Global Information Assurance Certification Paper

GCIA Practical V3.1 Part 1 − State of IDS

Considerations for the collection, logging and management of IDS data.

Ever since Cliff Stoll's printer ribbon ran dry[1], the acquisition and maintenance of Intrusion Detection System data has been an issue. Analysts debate the merits of sensor placement, commercial or open source solutions and signature versus anomaly detection. Only rarely does the issue of IDS log collection, archival and management get raised. Yet, this is crucial to a successful implementation of an IDS solution.

This paper will look into five key elements an intrusion analyst must consider when deploying an IDS logging system:

How will the IDS data be used?
Collection of IDS data
Logging
Analysis and auditing
Maintenance of log data

The Australian Communications−Electronic Security Instruction 33 (ACSI 33) Handbook 13 identifies [2] many of the logging issues a security administrator will face, but misses a key driving factor− The purpose for collecting IDS data. The considerations are numerous and can be complex, however if an intrusion sensor's logs will simply be ignored, then there is no reason to collect the data in the first place.

How will the IDS data be used?

Answering this seemingly simple question will guide the intrusion analyst along the path of establishing a robust intrusion logging solution.

Before one can begin to design an IDS logging system, the overall security posture and policies of the defending network must be reviewed. These policies will hopefully provide an answer to how IDS data will be used. An ISP has very different goals for IDS data than a university might. Likewise, a large company with a big budget and full security staff will have different requirements than a smaller company or home hobbyist. Analyst experience levels and available time also play a part in this decision.

What about tracking internal abuses or industrial espionage? Sensors watching interior networks generally trigger on very different patterns than those

facing the Internet.

How about sharing the log information with other organizations (such as incidents.org)? This data must be sanitized prior to distribution.

Will the data be used to help prosecute attackers? If so, this matter becomes much more complicated. The data moves from simple logging into the rules of evidence realm. The generation of evidence must be provable and authentic[3]. It must also be preserved from alteration.

Collection of IDS data

In all but the smallest of infrastructures, best current practices call for a centralized logging facility. This is often established via an out−of−band management network connecting the sensors and log collector. However, there are occasions where this becomes impractical. For example, an isolated network or branch office reachable only via the Internet. In this case, it is imperative to encrypt the management traffic using a VPN or utility such a ssh or stunnel. Depending on how the log data will be used, there may be a requirement to implement verifications ensuring the data has not been altered in transit

Determining what to log from the sensors is also an early step in the collection process. Should the sensor perform rudimentary analysis and report only events and alarms? Or will full network packet captures be required? Generally some type of compromise will be implemented depending on the sensor type and placement.

It is important to consider other data sources as well. Firewall logs, router logs, SNMP traps and even application logs can provide valuable correlation to the intrusion analyst. The challenges lie in getting this data to the log collector and normalizing it if necessary. The IETF IDS Work Group is developing standards[4] that will help in this area.

Data volumes will have an impact on collection. High bandwidth pipes will compound the volume. Sensor placement often determines the types and volumes of data generated. Scott Sanchez, CISSP has produced an IDS Zone Theory diagram [5] illustrating a typical three−legged firewall with IDS sensors.

Perimeter sensors are often configured to collect a broad range of data, sometimes taxing the capacity of the management network or logging facility. DMZ sensors may be tuned to closely match screening firewalls, generating log data only when an attack breaches the firewall. Sensors placed on a trusted

LAN would presumably produce a relatively small amount of data.

Care must be exercised not to introduce a network denial of service with the log traffic. This is especially true for remote offices where the monitored and management networks are share the same pipe.

Logging

The primary consideration for logging is to ensure a common reference time. Log data will be of little value if multiple sensors and the collector clocks are not synchronized. NTP provides a facility to do this.

The centralized log host should be hardened to bastion standards. Because this host will contain sensitive data, it must be stringently protected against attack and abuse.

A back−end database to store collected logs can improve the scalability of the overall IDS solution. It will become important as more sensors and additional logging hosts are added. A database will also aid in the analysis, reporting and auditing of the data.

Analysis, reporting and auditing

In general, real time analysis and post event reporting based on IDS data is a minor piece of the overall log puzzle. It is important if active and automated response technologies are employed to remember to log both the response and the result. For example, an active response might be for the sensor to reset (by injecting a TCP RST) connections deemed inappropriate. A sensor could also modify router ACLs or firewall rules in response to an attack. Both the fact the sensor reacted to intrusive traffic and the result of that action must be logged.

A common post−incident item is the gathering of whois information and generating an email to responsible parties. This too should be logged and replies tracked. Large organizations with established trouble ticketing and alerting systems may wish to integrate the IDS log server in this process.

Maintenance of log data

Once the data has been processed, there are issues with retention and housekeeping. How long should IDS data be archived? What about historical and trending uses? Can the data be 'mined' to help predict future attacks?

CERT provides a standards paper with examples for log rotation and archival[6]. RFC3227 outlines "Guidelines for Evidence Collection and Archiving"[7]. These can be used as building blocks for establishing a data retention policy.

There are also legal issues to consider for data retention. For example, US Code Title 18 Section 2307(f) while not dictating a strict logging policy, does require the custodian to prevent data destruction for a period of 90 days (can be extended to 180 days) while under subpoena[8]. The European Union is rumored to also have laws requiring special handling of log data[9]. In all cases, it is imperative to consult a lawyer before proceeding.

Summary

This paper has examined several essential areas an IDS analyst must consider when implementing an intrusion logging system. The primary issue is to determine how the collected data will be used. The answer becomes the driving factor for each of the additional considerations.

Further Reading

Characteristics of a good IDS
URL: http://www.cerias.purdue.edu/coast/intrusion−detection/detection.html

Derek Atkins, et al.
Internet Security Professional Reference, Second Edition
New Riders, 1997 ISBN: 1−56205−706−X
Using System Logs to Discover Intruders pg. 167

Simpson Garfinkel and Gene Spafford
Practical Unix and Internet Security, Second Edition
O'Reilly & Associates, Inc., 1996 ISBN: 1−56592−148−8
Chapter 10: Auditing and Logging

James Mendelsohn Feb. 2001 Successful Deployment of an Intrusion Detection System URL: http://dcb.sun.com/practices/howtos/intrusion.jsp

Log Analysis Resources
URL: http://www.counterpane.com/log−analysis.html

References

[1] Clifford Stoll, The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage
Doubleday, 1989 ISBN: 0−385−24946−2 page 27

[2] Australian Communications−Electronic Security Instruction 33 (ACSI 33) − Handbook 13 URL: http://www.dsd.gov.au/infosec/acsi33/HB13.html

[3] Orin S. Kerr Computer Records and the Federal Rules of Evidence
URL: http://www.usdoj.gov/criminal/cybercrime/usamarch2001_4.htm

[4] IETF Intrusion Detection System Work Group
URL: http://www.ietf.org/html.charters/idwg−charter.html
Works in progress.

[5] Scott C. Sanchez, CISSP IDS Zone Theory Diagram
URL: http://infosec.gungadin.com/papers/scott_c_sanchez_cissp−ids−zone−theory−diagram.pdf
MIRROR URL: http://www.snort.org/docs/scott_c_sanchez_cissp−ids−zone−theory−diagram.pdf

[6] A practice from the CERT Security Improvement Modules
"Manage logging and other data collection mechanisms"
URL: http://www.cert.org/security−improvement/practices/p092.html

[7] RFC3227 Guidelines for Evidence Collection and Archiving
URL: http://www.ietf.org/rfc/rfc3227.txt

[8] US law 18 U.S.C. 2703(f) URL: http://www.cybercrime.gov/usc2703.htm

[9] Nick Farrell 31−05−2002 "EU snooping law shocks privacy groups"
URL: http://vnunet.com/News/1132301

GCIA Practical V3.1 Part 2 − Network Detects

Detect #1

This detect was selected to illustrate a different IDS source than a traditional intrusion detection system such as snort− in this case Linux ipchains syslog entries. It is also a perfect example of how not all suspicious traffic is malicious. For a breakdown of the fields recorded in an ipchains syslog entry, refer to Appendix 2.

1. Source of Trace

This detect is from my home DSL circuit protected by a Linux ipchains firewall. The home network consists of two Linux servers and several Linux/Windows95 dual−boot workstations NAT'd behind the firewall. The ipchains policy is to reject any TCP SYN packets and silently drop all ICMP echo−request packets. The traffic is logged to a remote syslog host.

2. Detect was generated by

Interest in this traffic was generated by a nightly report summarizing the ipchains syslog entries. A portion of that report is shown below. The report is generated by a simple bash and awk script (see top_attackers in Appendix 1) listing the top attackers by IP address, the destination ports that were targeted, along with a count for the port and a total for the source IP.

Report from Jun 2 04:03:33 thru Jun 7 23:58:00

Attacker	DST Port	Port Count	IP TOTAL
216.52.62.75	0/icmp	16	16
216.52.62.73	0/icmp	16	16
216.52.62.72	0/icmp	16	16
216.52.62.71	0/icmp	16	16
216.52.62.70	0/icmp	16	16
216.52.62.69	0/icmp	16	16
216.52.62.68	0/icmp	16	16
216.52.62.67	0/icmp	16	16
216.52.62.65	0/icmp	16	16

Typically, ICMP probes are nothing to worry about, but this report clearly showed a suspicious pattern− a consistent number of probes from a group of hosts in the same network block. My initial concern was whether my host was being used as part of a distributed denial of service attack against the 216.52.62.0/24 network.

Or perhaps those hosts were zombies and I was the victim. The total packet count was low, but perhaps the report was coincidentally produced at the leading edge of the attack. Spoofing an ICMP source address is trivial and given enough agents, a victim network could easily become saturated with echo−replies. This demanded further investigation.

A review of the syslog entries (see below) for these source addresses revealed the 144 probes were spread out over several days, beginning on June 5th at 09:31:21 continuing until June 7th at 4:03:03 (times are EST ntp sync'd). This lessened the likelihood of this being a DDOS attack, but was interesting enough to warrant additional research.

The packet lengths were always 84 bytes and the Time−To−Live (TTL) values varied only slightly. The small packet size further confirmed this probably wasn't a DoS attack (larger packets would more efficiently consume bandwidth). The TTL's suggested the source IPs were actually from the same network and not necessarily spoofed.

```
Jun  5 09:31:21 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.69:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=51 (#22)

Jun  5 09:32:06 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.69:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=51 (#22)

Jun  5 09:32:06 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.75:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=48 (#22)

Jun  5 09:32:50 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.75:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=48 (#22)

Jun  5 09:44:19 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.70:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=49 (#22)

Jun  5 09:44:24 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.70:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=49 (#22)

[snip ]

Jun  7 04:02:01 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.71:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=49 (#22)

Jun  7 04:02:29 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.67:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=51 (#22)

Jun  7 04:03:03 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.67:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=51 (#22)
```

3. Probability the source address was spoofed

The next step in the research process was to investigate the source addresses. I used the ‘host‘ command to determine DNS names for the addresses. Then I used ‘whois‘ to gather contact information for the domain.

```
$ host 216.52.62.75
75.62.52.216.IN-ADDR.ARPA domain name pointer performance-75.bos.pnap.net
```

Results for the other addresses (not shown) were similar. While it is unwise to completely trust information in a DNS entry, the ’bos’ in the name suggests Boston, MA and the ’performance−xx’ indicates these hosts might be part of a performance measurement for a global load balancing service.

```
$ whois pnap.net
[whois.networksolutions.com]
    [snip]
Registrant:
InterNAP Network Services (PNAP-DOM)
  Two Union Square, 601 Union St Suite 1000
  Seattle, WA 98101
  US
  Domain Name: PNAP.NET
  Administrative Contact, Technical Contact:
    Operations Center, InterNAP  (INO3)          noc@INTERNAP.COM
    InterNAP Network Services
    601 Union Street, Suite 1000
    Seattle, WA  98101
    US
    206.256.9500 206.748.0320

  Record expires on 21-Jun-2003.
  Record created on 20-Jun-1996.
  Database last updated on 29-Jul-2002 22:52:05 EDT.

  Domain servers in listed order:
  NS-A.PNAP.NET          64.94.123.4
  NS-C.PNAP.NET          64.95.61.4
  NS-B.PNAP.NET          64.94.123.36
  NS-D.PNAP.NET          64.95.61.36
```

Even with−out a three−way handshake to confirm the source of the connection, the addresses in question resolve and appear to be associated with a legitimate network performance metric service, so the spoof probability is nil. The TTL values also support this position.

4. Description of attack

Tribal Flood Network (TFN), Trinoo and Stacheldraht are examples of popular Denial of Service attacks that employ ICMP (among other types) packets. The CVE dictionary has assigned CAN−2000−0138 to identify the various DDoS attacks.

http://cve.mitre.org/cgi−bin/cvename.cgi?name=cve−2000−0138

The seminal authority on DDoS attacks is David Dettrich from the University of Washington . http://staff.washington.edu/dittrich/misc/ddos/

5. Attack mechanism

Distributed Denial of Service attacks generally work by building a network of compromised hosts (known as agents, handlers or zombies). A master controller communicates with the network over covert channels, such as hiding command and control information in ICMP echo request and reply packets.

When an attack is triggered, the agents/zombies will begin flooding the victim with packets. TCP SYNs and ICMP requests are often used as these will generally trigger a response (TCP reset, ICMP reply) further consuming bandwidth.

Another popular technique has the handlers spoofing the victims address and sending ICMP echo requests to a network broadcast address. This results in each active host on that network responding with echo replies to flood the victim.

It was initially suspected in this incident either 1) the home network was the victim (with the 215.15.62.0/24 hosts being compromised zombies) or 2) attackers were using the home system (by sending spoofed ICMP echo requests) to flood of the 215.16.62.0/24 network with ICMP echo replies. This second method would require many other innocent reflectors to be effective.

6. Correlations

A google search for ’performance’ and ’pnap.net’ yielded several correlations for this activity.

On Oct 8th, 2001 Richard Bejtlich posted to the SANS intrusions mailing list some UDP traceroute packets from performance.lax.pnap.net. UDP is an alternate method to ICMP to map the network performance between two hosts.

URL: http://www.incidents.org/archives/intrusions/msg01189.html

On Jan 23rd, 2002 Johannes B. Ullrich responded to Alexander Rayborn's query about pnap.net on the Dshield mailing list.

URL: http://www.dshield.org/pipermail/list/2002−January/002549.html

Both of these correlations support the findings of this investigation.

7. Evidence of active targeting

This traffic is an example of active targeting. The packets were destined for a specific address, not a network broadcast address. In addition, information on the pnap.net web site suggests these probes were triggered by a visit from the target network to an InterNAP customer's web site.

8. Severity: Severity should be calculated with the following formula severity = (criticality + lethality) − (system countermeasures + network countermeasures)

Criticality = 2. This is a home network, not a 24x7 e−commerce site.

Lethality = 2. The volume of packets was insufficient to flood the network.

System countermeasures = 5. The firewall is a relatively current version of the Linux kernel, is fully patched and offers no services to the Internet.

Network countermeasures = 5. ICMP echo requests are dropped by the firewall.

(2 + 2) − (5 + 5) = −6

As shown by the research, this traffic was not malicious. Suspicious and possibly an annoyance, but not malicious. A negative severity rating is appropriate in this case.

9. Defensive recommendation

A good defense against ICMP flood attacks is to rate limit or block echo−request packets at the network border. Care must be exercised not to block all ICMP packets, as this can break many Internet protocols.

Most operators of performance measurement systems will provide a method to

exclude addresses if their traffic from becomes an annoyance. Follow these instructions to be removed from the pnap.net monitoring list.
http://www.internap.com/measurements/readme.html

10. Multiple choice test question

Distributed Denial of Service attacks often employ SYN and ICMP packet floods. Why is it important to block ICMP−echo requests (type 8) at the network border?

a. The resulting ICMP redirects (type 5) will mean more bandwidth is lost to the attack.

b. ICMP type 0 packets carry the much of the original packet, resulting in additional bandwidth consumption.

c. Routers may drop the responses because they are low priority.

d. Time−to−Live (TTL) values in the responses can be used for further attacks.

Correct answer: b

Detect #2

This is an example of a common web server reconnaissance attack. It was chosen to illustrate the importance of proper host−based security measures, especially web server configuration.

[**] [1:1071:5] WEB−MISC .htpasswd access [**]
[Classification: Web Application Attack] [Priority: 1]
08/02−07:18:10.558038 0:40:F4:3B:8F:DA −> 0:A0:C9:D5:C5:58 type:0x800 len:0xF6
64.160.25.169:4110 −> MY.NET.231.206:80 TCP TTL:49 TOS:0x0 ID:42073 IpLen:20 DgmLen:232 DF
AP Seq: 0xED45B131 Ack: 0xE6542E48 Win: 0xF990 TcpLen: 20

1. Source of Trace

This trace was taken from a bastion Linux host on the Internet. The server is housed at an ISP co−location facility and does not have a screening firewall or router ACLs protecting it. In addition to providing virtual web hosting, the server offers ssh, IRC, ftp, mail and DNS services.

2. Detect was generated by

This detect was generated by the following snort rule (from web−misc.rules):

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-MISC
.htpasswd access"; flags:A+; content:".htpasswd"; nocase; classtype:web-application-
attack; sid:1071;  rev:5;)
```

This rule triggers on packets with an ACK flag and '.htpasswd' in the payload. To confirm the attack, it is necessary to review the packet data:

```
$ snort -dver 0802\@00-snort.log 'host 64.160.25.169 and port 4110'
Log directory = /var/log/snort
TCPDUMP file reading mode.
Reading network traffic from "0802@00-snort.log" file.
snaplen = 1514

        --== Initializing Snort ==--

        --== Initialization Complete ==--

-*> Snort! <*-
Version 1.8.7 (Build 128)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)
```

```
08/02-06:18:10.558038 0:40:F4:3B:8F:DA -> 0:A0:C9:D5:C5:58 type:0x800 len:0xF6
64.160.25.169:4110 -> MY.NET.231.206:80 TCP TTL:49 TOS:0x0 ID:42073 IpLen:20
DgmLen:232 DF
***AP*** Seq: 0xED45B131  Ack: 0xE6542E48  Win: 0xF990  TcpLen: 20
47 45 54 20 2F 2E 68 74 70 61 73 73 77 64 20 48  GET /.htpasswd H
54 54 50 2F 31 2E 30 0D 0A 41 63 63 65 70 74 3A  TTP/1.0..Accept:
20 69 6D 61 67 65 2F 67 69 66 2C 20 69 6D 61 67   image/gif, imag
65 2F 78 2D 78 62 69 74 6D 61 70 2C 20 69 6D 61  e/x-xbitmap, ima
67 65 2F 6A 70 65 67 2C 20 69 6D 61 67 65 2F 70  ge/jpeg, image/p
6A 70 65 67 2C 20 2A 2F 2A 0D 0A 55 73 65 72 2D  jpeg, */*..User-
41 67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 33  Agent: Mozilla/3
2E 30 20 28 63 6F 6D 70 61 74 69 62 6C 65 29 0D  .0 (compatible).
0A 48 6F 73 74 3A 20 77 77 77 2E xx xx xx xx xx  .Host: www.MY.NE
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx  T.really_long.UR
xx xx xx xx xx xx 2E 63 6F 6D 0D 0A 50 72 61 67  L......com..Prag
6D 61 3A 20 6E 6F 2D 63 61 63 68 65 0D 0A 0D 0A  ma: no-cache....

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

08/02-06:18:10.934962 0:A0:C9:D5:C5:58 -> 0:40:F4:3B:8F:DA type:0x800
len:0x257
MY.NET.231.206:80 -> 64.160.25.169:4110 TCP TTL:64 TOS:0x0 ID:5132 IpLen:20
DgmLen:585 DF
***AP*** Seq: 0xE6542E48  Ack: 0xED45B1F1  Win: 0x1920  TcpLen: 20
48 54 54 50 2F 31 2E 31 20 34 30 33 20 46 6F 72  HTTP/1.1 403 For
62 69 64 64 65 6E 0D 0A 44 61 74 65 3A 20 46 72  bidden..Date: Fr
69 2C 20 30 32 20 41 75 67 20 32 30 30 32 20 31  i, 02 Aug 2002 1
31 3A 31 38 3A 31 30 20 47 4D 54 0D 0A 53 65 72  1:18:10 GMT..Ser
76 65 72 3A 20 41 70 61 63 68 65 2F 31 2E 33 2E  ver: Apache/1.3.
32 36 20 28 55 6E 69 78 29 20 6D 6F 64 5F 70 65  26 (Unix) mod_pe
72 6C 2F 31 2E 32 36 20 6D 6F 64 5F 74 68 72 6F  rl/1.26 mod_thro
74 74 6C 65 2F 32 2E 31 31 20 50 48 50 2F 34 2E  ttle/2.11 PHP/4.
31 2E 30 20 46 72 6F 6E 74 50 61 67 65 2F 34 2E  1.0 FrontPage/4.
30 2E 34 2E 33 20 6D 6F 64 5F 73 73 6C 2F 32 2E  0.4.3 mod_ssl/2.
38 2E 39 20 4F 70 65 6E 53 53 4C 2F 30 2E 39 2E  8.9 OpenSSL/0.9.
36 62 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20  6b..Connection:
63 6C 6F 73 65 0D 0A 43 6F 6E 74 65 6E 74 2D 54  close..Content-T
79 70 65 3A 20 74 65 78 74 2F 68 74 6D 6C 3B 20  ype: text/html;
63 68 61 72 73 65 74 3D 69 73 6F 2D 38 38 35 39  charset=iso-8859
2D 31 0D 0A 0D 0A 3C 21 44 4F 43 54 59 50 45 20  -1....<!DOCTYPE
48 54 4D 4C 20 50 55 42 4C 49 43 20 22 2D 2F 2F  HTML PUBLIC "-//
49 45 54 46 2F 2F 44 54 44 20 48 54 4D 4C 20 32  IETF//DTD HTML 2
2E 30 2F 2F 45 4E 22 3E 0A 3C 48 54 4D 4C 3E 3C  .0//EN">.<HTML><
48 45 41 44 3E 0A 3C 54 49 54 4C 45 3E 34 30 33  HEAD>.<TITLE>403
20 46 6F 72 62 69 64 64 65 6E 3C 2F 54 49 54 4C   Forbidden</TITL
45 3E 0A 3C 2F 48 45 41 44 3E 3C 42 4F 44 59 3E  E>.</HEAD><BODY>
0A 3C 48 31 3E 46 6F 72 62 69 64 64 65 6E 3C 2F  .<H1>Forbidden</
48 31 3E 0A 59 6F 75 20 64 6F 6E 27 74 20 68 61  H1>.You don't ha
76 65 20 70 65 72 6D 69 73 73 69 6F 6E 20 74 6F  ve permission to
20 61 63 63 65 73 73 20 2F 2E 68 74 70 61 73 73   access /.htpass
77 64 0A 6F 6E 20 74 68 69 73 20 73 65 72 76 65  wd.on this serve
72 2E 3C 50 3E 0A 3C 48 52 3E 0A 3C 41 44 44 52  r.<P>.<HR>.<ADDR
45 53 53 3E 41 70 61 63 68 65 2F 31 2E 33 2E 32  ESS>Apache/1.3.2
36 20 53 65 72 76 65 72 20 61 74 20 77 77 77 2E  6 Server at www.
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx  MY.NET.really_lo
xx xx xx xx xx xx xx xx xx xx xx 2E 63 6F 6D 20  ng.URL......com
50 6F 72 74 20 38 30 3C 2F 41 44 44 52 45 53 53  Port 80</ADDRESS
3E 0A 3C 2F 42 4F 44 59 3E 3C 2F 48 54 4D 4C 3E  >.</BODY></HTML>
0A                                               .
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The packet data clearly shows this was an attack with the client issuing an HTTP GET request for the .htpasswd file. The server properly responded with a 403−Forbidden page.

3. Probability the source address was spoofed

The source address was not spoofed, as evidenced by the completion of the three−way handshake (ACK−PUSH set on the HTTP GET packet) and the 403−Forbidden response from the web server.

The host and whois commands indicate the attacker is on a DSL circuit provided by Pacific Bell.

```
$ host 64.160.25.169
169.25.160.64.in-addr.arpa. domain name pointer adsl-64-160-25-
169.dsl.lsan03.pacbell.net.
$ whois pacbell.net
[whois.networksolutions.com]

   [snip]

Registrant:
Pacific Bell Internet Services (PACBELL2-DOM)
  PO Box 940972
  Plano, TX 75075

  Domain Name: PACBELL.NET

  Administrative Contact:
    PBI DNS Administration  (PDA-ORG)       dnsadmin@PBI.NET
    Pacific Bell Internet
    940972
    Plano, TX 75075
    US
    800-463-8724
    Fax- - - - - 415-442-4999
  Technical Contact:
    Pacific Bell Internet NetCenter  (PB401-ORG)          trouble@PBI.NET
    P.O. Box 940972
    Plano, TX 75075
    US
    1-800-4NETPBI (463-8724)
    Fax- - - (415) 442-4999

  Record expires on 06-Apr-2010.
  Record created on 05-Apr-1996.
  Database last updated on 3-Aug-2002 17:37:32 EDT.

  Domain servers in listed order:

  NS1.PBI.NET             206.13.28.11
  NS2.PBI.NET             206.13.29.11
```

4. Description of attack

Misconfigured Apache web servers can reveal the contents of .htaccess and .htpasswd files. When implemented, these files are critical to the security of the web content and web clients should not be allowed to browse them.

The CVE dictionary lists several candidates for htpasswd problems, but none specific to the misconfiguration of Apache allowing remote access.

http://www.cve.mitre.org/cgi−bin/cvekey.cgi?keyword=htpasswd

The Apache web site offers this security tip about .htpasswd files:

http://httpd.apache.org/docs−2.0/misc/security_tips.html#watchyourlogs

5. Attack mechanism

By obtaining the user IDs and passwords contained in the .htpasswd file, a malicious user can view protected content on a web server. It may also be possible to misuse the information for further attacks. For example a web user ID and password might also match the operating system ID and password.

6. Correlations

As of August 3rd, 2002, dshield.org did not have any reports against this source address. A google search for the source IP address also came up empty.

Lenny Zeltser reported a web vulnerability scanner that includes a request for .htpasswd files as part of his GCIA practical:

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

7. Evidence of active targeting

This portion of a SnortSnarf report on the source IP shows evidence of active targeting. The attacker is scanning for various web vulnerabilities on this host. Although there are many IP addresses on this host, the attacker has selected only one to target. In addition, all the alerts were generated within a single second suggesting this was a scripted attack.

```
18 such alerts found using input module SnortFileInput, with sources:

* /var/log/snort/alert

Earliest: 07:18:09.458731 on 08/02/2002
Latest: 07:18:10.717006 on 08/02/2002

9 different signatures are present for 64.160.25.169 as a source

    * 1 instances of WEB-CGI test-cgi access
    * 1 instances of WEB-CGI nph-test-cgi access
    * 1 instances of WEB-CGI cart32.exe access
    * 2 instances of WEB-MISC .htpasswd access
    * 2 instances of WEB-CGI finger access
    * 2 instances of WEB-MISC .htaccess access
    * 2 instances of WEB-MISC http directory traversal
    * 3 instances of WEB-CGI Web Shopper shopper.cgi access
    * 4 instances of WEB-CGI search.cgi access
```

8. Severity: Severity should be calculated with the following formula severity = (criticality + lethality) − (system countermeasures + network countermeasures)

Criticality = 5. This is a commercial virtual server and the customers are relying on the the hosting company to provide security expertise and server configuration.

Lethality = 5. This is an attempt to obtain a password for access to protected resources.

System countermeasures = 5. The Apache web server is properly configured to block web access to the .htpasswd file.

Network countermeasures = 0. This is a bastion host without additional network protections.

(5 + 5) − (5 + 0) = 5

Although the attack failed, this is still a severe incident and should be reported to the source address ISP.

9. Defensive recommendation

This Apace web server is properly configured − it includes the following section in the httpd.conf file.

```
<Files ~ "^\.ht">
 Order allow,deny
 Deny from all
</Files>
```

This mechanism denies access to both .htpasswd and .htaccess files.

This IP address should be placed on a watch list because the recon scanning could be a precursor to further attacks.

10. Multiple choice test question

When an NIDS (such as Snort) uses signature based pattern matching, why is it important to verify an alert by reviewing the triggering packet?

a. The attacker may have used NIDS avoidance techniques.
b. Pattern matching is inferior to statistical based NIDS.
c. The rule might be poorly written, generating some false positives.
d. The ACK flag is often set in the course of normal TCP/IP traffic.

Correct answer: c

Detect #3

This detect was selected for review to demonstrate how an analyst must deal with incomplete data and situations where he or she has no knowledge of the target network. In other words, as Stephen Northcutt is fond of saying "Gee, I wish we had the hex..."

1. Source of Trace

On 16 Apr 2002, Mike Maxwell, a Systems Manager at Green Mountain Access requested assistance on the intrusions@incidents.org mailing list with the following tcpdump data. There was no reply from the list membership. A copy of the original posting is available at:

http://www.incidents.org/archives/intrusions/msg06467.html

```
12:14:24.555315 195.33.98.115.50831 > 1.2.3.31.domain: S
1705035803:1705035827(24) win 2048
12:14:24.555418 1.2.3.31.domain > 195.33.98.115.50831: S
1361705237:1361705237(0) ack 1705035804 win 5840 <mss 1460> (DF)
12:14:24.659172 195.33.98.115.50831 > 1.2.3.31.domain: R
1705035804:1705035804(0) win 0
12:14:24.659316 195.33.98.115.50832 > 1.2.3.31.domain: S
234491061:234491085(24) win 2048
12:14:24.659371 1.2.3.31.domain > 195.33.98.115.50832: S
1360635328:1360635328(0) ack 234491062 win 5840 <mss 1460> (DF)
12:14:24.761565 195.33.98.115.50832 > 1.2.3.31.domain: R
234491062:234491062(0) win 0
12:14:24.761703 195.33.98.115.50833 > 1.2.3.31.domain: S
893780373:893780397(24) win 2048
12:14:24.761753 1.2.3.31.domain > 195.33.98.115.50833: S
1366686861:1366686861(0) ack 893780374 win 5840 <mss 1460> (DF)
12:14:24.863707 195.33.98.115.50833 > 1.2.3.31.domain: R
893780374:893780374(0) win 0
12:54:24.719808 195.33.98.115.51146 > 1.2.3.31.domain: S
2650795914:2650795938(24) win 2048
12:54:24.719851 1.2.3.31.domain > 195.33.98.115.51146: S
3894676848:3894676848(0) ack 2650795915 win 5840 <mss 1460> (DF)
12:54:24.823526 195.33.98.115.51146 > 1.2.3.31.domain: R
2650795915:2650795915(0) win 0
12:54:24.823667 195.33.98.115.51147 > 1.2.3.31.domain: S
3306436458:3306436482(24) win 2048
12:54:24.823713 1.2.3.31.domain > 195.33.98.115.51147: S
3892431384:3892431384(0) ack 3306436459 win 5840 <mss 1460> (DF)
12:54:24.929220 195.33.98.115.51147 > 1.2.3.31.domain: R
3306436459:3306436459(0) win 0
12:54:24.929400 195.33.98.115.51148 > 1.2.3.31.domain: S
```

```
1834245808:1834245832(24) win 2048
12:54:24.929452 1.2.3.31.domain > 195.33.98.115.51148: S
3900773009:3900773009(0) ack 1834245809 win 5840 <mss 1460> (DF)
12:54:25.031556 195.33.98.115.51148 > 1.2.3.31.domain: R
1834245809:1834245809(0) win 0
13:34:26.258797 195.33.98.115.51414 > 1.2.3.31.domain: S
2109392221:2109392245(24) win 2048
13:34:26.258886 1.2.3.31.domain > 195.33.98.115.51414: S
2132968115:2132968115(0) ack 2109392222 win 5840 <mss 1460> (DF)
13:34:26.363952 195.33.98.115.51414 > 1.2.3.31.domain: R
2109392222:2109392222(0) win 0
13:34:26.364153 195.33.98.115.51416 > 1.2.3.31.domain: S
1300960383:1300960407(24) win 2048
13:34:26.364202 1.2.3.31.domain > 195.33.98.115.51416: S
2144739002:2144739002(0) ack 1300960384 win 5840 <mss 1460> (DF)
13:34:26.469338 195.33.98.115.51416 > 1.2.3.31.domain: R
1300960384:1300960384(0) win 0
13:34:26.469477 195.33.98.115.51418 > 1.2.3.31.domain: S
497463909:497463933(24) win 2048
13:34:26.469525 1.2.3.31.domain > 195.33.98.115.51418: S
2139545531:2139545531(0) ack 497463910 win 5840 <mss 1460> (DF)
13:34:26.576441 195.33.98.115.51418 > 1.2.3.31.domain: R
497463910:497463910(0) win 0
```

2. Detect was generated by

The data was produced from tcpdump. A field breakdown of tcpdump output is shown in Appendix 3. Note that Mr. Maxwell used 1.2.3.31, rather than MY.NET.xx.xx notation for the target address. It is unknown if the target actually provides DNS services, but since it is responding to requests on port 53, we shall assume is it a name server. (A whois of the GMA records reveals the last octet of the secondary name server is indeed .31).

3. Probability the source address was spoofed

The three−way hand−shake is never completed, but it is unlikely the source address is spoofed.

The source address does not have a reverse DNS entry. A traceroute (months after the original posting) shows the host is active:

```
$ traceroute 195.33.98.115
traceroute to 195.33.98.115 (195.33.98.115), 30 hops max, 38 byte packets
     :
     :
13  gar2-p360.wswdc.ip.att.net (12.123.9.57)  227.728 ms  215.958 ms  45.609 ms
```

```
14  12.124.234.18 (12.124.234.18)  209.889 ms  216.571 ms  231.028 ms
15  linx1nap.lo.uk.prserv.net (152.158.16.69)  238.848 ms  221.662 ms  245.854 ms
16  dhclonuk-rt1.attemea.uk.prserv.net (152.158.21.66)  238.772 ms  210.669 ms
231.543 ms
17  dhcbhxuk-rt1-pos1-0.attemea.net (195.33.96.1)  225.819 ms  231.457 ms
231.693 ms
18  195.33.98.115 (195.33.98.115)  225.154 ms  224.166 ms  221.763 ms
```

The last resolvable hop indicates the source address is part of attemea.net (AT&T in Amsterdam, NL).

```
$ whois attemea.net
[whois.networksolutions.com]

    [snip]

Registrant:
AT&T ICoE (ATTEMEA3-DOM)
   Laarderhoogtweg 25
   Amsterdam, 1101 EB
   NL

   Domain Name: ATTEMEA.NET

   Administrative Contact, Technical Contact:
      ICoE  (ICO-ORG)           dns@ATT.NL
      AT&T Internetworking Center of Expertise
      Laarderhoogtweg 25
      Amsterdam
      NL
      +31 20 4097 600
      Fax- +31 20 609 0128

   Record expires on 09-Feb-2003.
   Record created on 09-Feb-1999.
   Database last updated on 3-Aug-2002 22:28:50 EDT.

   Domain servers in listed order:

   NS1.ATT.NL                    194.151.2.22
   NS2.ATT.NL                    194.151.2.77
```

4. Description of attack

The trace shows a pattern of SYN,SYN−ACK, RST sequences. There are three groups of three, with forty minutes between each group. Each cycle of SYN, SYN−ACK, RST (9 packets) completes within one second. The empherical

source port increments sequentially for each group. The last group increments the port number by two, suggesting other TCP/IP traffic may have been generated at this time by the source.

The trace is unusual in several respects. First there is 24 bytes of data in each SYN packet. While this technically doesn't violate any RFCs, it is unusual. Second, domain services primarily use UDP packets, switching to TCP only when the answer records will exceed the UDP 512 byte packet size. Finally, since the source host initiated the connection, one would expect it to complete the three−way hand shake. Instead, a RST is sent to tear down the connection.

5. Attack mechanism

Although DNS resolver routines typically use UDP packets, TCP is the primary protocol for DNS zone transfers. It is possible this trace represents an automated test for zone transfers. The time to complete a cycle of 9 packets (one second) also suggests this is an automated tool.

6. Correlations

A google search for "DNS data tcp syn" produced few leads to explain this traffic:

http://www.geocrawler.com/archives/3/4890/2001/12/0/7405346/

This posting to the snort−user mailing list in 2001, matches the pattern and payload length. A sample packet suggests the payload may be nulls. Speculation on the cause ranged from a broken DNS client (highly likely) to an attempt at tunneling the DNS thru a firewall.

The SANS Intrusion FAQ also shows the pattern, but suggests the source port should be divisible by 100.

http://www.sans.org/newlook/resources/IDFAQ/DNS.htm

The SANS FAQ also suggests this traffic may be an artifact of Foundry Systems 3/DNS appliances gathering load balancing information.

http://www.f5.com/f5products/3dns/index.html

http://www.f5.com/solutions/whitepapers/improvingdns.html

7. Evidence of active targeting

This is clearly active targeting. Mr. Maxwell indicated the probes were happening all day and implied only the one target was involved.

8. Severity: Severity should be calculated with the following formula severity = (criticality + lethality) − (system countermeasures + network countermeasures)

Criticality = 3. The target appears to be a DNS server on a commercial network.

Lethality = 1. The connection is torn down immediately after the SYN−ACK packet is sent making the traffic mostly innocuous. Additionally, the volume of packets was insufficient to flood the network.

System countermeasures = 1. Information about the target host (such as the version of BIND and patch levels) is unknown.

Network countermeasures = 1. Information about the target network is unknown.

(3 + 1) − (1 + 1) = 2

As shown by the research, this traffic was not necessarily malicious and is probably caused by a Foundry 3DNS appliance. Nevertheless, the lack of details about the target network and server elevate the severity rating.

9. Defensive recommendation

To confirm the theory these packets are from a Foundry 3DNS appliance, it might be prudent the send an email to the point of contact for the source address.

As a general practice, the name server should be checked to ensure patch levels are current and zone transfer security measures are in place. The attackers IP address should also be added to a watch list in case the probes were the prelude to a larger attack.

10. Multiple choice test question

Tcpdump uses Berkeley Packet Filter (BPF) primitives to help reduce the amount of data captured or displayed. Given the packets below, what filter could be used to display similar packets.

```
12:14:24.761703 195.33.98.115.50833 > 1.2.3.31.domain: S
893780373:893780397(24) win 2048
12:14:24.761753 1.2.3.31.domain > 195.33.98.115.50833: S
1366686861:1366686861(0) ack 893780374 win 5840 <mss 1460> (DF)
12:14:24.863707 195.33.98.115.50833 > 1.2.3.31.domain: R
893780374:893780374(0) win 0
```

a. 'host 195.33.98.115'

b. 'port domain and source 195.33.98.115'

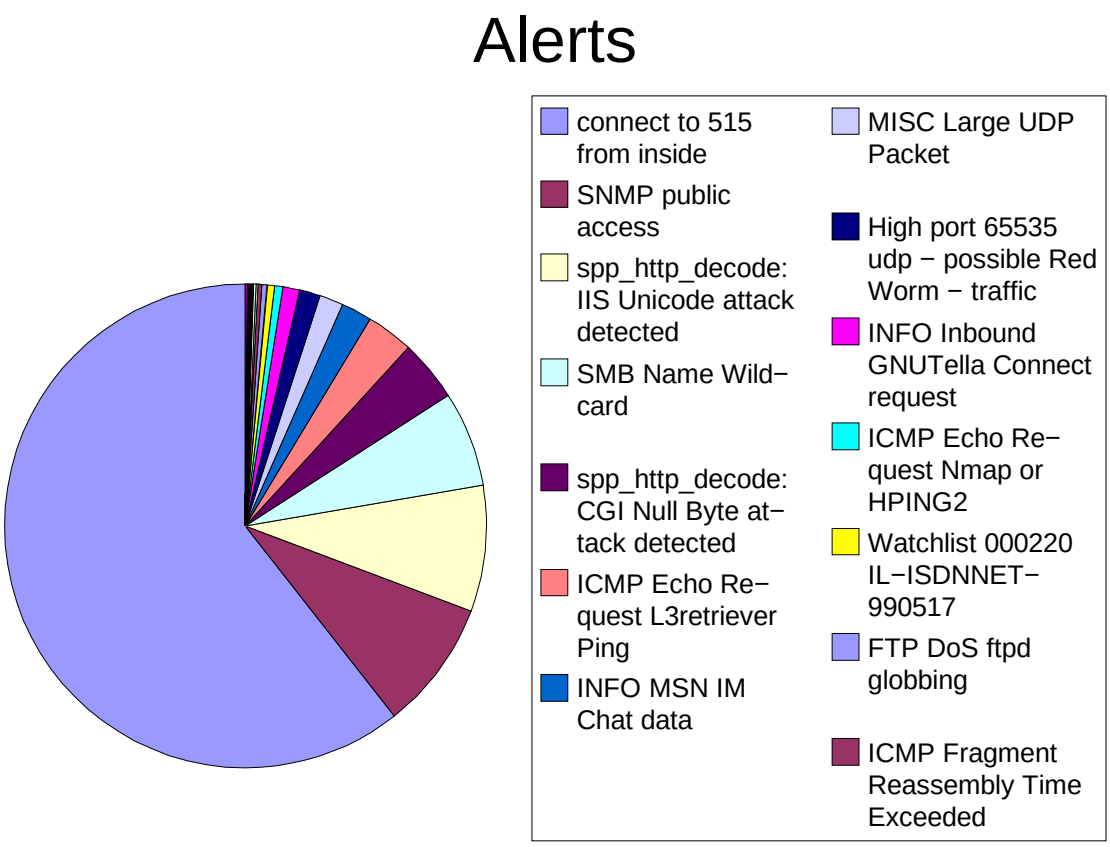
c. 'tcp[13] && (0x16 != 0)'

d. 'tcp[4:4] > 893780373 or tcp[8:4] > 893780374'

Correct answer: a

Executive Summary

For this security analysis, the University provided five days of snort Intrusion Detection System log files covering the period April 1 to April 6, 2002. The data consisted of over 450Mb of raw logs representing 1,049,957 individual alerts and 82 unique types. The pie graph shown below summarizes the top alerts by volume.



Before the analysis, little was known about the University network topology, the number of sensors or placement. Also unknown is the exact snort rules used to generate the alerts (some are obviously 'home grown' as they are not part of the snort distribution, or popular add−ons such as arachNIDS). There is evidence

of asymmetric routing within MY.NET and spotty IDS sensor coverage. IDS alert data was also logged out of sequence (by time) suggesting either a clock synchronization problem or network congestion between the sensors and log server.

There is also evidence of active recon and packet crafting (including spoofed source addresses) against the University networks. From the 82 unique alerts, the top five alerts (by volume) and five high−priority events of interest were selected for detailed analysis.

The detailed analysis revealed possible abuses of Peer−to−Peer (P2P) file sharing protocols such as KaZaa, eDonkey, and gnutella; evidence of IIS attacks such as Code Red and Nimda; probable Sub−Seven backdoor trojan infections and several attempts from external hosts to compromise University NTP (Network Time Protocol) servers.

The corrective measures and defensive recommendations range from eliminating noise by tuning the IDS rules, to a complete review of select University hosts to confirm and clean up the compromise. Terms of Service agreements and Acceptable Use Policies deterring the abuse of P2P protocols should also be reviewed and communicated to University network users. A tightening of the perimeter defense (thru firewalls or border routers) may also be in order.

The list of the files.

The following files were obtained from the incidents.org web site to complete this report.

Alert files	Out of Spec files	Scan files
alert.020401.gz	oos_Apr.1.2002.gz	scans.020401.gz
alert.020402.gz	oos_Apr.2.2002.gz	scans.020402.gz
alert.020403.gz	oos_Apr.3.2002.gz	scans.020403.gz
alert.020404.gz	oos_Apr.4.2002.gz	scans.020404.gz
alert.020405.gz	oos_Apr.5.2002.gz	scans.020405.gz

Detailed analysis

A review of the captured out−of−spec packets shows evidence of active recon and packet crafting against MY.NET. There are for example, many packets with odd flag values that would not normally occur, a packet with the same src/dst port numbers and an instance of port 0 which should never appear as a source.

```
04/01-00:54:21.144077 209.176.66.227:514 -> MY.NET.153.191:514
2*SF**** Seq: 0xA80000   Ack: 0x3C313330   Win: 0x6162
04/02-17:59:33.798812 217.226.38.82:1213 -> MY.NET.153.191:1214
**SF**AU Seq: 0x8DE74E   Ack: 0x5C110001   Win: 0x5010
04/02-19:53:18.792710 24.141.97.182:0 -> MY.NET.153.153:6699
21**R*A* Seq: 0x66000F5   Ack: 0x4BFC0007   Win: 0x5010
04/02-19:56:51.296002 24.141.97.182:0 -> MY.NET.153.153:6699
*1SFRP*U Seq: 0x66000F9   Ack: 0x73200009   Win: 0x5010
04/02-19:57:54.580824 24.141.97.182:6699 -> MY.NET.153.153:1632
*1SFR**U Seq: 0xFA4FA6   Ack: 0xA   Win: 0x5010
04/02-22:23:06.033684 216.232.85.237:3071 -> MY.NET.88.162:1214
21*F**A* Seq: 0x16DE253   Ack: 0x2B45C   Win: 0x5018
04/02-23:19:11.211878 24.191.0.222:44234 -> MY.NET.150.209:6346
21*F*PA* Seq: 0x1058731F   Ack: 0xEDC20000   Win: 0x5018
04/02-23:46:46.502222 211.37.21.179:156 -> MY.NET.150.46:1754
2*SFRPA* Seq: 0xA041007E   Ack: 0x7564007F   Win: 0x5010
04/02-23:50:59.578521 211.37.21.179:1754 -> MY.NET.150.46:41025
**SF*P** Seq: 0x7E7564   Ack: 0x976B07   Win: 0x5010
04/03-10:23:38.806971 68.55.20.174:68 -> MY.NET.5.96:1215
21S*R*AU Seq: 0x500004   Ack: 0x34170E8B   Win: 0x5010
04/03-18:28:12.732736 68.82.88.138:1214 -> MY.NET.153.153:2409
*1SFRP** Seq: 0x50000F   Ack: 0xB4D60016   Win: 0x5010
04/03-19:40:06.155553 205.251.182.200:1964 -> MY.NET.150.133:1214
21SFR*** Seq: 0x680EB91   Ack: 0x0   Win: 0x7002
04/04-00:42:47.978710 142.51.44.123:1900 -> MY.NET.88.162:1214
*1SFRPAU Seq: 0x12CA55B   Ack: 0xBBBAA0   Win: 0x5010
04/04-01:49:54.178235 142.51.44.123:1900 -> MY.NET.88.162:1214
*1SF**** Seq: 0x12CA55B   Ack: 0x9C021   Win: 0x5010
04/04-01:52:50.349038 142.51.44.123:9 -> MY.NET.88.162:1900
2*SFR**U Seq: 0x4BE012C   Ack: 0xA55BC07F   Win: 0x5010
04/04-01:54:27.669069 142.51.44.123:21 -> MY.NET.88.162:1900
*1SF*PAU Seq: 0x4BE012C   Ack: 0xA55BC0A8   Win: 0x5010
04/04-01:54:39.727093 142.51.44.123:1900 -> MY.NET.88.162:1214
*1SFRPAU Seq: 0x12CA55B   Ack: 0x33C0AC   Win: 0x5010
04/04-02:04:34.504903 142.51.44.123:1900 -> MY.NET.88.162:1214
**SFR*AU Seq: 0x11012C   Ack: 0xA55BC191   Win: 0x5010
04/04-02:15:33.804864 142.51.44.123:1900 -> MY.NET.88.162:1214
21SFRP** Seq: 0x12CA55B   Ack: 0xC0C2C3   Win: 0x5010
```

The alert_summary−no−portscan script (see Appendix 1) was used to produce a summary report of all the non−portscan alerts calculating the total for each. The results are shown below.

```
$ ./alert_summary-no-portscan > ./alert.report-no-portscan
$ cat ./alert.report-no-portscan
Using ./alert.all
Report from 04/01-00:16:01.549951 thru 04/05-23:59:57.678969
Count   Alert Description
 636038 connect to 515 from inside
  92595 SNMP public access
  86587 spp_http_decode: IIS Unicode attack detected
  66946 SMB Name Wildcard
  44305 spp_http_decode: CGI Null Byte attack detected
  33491 ICMP Echo Request L3retriever Ping
  22006 INFO MSN IM Chat data
  16799 MISC Large UDP Packet
  14653 High port 65535 udp - possible Red Worm - traffic
  11680 INFO Inbound GNUTella Connect request
   5664 ICMP Echo Request Nmap or HPING2
   4840 Watchlist 000220 IL-ISDNNET-990517
   4048 FTP DoS ftpd globbing
   2228 ICMP Fragment Reassembly Time Exceeded
   1490 ICMP Router Selection
   1317 WEB-IIS view source via translate header
    841 NMAP TCP ping!
    723 WEB-MISC Attempt to execute cmd
    546 INFO Outbound GNUTella Connect request
    322 WEB-IIS _vti_inf access
    320 Watchlist 000222 NET-NCFC
    301 ICMP Echo Request Windows
    299 WEB-FRONTPAGE _vti_rpc access
    271 Null scan!
    158 WEB-CGI scriptalias access
    138 Possible trojan server activity
    137 SCAN Proxy attempt
    122 INFO napster login
    103 ICMP Destination Unreachable (Communication Administratively
Prohibited)
     91 INFO Napster Client Data
     91 ICMP traceroute
     89 INFO Possible IRC Access
     74 WEB-CGI ksh access
     60 ICMP Echo Request BSDtype
     47 MISC traceroute
     46 INFO - Possible Squid Scan
     44 INFO FTP anonymous FTP
```

```
     40 Queso fingerprint
     30 Attempted Sun RPC high port access
     28 ICMP Destination Unreachable (Protocol Unreachable)
     28 EXPLOIT x86 NOOP
     25 WEB-MISC compaq nsight directory traversal
     25 EXPLOIT NTPDX buffer overflow
     24 SCAN Synscan Portscan ID 19104
     23 Back Orifice
     22 MYPARTY - Possible My Party infection
     22 INFO napster upload request
     20 SUNRPC highport access!
     18 WEB-MISC 403 Forbidden
     15 High port 65535 tcp - possible Red Worm - traffic
     14 EXPLOIT x86 setuid 0
     11 Port 55850 tcp - Possible myserver activity - ref. 010313-1
     11 EXPLOIT x86 stealth noop
      8 RPC tcp traffic contains bin_sh
      8 Port 55850 udp - Possible myserver activity - ref. 010313-1
      7 WEB-MISC http directory traversal
      7 IDS552/web-iis_IIS ISAPI Overflow ida nosize
      6 EXPLOIT x86 setgid 0
      5 SCAN FIN
      5 Incomplete Packet Fragments Discarded
      4 WEB-IIS encoding access
      4 TFTP - External UDP connection to internal tftp server
      4 RFB - Possible WinVNC - 010708-1
      3 MISC PCAnywhere Startup
      3 INFO Outbound GNUTella Connect accept
      2 WEB-MISC whisker head
      2 WEB-MISC webdav search access
      2 WEB-IIS asp-dot attempt
      2 WEB-CGI formmail access
      2 TFTP - Internal UDP connection to external tftp server
      2 TELNET access
      2 suspicious host traffic
      2 Probable NMAP fingerprint attempt
      2 MISC source port 53 to <1024
      2 MISC Invalid PCAnywhere Login
      1 x86 NOOP - unicode BUFFER OVERFLOW ATTACK
      1 WEB-MISC ICQ Webfront HTTP DOS
      1 WEB-CGI redirect access
      1 TCP SMTP Source Port traffic
      1 INFO Inbound GNUTella Connect accept
      1 IDS475/web-iis_web-webdav-propfind
      1 ICMP Router Selection (Undefined Code!)
======== 82  Unique alerts
======== 1049957  Total alerts
```

A list of detects and a brief description each

The top five alerts (by volume) and five high−priority events of interest were selected for analysis.

```
636038 connect to 515 from inside
 92595 SNMP public access
 86587 spp_http_decode: IIS Unicode attack detected
 66946 SMB Name Wildcard
 44305 spp_http_decode: CGI Null Byte attack detected

  4840 Watchlist 000220 IL-ISDNNET-990517
   320 Watchlist 000222 NET-NCFC
   138 Possible trojan server activity
    25 EXPLOIT NTPDX buffer overflow
     4 TFTP - External UDP connection to internal tftp server
```

For each analyzed detect, a sample from the alert file is presented along with any CVE numbers for the attack, followed by commentary on the detect.

Appendix 4 'Selected Detect Reports' provides additional supporting data that could not easily be included with the commentary.

Finally, correlations from other sources is provided.

connect to 515 from inside

Sample alert:

```
04/01-07:33:26.000846  [**] connect to 515 from inside [**]
MY.NET.152.171:2348 -> MY.NET.150.198:515
```

CVE−2001−0353 CVE−2001−0670 CVE−2001−1002 CVE−2002−0003

Port 515 is the Unix lpr/lpd printer facility. While there are some known vulnerabilities and remote exploits for various implementations, most of this traffic appears to be valid.

```
$ cat 515.targets
Using ./alert.515
Report from 04/01-07:33:26.005349 thru 04/05-23:59:57.678969
Count   Target
```

```
 331784 MY.NET.150.198
 299713 MY.NET.151.77
   4515 MY.NET.150.83
     25 MY.NET.1.63
      1 MY.NET.5.35
======== 636038  Total alerts
======== 5  Unique targets
```

There are 163 talkers (see Appendix 4), but only 5 unique targets for this alert. The host at MY.NET.150.198 appears to be a valid print server based on the diversity of source addresses. MY.NET.151.77 and MY.NET.150.83 comprise the 2nd and 3rd top targets, but only talk to each other. MY.NET.1.63 has several sources, but it also sends a single request to MY.NET.5.35.

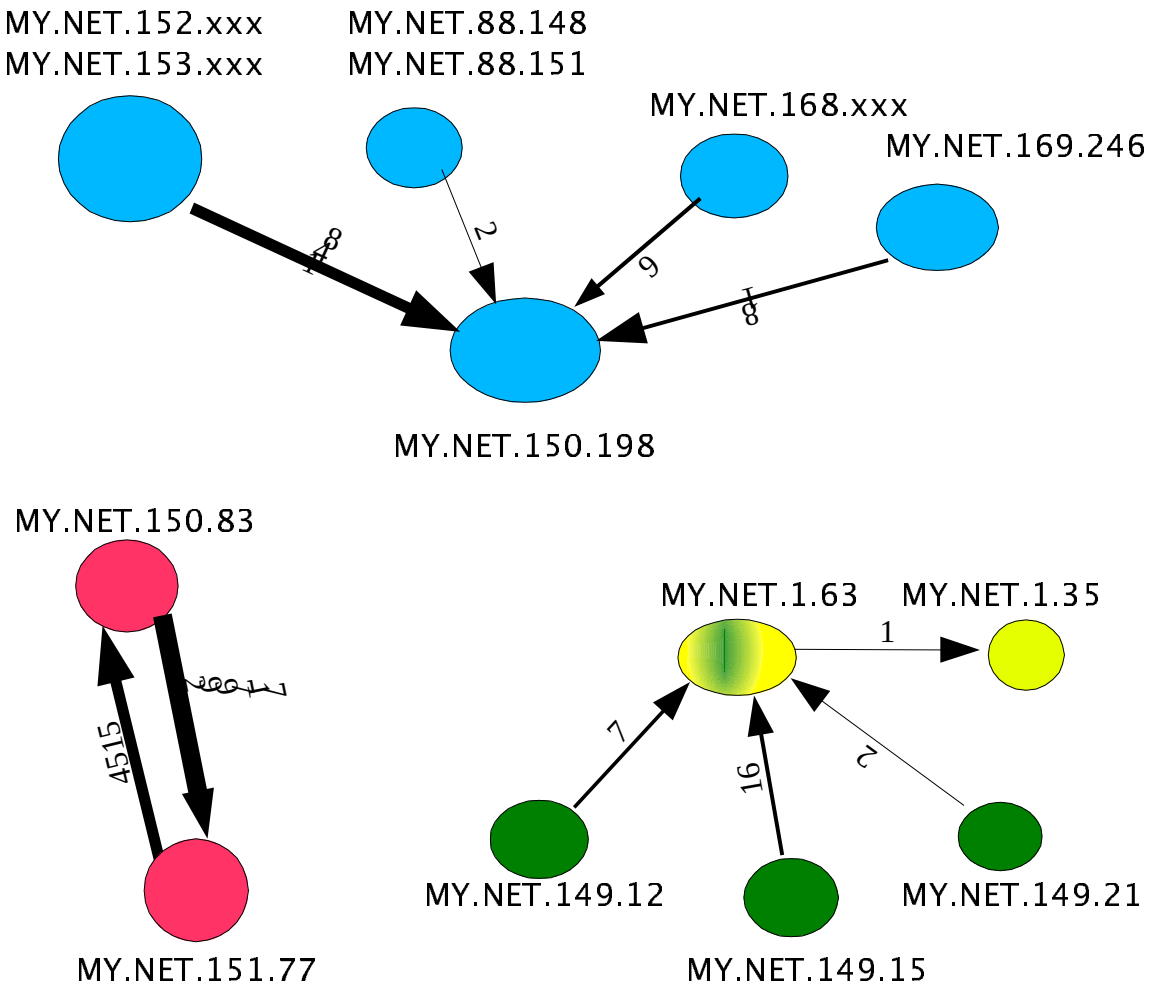
The link graph on the next page illustrates this print server traffic relationship.

The unusual relationship between MY.NET.151.71 and MY.NET.150.83 might be the result of spotty IDS sensor placement. Typically, one would expect to see more sources for a print server. This high level of traffic could also be malicious as David Singer reports in his GCIA practical:

> "This is to alert a possible DOS attack. There is a candidate for inclusion in the CVE list [CAN−2000−0839] that states: "A continuous stream of LPD options, sent to the LPD port (default TCP port 515) on the host running WinCOM, will eventually consume all the memory on that host"

URL: http://www.sans.org/y2k/practical/David_Singer_GCIA.doc

With the exception of the MY.NET.151.71 and MY.NET.150.83 traffic, this alert can be considered low severity noise.



Link graph representing the port 515 lpr/lpd traffic.

The blue and green icons depicts typical lpr/lpd traffic patterns. The yellow, a somewhat unusual pattern (a single request) and the red indicates suspicious traffic.

SNMP public access

Sample alert:

```
04/01-00:00:09.029895  [**] SNMP public access [**]
MY.NET.153.191:1029 -> MY.NET.150.147:161
```

CAN−2002−0012 CAN−2002−0013 CAN−2002−0053

URL: http://www.cert.org/advisories/CA−2002−03.html

URL: http://www.ee.oulu.fi/research/ouspg/protos/testing/c06/snmpv1/index.html

In February, 2002 researchers at the University of Oulu, Finland and CERT released details on multiple vulnerabilities in almost every implementation of the Simple Network Management Protocol version 1 (SNMPv1). Concurrent with the release of the research, the Oulu University Secure Programming Group provided the PROTOS test suite to the community. This suite of java programs tested and reported on the SNMP vulnerabilities.

One of the primary vulnerabilities of SNMP is the use of well known community strings. Many implementations ship with 'public' as a default.

SNMPv1 (described in RFC 1157) typically uses two UDP ports for communication− 161 for management and 162 for alert traps. A review of the SNMP alerts reveals all were generated as the result of probes to port 161.

There are 25 unique sources and 154 unique targets (see Appendix 4) for this alert. All addresses are internal to MY.NET suggesting this alert was not triggered as a result of wide spread probing of SNMP hosts.

The use of default SNMP community strings is so widespread, the vulnerability made the SANS top 20 (and the earlier top 10) 'Most Critical Internet Security Threats' list.

URL: http://www.sans.org/top20.htm

URL: http://www.sans.org/top10.htm

IIS Unicode attack

Sample alert:

```
04/01-01:54:56.247338  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.87.17.168:1718 -> MY.NET.150.195:80
```

CVE−2000−0884 CAN−2001−0709

This alert is triggered, not by a specific snort rule, but by the http_decode processor. This takes the remote input and normalizes it (by converting Unicode and %00 nulls) before applying text signatures. This reduces the attackers ability to avoid detection, but at the cost of additional processing cycles. It can be disabled by specifying the following in snort.conf:

```
preprocessor http_decode: 80 −unicode
```

A typical Unicode attack would be an attempt to perform directory traversal as part of an HTTP URL request (Nimda and Code Red use this method). A properly patched and configured IIS webserver would not allow this activity. Unicode in the URL request would not usually be valid, so sources generating this traffic become suspect.

A review of the timestamps for this attack reveals many occur within the same second for the same source and destination. This is consistent with the scripted nature of the Code Red and Nimda worms.

There were 86587 total alerts directed at 1017 targets. Of these targets, 32 were within MY.NET. There were 154 Unicode requests from MY.NET sources.

In addition to reporting a correlation for this attack, Jeff Zahr has an example of the Unicode packet capture in his GCIA practical.

URL: http://www.giac.org/practical/Jeff_Zahr_GCIA.doc

It is very likely Code Red and/or Nimda is active on the University network.

SMB Name Wildcard

Sample alert:

```
04/01-00:00:04.585824  [**] SMB Name Wildcard [**]
MY.NET.152.182:137 -> MY.NET.11.7:137
```

This is a custom alert description not part of the standard snort ruleset. It triggers on a UDP request to port 137 (Windows and Samba NETBIOS name service) with a specific pattern in the name field− bytes 13−45.

Bryce Alexander provided this write−up for the SANS IDS FAQ:

> "This has two sources, an increase in awareness among script kiddies of the ability to discover information about a target host using NBTSTAT and the spread of an internet worm known as network.vbs."
>
> "This particular trace was crafted by using the windows command: NBTSTAT −A (Target IP Address)".

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

```
$ grep -v MY.NET SMB.sources
Using alert.SMB
Report from 04/01-00:00:23.667727
thru 04/05-23:59:30.852097
Count   Source
     15 169.254.22.29
======== 66946  Total alerts
======== 300  Unique sources
```

```
$ grep -v MY.NET SMB.targets
Using alert.SMB
Report from 04/01-00:00:23.667727
thru 04/05-23:59:30.852097
Count   Target
======== 66946  Total alerts
======== 315  Unique targets
```

Of the 66946 total alerts, there were 300 unique sources and 315 targets. All the addresses were within MY.NET with the exception of one− 169.254.22.29. Addresses from the 169.254.0.0/16 range are assigned by a Microsoft Windows machine when it cannot obtain a DHCP address. This network block is typically dropped at border routers, suggesting this machine is also within MY.NET.

According to the CERT description of the network.vbs worm, target addresses are generated randomly, implying these alerts were not the result of the network.vbs worm or a variant. It is highly unlikely a worm would generate random addresses entirely contained within the host network. A timestamp review also confirms this activity is probably not scripted or automated and

therefore can be considered noise.

URL: http://www.cert.org/incident_notes/IN-2000-02.html

Matthew Fiddler also concludes this alert is low severity noise.

> "Our analysis of GIAC University has determined that all SMB Name Wildcard traffic is in fact internal traffic and therefore is not deemed malicious."

URL: http://www.giac.org/practical/Matthew_Fiddler_GCIA.doc

CGI Null Byte attack

Sample alert:

```
04/01-08:16:23.543409  [**] spp_http_decode: CGI Null Byte attack
detected [**] MY.NET.153.211:2073 -> 205.188.180.25:80
```

CVE-2000-0149 CVE-2000-0332

This alert is triggered, not by a specific snort rule, but by the http_decode processor. This takes the remote input and normalizes it (by converting Unicode and %00 nulls) before applying text signatures. This reduces the attackers ability to avoid detection, but at the cost of additional processing cycles. It can be disabled by specifying the following in snort.conf:

```
preprocessor http_decode: 80 -cginull
```

Like the related IIS Unicode attack, CGI Null attacks typically take advantage of directory traversal vulnerabilities. Some vulnerabilities expose the CGI source code (which may contain passwords or other restricted information) to the attacker.

A review of the timestamps for this attack reveals many occur within the same second for the same source and destination. This is consistent with the scripted nature of the Code Red and Nimda worms.

Within MY.NET, there are 32 unique sources and only 2 targets. The CGINull top_talker report (see Appendix 4) shows MY.NET.153.197 -> 209.10.239.135 at the top. This source address also appears in the top ten overall talkers. This

host deserves further investigation.

```
$ grep 'MY.NET.153.197' ./alert.all > ./alert.153.197
$ ./alert_summary-no-portscan ./alert.153.197
Using ./alert.153.197
Report from 04/01-10:47:47.916193 thru 04/05-19:16:35.945006
Count   Alert Description
  15829 spp_http_decode: CGI Null Byte attack detected
    803 connect to 515 from inside
    210 spp_http_decode: IIS Unicode attack detected
    123 FTP DoS ftpd globbing
     56 High port 65535 udp - possible Red Worm - traffic
     35 ICMP Fragment Reassembly Time Exceeded
      4 SMB Name Wildcard
      2 NMAP TCP ping!
      1 ICMP Echo Request Nmap or HPING2
======== 9  Unique alerts
======== 17063  Total alerts
```

Watchlist 000220 IL−ISDNNET−990517

Sample alert:

```
04/01-00:39:03.551766  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.32.109:24048 -> MY.NET.153.191:1214
```

This is a custom alert signature based on previous suspicious activity for the the source netblock. Watchlists are created to trigger on followup traffic and must be investigated.

```
$ tail -2 WL-220.sources
======== 4840  Total alerts
======== 19  Unique sources
$ tail -2 WL-220.targets
======== 4840  Total alerts
======== 15  Unique targets
```

There are 19 unique sources from the 212.179.0.0/16 network triggering this alert against 15 targets. At first glance, there is no obvious pattern to the traffic (see Appendix 4). However a closer review leads to some observations:

port 80 is often used a source port
port 1214 (KaZaa) is a popular source and destination

212.179.40.132 talks a lot to MY.NET.150.143:4662 (eDonkey)

The date in the alert name suggests this watch list was created on May 19th, 1999. Since that time. there have been many reports of this alert appearing on the SANS incidents list and in GCIA practicals Including Kevin Martin's:

> "According to previous SANS practical assignments − These appear to be locally created rules so that traffic originating from hosts in Jerusalem and/or Beijing will fire an alert. Additionally, there may be IRC and Gnutella communication between you site and hosts on these networks."

URL: http://www.giac.org/practical/Kevin_Martin_GCIA.doc

URL: http://www.sans.org/y2k/052000.htm

Watchlist 000222 NET−NCFC

Sample alert:

```
04/02-20:22:23.783156  [**] Watchlist 000222 NET-NCFC [**]
159.226.83.23:28117 -> MY.NET.150.143:4662
```

This is a custom alert signature based on previous suspicious activity for the the source netblock. Watchlists are created to trigger on followup traffic and must be investigated.

This alert triggered on 4 source addresses from the 159.226.0.0/16 network.

```
$ cat WL-222.talkers
Using alert.WL-222
Report from 04/02-20:22:28.718343 thru 04/05-16:04:44.373913
Count   Source             Destination
    242 159.226.83.23   -> MY.NET.150.143
     50 159.226.47.197  -> MY.NET.153.153
     24 159.226.236.23  -> MY.NET.88.186
      4 159.226.87.6    -> MY.NET.153.164
======== 4  Unique talkers
======== 320  Total alerts
```

As seen in the WL−220 alerts, MY.NET.150.143 is a suspected eDonkey (port 4662) server. The host at 159.226.47.197 is using a source port of 80 to

connect to ports 1752−1754 on MY.NET.153.153. The host at 159.226.236.23 is using port 8080 to connect to MY.NET.88.186 on four ports (2497, 2502, 2503, 2506). The host at 159.226.87.6 is possibly running a gnutella server on port 6346 and MY.NET.153.164 is connected.

```
$ cat WL-222.talkers-with-ports
Using ./alert.WL-222
Report from 04/02-20:22:28.718343 thru 04/05-16:04:44.373913
Count   Source             Destination
     25 159.226.47.197:80         -> MY.NET.153.153:1754
     16 159.226.236.23:8080       -> MY.NET.88.186:2506
     15 159.226.47.197:80         -> MY.NET.153.153:1753
     12 159.226.83.23:46256       -> MY.NET.150.143:4662
     12 159.226.83.23:34225       -> MY.NET.150.143:4662
     12 159.226.83.23:29254       -> MY.NET.150.143:4662
     12 159.226.83.23:23507       -> MY.NET.150.143:4662
     12 159.226.83.23:12946       -> MY.NET.150.143:4662
     11 159.226.83.23:53974       -> MY.NET.150.143:4662
     11 159.226.83.23:41708       -> MY.NET.150.143:4662
     11 159.226.83.23:41083       -> MY.NET.150.143:4662
     11 159.226.83.23:31047       -> MY.NET.150.143:4662
     10 159.226.83.23:52292       -> MY.NET.150.143:4662
     10 159.226.83.23:47934       -> MY.NET.150.143:4662
     10 159.226.83.23:44935       -> MY.NET.150.143:4662
     10 159.226.83.23:34748       -> MY.NET.150.143:4662
     10 159.226.83.23:13964       -> MY.NET.150.143:4662
     10 159.226.47.197:80         -> MY.NET.153.153:1752
      9 159.226.83.23:44783       -> MY.NET.150.143:4662
      9 159.226.83.23:28117       -> MY.NET.150.143:4662
      9 159.226.83.23:21023       -> MY.NET.150.143:4662
      8 159.226.83.23:54051       -> MY.NET.150.143:4662
      7 159.226.83.23:43363       -> MY.NET.150.143:4662
      7 159.226.83.23:38572       -> MY.NET.150.143:4662
      7 159.226.83.23:31327       -> MY.NET.150.143:4662
      6 159.226.83.23:39510       -> MY.NET.150.143:4662
      6 159.226.83.23:16206       -> MY.NET.150.143:4662
      5 159.226.83.23:37797       -> MY.NET.150.143:4662
      5 159.226.83.23:36863       -> MY.NET.150.143:4662
      4 159.226.87.6:6346         -> MY.NET.153.164:2353
      3 159.226.83.23:4662        -> MY.NET.150.143:1608
      3 159.226.236.23:8080       -> MY.NET.88.186:2503
      3 159.226.236.23:8080       -> MY.NET.88.186:2502
      2 159.226.83.23:4662        -> MY.NET.150.143:1098
      2 159.226.83.23:34205       -> MY.NET.150.143:4662
      2 159.226.236.23:8080       -> MY.NET.88.186:2497
      1 159.226.83.23:38029       -> MY.NET.150.143:4662
      1 159.226.83.23:18705       -> MY.NET.150.143:4662
      1 159.226.83.23:18702       -> MY.NET.150.143:4662
======== 39  Unique talkers
======== 320  Total alerts
```

Previous GCIA practicals indicated hosts triggering this alert were targeting SMTP mail hosts. This recent activity shows primarily suspected P2P type traffic.

URL: http://www.zeltser.com/sans/idic−practical/
URL: http://www.giac.org/practical/Ben_Thomas_GCIA.doc
URL: http://www.giac.org/practical/Dennis_Davis_GCIA.doc

Possible trojan server activity

Sample alert:

```
04/01-00:04:28.709118  [**] Possible trojan server activity [**]
MY.NET.5.42:27374 -> MY.NET.5.83:7938
```

URL: http://www.cert.org/incident_notes/IN−2001−01.html

URL: http://www.cert.org/incident_notes/IN−2001−07.html

This attack description is not part of the standard snort rule set, but a review of the alerts shows this triggered on a source or destination port of 27374. This port number is well known for the Windows Sub−seven backdoor and Unix ramen worm. In addition, the W32/Leaves virus often leverages a previously compromised Sub−seven Windows host.

The trojan talkers (with port numbers) report shows 30 unique talkers. Additionally, the report reveals a clear pattern of two−way communications between many of the hosts. Some of the traffic appears to be related to web (port 80) or P2P file sharing (ports 4662, 1214) , but a majority remains suspicious.

```
$ cat trojan.talkers-with-ports
Using ./alert.trojan
Report from 04/01-00:04:28.716718 thru 04/05-10:18:18.410271
Count   Source             Destination
      9 61.222.188.226:27374      -> MY.NET.150.143:4662
      8 MY.NET.70.229:27374       -> MY.NET.5.83:8903
      8 MY.NET.5.42:27374         -> MY.NET.5.83:7938
      8 MY.NET.5.29:27374         -> MY.NET.5.83:8903
```

```
      7 MY.NET.5.83:8903          -> MY.NET.70.229:27374
      7 MY.NET.5.83:8903          -> MY.NET.5.29:27374
      7 MY.NET.5.83:7938          -> MY.NET.5.42:27374
      7 MY.NET.5.42:27374         -> MY.NET.5.83:8139
      7 MY.NET.150.143:4662       -> 61.222.188.226:27374
      5 MY.NET.5.83:8139          -> MY.NET.5.42:27374
      5 MY.NET.5.50:27374         -> MY.NET.5.83:8903
      5 MY.NET.5.44:27374         -> MY.NET.5.83:7938
      5 MY.NET.191.20:27374       -> MY.NET.5.83:7938
      5 MY.NET.185.28:27374       -> MY.NET.5.83:7938
      4 MY.NET.5.83:8903          -> MY.NET.5.50:27374
      4 MY.NET.5.83:7938          -> MY.NET.5.44:27374
      4 MY.NET.5.83:7938          -> MY.NET.191.20:27374
      4 MY.NET.5.83:7938          -> MY.NET.185.28:27374
      3 MY.NET.5.43:27374         -> MY.NET.5.83:7938
      3 MY.NET.150.113:1214       -> 64.119.138.20:27374
      3 MY.NET.150.113:1214       -> 12.237.74.7:27374
      3 64.119.138.20:27374       -> MY.NET.150.113:1214
      3 208.212.50.2:27374        -> MY.NET.5.96:80
      3 12.237.74.7:27374         -> MY.NET.150.113:1214
      2 MY.NET.5.96:80            -> 208.212.50.2:27374
      2 MY.NET.5.83:8139          -> MY.NET.5.55:27374
      2 MY.NET.5.83:7938          -> MY.NET.5.43:27374
      2 MY.NET.150.113:1214       -> 68.3.226.233:27374
      2 68.3.226.233:27374        -> MY.NET.150.113:1214
      1 MY.NET.5.55:27374         -> MY.NET.5.83:8139
======== 30  Unique talkers
======== 138  Total alerts
```

Notice also, the mismatch in packet counts for a source/target pair. This suggests the possibility of asynchronous routing within MY.NET or IDS sensors dropping packets.

David Stewart included an excellent description of the SubSeven backdoor in his GCIA practical:

URL: http://www.giac.org/practical/david_stewart_gcia.doc

EXPLOIT NTPDX buffer overflow

Sample alert:

```
04/01-22:43:39.335116  [**] EXPLOIT NTPDX buffer overflow [**]
64.124.157.16:1523 -> MY.NET.153.45:123
```

CVE−2001−0414

Network Time Protocol (NTP) is a service to synchronize system clocks to a known source. A buffer overflow vulnerability exists in early versions allowing remote attackers to execute commands and gain elevated privileges.

There are 9 sources (all external networks) targeting 6 MY.NET hosts.

```
$ cat NTPDX.talkers
Using alert.NTPDX
Report from 04/02-10:15:21.374413 thru 04/05-18:03:15.067714
Count   Source             Destination
      8 64.232.138.142  -> MY.NET.151.125
      5 63.250.205.34   -> MY.NET.153.46
      3 64.124.157.16   -> MY.NET.153.45
      2 64.124.157.16   -> MY.NET.153.211
      2 64.124.157.10   -> MY.NET.153.45
      1 66.77.13.134    -> MY.NET.152.246
      1 63.250.219.190  -> MY.NET.153.46
      1 63.250.205.44   -> MY.NET.153.46
      1 63.250.205.3    -> MY.NET.153.46
      1 63.146.181.125  -> MY.NET.88.155
======== 25  Total alerts
======== 10  Unique talkers
```

Several of the source addresses resolve to the yahoo.com and akamaitechnologies.com domains. It's likely these are spoofed. Because it isn't necessary for the attacker to receive a reply, this attack should be treated as high severity and the targets investigated throughly.

There were several instances of this alert appearing in GCIA practical summary lists, but no one chose to analyze the traffic in detail. More information on this exploit, including and exhaustive list of vulnerable implementations and a copy of the exploit code is available from SecurityFocus.

URL: http://online.securityfocus.com/bid/2540/info/

TFTP − External UDP connection to internal tftp server

Sample alert:

```
04/01-12:04:08.755839  [**] TFTP - External UDP connection to
internal tftp server [**] 63.250.205.10:256 -> MY.NET.153.46:69
```

CVE−1999−0183 CAN−2001−0783 CAN−2002−0813

TFTP is an unauthenticated UDP file transfer protocol. It is typically used by network equipment vendors (such as Cisco) to provide a method to backup/restore a configuration or network boot the device. TFTP is also one of the propagation vectors for the Nimda worm.

Because the data connections are unauthenticated, there is little reason to allow external hosts to initiate a transfer using the protocol. This alert was triggered 4 times as shown below.

```
04/01-12:04:08.755839  [**] TFTP - External UDP connection to internal tftp
server [**] 63.250.205.10:256 -> MY.NET.153.46:69
04/02-11:41:05.826050  [**] TFTP - External UDP connection to internal tftp
server [**] 63.250.219.189:0 -> MY.NET.153.46:69
04/03-11:21:28.527684  [**] TFTP - External UDP connection to internal tftp
server [**] 63.250.205.36:256 -> MY.NET.153.46:69
04/03-13:30:54.909734  [**] TFTP - External UDP connection to internal tftp
server [**] 63.250.219.189:16495 -> MY.NET.153.45:69
```

Surprisingly, each of the source addresses are registered to the yahoo.com domain. Because this is UDP traffic, it's probable the source addresses are spoofed.

Correlating this attack with the scans.all file uncovers this might simply be part of a somewhat noisy UDP scan. However, it's also possible the scan is a smoke screen to hide the TFTP attack− many port patterns recur throughout the scan, but these are the only times port 69 is a destination. A snippet of one scan is shown below (others were similar).

```
Apr  1 12:04:05 63.250.205.10:0 -> MY.NET.153.46:0 UDP
Apr  1 12:04:02 63.250.205.10:43693 -> MY.NET.153.46:59030 UDP
Apr  1 12:04:03 63.250.205.10:3705 -> MY.NET.153.46:1334 UDP
Apr  1 12:04:04 63.250.205.10:4443 -> MY.NET.153.46:17589 UDP
Apr  1 12:04:05 63.250.205.10:3485 -> MY.NET.153.46:2790 UDP
Apr  1 12:04:06 63.250.205.10:7000 -> MY.NET.153.46:7001 UDP
Apr  1 12:04:09 63.250.205.10:0 -> MY.NET.153.46:0 UDP
Apr  1 12:04:08 63.250.205.10:256 -> MY.NET.153.46:69 UDP
Apr  1 12:04:09 63.250.205.10:3705 -> MY.NET.153.46:1334 UDP
Apr  1 12:04:13 63.250.205.10:0 -> MY.NET.153.46:0 UDP
Apr  1 12:04:10 63.250.205.10:3485 -> MY.NET.153.46:2790 UDP
Apr  1 12:04:12 63.250.205.10:7001 -> MY.NET.153.46:7000 UDP
Apr  1 12:04:13 63.250.205.10:23345 -> MY.NET.153.46:23854 UDP
Apr  1 12:04:13 63.250.205.10:27258 -> MY.NET.153.46:26735 UDP
Apr  1 12:04:14 63.250.205.10:13275 -> MY.NET.153.46:35350 UDP
```

Because full packet data was not provided, it is impossible to make a clear

determination for this alert. The target hosts should be checked for compromise.

Top Talkers lists

The top_talkers script reports 1,049,957 total alerts from 18,474 unique source/destination pairs. The top ten are shown below.

```
$ head -13 top_talkers.report
Using ./alert.all
Report from 04/01-00:16:01.549951 thru 04/05-23:59:57.678969
Count   Source             Destination
 299713 MY.NET.150.83   -> MY.NET.151.77
  74895 MY.NET.153.164  -> MY.NET.150.198
  57104 MY.NET.153.118  -> MY.NET.150.198
  28102 MY.NET.153.126  -> MY.NET.150.198
  16451 MY.NET.153.119  -> MY.NET.150.198
  15829 MY.NET.153.197  -> 209.10.239.135
   9713 MY.NET.88.203   -> MY.NET.150.195
   9624 MY.NET.88.181   -> MY.NET.150.195
   9596 MY.NET.88.207   -> MY.NET.150.195
   9581 MY.NET.88.145   -> MY.NET.150.195
```

As shown previously, MY.NET.151.77 and MY.NET.150.198 are print servers, and MY.NET.150.195 is a web server. The following command parses the top talkers report for the top ten external sources.

```
$  awk '$2 !~ /^MY\.NET/' top_talkers.report | head -13
Using ./alert.all
Report from 04/01-00:16:01.549951 thru 04/05-23:59:57.678969
Count   Source
   2129 63.240.15.205   -> MY.NET.153.153
   2106 61.78.35.42     -> MY.NET.153.171
   2027 61.78.35.44     -> MY.NET.153.171
   1584 210.94.0.146    -> MY.NET.153.164
   1504 163.239.2.31    -> MY.NET.153.110
   1474 216.106.173.144 -> MY.NET.153.174
   1297 216.106.173.150 -> MY.NET.153.174
   1285 212.179.40.132  -> MY.NET.150.143
   1216 63.240.15.207   -> MY.NET.153.171
   1213 169.232.80.45   -> MY.NET.153.171
```

The top_scanners script produces a report of scanners. The top ten overall and the top ten external hosts are shown.

```
$ head -13 scanner.report
Using ./scans.all
Report from Apr 1 00:00:00 thru Apr 5 23:59:26
Attacker          TOTAL scans
MY.NET.60.43         462096
MY.NET.150.143       283592
MY.NET.6.45          196947
MY.NET.6.48          181565
MY.NET.6.49          179920
MY.NET.6.52          168898
MY.NET.6.50          136484
MY.NET.11.8           88843
MY.NET.6.53           83955
MY.NET.6.60           72094
```

```
$ grep -v MY.NET scanner.report | head -13
Using ./scans.all
Report from Apr 1 00:00:00 thru Apr 5 23:59:26
Attacker          TOTAL scans
64.124.157.16         14867
64.124.157.10          4860
205.188.228.33         3560
66.28.225.156          3314
64.124.157.64          3272
64.232.138.142         3251
66.28.8.69             3033
205.188.228.129        3001
63.250.219.154         2812
66.28.14.37            2798
```

Correlating the top external scanner IP addresses with the alerts reveals no TCP traffic, only ICMP and UDP.

$ grep {IP} ./alerts.all | grep −v 'TCP(0)' | more

This suggests these addresses are spoofed. Reducing the scans.all file to just external sources and TCP traffic, then producing the top scanners report results in these less likely to be spoofed addresses:

```
$ grep -v UDP scans.all |awk '$4 !~ /MY.NET/' > scans.TCP-only
$ ./top_scanners ./scans.TCP-only > scanner.report-TCP-only
$ head -13 ./scanner.report-TCP-only
Using ./scans.TCP-only
Report from Apr 1 00:04:07 thru Apr 5 23:46:01
Attacker          TOTAL scans
217.85.93.106           586
```

```
130.104.56.130          512
24.27.182.214           500
198.182.119.70          440
195.188.221.3           378
218.146.252.36          358
213.66.213.194          343
212.131.130.57          327
62.16.193.163           313
194.86.228.200          300
```

Registration information for five external source addresses.

Top talker (external to MY.NET) 63.240.15.205

```
$ whois 63.240.15.205@whois.arin.net
[whois.arin.net]
AT&T CERFnet (NETBLK-CERFNET-BLK-5)
   P.O. Box 919014
   San Diego, CA  92191
   US

   Netname: CERFNET-BLK-5
   Netblock: 63.240.0.0 - 63.242.255.255
   Maintainer: CERF

   Coordinator:
      AT&T Enhanced Network Services  (CERF-HM-ARIN)  notify@attens.com
      (858) 812-5000

   Domain System inverse mapping provided by:

   DBRU.BR.NS.ELS-GMS.ATT.NET   199.191.128.106
   CBRU.BR.NS.ELS-GMS.ATT.NET   199.191.128.105
   DMTU.MT.NS.ELS-GMS.ATT.NET   12.127.16.70
   CMTU.MT.NS.ELS-GMS.ATT.NET   12.127.16.69

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 06-Aug-2001.
   Database last updated on  23-Aug-2002 16:56:03 EDT.
```

CGINull external 12.91.161.167 and 24.162.83.132

```
$ whois  12.91.161.167@whois.arin.net
```

```
AT&T (NET-ATT)
   AT&T ITS
   200 Laurel Avenue South
   Middletown, NJ 07748
   US

   Netname: ATT
   Netblock: 12.0.0.0 - 12.255.255.255
   Maintainer: ATTW

   Coordinator:
      Kostick, Deirdre  (DK71-ARIN)  help@ip.att.net
      1-919-319-8249

   Domain System inverse mapping provided by:

   DBRU.BR.NS.ELS-GMS.ATT.NET        199.191.128.106
   DMTU.MT.NS.ELS-GMS.ATT.NET        12.127.16.70
   CBRU.BR.NS.ELS-GMS.ATT.NET        199.191.128.105
   CMTU.MT.NS.ELS-GMS.ATT.NET        12.127.16.69

   For abuse issues contact abuse@att.net

   Record last updated on 23-Aug-2002.
   Database last updated on  23-Aug-2002 16:56:03 EDT.
```

```
$ whois 24.162.83.132@whois.arin.net

ROADRUNNER (NET-ROAD-RUNNER-5)
   13241 Woodland Park Road
   Herndon, VA 20171
   US

   Netname: ROAD-RUNNER-5
   Netblock: 24.160.0.0 - 24.170.127.255
   Maintainer: RRMA

   Coordinator:
      ServiceCo LLC  (ZS30-ARIN)  abuse@rr.com
      1-703-345-3416

   Domain System inverse mapping provided by:

   DNS1.RR.COM               24.30.200.3
   DNS2.RR.COM               24.30.201.3
   DNS3.RR.COM               24.30.199.7
   DNS4.RR.COM               65.24.0.172

   Record last updated on 22-Aug-2002.
   Database last updated on  23-Aug-2002 16:56:03 EDT.
```

NTPDX exploit top source 64.232.136.142

```
$ host 64.232.136.142
142.136.232.64.in-addr.arpa. domain name pointer 142.136.232.64.transedge.com.

$ whois transedge.com@whois.bulkregister.com

   [snip]

New Edge Networks, Inc.
   3000 Columbia House Blvd. Ste 106
   Vancouver, WA 98661
   US

   Domain Name: TRANSEDGE.COM

   Administrative Contact:
        NEN Hostmaster    hostmaster@newedgenetworks.com
        New Edge Networks
        3000 Columbia House Blvd. Suite 106
        Vancouver, WA 98661
        US
        Phone: (360) 906-9749
        Fax:
   Technical Contact:
        NEN Hostmaster  hostmaster@newedgenetworks.com
        New Edge Networks
        3000 Columbia House Blvd. Suite 106
        Vancouver, WA 98661
        US
        Phone: (360) 906-9749
        Fax:

   Record updated on 2002-08-05 16:16:01.
   Record created on 2000-04-29.
   Record expires on 2003-04-29.
   Database last updated on 2002-08-24 21:50:59 EST.

   Domain servers in listed order:

   HNS1.NEWEDGENETWORKS.COM      64.232.128.3
   HNS2.NEWEDGENETWORKS.COM      209.125.236.3


$ whois 64.232.136.142@whois.arin.net

New Edge Networks (NET-NEN-AW5)      NEN-AW5
64.232.0.0 - 64.232.255.255
SAMPLEREELS, INC. (DSL REPLACEMENT) (NETBLK-ATWORK-53470-45405)    ATWORK-53470-45405
64.232.136.128 - 64.232.136.143

$ whois \!netblk-atwork-53470-45405@whois.arin.net

SAMPLEREELS, INC. (DSL REPLACEMENT) (NETBLK-ATWORK-53470-45405)
   1011 PICO BLVD., #8
   SANTA MONICA, CA 90405
   US

   Netname: ATWORK-53470-45405
   Netblock: 64.232.136.128 - 64.232.136.143
```

```
   Coordinator:
      KLINE, JOSH  (JK1275-ARIN)  JKLINE@samplereels.com
      310-396-9151

   Record last updated on 26-Jul-2001.
   Database last updated on  23-Aug-2002 16:56:03 EDT.
```

Top scanner 217.85.93.106 (non-spoofed)

```
$ host 217.85.93.106
106.93.85.217.in-addr.arpa. domain name pointer pD9555D6A.dip.t-dialin.net.
$ whois t-dialin.net
[whois.networksolutions.com]

[snip]

Registrant:
Deutsche Telekom Online Service GmbH (T-DIALIN2-DOM)
   Waldstrasse 3
   Weiterstadt, D-64331
   DE

   Domain Name: T-DIALIN.NET

   Administrative Contact, Technical Contact:
      Kaufmann, Daniel  (DK162-RIPE)            d.kaufmann@T-ONLINE.NET
      Deutsche Telekom Online Service GmbH
      Julius-Reiber-Str.37
      Darmstadt
      Germany
      D-6429
      DE
      +49 61 51 680 537 (FAX) +49 61 51 680 519

   Record expires on 10-Feb-2003.
   Record created on 10-Feb-1999.
   Database last updated on 26-Aug-2002 18:03:55 EDT.

   Domain servers in listed order:

   DNS00.SDA.T-ONLINE.DE        195.145.119.62
   DNS01.SDA.T-ONLINE.DE        195.145.119.189
   DNS00.SUL.T-ONLINE.DE        62.153.158.62
   DNS01.SUL.T-ONLINE.DE        194.25.134.203

$ whois 217.85.93.106@whois.ripe.net
[whois.ripe.net]

[snip]

inetnum:      217.80.0.0 - 217.89.31.255
netname:      DTAG-DIAL14
descr:        Deutsche Telekom AG
country:      DE
```

```
admin-c:      DTIP-RIPE
tech-c:       ST5359-RIPE
status:       ASSIGNED PA
remarks:      ********************************************************
remarks:      * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
remarks:      * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.   *
remarks:      ********************************************************
notify:       auftrag@nic.telekom.de
notify:       dbd@nic.dtag.de
mnt-by:       DTAG-NIC
changed:      auftrag@nic.telekom.de 20020108
source:       RIPE

route:        217.80.0.0/12
descr:        Deutsche Telekom AG, Internet service provider
origin:       AS3320
mnt-by:       DTAG-RR
changed:      rv@NIC.DTAG.DE 20001027
source:       RIPE

person:       DTAG Global IP-Adressing
address:      Deutsche Telekom AG
address:      Bayreuther Strasse 1
address:      D-90409 Nuernberg
address:      Germany
phone:        +49 911 68909856
e-mail:       ripe.dtip@telekom.de
nic-hdl:      DTIP-RIPE
mnt-by:       DTAG-NIC
changed:      ripe.dtip@telekom.de 20020717
source:       RIPE

person:       Security Team
address:      Deutsche Telekom AG
address:      Technikniederlassung Schwaebisch Hall
address:      D-89070 Ulm
address:      Germany
phone:        +49 731 100 84055
fax-no:       +49 731 100 84150
e-mail:       abuse@t-ipnet.de
nic-hdl:      ST5359-RIPE
notify:       auftrag@nic.telekom.de
notify:       dbd@nic.dtag.de
mnt-by:       DTAG-NIC
changed:      auftrag@nic.telekom.de 20010321
source:       RIPE
```

Summary and defensive recommendations.

There is evidence spotty IDS sensor coverage along with either a clock synchronization problem or network congestion between the sensors and log server. Code Red and Nimda, despite the availablity of patches, still runs rampant generating a significant number of alerts. SubSeven also appears to be a problem, but to a lesser degree.

Improvements to the University security posture and Intrusion Detection Systems can be made by implementing the following:

Tune IDS rules to reduce alert volume and false positives. Review sensor placement and ensure times are synchronized.

Review these hosts for compromise or TOS violations:

MY.NET.151.77 / MY.NET.150.83 (515 connects)
MY.NET.153.197 (top talker to external and CGINull, etc)
MY.NET.153.153 / MY.NET.88.186 (Watch List 222)
MY.NET.153.45 / MY.NET.153.46 (TFTP / NTP targets)
MY.NET.151.125 / MY.NET.153.211 (NTP exploit targets)
MY.NET.151.246 / MY.NET.88.155 (NTP exploit targets)
MY.NET.150.143 / MY.NET.153.164 (P2P abusers)
The 154 sources within MY.NET from the Unicode sources report
The 32 sources within MY.NET from the CGINull sources report
The following suspected sub−seven infected hosts

MY.NET.5.83	MY.NET.70.229	MY.NET.5.29
MY.NET.191.20	MY.NET.185.28	MY.NET.5.50
MY.NET.5.44	MY.NET.5.42	MY.NET.5.55

Change SNMP community strings to something other than ’public’

Block TFTP at border routers, check targets for compromise

Consider blocking ntp from external sources, except for trusted hosts. Ensure ntp servers are fully patched.

Finally, Terms of Service agreements and Acceptable Use Policies deterring the abuse of P2P protocols should also be reviewed and communicated to University network users.

The analysis process

After downloading and uncompressing the five days of files, I concatenated each type into a single file (alert.all, scans.all, oos.all) for the time period.

It became obvious very early on that a standard tool in my box, SnortSnarf, would not help with these volumes of data. The complete alert file for the five days amounted to 228Mb, causing SnortSnarf to consume all memory and swap space after just 20 minutes of processing. Running against just the first day's file (45Mb) resulted in over 14 hours of thrashing before the process was stopped. Later reading of other GCIA papers shows this to be a common problem.

The proper course of action would normally be to setup a database, but in the interest of time, I relied on GNU/Linux tools such as shell scripting, grep, awk, etc. The decision to follow this path was was easy to make as I already had several test awk scripts to process alert and portscan files. These scripts required only a small amount of tweaking to help with this project. The source for the scripts is listed in Appendix 1.

There are six primary scripts:

alert_summary−no−portscan
alert_sources
alert_targets
alert_talkers
top_talkers
top_scanners

Each script was written to accept a filename on the command line. This allows the scripts to be reused and process alerts for different data sets.

Processing alerts

The first step involved using the alert_summary−no−portscan script to produce a report from the alert.all file showing the total number of (non−portscan) alerts and the number for each unique alert. This resulted in 82 unique alerts from 1049957 overall.

```
$ ./alert_summary−no−portscan > ./alert.report−no−portscan
```

Then each alert selected for review was grep'd into it's own file for processing by the alert_sources and alert_targets scripts.

```
$ grep 'alert string' ./alert.all > ./alert.{ID}
$ ./alert_sources ./alert.{ID} > ./{ID}.sources
$ ./alert_targets ./alert.{ID} > ./{ID}.targets
```

The top_talkers script produces a list of unique source and destination pairs, along with totals for each.

```
$ ./top_talkers ./alert.{ID} > ./{ID}.talkers
```

This reduced the massive volume of data to a much more manageable amount in the following report files:

```
alert.515         alert.NTPDX     alert.SNMP     alert.Unicode     alert.WL-222
alert.CGINull     alert.SMB       alert.TFTP     alert.WL-220      alert.trojan

515.sources       NTPDX.sources   SNMP.sources   Unicode.sources   WL-222.sources
515.talkers       NTPDX.talkers   SNMP.talkers   Unicode.talkers   WL-222.talkers
515.targets       NTPDX.targets   SNMP.targets   Unicode.targets   WL-222.targets
CGINull.sources   SMB.sources     TFTP.sources   WL-220.sources    trojan.sources
CGINull.talkers   SMB.talkers     TFTP.talkers   WL-220.talkers    trojan.talkers
CGINull.targets   SMB.targets     TFTP.targets   WL-220.targets    trojan.targets
```

Processing scans

The top_scanners script was used to produce the list of top ten scanning hosts overall and the top ten external to MY.NET.

```
$ ./top_scanners ./scans.all > scanner.report
$ head −13 scanner.report
$ grep −v MY.NET scanner.report | head −13
```

Processing out−of−spec packets

Because there were only 60 out−of−spec packets, I decided against writing scripts to process the data. Using egrep to display just the address and flag lines in the oos.all file reveals a majority of packets use the ECN flags (see sample packets below).

```
04/02-11:17:49.303666 217.96.21.210:51309 -> MY.NET.153.143:6346
21S***** Seq: 0xD0F8FFB7   Ack: 0x0   Win: 0x16D0
04/02-11:43:57.594927 24.232.140.16:40355 -> MY.NET.153.143:6346
21S***** Seq: 0x44AA2194   Ack: 0x0   Win: 0x16D0
```

This is a fairly recent addition to the Linux networking support and is responsible for many IDS false alarms. Most of these ECN packets also bear the passive fingerprint of Linux (window size of 0x16D0, MSS, timestamp, SackOK).

Ignoring the '21S' (ECN/SYN) packets left only 18 to be reviewed.

Administrivia

The analysis workstation is a dual proc Pentium II/400 with 256Mb RAM and multiple SCSI disks. The system is running Mandrake 7.1 (2.2.15−4mdksmp kernel). Work in progress is rsync'd to another server daily , then occasionally burned to CD−ROM.

Besides the standard GNU utilities the following are part of the toolbox:

Snort Version 1.8.7 (Build 128)
SnortSnarf version 020516.1
Snort Report 1.11
alert2db.pl v1.0
acid 0.9.6b13
tcpdump version 3.4
ethereal 0.9.5
tcpreplay 1.1

This report was produced with StarOffice 5.2, printed to a PostScript file and converted to PDF with ps2pdf to meet the GIAC publishing standard requirement. The resulting document was tested for proper formatting and layout using kghostview under Linux and Adobe ACROREAD under Windows 95 & 2000.

GCIA Practical V3.1 − Appendix 1 source code for scripts

top_attackers −− from detect #1

```
#!/bin/sh
#
#   Prints the top attackers
#
if [ $# = 0 ] ; then
       MESSAGES=/var/log/messages
else
       MESSAGES=$1
fi
if ! [ -f $MESSAGES ] ; then
       echo "$MESSAGES file is not found!"
       exit 1
fi

echo "Using $MESSAGES"
START=$(head -1 $MESSAGES | awk '{print $1, $2, $3}')
END=$(tail -1 $MESSAGES | awk '{print $1, $2, $3}')
echo "Report from $START thru $END"
echo "Attacker          DST Port     Port Count  IP TOTAL"


#
#  note the 'not /CROND' below...   This prevents the copy
#   of a cron job from being counted.
#
awk '/ACCEPT|REJECT|DENY/ && ! /CROND/ \
{
# some common protocols from /etc/protocols

proto[0] = "ip";
proto[1] = "icmp";
proto[2] = "igmp";
proto[6] = "tcp";
proto[17] = "udp";

fmtstr = "%-15s  %10s  %10s %10s\n";
srcip = substr($12,1,index($12,":")-1);
dstport = sprintf("%5s/%s", substr($13,index($13,":")+1),
                        proto[substr($11,index($11,"=")+1)]);

attacker[srcip];
service[dstport];
total[srcip] ++;
count[srcip, dstport] ++;
}
END {
```

```
  for (ip in attacker) {
      for ( port in service) {
         if ( count[ip, port] != 0 ) {
            printf(fmtstr, ip, port, count[ip, port], total[ip]);
         }
      }
  }

}' \
 $MESSAGES | sort -nr -k4
```

alert_summary−no−portscan −− from Analyze This!

```
#!/bin/sh
#
#   Prints a summary of the alerts
#

if [ $# = 0 ] ; then
       MESSAGES=./alert.all
else
       MESSAGES=$1
fi
if ! [ -f $MESSAGES ] ; then
       echo "$MESSAGES file is not found!"
       exit 1
fi

echo "Using $MESSAGES"
START=$(head -4 $MESSAGES | tail -1 | awk '{print $1}')
END=$(tail -1 $MESSAGES | awk '{print $1}')
echo "Report from $START thru $END"
echo "Count    Alert Description"

#
#
awk '/\[\*\*\]/ && ! /PORTSCAN/ &&! /portscan status/ && ! /End of portscan/ \
{

fmtstr = "%7s %-70s\n";

#
#  Breakdown description (between [**] markers)
#
#
   temp = substr($0, index($0, "\[\*\*\]")+5);
```

```
    desc = substr(temp, 1, index(temp, "\[\*\*\]")-1);


alert[desc];
total[desc] ++;
tot_alerts ++;
}
END {
   for (desc in alert) {
              tot_desc ++;
   }

   print("========",tot_desc," Unique alerts");

   for (desc in alert) {
             printf(fmtstr, total[desc], desc);
   }

print("========",tot_alerts," Total alerts");

}' \
 $MESSAGES | sort -nr -k1
```

alert_sources −− from Analyze This!

```
#!/bin/sh
#
#   Prints a summary of the alerts by source -> target
#

if [ $# = 0 ] ; then
       MESSAGES=./alert.all
else
       MESSAGES=$1
fi
if ! [ -f $MESSAGES ] ; then
       echo "$MESSAGES file is not found!"
       exit 1
fi

echo "Using $MESSAGES"
START=$(head -4 $MESSAGES | tail -1 | awk '{print $1}')
END=$(tail -1 $MESSAGES | awk '{print $1}')
echo "Report from $START thru $END"
echo "Count   Source"

#
```

```
#
awk '/->/  \
{

fmtstr = "%7s %-15s\n";

split($0, s1, " -> ");
addr = split(s1[1], junk, " ");

source = junk[addr];
if(match(source, "\:")) {
   source = substr(source, 1, match(source, "\:")-1);
}


alert[source];
total[source] ++;
tot_alerts++;
}
END {
  for (source in alert) {
            tot_source ++;
  }

  print("========",tot_source," Unique sources");

  for (source in alert) {
           printf(fmtstr, total[source], source);
  }

  print("========",tot_alerts," Total alerts");


}' \
 $MESSAGES | sort -nr -k1
```

alert_targets −− from Analyze This!

```
#!/bin/sh
#
#   Prints a summary of the alerts by source -> target
#

if [ $# = 0 ] ; then
      MESSAGES=./alert.all
else
      MESSAGES=$1
fi
```

```
if ! [ -f $MESSAGES ] ; then
       echo "$MESSAGES file is not found!"
       exit 1
fi

echo "Using $MESSAGES"
START=$(head -4 $MESSAGES | tail -1 | awk '{print $1}')
END=$(tail -1 $MESSAGES | awk '{print $1}')
echo "Report from $START thru $END"
echo "Count   Target"

#
#
awk '/->/  \
{

fmtstr = "%7s %-15s\n";

split($0, s1, " -> ");
addr = split(s1[2], junk, " ");

target = junk[1];
if(match(target, "\:")) {
   target = substr(target, 1, match(target, "\:")-1);
}


alert[target];
total[target] ++;
tot_alerts++;

}
END {
  for (target in alert) {
            tot_targets++;
  }

  print("========",tot_targets," Unique targets");


  for (target in alert) {
            printf(fmtstr, total[target], target);
  }

  print("========",tot_alerts," Total alerts");

}' \
 $MESSAGES | sort -nr -k1
```

top_talkers −− from Analyze This!

```
#!/bin/sh
#
#   Prints a summary of the alerts by source & target pair
#

if [ $# = 0 ] ; then
        MESSAGES=./alert.all
else
        MESSAGES=$1
fi
if ! [ -f $MESSAGES ] ; then
        echo "$MESSAGES file is not found!"
        exit 1
fi

echo "Using $MESSAGES"
START=$(head -4 $MESSAGES | tail -1 | awk '{print $1}')
END=$(tail -1 $MESSAGES | awk '{print $1}')
echo "Report from $START thru $END"
echo "Count   Source             Destination"

#
#
awk '/->/  \
{

fmtstr = "%7s %-35s\n";
pairstr= "%-15s -> %-15s";

split($0, s1, " -> ");
addr = split(s1[1], junk, " ");

source = junk[addr];

split(s1[2], junk, " ");
target = junk[1];

if(match(source, "\:")) {
    source = substr(source, 1, match(source, "\:")-1);
}
if(match(target, "\:")) {
    target = substr(target, 1, match(target, "\:")-1);
}

pair = sprintf(pairstr,source,target);

alert[pair];
```

```
total[pair] ++;
tot_alerts++;
}
END {
   for (pair in alert) {
                tot_pair ++;
   }

   print("========",tot_pair," Unique talkers");

   for (pair in alert) {
               printf(fmtstr, total[pair], pair);
   }

   print("========",tot_alerts," Total alerts");


}' \
 $MESSAGES | sort -nr -k1
```

NOTE: For the some detects, it was advantageous to view the top talkers including the source & destination port numbers. This report was produced by simply expanding the output format strings, and commenting out the lines removing the ports from the top_talkers script:

```
$ diff top_talkers top_talkers.withports
27,28c27,28
< fmtstr = "%7s %-35s\n";
< pairstr= "%-15s -> %-15s";
---
> fmtstr = "%7s %-55s\n";
> pairstr= "%-25s -> %-25s";
38,43c38,43
< if(match(source, "\:")) {
<    source = substr(source, 1, match(source, "\:")-1);
< }
< if(match(target, "\:")) {
<    target = substr(target, 1, match(target, "\:")-1);
< }
---
> ###if(match(source, "\:")) {
> ###    source = substr(source, 1, match(source, "\:")-1);
> ###}
> ###if(match(target, "\:")) {
> ###    target = substr(target, 1, match(target, "\:")-1);
> ###}
```

top_scanners −− from Analyze This!

```
#!/bin/sh
#
#   Prints the top scanners
#

if [ $# = 0 ] ; then
        MESSAGES=./scans.all
else
        MESSAGES=$1
fi
if ! [ -f $MESSAGES ] ; then
        echo "$MESSAGES file is not found!"
        exit 1
fi

echo "Using $MESSAGES"
START=$(head -4 $MESSAGES | tail -1 | awk '{print $1, $2, $3}')
END=$(tail -1 $MESSAGES | awk '{print $1, $2, $3}')
echo "Report from $START thru $END"
echo "Attacker          TOTAL scans"

#
#
awk '/->/ \
{

fmtstr = "%-15s  %10s\n";
srcip = substr($4,1,index($4,":")-1);

attacker[srcip];
total[srcip] ++;
}
END {

   for (srcip in attacker) {
        printf(fmtstr, srcip, total[srcip]);
   }

}' \
 $MESSAGES | sort -nr -k2
```

GCIA Practical V3.1 − Appendix 2 Linux ipchains syslog entry

An earlier version of this Linux ipchains syslog breakdown was posted by the author here:

http://lists.leap−cf.org/pipermail/leaplist/2001−August/014126.html

This example is the first syslog entry in detect #1.

NOTE: ipchains normally places the TCP or UDP source and destination ports following the addresses. However, because ICMP is port−less, these fields are used for the ICMP type and code.

```
Jun  5 09:31:21 firewall kernel: Packet log: input DENY ppp0 PROTO=1
216.52.62.69:8 MY.NET.238.28:0 L=84 S=0x00 I=0 F=0x4000 T=51 (#22)


     syslog component          Description

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

     Jun 5 09:31:21            At this date and time
     firewall                  the computer named 'firewall'
     kernel: Packet log:       logged a packet at the kernel level.
     input                     The 'input' chain
     DENY                      denied the packet
     ppp0                      on interface 'ppp0'.
     PROTO=1                   The packet used ICMP  (grep
                                    /etc/protocols)
     216.52.62.69:8            and came from this source IP address
                                    The ICMP type is 8 (echo)
     MY.NET.238.28:0           It was destined for this IP address
                                    The ICMP code is 0
     L=84                      The length was 84 bytes
     S=0x00                    )               (  Type of service
     I=0                       ) TCP/IP info   (  IP Identification
     F=0x4000                  )               (  Fragmentation &
                                                      Flags
     T=51                      the TTL was 51
     (#22)                     ipchains rule number 22 was
                                    responsible for causing
                                    this log to be generated.
```

GCIA Practical V3.1 − Appendix 3 tcpdump entry

This example breaks down the first tcpdump entry in detect #3. Refer to the tcpdump man page for more details on other packet types.

```
12:14:24.555315 195.33.98.115.50831 > 1.2.3.31.domain: S
1705035803:1705035827(24) win 2048
```

tcpdump component	Description
12:14:24.555315	At this time (hh:mm:ss.frac),
195.33.98.115	A host with this IP address
50831	Used this source port #
>	to send a packet
1.2.3.31	to this destination IP address
domain	and this resolved port name
S	The SYN flag was set. Note: this also indicates TCP traffic
1705035803 1705035827	The packet sequence numbers.
(24)	There were 24 bytes of data
win 2048	The receive window size of the source is 2048 bytes.
	There were no IP options. If present, they would be enclosed in angle brackets <>.

GCIA Practical V3.1 − Appendix 4 − selected detect reports

This appendix provides additional supporting data where the reports would be too lengthy to place in line with the 'Analyze this!' detect commentary.

connect to 515 from inside

```
$ cat 515.talkers
Using ./alert.515
Report from 04/01-07:33:26.005349 thru 04/05-23:59:57.678969
Count   Source             Destination
 299713 MY.NET.150.83   -> MY.NET.151.77
  74895 MY.NET.153.164  -> MY.NET.150.198
  57104 MY.NET.153.118  -> MY.NET.150.198
  28102 MY.NET.153.126  -> MY.NET.150.198
  16451 MY.NET.153.119  -> MY.NET.150.198
   9127 MY.NET.153.113  -> MY.NET.150.198
   8978 MY.NET.153.136  -> MY.NET.150.198
   8263 MY.NET.153.211  -> MY.NET.150.198
   6186 MY.NET.153.121  -> MY.NET.150.198
   4554 MY.NET.153.123  -> MY.NET.150.198
   4515 MY.NET.151.77   -> MY.NET.150.83
   4299 MY.NET.153.105  -> MY.NET.150.198
   4162 MY.NET.153.117  -> MY.NET.150.198
   3838 MY.NET.153.106  -> MY.NET.150.198
   3833 MY.NET.153.184  -> MY.NET.150.198
   3711 MY.NET.153.114  -> MY.NET.150.198
   3446 MY.NET.153.110  -> MY.NET.150.198
   3110 MY.NET.153.127  -> MY.NET.150.198
   2863 MY.NET.153.120  -> MY.NET.150.198
   2842 MY.NET.153.112  -> MY.NET.150.198
   2666 MY.NET.153.135  -> MY.NET.150.198
   2632 MY.NET.153.125  -> MY.NET.150.198
   2514 MY.NET.153.124  -> MY.NET.150.198
   2459 MY.NET.153.115  -> MY.NET.150.198
   2302 MY.NET.152.165  -> MY.NET.150.198
   2271 MY.NET.153.179  -> MY.NET.150.198
   1922 MY.NET.153.150  -> MY.NET.150.198
   1518 MY.NET.153.176  -> MY.NET.150.198
   1472 MY.NET.153.111  -> MY.NET.150.198
   1454 MY.NET.153.108  -> MY.NET.150.198
   1437 MY.NET.153.140  -> MY.NET.150.198
   1414 MY.NET.152.172  -> MY.NET.150.198
   1372 MY.NET.152.166  -> MY.NET.150.198
   1351 MY.NET.153.168  -> MY.NET.150.198
   1337 MY.NET.152.160  -> MY.NET.150.198
   1287 MY.NET.153.204  -> MY.NET.150.198
   1246 MY.NET.152.170  -> MY.NET.150.198
```

```
1188 MY.NET.153.107  -> MY.NET.150.198
1176 MY.NET.152.171  -> MY.NET.150.198
1123 MY.NET.153.209  -> MY.NET.150.198
1101 MY.NET.153.162  -> MY.NET.150.198
1081 MY.NET.153.161  -> MY.NET.150.198
1039 MY.NET.153.206  -> MY.NET.150.198
1036 MY.NET.152.45   -> MY.NET.150.198
1023 MY.NET.152.13   -> MY.NET.150.198
1019 MY.NET.153.180  -> MY.NET.150.198
1010 MY.NET.152.184  -> MY.NET.150.198
 999 MY.NET.152.161  -> MY.NET.150.198
 971 MY.NET.153.148  -> MY.NET.150.198
 944 MY.NET.152.175  -> MY.NET.150.198
 934 MY.NET.152.178  -> MY.NET.150.198
 931 MY.NET.152.177  -> MY.NET.150.198
 920 MY.NET.152.163  -> MY.NET.150.198
 878 MY.NET.153.195  -> MY.NET.150.198
 864 MY.NET.152.159  -> MY.NET.150.198
 861 MY.NET.152.16   -> MY.NET.150.198
 805 MY.NET.152.10   -> MY.NET.150.198
 803 MY.NET.153.197  -> MY.NET.150.198
 794 MY.NET.153.202  -> MY.NET.150.198
 782 MY.NET.152.46   -> MY.NET.150.198
 777 MY.NET.152.216  -> MY.NET.150.198
 760 MY.NET.152.174  -> MY.NET.150.198
 755 MY.NET.153.142  -> MY.NET.150.198
 750 MY.NET.153.109  -> MY.NET.150.198
 735 MY.NET.153.149  -> MY.NET.150.198
 723 MY.NET.152.183  -> MY.NET.150.198
 717 MY.NET.153.187  -> MY.NET.150.198
 695 MY.NET.153.181  -> MY.NET.150.198
 676 MY.NET.88.148   -> MY.NET.150.198
 665 MY.NET.153.193  -> MY.NET.150.198
 648 MY.NET.152.12   -> MY.NET.150.198
 620 MY.NET.153.169  -> MY.NET.150.198
 604 MY.NET.152.19   -> MY.NET.150.198
 583 MY.NET.152.181  -> MY.NET.150.198
 582 MY.NET.153.143  -> MY.NET.150.198
 565 MY.NET.153.166  -> MY.NET.150.198
 551 MY.NET.152.22   -> MY.NET.150.198
 534 MY.NET.152.180  -> MY.NET.150.198
 532 MY.NET.152.215  -> MY.NET.150.198
 522 MY.NET.153.207  -> MY.NET.150.198
 511 MY.NET.153.203  -> MY.NET.150.198
 511 MY.NET.153.185  -> MY.NET.150.198
 500 MY.NET.152.176  -> MY.NET.150.198
 500 MY.NET.152.167  -> MY.NET.150.198
 491 MY.NET.152.250  -> MY.NET.150.198
 479 MY.NET.153.71   -> MY.NET.150.198
 479 MY.NET.153.199  -> MY.NET.150.198
 475 MY.NET.152.251  -> MY.NET.150.198
 458 MY.NET.152.179  -> MY.NET.150.198
```

```
457 MY.NET.153.163  -> MY.NET.150.198
453 MY.NET.153.146  -> MY.NET.150.198
450 MY.NET.153.175  -> MY.NET.150.198
448 MY.NET.152.162  -> MY.NET.150.198
439 MY.NET.153.171  -> MY.NET.150.198
439 MY.NET.152.173  -> MY.NET.150.198
438 MY.NET.153.141  -> MY.NET.150.198
431 MY.NET.153.189  -> MY.NET.150.198
423 MY.NET.152.158  -> MY.NET.150.198
415 MY.NET.153.177  -> MY.NET.150.198
410 MY.NET.152.164  -> MY.NET.150.198
408 MY.NET.153.205  -> MY.NET.150.198
407 MY.NET.153.194  -> MY.NET.150.198
384 MY.NET.152.246  -> MY.NET.150.198
354 MY.NET.152.182  -> MY.NET.150.198
339 MY.NET.152.247  -> MY.NET.150.198
337 MY.NET.152.213  -> MY.NET.150.198
336 MY.NET.152.21   -> MY.NET.150.198
332 MY.NET.168.143  -> MY.NET.150.198
329 MY.NET.152.17   -> MY.NET.150.198
321 MY.NET.153.190  -> MY.NET.150.198
314 MY.NET.152.249  -> MY.NET.150.198
314 MY.NET.152.157  -> MY.NET.150.198
294 MY.NET.152.11   -> MY.NET.150.198
281 MY.NET.153.153  -> MY.NET.150.198
279 MY.NET.152.168  -> MY.NET.150.198
278 MY.NET.153.182  -> MY.NET.150.198
274 MY.NET.152.252  -> MY.NET.150.198
257 MY.NET.153.145  -> MY.NET.150.198
255 MY.NET.153.198  -> MY.NET.150.198
253 MY.NET.152.169  -> MY.NET.150.198
245 MY.NET.153.210  -> MY.NET.150.198
244 MY.NET.153.174  -> MY.NET.150.198
232 MY.NET.153.160  -> MY.NET.150.198
223 MY.NET.152.14   -> MY.NET.150.198
217 MY.NET.153.159  -> MY.NET.150.198
203 MY.NET.153.137  -> MY.NET.150.198
188 MY.NET.168.141  -> MY.NET.150.198
187 MY.NET.153.200  -> MY.NET.150.198
185 MY.NET.152.15   -> MY.NET.150.198
182 MY.NET.153.147  -> MY.NET.150.198
181 MY.NET.152.245  -> MY.NET.150.198
169 MY.NET.153.196  -> MY.NET.150.198
149 MY.NET.153.165  -> MY.NET.150.198
144 MY.NET.153.157  -> MY.NET.150.198
143 MY.NET.152.186  -> MY.NET.150.198
141 MY.NET.152.185  -> MY.NET.150.198
133 MY.NET.168.15   -> MY.NET.150.198
126 MY.NET.88.151   -> MY.NET.150.198
120 MY.NET.152.20   -> MY.NET.150.198
115 MY.NET.152.244  -> MY.NET.150.198
102 MY.NET.153.46   -> MY.NET.150.198
```

```
     91 MY.NET.152.248  -> MY.NET.150.198
     91 MY.NET.152.214  -> MY.NET.150.198
     87 MY.NET.153.152  -> MY.NET.150.198
     86 MY.NET.153.186  -> MY.NET.150.198
     83 MY.NET.152.44   -> MY.NET.150.198
     76 MY.NET.153.167  -> MY.NET.150.198
     72 MY.NET.153.172  -> MY.NET.150.198
     57 MY.NET.153.188  -> MY.NET.150.198
     53 MY.NET.153.154  -> MY.NET.150.198
     49 MY.NET.168.11   -> MY.NET.150.198
     39 MY.NET.153.173  -> MY.NET.150.198
     37 MY.NET.153.144  -> MY.NET.150.198
     19 MY.NET.168.17   -> MY.NET.150.198
     18 MY.NET.169.246  -> MY.NET.150.198
     17 MY.NET.153.45   -> MY.NET.150.198
     16 MY.NET.149.15   -> MY.NET.1.63
     14 MY.NET.168.142  -> MY.NET.150.198
     10 MY.NET.152.18   -> MY.NET.150.198
      9 MY.NET.153.170  -> MY.NET.150.198
      7 MY.NET.149.12   -> MY.NET.1.63
      2 MY.NET.149.21   -> MY.NET.1.63
      1 MY.NET.1.63     -> MY.NET.5.35
======== 636038  Total alerts
======== 163  Unique talkers
```

SNMP public access sources (report wrapped to two columns)

```
$ cat SNMP.sources
Using ./alert.SNMP
Report from 04/01-00:01:14.514015
thru 04/05-23:56:31.437679
Count   Source
  12333 MY.NET.70.177
   9713 MY.NET.88.203
   9624 MY.NET.88.181
   9596 MY.NET.88.207
   9581 MY.NET.88.145
   9524 MY.NET.88.159
   9197 MY.NET.88.136
   5596 MY.NET.88.251
   5096 MY.NET.150.198
   2772 MY.NET.88.212
   2232 MY.NET.153.220
```

```
   2004 MY.NET.150.41
   1182 MY.NET.150.245
   1091 MY.NET.88.185
   1031 MY.NET.88.138
   1008 MY.NET.153.191
    532 MY.NET.88.225
    290 MY.NET.186.10
    122 MY.NET.70.42
     22 MY.NET.183.11
     21 MY.NET.111.30
     18 MY.NET.165.20
      6 MY.NET.71.87
      3 MY.NET.150.114
      1 MY.NET.6.51
======== 92595  Total alerts
======== 25  Unique sources
```

SNMP public access targets (report wrapped to two columns)

```
$ cat SNMP.targets
Using ./alert.SNMP
Report from 04/01-00:01:14.514015
thru 04/05-23:56:31.437679
Count   Target
  65612 MY.NET.150.195
   5441 MY.NET.152.109
   2553 MY.NET.5.127
   2325 MY.NET.5.97
   2209 MY.NET.5.96
   1637 MY.NET.150.84
   1602 MY.NET.151.114
   1482 MY.NET.113.202
   1049 MY.NET.150.231
   1016 MY.NET.150.147
    918 MY.NET.5.92
    839 MY.NET.5.95
    778 MY.NET.5.83
    495 MY.NET.5.248
    415 MY.NET.5.137
    398 MY.NET.5.143
    390 MY.NET.5.243
    329 MY.NET.5.31
    322 MY.NET.153.219
    196 MY.NET.5.141
     74 MY.NET.104.200
     46 MY.NET.5.79
     43 MY.NET.5.128
     40 MY.NET.151.86
     38 MY.NET.88.187
     36 MY.NET.88.160
     33 MY.NET.5.87
     32 MY.NET.88.217
     30 MY.NET.5.85
     29 MY.NET.5.90
     29 MY.NET.5.104
     29 MY.NET.150.14
     28 MY.NET.5.102
     28 MY.NET.5.101
     26 MY.NET.5.32
     26 MY.NET.5.204
     26 MY.NET.151.77
     25 MY.NET.86.14
     25 MY.NET.71.24
     25 MY.NET.163.138
     25 MY.NET.163.12
     25 MY.NET.162.241
     25 MY.NET.145.75
     25 MY.NET.143.245
     25 MY.NET.138.230
     25 MY.NET.138.205
     25 MY.NET.116.70
     25 MY.NET.106.205
     25 MY.NET.106.202
     25 MY.NET.106.200
     25 MY.NET.106.195
     25 MY.NET.104.208
     25 MY.NET.100.39
     24 MY.NET.86.6
     24 MY.NET.86.13
     24 MY.NET.71.16
     24 MY.NET.5.242
     24 MY.NET.5.108
     24 MY.NET.190.13
     24 MY.NET.182.246
     24 MY.NET.178.189
     24 MY.NET.178.107
     24 MY.NET.160.148
     24 MY.NET.151.52
     24 MY.NET.138.228
     24 MY.NET.138.215
     24 MY.NET.138.214
     24 MY.NET.107.37
     24 MY.NET.106.210
     24 MY.NET.106.206
     24 MY.NET.106.199
     23 MY.NET.5.107
     23 MY.NET.5.103
     23 MY.NET.177.38
     23 MY.NET.160.118
     22 MY.NET.86.8
     21 MY.NET.5.110
     21 MY.NET.5.105
     20 MY.NET.99.121
     20 MY.NET.145.100
     20 MY.NET.108.246
     20 MY.NET.104.206
     20 MY.NET.104.119
     17 MY.NET.87.215
     17 MY.NET.86.9
     17 MY.NET.178.183
     15 MY.NET.87.218
     15 MY.NET.86.55
     15 MY.NET.86.18
     15 MY.NET.86.10
     15 MY.NET.70.75
     15 MY.NET.53.229
     15 MY.NET.53.228
```

```
     15 MY.NET.5.109
     15 MY.NET.178.131
     15 MY.NET.163.56
     15 MY.NET.163.44
     15 MY.NET.163.43
     15 MY.NET.163.42
     15 MY.NET.163.108
     15 MY.NET.162.242
     15 MY.NET.162.123
     15 MY.NET.162.109
     15 MY.NET.162.104
     15 MY.NET.156.29
     15 MY.NET.156.123
     15 MY.NET.139.26
     15 MY.NET.130.182
     15 MY.NET.130.181
     15 MY.NET.116.81
     15 MY.NET.109.73
     15 MY.NET.109.72
     15 MY.NET.109.51
     15 MY.NET.104.128
     15 MY.NET.104.114
     15 MY.NET.10.183
     14 MY.NET.87.214
     14 MY.NET.86.39
     14 MY.NET.86.17
     14 MY.NET.85.40
     14 MY.NET.70.85
     14 MY.NET.70.14
     14 MY.NET.178.139
     14 MY.NET.178.109
     14 MY.NET.162.31
```

```
     14 MY.NET.162.30
     14 MY.NET.162.203
     14 MY.NET.162.175
     14 MY.NET.115.12
     14 MY.NET.111.152
     14 MY.NET.10.179
     13 MY.NET.70.170
     13 MY.NET.185.84
     13 MY.NET.163.11
     13 MY.NET.115.163
     12 MY.NET.85.25
     12 MY.NET.162.240
     11 MY.NET.86.21
     11 MY.NET.150.54
     11 MY.NET.150.178
     10 MY.NET.150.243
     10 MY.NET.150.171
     10 MY.NET.150.169
      9 MY.NET.5.240
      9 MY.NET.150.172
      8 MY.NET.150.170
      7 MY.NET.150.51
      6 MY.NET.150.55
      5 MY.NET.110.79
      4 MY.NET.5.99
      3 MY.NET.5.38
      3 MY.NET.5.106
      3 MY.NET.150.52
      1 MY.NET.152.180
======== 92595  Total alerts
======== 154  Unique targets
```

IIS Unicode attack

```
$ cat Unicode.targets
Using ./alert.Unicode
Report from 04/01-01:54:57.860628 thru 04/05-23:16:15.603966
Count   Target
[snip]
======== 86587  Total alerts
======== 1017  Unique targets
```

This report shows the Unicode targets within MY.NET (32 total).

```
$ grep 'MY.NET' Unicode.targets |
wc -l
     32
$ grep 'MY.NET' Unicode.targets
     53 MY.NET.150.195
     29 MY.NET.150.83
     28 MY.NET.88.187
     25 MY.NET.88.217
     23 MY.NET.150.101
     21 MY.NET.150.133
     18 MY.NET.150.231
     16 MY.NET.153.159
     16 MY.NET.151.114
     14 MY.NET.150.63
     13 MY.NET.150.228
     10 MY.NET.5.79
     10 MY.NET.150.147
      9 MY.NET.151.77
      9 MY.NET.150.6
      8 MY.NET.5.95
      8 MY.NET.150.246
      8 MY.NET.150.220
      7 MY.NET.5.243
      7 MY.NET.150.243
      6 MY.NET.150.107
      5 MY.NET.153.220
      3 MY.NET.88.156
      2 MY.NET.5.92
      2 MY.NET.150.143
      1 MY.NET.5.96
      1 MY.NET.150.84
      1 MY.NET.150.51
      1 MY.NET.150.197
      1 MY.NET.150.16
      1 MY.NET.150.139
      1 MY.NET.11.4
```

This report shows the Unicode attackers within MY.NET (154 in all).

```
$ grep 'MY.NET' Unicode.sources |
wc -l
    154
$ grep 'MY.NET' Unicode.sources
   4850 MY.NET.153.146
   3434 MY.NET.153.120
   3336 MY.NET.153.124
   3136 MY.NET.153.110
   3097 MY.NET.153.171
   2731 MY.NET.153.199
   2569 MY.NET.153.189
   2444 MY.NET.153.180
   2244 MY.NET.153.165
   2138 MY.NET.153.112
   2052 MY.NET.153.106
   2000 MY.NET.153.203
   1968 MY.NET.88.148
   1925 MY.NET.88.254
   1789 MY.NET.153.108
   1770 MY.NET.153.163
   1749 MY.NET.153.160
   1717 MY.NET.153.113
   1685 MY.NET.153.119
   1526 MY.NET.153.141
   1494 MY.NET.88.171
   1485 MY.NET.153.211
   1469 MY.NET.153.142
   1385 MY.NET.153.111
   1377 MY.NET.88.243
   1275 MY.NET.153.176
   1265 MY.NET.153.153
   1210 MY.NET.153.164
   1207 MY.NET.153.167
   1188 MY.NET.152.247
   1059 MY.NET.153.114
    990 MY.NET.153.144
    970 MY.NET.153.205
    869 MY.NET.153.166
    866 MY.NET.153.115
    790 MY.NET.153.143
    719 MY.NET.153.182
    648 MY.NET.153.125
    644 MY.NET.152.215
    608 MY.NET.153.193
    585 MY.NET.153.179
    585 MY.NET.153.150
    569 MY.NET.153.196
    545 MY.NET.153.204
    539 MY.NET.153.168
    502 MY.NET.153.206
    460 MY.NET.152.182
    459 MY.NET.153.154
    451 MY.NET.153.194
    428 MY.NET.153.169
    414 MY.NET.153.159
    388 MY.NET.152.249
```

```
386 MY.NET.153.172
360 MY.NET.153.117
342 MY.NET.153.123
336 MY.NET.153.210
302 MY.NET.153.148
294 MY.NET.153.209
285 MY.NET.150.97
282 MY.NET.153.118
277 MY.NET.153.195
263 MY.NET.153.109
262 MY.NET.153.162
253 MY.NET.153.145
252 MY.NET.152.19
240 MY.NET.152.46
235 MY.NET.151.73
223 MY.NET.153.121
219 MY.NET.153.149
215 MY.NET.152.21
214 MY.NET.152.169
212 MY.NET.152.162
210 MY.NET.153.197
209 MY.NET.150.103
207 MY.NET.153.152
195 MY.NET.152.175
180 MY.NET.153.161
167 MY.NET.153.175
166 MY.NET.153.208
163 MY.NET.88.165
163 MY.NET.88.151
160 MY.NET.153.137
154 MY.NET.153.107
140 MY.NET.152.15
132 MY.NET.152.216
130 MY.NET.153.174
126 MY.NET.150.165
124 MY.NET.152.12
121 MY.NET.153.186
121 MY.NET.153.177
113 MY.NET.152.20
108 MY.NET.153.71
104 MY.NET.152.160
 97 MY.NET.151.14
 94 MY.NET.152.183
 93 MY.NET.150.210
 90 MY.NET.152.165
 80 MY.NET.152.248
 78 MY.NET.152.11
 75 MY.NET.153.127
 75 MY.NET.152.163
 71 MY.NET.152.161
 71 MY.NET.151.64
```

```
 69 MY.NET.153.126
 68 MY.NET.152.244
 67 MY.NET.153.140
 62 MY.NET.153.135
 61 MY.NET.152.171
 54 MY.NET.153.202
 54 MY.NET.152.16
 50 MY.NET.152.172
 49 MY.NET.152.157
 48 MY.NET.152.252
 48 MY.NET.149.27
 47 MY.NET.153.190
 47 MY.NET.153.157
 46 MY.NET.88.251
 32 MY.NET.153.147
 28 MY.NET.153.188
 28 MY.NET.152.178
 27 MY.NET.153.181
 26 MY.NET.150.232
 26 MY.NET.150.226
 22 MY.NET.152.164
 21 MY.NET.153.198
 19 MY.NET.153.185
 16 MY.NET.152.250
 12 MY.NET.88.140
 12 MY.NET.152.166
 11 MY.NET.152.181
 10 MY.NET.152.180
 10 MY.NET.150.73
  9 MY.NET.153.207
  8 MY.NET.88.137
  8 MY.NET.153.184
  8 MY.NET.152.13
  7 MY.NET.88.145
  6 MY.NET.153.136
  6 MY.NET.152.150
  5 MY.NET.152.14
  4 MY.NET.153.170
  4 MY.NET.152.213
  3 MY.NET.88.249
  3 MY.NET.153.45
  3 MY.NET.152.45
  3 MY.NET.152.44
  2 MY.NET.152.49
  2 MY.NET.152.185
  2 MY.NET.152.158
  2 MY.NET.152.142
  1 MY.NET.152.17
  1 MY.NET.152.159
  1 MY.NET.150.45
  1 MY.NET.150.107
```

CGI Null Byte attack

```
$ cat CGINull.sources
Using ./alert.CGINull
Report from 04/01-10:40:39.001154
thru 04/05-14:22:43.816944
Count   Source
  15829 MY.NET.153.197
   8730 MY.NET.153.193
   4386 MY.NET.153.149
   4279 MY.NET.153.208
   4139 MY.NET.153.171
   2222 MY.NET.153.153
   1365 MY.NET.153.184
   1169 MY.NET.152.11
    946 MY.NET.153.194
    627 MY.NET.153.210
    126 MY.NET.88.189
     97 MY.NET.218.194
     74 MY.NET.152.46
     64 MY.NET.153.121
     64 MY.NET.150.206
     44 MY.NET.218.182
     33 MY.NET.152.15
     30 MY.NET.152.21
     16 12.91.161.167
     15 MY.NET.204.102
     15 MY.NET.152.215
     11 24.162.83.132
      4 MY.NET.152.160
      3 MY.NET.153.205
      3 MY.NET.153.185
      3 MY.NET.153.154
      2 MY.NET.153.198
      2 MY.NET.152.169
      2 MY.NET.150.103
      1 MY.NET.153.211
      1 MY.NET.153.206
      1 MY.NET.153.152
      1 MY.NET.152.247
      1 MY.NET.152.181
======== 44305  Total alerts
======== 34  Unique sources
```

```
$ cat CGINull.targets
Using ./alert.CGINull
Report from 04/01-10:40:39.001154
thru 04/05-14:22:43.816944
Count   Target
  26730 209.10.239.135
   6300 152.163.210.75
   3792 207.189.79.124
   2658 207.189.75.40
   2232 205.188.132.67
   1169 216.241.219.22
    402 206.61.145.3
    384 63.162.230.3
    172 MY.NET.5.96
    106 216.33.88.141
     64 209.143.193.70
     64 199.104.95.15
     63 216.32.120.220
     29 216.32.114.16
     24 206.61.145.195
     16 204.253.104.95
     12 63.251.36.20
     12 205.188.180.57
     11 MY.NET.153.159
      9 205.188.180.25
      7 216.33.157.32
      6 131.118.254.40
      6 131.118.254.37
      5 208.184.29.70
      5 208.184.29.210
      3 216.32.120.183
      3 206.65.183.25
      3 206.132.135.71
      2 63.251.36.22
      2 216.32.120.130
      2 208.185.54.13
      2 208.184.29.150
      2 208.184.29.110
      1 66.37.219.2
      1 66.135.193.137
      1 64.215.175.131
      1 216.33.156.119
      1 216.32.120.159
      1 216.32.120.137
      1 208.184.29.190
      1 206.65.183.40
======== 44305  Total alerts
======== 41  Unique targets
```

Watchlist 000220 IL−ISDNNET−990517

NOTE: See the note following the top_talkers script in Appendix 1 explaining how this report was created.

```
$ cat WL-220.talkers-with-ports
Using ./alert.WL-220
Report from 04/01-08:55:49.788419 thru 04/05-15:34:37.180934
Count   Source             Destination
    644 212.179.35.118:80         -> MY.NET.153.164:1454
    427 212.179.35.8:80           -> MY.NET.150.204:1336
    413 212.179.40.132:64360      -> MY.NET.150.143:4662
    380 212.179.35.118:80         -> MY.NET.153.174:1971
    342 212.179.35.118:80         -> MY.NET.153.163:1185
    299 212.179.35.118:80         -> MY.NET.153.153:1983
    225 212.179.35.118:80         -> MY.NET.153.174:1143
    117 212.179.40.132:64037      -> MY.NET.150.143:4662
     48 212.179.48.2:40241        -> MY.NET.88.162:1214
     42 212.179.27.176:80         -> MY.NET.153.153:2016
     40 212.179.27.176:80         -> MY.NET.153.164:1394
     39 212.179.27.176:80         -> MY.NET.153.174:1976
     35 212.179.48.2:41873        -> MY.NET.88.162:1214
     34 212.179.27.176:80         -> MY.NET.153.164:1480
     28 212.179.48.2:40373        -> MY.NET.88.162:1214
     25 212.179.48.2:42036        -> MY.NET.88.162:1214
     25 212.179.27.176:80         -> MY.NET.153.164:1398
     24 212.179.112.100:80        -> MY.NET.153.196:2825
     23 212.179.27.176:80         -> MY.NET.153.164:1400
     22 212.179.48.2:41971        -> MY.NET.88.162:1214
     22 212.179.27.176:80         -> MY.NET.153.164:1768
     22 212.179.27.176:80         -> MY.NET.153.163:1096
     22 212.179.112.100:80        -> MY.NET.153.196:2788
     21 212.179.27.176:80         -> MY.NET.153.163:1217
     20 212.179.27.176:80         -> MY.NET.153.164:1763
     19 212.179.27.176:80         -> MY.NET.153.164:1399
     19 212.179.27.176:80         -> MY.NET.153.163:1092
     18 212.179.35.8:80           -> MY.NET.150.204:1337
     17 212.179.27.176:80         -> MY.NET.153.174:1148
     17 212.179.27.176:80         -> MY.NET.153.163:1097
     17 212.179.27.176:80         -> MY.NET.153.153:2022
     17 212.179.112.100:80        -> MY.NET.153.196:2828
     16 212.179.40.132:62801      -> MY.NET.150.143:4662
     16 212.179.112.100:80        -> MY.NET.153.196:2829
     16 212.179.112.100:80        -> MY.NET.153.196:2784
     15 212.179.48.2:36675        -> MY.NET.153.191:1214
     14 212.179.40.132:64207      -> MY.NET.150.143:4662
     14 212.179.27.176:80         -> MY.NET.153.164:1767
     13 212.179.40.132:64669      -> MY.NET.150.143:4662
     13 212.179.40.132:63484      -> MY.NET.150.143:4662
     13 212.179.40.132:62484      -> MY.NET.150.143:4662
     13 212.179.40.132:61568      -> MY.NET.150.143:4662
```

```
    13 212.179.27.176:80         -> MY.NET.153.164:1769
    13 212.179.27.176:80         -> MY.NET.153.163:1098
    12 212.179.40.132:64479      -> MY.NET.150.143:4662
    12 212.179.40.132:63070      -> MY.NET.150.143:4662
    12 212.179.40.132:62146      -> MY.NET.150.143:4662
    11 212.179.40.132:65087      -> MY.NET.150.143:4662
    11 212.179.40.132:64684      -> MY.NET.150.143:4662
    11 212.179.40.132:62823      -> MY.NET.150.143:4662
    11 212.179.40.132:61560      -> MY.NET.150.143:4662
    11 212.179.40.132:61009      -> MY.NET.150.143:4662
    10 212.179.40.132:64372      -> MY.NET.150.143:4662
    10 212.179.40.132:64335      -> MY.NET.150.143:4662
    10 212.179.40.132:64075      -> MY.NET.150.143:4662
    10 212.179.40.132:64064      -> MY.NET.150.143:4662
    10 212.179.40.132:64005      -> MY.NET.150.143:4662
    10 212.179.40.132:63821      -> MY.NET.150.143:4662
    10 212.179.40.132:63748      -> MY.NET.150.143:4662
    10 212.179.40.132:63713      -> MY.NET.150.143:4662
    10 212.179.40.132:63703      -> MY.NET.150.143:4662
    10 212.179.40.132:63675      -> MY.NET.150.143:4662
    10 212.179.40.132:63007      -> MY.NET.150.143:4662
    10 212.179.40.132:62867      -> MY.NET.150.143:4662
    10 212.179.40.132:62730      -> MY.NET.150.143:4662
    10 212.179.40.132:62654      -> MY.NET.150.143:4662
    10 212.179.40.132:62618      -> MY.NET.150.143:4662
    10 212.179.40.132:62536      -> MY.NET.150.143:4662
    10 212.179.40.132:62373      -> MY.NET.150.143:4662
    10 212.179.40.132:62280      -> MY.NET.150.143:4662
    10 212.179.40.132:62083      -> MY.NET.150.143:4662
    10 212.179.40.132:62026      -> MY.NET.150.143:4662
    10 212.179.40.132:61896      -> MY.NET.150.143:4662
    10 212.179.40.132:61661      -> MY.NET.150.143:4662
    10 212.179.40.132:61655      -> MY.NET.150.143:4662
    10 212.179.40.132:61603      -> MY.NET.150.143:4662
    10 212.179.40.132:61489      -> MY.NET.150.143:4662
    10 212.179.40.132:61392      -> MY.NET.150.143:4662
    10 212.179.40.132:61244      -> MY.NET.150.143:4662
    10 212.179.40.132:61174      -> MY.NET.150.143:4662
    10 212.179.35.8:80           -> MY.NET.150.204:1344
    10 212.179.35.8:80           -> MY.NET.150.204:1341
    10 212.179.27.176:80         -> MY.NET.153.174:1989
     9 212.179.48.2:41933        -> MY.NET.88.162:1214
     9 212.179.48.2:41845        -> MY.NET.88.162:1214
     9 212.179.40.132:64724      -> MY.NET.150.143:4662
     9 212.179.40.132:64144      -> MY.NET.150.143:4662
     9 212.179.40.132:64138      -> MY.NET.150.143:4662
     9 212.179.40.132:63781      -> MY.NET.150.143:4662
     9 212.179.40.132:63493      -> MY.NET.150.143:4662
     9 212.179.40.132:63067      -> MY.NET.150.143:4662
     9 212.179.40.132:62758      -> MY.NET.150.143:4662
     9 212.179.40.132:62610      -> MY.NET.150.143:4662
     9 212.179.40.132:62317      -> MY.NET.150.143:4662
```

```
      9 212.179.40.132:62211      -> MY.NET.150.143:4662
      9 212.179.40.132:61852      -> MY.NET.150.143:4662
      9 212.179.40.132:61752      -> MY.NET.150.143:4662
      9 212.179.40.132:61123      -> MY.NET.150.143:4662
      9 212.179.40.132:61023      -> MY.NET.150.143:4662
      9 212.179.40.132:61015      -> MY.NET.150.143:4662
      9 212.179.112.100:80        -> MY.NET.153.196:2787
      8 212.179.40.132:65044      -> MY.NET.150.143:4662
      8 212.179.40.132:64756      -> MY.NET.150.143:4662
      8 212.179.40.132:64624      -> MY.NET.150.143:4662
      8 212.179.40.132:63530      -> MY.NET.150.143:4662
      8 212.179.40.132:63457      -> MY.NET.150.143:4662
      8 212.179.40.132:63289      -> MY.NET.150.143:4662
      8 212.179.40.132:63264      -> MY.NET.150.143:4662
      8 212.179.40.132:62600      -> MY.NET.150.143:4662
      8 212.179.40.132:62292      -> MY.NET.150.143:4662
      8 212.179.40.132:61720      -> MY.NET.150.143:4662
      8 212.179.35.119:1214       -> MY.NET.153.163:1280
      8 212.179.35.119:1214       -> MY.NET.153.163:1272
      8 212.179.27.176:80         -> MY.NET.153.174:1158
      8 212.179.112.100:80        -> MY.NET.153.196:3085
      7 212.179.40.132:65090      -> MY.NET.150.143:4662
      7 212.179.40.132:64315      -> MY.NET.150.143:4662
      7 212.179.40.132:62951      -> MY.NET.150.143:4662
      7 212.179.40.132:62595      -> MY.NET.150.143:4662
      7 212.179.40.132:62309      -> MY.NET.150.143:4662
      7 212.179.40.132:61981      -> MY.NET.150.143:4662
      7 212.179.40.132:61853      -> MY.NET.150.143:4662
      7 212.179.35.119:1214       -> MY.NET.153.174:2074
      7 212.179.35.119:1214       -> MY.NET.153.174:2073
      7 212.179.35.119:1214       -> MY.NET.153.163:2171
      7 212.179.35.119:1214       -> MY.NET.153.163:1332
      7 212.179.35.119:1214       -> MY.NET.153.163:1281
      7 212.179.35.119:1214       -> MY.NET.153.163:1275
      7 212.179.35.119:1214       -> MY.NET.153.163:1270
      7 212.179.35.119:1214       -> MY.NET.153.163:1269
      7 212.179.35.119:1214       -> MY.NET.153.163:1268
      7 212.179.27.176:80         -> MY.NET.153.153:2021
      6 212.179.48.2:40483        -> MY.NET.88.162:1214
      6 212.179.40.132:64392      -> MY.NET.150.143:4662
      6 212.179.35.8:80           -> MY.NET.150.204:1343
      6 212.179.35.121:80         -> MY.NET.153.174:2025
      6 212.179.35.121:80         -> MY.NET.153.174:1982
      6 212.179.35.121:80         -> MY.NET.153.164:1639
      6 212.179.35.121:80         -> MY.NET.153.164:1483
      6 212.179.35.121:80         -> MY.NET.153.153:2020
      6 212.179.35.119:1214       -> MY.NET.153.164:1565
      6 212.179.35.119:1214       -> MY.NET.153.163:1273
      6 212.179.27.176:80         -> MY.NET.153.174:1990
      6 212.179.27.176:80         -> MY.NET.153.164:1517
      6 212.179.27.176:80         -> MY.NET.153.163:1224
      6 212.179.27.176:80         -> MY.NET.153.153:2023
```

```
      5 212.179.40.132:63485      -> MY.NET.150.143:4662
      5 212.179.40.132:63055      -> MY.NET.150.143:4662
      5 212.179.40.132:62449      -> MY.NET.150.143:4662
      5 212.179.40.132:62204      -> MY.NET.150.143:4662
      5 212.179.35.8:80           -> MY.NET.150.204:1342
      5 212.179.35.119:1214       -> MY.NET.153.174:1220
      5 212.179.35.119:1214       -> MY.NET.153.174:1188
      5 212.179.35.119:1214       -> MY.NET.153.164:1627
      5 212.179.35.119:1214       -> MY.NET.153.163:1321
      5 212.179.35.119:1214       -> MY.NET.153.153:2067
      5 212.179.27.176:80         -> MY.NET.153.174:2068
      5 212.179.27.176:80         -> MY.NET.153.174:1988
      5 212.179.27.176:80         -> MY.NET.153.164:1560
      5 212.179.27.176:80         -> MY.NET.153.163:1263
      5 212.179.27.176:80         -> MY.NET.153.163:1225
      4 212.179.5.90:2058         -> MY.NET.88.162:1214
      4 212.179.48.2:42111        -> MY.NET.88.162:1214
      4 212.179.48.2:42095        -> MY.NET.88.162:1214
      4 212.179.48.2:40439        -> MY.NET.88.162:1214
      4 212.179.44.99:61918       -> MY.NET.150.113:1214
      4 212.179.44.99:61384       -> MY.NET.150.113:1214
      4 212.179.44.99:61263       -> MY.NET.150.41:1214
      4 212.179.40.132:61109      -> MY.NET.150.143:4662
      4 212.179.35.97:80          -> MY.NET.152.14:1225
      4 212.179.35.121:80         -> MY.NET.153.163:1339
      4 212.179.35.121:80         -> MY.NET.153.163:1218
      4 212.179.35.119:1214       -> MY.NET.153.163:1282
      4 212.179.27.176:80         -> MY.NET.153.164:1649
      4 212.179.27.176:80         -> MY.NET.153.164:1647
      4 212.179.27.176:80         -> MY.NET.153.164:1646
      4 212.179.27.176:80         -> MY.NET.153.164:1643
      4 212.179.27.176:80         -> MY.NET.153.163:1223
      4 212.179.125.79:19711      -> MY.NET.150.133:1214
      3 212.179.44.99:61183       -> MY.NET.150.113:1214
      3 212.179.40.132:64139      -> MY.NET.150.143:4662
      3 212.179.40.132:63054      -> MY.NET.150.143:4662
      3 212.179.38.83:4581        -> MY.NET.150.133:1214
      3 212.179.38.83:4388        -> MY.NET.150.220:1214
      3 212.179.38.83:4357        -> MY.NET.150.133:1214
      3 212.179.38.83:4171        -> MY.NET.150.220:1214
      3 212.179.38.83:4095        -> MY.NET.150.133:1214
      3 212.179.38.83:3873        -> MY.NET.150.133:1214
      3 212.179.38.83:3437        -> MY.NET.150.220:1214
      3 212.179.35.119:1214       -> MY.NET.153.174:1190
      3 212.179.27.176:80         -> MY.NET.153.174:2038
      3 212.179.27.176:80         -> MY.NET.153.174:2037
      3 212.179.27.176:80         -> MY.NET.153.174:2033
      3 212.179.27.176:80         -> MY.NET.153.174:2032
      3 212.179.27.176:80         -> MY.NET.153.174:2031
      3 212.179.27.176:80         -> MY.NET.153.174:2030
      3 212.179.27.176:80         -> MY.NET.153.174:2029
      3 212.179.27.176:80         -> MY.NET.153.174:2022
```

```
      3 212.179.27.176:80         -> MY.NET.153.174:1157
      3 212.179.27.176:80         -> MY.NET.153.164:1659
      3 212.179.27.176:80         -> MY.NET.153.164:1656
      3 212.179.27.176:80         -> MY.NET.153.164:1650
      3 212.179.27.176:80         -> MY.NET.153.164:1645
      3 212.179.27.176:80         -> MY.NET.153.163:1365
      3 212.179.27.176:80         -> MY.NET.153.163:1364
      3 212.179.27.176:80         -> MY.NET.153.163:1363
      3 212.179.27.176:80         -> MY.NET.153.163:1358
      3 212.179.27.176:80         -> MY.NET.153.163:1357
      3 212.179.27.176:80         -> MY.NET.153.163:1356
      3 212.179.27.176:80         -> MY.NET.153.163:1349
      3 212.179.27.176:80         -> MY.NET.153.163:1348
      3 212.179.27.176:80         -> MY.NET.153.163:1336
      3 212.179.125.79:7768       -> MY.NET.153.191:1214
      3 212.179.125.79:7255       -> MY.NET.153.191:1214
      3 212.179.112.100:80        -> MY.NET.153.196:3087
      3 212.179.112.100:80        -> MY.NET.153.196:2827
      3 212.179.112.100:80        -> MY.NET.153.196:2826
      3 212.179.112.100:80        -> MY.NET.153.196:2793
      3 212.179.112.100:80        -> MY.NET.153.196:2789
      2 212.179.59.176:1437       -> MY.NET.150.133:1214
      2 212.179.40.132:64791      -> MY.NET.150.143:4662
      2 212.179.38.83:3897        -> MY.NET.150.220:1214
      2 212.179.38.83:3691        -> MY.NET.150.220:1214
      2 212.179.38.83:3656        -> MY.NET.150.133:1214
      2 212.179.38.75:1224        -> MY.NET.150.133:1214
      2 212.179.38.233:3240       -> MY.NET.150.220:1214
      2 212.179.38.233:3173       -> MY.NET.150.133:1214
      2 212.179.35.121:80         -> MY.NET.153.174:1151
      2 212.179.35.119:1214       -> MY.NET.153.174:1189
      2 212.179.27.176:80         -> MY.NET.153.174:2046
      2 212.179.27.176:80         -> MY.NET.153.174:2045
      2 212.179.27.176:80         -> MY.NET.153.174:2044
      2 212.179.27.176:80         -> MY.NET.153.174:2043
      2 212.179.27.176:80         -> MY.NET.153.174:2039
      2 212.179.27.176:80         -> MY.NET.153.164:1660
      2 212.179.27.176:80         -> MY.NET.153.164:1657
      2 212.179.27.176:80         -> MY.NET.153.164:1654
      2 212.179.27.176:80         -> MY.NET.153.164:1635
      2 212.179.27.176:80         -> MY.NET.153.164:1516
      2 212.179.27.176:80         -> MY.NET.153.163:1361
      2 212.179.27.176:80         -> MY.NET.153.163:1352
      2 212.179.27.176:80         -> MY.NET.153.163:1351
      2 212.179.112.100:80        -> MY.NET.153.196:3090
      2 212.179.112.100:80        -> MY.NET.153.196:3088
      2 212.179.112.100:80        -> MY.NET.153.196:3086
      2 212.179.112.100:80        -> MY.NET.153.196:2830
      1 212.179.63.55:61950       -> MY.NET.150.133:1214
      1 212.179.44.99:62016       -> MY.NET.150.41:1214
      1 212.179.38.163:1673       -> MY.NET.150.133:1214
      1 212.179.35.97:80          -> MY.NET.153.185:1279
```

```
      1 212.179.32.109:24048      -> MY.NET.153.191:1214
      1 212.179.27.176:80         -> MY.NET.153.164:1658
      1 212.179.27.176:80         -> MY.NET.153.164:1655
      1 212.179.27.176:80         -> MY.NET.153.163:1350
      1 212.179.112.100:80        -> MY.NET.153.196:3089
======== 4840  Total alerts
======== 255  Unique talkers
```