WinGate 1080 Attempt [**] 63.193.210.208:1683->
192.66.1.88:1080
10/05-18:58:22.628002 [**] WinGate 1080 Attempt [**] 63.193.210.208:1715->
```

The partial screen shots are from the perspective of all "Snort signatures" identified in the specified logs.

SnortSnarf start page
All Snort signatures
SnortSnarf v111500.1

110534 alerts found among the files:

- SnortA10.txt.m
- SnortA11.txt.m
- SnortA12.txt.m

Earliest alert at **00:00:52**.873106 *on 09/26*
Latest alert at **23:32:20**.988483 *on 11/22*

Signature (click for definition)	# Alerts	# Sources	# Destinations	Detail link
Happy 99 Virus	2	2	2	Summary
site exec - Possible wu-ftpd exploit - GIAC000623	6	4	4	Summary
SITE EXEC - Possible wu-ftpd exploit - GIAC000623	7	1	4	Summary

Appendix A:

The following script changed the home net address and removed the header lines.

```
for file in ls /var/log/snort/sanslogs/Snort*.txt
do
    echo sed -e "s/MY.NET./192.66./g" -e "1,16d" "$file > $file.m"
    sed -e "s/MY.NET./192.66./g" -e "1,16d" $file > $file.m
done
```

A number of grep's were used to pick through the OOS logs. The following picked out all the internally generated traffic and the TCP flags:

```
grep -e "[0123456789] MY.NET" -e "Ack"
```

The following script was written to process the OOS trace records. Like I said it isn't pretty.

```
#
#   scan OOS snort logs and create something usefull. the ports.list file
#   is a list containing ports and common trojans.
#
portlist=/var/log/snort/sanslogs/ports.list
#
for file in /var/log/snort/sanslogs/OOSche*.txt
do
    echo sed -e "s/MY.NET./192.66./g"  "$file > $file.a"
#
#   read OOS???? file and adjust home ip address and get first record
#
    grep "^../.." $file | sed -e "s/MY.NET./192.66./g"   | tr -s "\015" " " >  temp.a
    echo "sed completed"
#
#   temp.a - network record first part of trace record
#   08/17-00:46:13.596937 24.23.198.174:0 -> 192.66.217.46:2855
#
#   sort on ip source  address to get unique list and  sort again on ip address
#   then read each port number and check against known trojan port list.
#
    sort -k2 temp.a | cut -f2 -d" "  > temp.sa
    cat temp.sa |  sort -u | sort -g  > temp.su
    >temp.st
    while read ipaddr
```

```
    do
      trojan=$(echo $ipaddr | cut -f2 -d":" | xargs -n1 -i grep -w "^{}" $portlist )
      if [ "$trojan" = "" ] ; then
         trojan="______"
      fi
      echo $trojan >> temp.st
   done < temp.su

   cat temp.su | xargs -n1  -i -P 5  grep -c -w -F {} temp.sa > temp.sc
   echo "ip source analysis complete"
#
#   working on destination addresses.
#
    cat temp.a | cut -f4 -d" "      > temp.da
    cat temp.da |  sort -u | sort -g   > temp.du2
    > temp.dc2
    > temp.dc22
    > temp.out
    > temp.du.total
    lastipaddr=""
    while read ipaddrport
    do
       ipaddr=$(echo $ipaddrport | cut -f1 -d":")
       if [ "$lastipaddr" != "$ipaddr" ] ; then
          grep -w -F "$ipaddr"  temp.da > temp.out
          totalrecs=$(wc -l temp.out | tr -s " " | cut -f2 -d" ")
          echo "$ipaddr          $totalrecs"    >> temp.du.total
          lastipaddr=$ipaddr
       fi
#
       grep -c -w -F $ipaddrport  temp.out >> temp.dc2
       echo "$totalrecs"    >> temp.dc22
#
    done < temp.du2
    echo "destination completed"
#
#   temp.u - source ip addresses
#   212.187.21.156:21
#
#   take each source ip address and count number of trace records in OOS?? file
#
#
#   temp.c - source ip addresses and trace count
#   212.187.21.156:21      1136
#
#
#   sort by total hits for a port
#
    echo "$file.a  - $(wc -l temp.sa) " > $file.a
    echo >> $file.a
    echo "top 25 hits by port" >> $file.a
    echo "source              no. of     known " >> $file.a
    echo "ip addressess       records    trojans" >> $file.a
    paste temp.su temp.sc temp.st | sort -k2 -g -r | head -n 25 >> $file.a
    echo >> $file.a
    echo "report 1 complete"
#
#   sort by total hits
#
    echo >> $file.a
    echo "possible trojan   " >> $file.a
    echo "source              no. of     known " >> $file.a
    echo "ip addressess       records    trojans" >> $file.a
    paste temp.su temp.sc temp.st | grep -v -F "______"  | sort -k2 -g -r >> $file.a
    echo >> $file.a
    echo "report 2 complete"
#
```

```
#   sort by total port hits destination ip addresses
#
    echo >> $file.a
    echo "by port" >> $file.a
    echo "highest hit " >> $file.a
    echo "destination           no. of      no. of" >> $file.a
    echo "ip addressess         records     records" >> $file.a
    paste temp.du2 temp.dc2 temp.dc22 | sort -g -r -k2 | head -n 50 >> $file.a
    echo "report 3 complete"
#
#   sort by total destination hits
#
    echo >> $file.a
    echo "by " >> $file.a
    echo "highest hit " >> $file.a
    echo "destination           no. of      " >> $file.a
    echo "ip addressess         records     " >> $file.a
    cat  temp.du.total | sort -g -r -k2 | head -n 50 >> $file.a
    echo "report 4 complete"
done
# rm temp.*
```