SANS GIAC Certified Intrusion Detection Analyst Practical
Shong Chong, Network Security Engineer, Perot Systems

Assignment 1: Network Detect Analysis

Detect #1:

Snort Alert:

```
10/19-21:04:40.081633 [**] IDS277 - NAMED Iquery Probe [**]
211.171.245.146:4813 -> Our.SMTP.host:53
10/19-21:04:40.325413 [**] MISC-DNS-version-query [**]
211.171.245.146:4813 -> Our.SMTP.host:53
```

Snort Log:

```
10/19-21:04:40.081633
211.171.245.146:4813 -> our.DNS.host:53 UDP
TTL:46 TOS:0x0 ID:19439 Len: 35
F5 BB 09 80 00 00 00 01 00 00 00 00 00 00 01 00
     ................
01 00 00 7A 69 00 04 04 03 02 01             ...zi......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
+=+=+=+

10/19-21:04:40.325413
211.171.245.146:4813 -> our.DNS.host:53 UDP
TTL:46 TOS:0x0 ID:19482 Len: 38
22 D5 01 80 00 01 00 00 00 00 00 00 07 76 65 72
     "...........ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03
     sion.bind...
```

1. **Source of trace**: http://www.sans.org/y2k/102300.htm

2. **Detects Generated by**: Snort Intrusion Detection Systems
   Log Format:

   Snort Alert:
   Format: Date/Time [**] Type of Alert [**]
   Source IP: Source Port -> Destination IP: Destination Port

   Snort Detail Log:
   Date-Time Source IP Address: Source Port → Destination IP: Destination Port
   Time-To-Live Type Of Service IPID Length

3. **Probability the source address was spoofed**: Source address not spoofed since response is needed to gather information. Korean IP address.

4. **Description of attack:** Attack on UDP port 53 DNS. CVE-1999-0009.

5. **Attack Mechanism:**
   The attacker attempts to determine if a name server that supports IQUERY. At first he probed to determine if the target allows inverse query requests. The second probe is to determine the version of BIND running on the remote host. These two queries are usually seen as a pre-attack probe, prior to an attempted buffer overflow of Named.

   In 1998 a buffer overflow was discovered that affects certain versions of BIND, the name server daemon currently maintained by the Internet Software Consortium. These older versions of the BIND software would fail to properly bind the data received when processing an inverse query. Upon a memory copy, portions of the program would be overwritten, and arbitrary commands could be run on the affected host.

6. **Correlation:** This detect is not correlated to any other activity. This is older DNS bind version exploit that came out in April 08, 1998.
   Excerpts taken from:
   http://www.whitehats.com/IDS/277 and
   http://www.whitehats.com/IDS/278

   Also reference:
   CVE-1999-0009 → http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0009
   BugtraqID 133 →
http://www.securityfocus.com/vdb/bottom.html?vid=133
   BugtraqID 134 →
http://www.securityfocus.com/vdb/bottom.html?vid=134

7. **Evidence of active targeting**: PerHAPS. Not enough information is given on this trace. The trace is targeting the DNS server -- if the name server is the only target - then the answer is yes. The attacker has done some information gathering.
   If the same type of alert is seen target servers across the network, then the answer is no.

8. **Severity**: (Assuming target system is patched).
   (Critical + Lethal) - (System + Net Countermeasures)
   (5 + 4) - (4 + 2) = **3**
   Criticality: DNS Server: 5

Lethality: Possible Root Access: 5
System Countermeasure: Patched: 4
Net Countermeasure: Traffic is allow to target: 2

9. **Defensive Recommendation**: Do not run older version of BIND (pre 8.1.2 / 4.9.8). Make sure Operating systems and BIND patch level is up to date to prevent possible unknown exploits.

10. **Multiple Choice Question**:
    Examine the logs. What is UDP port 53 and what is the exploit?
    A. NMAP. Host and Port Scan.
    B. DNS. Overflow of older version BIND server.
    C. HTTP. Root access to Microsoft IIS server.
    D. Back Orifice 2K. Remote administration.
    E. DNS. Server Name Query (reconnaissance probe).

    Answer: B

Detect #2:

```
[**] IDS007 - MISC-Source Port Traffic 53 TCP [**]
10/11-16:34:24.533919 210.178.9.1:53 -> xxx.xxx.16.1:111
TCP TTL:238 TOS:0x0 ID:62128 **S***** Seq: 0x64 Ack: 0x0
Win: 0x200 –
[**] IDS007 - MISC-Source Port Traffic 53 TCP [**]
10/11-22:10:09.378322 210.178.9.1:53 -> xxx.xxx.16.2:111
TCP TTL:238 TOS:0x0 ID:62128 **S***** Seq: 0x64 Ack: 0x0
Win: 0x200 –
[**] IDS007 - MISC-Source Port Traffic 53 TCP [**]
10/12-04:01:40.930951 210.178.9.1:53 -> xxx.xxx.16.3:111
TCP TTL:238 TOS:0x0 ID:62128 **S***** Seq: 0x64 Ack: 0x0
Win: 0x200
[**] IDS007 - MISC-Source Port Traffic 53 TCP [**]
10/12-08:23:23.291955 210.178.9.1:53 -> xxx.xxx.16.4:111
TCP TTL:238 TOS:0x0 ID:62128 **S***** Seq: 0x64 Ack: 0x0
Win: 0x200 –
[**] IDS007 - MISC-Source Port Traffic 53 TCP [**]
10/12-12:44:20.783145 210.178.9.1:53 -> xxx.xxx.16.5:111
TCP TTL:238 TOS:0x0 ID:62128 **S***** Seq: 0x64 Ack: 0x0
Win: 0x200
```

1. **Source of trace**: http://www.sans.org/y2k/110100.htm

2. **Detects Generated by**: Snort Intrusion Detection Systems
   Log Format:

   Snort Alert:
   [**] Type of Alert [**]
   Date-Time Source IP: Source Port -> Destination IP: Destination Port
   Protocol: Time-To-Live: Type-Of-Service: IP-ID: TCP Flags (Syn????):Sequence# Window Size

3. **Probability the source address was spoofed**:
   Source address not spoofed since response is needed to gather information.

4. **Description of attack**: Suspicious Traffic: Source Port-53-tcp. Malicious Traffic: Destination port 111 (Port Mapper). Attacker is making a connection using the source port 53 (DNS) to a privileged port (RPC 4.0 port-mapper). This should not occur naturally - and is meant to fool old or mis-configured packet-filters into allowing the connection, as they sometimes allow all DNS traffic.

5. **Attack Mechanism**:
This is an example portmapper (port 111) probe in combination with source port 53(DNS) exploit. Unfortunately, additional data dump from snort is required to determine which RPC service the attacker is probing for.

An attacker is making a connection using the source port 53 (DNS) to a privileged port (RPC 4.0 port-mapper). In this case, if the packetfilter was mis-configured to allow source porting, this request would get through even if portmapper(TCP port111) was specifically filtered out. This problem is due to design flaws in the consideration of some packetfilters. A typical ruleset would have a rule to allow DNS traffic, by allowing traffic that had source port 53. However, attackers can easily set their source port to 53 using a tool like netcat, and circumvent this type of primitive filter.

6. **Correlation**: This detects is not correlated to any other activity. This is an exploit targeted towards mis-configured packetfilter.

Exerpts taken from:
http://www.whitehats.com/IDS/7 .

Also reference:

CVE Version: 20001013 (GENERIC-MAP-NOMATCH)→
http://cve.mitre.org/cgi-bin/cvename.cgi?name=GENERIC-MAP-NOMATCH

7. **Evidence of active targeting**: Probably. All packet ID's , TTL's, and other options are identical on each packet. The pattern is low and slow, one packet every 3-5 hours, at most 2 packets within an hour with a long gap. Whoever it is appears to be very patient or is scanning a really huge address range. Not enough information available.

8. **Severity**:(Assuming target system is a file server)
(Critical + Lethal) – (System + Net Countermeasures)
(3 + 4) – (3 + 5) = **-1**
Criticality: File Server: 3
Lethality: Possible Exploitable RPC Info Leak: 3
System Countermeasure: Decently Patched: 3
Net Countermeasure: Statefull Firewall Drop Packets: 5

9. **Defensive Recommendation**: Make sure Firewall or Packet Filtering Device is not vulnerable to this type of attack. If possible, deploy Statefull firewall. Use RPC authentication to secure RPC.

10. **Multiple Choice Question**:

Examine the logs. Why does the attacker have source port of 53?

A. It’s the ephemeral port on his PC.
B. It’s a probe to resolve DNS name.
C. It’s the default port for querying portmapper (port 111).
D. It’s a way to bypass mis-configure packet filter device.

Answer: D

Detect #3:

Date	*Time*	*Interface*	*Action*	*Protocol*	*Service*	*Source IP*	*Destination IP*	*Source Port*
20-Nov-00	11:00:46	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	61374
20-Nov-00	11:02:19	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	61374
20-Nov-00	11:03:55	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	61374
20-Nov-00	13:44:47	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62703
20-Nov-00	13:44:49	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62702
20-Nov-00	13:45:29	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62709
20-Nov-00	13:45:30	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62711
20-Nov-00	13:46:05	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62715
20-Nov-00	13:46:06	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62716
20-Nov-00	13:46:20	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62703
20-Nov-00	13:47:02	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62709
20-Nov-00	13:47:03	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62711
20-Nov-00	13:47:38	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62715
20-Nov-00	13:47:39	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62716
20-Nov-00	13:47:55	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62702
20-Nov-00	13:47:56	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62703
20-Nov-00	13:48:38	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62709
20-Nov-00	13:48:39	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62711
20-Nov-00	13:49:14	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62715
20-Nov-00	13:49:15	qfe1	drop	tcp	ident	209.115.70.194	My.FW.X.2	62716

1. **Source of trace**: My Network
2. **Detects Generated by**: Checkpoint Firewall-1 (version 4.0)

3. **Probability the source address was spoofed**: Source address not spoofed since response is needed to gather information.

4. **Description of attack**: Suspicious Traffic ident-version-probe: A remote user may have used the "ldistfp" tool to attempt to fingerprint an ident server. If the probe was successful the remote user will be able to determine the ident server software version, and the operating system of the target host.

5. **Attack Mechanism**:
   Not sure. This is why our network needs an Intrusion Detection System. The information from Checkpoint Firewall-1 logs is not bad for a firewall – but not sufficient to do Intrusion Detection. More information is needed to analyze and examine the content of the detects.

   One of the most common attack associated with IDENT protocol is ident-version-probe.

   The attacker use "ldistfp" program to attempt to connect to the identd authentication service of the target. It will try to determine the identd version running. It will then look up the response in a pre-made database and find the appropriate version line. This information can be mapped back to the distribution and it's version used on this host. Simple but effective, since the identd authentication service is used almost everywhere and most people don't know about its version capabilities.
   This information-gathering probe may be used by attackers to locate known-vulnerable Operating System versions for further exploitation.

   The IDENT protocol is sometimes used by POP mail, FTP, and HTTP servers to identify incoming users. When a user requests a service, the server tries to initiate an IDENT connection back towards the client behind the firewall. In this case, the source IP address is a HTTP server. A who-is lookup from American Registry for Internet Numbers (ARIN) indicates that the IP belongs to a company named Fiber Network Solutions, Inc. Typing in the IP address in web browser bring up a personal web site, which indicate that it could be a false alarm (example if someone from within our network visited the web site).

6. **Correlation**: They are more scans from different IP addresses for the same service. Each and every one of the rest of the IP addresses correlates to a web site respectively. Look like this is a false alarm.

Exerpts taken from:
http://www.whitehats.com/IDS/303

Also reference:

CVE Version (Candidate): CAN-1999-0629 →
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0629

7. **Evidence of active targeting**: No. Every source IP addresses correlate to a web site respectively.

8. **Severity**:
   (Critical + Lethal) – (System + Net Countermeasures)
   (0 + 2) – (4 + 5) = **-7**
   Criticality: (Non-attack): 0
   Lethality: Possible Information Gathering: 2
   System Countermeasure: Not Running IDENT: 4
   Net Countermeasure: Statefull Firewall Drop Packets : 5

9. **Defensive Recommendation**: None. If possible, deploy Snort outside Firewall to help network detect analysis.

10. **Multiple Choice Question**:
   Examine the logs. What is the exploit?
   A. TCP Port Scanning
   B. Teardrop Attack
   C. Information Gathering on My.FW.X.2
   D. Not enough information – could be just normal traffic from a web site a user is visiting.
   E. Both C & D

   Answer: E

Detect #4:

```
[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:07:29.531459 24.69.154.150:2461->
xxx.xxx.xxx.xxx:27374
TCP TTL:115 TOS:0x0 ID:53527 DF
**S***** Seq: 0x13584E Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:07:30.301525 24.69.154.150:2461->
xxx.xxx.xxx.xxx:27374
TCP TTL:115 TOS:0x0 ID:56087 DF
**S***** Seq: 0x13584E Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:07:30.979470 24.69.154.150:2461->
xxx.xxx.xxx.xxx:27374
TCP TTL:115 TOS:0x0 ID:61207 DF
**S***** Seq: 0x13584E Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:08:25.262217 24.69.154.150:2536->
xxx.xxx.xxx.xxx:27374
TCP TTL:115 TOS:0x0 ID:29210 DF
**S***** Seq: 0x13E00E Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

-----------------------------------------------------------
--------

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:47:17.211108 24.26.94.160:2477->
xxx.xxx.xxx.xxx:27374
TCP TTL:112 TOS:0x0 ID:58390 DF
**S***** Seq: 0x1CC8D1 Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:47:20.124738 24.26.94.160:2477->
xxx.xxx.xxx.xxx:27374
TCP TTL:112 TOS:0x0 ID:8215 DF
**S***** Seq: 0x1CC8D1 Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:47:26.129891 24.26.94.160:2477->
xxx.xxx.xxx.xxx:27374
TCP TTL:112 TOS:0x0 ID:12311 DF
**S***** Seq: 0x1CC8D1 Ack: 0x0 Win: 0x2000
```

```
TCP Options => MSS: 1460 NOP NOP SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-16:47:38.133247 24.26.94.160:2477->
xxx.xxx.xxx.xxx:27374
TCP TTL:112 TOS:0x0 ID:42263 DF
**S***** Seq: 0x1CC8D1 Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK

-----------------------------------------------------------
--------

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-11:32:17.941027 209.179.141.168:4973->
xxx.xxx.xxx.xxx:27374
TCP TTL:16 TOS:0x0 ID:38002 DF
**S***** Seq: 0xA8266AEA Ack: 0x0 Win: 0x2000
TCP Options => MSS: 536 NOP WS: 0 NOP NOP TS: 0 0 NOP NOP
SackOK

[**] IDS279 - BACKDOOR ATTEMPT-Subseven v2.1 [**]
10/09-11:32:18.485798 209.179.141.168:1061->
xxx.xxx.xxx.xxx:27374
TCP TTL:16 TOS:0x0 ID:38064 DF
**S***** Seq: 0xA8550CFA Ack: 0x0 Win: 0x2000
TCP Options => MSS: 536 NOP WS: 0 NOP NOP TS: 0 0 NOP NOP
SackOK
```

1. **Source of trace**: http://www.sans.org/y2k/101100.htm

2. **Detects Generated by**: Snort Intrusion Detection Systems

3. **Probability the source address was spoofed**: Source address not spoofed since response is needed to gather information.

4. **Description of attack**: Sub-Seven2.1 Scan:
   This is a scan or probe for a known trojan (Sub-Seven2.1) that may be operating on the host. If both syn/ack flags were set, then it would be a response to a connection request. This is the default port used by SubSeven-2.1/2.2-Gold.

5. **Attack Mechanism**:
   This is not a response to a connection request. The attacker is probably just scanning for any host that was compromised and responded to the probe.
   This trojan is normally distributed as a Win32 PE exe dropper that may be disguised as a JPG or BMP picture. When run, this dropper installs two files into the WINDOWS folder of the user's hard disk. These two files are the main server exe files, normally called

"MSREXE.EXE", and a loader program normally called "RUN.EXE", "WINDOS.EXE" or "MUEEXE.EXE".
This signature matches the known default port of the trojan. It is possible that other server software could listen at the same port.

Most commonly these trojans are limited "remote administration tools" that allow an attacker to take complete control over the victim server. Client desktop machines in Window 9x/NT environments are most likely to suffer from trojan infections. Trojans are usually installed as a disguise in an email attachment, or hidden in other software available for download.

This trojan is the result of further development of the BackDoor-G trojan (v1.0 - v1.9) and offers the usual access to the user files and data on his system via the Internet.

By default the Trojan uses TCP port 27374, but this is configurable by the configuration program.
If both syn/ack bits were set, then it would really be nervous time. That means it is a response to a connection request. If this is the case, the target hosts are probably compromised.

6. **Correlation**: They are three sets of scans from three different IP addresses for the same service.
Speculation: May be the attacker targeted this network/company via an email with attachment – for some reason hackers seems to think that Subseven is install here.

Exerpts taken from:
http://www.whitehats.com/IDS/279

Also reference:

CVE Version (Candidate): CAN-1999-0660 →
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0660
See also: http://vil.nai.com/vil/RAT10566.asp

7. **Evidence of active targeting**: If all three probes from three different IP addresses are targeting the same destination IP address, then answer is yes. The target system's IP address is totally masked out.

8. **Severity**: Assuming target is a User Workstation
   (Critical + Lethal) – (System + Net Countermeasures)
   (2 + 4) – (3 + 5) = **-2**
   Criticality: User Workstation: 2
   Lethality: Possible Compromise Host Inside Network (and spread to other critical systems from compromised system): 4
   System Countermeasure: Running Antivirus and Up to date Definition File: 3
   Net Countermeasure: Statefull Firewall Drop Packets : 5

9. **Defensive Recommendation**:
   Examine target host for possible active sub-seven trojan. Run good Antivirus software and up to the minute virus definition file. Implement an automated virus definition file update via either vendor software utilities (such as Norton Antivirus AutoUpdate) or SMS scripts.
   If it is confirmed that the target host is indeed compromised, then refer to the trojan removal help page at http://www.whitehats.com/ids/trojan/.

10. **Multiple Choice Question**:
    Examine the logs. Select the best answer:
    If both syn/ack bits were both set, _____________.
    A. It's a sub-seven scan.
    B. It's Teardrop Attack.
    C. Nervous Time – it's a response to a connection request. Possible active sub-seven host.
    D. Not enough information – could be just normal RPC traffic (Response to a RPC request).
    E. None of the above.

    Answer: C

Assignment 2: Evaluate an Attack

Location attacks was acquired from:

Nmap: http://www.insecure.com/nmap/index.html

Subseven Trojan: http://subseven.slak.org/main.html

Description of Tools:

Nmap V. 2.08*:*
Nmap is an information gathering tools that is widely available. Nmap is usually used for network exploration or security auditing. Because of the same reason, it is also a very powerful network-scanning tool. It is designed to scan large networks to determine which hosts are up and what services are listening on those hosts.
Nmap supports ping scanning (determine which hosts are up), many port scanning techniques (determine what services the hosts are offering), and TCP/IP fingerprinting (remote host operating system identification). Nmap also offers flexible target and port specification, decoy scanning, determination of TCP sequence predictability characteristics, sunRPC scanning, reverse-identd scanning, and more.

Example Usage:

```
#> nmap -P0 -v -sS    targeted_network/netmask_bits
```

Nmap will issue a TCP SYN (Half-Open) scan on target network, without pinging it first. This is usually use for sites that don't allow incoming ping.

```
Usage: nmap [Scan Type(s)] [Options] <host or net #1 ...
[#N]>

Nmap options of interest:

   -sT    tcp connect() port scan
   -sS    tcp SYN stealth port.
   -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (only works
against UNIX).
   -sP    ping "scan". Find which hosts on specified
network(s) are up but don't
          port scan them
   -sU    UDP port scan, must be r00t
   -b <   ftp_relay_host> ftp "bounce attack" port scan
   -f     use tiny fragmented packets for SYN, FIN, Xmas,
or NULL scan.
   -P0    Don't ping hosts (needed to scan
www.microsoft.com and others)
   -PT    Use "TCP Ping" to see what hosts are up (for
normal and ping scans).
   -PT21Use "TCP Ping" scan with probe destination port of
21 (or whatever).
   -PI    Use ICMP ping packet to determines hosts that are
up
   -PB    Do BOTH TCP & ICMP scans in parallel (TCP dest
port can be specified after the 'B')
   -PS    Use TCP SYN sweep rather than the default ACK
sweep used in "TCP ping"
   -O     Use TCP/IP fingerprinting to guess what OS the
remote host is running
   -p <range>  ports: ex: '-p 23' will only try port 23 of
the host(s)
      '-p 20-30,63000-' scans 20-30 and 63000-65535.
default: 1-1024 + /etc/services
   -D     decoy_host1,decoy2,ME,decoy3[,...]  Launch scans
from decoy host(s) along
          with the real one.  If you care about the order
your real IP appears,
          stick "ME" somewhere in the list.  Even if the
target detects the
          scan, they are unlikely to know which IP is
scanning them and which are decoys.
   -F     fast scan. Only scans ports in /etc/services, a
la strobe(1).
   -I     Get identd (rfc 1413) info on listening TCP
processes.
   -i <inputfile>   Grab IP numbers or hostnames from file.
Use '-' for stdin
   -g     <portnumber>   Sets the source port used for
scans.  20 and 53 are good choices.
   -S     <your_IP>      If you want to specify the source
address of SYN or FYN scan.
   -v     Verbose. Its use is recommended.  Use twice for
greater effect.

Hostnames specified as internet hostname or IP address.
Optional '/mask' specifies subnet.
```

SubSeven Trojan:

Subseven trojan is a "remote administration tools" that allow an attacker to take complete control over the victim server. Client desktop machines in Window 9x/NT environments are most likely to suffer from trojan infections. If somehow subseven server is installed on victim machine, it will listen on a pre-determine port, ready to be “enslaved”.

Sub-seven use a client and a server file. The client file (named server.exe) needs to be executed on the target system. Server.exe is also configurable by using Editserver.exe. Once the client is installed, the attacker simply run the Subseven.exe and connects to the victim. The client can also be configured to announce its presence via ICQ, IRC and SMTP mail.

Overview of Attack:

Nmap is usually used to probe a targeted network for active host and their listening port.
Sub-seven is usually are usually installed by disguise in an email attachment, or hidden in other software available for download.

How the attack could work (Imaginary Scenario):

Most of the following information is imaginary events -- read this like you are reading a novel – and keep in mind that not everything has make sense all the time!(especially in a novel) ☺

However, the attacks are real. The nmaps scans and subseven trojan attack and snort detects are real – they are run at my home network in an isolated environment.

A careful hacker has targeted VICTIMCORP, Inc. He quickly learned that the company utilized IP address in the range of 24.94.16.XXX via social engineering and some research on the internet. While doing his social engineering, he also realize that VICTIMCORP is not big on Network Security.
Now time for action: First he run a nmap host scan of the entire 24.94.16.XXX range to gather more information.
From this he start to map the servers that are up. Now he can target specific host.
From the map he ran a nmap port scan for the particular host he is interested using 32 bit netmask.

So now he knows one of the server is listening on port XX and YY. So he picked port XX and configures the subseven client to listen on port XX.

Now he sends hundreds of email to the targeted domain: XXX@VICTIMCORP.COM

For this assignment, let's imagine if the victim received an e-mail from StephenNorthcutt@hotmail.com with the following message:

Dear Student,
SANS want to thank you for attending SANS 2000 at Monterey California. If you have any feedback, please use the attached feedback form.

- Feedback

Thanks,
Stephen.

The file Feedback is actually a script that installs server.exe on the victim machine while run some bogus feedback form. Sharp-eyed user will be alerted by the @hotmail.com domain but for most users on Monday morning before coffee, this minor detail might just slip pass their weary eyes.

Anyhow, to prove this is doable and the unsecured nature of free Internet e-mail, I went ahead and register StephenNorthcutt@hotmail.com with the name of Stephen Northcutt. (I will gladly give up the account to SANS for a fee – just kidding ☺). This is just to illustrate how easy it is to exploit the open nature if the Internet.

Now, once he is notifed that the client software is installed, it's time for action. (Or he can just probe to see if any of the target systems is infected.)
Boom!
He got in and now he has more than enough access and information to really do some damage. He has taken complete control over the victim server.

How the attack really work

1. **Using Nmap to scan for live host on 24.94.16.X:**
   Command*:*
   #>nmap –v –sP 24.94.16.X/24

```
Host  (24.94.16.0) appears to be down.
Host  (24.94.16.0) appears to be down.
Host  (24.94.16.1) appears to be down.
Host  (24.94.16.2) appears to be down.
Host  (24.94.16.3) appears to be down.
Host  (24.94.16.4) appears to be down.
Host  (24.94.16.5) appears to be down.
Host  (24.94.16.6) appears to be down.
Host  (24.94.16.7) appears to be down.
Host  (24.94.16.8) appears to be down.
Host  (24.94.16.9) appears to be down.
Host  (24.94.16.10) appears to be down.
---Intermediate logs taken out---
Host  (24.94.16.251) appears to be down.
Host  (24.94.16.252) appears to be down.
Host  (24.94.16.253) appears to be down.
Host  (24.94.16.254) appears to be down.
Host  (24.94.16.255) appears to be down.
Nmap run completed -- 256 IP addresses (4 hosts up)
scanned in 0 seconds.
```

Snort Alert from the event:

```
   [**] spp_portscan: PORTSCAN DETECTED from X.X.X.16
   (THRESHOLD 10 connections exceeded in 0 seconds) [**]
   11/21-15:29:45.474338
   [**] IIS - Possible Attempt at NT INETINFO.EXE 100% CPU
Utilization [**]
   11/21-15:29:45.626630 24.94.16.16:49267 ->
24.94.16.200:1031
   TCP TTL:56 TOS:0x0 ID:49308
   **S***** Seq: 0x635BC4E3   Ack: 0x0   Win: 0x400

   [**] MISC-WinGate-1080-Attempt [**]
   11/21-15:29:45.772090 24.94.16.16:49267 ->
24.94.16.200:1080
   TCP TTL:56 TOS:0x0 ID:64777
   **S***** Seq: 0x635BC4E3   Ack: 0x0   Win: 0x40
```

Snort Logs from the event:

```
   [**] IIS - Possible Attempt at NT INETINFO.EXE 100% CPU
Utilization [**]
   11/21-15:29:45.626630 24.94.16.16:49267 ->
24.94.16.200:1031
   TCP TTL:56 TOS:0x0 ID:49308
   **S***** Seq: 0x635BC4E3   Ack: 0x0   Win: 0x400

   =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=+=+=+
   [**] MISC-WinGate-1080-Attempt [**]
   11/21-15:29:45.772090 24.94.16.16:49267 ->
24.94.16.200:1080
   TCP TTL:56 TOS:0x0 ID:64777
   **S***** Seq: 0x635BC4E3   Ack: 0x0   Win: 0x400

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

2. **Using Nmap to scan to listening port on target host 24.94.16.16**
   #>nmap -v -P0 -sS 24.94.16.16/32

```
   Starting nmap V. 2.08 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
   Initiating SYN half-open stealth scan against
(24.94.16.200)
   Adding TCP port 80 (state Open).
   Adding TCP port 139 (state Open).
```

```
The SYN scan took 1 seconds to scan 1477 ports.
Interesting ports on  (24.94.16.200):
Port    State       Protocol  Service
53      open        tcp         Name Service
139     open        tcp        netbios-ssn
Nmap run completed -- 1 IP address (1 host up) scanned in
1 second
```

Snort Alert from the event:

```
[**] spp_portscan: portscan status from 24.94.16.16: 1195
connections across 1 hosts: TCP(1195), UDP(0) [**]
11/21-15:32:41.566246

[**] spp_portscan: End of portscan from 24.94.16.16:
TOTAL time(0s) hosts(1) TCP(1195) UDP(0) [**]
11/21-15:37:56.489828
```

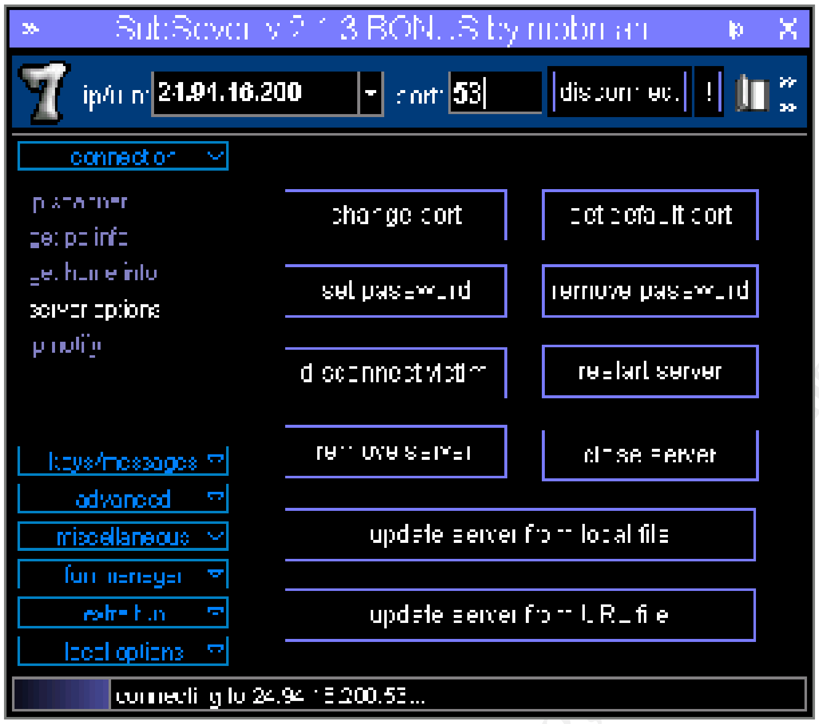
Snort Port-Scan Log:

```
  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:317 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:779 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:4321
SYN **S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:2025
SYN **S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:832 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:739 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:329 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:736 SYN
**S*****

  Nov 21 15:17:08 24.94.16.16:56879 -> 24.94.16.200:1992
SYN **S*****

  Nov 21 15:17:09 24.94.16.16:56880 -> 24.94.16.200:158 SYN
**S*****
```

```
   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:743 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:421 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:537 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:590 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:3086
SYN **S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:2011
SYN **S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:663 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:5717
SYN **S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:820 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:911 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:441 SYN
**S*****

   Nov 21 15:29:45 24.94.16.16:49267 -> 24.94.16.200:849 SYN
**S*****
   …………        ……………………………                    ………………..…
   …………        ……………………………                    ………………..…
   …
```

Notice how the port numbers are scan in random and how fast the scan is.

3. **Sub-seven Log-In:**
Screen shot from the subseven login:



Snort Alert from the event:

```
     [**] IDS279 - BACKDOOR SIGNATURE - SubSeven 2.1 Login
Detected! [**]
     11/21-15:32:41.545485 24.94.16.200:53 ->
24.94.16.20:1036
     TCP TTL:32 TOS:0x0 ID:25117  DF
     *****PA* Seq: 0x3CC3B4   Ack: 0x3EB943   Win: 0x2238
```

Snort Log from the event:

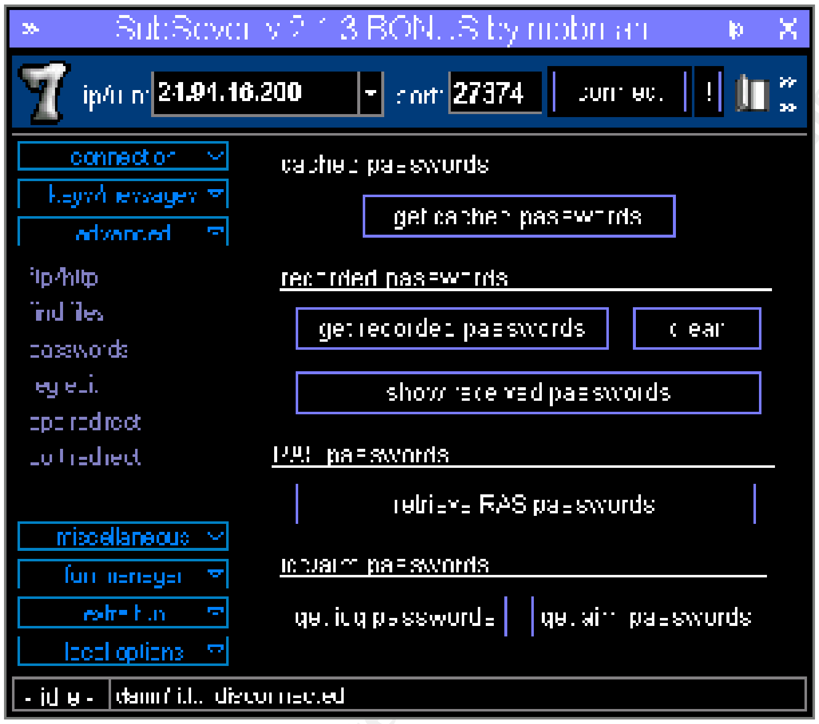
```
     [**] IDS279 - BACKDOOR SIGNATURE - SubSeven 2.1 Login
Detected! [**]
     11/21-15:32:41.545485 24.94.16.200:53 ->
24.94.16.20:1036
     TCP TTL:32 TOS:0x0 ID:25117  DF
     *****PA* Seq: 0x3CC3B4   Ack: 0x3EB943   Win: 0x2238
     63 6F 6E 6E 65 63 74 65 64 2E 20 74 69 6D 65 2F
connected. time/
     64 61 74 65 3A 20 31 35 3A 33 34 2E 31 35 20 2D  date:
15:34.15 -
     20 4E 6F 76 65 6D 62 65 72 20 32 31 2C 20 32 30
November 21, 20
     30 30 2C 20 54 75 65 73 64 61 79 2C 20 76 65 72  00,
Tuesday, ver
     73 69 6F 6E 3A 20 42 6F 4E 75 53 20 32 2E 31     sion:
BoNuS 2.1
```

+=+

4. SubSeven Activities/Features of Interest:

Password retrieval:

- Cached password and recorded password can be retrieved via this screen. Scary (Useful if you are a hacker – keep it for future use).

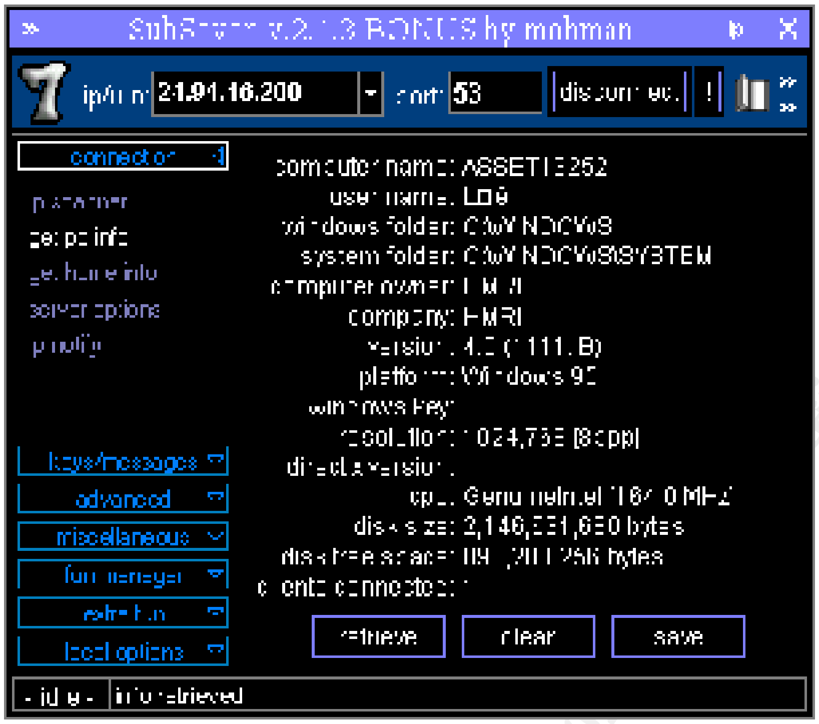


ip/uin: 24.91.16.200
port: 27374

- idle - damn! i... disconnected

PC (Target Information Retrieval):

- As we can see, it is quite detailed -- as good as you can get as if you were on the machine!

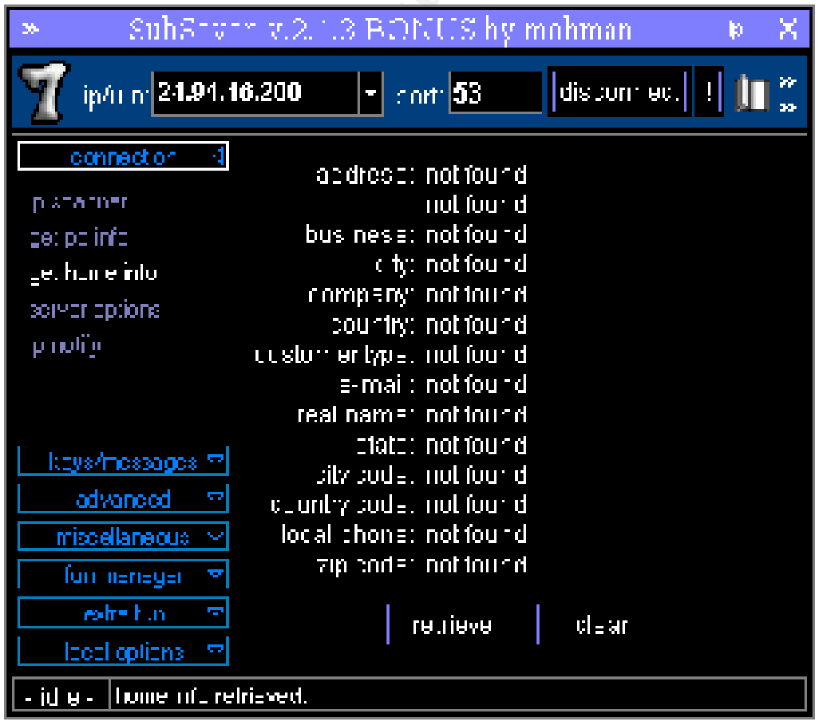


```
   computer name: ASSET13252
       user name: [illegible]
  windows folder: C:\WINDOWS
   system folder: C:\WINDOWS\SYSTEM
  computer owner: [illegible]
         company: HMRI
         version: 4.0 (1111.B)
        platform: Windows 95
     windows key:
      resolution: 1024,768 (8bpp)
directx version:
             cpu: GenuineIntel [illegible] MHz
       disk size: 2,146,631,680 bytes
 disk free space: [illegible] bytes
current connections:
```

- idle - info retrieved

Personal Information On PC:

- Aren’t you glad you did not enter any personal information into Microsoft “enhanced” program?

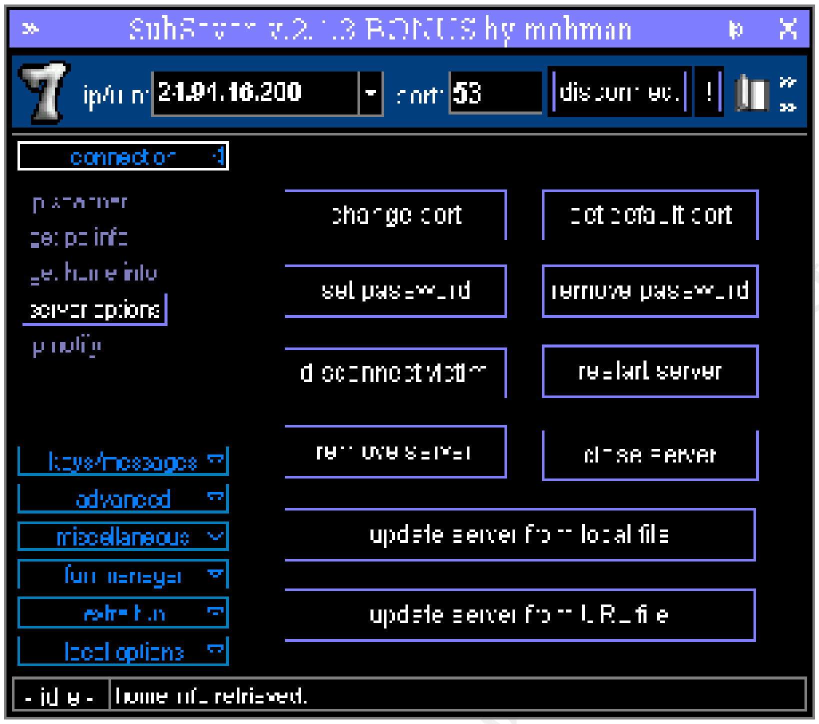


```
      address: not found
               not found
     business: not found
         city: not found
      company: not found
      country: not found
customer type: not found
       e-mail: not found
    real name: not found
        state: not found
    city code: not found
 country code: not found
  local phone: not found
     zip code: not found
```

- idle - home info retrieved.

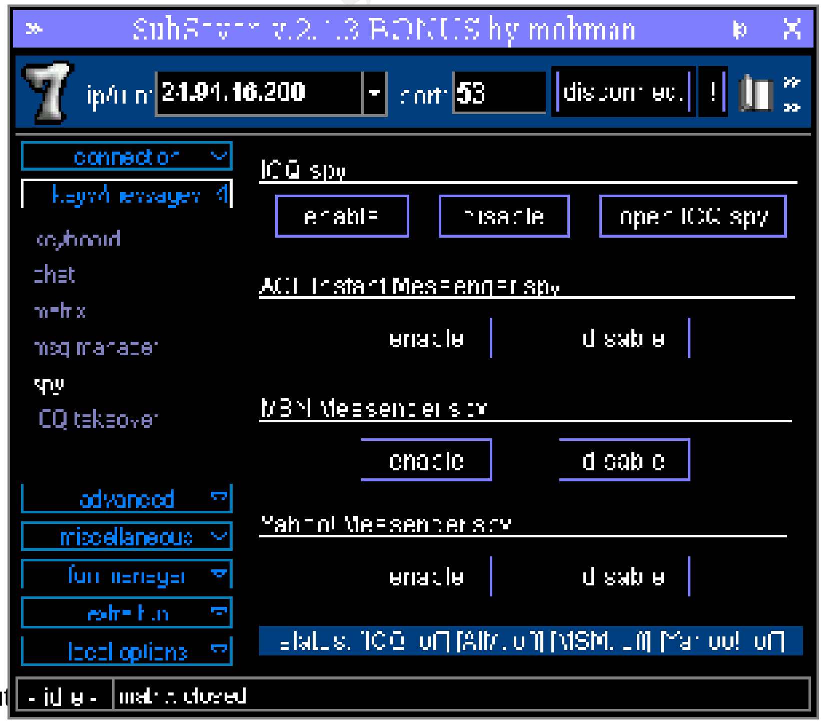
Server Information:

- You can change the communication port "on the fly". I managed to change port from port 80 to 53 and back. Another very "thoughtful" feature – scary if you are performing Intrusion Detection.



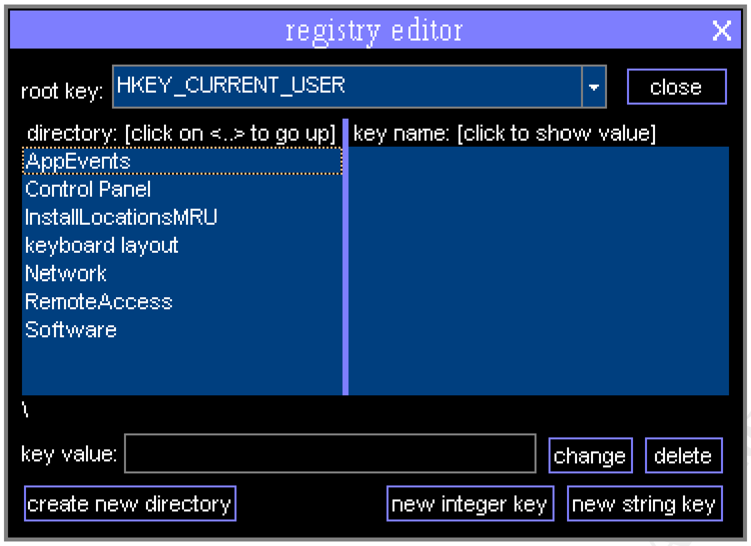
Spy on ICQ, IRQ

- You can spy on user ICQ and IRQ session.



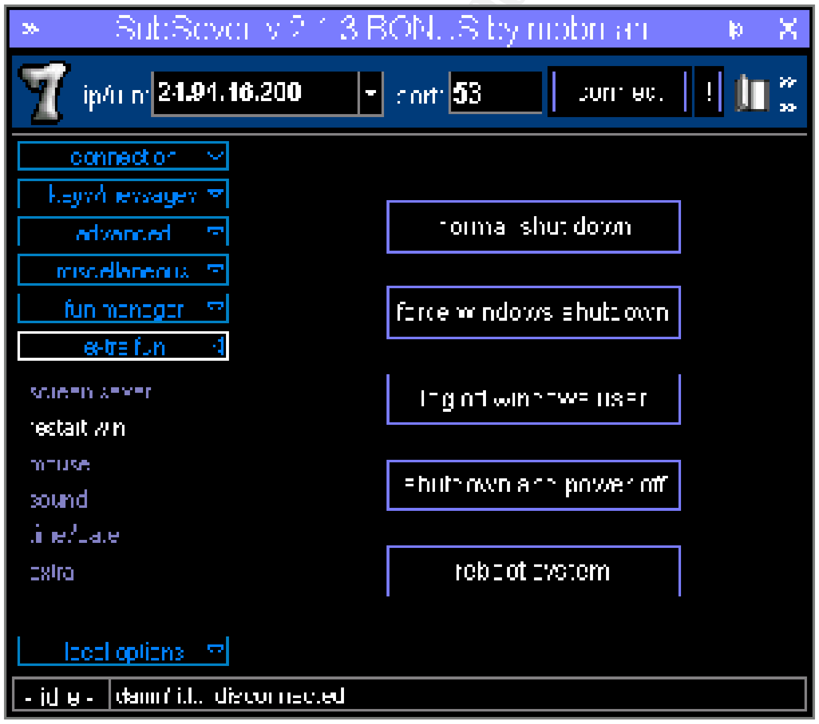
Registry Edit:

- This screen allows you to edit Windows registry. A hacker can do anything he wants on the poor victim.



Restarting Windows

- The attacker can restart windows, reboot the victim or force shutdown on the poor victim machine.



Conclusion:

There are quite a few more features on this sub-seven trojan, including screen flip, window color change, keyboard capture, mouse button flip and etc. It is quite a robust program.

The scary thing about this attack is while I spent hours playing with all the little features, snort only pick up the first log-in attempt. None of the rest of the traffic was pick up by Snort. I was running Snort 1.6.3 with 10102k.rules from whitehats.com.

Assignment 3: "Analyze This"

Scenario:
GIAC Enterprise, a dot.com startup that sells electronic fortune cookie sayings, has asked our organization to provide a bid to be their security services provider. About a month worth of data from a Snort system with fairly standard ruleset was provided. However, the snort log is not complete due to factors like power failure, disk full and other reasons. Our task is to analyze the data, look for signs of compromised systems or network problems and produce an analysis report. The goal is to demonstrate that our mastery of the subject material and analysis ability.

Analysis Report:

Hosts Report:

Host IP: My.Net.211.2

- 60 instances of *SUNRPC highport access!*

```
09/06-23:10:10.012419 [**] SUNRPC highport access! [**]
193.64.205.17:56880-> My.Net.211.2:32771
09/06-23:10:10.159763 [**] SUNRPC highport access! [**]
193.64.205.17:56880-> My.Net.211.2:32771
09/06-23:10:10.302667 [**] SUNRPC highport access! [**]
193.64.205.17:56880-> My.Net.211.2:32771
```

Looks like this server has been probed for port 32771 quite a bit. Port 32771 is SUN RPC high port. It is usually reserved for use inside a LAN. It could be one of the following 4 activities, but there are no corresponding detail log available for these alerts.

```
IDS26/nfs-showmount [TCP any -> 32771:] CAN-1999-0631
IDS429/portmap-listing-32771 [TCP any -> 32771] CAN-1999-
0632
IDS241/rpc.ttdbserv-solaris-kill [TCP any -> 32771:34000]
CVE-1999-0003
IDS242/rpc.ttdbserv-solaris-overflow [TCP any ->
32771:34000] CVE-1999-0003
```

Host IP: My.Net.179.80

- 1 instances of *Happy 99 Virus*

```
08/20-15:41:12.157972 [**] Happy 99 Virus [**]
24.2.2.66:58102-> My.Net.179.80:25
```

My.Net.179.80 seems to be a smtp server. Here we see a SMTP mail with Happy 99 Trojan Horse Virus signature. This server really needs to be examined in further detail, especially scanning for viruses. It demonstrate how a virus can infect the system. If My.Net.179.80 is a SMTP server running an out of date Virus Scan or no virus scan, then chances are the virus has slipped into some users PCs.

Following is some information on Happy 99 Virus:

- How does it propagate?
  Email or Usenet attachment email or Usenet attachment.
- Where does it reside?
  Modified WSOCK32.DLL Macro in Microsoft Word documents
- Who is it sent to?
  The recipients of the last message you sent out that are not in the LISTE.SKA file First 50 entries in each address
- Also known as: SKA, WSOCK32.SKA, SKA.EXE, I-Worm.Happy, PE_SKA, Trojan.Happy99, Win32/SKA, and Happy99.Worm.

Two possibly related CVE to keep an eye for:

CVE-2000-0277 Microsoft Excel 97 and 2000 does not warn the user when executing Excel Macro Language (XLM) macros in external text files, which could allow an attacker to execute a macro virus, aka the "XLM Text Macro" vulnerability.
CVE-2000-0478 In some cases, Norton Antivirus for Exchange (NavExchange) enters a "fail-open" state which allows viruses to pass through the server.

Host IP: My.Net.6.35

- 1 instances of *Happy 99 Virus*
- 2 instances of *SYN-FIN scan!*
- 9 instances of *Queso fingerprint*

```
08/16-14:36:46.954418 [**] Happy 99 Virus [**]
128.8.198.101:12805-> My.Net.6.35:25 09/09-15:01:09.888516
[**] Queso fingerprint [**] 216.15.191.130:56815->
My.Net.6.35:25 09/11-01:04:47.776874 [**] Queso fingerprint
[**] 216.15.191.130:34527-> My.Net.6.35:25 09/11-
01:04:47.776874 [**] Queso fingerprint [**]
216.15.191.130:34527-> My.Net.6.35:25 09/11-01:05:29.792635
[**] Queso fingerprint [**] 216.15.191.130:34527->
My.Net.6.35:25 09/11-01:05:29.792635 [**] Queso fingerprint
[**] 216.15.191.130:34527-> My.Net.6.35:25 09/11-
01:42:49.740995 [**] Queso fingerprint [**]
216.15.191.130:56906-> My.Net.6.35:25 09/11-01:42:49.740995
[**] Queso fingerprint [**] 216.15.191.130:56906->
```

```
My.Net.6.35:25 09/11-01:42:58.735557 [**] Queso fingerprint
[**] 216.15.191.130:56906-> My.Net.6.35:25 09/11-
01:42:58.735557 [**] Queso fingerprint [**]
216.15.191.130:56906-> My.Net.6.35:25 09/11-06:45:39.223977
[**] SYN-FIN scan! [**] 210.61.144.125:21-> My.Net.6.35:21
09/11-06:45:39.223977 [**] SYN-FIN scan! [**]
210.61.144.125:21-> My.Net.6.35:21
```

Host: My.Net.6.15

- 2 instances of *SMB Name Wildcard*
- 2 instances of *WinGate 1080 Attempt*
- 3 instances of *SYN-FIN scan!*
- 3 instances of *SUNRPC highport access!*
- 35 instances of *External RPC call*

Searching through Snort logs unveil additional interesting information:

```
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+
09/04-16:15:21.867041 MY.NET.222.110:1356 ->
24.18.91.196:5190
TCP TTL:126 TOS:0x0 ID:26570  DF
**SFR**U Seq: 0x89F   Ack: 0x40BA0001   Win: 0x5018
TCP Options => EOL EOL
9D 48                                             .H
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=
09/11-06:47:22.600679 210.61.144.125:21 -> MY.NET.26.15:21
TCP TTL:25 TOS:0x0 ID:39426
**SF**** Seq: 0x30CB0F57   Ack: 0x3C26C647   Win: 0x404
00 00 00 00 00 00                                 ......
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=
08/29-21:00:22.509513 210.205.94.1:111 -> MY.NET.6.15:111
TCP TTL:25 TOS:0x0 ID:39426
**SF**** Seq: 0x1AD3D9AF   Ack: 0x412B3791   Win: 0x404
00 00 00 00 00 00                                 ......
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+=+=
09/02-10:14:20.656157 210.101.101.110:23 -> MY.NET.6.15:23
TCP TTL:23 TOS:0x0 ID:39426
**SF**** Seq: 0x299D8B90   Ack: 0x57DD81F3   Win: 0x404
00 64 06 00 98 F9                                 .d....
```

These are syn/fin scans – information gathering phrase. The attacker seems to be quite patient, scanning a port once every hour or so. Anyhow, seems like 210.101.101.110 and 210.61.144.125 are actively targeting this system, probing for information and open ports. They are employing SYN/FIN scan which fools some older packet filter machine because it is not supposed to happen in real world situation.

Attacker IP: 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
```

Attacker IP: 210.61.144.125 :

```
210.61.144.0 - 210.61.144.255 netname: HINET8-144-TW
descr: Abnet Information Co., Ltd
descr: 6F. No.22-5
descr: Ning Hsia Rd Taipei, Taiwan country: TW admin-c: HWLL-TW tech-c:
HWLL-TW notify: hostmaster@twnic.net changed: hostmaster@twnic.net
971119 source: APNIC
```

Interestingly, these two IP addresses belong to traditional US friendly countries. However, considering that GIAC is an electronic fortune cookie company, they could be foreign competitors.

WinGate 1080 SOCKS:
Scans on port 1080 are usually looking for WinGate, a firewall-proxy for windows. It was really popular until a year or two ago. However, if we are running it here, we need to make sure this server is not vulnerable to the following CVE:

CVE-1999-0290 The WinGate telnet proxy allows remote attackers to cause a denial of service via a large number of connections to localhost.
CVE-1999-0291 The WinGate proxy is installed without a password, which allows remote attackers to redirect connections without authentication.
CVE-1999-0441 Remote attackers can perform a denial of service in WinGate machines using a buffer overflow in the Winsock Redirector Service.
CVE-1999-0494 Denial of service in WinGate proxy through a buffer overflow in POP3.
CAN-1999-0657 ** CANDIDATE (under review) ** WinGate is being used.

```
Snort Alert Logs:
08/18-06:10:13.466733 [**] SYN-FIN scan! [**]
18.116.0.75:111-> My.Net.6.15:111
08/18-06:10:13.484691 [**] External RPC call [**]
18.116.0.75:111-> My.Net.6.15:111
08/18-06:10:13.530943 [**] External RPC call [**]
18.116.0.75:1661-> My.Net.6.15:111
08/18-06:10:13.546205 [**] External RPC call [**]
18.116.0.75:1661-> My.Net.6.15:111
08/18-06:10:13.546328 [**] External RPC call [**]
18.116.0.75:693-> My.Net.6.15:1
08/18-12:13:28.981943 [**] SMB Name Wildcard [**]
4.17.88.66:137-> My.Net.6.15:137 08/18-12:13:30.463668 [**]
SMB Name Wildcard [**] 4.17.88.66:137-> My.Net.6.15:137
08/19-01:39:20.501009 [**] External RPC call [**]
141.223.124.31:2796-> My.Net.6.15:111 08/19-01:39:20.721751
[**] External RPC call [**] 141.223.124.31:2796-> My.Net
08/19-12:12:52.959446 [**] External RPC call [**]
209.160.238.215:782-> My.Net.6.15:111 09/02-00:28:03.467564
[**] SYN-FIN scan! [**] 210.101.101.110:111->
My.Net.6.15:111 09/02-00:28:06.989407 [**] External RPC
call [**] 210.101.101.110:861-> My.Net.6.15:111 09/02-
09:39:11.608534 [**] SUNRPC highport access! [**]
212.204.196.241:857-> My.Net.6.15:32771 09/02-
10:14:17.559327 [**] SYN-FIN scan! [**] 210.101.101.110:23-
> My.Net.6.15:23
09/10-03:15:34.249462 [**] External RPC call [**]
161.31.208.237:2223-> My.Net.6.15:111 09/11-18:44:32.987577
[**] WinGate 1080 Attempt [**] 168.187.26.157:3679->
My.Net.6.15:1080 09/11-18:44:32.987577 [**] WinGate 1080
Attempt [**] 168.187.26.157:3679-> My.Net.6.15:1080
```

The above traffic seems to be all information gathering – but need to be confirmed with full Snort log, which is not available. We're kind of caught between a rock and a hard place. Seems like we are always missing either Snort Logs or Snorts Alert. It does looks fishy. Combined with the intrusion system crash so often is attacker trying to cover up their trail. GIAC Cookie.com needs to upgrade to latest Snort ruleset ASAP.

Host IP: My.Net.202.202

- 1 instances of *Possible wu-ftpd exploit - GIAC000623*
- 2 instances of *site exec - Possible wu-ftpd exploit - GIAC000623*

```
09/08-05:59:01.961301
[**] site exec - Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:2362-> My.Net.202.202:21
```

```
09/08-05:59:04.101862
[**] site exec - Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:2362-> My.Net.202.202:21
09/08-05:59:04.191384
[**] Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:2362-> My.Net.202.202:21
```

Washington University ftp daemon (wu-ftpd) is a very popular unix ftp server shipped with many distributions of Linux. Wu-ftpd is vulnerable to a very serious remote attack in the SITE EXEC implementation.

While exploited in a manner similar to a buffer overflow, it is actually an input validation problem. Anonymous ftp is exploitable making it even more serious as attacks can come anonymously from anywhere on the Internet.

Following are some wu-ftpd related CVP:

CVE-1999-0075 PASV core dump in wu-ftpd daemon when attacker uses a QUOTE PASV command after specifying a username and password.
CVE-1999-0080 wu-ftp FTP server allows root access via "site exec" command.
CVE-1999-0081 wu-ftp allows files to be overwritten via the rnfr command.
CVE-1999-0368 Buffer overflows in wuarchive ftpd (wu-ftpd) and ProFTPD lead to remote root access, a.k.a. palmetto.
CVE-1999-0720 The pt_chown command in Linux allows local users to modify TTY terminal devices that belong to other users.
CVE-1999-0878 Buffer overflow in WU-FTPD and related FTP servers allows remote attackers to gain root privileges via MAPPING_CHDIR.
CVE-1999-0879 Buffer overflow in WU-FTPD and related FTP servers allows remote attackers to gain root privileges via macro variables in a message file.
CVE-1999-0880 Denial of service in WU-FTPD via the SITE NEWER command, which does not free memory properly.
CVE-1999-0955 Race condition in wu-ftpd and BSDI ftpd allows remote attackers gain root access via the SITE EXEC command.
CVE-1999-0997 wu-ftp with FTP conversion enabled allows an attacker to execute commands via a malformed file name that is interpreted as an argument to the program that does the conversion, e.g. tar or uncompress.
CAN-1999-0076 ** CANDIDATE (under review) ** Buffer overflow in wu-ftp from PASV command causes a core dump.
CAN-1999-0156 ** CANDIDATE (under review) ** wu-ftpd FTP daemon allows any user and password combination.
CAN-1999-0661 ** CANDIDATE (under review) ** A system is running a version of software that was replaced with a Trojan Horse at its distribution point, e.g. TCP Wrappers, wuftpd, etc.
CAN-1999-0911 ** CANDIDATE (under review) ** Buffer overflow in ProFTPD, wu-ftpd, and beroftpd allows remote attackers to gain root access via a series of MKD and CWD commands that create nested directories.
CAN-2000-0573 ** CANDIDATE (under review) ** The lreply function in wu-ftpd 2.6.0 and earlier does not properly cleanse an untrusted format string, which allows remote attackers to execute arbitrary commands via the SITE EXEC command.

Not that they are all applicable here - but it seems like this could be a prime suspect.
In addition, there are 8 SYN/FIN Scans (see example as follow) targeted at this address. Again, there are no corresponding Snort logs.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
08/28-09:55:10.288780 MY.NET.202.202:0 ->
128.61.68.140:1694
TCP TTL:126 TOS:0x0 ID:10013  DF
2*SF*PAU Seq: 0x1A2B005F   Ack: 0x31330343   Win: 0x5010
TCP Options => Opt 32 (32): 2020 2000 0402 0101 080A 0023
0000 0000 0000 0000 0000 0000 0000 0000 0000


=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
08/28-09:55:13.260265 MY.NET.202.202:1694 ->
128.61.68.140:6699
TCP TTL:126 TOS:0x0 ID:56350  DF
2*SF*PAU Seq: 0x5F   Ack: 0x3133034B   Win: 0x5010
```

Host: My.Net.210.2

- 1 instances of *SUNRPC highport access!*

09/08-16:34:54.280910 [**] SUNRPC highport access! [**]
205.188.4.42:5190-> My.Net.210.2:32771

Please refer to My.Net.211.2

Host IP: My.Net.150.24

- 1 instances of *site exec - Possible wu-ftpd exploit - GIAC000623*
- 1 instances of *Possible wu-ftpd exploit - GIAC000623*

09/08-05:25:41.092146
[**] site exec - Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:4640-> My.Net.150.24:21
09/08-05:25:41.167678
[**] Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:4640-> My.Net.150.24:21

Host IP: My.Net.99.104

- 1 instances of *site exec - Possible wu-ftpd exploit - GIAC000623*

```
09/08-04:53:17.038845
[**] site exec - Possible wu-ftpd exploit - GIAC000623 [**]
24.17.189.83:3446-> My.Net.99.104:21
```

Host IP: My.Net.97.181

- 1 instances of *WinGate 1080 Attempt*
- 5 instances of *TCP SMTP Source Port traffic*

```
08/17-00:06:16.011962 [**] TCP SMTP Source Port traffic
[**] 206.46.170.21:25-> My.Net.97.181:25
08/17-00:06:19.582072 [**] TCP SMTP Source Port traffic
[**] 206.46.170.21:25-> My.Net.97.181:25
08/17-00:06:20.458283 [**] TCP SMTP Source Port traffic
[**] 206.46.170.21:25-> My.Net.97.181:25
08/17-00:06:46.492860 [**] TCP SMTP Source Port traffic
[**] 206.46.170.21:25-> My.Net.97.181:25
08/17-00:06:46.619655 [**] TCP SMTP Source Port traffic
[**] 206.46.170.21:25-> My.Net.97.181:25
09/05-15:51:53.126652 [**] WinGate 1080 Attempt [**]
216.225.7.166:4757-> My.Net.97.181:1080
```

SMTP source port traffic is unusual. This looks like another crafted packet.

Host IP: My.Net.253.53

- 2 instances of *Watchlist 000222 NET-NCFC*
- 3 instances of *TCP SMTP Source Port traffic*
- 6 instances of *SMB Name Wildcard*

```
08/11-00:57:11.251110 [**] Watchlist 000222 NET-NCFC [**]
159.226.120.14:49107-> My.Net.253.53:113 08/11-
16:28:46.525879 [**] SMB Name Wildcard [**]
207.79.66.3:614-> My.Net.253.53:137 08/11-16:28:46.525941
[**] SMB Name Wildcard [**] 207.79.66.3:137->
My.Net.253.53:137 08/11-16:28:48.020537 [**] SMB Name
Wildcard [**] 207.79.66.3:137-> My.Net.253.53:137 08/11-
16:28:48.020675 [**] SMB Name Wildcard [**]
207.79.66.3:614-> My.Net.253.53:137 08/11-16:28:49.521266
[**] SMB Name Wildcard [**] 207.79.66.3:614->
My.Net.253.53:137 08/11-16:28:49.522863 [**] SMB Name
Wildcard [**] 207.79.66.3:137-> My.Net.253.53:137 09/05-
09:57:45.008361 [**] Watchlist 000222 NET-NCFC [**]
159.226.45.3:1156-> My.Net.253.53:113 09/10-15:36:32.348040
[**] TCP SMTP Source Port traffic [**] 156.40.66.2:25->
My.Net.253.53:757 09/10-16:23:54.694617 [**] TCP SMTP
```

```
Source Port traffic [**] 156.40.66.2:25-> My.Net.253.53:902
09/10-16:24:01.024055 [**] TCP SMTP Source Port traffic
[**] 156.40.66.2:25-> My.Net.253.53:902
```

Possible Compromised Host:

Host IP: My.Net.217.218

This host is sending out SYN/FIN scans and a sort of weird TCP flags combination packet out to another outside host. These combinations of flags do not occur naturally.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/08-01:16:35.034978 MY.NET.217.218:1095 ->
207.172.3.46:119
TCP TTL:126 TOS:0x0 ID:34870  DF
21SFRPAU Seq: 0x33D6C   Ack: 0x960219C9   Win: 0x5010
04 47 00 77 00 03 3D 6C 96 02 19 C9 00 FF 50 10
.G.w..=l......P.
22 38 6C 99 20 20 20 20 20 00                    "8l.     .
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/08-01:20:07.832604 MY.NET.217.218:1095 ->
207.172.3.46:119
TCP TTL:126 TOS:0x0 ID:13173  DF
21SFRPAU Seq: 0x340CC   Ack: 0x197FE   Win: 0x5010
22 38 3D 07 20 20 20 20 20 00                    "8=.     .
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/08-01:20:55.858680 MY.NET.217.218:0 -> 207.172.3.46:1095
TCP TTL:126 TOS:0x0 ID:5515  DF
*1SFR*AU Seq: 0x770003   Ack: 0x41FC98AC   Win: 0x5010
00 00 04 47 00 77 00 03 41 FC 98 AC 0C B7 50 10
...G.w..A.....P.
22 38 72 71 20 20 20 20 20 00                    "8rq     .
--
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
09/08-02:31:35.313799 MY.NET.217.218:6699 ->
128.118.215.123:1823
TCP TTL:126 TOS:0x0 ID:4697  DF
21SFRPA* Seq: 0x2B0056   Ack: 0x217B08B8   Win: 0x5010
21 7B 08 B8 22 DF 50 10 22 38 6A C8 20 20 20 20
!{..".P."8j.
20 00                                             .
--
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
09/08-02:32:05.728602 MY.NET.217.218:6699 ->
128.118.215.123:1823
TCP TTL:126 TOS:0x0 ID:5739  DF
21SFRPA* Seq: 0x550056   Ack: 0x217B0906   Win: 0x5010
1A 2B 07 1F 00 55 00 56 21 7B 09 06 0A DF 50 10
.+...U.V!{....P.
22 38 82 7A 20 20 20 20 20 00                    "8.z     .
--
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/08-13:11:34.738996 MY.NET.217.218:1081 ->
207.172.3.46:119
```

```
TCP TTL:126 TOS:0x0 ID:19078  DF
21SFRP*U Seq: 0x220003   Ack: 0x5653B6F7   Win: 0x5010
20 00                                                .
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
09/08-13:27:36.231673 MY.NET.217.218:1081 ->
207.172.3.46:119
TCP TTL:126 TOS:0x0 ID:45028  DF
**SFRPA* Seq: 0x500003   Ack: 0x66C3C0A3   Win: 0x5010
TCP Options => Opt 32 (32): 2020 2000 0402 1521 4A59 3839
0000 0000 0000 0000 0000 0000 0000 0000 0000 EOL EOL EOL
EOL EOL EOL EOL EOL
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

Observations:

Huge "Coorperative" SYN Scan:

On 8/15 to 9/2 huge SYN scans coming from a wide variety of IP addresses targeted the entire My.Net subnet. During that period of time, a total of 97398 SYN scans were detected coming from 46 different sources scanning 28627 Destination IP addresses on MY.NET.X.X. The vast majority of them came from:

195.114.226.41	42652Scans
35.10.82.111	25469Scans
212.141.100.97	19965Scans

This could either be a real loud scan – more than likely those three IP addresses are slaves to the real scanning host. It could also be a Denial of Service attack agaisnt the intrusion detection analyst – real loud noise generator to hide out the real attack. From all the logs and data, it seems to be more of the latter case (since the attacker might have scan MY.NET before August).

During the same period of time, there are 1347 udp port scans from 34 sources going to 44 targets.

Overall, there are tons of SYN scans – a total of over 118000 SYN scans from over 50 sources. Incredible. Syn Scan is not that stealthy which prompts me to think this is likely just to generate noise to cover some other attack.

Suspicion

I did a "cat AllLogs > grep 6669" (Napster) and I got tons of matches! Woohoo! However, I skim through the list and every one of those packets has SYN/FIN bit set.Well, almost. At least all those that I managed to observe (because there are so many matches). See what there are doing to me? They are "DOSing" myself. Something strange is going on – or something I am really missing (which happened more often than I like it to). It is really hard for me to pick out the real suspicious packet – since they are so many of them that match my search.

Conclusion:

Seems like we are always missing either Snort Logs or Snorts Alert. It does looks fishy. May be the reason the intrusion system crash so often is because attacker is trying to cover up his trail. GIAC Cookie need to upgrade to latest Snort ruleset ASAP.
May be the intrusion system crash so often is attacker trying to cover up their trail.
Most of the aforementioned hosts could possibly be compromised – depends on how good the perimeter and host defense is. Without additional information, we recommend examining those hosts in detail to ensure they are indeed "clean".

A lot of these scans seem very specific – looks like Korea & Israel are really targeting you. If we can verify that GIAC is indeed running these servers listening on those ports, then we are pretty sure this is active targeting and even beyond.

We definitely see a lot of very dubious traffic and we feel that chances are good something slips through the crack and some of your hosts are compromised. However, without additional knowledge of GIAC Cookie Inc Network architecture from firewall information to internal IP addresses to network security model, we feel it is hard to confirm our finding.

[My personal notes:
For the amount of data provided, I think we need more warning/warning on Assignment three. It could speed things up if we have some additional information like

- if the client are running firewall
- if the firewall is statefull (which take cares of "defragmented packet" exploit.
- whether their internal IP address is legal address
- If they do run Web server
- Servers that have incoming DNS traffic alert – is it running as a DNS server? i.e. IP address of DNS server, Web server, SMTP mail server etc. You know what I mean ☺

- More time! More toy! More powerful machine!
  I was overwhelmed by the amount of data – My linux server could not run snortsnarf on the combined portscans_file and snortlogs file. It ran out of memory! I took all the memory from my other home PCs and plug them all into one linux machine! Then I have to split the file up to three chunks and run snortsnarf. It makes it harder to correlate the traffic pattern.]

Assignment 4: Analysis Process:

In general, there are three types of data file:

- Snort Alerts (SnortA* -- 20 files)
- Snort Logs   (SOOS* -- 18 files)
- Snort Port-Scan Log (SnortS* -- 18 files)

1) First I combined similar logs into one file – i.e. all 20 alerts files into one huge alert file. (I thought this would really help my event correlation when I utilize SnortSnarf). To do this, I utilized "cat" command and ">" I/O redirection on UNIX.

2) In order to use SnortSnarf, "My.Net" will need to be replaced. In this case, I use VI substitute and regular expression command to substitute it to 'My.Net" with "254.254".

3) I downloaded SnortSnarf from http://www.silicondefense.com/snortsnarf/.
4) Then, the smart and convenient tool SnortSnarf take all these data and produce HTML files for better into html format for easier diagnostic and analysis. I did this to both Snort Alert file and Snort Port-Scan file.
5) I ran into problem where the combined Port-Scan is too large – my tiny linux machine ran out of memory when processing it. SO I split the file up to three smaller chunks.
6) The traffic is then analyzed by using SnortSnarf, by looking for common IP's and ports, etc.
7) When a suspicious alert is observed, I use grep command to find relevant information from the SOOS file (Snort Logs)
8) In addition, commonly know ports and exploits were search for in the combined Snort logs file by using grep command, again.
9) When I have any exploit that I haven't seen before or not familiar with, I research them on the following web-sites:
   - http://www.whitehats.com/

- http://www.sans.org/giac.htm
- http://www.cert.org/nav/alerts.html
- http://www.snort.org
- http://www.silicondefense.com

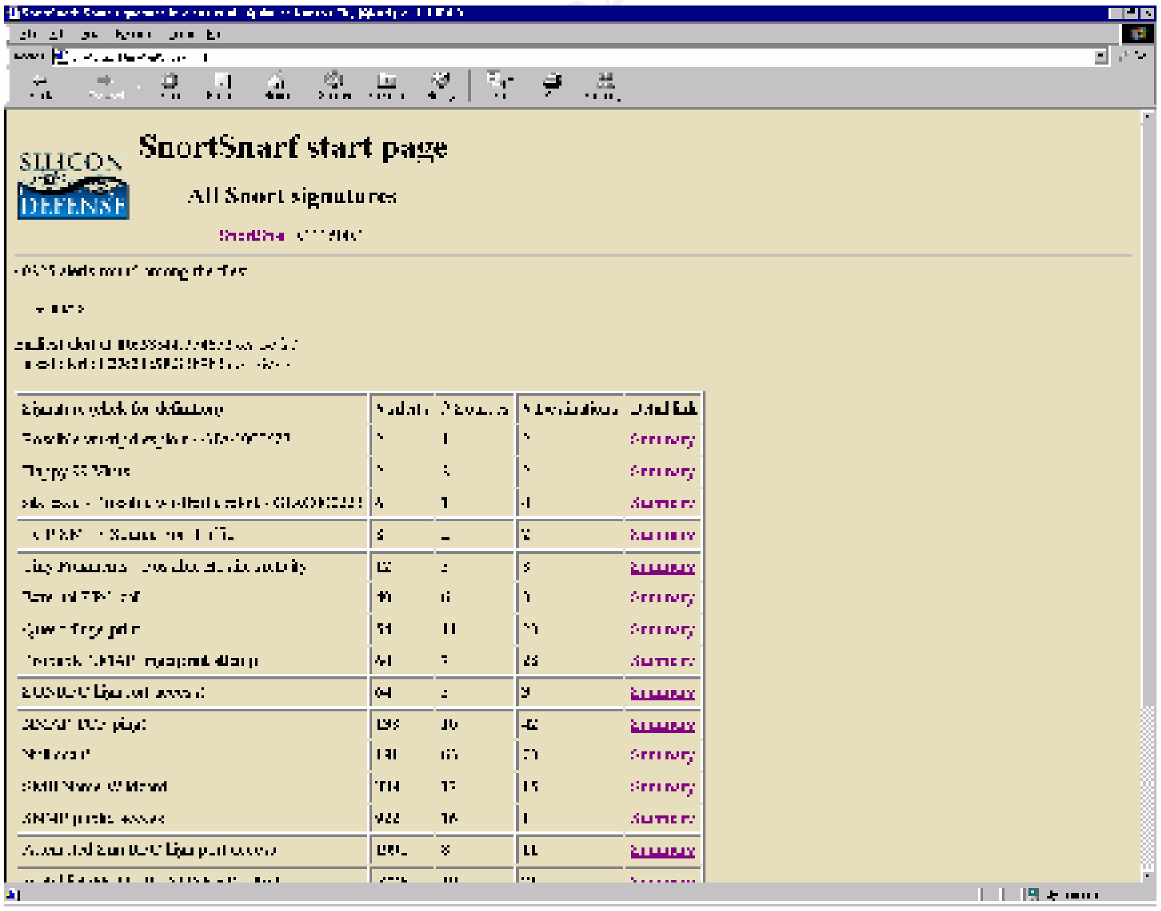
If I still can't find them, then it's a generic search on the Internet.

Of course, my SANS Security 2000 Monterey Text Books.

10) I analyze by correlating information from step 6,7, 8 and 9 above.

Screen Short from SnortSnarf:

Summary Page of Alerts



SnortSnarf start page

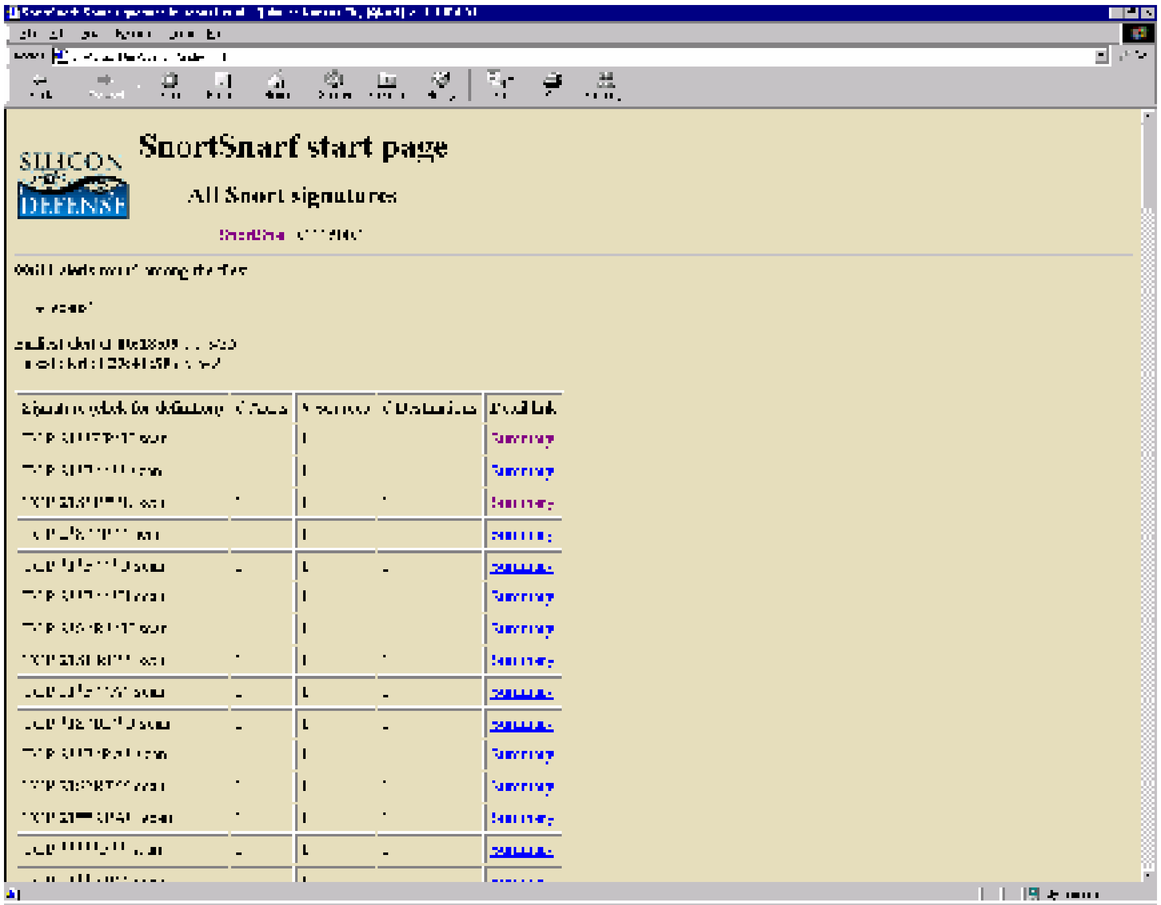
All Snort signatures

[illegible]

[illegible]

Signature (click for definition)	# Alerts	# Sources	# Destinations	Detail link
[illegible]	[illegible]	1	[illegible]	Summary
Happy 99 Virus	[illegible]	[illegible]	[illegible]	Summary
[illegible]	4	1	4	Summary
[illegible]	3	[illegible]	2	Summary
[illegible]	12	2	3	Summary
[illegible]	40	6	1	Summary
Queso fingerprint	51	11	[illegible]	Summary
[illegible]	61	7	23	Summary
[illegible]	64	2	3	Summary
[illegible]	139	10	42	Summary
Null scan!	181	65	[illegible]	Summary
SMB Name Wildcard	714	17	15	Summary
SNMP public access	922	16	1	Summary
[illegible]	[illegible]	8	11	Summary
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Summary Page for a Specific Signature: (Attempted Sun RPC high port access)



SnortSnarf start page

All Snort signatures

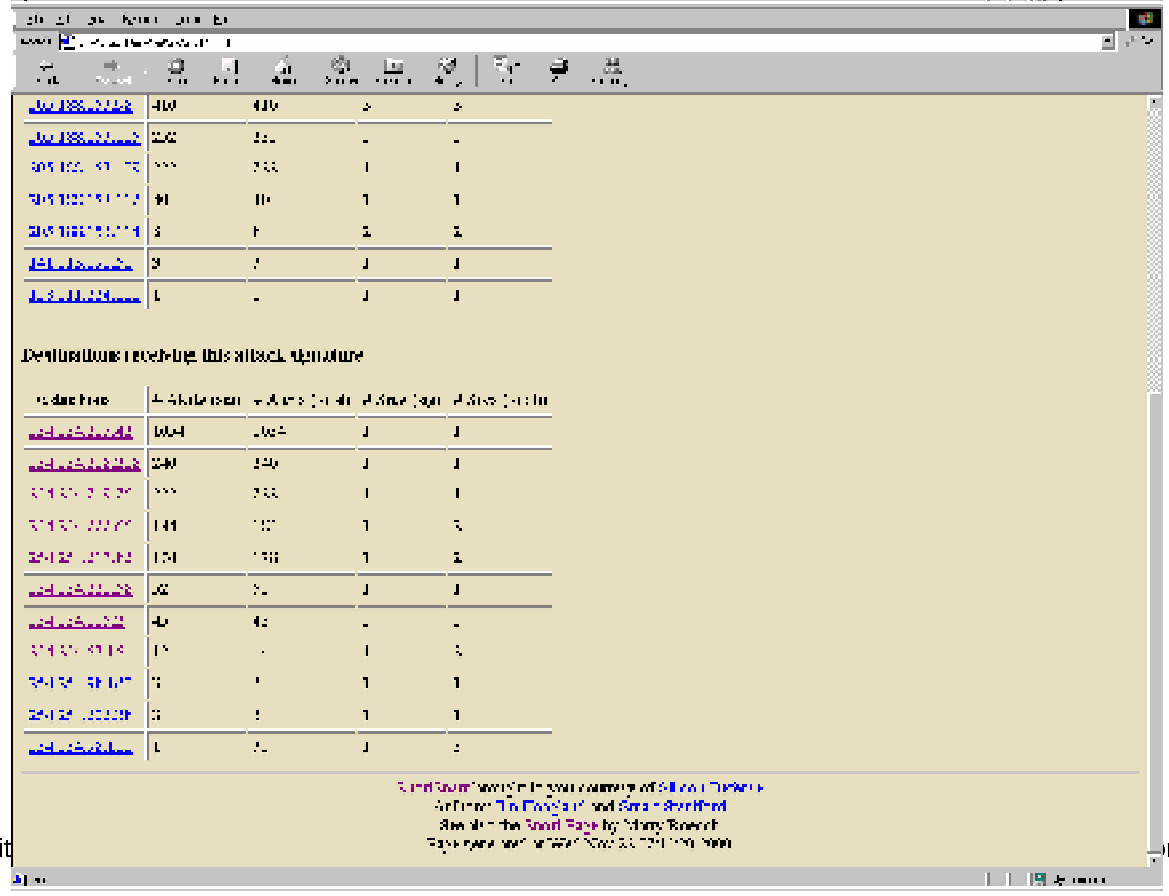
[illegible]

[illegible]

[illegible]

[illegible]

Signature (click for definition)	# Alerts	# Sources	# Destinations	Detail link
[illegible]		1		Summary
[illegible]		1		Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]		1		Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]		1		Summary
[illegible]		1		Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]		1		Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]	[illegible]	1	[illegible]	Summary
[illegible]	[illegible]	1	[illegible]	Summary



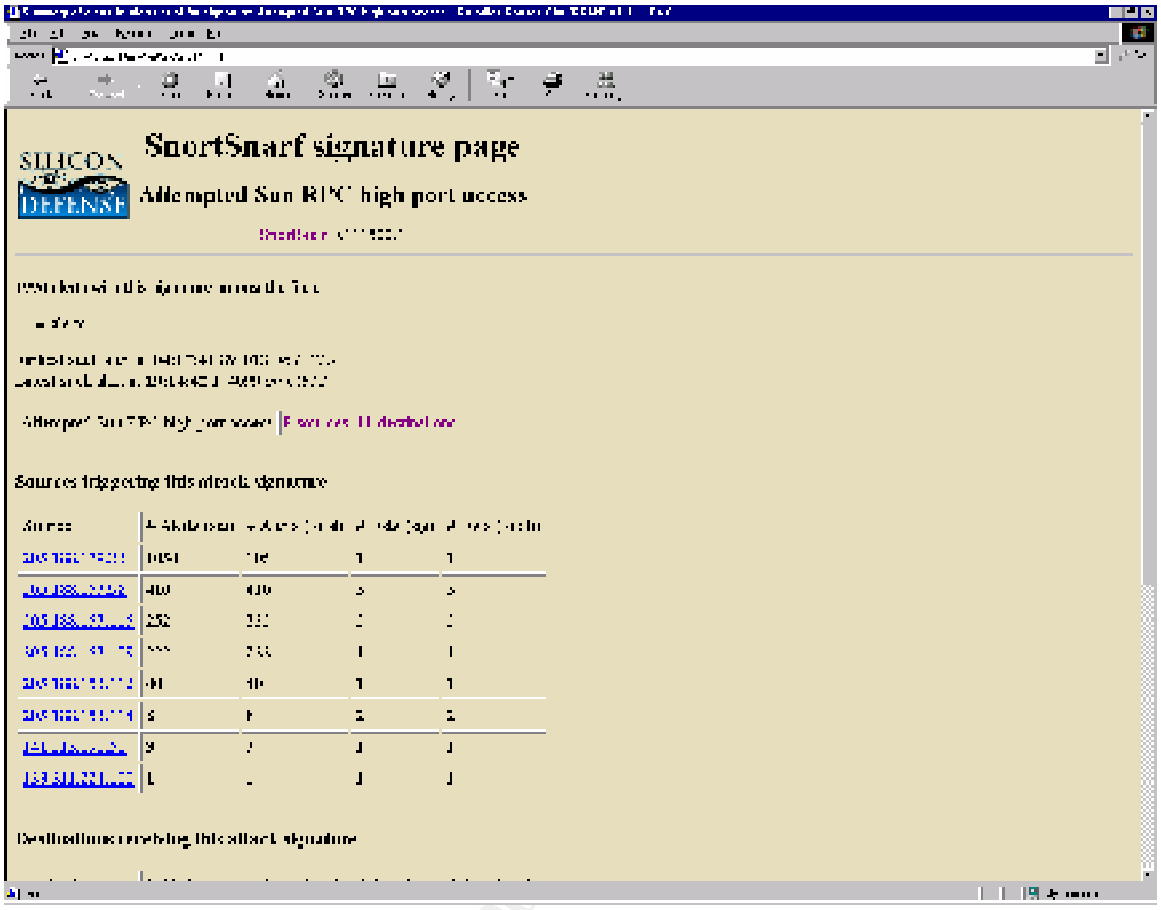
[illegible]	410	410	3	3
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	9	[illegible]	1	1
[illegible]	1	[illegible]	1	1

Destinations receiving this attack signature

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	1
[illegible]	240	240	1	1
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	1	[illegible]
[illegible]	[illegible]	[illegible]	1	[illegible]
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	[illegible]
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	1	1
[illegible]	1	[illegible]	1	[illegible]

[illegible]

Summary Page of Port-Scans Log:



SnortSnarf signature page

Attempted Sun RPC high port access

[illegible]

[illegible]

- [illegible]

[illegible]
[illegible]

Attempted Sun RPC high port access | [illegible]

Sources triggering this attack signature

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	1051	[illegible]	1	1
[illegible]	400	400	[illegible]	[illegible]
[illegible]	252	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	1	1
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Destinations receiving this attack signature