

Network Security

Gerald A. Marin

Goals:

- Help students achieve a good understanding of network security including:
 - Security principles
 - Attack fundamentals
 - Hacker methods
 - Network intrusion detection
 - Signature or "misuse" detection
 - Statistical anomaly detection
 - Network perimeter security

Course prerequisites

- ❑ One semester course in networking (or equivalent)
- ❑ Two semesters of Calculus
- ❑ One semester of probability/statistics

Responsibility: Do's and Don't

You will use what you learn in this course in a responsible, ethical and professional manner

- ❑ Don't use these tools against any system(s) unless you own them, meaning that the system is your personal property.
- ❑ If you're going to use these tools against a system other than your own, make certain you have explicit permission by an authorized officer of the target organization.
- ❑ Penalties may be severe for scanning a network without permission or other intrusive behavior. Laws and regulations you should be aware of are,
 - Computer Security Act
 - Patriot Act
 - Digital Millennium Copyright Act

Good papers to read,

- <http://www.witsa.org/papers/McConnell-cybercrime.pdf>
- http://www.asianlaws.org/cyberlaw/library/cc/cc_intlaw.pdf

Outline of Course Content - Security Principles

Confidentiality: only sender, intended receiver should "understand" message contents

- sender encrypts message
- receiver decrypts message

Authentication: sender, receiver want to confirm identity of each other

Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

Access Control and Availability: services must be accessible and available to intended users

Non-repudiation: sender should not be able to disavow later.

Outline Continued - Security Principles Topics Include:

- ❑ Principles of Cryptography
 - Symmetric key: DES and AES
 - Public key: RSA
- ❑ Authentication
 - Authentication protocols
 - Man-in-middle attack (replay)
- ❑ Integrity
 - Digital signatures and message digests
- ❑ Key Distribution and Certification
 - Roles of key distribution centers and certification authorities.

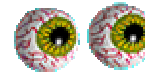
Outline Continued - Traffic Analysis and Filtering

- ❑ Using Ethereal and TCPDUMP
- ❑ Examining IP Header Fields
- ❑ Examining TCP Header Fields
- ❑ Recognizing Attacks
- ❑ Examining Embedded Protocol Header Fields
- ❑ Real-world Analysis
- ❑ Writing TCPDUMP Filters

Outline Continued - Intrusion Detection Systems

- ❑ Overview
- ❑ The False Alarm Problem
- ❑ Introduction to SNORT
- ❑ SNORT Filtering Rules
- ❑ Real-world Analysis

Network Reconnaissance: A Hackers Perspective



Scanners/Fingerprinting/Sniffers

Agenda

- Background/Overview: Tools of the Trade
 - Scanners
 - What is a scanner?
 - Types of scanners
 - Examples of scanners
 - OS Fingerprinting tools
 - What is OS Fingerprinting
 - Types of fingerprinting
 - Examples of OS fingerprinting tools
 - Sniffers/Protocol Analyzers:
 - What is a Sniffers/Protocol Analyzer?
 - Types of Sniffers/Protocol Analyzers
 - Examples of sniffers
- A Detailed view
 - Analyzing the Tools and the traffic they generate

Background: Scanners

- ❑ What is a scanner?
 - A scanner, in network terms, is a program that uses a network to analyze responses from a given set of targets and returns information based on certain criteria.
- ❑ Types of scanners include
 - IP Scanners: Returns a list of active IPs
 - Superscan , [NMAP](#)
 - Port/ Service scanners: Returns what ports are open on a target and what services are being provided on it.
 - [IPTools](#), NMAP
 - Vulnerability Scanners: Returns a list of exploits which the target might be vulnerable to.
 - Nessus, [Retina](#)
 - NAT Scanners: attempts to determine the number of systems running behind the natted firewall and their operating systems
 - firewalk

Background: OS Fingerprinting Tools

- What is OS fingerprinting?
 - A technique that queries the TCP/IP Stack of a host to determine what operating system is running on it.

- There are two different types of OS fingerprinting tools,
 - Active:
 - Generates network traffic
 - May be detected
 - Specially crafted packets
 - Catches variability's in TCP/IP stack
 - Passive:
 - No traffic is generated
 - Virtually undetectable

Background: OS Fingerprinting

Tools

Today's tools

- Xprobe2, by Ofir Arkin, <http://www.sys-security.com>
 - Uses ICMP as the method to do fingerprinting
 - Generates fingerprints of systems scanned
- NMAP, by Fyodor, <http://www.isecure.org>
 - detects 100's of different OS versions and network devices
 - By far the most sophisticated fingerprinting tool on the net
 - IP/Service scanner
 - Portscanner
 - OS fingerprinting
 - Network Device fingerprinting
 - 12 different modes of scanning
 - 4 different ways to discover systems
 - 8 different packet modification options (you can select multiple options simultaneously)
 - 6 different timing options and 6 different method of detecting

Background: OS Fingerprinting Tools

Today's tools

Passive Tools

- Siphon, by Subterrain Security Group
 - <http://www.blackhat.com/presentations/bh-usa-01/AbadBeddoe/1>
 - Runs as a service and logs detected operating systems to a file and stdout
- POf, by Michal Zalewski
 - <http://lcamtuf.coredump.cx/pOf.shtml>
 - Analyzes tcpdump formatted files
 - Excellent tool for network analysis to use with windump and tcpdump

Resources

Web Sites

- ❑ <http://www.isecure.org>
- ❑ <http://www.sys-security.com>
- ❑ <http://securify.packetstorm.org>
- ❑ <http://www.protocols.com>
- ❑ <http://www.sans.org>
- ❑ <http://www.networksorcery.com/enp/default0601.htm>

Books

- ❑ Network Intrusion Detection, Northcutt
- ❑ TCP/IP Illustrated vol1, Stevens

Outline Continued - Statistical Anomaly Detection

- ❑ Characterizing Normal/Abnormal Traffic
 - Protocol distributions
 - Address distributions
 - Port distributions
 - Protocol State Violation
- ❑ Statistical Distributions of Protocols
- ❑ Self-Similarity and Fractal Dimension
- ❑ Establishing RoC Characteristics
- ❑ Clustering techniques
- ❑ Key Papers

Outline Continued - Perimeter Defense

- ❑ Router filtering
- ❑ Firewalls
- ❑ Honeypots
- ❑ DMZs and Screened Subnets

Overview: Traffic Analysis and Filtering

Network Security Course
Gerald A. Marin

TCPDump

- ❑ UNIX tool that collects network data and displays it in specified format.
- ❑ It may be run "live" on a specified interface - **but only if authorized.**
- ❑ It may read data from a file that has previously been saved using TCPDump.
- ❑ It offers a number of filtering capabilities.
- ❑ Must be downloaded with libcap or the windows equivalent. (Do this by next class!)

Man Page

- ❑ If not on unix system go to <http://www.rt.com/man/tcpdump.1.html>
- ❑ **NAME** tcpdump – dump traffic on a network
SYNOPSIS tcpdump [- adeflnNOpqStvx] [-c *count*] [-F *file*] [-i *interface*] [-r *file*] [-s *snaplen*] [-T *type*] [-w *file*] [*expression*]
DESCRIPTION *Tcpdump* prints out the headers of packets on a network interface that match the boolean *expression*.

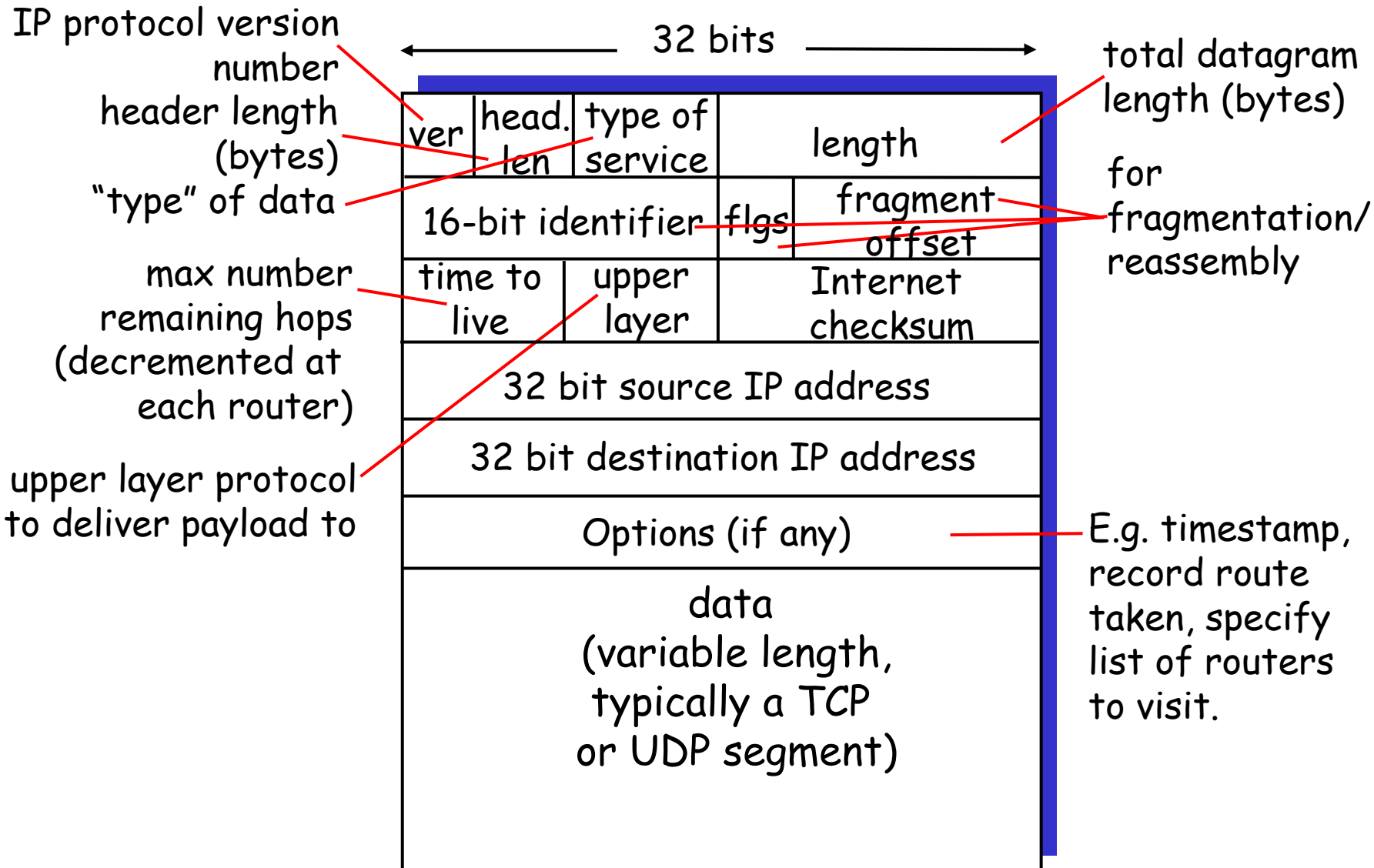
TCPDump Traffic Capture

```
00:28:24.573542 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 536784:536912(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.573652 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 536912 win 63376 (DF)
00:28:24.573676 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 536912:537072(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.573796 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537072:537232(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.573918 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537232:537392(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.573912 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 537232 win 63056 (DF)
00:28:24.574032 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537392:537520(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574143 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 537520 win 64240 (DF)
00:28:24.574166 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537520:537680(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574276 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537680:537840(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574375 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537840:537968(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574392 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 537840 win 63920 (DF)
00:28:24.574513 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 537968:538128(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574634 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 538128:538288(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574629 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 538128 win 63632 (DF)
00:28:24.574758 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 538288:538416(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.574868 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 538416 win 63344 (DF)
00:28:24.575000 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 538576:538736(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.575099 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 538736:538864(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.575234 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 538864:539024(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.575340 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 539024:539184(160) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.575470 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 539184:539312(128) ack 7073 win 19872 (DF) [tos 0x10]
00:28:24.575581 163.118.231.25.3197 > blackwidow.se.fit.edu.ssh: . ack 539312 win 63952 (DF)
00:28:24.575713 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 539472:539600(128) ack 7073 win 19872 (DF) [tos 0x10]
```

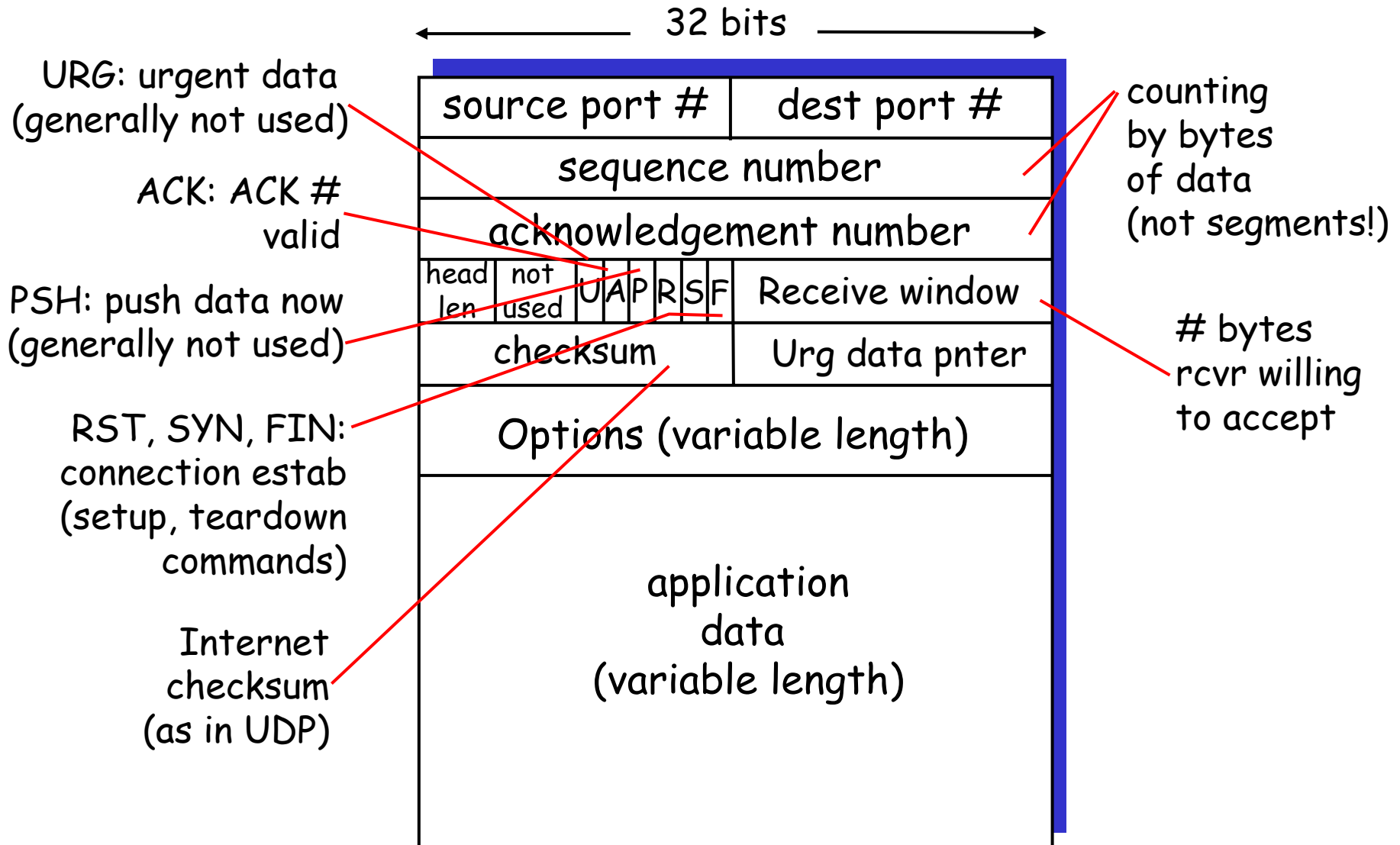
TCPDump Traffic Capture cont...

- 00:28:24.573542 blackwidow.se.fit.edu.ssh > 163.118.231.25.3197: P 536784:536912(128) ack 7073 win 19872 (DF) [tos 0x10]
 - 00:28:24.573542 - time packet was received
 - blackwidow.se.fit.edu.ssh - source host and port. In this case the port is SSH or 22
 - > - direction of the traffic
 - 163.118.231.25.3197 - destination IP and port
 - P - flag set, in this case is P for push. Pushes data from the sending host to the receiving host
 - 536784:536912 - beginning and ending sequence numbers. This is used to order the data received.
 - (128) - bytes in the packet
 - ack 7073- TCP flag, ACK represents the acknowledgement of data received. The 7073 is the acknowledgement number
 - Win 19872 - this is the windows size. This means that the client has a window size or incoming buffer of 19872 bytes.
 - (DF) - don't fragment.
 - [tos 0x10] - type of service. This this case is 10 which stands for minimize delay

IP datagram format



TCP segment structure



Absolute and Relative Seq Nos

□ Consider the following:

- client.com.38060 > telnet.com.telnet: S
3774957990:3774957990(0) win 8760 <mss 1460> (DF)
- telnet.com.telnet > client.com.38060: S
2009600000:2009600000(0) ack 3774957991 win 1024
<mss 1460>
- client.com.38060 > telnet.com.telnet: . ack 1 win 8760
(DF)
- client.com.38060 > telnet.com.telnet: P 1:28(27) ack 1 win
8760 (DF)

□ Note use of relative sequence numbers beginning with 3rd packet.

Ethereal Traffic Capture

The screenshot displays the 'The Ethereal Network Analyzer' application window. At the top, there is a menu bar with 'File', 'Edit', 'Capture', 'Display', 'Tools', and 'Help'. Below the menu bar is a table listing captured packets:

No. .	Time	Source	Destination	Protocol	Info
1	0.000000	163.118.134.11	163.118.231.25	UDP	Source port: 49015 Destination port: 3023
2	2.047481	163.118.231.25	163.118.134.1	DNS	Standard query PTR 1.134.118.163.in-addr.arpa
3	2.049967	163.118.134.1	163.118.231.25	DNS	Standard query response PTR redwidow.se.fit.edu
4	35.703787	163.118.134.10	163.118.231.25	NBSS	NBSS Continuation Message
5	35.703850	163.118.231.25	163.118.134.10	TCP	3225 > microsoft-ds [ACK] Seq=2515682704 Ack=3197936079 win=65394 Len

Below the table, the details for the selected packet (Frame 1) are shown:

- Frame 1 (60 bytes on wire, 60 bytes captured)
 - Arrival Time: oct 21, 2003 01:19:24.824793000
 - Time delta from previous packet: 0.000000000 seconds
 - Time relative to first packet: 0.000000000 seconds
 - Frame Number: 1
 - Packet Length: 60 bytes
 - Capture Length: 60 bytes
- Ethernet II, Src: 00:e0:52:92:31:00, Dst: 00:0d:61:02:b5:3a
 - Destination: 00:0d:61:02:b5:3a (Giga-Byt_02:b5:3a)
 - Source: 00:e0:52:92:31:00 (FoundryN_92:31:00)
 - Type: IP (0x0800)
 - Trailer: 00000000000000000000
- Internet Protocol, Src Addr: 163.118.134.11 (163.118.134.11), Dst Addr: 163.118.231.25 (163.118.231.25)
 - Version: 4
 - Header length: 20 bytes
 - Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
 - 0000 00.. = Differentiated Services Codepoint: Default (0x00)
 -0. = ECN-Capable Transport (ECT): 0
 -0 = ECN-CE: 0
 - Total Length: 36
 - Identification: 0x8726 (34598)
 - Flags: 0x00
 - Fragment offset: 0
 - Time to live: 126
 - Protocol: UDP (0x11)
 - Header checksum: 0x0191 (correct)
 - Source: 163.118.134.11 (163.118.134.11)
 - Destination: 163.118.231.25 (163.118.231.25)
- User Datagram Protocol, Src Port: 49015 (49015), Dst Port: 3023 (3023)
 - Source port: 49015 (49015)
 - Destination port: 3023 (3023)
 - Length: 16
 - Checksum: 0xba8d (correct)
 - Data (8 bytes)

At the bottom, a hex dump of the packet data is shown:

```
0000 00 0d 61 02 b5 3a 00 e0 52 92 31 00 08 00 45 00  .a...R.l...E.
0010 00 24 87 26 00 00 7e 11 01 91 a3 76 86 0b a3 76  $.&...v...v
0020 e7 19 bf 77 0b cf 00 10 ba 8d 50 28 b1 02 cb 44  ...w...P(...D
0030 f9 77 00 00 00 00 00 00 00 00 00 00  .w.....
```

The bottom of the window features a 'Filter' field with the text 'Ethernet (eth), 14 bytes' and buttons for 'Reset' and 'Apply'.

Ethereal Traffic Capture

The screenshot displays a single captured packet in Wireshark. The packet list pane shows 'Frame 1 (60 bytes on wire, 60 bytes captured)'. The packet details pane is expanded to show the following layers:

- Ethernet II, Src: 00:e0:52:92:31:00, Dst: 00:0d:61:02:b5:3a
- Internet Protocol, Src Addr: 163.118.134.11, Dst Addr: 163.118.231.25
- Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
- Total Length: 36
- Identification: 0x8726 (34598)
- Flags: 0x00
- Protocol: UDP (0x11)
- User Datagram Protocol, Src Port: 49015, Dst Port: 3023

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000  00 0d 61 02 b5 3a 00 e0 52 92 31 00 08 00 45 00  ..a... R.1...E.
0010  00 24 87 26 00 00 7e 11 01 91 a3 76 86 0b a3 76  .$.&... ..V...V
0020  e7 19 bf 77 0b cf 00 10 ba 8d 50 28 b1 02 cb 44  ...w.....P(...D
0030  f9 77 00 00 00 00 00 00 00 00 00 00  .w.....
```

TCP 3-way Handshake

- ❑ tclient.net.39904 > telnet.com.23: S
733381829:733381829(0) win 8760 <mss
1460> (DF)
- ❑ telnet.com.23 > tclient.net.39904: S
1192930639:1192930639(0) ack
733381830 win 1024 <mss 1460> (DF)
- ❑ tclient.net.39904 > telnet.com.23: . Ack 1
win 8760 (DF)

TCP Takedown

- ❑ tclient.net.39904 > telnet.com.23: F 14:14(0) ack 186 win 8760 (DF)
- ❑ telnet.com.23 > tclient.net.39904: . ack 15 win 1024 (DF)
- ❑ Server initiates a FIN and client acks to finally close the connection.
- ❑ Abrupt version uses reset:
 - tclient.net.39904 > telnet.com.23: R 28:28(0) ack 1 8760 (DF)

Rudimentary Analysis

- ❑ Was the three-way handshake completed between two hosts?
- ❑ Were data transmitted?
- ❑ Who began and/or ended the connection?
- ❑ Recall Syn Flood (Neptune) Attack

SYN Flood (Neptune)

- ❑ Leverages TCP 3-way Handshake
- ❑ Attacker sends opening "SYN"
- ❑ Target responds with "SYN/ACK" and builds a record in a data structure to hold connection information
- ❑ The attack consists of many SYN packets being sent from unreachable sources (non-existent) so that handshake is not completed and data structure overflows.

Observations

- ❑ No sure way to filter at single packet level
- ❑ Characteristics:
 - Unusually large number of TCP SYNs directed at a single destination address
 - Unusually large number of destination unreachable responses to SYN/ACKs
 - Unusual source address patterns

Ack Scan (page 39 of NID)

- ❑ Attacker sends lone ack to probe specific ports
 - Live hosts respond with reset to unexpected ack.
 - May be used by hacker to determine location of live hosts.

- ❑ Note that lone ack should be found as follows:
 - Final transmission of 3-way handshake
 - Acknowledgement of received data or data in progress
 - Acknowledgement of received FIN
 - Do you see evidence of any such normal use?

TCP Session Hijacking

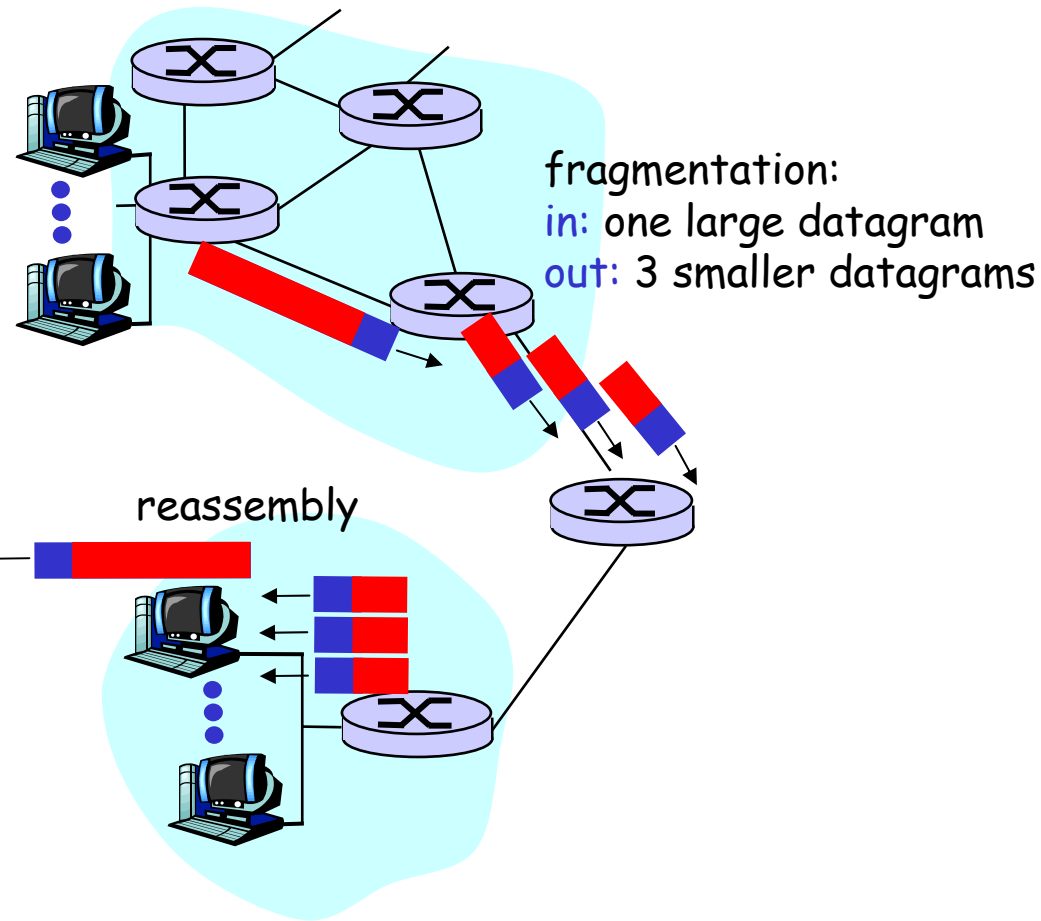
- ❑ Objective is to intercept an established TCP session and capture (impersonate) one end of the connection.
- ❑ Nontrivial effort that must maintain:
 - IP number
 - Established port numbers
 - Proper sequence number increments
 - Proper ack increments.

Fragmentation

- ❑ Fragmentation allows an IPV4 datagram to cross a network that has an MTU smaller than the IP datagram.
 - Recall that MTU is the max payload of the link layer frame.
 - Fragment ID
 - Offset number (13 bits)
 - Fragment Length
 - More Fragments Flag

IP Fragmentation & Reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame.
 - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
 - one datagram becomes several datagrams
 - "reassembled" only at final destination
 - IP header bits used to identify, order related fragments



Ping O' Death

- ❑ ICMP Echo request is sent with an illegally long payload (greater than 64k bytes).
- ❑ Older attack that could cause operating systems to lock or reboot.
 - Similar in effect to LAND attack
- ❑ Observation: Look closely at any ICMP packet that has been fragmented.

Analysis

- ❑ Mal.com.139 > target.net.139: udp 28 (frag 242:36@0+)
- ❑ Mal.com.139 > target.net.139: (frag 242:4@24)

Notice 36 data bytes in first fragment beginning at 0. Next are 4 bytes beginning at 24. Illegal overlap known as Teardrop attack.

Teardrop

- ❑ IPV4 packets support fragmentation, but fragments not permitted to overlap.
- ❑ In this attack packets are created with illegal overlap of fragments.
- ❑ Older operating systems may crash upon receipt of such fragments.
- ❑ Observation: Can check all arriving packets for illegal fragmentation.
 - Requires some state be maintained (previous termination point for this src,dest,ID)

RFC 792

- Occasionally a gateway or destination host will communicate with a source host, for example, to report an error in datagram processing. For such purposes this protocol, the Internet Control Message Protocol (ICMP), is used. ICMP, uses the basic support of IP as if it were a higher level protocol, however, ICMP is actually an integral part of IP, and must be implemented by every IP module.

RCF792 Continued

- ❑ ICMP messages are sent in several situations: for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route.
- ❑ The ICMP messages typically report errors in the processing of datagrams. To avoid the infinite regress of messages about messages etc., no ICMP messages are sent about ICMP messages. Also ICMP messages are only sent about errors in handling fragment zero of fragmented datagrams.

Overview of Normal ICMP Msgs

❑ Host Unreachable

- Router > sending.host: icmp: host target.host unreachable

❑ Port Unreachable

- Target.host > sending.host: icmp: target.host udp port ntp unreachable (DF)

❑ Admin Prohibited

- Router > sending.host: icmp: host target.host unreachable
- admin prohibited

❑ Need to frag

- Router > sending.host.net: icmp: target.host unreachable
- need to frag (mtu 1500)

❑ Others...

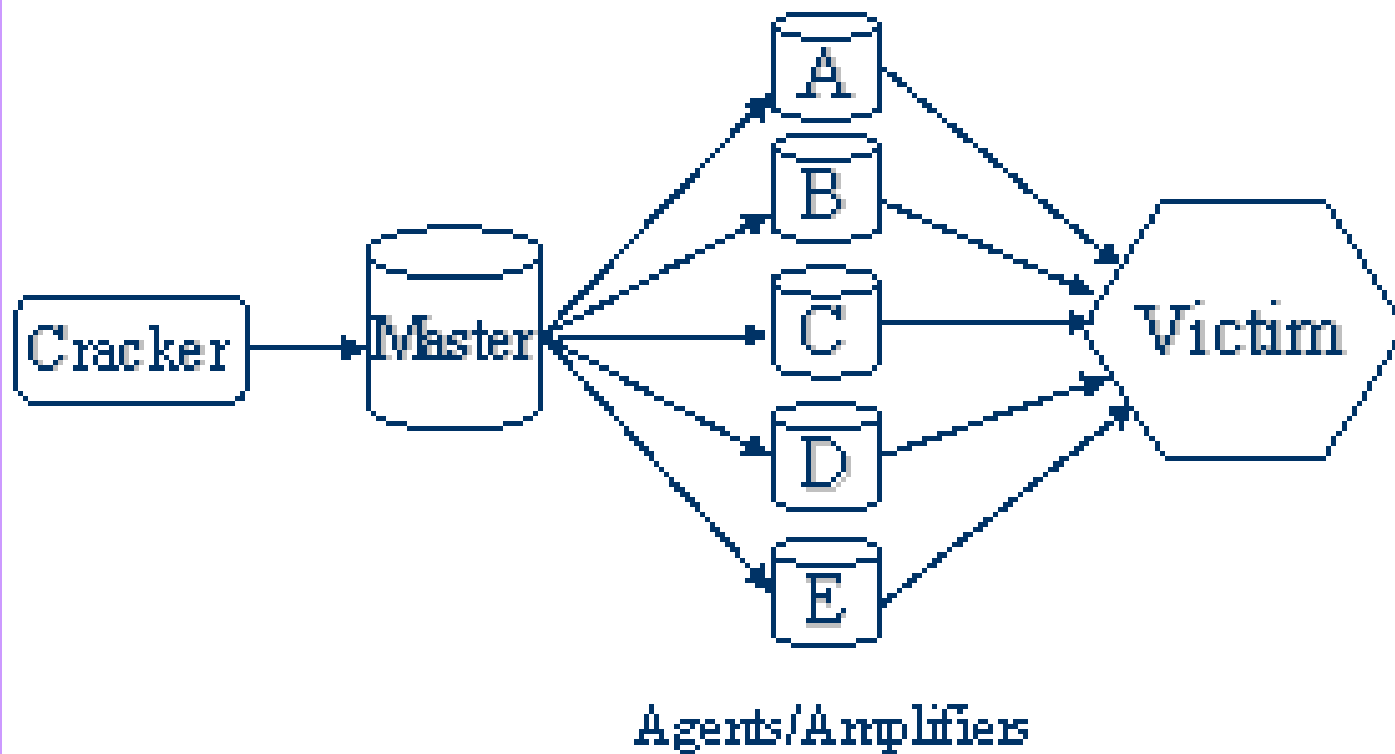
SMURF Attack

- ❑ Attacker spoofs its IP address to be that of target.
- ❑ Sends ICMP echo request to x.y.255.255
- ❑ Potentially thousands of machines in the x.y destination network may respond with ICMP echo replies
- ❑ Responses will all go to spoofed address of target.

Observations

- ❑ No way to observe attack based on a single packet.
- ❑ Target may suddenly observe all bandwidth being consumed at its network interface.
- ❑ Network monitoring may observe an unusually large number of ICMP echo response packets (especially directed at one target).
- ❑ Some systems now block all ICMP packets or block all packets destined to ...255 address.

Distributed Denial Of Service



Must Reading

Distributed Reflection Denial of Service

Description and analysis of a potent, increasingly prevalent, and worrisome Internet attack

By Steve Gibson of GRC

- <http://www.grc.com/dos/drDOS.htm>

Tribal Flood Network Attack

- ❑ Requires master and daemon hosts to be established.
- ❑ Master instructs daemons by sending commands in ICMP echo replies.
 - The ICMP identification number field in the ICMP header of the ICMP echo reply is used to direct daemons with args provided in ICMP data portion.

WinFreeze Attack

- ❑ Takes advantage of ICMP redirect message which informs a sending host that it has tried to use a nonoptimal router and directs the adding of more optimal router to host's table.
- ❑ Router > victim.com: icmp: redirect 243.148.16.61 to host victim.com
- ❑ Router > victim.com: icmp: redirect 110.161.152.156 to host victim.com
- ❑ Router > victim.com: icmp: redirect 245.211.87.115 to host victim.com
- ❑ Router > victim.com: icmp: redirect 49.130.233.15 to host victim.com
- ❑ ...host attacks itself

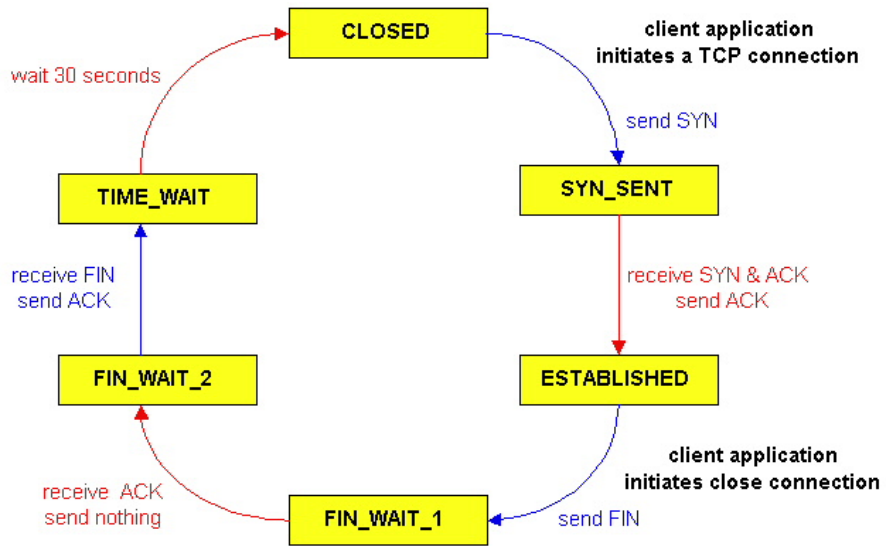
Loki

- ❑ Prior to Loki ICMP used for DoS attacks and network mapping.
- ❑ Loki uses ICMP as a tunneling protocol for a covert channel.
 - Loki server must be installed in a compromised host.
 - ICMP carries covert messages to the Loki server.
- ❑ More information at www.phrack.com issue 49 article 6.

Why not block ICMP?

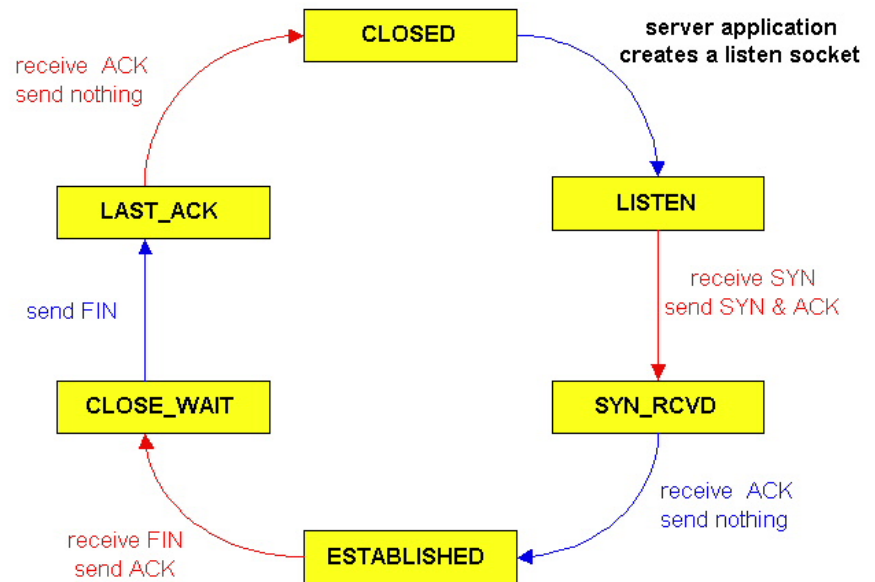
- ❑ You cannot then issue pings for your own diagnostic purposes. If you allow them outbound, then you still are vulnerable to echo-replies that are inbound.
- ❑ You cannot use the windows trace route utility. Unix uses UDP.
- ❑ Don't get any of the info messages that routers attempt to send with ICMP.
- ❑ Can't use MTU discovery because won't receive the "need to frag."

TCP Connection Management (cont)



TCP client lifecycle

TCP server lifecycle



From RFC 793: TCP/IP

Review principal state diagram: Figure 6.
Then...

Reset Generation As a general rule, reset (RST) must be sent whenever a segment arrives which apparently is not intended for the current connection. A reset must not be sent if it is not clear that this is the case. There are three groups of states:

1. If the connection does not exist (CLOSED) then a reset is sent in response to any incoming segment except another reset. In particular, SYNs addressed to a non-existent connection are rejected by this means. If the incoming segment has an ACK field, the reset takes its sequence number from the ACK field of the segment, otherwise the reset has sequence number zero and the ACK field is set to the sum of the sequence number and segment length of the incoming segment. The connection remains in the CLOSED state.

Moral of the story

- ❑ RFCs describe how TCP/IP is supposed to work.
 - Available at www.ietf.org
- ❑ Hackers know that different TCP/IP implementations react differently to protocol violations.
- ❑ Hackers can also use normal responses to find out which ports are listening (services are available to exploit).

Analyze this

❑ Router.com>1.2.10.72: icmp: time exceeded in-transit

❑ Router.com>1.2.18.13: icmp: time exceeded in-transit

❑ Router.com>1.2.11.67: icmp: time exceeded in-transit

❑ Router.com>1.2.16.13: icmp: time exceeded in-transit...

Ex: Unexpected Responses

- ❑ Router.com>1.2.10.72: icmp: time exceeded in-transit
- ❑ Router.com>1.2.18.13: icmp: time exceeded in-transit
- ❑ Router.com>1.2.11.67: icmp: time exceeded in-transit
- ❑ Router.com>1.2.16.13: icmp: time exceeded in-transit...
- ❑ Note all "responses" from Router.com but no traffic sent from the 1.2. Network.
- ❑ Can't be surveillance of 1.2 network because no responses to ICMP traffic.
- ❑ Most likely explanation is traffic sent to Router.com by someone spoofing the 1.2 network.
 - Sometimes such traffic is called "backscatter."

READ: Ref: Inferring Internet DoS Activity

Paper by Moore, Voelker, and Savage in 2001 (www.cs.ucsd.edu/~savage/papers/UsenixSec01.pdf).

- Introduces "Backscatter Analysis" being used in a Class A Network to project worldwide DoS attack numbers.
- Observed more than 12,000 attacks against more than 5,000 targets.
- Found that 90-94% of attacks are TCP based followed by UDP and ICMP.
- Obtained many other characteristics including types, rates, durations of attacks.

The SANS Practicals

- ❑ Based on work done by candidates for the Intrusion Detection Professional Certification from the Global Incident Analysis Center.
- ❑ Standard Analysis Pattern:
 - Network or system log trace of event of interest.
 - Source of the detect - such as SNORT.
 - Probability that source address was spoofed.
 - Description of attack
 - Attack mechanism
 - Correlation
 - Evidence of active targeting.

Practical Assignment

- Go to: <http://www.giac.org/GCIA.php>
 - Also see www.giac.org/webcasts/
- Assignment 3. In class on 2/3/04 each student to announce practical choice and topics it contains. Presentations will be 2/12 and 2/17.
 - 15-20 minutes each.

Essential Concepts of Probability and Statistics

Gerald A. Marin

Principal Topics

- ❑ Continuous random variables (focus)
- ❑ CDFs and PDFs
 - Exponential and Poisson distributions
 - Hypoexponential distribution
 - Erlang distribution
 - Gamma function
 - Normal distribution
 - Uniform distribution
- ❑ Joint distributions and independence
- ❑ Moments
- ❑ Goodness-of-fit (Chi-square and Kolmogorov-Smirnov)
- ❑ Introduction to Stochastic Processes
- ❑ Covariance and autocorrelation

Kolmogorov-Smirnov

We begin with a random sample of n values from the random variable and arrange them in increasing order so that

$$x_1 \leq x_2 \leq \dots \leq x_n.$$

The empirical distribution function based on this sample is:

$$F_n(x) = \begin{cases} 0, & x < x_1 \\ \frac{i}{n}, & x_i \leq x < x_{i+1} \\ 1, & x_n \leq x. \end{cases} \quad (\text{Alternatively, } F_n(x) \text{ can be thought$$

of as [the number of sample points $\leq x$] divided by n).

We measure the deviation of this empirical distribution function from the hypothesized distribution using the difference:

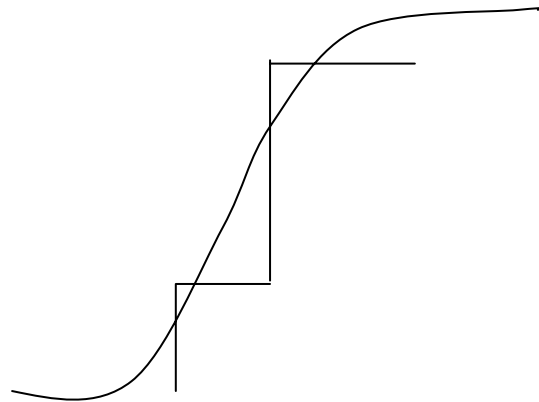
$d_n(x) = |F_n(x) - F_0(x)|$. We know F_0 so we can compute this difference for each sample value. The largest of these difference values is an indication of how well $F_n(x)$ approximates $F_0(x)$.

The K-S Statistic

The Kolmogorov-Smirnov statistic is defined as

$$D_n = \sup_x |F_n(x) - F_0(x)|, \text{ where sup means supremum and}$$

represents the "least upper bound" obtained over all possible values of x . Because $F_0(x)$ is non-decreasing and the values of $F_n(x)$ are constant over each interval $x_i \leq x < x_{i+1}$, the sup will occur at one of the end points so D_n is found as a simple max of a finite number of values.



Note. The exact distribution of this statistic is known even for small values of n because it does not depend on the assumed "true" distribution.

Example

Ten values are given below. We wish to test the null hypothesis that the population distribution function is $F_X(x) = 1 - e^{-\lambda x^2}$, x real.

(Known as the Weibull distribution with shape parameter 2.)

1.8453 0.5616 1.6178 2.6884 1.7416

0.7111 1.5430 1.5831 0.5688 0.8961

First order: 0.5616, 0.5688, 0.7111, 0.8961, 1.5430, 1.5831, 1.6178, 1.7416, 1.8453, 2.6884. The average is: 1.3757 and the mean of this Weibull

distribution is: $\left(\frac{1}{\lambda}\right)^{\frac{1}{2}} \Gamma\left(1 + \frac{1}{2}\right) = \left(\frac{1}{\lambda}\right)^{\frac{1}{2}} \left(\frac{\pi}{2}\right)$. Thus, we estimate $\lambda = 1.304$.

K-S Procedure

From the data we estimate $F(0.5688) = \frac{2}{10}$ and from the Weibull-2 we estimate $F_w(0.5688) = 0.3442$. This gives $d_2 = |0.3442 - 0.2| = 0.1442$.

Procedure: Compute d_1, d_2, \dots, d_{10} from the data and compare largest $D_{10} = \sup_i d_i$ with the theoretical value of the K-S statistic.

The computed value of the D_{10} statistic is 0.455. From a table we find the theoretical value $D_{10,0.05} = 0.41$. (There is only a 5% chance of getting a difference larger than 0.41 if the sample comes from the hypothesized distribution.) Thus we reject the null hypothesis.

Homework

1. The number of busy ports in a telephone switch were the following:

<i>Number busy</i>	<i>Observed frequency</i>	<i>Number busy</i>	<i>Observed frequency</i>
0	0	12	413
1	5	13	358
2	14	14	219
3	24	15	145
4	57	16	109
5	111	17	57
6	197	18	43
7	278	19	16
8	378	20	7
9	418	21	8
10	461	22	3
11	433		

Test whether the corresponding theoretical distribution is Poisson.

Homework

Observed times between successive crashes of a computer system were noted for a 6-month period as follows (time in hours):

1 10 20 30 40 52 63 70 80 90 100 102 130 140 190
210 266 310 530 590 640 1340.

Using the D_n statistic, test goodness of fit against, first, an exponential distribution and, second, against a normal distribution. The value of $D_{22,0.1}$ is 0.2512 (approx).

Network Monitoring

G. A. Marin

Major Topics

- ❑ Modeling Network Traffic
- ❑ Visualizing Network Traffic
- ❑ Detection Based on Arrivals
- ❑ Fractal Dimension and Detection
- ❑ Activity Profiling and Clustering
- ❑ Viruses and Worm

Generally Accepted Models

Protocol	Variables	Model
TELNET	Session arrivals	Poisson
	Originator bytes	Log2-extreme
	Responder bytes	Log2-normal
	Conn size (in packets)	Log2-normal
SMTP	Session arrivals	Poisson
	Originator bytes	Log2-normal
FTP	Session arrivals	Poisson
	Connection bytes	Log2-normal
	Burst Bytes	Pareto
	Session bytes	Log2-normal

Number of Originator Bytes in Telnet Session

A random variable, X , is said to have a log-f distribution if $\log(X)$ has an f distribution. In this case, if we take $\log_2 = \lg$ of the number of originator bytes collected in Telnet sessions, these values often have an "extreme" distribution. A random variable Y has an extreme distribution if its density is given by:

$$f(x) = \frac{1}{\beta} e^{-\frac{x-\alpha}{\beta}} e^{-e^{-\frac{x-\alpha}{\beta}}}$$

Note. If X has a Weibull distribution (next) with scale parameter α and shape parameter γ , then $\ln X$ has an extreme value distribution with $\mu = \ln \alpha$ and $\beta = \frac{1}{\gamma}$.

The distribution often referred to as the **Extreme Value Distribution (Type I)** is the limiting distribution of the minimum of a large number of unbounded identically distributed random variables.

Burst of Bytes in FTP Session

A random variable, X , has a Pareto distribution if its density

is given by $f(x) = \frac{\alpha k^\alpha}{x^{\alpha+1}}$ for $x > 0$. The cdf is given by

$F(x) = 1 - \left(\frac{k}{x}\right)^\alpha$, for $x \geq k$ with α and k positive real numbers.

Mean is $\frac{\alpha k}{\alpha - 1}$ and variance is $\frac{\alpha k^2}{(\alpha - 1)^2 (\alpha - 2)}$.

FTP Example

The number of bytes in an FTP session burst has a Pareto distribution with mean 50 and variance 100. What is the probability that the number of bytes will be less than or equal to 75?

The difficulty in using the Pareto formula is that the mean is

given by $\mu = \frac{\alpha k}{\alpha - 1}$ and the variance by $\sigma^2 = \frac{\alpha k^2}{(\alpha - 1)^2 (\alpha - 2)}$. Thus,

we need to find α and k . Let $r = \frac{\sigma^2}{\mu} = \frac{k}{(\alpha - 1)(\alpha - 2)}$. Then

the defn of μ implies that $k = \frac{\mu(\alpha - 1)}{\alpha}$, and it follows that

$r = \frac{\mu(\alpha - 1)}{\alpha(\alpha - 1)(\alpha - 2)} = \frac{\mu}{\alpha(\alpha - 2)}$. Thus, $r\alpha^2 - 2r\alpha - \mu = 0$, and the

quadratic formula yields $\alpha = 1 + \frac{1}{\sigma} \sqrt{\mu^2 + \sigma^2}$.

FTP Example Continued

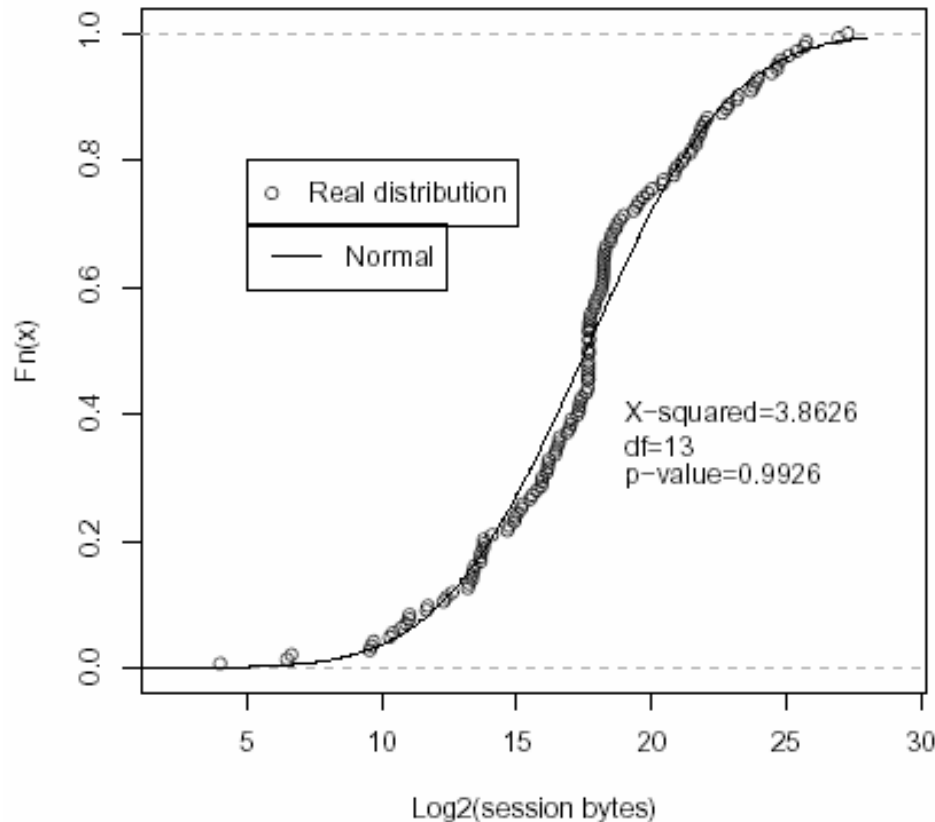
In this case we have $\alpha = 1 + \frac{1}{100} \sqrt{2600} = 1.510$ and

$k = \frac{50(.510)}{1.510} = 16.884$. It follows that

$$F(75) = 1 - \left(\frac{16.884}{75} \right)^{1.510} = 0.895.$$

Number of Bytes in FTP Session NOT Bytes in a Burst

CDF of FTP session bytes on Feb.05

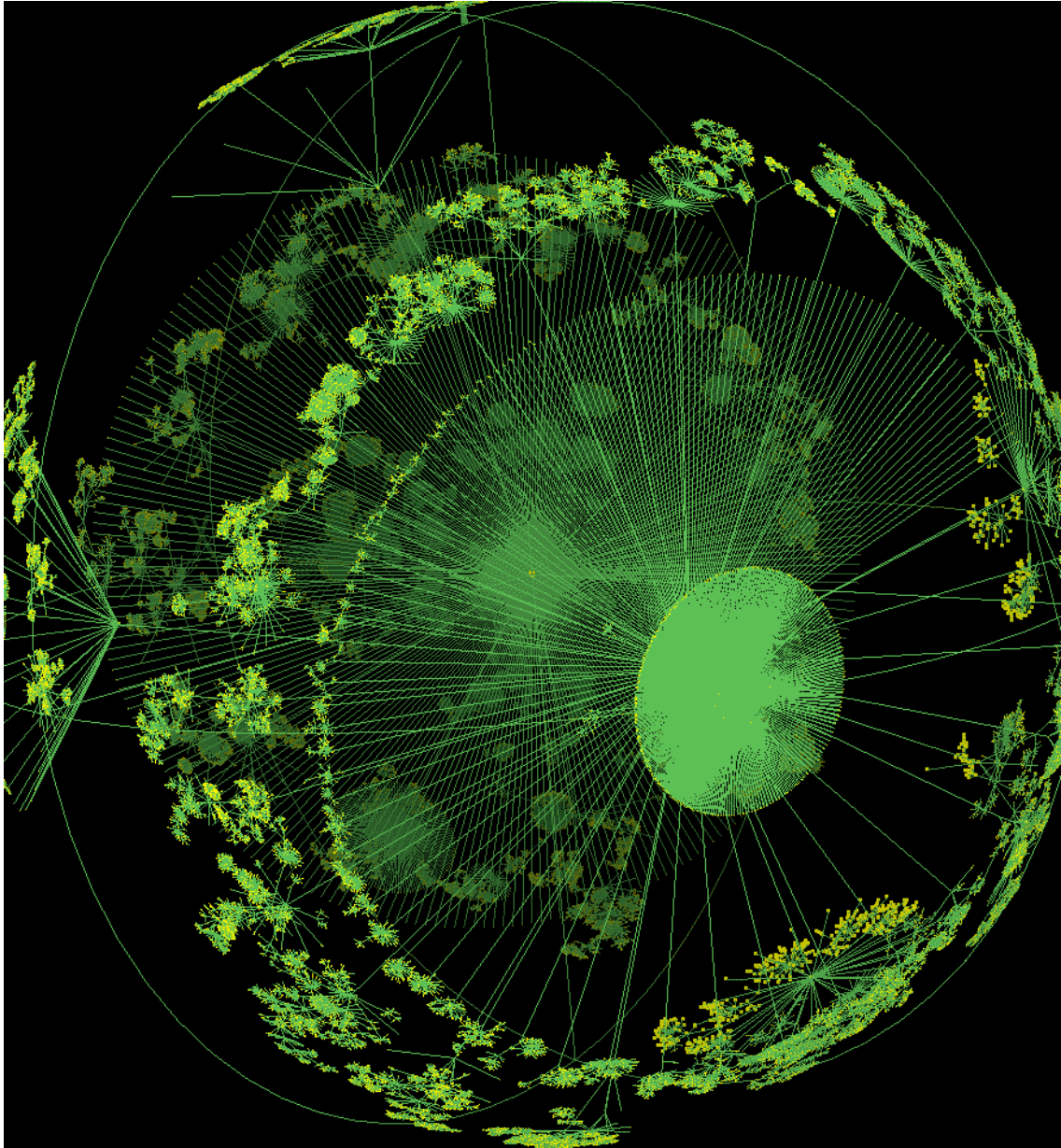


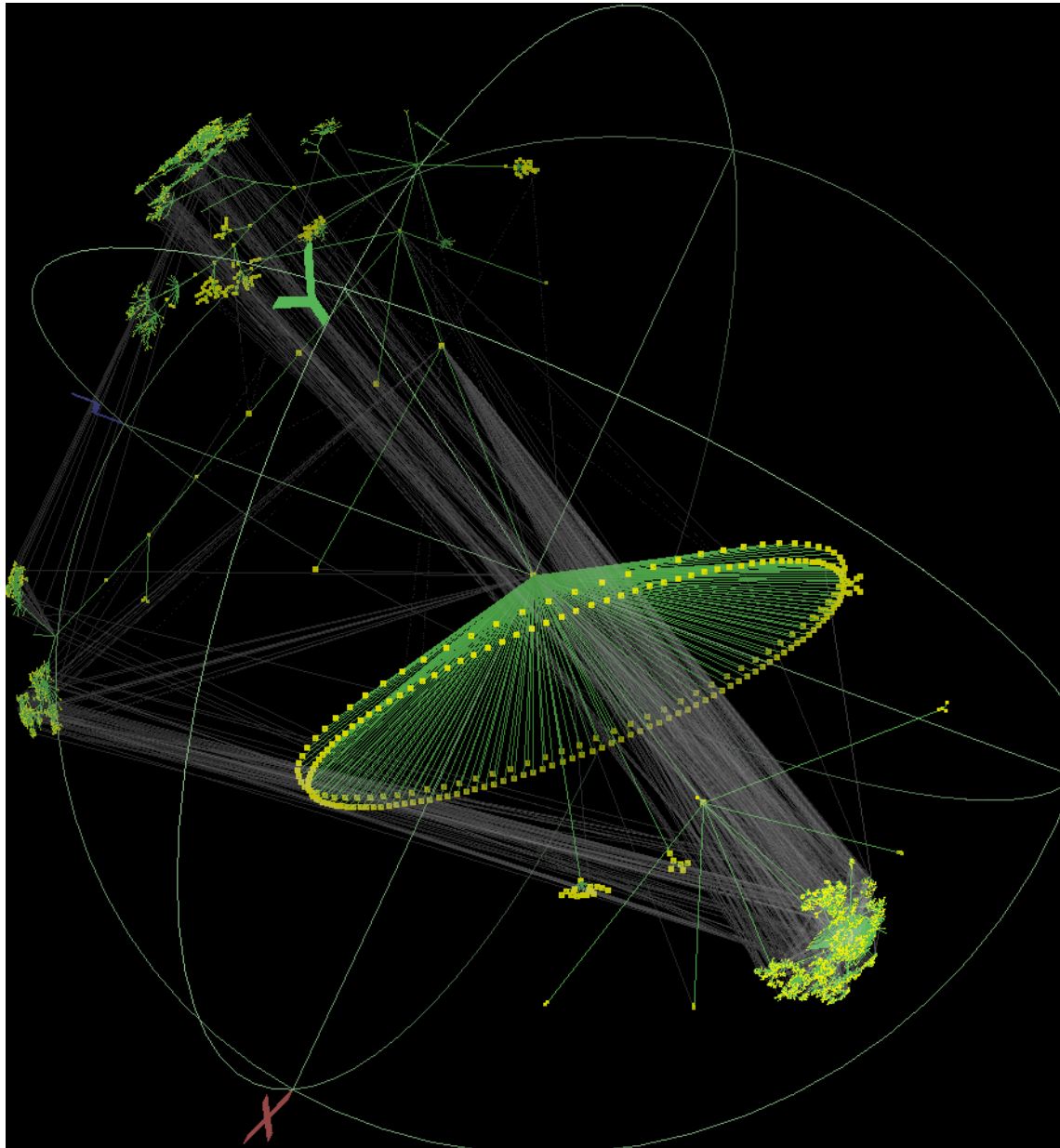
Visualizing Network Traffic

- ❑ Top level protocol statistics
- ❑ Busiest source and destination addresses
- ❑ Port distributions
- ❑ Traffic statistics over time
- ❑ ...others?

Internet Mapping Efforts

- ❑ The Internet Mapping Project www.cs.bell-labs.com/who/ches/map/
- ❑ www.caida.org/tools
- ❑ John Quarterman
<http://www.quarterman.com/~jsq/index.html>
- ❑ <http://www.cybergeography.org/>
- ❑ Others ...see Google





Visualizing the CS LAN at UCF

Day 1 - Feb. 05

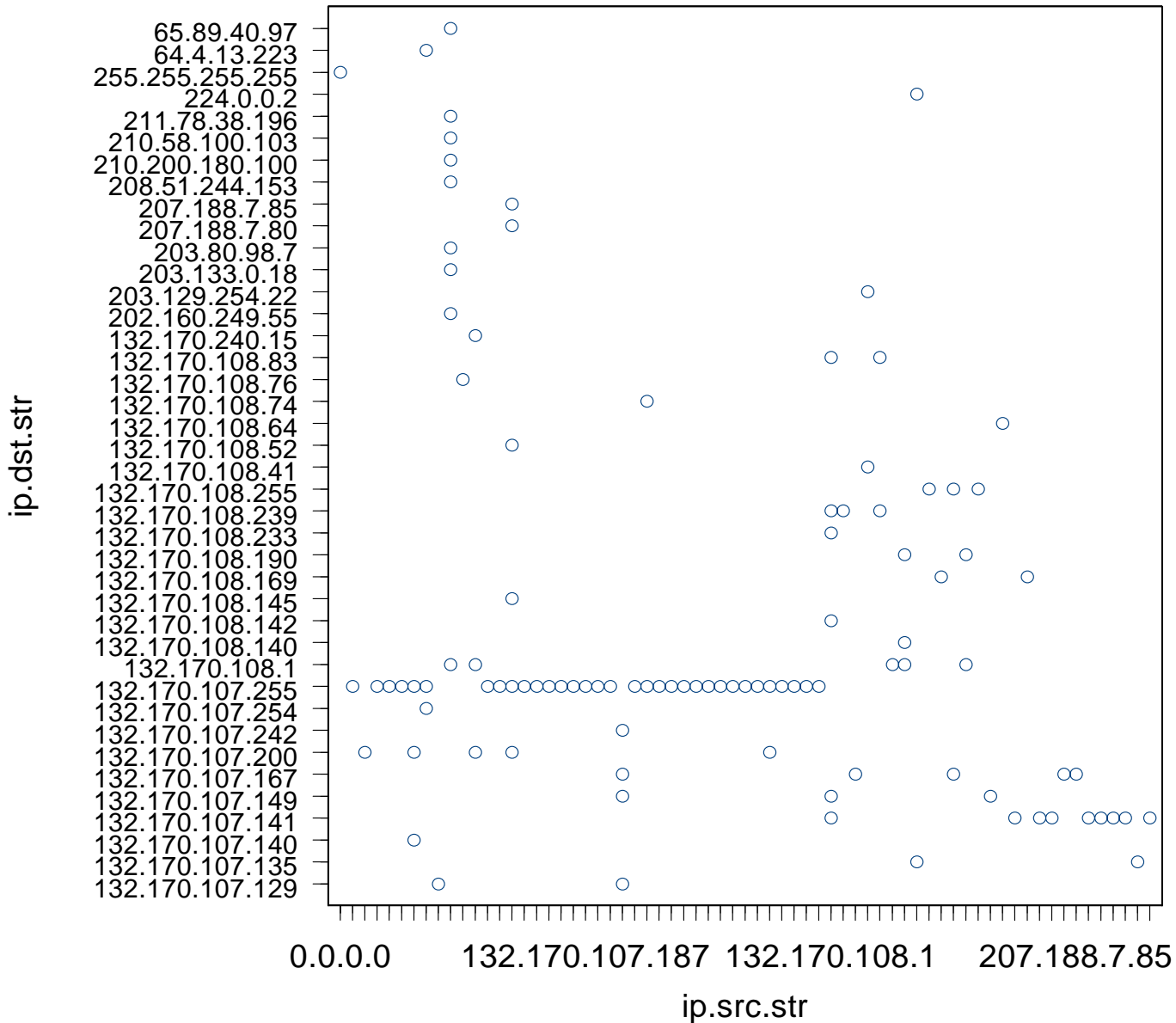
Day 2 - Feb. 07

PKTS 32,070,078

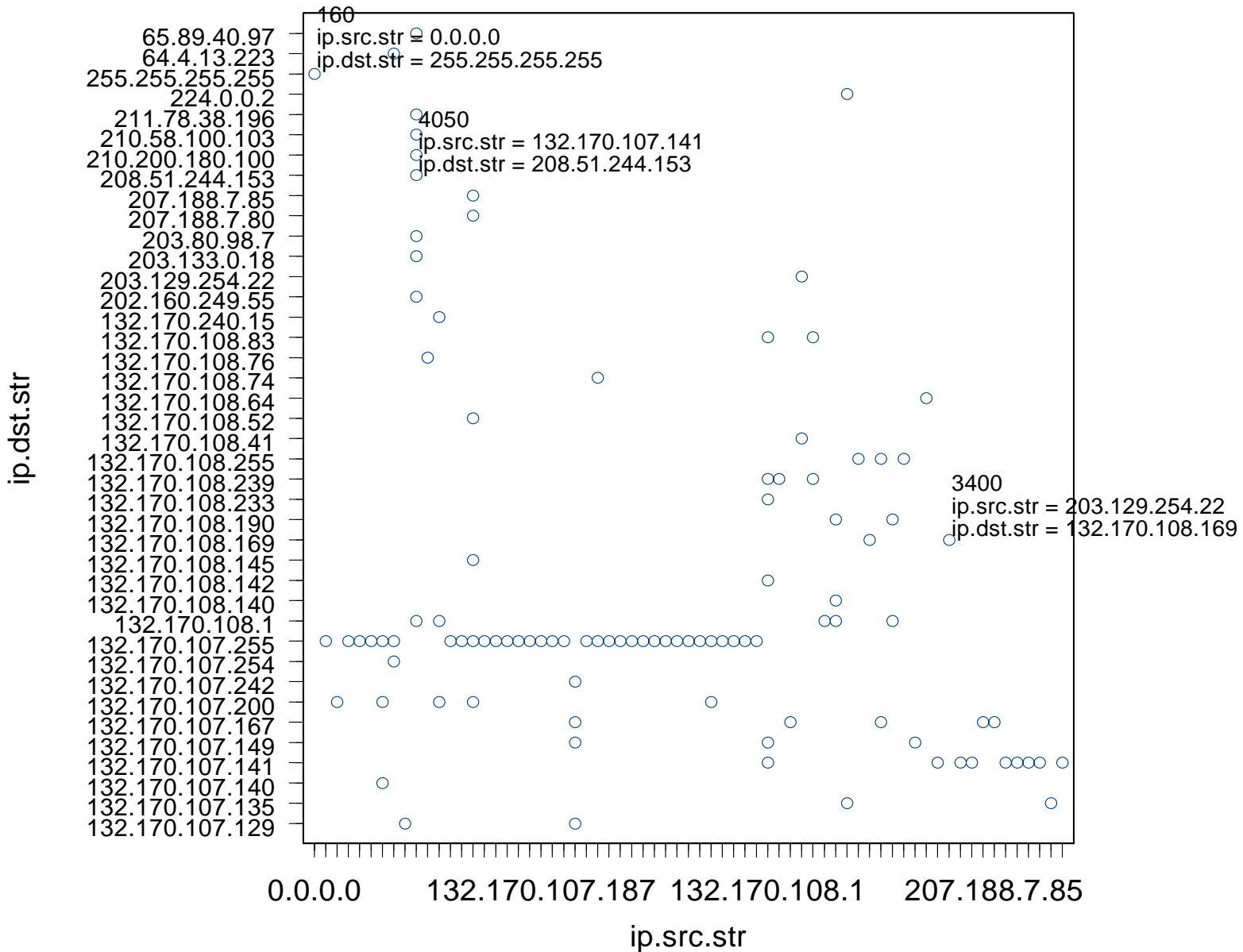
PKTS 20,162,983

	IP	32,070,078		IP	20,162,983
	ICMP	29,197		ICMP	26,694
	UDP	1,740,446		UDP	1,631,138
	TCP	30,300,214		TCP	18,504,809
BYTES	19,857,363,053		BYTES	13,295,981,557	
	ICMP	2,994,825		ICMP	2,658,906
	UDP	772,239,405		UDP	657,411,561
	TCP	19,082,115,557		TCP	12,635,890,568
TCPCONN	212,716		TCPCONN	170,550	
	HTTP	92,034		HTTP	78,581
	SMTP	5,520		SMTP	5,214
	FTPDATA	2,236		FTPDATA	3,817
	TELNET	39		TELNET	37
	FINGER	8		FINGER	10
	FTP	435		FTP	466
	POP3	13,586		POP3	12,018
	TIME	2		TIME	2
	SSH	833		SSH	690
	IRC	0		IRC	0
	IDENT	315		IDENT	246

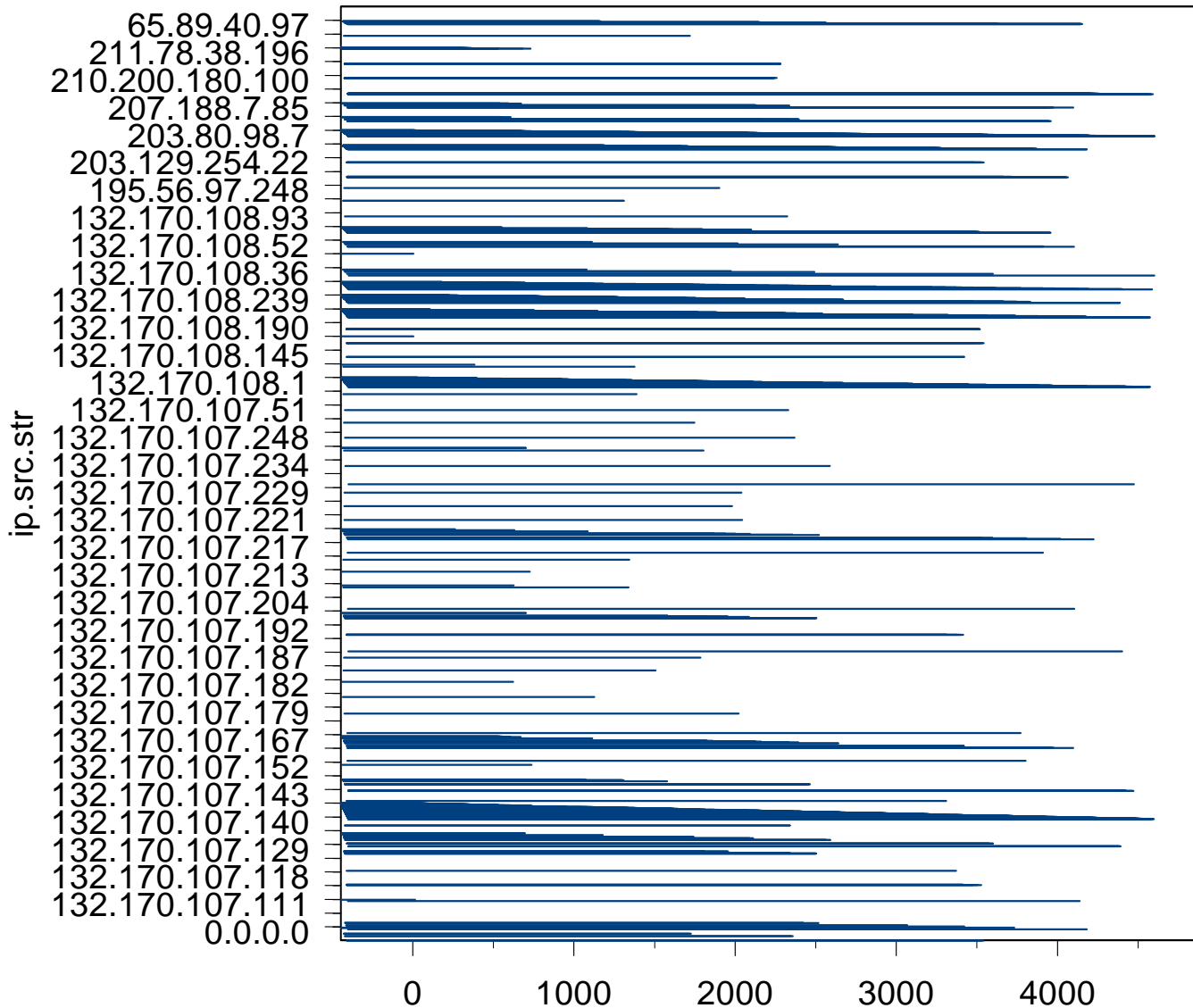
Sources and Destinations - Scatter Plot



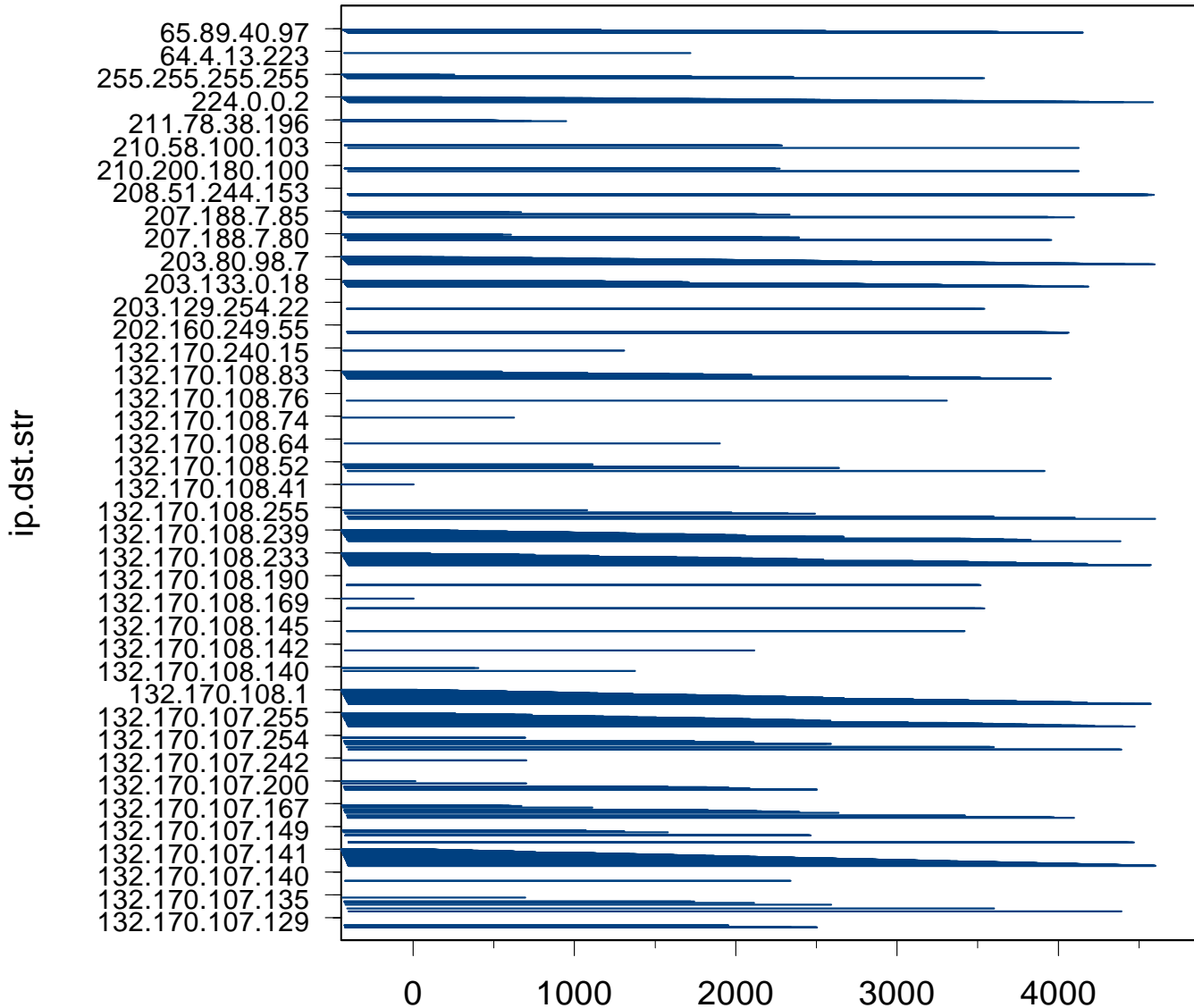
Add a Few Labels

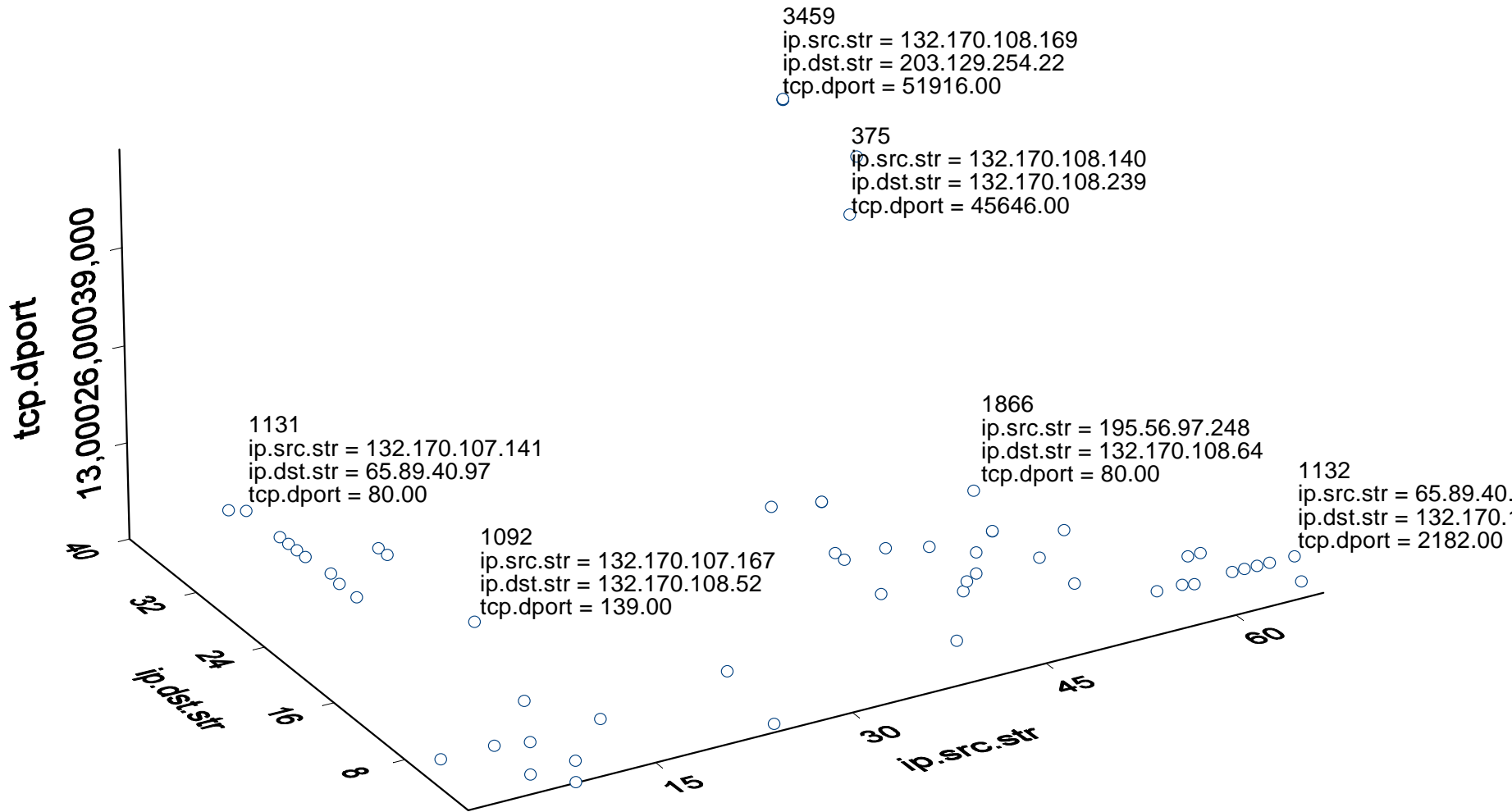


Packet Counts by Source

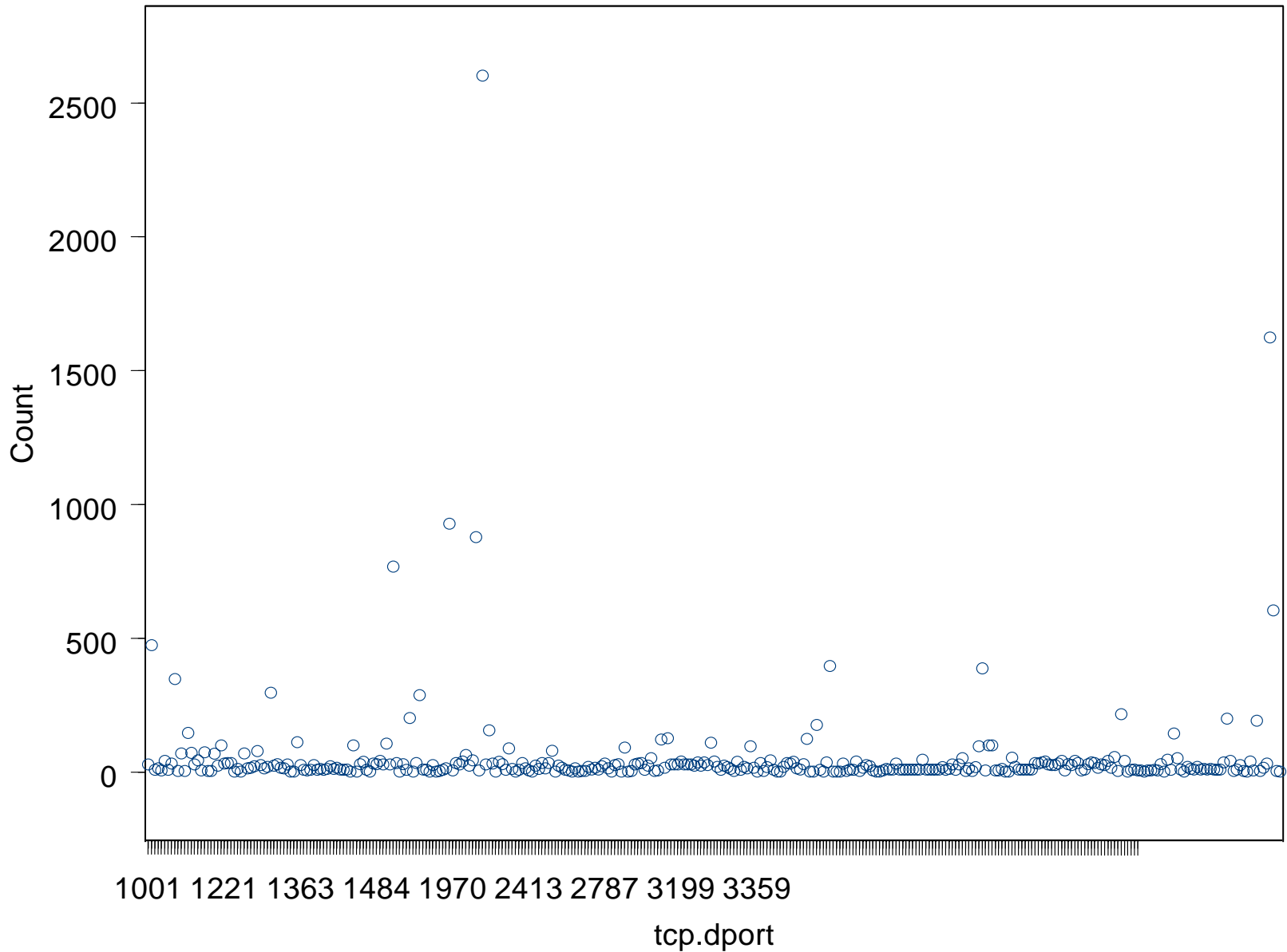


Packet Counts by Destination

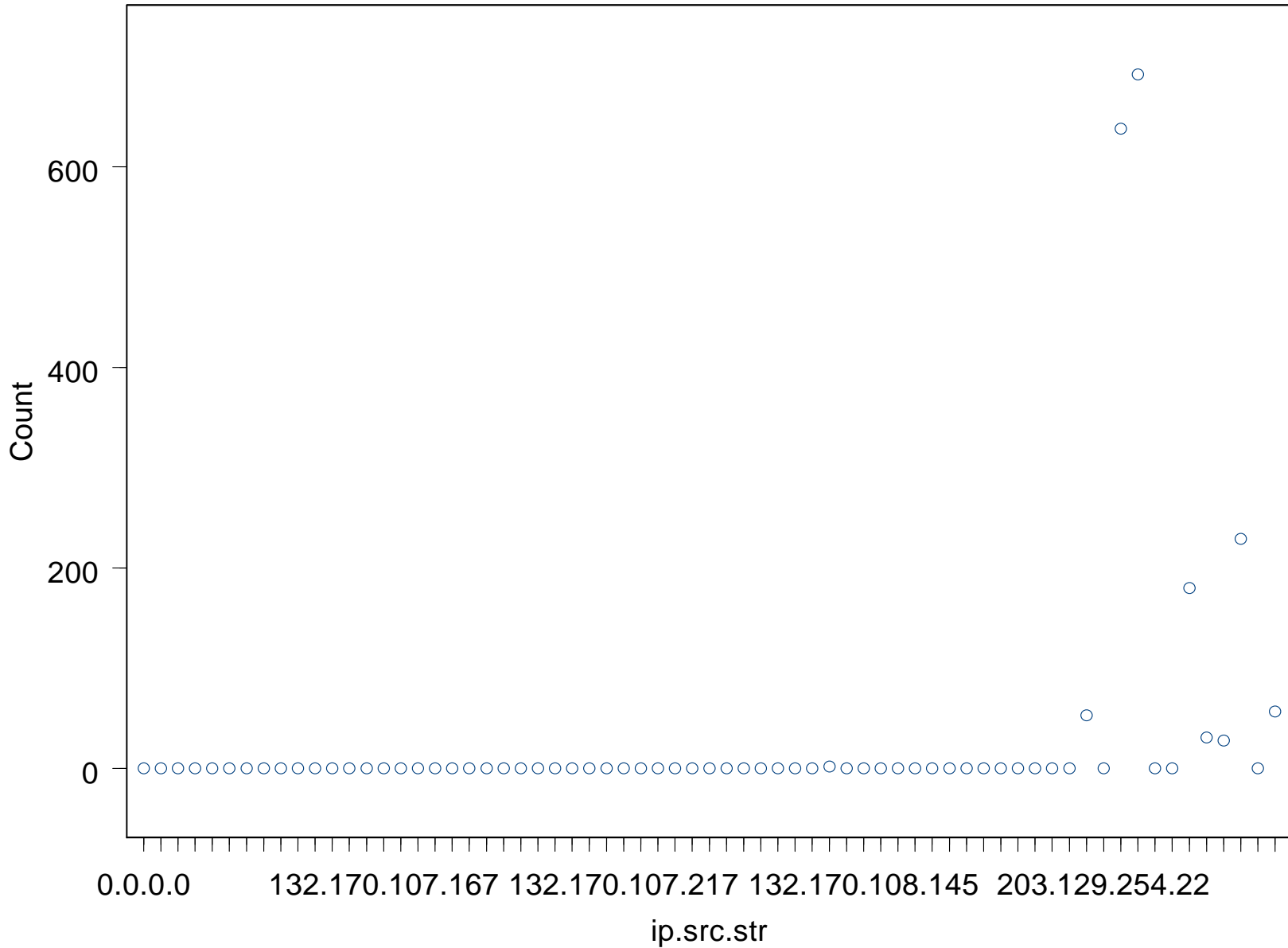


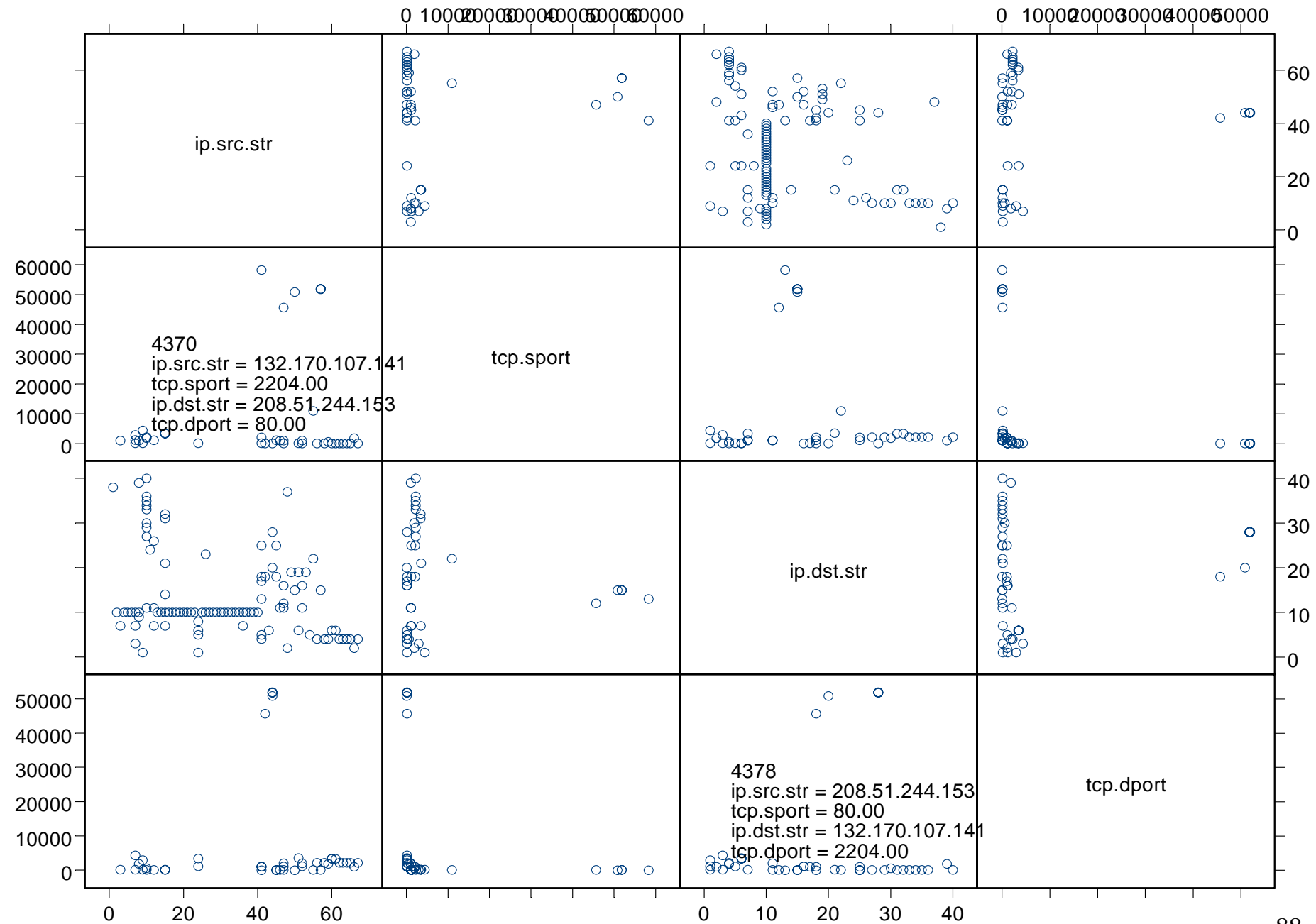


Counts to TCP Destination Ports - All Destination IPs



Counts to 132.170.107.141 by Source IP Address





Detection Based on Arrival Distributions

Gerald A. Marin

Arrival Characteristics Only

- ❑ Lincoln Lab Data
- ❑ Connection Counting and RoC Characterization
- ❑ Self-similarity
- ❑ LoSS Detector
- ❑ Additional Background Data
- ❑ LoSS Results
- ❑ Fractal Dimension and Detection
- ❑ Modifying the Method of Li et. al.

The Lincoln Labs Evaluation of DARPA-Funded ID Systems

The Lincoln Labs evaluation ran 1998-99

- modeled real traffic and collected attack code
- generated attack-free traffic from model
- inserted attacks and logged their location
- 1999 evaluation produced five weeks of data:
 - Two weeks of attack-free traffic and one week of attack training data
 - Two weeks (4 and 5) were the evaluation datasets which contained many different types of attacks
 - Attack databases listed attack occurrences

Attacks Found in the Lincoln Lab Data

23 DoS-TE attacks and high-intensity scans were found in the LL data

- Apache - DoS-TE attack - 4 instances
- Back - DoS-TE attack - 4 instances
- Mailbomb - DoS-TE attack - 3 instances
- Neptune - DoS-TE attack - 3 instances
- Satan - surveillance scan - 2 instances
- Smurf - DoS-TE attack - 5 instances
- Udpstorm - DoS-TE attack - 2 instances

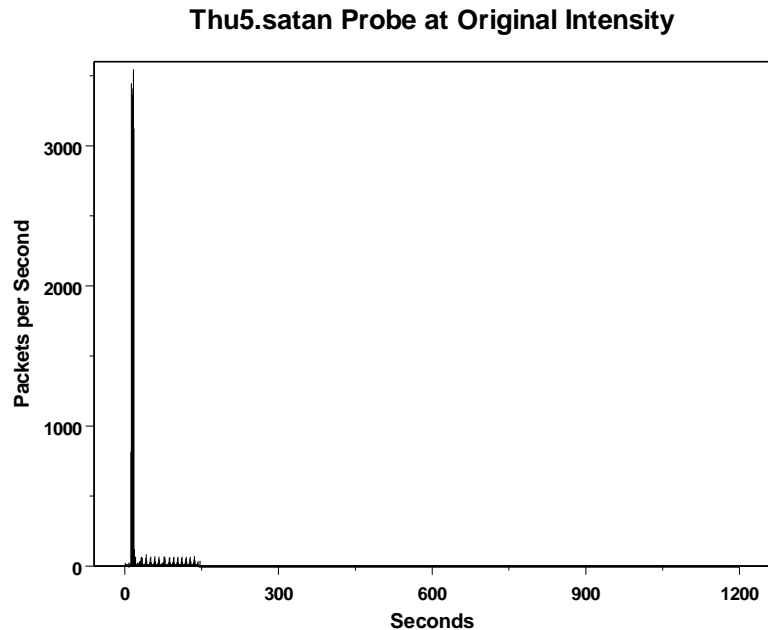
Types of Denial-of-Service Attack

Traffic Exploit (DoS-TE) - uses a continuous stream of traffic to overwhelm the target
examples: *mailbomb, neptune, apache*

System Exploit (DoS-SE) - takes advantage of a software vulnerability to cause a system to fail, usually only needs a few packets

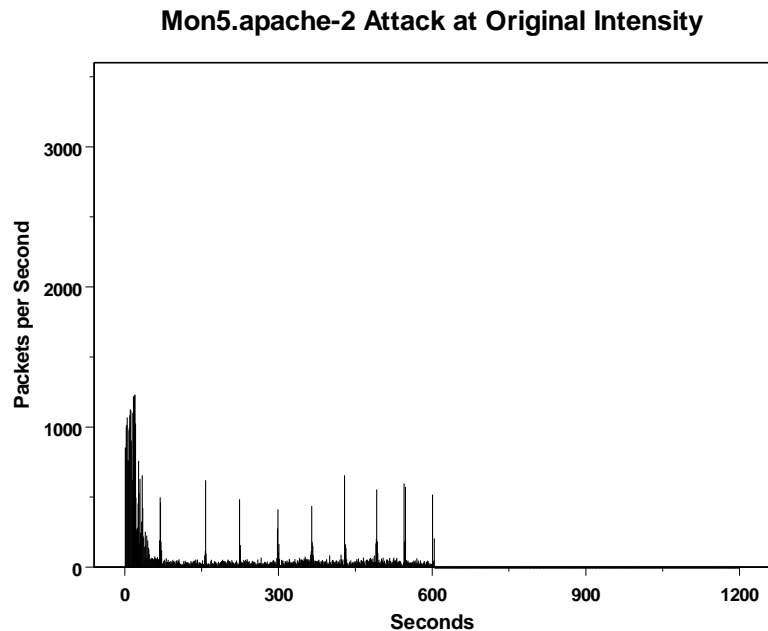
examples: *land, teardrop, ping o' death*

Satan Probe



- surveillance scan
- intensity varies, but one LL instance was quite intense
- plot shows attack's traffic distribution
- 2 instances in LL data

Apache Attack



- DoS attack against *apache* web server
- usually intense attack
- plot shows attack's traffic distribution

Connection Counting*

Using training data:

Let $X_N^T(k) =$ Total number of normal connections initiated in the interval $[kT, (k+1)T)$.

Let $X_A^T(k) =$ Total number of attack connections initiated in the interval $[kT, (k+1)T)$.

Let $X_C^T(k) =$ Total number of connections initiated in the interval $[kT, (k+1)T)$. Clearly $X_C^T(k) = X_N^T(k) + X_A^T(k)$.

* From Statistical Traffic Modeling for Network Intrusion Detection, Joao Cabrera, B. Ravichandran, Raman Mehra, IEEE 2000.

Example Approach

Normal connection arrivals have a Poisson distribution with parameter λ_N . Attack connection arrivals have a Poisson distribution with parameter λ_A . If both Normal and Attack data are present, then the arrival distribution is Poisson with parameter $\lambda_A + \lambda_N = \lambda_C$. We want to pick a threshold for detection, n_d , and use it to determine that an attack is present if $X_C^T > n_d$.

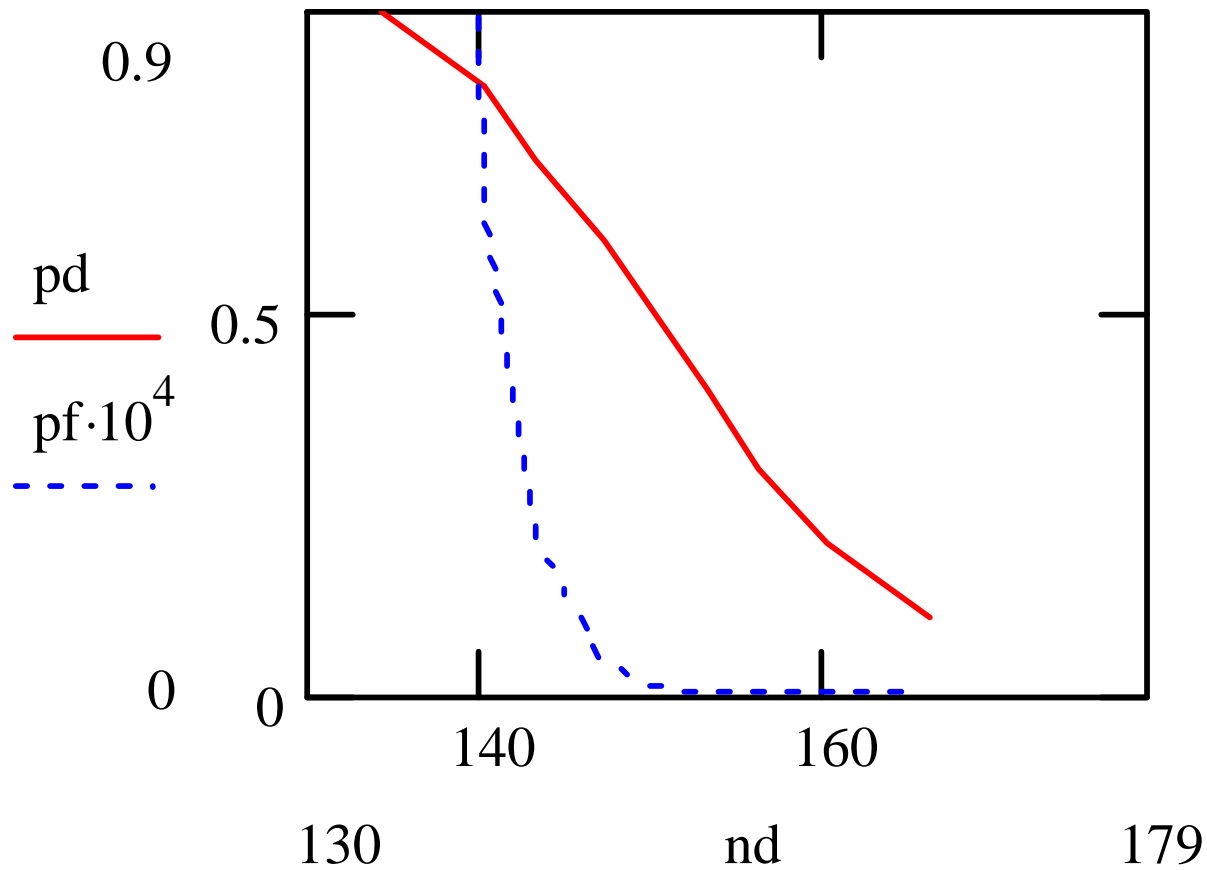
If an attack is present in the arrivals, then
$$P[X_C^T > n_d] = \sum_{k=n_d}^{\infty} \frac{(\lambda_C T)^k e^{-\lambda_C T}}{k!}.$$

Approach: find the value of n_d that causes $P[X_C^T > n_d] = p_d$, the chosen value for probability of detection. Signal an attack when the number of arrivals (over a time period of your choosing) is greater than that number.

Note: if you want the time period to be $1000T$, simply make this substitution in the above equations.

BUT what is the probability of a false alarm?

RoC Curves for Poisson Example

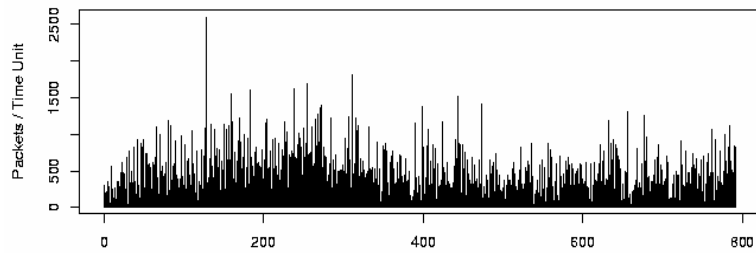


The Lincoln Labs Data and Self-similarity

- ❑ Leland et al. (1994) demonstrated that LAN traffic tends to be self-similar
- ❑ Weeks 1 and 3 of the Lincoln Labs dataset were consistently self-similar between 8am and 6pm
- ❑ Only 50% of the evening data were self-similar
- ❑ See "*On the Self-similarity of Synthetic Traffic for the Evaluation of ID Systems*" (SAINT '03)
- ❑ **Question:** can the loss of self-similarity be a reliable indicator of an attack?

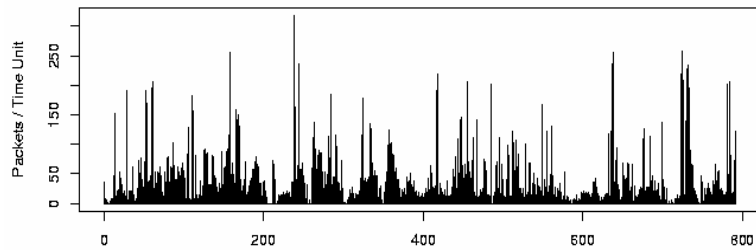
Self-Similar vs. Poisson Traffic at Different Time Scales

Friday Week 1 – Traffic Distribution



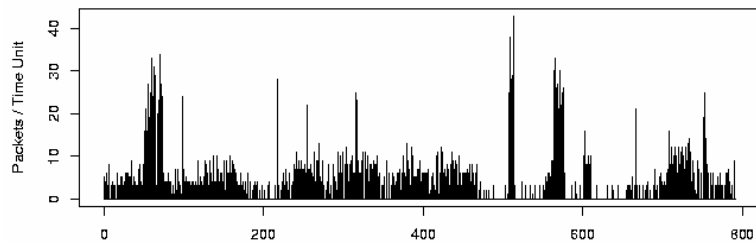
Time Unit = 10 Seconds

Friday Week 1 – Traffic Distribution



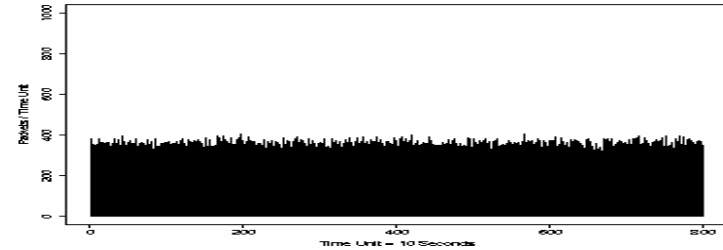
Time Unit = 1 Seconds

Friday Week 1 – Traffic Distribution



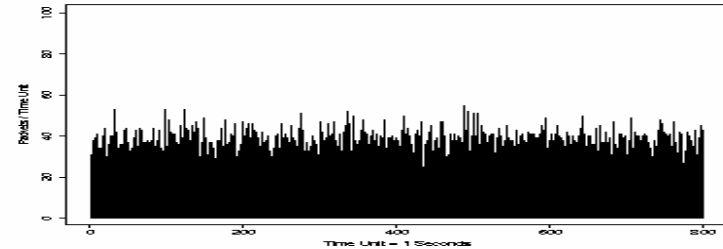
Time Unit = 0.1 Seconds

Poisson-distributed Traffic



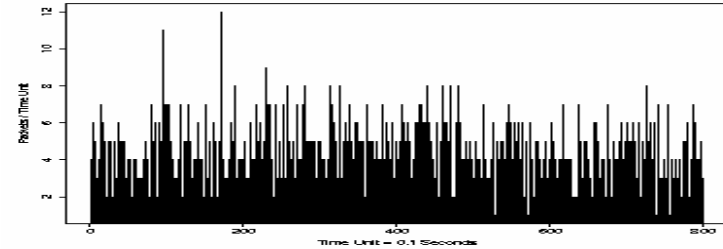
Time Unit = 10 Seconds

Poisson-distributed Traffic



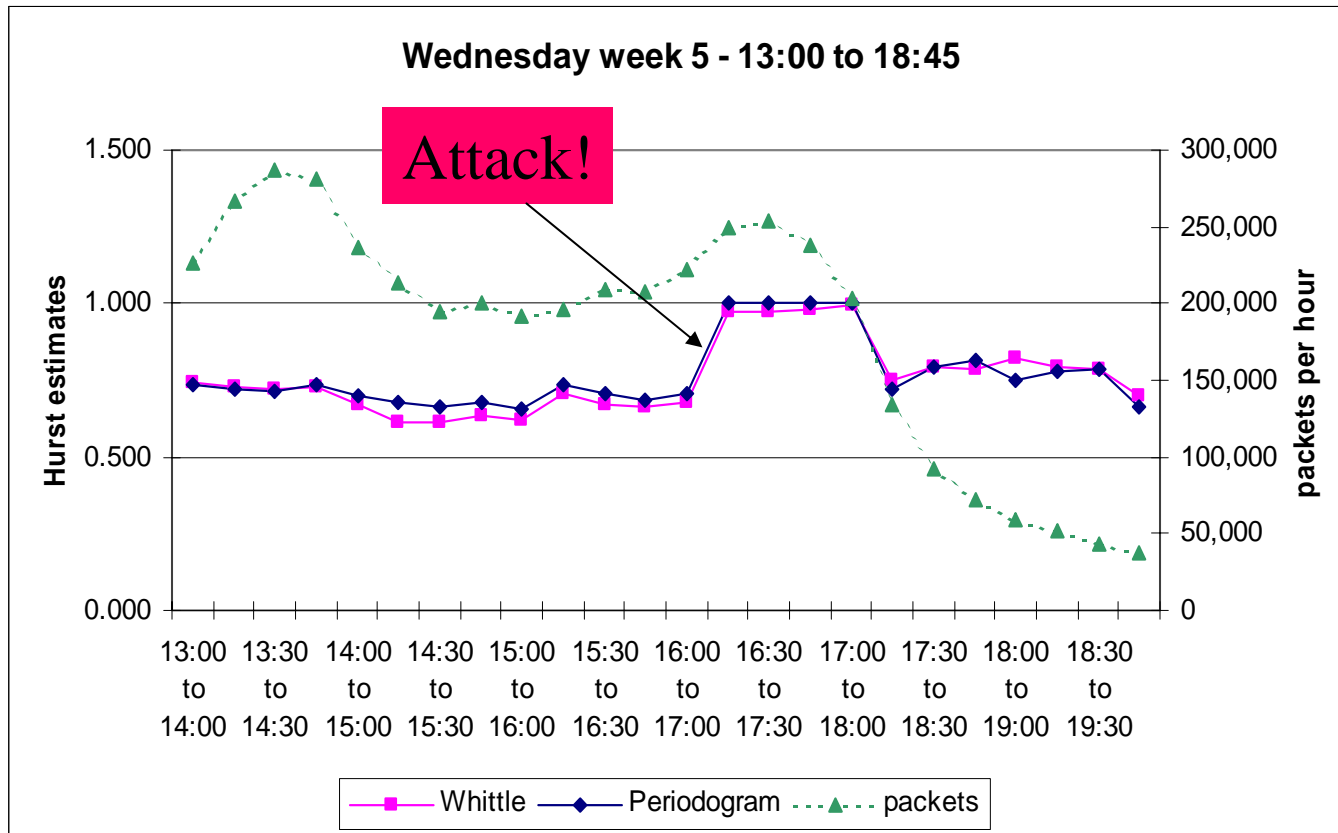
Time Unit = 1 Seconds

Poisson-distributed Traffic



Time Unit = 0.1 Seconds

A Loss of Self-similarity in the Presence of an Attack



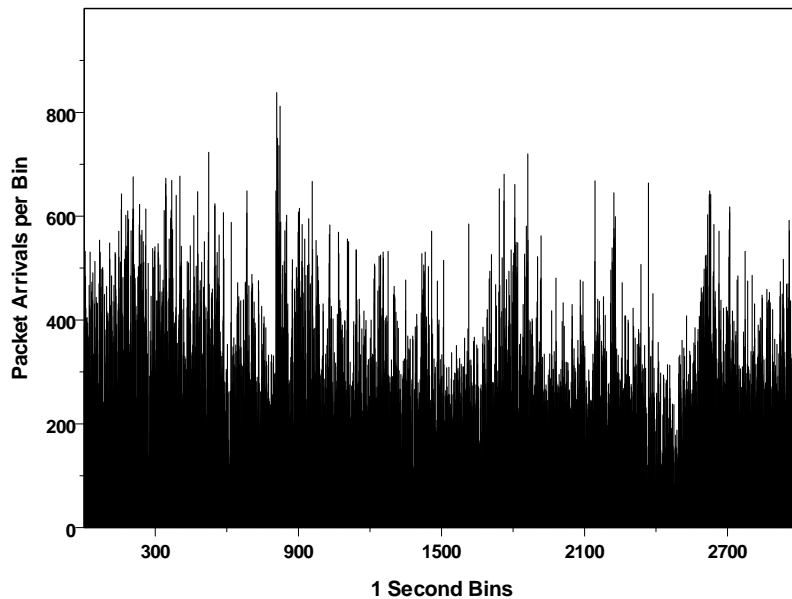
New Background Datasets For Testing Detection Techniques

- Sought more robust background data to test a wider range of attacks
- Must be self-similar and of higher bandwidth than Lincoln Lab background data

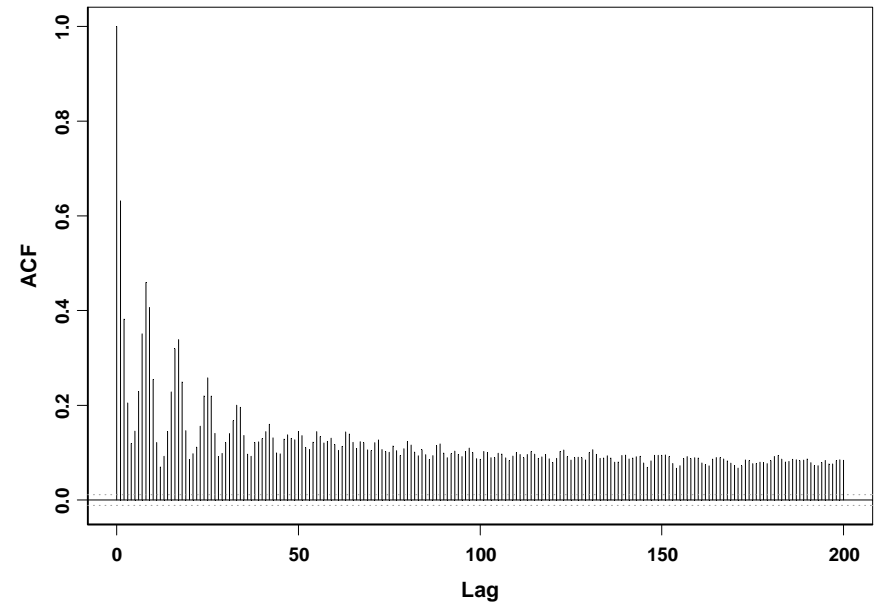
Dataset Name	Source and Date of Capture	Packet Count	Capture Duration	Average Pkts/Sec
BC	BellCore - 08-29-1989	941,299	50 min	313.8
LBL	LBL-4 - 01-21-1994	862,945	60 min	239.7
UCF	UCF LAN - 02-05-2003	1,966,418	90 min	364.2

BC Background Traffic Distribution and Autocorrelations

BellCore Dataset from August 1989



BellCore Autocorrelation



LoSS Results

day-wk-name	attack	attack	LL bkg	BC	LBL	UCF
of attack	peak pps	seconds	avg pps	detect	detect	detect
thu5.satan	3544	147	80.8	Yes+	Yes+	Yes+
mon4.smurf	3385	16	5.9	Yes+	Yes+	Yes+
fri4.smurf	2200	2	8.1	Yes+	Yes+	Yes+
wed4.smurf	2196	2	16.4	Yes+	Yes+	Yes+
mon5.smurf-1	1610	4	55.6	Yes+	Yes+	Yes+
mon5.smurf-2	1605	2	38.3	Yes+	Yes+	
thu5.apache	1394	1132	40.7	Yes+	Yes+	Yes+
tue5.back	1380	1240	46.6	Yes+	Yes+	Yes+
mon5.apache-1	1364	1057	44.2	Yes+	Yes+	Yes+
wed5.apache	1356	519	54.6	Yes+	Yes+	Yes+
mon5.apache-2	1228	604	45.8	Yes+	Yes+	Yes+
wed5.back-1	398	4	25.9	Yes+	Yes+	Yes-
mon5.udpstorm	224	1291	12.8	Yes+	Yes+	Yes+
tue5.neptune	214	820	46.6	Yes+	Yes+	Yes+
tue5.udpstorm	208	1778	0.5	Yes+	Yes+	Yes+
mon5.neptune	134	410	24.6	Yes+	Yes+	Yes+
wed5.back-2	120	109	54.0			
fri5.back	120	109	16.5			
fri5.neptune	56	75	50.0	Yes-	Yes-	Yes-
wed4.satan	28	13	15.5			
thu4.mailbomb	22	607	29.8	Yes-	Yes-	Yes-
fri4.mailbomb	22	600	46.2	Yes-	Yes-	
wed4.mailbomb	21	181	69.3	Yes-	Yes-	Yes-

Will a Threshold Improve Detection?

Li et al. (2001)* - showed that, given an accurate template of non-attack traffic, attacks can theoretically be detected by their effect on self-similarity

*Decision Analysis of Network-Based Intrusion Detection Systems for Denial-of-Service Attacks, Ming Li, Weijia Jia, Wei Zhao, IEEE2001.

The Detector

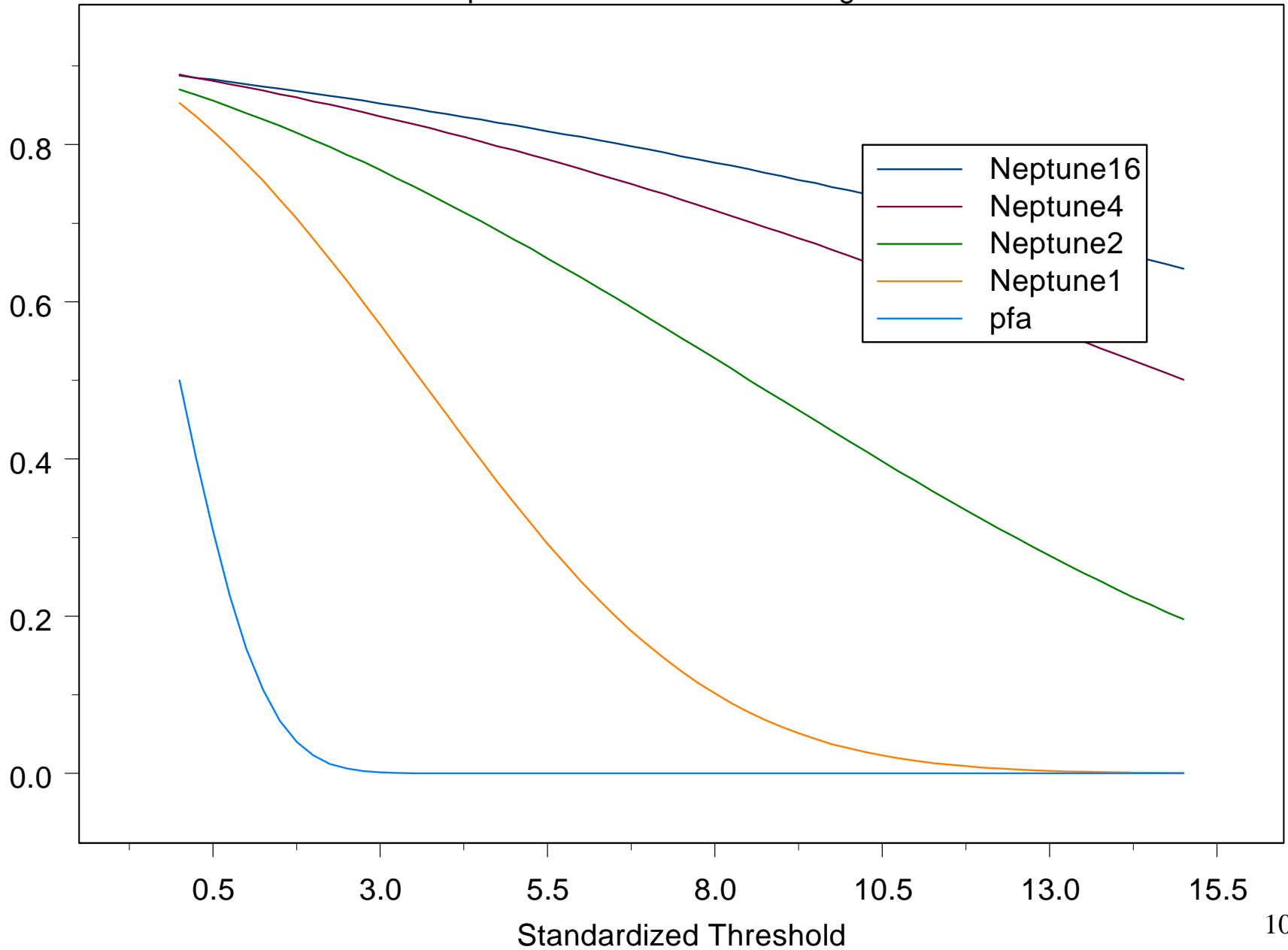
Define the random variable $D = \|r_X - r_Y\|$ for a suitable norm.

For example, recall the Euclidean norm $\|r_X - r_Y\| = \sqrt{\sum_k (r_X(k) - r_Y(k))^2}$.

The authors postulate that with large enough samples the distribution of D will be approximately $N(\mu_D, \sigma_D)$. Now we can determine a threshold number, n_D , so that, IF an attack is present, $P[D > n_D] = p_d$ for whatever detection probability we choose. The probability that we miss an attack becomes $p_m = P[D \leq n_D \text{ when attack is present}]$. The probability of a false alarm is $p_f = P[D > n_D \text{ when attack is not present}]$.

Figure 12: Probability of Detection vs False Alarm

Neptune Attacks in UCF Background



Detecting Attacks by Their Effect on Fractal Dimension

Observations:

- the correlation fractal dimension is a measure of the local structure of the traffic sequence
- attacks will alter the local structure of the traffic, thus they will alter its fractal dimension

Hypothesis:

- changes in the correlation fractal dimension of network traffic can indicate the presence of certain types of attack in that traffic

Correlation Fractal Dimension

- Define a correlation function $C(r)$ as the probability that two points x_i and x_j are within distance r :

$$C(r) = \lim_{N \rightarrow \infty} \frac{1}{N^2} \sum_{i,j=1, i \neq j}^N \theta(r - |x_i - x_j|)$$

where: if $x < 0$ then $\theta(x)=0$, else $\theta(x)=1$

- We know that $C(r)$ obeys a power law:

$$C(r) \sim r^{D_2}$$

Related Work

- ❑ Leung and Haykin (1989) used the correlation fractal dimension to show that radar clutter is a fractal
- ❑ Schouten (1994) proposed a more robust algorithm for calculating correlation fractal dimension
- ❑ Ayedemir (...Marin, et al. 2001) demonstrated a method for calculating the correlation fractal dimension of network traffic to compare two traffic sequences

Intrusion Detection Based on the Correlation Fractal Dimension

- ❑ We performed experiments which show that the correlation fractal dimension is affected by the presence of a network attack.
- ❑ We compared the fractal dimension of data containing an attack with a *traffic template* that did not contain any attacks
- ❑ We used the algorithm given by Ayedemir et al., modified to calculate the fractal dimension based on packet counts rather than inter-arrival times

Sliding Window Example

Block Number	LBL-1 Template	scaled to 1.0 times	scaled to 2.0 times	scaled to 4.0 times	scaled to 8.0 times			
1	0.146	0.507	0.507	0.507	0.507			
2	0.447	1.183	1.183	1.183	1.183			
3	0.403	0.527	0.527	0.527	0.527			
4	0.713	0.262	0.262	0.262	0.262			
5	1.178	0.278	0.278	0.278	0.278			
6	1.300	0.173	0.173	0.173	0.173			
7	2.370	0.192	0.192	0.192	0.192			
8	0.864	0.597	0.597	0.597	0.597			
9	1.695	2.341	2.341	2.341	2.341			
10	0.316	2.098	4.230	8.387	16.048			
11	1.762	2.388	2.388	2.388	2.388			
12	0.548	1.977	1.977	1.977	1.977			
13	0.321	2.717	2.717	2.717	2.717			
Detection Window		Window p-value	Window p-value	Window p-value	Window p-value			
1 to 9		0.98022	0.98022	0.98022	0.98022			
2 to 10		0.56899	0.06468	0.00086	<= 0.05	0.00001	<= 0.05	
3 to 11		0.31959	0.04924	<= 0.05	0.00084	<= 0.05	0.00001	<= 0.05
4 to 12		0.24574	0.04284	<= 0.05	0.00075	<= 0.05	0.00001	<= 0.05
5 to 13		0.25689	0.06473		0.00144	<= 0.05	0.00001	<= 0.05

Detection Results for Fractal Dimension Detector

- ❑ All 23 attacks detected in at least two background datasets
- ❑ 20 of 23 in all three backgrounds
 - results ordered by attack peak pps
 - Yes+ \Rightarrow detected at intensity \leq original level
 - Yes- \Rightarrow detected at intensity $>$ original level
- ❑ Attacks as short as 2 seconds detected
- ❑ No false alarms in any background

Fractal Dimension Results

day-wk-name	attack	attack	LL bkg	BC	LBL	UCF
of attack	peak pps	seconds	avg pps	detect	detect	detect
thu5.satan	3544	147	80.8	Yes+		Yes-
mon4.smurf	3385	16	5.9	Yes+	Yes+	
fri4.smurf	2200	2	8.1	Yes+	Yes+	Yes+
wed4.smurf	2196	2	16.4	Yes+	Yes+	Yes+
mon5.smurf-1	1610	4	55.6	Yes+	Yes+	Yes-
mon5.smurf-2	1605	2	38.3	Yes+	Yes+	Yes-
thu5.apache	1394	1132	40.7	Yes+	Yes+	Yes+
tue5.back	1380	1240	46.6	Yes+	Yes+	Yes+
mon5.apache-1	1364	1057	44.2	Yes+	Yes+	Yes+
wed5.apache	1356	519	54.6	Yes+	Yes+	Yes-
mon5.apache-2	1228	604	45.8	Yes+	Yes+	Yes+
wed5.back-1	398	4	25.9	Yes+	Yes+	Yes-
mon5.udpstorm	224	1291	12.8	Yes+	Yes+	Yes+
tue5.neptune	214	820	46.6	Yes-	Yes-	Yes-
tue5.udpstorm	208	1778	0.5	Yes+	Yes+	Yes+
mon5.neptune	134	410	24.6	Yes+	Yes-	Yes-
wed5.back-2	120	109	54.0	Yes+	Yes-	Yes-
fri5.back	120	109	16.5	Yes+	Yes+	Yes-
fri5.neptune	56	75	50.0	Yes-		Yes-
wed4.satan	28	13	15.5	Yes+	Yes-	Yes-
thu4.mailbomb	22	607	29.8	Yes+	Yes-	Yes-
fri4.mailbomb	22	600	46.2	Yes-	Yes-	Yes-
wed4.mailbomb	21	181	69.3	Yes-	Yes-	Yes-

Combined Detection Results

- ❑ All attacks detected in all backgrounds by one method or the other
- ❑ No false alarms
- ❑ All attacks with peak ≥ 120 pps detected at Yes+ level, including attacks with a 2 second duration

	BC	LBL	UCF
Yes+	87%	74%	61%
Yes-	13%	26%	39%
Total	100%	100%	100%

Characterizing Activity

Dr. G. A. Marin

Activity Profiling

- Collecting statistics that summarize the kinds of activities that occur regularly on the network.
 - Get a description that can be used to identify deviations from normal behavior.
 - Develop profiles of machine behavior
 - Group machines into activity clusters

Characterize Machine Activity

- ❑ Count number of SYNs to each port (by machine)
 - Services sought
- ❑ Count SYN/ACKs from each port
 - Services provided
- ❑ Count SYNs by port (to destination machines)
 - Requests for service
- ❑ Keep a list of all IP addresses interacting with each machine in our network?
 - External machines only?
- ❑ Others?

Activity Profile

- ❑ An activity profile for a machine is a vector of counts or probabilities. Each count is associated with a specific activity.
 - E.g. TCP SYN packets sent to a specific port.
- ❑ Note that counts are generally time sensitive so vectors should be collected by hour of day.
- ❑ Thus consider the activity vector to be a vector of counts or probabilities relative to a given time period on a given day of the week.

Cluster Analysis

- ❑ Developed largely in biological and physical sciences to classify items or individuals into groups.
- ❑ "Clusters" are those with similar characteristics that seem to belong to a single group.
- ❑ Cluster analysis is the commonly used term for using procedures to identify groups in data.

Our Goal

- ❑ Divide machines into clusters based on their activity vectors (either network side or system side).
- ❑ Characterize the activity of machines that are similar (belong to one cluster).
- ❑ Compare new data (or process behavior) with behavior in existing clusters.
- ❑ Alarm if deviation exceeds a threshold.

Principal Topics

- ❑ Distance and proximity matrices
- ❑ System call example
 - Process Activity Matrix
- ❑ Distance measures and scaling
- ❑ Clustering
- ❑ Principal Components Analysis
- ❑ Correct Number of Clusters

A Statistical Method for Profiling Network Traffic*

- *Paper: David Marchette, Published in Proceedings of the Workshop on Intrusion Detection and Network Monitoring, Santa Clara, Ca, April, 1999.
- "Two clustering methods described and applied to NETWORK data. These allow the clustering of machines into 'activity groups', which consist of machines which tend to have similar activity profiles. In addition these methods allow the user to determine whether current activity matches these profiles and hence to determine when there is 'abnormal' activity on the network. A method for visualizing the clusters is described, and the approaches are applied to a data set consisting of a months worth of data from 993 machines."

Example: Counts of Incoming Telnets (1999)

Table 1: Telnet access counts

Destination IP	Daily Count	March Count	March Counts/Day	Feb Count	Feb Counts/Day
331.409.17.39	2	14	0.5	8	0.26
331.409.25.95	3	3	0.1	15	0.48
331.409.28.98	20	323	11.5	834	26.9
331.409.48.49	1	5	0.18	0	0
331.409.6.81	1	2	0.07	17	0.55
331.409.66.35	1	8	0.29	11	0.35
331.409.66.59	2	12	0.43	20	0.65
331.409.50.73	1	1	0.04	0	0
331.409.88.26	10	78	2.8	32	1.03
331.409.90.10	1	64	2.29	43	1.39
331.409.90.4	8	28	1	31	1

Possible Approach

- ❑ Tabulate incoming telnet sessions for current day and compare with activity for previous two months.
- ❑ Counts can be normalized as probabilities.
- ❑ Examine abnormal activity closely
- ❑ Can only be done for major services and a limited number of machines.
- ❑ Marchette suggests using clustering to group large number of machines in order to find activity abnormal for the cluster.

Example

- ❑ Counts kept for first 1024 ports in both TCP and UDP.
- ❑ Separate counts for ports > 1023 .
- ❑ Normalized by total counts to produce probability (activity) vectors of dimension 2050 from data for 993 machines.
 - $1024 + 1 + 1024 + 1$
 - Eliminate ports with prob < 0.2 leaves vectors of length 61.
- ❑ Data plotted with pixel values \sim probabilities.
- ❑ K-means algorithm used for clustering.

Clusters from port counts of 993 machines created with k-means algorithm.

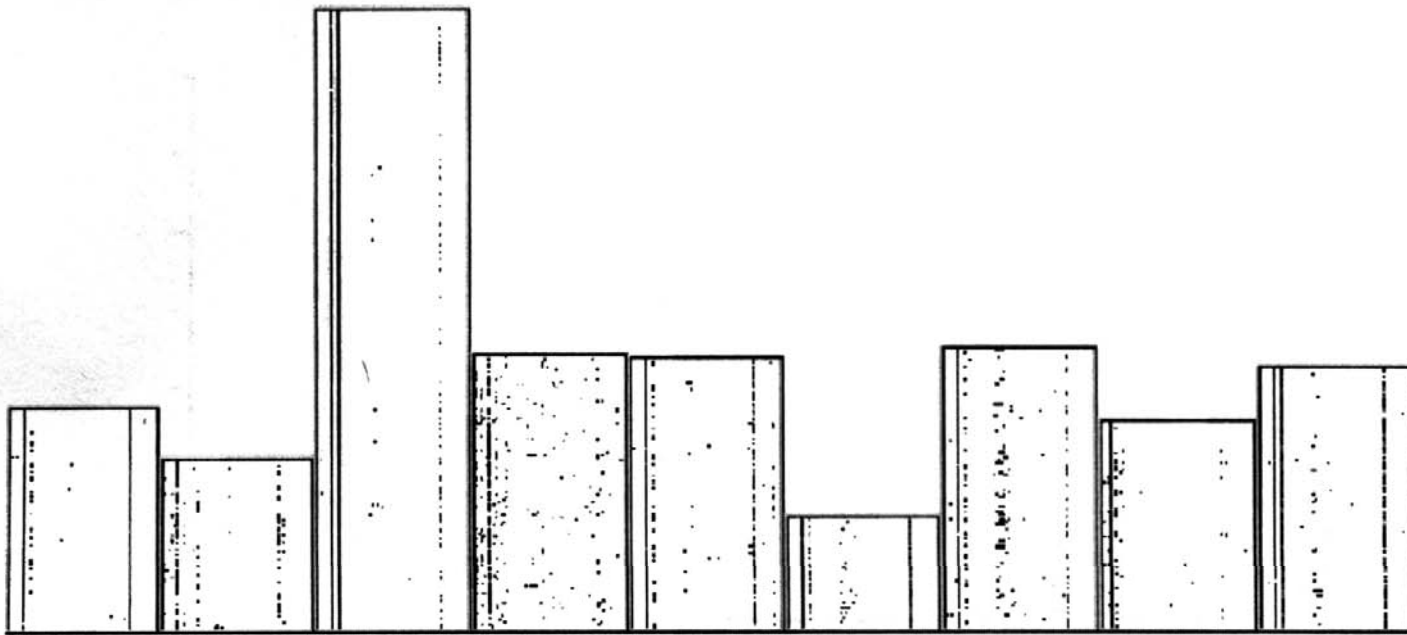


Figure 1: Clusters generated by the k-means algorithm. Each rectangle corresponds to a cluster. The x-axis corresponds to port number, while the y-axis corresponds to individual machines. Only those ports that have a probability above 0.2 for some machine are shown, for a total of 61 ports. The probabilities are indicated by gray scale value, with black corresponding to a probability close to 1.

Idea for "flagging" inbound packets as abnormal.

- ❑ Use the destination address to determine the appropriate cluster profile.
- ❑ Look at the activity probability vector (dim 2050) for that cluster.
- ❑ Pick a threshold and if $P(\text{dest_port}) \leq \text{threshold}$, flag this packet as abnormal.
- ❑ Record the source address as possible attacker.

Marchette Results

Table 5: Results on March Test Data: k means Results

Threshold	Number of Records	Number of Attackers	Type of Attacks Missed
0	61,023	12	Telnets, netbios, ftp, misc
0.0001	78,642	20	netbios, ftp
0.001	112,961	21	netbios
0.005	131,393	23	4 netbios
0.01	146,742	23	4 netbios

"There were [actually] 27 source IPs that were determined to be attackers against one or more of the 993 machines in the data set."
Total number of records analyzed: 1,757,206.

Assignment: Read this paper.

Intrusion detection and response: An empirical analysis of NATE: Network Analysis of Anomalous Traffic Events

Carol Taylor and Jim Alves-Foss

September 2002 **Proceedings of the 2002 workshop on New security paradigms**

Note: Available in ACM Digital Library

This paper presents results of an empirical analysis of NATE (Network Analysis of Anomalous Traffic Events), a lightweight, anomaly based intrusion detection tool. Previous work was based on the simulated Lincoln Labs data set. Here, we show that NATE can operate under the constraints of real data inconsistencies. In addition, new TCP sampling and distance methods are presented. Differences between real and simulated data are discussed in the course of the analysis.