Global Information Assurance Certification Paper

GIAC INTRUSION DETECTION CURRICULUM
PRACTICAL ASSIGNMENT
Version 2.2.5

Mark Menke

Section 1. Network Detects

Detect #1

```
Oct 22 18:09:07 hop proftpd[8761]: refused connect from cc666259-
a.avnl1.nj.*.com
Oct 22 18:54:52 hop proftpd[8798]: refused connect from cc666259-
a.avnl1.nj.*.com
Oct 23 20:47:36 hop proftpd[9116]: refused connect from host10.*.com
Oct 27 16:55:43 hop proftpd[10841]: refused connect from
k114.dialup.*.com
Oct 28 22:09:01 hop proftpd[11064]: refused connect from 210.154.91.222
```

Source of trace:

Test Solaris 2.6 server on home DSL line. The test server was set up with proftpd with a "ALL:ALL" line in the hosts.deny file.

The format of the trace is

Date	Hostname of server	Daemon	Process ID	Action	Ip address or hostname of attacker
Oct 22 18:09:07	hop	proftpd	[8761]	refused connect from	cc666259-a.avnl1.nj.*.com

Detect was generated by:

Solaris 2.6 syslog, logged by tcpd (tcpwrappers) running under inetd.

Probability the source address was spoofed:

Unlikely, the attacker needs to use a valid address to gather the information about the server. Also, the three-way handshake is completed, and then the connection is torn down by tcpwrappers.

Description of Attack:

This is a connection on TCP port 21 (ftp-command) on a server that has no valid ftp daemon running. This system does not have any registered DNS names associated with it. It may be part of a reconnaissance sweep of the IP subnet.

Attack Mechanism:

The scan was probably done using an automated tool like nmap. Attackers generally look for open anonymous ftp sites to set up redistribution points for pirated software or root kits. More advanced attackers may upload malicious code to the ftp user directory, then use a buffer overflow to execute it. Since the connection was torn down post-connection, there is no way to accurately analyze the attacker's true motive.

Correlations:

Logs from other servers on the same subnet were also examined. Those with active ftp servers running showed that anonymous ftp connection were attempted and rejected from the same ip addresses. Systems without active ftp services running have logs displaying attempted connection that were refused.

Evidence of Active Targeting:

The server was not actively targeted. It was scanned along with several other machines on the same subnet, most of which do not have any registered DNS entries.

Severity:

Lethality of Attack	If the attack was successful, the attacker would not have root access, but the attacker did gain some information	2
Criticality of Server	Registered DNS server for test domain	4
Risk Rating		**2+4 = 6**
Network defenses	The machine is in the clear on the internet	0
Host defenses	The machine is fully patched and actively refuses the ftp connection	5
Defense Rating		**0+5 = 5**
Total Attack Severity		**6-5 = 1**

Severity of 1

Defensive Recommendation:

Implement some form of packet filtering access control list on the Internet Access router and/or install a stateful inspection firewall to control network traffic flow and to provide more extensive logging capabilities.

Multiple Choice Question

What are some of the possible reasons for the above actions to be logged?

a) User has improperly configured the mail client to connect to our system.
b) User is trying to gather information about systems on a specific subnet, that may be available for future access/services
c) User has a pre-configured FTP client and is now trying to access resources from a non-authorized location.
d) User is trying to execute a denial of service to the systems http service.

Answer:

B & C are valid answers, but B is more likely to be correct, since we have support evidence from other systems that the same activity was also reported during the same time frame and from the same source host. From the raw trace that information is not available, making answer C legitimate.

Detect #2

```
Oct 27 09:29:27 power fingerd[1664]: rejected @
Oct 27 09:29:40 power sshd[1685]: Bad protocol version identification 'USER saint  ' from
xxx.yyy.150.10
Oct 27 09:29:59 power telnetd[1686]: ttloop: peer died: EOF
Oct 27 09:30:16 power rshd[1689]: Connection from xxx.yyy.150.10 on illegal port
Oct 27 09:30:43 power ftpd[1684]: FTP session closed
Oct 27 09:30:44 power PAM_unix[1699]: check pass; user unknown
Oct 27 09:30:44 power PAM_unix[1699]: authentication failure; (uid=0) -> wank for system-auth
service
Oct 27 09:30:44 power login[1699]: FAILED LOGIN 1 FROM host10.snooper.com FOR wank,
Authentication failure
Oct 27 09:30:45 power PAM_unix[1701]: check pass; user unknown
Oct 27 09:30:45 power PAM_unix[1701]: authentication failure; (uid=0) -> rewt for system-auth
service
Oct 27 09:30:45 power login[1701]: FAILED LOGIN 1 FROM host10.snooper.com FOR rewt,
Authentication failure
Oct 27 09:31:13 power pam_rhosts_auth[1696]: denied to root@host10.snooper.com as root:
access not allowed
Oct 27 09:31:13 power in.rshd[1696]: rsh denied to root@host10.snooper.com as root:
Permission denied.
Oct 27 09:31:13 power in.rshd[1696]: rsh command was 'file /bin/sh'
Oct 27 09:31:16 power ftpd[1702]: FTP session closed
Oct 27 09:31:36 power pam_rhosts_auth[1703]: denied to root@host10.snooper.com as root:
access not allowed
Oct 27 09:31:46 power pam_rhosts_auth[1704]: denied to bin@host10.snooper.com as bin: access
not allowed
Oct 27 09:31:46 power in.rshd[1704]: rsh denied to bin@host10.snooper.com as bin: Permission
denied.
```

Source of Trace

A user's Redhat 7.0 workstation on home DSL line. The workstation was in the state of just being installed. There was very little customization, configuration hardening or updated patches applied to the operating system. .

The format of the trace is

Date	Hostname of server	Daemon	Process ID	Action
Oct 27 09:29:27	power	fingerd	[1664]	rejected @

Detect was generated by:

Redhat 7.0 syslog

Probability the source address was spoofed:

Unlikely, the attacker needs to use a valid address to gather the information about the server.

Description of Attack:

This appears to be saint scan. In less than two minutes, many connections to the server were made from a single IP address. The connections were all different in service type. The workstation was running services on many of the ports that were attacked. The scan starts with a connection to finger, trying to list users (finger @workstation). Then it progresses to trying ftp, rlogin, telnet, etc..

Attack Mechanism:

The scan was probably done using saint. The attacker let a complete and non-"stealthed" SAINT scan run against the system's IP address.

Correlations

The workstation was on a DSL connection running PPPOE. No other logs from other machines were available. If we look at the syslog information, we can see that some of the usernames the attacker tried to connect were root, rewt and SAINT. The amount of time between the different types of scans were so short, it would appear that the attacker was using some sort of automated script. Drawing upon the username saint and the speed of the attacks/probes, it would not be unlikely that the user is using SAINT to analyze our system for weaknesses.

Evidence of active Targeting:

The workstation was probably not targeted. There were no registered DNS names to this system, except for the reverse-record from the ISP. A SAINT scan is inelegant and a noisy system vulnerability scanner (www.saint.org). The attacker probably completed a ping-sweep (using a simple script or a compiled tool) and was simply scanning the systems that responded to the initial pings.

Severity:

Lethality of Attack	The attacker gained full OS and services offers	3
Criticality of Server	User workstation, no data	1
Risk Rating		**3+1 = 4**
Network defenses	The workstation is in the clear on the internet	0
Host defenses	The workstation is a default install, with no	3

	hardening or patching done	
Defense Rating		**0+3 = 3**
Total Attack Severity		**4-3 = 1**

Severity of 1

Defensive Recommendation:

The workstation needs to be built and fully configured before being exposed to an untrusted network. The access control lists should be applied to the Internet access router, to allow only specific ports open to the system. TCP wrappers should also be installed, to help network services become more secured, when exposed to the Internet.

Multiple Choice question

When a machine has been compromised, what are some common backdoor logins?

a) root
b) rewt
c) wank
d) All of the Above

Answer: d, those logins are commonly used in known rootkits.

Detect #3

Server used for this query: [whois.arin.net]
Cablevision Systems Corp (NETBLK-OOL-HNTNNY-UBR1-A)
111 New South Road Hicksville, NY 11801 US
Netname: OOL-HNTNNY-UBR1-A
Netblock: 24.188.0.0 - 24.188.5.255

Nov 16 14:07:08 24.188.1.147:4040 -> a.b.c.67:515 SYN ******S*
Nov 16 14:07:08 24.188.1.147:4053 -> a.b.c.80:515 SYN ******S*
Nov 16 14:07:08 24.188.1.147:4074 -> a.b.c.101:515 SYN ******S*
Nov 16 14:07:08 24.188.1.147:4079 -> a.b.c.106:515 SYN ******S*
Nov 16 14:07:08 24.188.1.147:4087 -> a.b.c.114:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4589 -> a.b.e.103:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4601 -> a.b.e.115:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4614 -> a.b.e.128:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4623 -> a.b.e.137:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4630 -> a.b.e.144:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4631 -> a.b.e.145:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4637 -> a.b.e.151:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4640 -> a.b.e.154:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4641 -> a.b.e.155:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4654 -> a.b.e.168:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4681 -> a.b.e.195:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4702 -> a.b.e.214:515 SYN ******S*
Nov 16 14:07:09 24.188.1.147:4705 -> a.b.e.217:515 SYN ******S*

Nov 16 14:07:11 hosta portsentry[244]: attackalert: Connect from host:
ool-18bc0193.dyn.optonline.net/24.188.1.147 to TCP port: 515

Nov 16 17:35:19 hosth inetd[19999]: refused connection from
ool-18bc0193.dyn.optonline.net, service ftpd (tcp)

Source of Trace

This trace is from the GIAC website.

The format of the trace is

Date/Time	IP and Port of the source	Direction of Communication	IP and Port of the Destination	Packet Type / Flag
Oct 27 09:29:27	24.188.1.147:4631	→	a.b.e.145:515	SYN

Detect was generated by:

A network IDS system, from the format of the trace, it is most likely a snort portscan report.

Probability the source address was spoofed:

Unlikely, the attacker needs to use a valid address to gather the information about the server.

Description of Attack:

This appears to be a network scan of the IP block for systems that have the LPD service running and accessible to Internet/external access. The attacker maybe trying to use an LPD exploit to find systems that are using an older or unpatched daemon.

Attack Mechanism:

If we look at the difference in time between SYN packets (5-13 packets per second), we can most likely assume that the user is using a simple port scanner or possibly nmap to scan the IP block for tcp port 515.

Correlations

The attacker is coming from an IP block that belongs to the Cablevision Systems Corporation. Without further analysis of that IP address, we believe the attacker has either rooted another system on a cable modem or is coming from the cable modem provider. The attacker maybe using a nmap or another fast port scanner to identify systems on the a.b.c.X subnet that has tcp port 515 open and available. Once the attacker has identified those systems, we can assume that s/he would proceed to use a discovered LPD exploit to attempt to get privileged access to the system.

Evidence of Active Targeting:

This was obviously a general scan of the network, since the IP address of the target was in an ascending order and the response port of the attacker's system

was also in a ascending order, we can assume that the program or script the attacker was running was simply incrementing the response port as it attempted to connect to hosts on the subnet.

Severity:

Lethality of Attack	Very low, just information gathering	1
Criticality of Server	No specific server targeted	1
Risk Rating		**1+1=2**
Network defenses	The subnet is available from the Internet	0
Host defenses	The servers appear to be running portsenty and have tcpwrappers enabled	4
Defense Rating		**0+4 = 4**
Total Attack Severity		**2-4 = -2**

Severity of 1

Defensive Recommendation:

Put access lists on the Internet access routers to block any ports other than the ones you want accessible from the Internet. Install TCP wrappers on the systems and configure each service to check the host.allow and hosts.deny files to ensure that users are coming from appropriate sources. Install and apply any patches to bring systems to a "current" supported level.

Multiple Choice question

When tcpwrappers blocks a connection to a TCP port, what information does the attacker gain?

a) nothing, the server is silent
b) the attacker learns that the port is open
c) the attacker learns that the host is alive
d) tcpwrappers only protects UDP ports

Answer c) the attacker learns that the host is alive. The three way handshake is completed with the host, then the connection is dropped by tcpwrappers.

Detect #4

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

Wendy Heffner (NETBLK-PBI-CUSTNET-4713)

1134 Walnut Street Berkeley, CA 94707 USA

Netname: PBI-CUSTNET-4713

Netblock: 216.101.170.128 - 216.101.170.135

Nov 19 18:44:09 hosta portsentry[2611]: attackalert: Connect from host: adsl-216-101-170-130.dsl.snfc21.pacbell.net/216.101.170.130 to TCP port: 1524
Nov 19 18:44:09 hosta portsentry[2611]: attackalert: Connect from host: adsl-216-101-170-130.dsl.snfc21.pacbell.net/216.101.170.130 to TCP port: 1524
Nov 19 18:44:09 hosta portsentry[2611]: attackalert: Connect from host: adsl-216-101-170-130.dsl.snfc21.pacbell.net/216.101.170.130 to TCP port: 1524

Source of Trace

This trace is from the GIAC website.

The format of the trace is

Date/Time	Name of Host (Destination of the packet)	Trace Trigger	Source IP of attacker (with reverse lookup)	Packet Type and Port Number
Oct 27 09:29:27	hosta	PortSentry	216.101.170.130	TCP Port 1524

Detect was generated by:

The portsentry process (process ID 2611).

Probability the source address was spoofed:

Unlikely, the attacker needs to use a valid address to gather the information about the server and it was an attempt to connect via a TCP port (requiring a 3way handshake).

Description of Attack:

The attacker is attempting to connect via port 1524 (Ingress Service). This service is commonly exploited in Solaris to gain root shell via bugs in the ttdbserverd and rpc.cmsd daemons.

Attack Mechanism:

If we look at the time between SYN connect attempts (3 per second), it is most likely that the attacker is using some sort of script to quickly identify IP addresses that have the Ingress service running.

Correlations

The attacker is coming from an IP address that has a reverse lookup for a Pacbell DSL account. The attacker made three attempts in the same second to try and connect to the Ingress port of HostA. It is believed that since there was no response, the attacker continued to scan other hosts on for service accessibility on other IP addresses.

Evidence of Active Targeting:

It appears that the attacker may have specifically targeted HostA, since there were no other reported traces to IP addresses on the same subnet (but since this was a reported trace, the reporting party may have removed any evidence to corroborate the scan was of the subnet and not just a single host).

Severity:

Lethality of Attack	Information gathering	1
Criticality of Server	Unknown Systems	?
Risk Rating		**Unknown, less than 6**
Network defenses	The subnet is available from the Internet	1
Host defenses	Good, the host is running portsentry, and the log is being watched by someone.	4
Defense Rating		**1+4 = 5**
Total Attack Severity		**Unknown, most likely negative**

Severity: most likely negative.

Defensive Recommendation:

Put access lists on the Internet access routers to block any ports other than the ones you want accessible from the Internet. If the Ingress or other services are not being used on the system, they should be commented out of the services, inetd.conf or the appropriate rc.d files.

Multiple Choice Question

What is a disadvantage of using automated response software like portsentry?

a) The attacker is stopped and logged
b) The server can only be used for portsentry
c) The server can be vulnerable to a DOS attack from spoofed packets
d) The server can not detect stealth scans

Answer: c Automated response software can remove routes to legitimate hosts if the attacker spoofs packets.

Section #2, Analyze an attack.

Source of the Attack:

http://rootshell.com/archive-j457nxiqi3gq59dv/200006/Xsh0k.c.html

This attack is denial of service attack against machines running X displays. The attacker attempts to stop or slow the display on the victim server. To be successful, the attacker needs to know that there is an X display running on the victim.

This attack was tested in a lab environment with two Redhat Linux workstations. The attack was done against the default display on TCP port 6000 on the workstation.

The attack was successful in slowing and stopping the X display on the victim. The machine became unresponsive to mouse and keyboard input. When the attack ceased, the machine recovered quickly.

This is a trace of three connections from stout (attacker) to the victim (10.11.1.108).

```
10:42:17.697697 eth0 < stout.1683 > 10.11.1.108.X: S 770637654:770637654(0) win
32120 <mss 1460,sackOK,timestamp 17021594 0,nop,wscale 0> (DF)
10:42:17.697789 eth0 > 10.11.1.108.X > stout.1683: S 694264540:694264540(0) ack
770637655 win 32120 <mss 1460,sackOK,timestamp 28966 17021594,nop,wscale 0> (DF)
10:42:17.697944 eth0 < stout.1683 > 10.11.1.108.X: . 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17021594 28966> (DF)
10:42:23.576574 eth0 < stout.1683 > 10.11.1.108.X: F 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17022182 28966> (DF)
10:42:23.576801 eth0 > 10.11.1.108.X > stout.1683: . 1:1(0) ack 2 win 32120
<nop,nop,timestamp 29554 17022182> (DF)

10:42:17.698233 eth0 < stout.1684 > 10.11.1.108.X: S 773634319:773634319(0) win
32120 <mss 1460,sackOK,timestamp 17021594 0,nop,wscale 0> (DF)
10:42:17.698268 eth0 > 10.11.1.108.X > stout.1684: S 697299842:697299842(0) ack
773634320 win 32120 <mss 1460,sackOK,timestamp 28966 17021594,nop,wscale 0> (DF)
10:42:17.698381 eth0 < stout.1684 > 10.11.1.108.X: . 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17021594 28966> (DF)
10:42:23.576604 eth0 < stout.1684 > 10.11.1.108.X: F 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17022182 28966> (DF)
10:42:23.576929 eth0 > 10.11.1.108.X > stout.1684: . 1:1(0) ack 2 win 32120
<nop,nop,timestamp

10:42:17.698469 eth0 < stout.1685 > 10.11.1.108.X: S 766970755:766970755(0) win
32120 <mss 1460,sackOK,timestamp 17021594 0,nop,wscale 0> (DF)
10:42:17.698501 eth0 > 10.11.1.108.X > stout.1685: S 697282632:697282632(0) ack
766970756 win 32120 <mss 1460,sackOK,timestamp 28966 17021594,nop,wscale 0> (DF)
10:42:17.698614 eth0 < stout.1685 > 10.11.1.108.X: . 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17021594 28966> (DF)
10:42:23.576626 eth0 < stout.1685 > 10.11.1.108.X: F 1:1(0) ack 1 win 32120
<nop,nop,timestamp 17022182 28966> (DF)
10:42:23.576978 eth0 > 10.11.1.108.X > stout.1685: . 1:1(0) ack 2 win 32120
<nop,nop,timestamp 29554 17022182> (DF)
```

The connections consisted of the standard TCP three-way handshake to the X server, and then the attacker left the connection open, which transferred no data. After approximately100 connections, the victim sent Fin packets in an attempt to remove recently made connections. The attacker used incrementing port numbers, that were followed to determine what the victim was doing.

Timeline

Connections with source port 1683-1791
The victim completed three-way handshake, and left these connections open.

Connections with source ports 1792-1820
The victim completed three-way handshake, and immediately sent a Fin packet

Connections with source port 1820-2045
The victim completed three-way handshake, and left the connection open.

When the attack program was stopped, the attacker sent Fin packets for all the open connections. The display recovered after a period of time on the victim, but the syslog showed that the kernel was experiencing TX buffer problems on the Ethernet card. If the victim was unable to keep up with the traffic, the outgoing Fin packets could be discarded, leaving the connection open.

Summary

The Xsh0k attack could be used in conjunction with a distributed DOS attack to cause problems for servers who run X displays. The exploit was successful in making the victim machine unusable.

To detect this attack, any IDS software with a port-scanning processor could be used. While there was no signature source port number or sequence number, the attack does rely on making many connections to the same port within a very limited amount of time. This is easy to detect on the network with snort or on the host with portsentry.

Section 3, Analyze This!

After reviewing the network data, the general examination format is outlined below:

Overall alerts	Breakdown of alerts, with metrics of quantity, source and destination.
Scans	An examination of various network scanning methods
Attacks	An examination of the attacks launched against your network
Problems	A listing of any hosts that appear to be compromised
Summary	A report of findings, and a graph of alerts vs. time.

Overall Alerts	# of Times	# of Source IP's	# of Dest IP's
Attempted Sun RPC high port access	1990	8	11
External RPC call	40	6	3
Happy 99 Virus	2	2	2
NMAP TCP ping!	138	10	42
Null scan!	181	63	73
Possible wu-ftpd exploit - GIAC000623	2	1	4
Probable NMAP fingerprint attempt	64	7	28
Queso fingerprint	54	11	23
site exec - Possible wu-ftpd exploit - GIAC000623	6	1	4
SMB Name Wildcard	338	17	15
SNMP public access	922	16	1
SUNRPC highport access!	64	5	3
SYN-FIN scan!	5457	6	3005
TCP SMTP Source Port traffic	8	2	2
Tiny Fragments - Possible Hostile Activity	12	5	8
Watchlist 000220 IL-ISDNNET-990517	5276	19	21
Watchlist 000222 NET-NCFC	19478	45	19
WinGate 1080 Attempt	6193	347	2156
Totals	**40225**	**553**	**4939**

- Note – total ip's do not add up due to individual hosts causing multiple types of alerts.

Alert categorization:

- Reconnaissance
  - NMAP TCP ping!
  - Null scan!
  - Probable NMAP fingerprint attempt
  - Queso fingerprint
  - SYN-FIN scan!

    - WinGate 1080 Attempt
- Attacks
    - External RPC call
    - Happy 99 Virus
    - Possible wu-ftpd exploit - GIAC000623
    - site exec - Possible wu-ftpd exploit - GIAC000623
    - SMB Name Wildcard
    - SUNRPC highport access!
    - Tiny Fragments - Possible Hostile Activity
- Other
    - SMNP public Access.
    - TCP SMTP Source Port traffic
    - Watchlist 000220 IL-ISDNNET-990517
    - Watchlist 000222 NET-NCFC

Scans -

Overall, the most reconnaissance was done by the ip address 24.180.134.156. This ip was doing many types of scans, with very little attempt at disguising their work. The machine is in the .home.com domain, and is most likely a compromised system. The owner of this machine must be notified immediately to ensure that no data is obtainable form the server. Another noisy scanner is coming from a university in Berlin. The server (130.149.41.70) resolves to bessy.physik.TU-Berlin.DE. This server uses most known scans attempting to find out what servers are alive, and what services and OS they are running. The administrators of this site need to be notified of this activity.

NMAP TCP Ping!

Sample: 08/11-12:57:26.064901 ** NMAP TCP ping! ** 205.128.11.157:80 -> MY.NET.1.8:53

This detect is most likely done by a snort rule checking for any packet with the ack sequence number set equal to 0. While it is theoretically possible for this to happen naturally in a network, it is most likely the signature of a scan using nmap. The scan can be used to detect what services are running on various boxes. The IP that was mostly involved in scanning your network was 24.180.134.156. This ip is registered to a cable modem user. Most likely this machine has been compromised and is being used to do a very quick and noisy scan.

The other hosts can be categorized two ways high risk and low risk. The high risks include those from home.com, home.net, uswest.net, and cinenet.net. The addresses are associated with somewhat anonymous access to the network. Some of the low risk addresses are netmeasure.lerc.nasa.gov and atl-lb.headhunter.net. These are most likely doing some kind or load balancing or network research.

The most troubling addresses in this series of alerts are 2.2.2.2, 213.8.52.189, and 202.187.24.3. The first one is indicative of a spoofed source address. While there is nothing to be gained directly from sending this packet, it could be used to test the responsiveness of the security team. The other two addresses are registered to foreign countries with no DNS reverse records. These two addresses need further investigation, as they are not shown in other alerts.

Null Scan

Sample: 08/11-20:18:48.417718 ** Null scan! ** 200.52.201.4:1409 -> MY.NET.217.222:6699

This series of alerts was most likely triggered by an NMAP null scan rule. This rule looks for a TCP packet with no flags set, a sequence number of 0, and an ack number of 0. A TCP packet with no flags set is most definitely a sign of an artificially crafted packet. As with the TCP ping, most of the scans are coming from dialup or home DSL/cable modems and universities.

NMAP fingerprint

Sample: 08/15-10:23:16.502216 ** Probable NMAP fingerprint attempt ** 216.181.188.154:1951 -> MY.NET.6.44:110

This series of alerts is triggered by a snort rule lookup for an illegal combination of SFPU. Again, most of the scans came from .edu and dialup access type networks, and one of the noisiest was 24.180.134.156. This was also noted in the portscan log.

Queso fingerprint

Sample: 08/15-15:30:15.258499 ** Queso fingerprint ** 216.123.63.13:4232 -> MY.NET.75.106:1488

The signature of a Queso OS identification scan is that a packet with the TCP syn bit and reserved bits 2 and 1 are set. This is used to identify what OS the remote machine is running by how it responds to the illegal bit combination. Again, the scans are mostly from dialup/home ISP's, and .edu domains.

One interesting detect in their series is from 213.228.1.13. This address belongs to an ISP in France, but the description for block lists it as being reserved for staff. If time allows, a call should be placed to the ISP asking if they have a reason for doing a queso scan.

WinGate 1080 Attempt

Sample: 08/11-00:46:56.976538 ** WinGate 1080 Attempt ** 208.240.218.220:4390 -> MY.NET.217.46:1080

WinGate is a network proxy server that usually runs on port 1080. This software has had several high profile bug announcements. The vulnerabilities in this software allow it to be used in two ways. The vulnerabilities are listed in the CVE database under the entries CVE-1999-290, CVE-1999-291, CVE-1999-441, and CVE-1999-494. If Wingate is run at your site, it should be monitored to make sure that the admins have correctly configured it. Since there are no alerts for "IDS366 - TELNET - WinGate-Active", any WinGate servers at your site are probably protected.

One interesting detect was from 168.187.26.157. This address corresponds to a dialup in Kuwait. The attacker in this case used the 1080 to attempt to disguise a portscan. With the 1080 rule making up approximately 15% of all detects on this network, some sites may be tempted to disable any scans destined for port 1080. If the border routers at the site are not configured with "no ip unreachables", the attacker would have a map of which hosts are active on the network. The ISP in Kuwait should be notified to investigate this scan.

SYN-FIN scan!

Sample: 08/17-09:39:01.814788 ** SYN-FIN scan! ** 130.149.41.70:1242 -> MY.NET.217.46:994

This is a very noticeable alert that is easy to detect. The packets are not any part of a normal TCP connection. There were 6 hosts doing scans during this time. Two are at universities, one is a cable modem connection, one is a polish web server, and two have no reverse. The two with no reverse belong to a Korean ISP (210.101.101.110), and another to Abnet Information in Taiwan. The records need to be correlated to make sure that no network misconfigurations have allowed any information to be gathered from these scans.

The attack from 210.101.101.110 is directed at only one host, MY.NET.6.15, looking at the telnet (TCP 23) and portmapper (TCP 111) ports. This is the sign that the attacker has already done some work on scanning, and knows that the machine is a unix machine. Immediately afterwards, the attacker attempts to connect to this host on port 111. This is further explained under the alert External RPC call.

Attacks -

Happy 99 Virus

Sample: 08/20-15:41:12.157972 ** Happy 99 Virus ** 24.2.2.66:58102 -> MY.NET.179.80:25

This alert is from a know virus signature. Both alerts were from incoming mail transfers using SMTP. If this site has a good virus detection program built into the email gateways, the detect should cause no alarm. If the detect was outgoing, or the condition of the virus removal software is unknown, the detect should be forwarded to those responsible for containing it. More information on the characteristics can be found at http://vil.nai.com/vil/virusChar.asp?virus_k=10144

External RPC call

Sample: 09/02-00:28:06.989407 ** External RPC call ** 210.101.101.110:861 -> MY.NET.6.15:111

This is an attack that is used to map the services that are available on a Unix server. The attacks came from various sources, including two universities, one ISP in Nevada, and three from separate sources in Korea. The external RPC calls are all to the portmapper services. This has been a common source of attacks, and is used to find out what services are offered by the host and on what port they are offered. These detects need to be tracked as they are sometimes followed by "Attempted Sun RPC high port access". This indicates that the server has responded to the rpcinfo request and has told the attacker what ports offer services. In this case, the server can be compromised without needing to do a "noisy" high port scan to find the services.

The owner of the server MY.NET.6.15 needs to be notified of the increased interest in the server. Also, the SIRT needs to be notified of the interaction with the server. Past records need to be reviewed to see if the server was previously compromised.

Attempted Sun RPC high port access

Sample: 09/02-00:14:19.544728 ** Attempted Sun RPC high port access ** 205.188.153.109:4000 -> MY.NET.219.26:32771

This alert is subject to high false alarms. Of the 1990 alerts, 1986 were from hosts at AOL.com. This is usually associated with instant messaging type traffic. The other four detects came from two universities, with the detect being part of a larger pattern of UDP portscans of hosts on your network.

Possible wu-ftpd exploit - GIAC000623 and site exec - Possible wu-ftpd exploit - GIAC000623

Sample: 09/08-04:53:17.038845 ** site exec - Possible wu-ftpd exploit - GIAC000623 ** 24.17.189.83:3446 -> MY.NET.99.104:21

These alerts were triggered from one ip address, 24.17.189.83. This address is in the home.com domain, and is most likely a compromised host. This host was used on 9/8/00

from 4-6:30am doing a very noisy port scan looking for ftp sites. The alerts for the site exec indicate that the attacker found the ftp sites that they were looking for and attempted to attack them at approx 5 to 5:30. The owners of the four servers (MY.NET.150.24 MY.NET.202.190 MY.NET.202.202 MY.NET.99.104) need to be notified that they may have a possible compromised machine. Also, until the server admins report back, all traffic to and from these servers needs to be logged and checked.

SMB Name Wildcard

Sample: 09/09-10:52:16.881113 ** SMB Name Wildcard ** 129.37.160.81:137 -> MY.NET.100.130:137

These alerts are triggered based on an attacker doing a Net bios wildcard request to the net bios port. This can be a cause for concern if the server actually responds, but it can be a false alarm if the attacker is identified as a remote user of your site.

Tiny Fragments - Possible Hostile Activity

Sample: 09/11-13:20:45.833385 ** Tiny Fragments - Possible Hostile Activity ** 24.68.58.96 -> MY.NET.217.82

This attack looks for packets that have been crafted to be fragmented unnaturally. Incorrectly fragmented packets have been used in the past for denial of service attacks like winnuke and teardrop. Also, these packets can be used to attempt to bypass packet filtering routers and firewalls.

The most interesting alerts for this come from the ssau.ru domain. This attacker sent a fragment to 3 hosts on your network. The packets could not be located in the packet captures, so this would be a good opportunity to add a further rule to do packet capture on this host and examine what the payload of these were.

Other –

SNMP public access

Sample: 09/11-13:20:45.833385 ** Tiny Fragments - Possible Hostile Activity ** 24.68.58.96 -> MY.NET.217.82

There were 922 detects flagged with the SNMP public access title. This is usually a high false alarm rule, as most equipment from the manufacturer contains an snmp public string. Attackers can use this information to gain a network map. In this case, the snmp information was being transferred entirely within the local networks. The administrator for these machines, MY.NET.97.154, MY.NET.97.206,MY.NET.97.217,MY.NET.97.244,MY.NET.97.246,MY.NET.98.109, MY.NET.98.114,MY.NET.98.148,MY.NET.98.159,MY.NET.98.171,MY.NET.98.172,

MY.NET.98.177,MY.NET.98.181,MY.NET.98.190,MY.NET.98.191, and MY.NET.98.201. The machines are sending packets to the server MY.NET.101.192 on the SNMP trap port UDP 161. The hosts are most likely left with their default configuration, and need to be changed to protect them.

TCP SMTP Source Port traffic

Sample: 08/17-00:06:16.011962 ** TCP SMTP Source Port traffic ** 206.46.170.21:25 -> MY.NET.97.181:25

Two hosts triggered this rule. The alert is most likely triggering on any host that initiates traffic from port 25. Both hosts are in known domains, gte.net and nih.gov, and both ip addresses are running SMTP services. These most likely are false alerts.

Watchlist 000220 IL-ISDNNET-990517 and

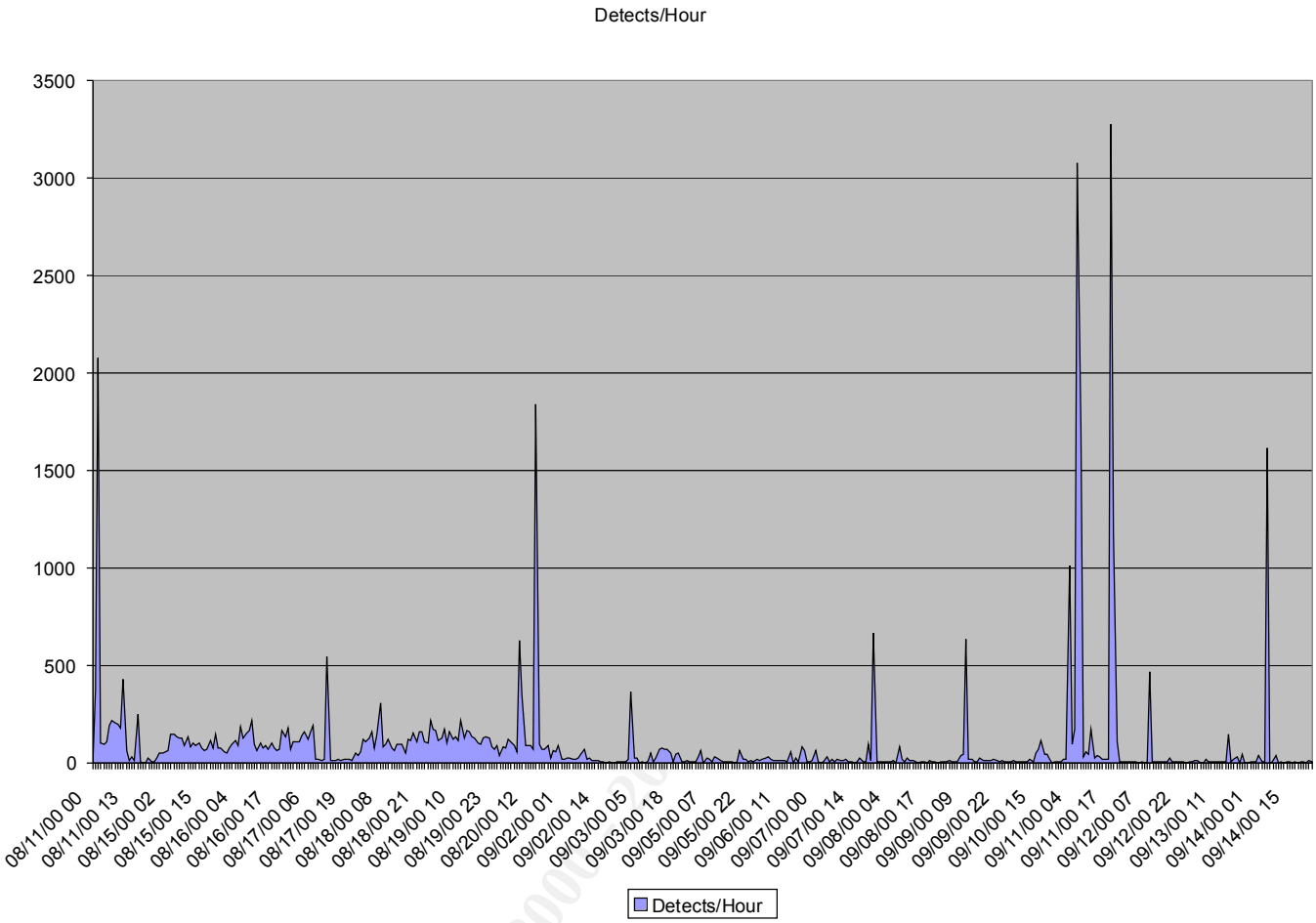
Watchlist 000222 NET-NCFC

I do not have access to the exact rules that generated these alerts, but it can be inferred from their geographical locations that there is a possibility of previous attacks from these netblocks, and that they are being monitored more closely for this reason. The netblocks were found to belong to Israel and China. The traffic captured by these rules appears to be mostly innocuous, although incoming telnet to MY.NET.6.7 is suspicious. I do not know of many legitimate networks that use telnet to communicate remotely over the Internet. Also, the address 212.179.58.2 appears to allow incoming telnet from your network. The acceptable use policy on your network needs to examined to see if this is allowed.

Summary –

Your network is a target for many types of network scans. With the amount of interest in servers on your site, maintaining good logs of this information should be a high priority. The most interesting highlights of your traffic include:

1. Possible compromise in progress – the FTP sites flagged by the wu-ftpd alert need to be examined soon. These machines were uncovered by scans, then immediately had exploit code run against them
2. The detection of a packet with source address 2.2.2.2 indicates that your ISP and border routers may not have complete anti-spoof access lists. The border security needs to be re-evaluated, as well as checked for any new entry points caused by rogue modems in your network.
3. SNMP public access. The machines responsible for these alerts need to be reconfigured to make sure that they no longer use public as their string. SNMP strings are usually treated as passwords, and leaving the default password is a very bad idea.



Section 4. Analysis Process

The amount of data to be processed is an indication that the analysis should be done on an on-going basis, rather than waiting a month for the data to collect. There were over 45,000 alerts to be analyzed, and the portscans detections and sample packet captures were significant.

To start the analysis, I downloaded the data and began by writing some awk scripts to categorize the alerts. The first step was to change the snort output format by changing [**] to **. This was done to make writing the awk scripts a little easier, as I was having trouble getting it to recognize [**] as a field separator. The alert data was then combined into a single file, and sorted by chronological order. That file was then separated into separate files for each alert type. Each alert was then analyzed by source and destination ip address. The portscan preprocessor data was used to correlate the attacks to see if the target was chosen based on previous knowledge, or was part of a larger overall attack.

For a sample attack like the wu-ftpd exploits, a file was extracted containing the entries

09/08-04:53:17.038845 ** site exec - Possible wu-ftpd exploit - GIAC000623 ** 2
4.17.189.83:3446 -> MY.NET.99.104:21
09/08-05:25:41.092146 ** site exec - Possible wu-ftpd exploit - GIAC000623 ** 2
4.17.189.83:4640 -> MY.NET.150.24:21

This file was then processed further using awk scripts to extract the source and destination ip addresses. These addresses were then sorted, counted, and looked up using www.arin.net and nslookup.

Let's look at using awk to separate out entries and sorting them. Lets assume that all of the alert files from the web site are put into one text file named all.sort, and that the entry [**] has been changed to **.

To list all unique source ip's, I would do

```
%awk -F "**" '{ print $3 }' all.sort | awk -F "->" '{ print
$1 }' | awk -F ":" '{ print $1 }' | sort -u
```

For this above example, this would have returned

```
24.17.189.83
```

To see the number of times this ip address appears as a source, the awk script could be used with grep and wc (word count)

```
%awk -F "**" '{ print $3 }' all.sort | awk -F "->" '{ print
$1 }' | awk -F ":" '{ print $1 }' | grep 24.17.189.83 | wc
-l
```

This run on the above sample would return 2.

By changing the three print statements in awk commands, I can control what information is gathered out of the files. I chose to create a series of files for each alert, containing the ip addresses, ports, and times. I then proceeded to actually start looking at the contents each file. In the background, I had a script running to lookup the source addresses.

```
#!/bin/bash

SRC=`sort -u all.src.ips`

for foo in $SRC; do nslookup $foo; done
```

This information was used to correlate where the attackers were located. For addresses that did not have a valid DNS reverse record, they were looked up in the arin records at http://www.arin.net/whois/index.html, http://www.ripe.net/cgi-bin/whois, and http://www.apnic.net/.