



Packet Analysis



Why Packets?

- Allows complete reconstruction of network events
- Contains all the raw data
- Allows humans to analyze and tell the machine what to do

Size of Full PCAP Storage

How much storage would be needed to monitor the following network?

- 1 Gbps link speed
- Fully saturated
- 24 Hours of traffic

PCAP



Session Info



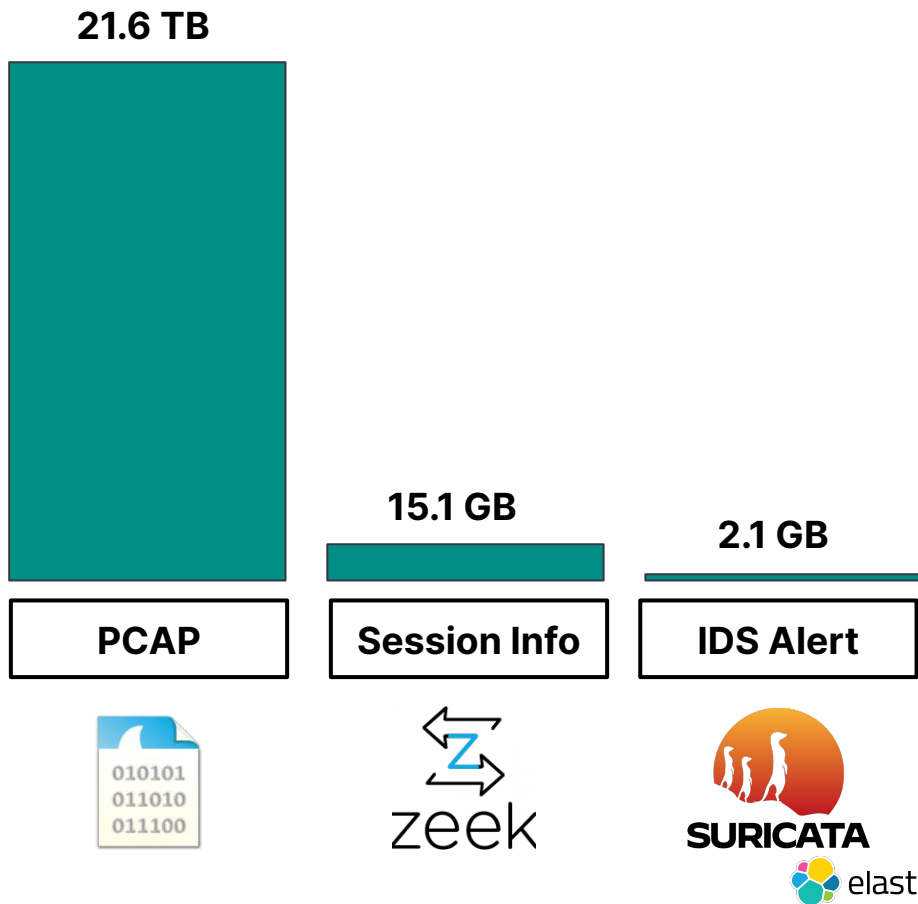
IDS Alert



Size of Full PCAP Storage

How much storage would be needed to monitor the following network?

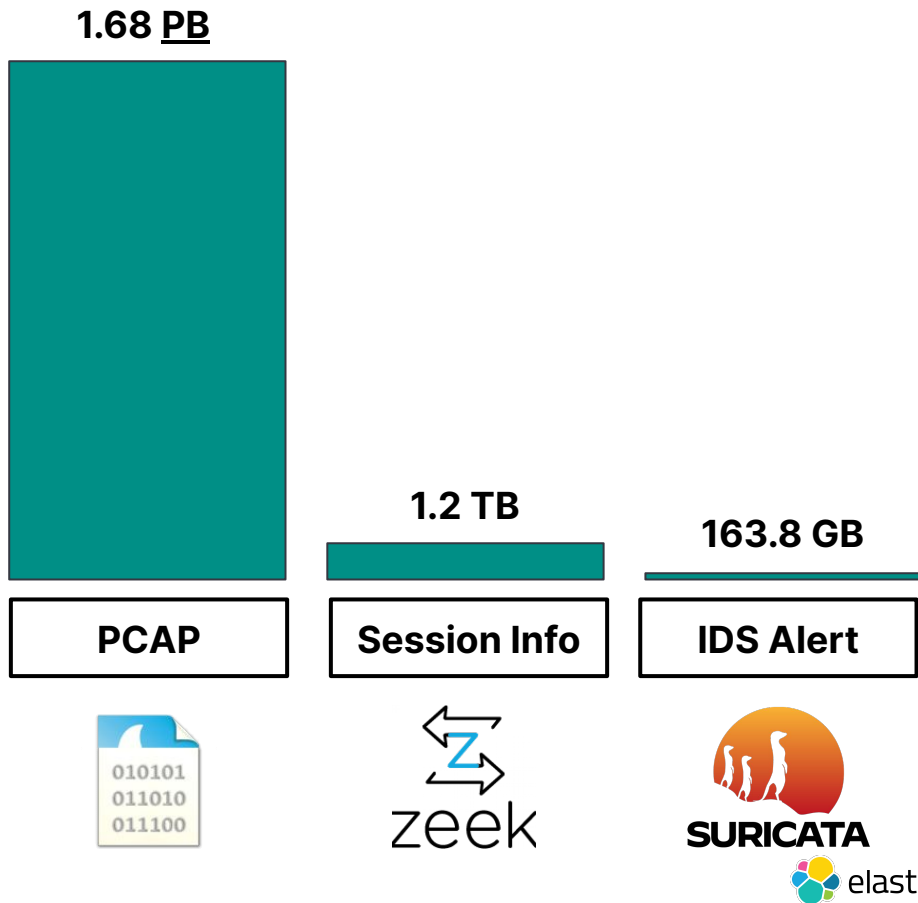
- 1 Gbps link speed
- Fully saturated
- 24 Hours of traffic



Size of Full PCAP Storage - Average Dwell Time

The average **dwell time** of an attacker is **78 days**.

Based on that information, how much storage is required for adequate retention?



PCAP Math

$$\text{Storage} = 2 \times \frac{B_{\text{Avg}} \times 86,400}{8}$$

- 1 day @ saturated 1 Gbps = 21,600 GB file storage
- B_{Avg} is the average link speed

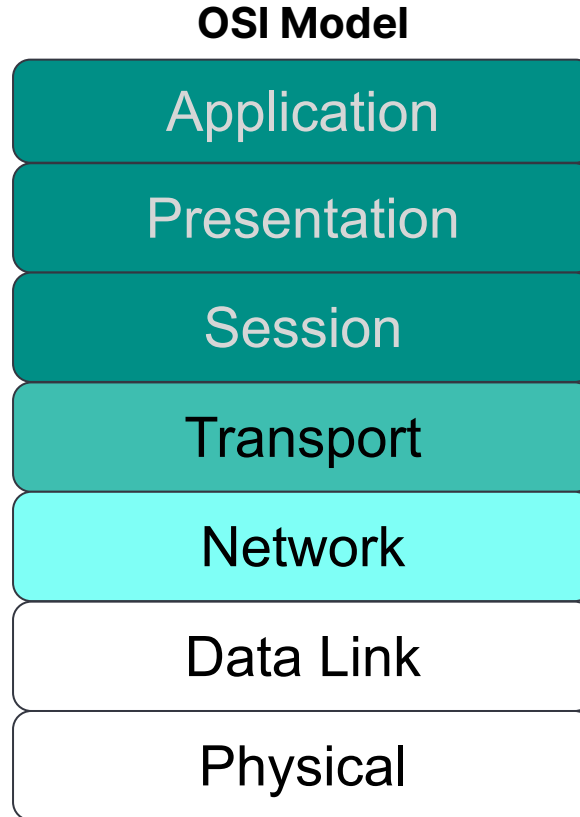
Packet Analysis

1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

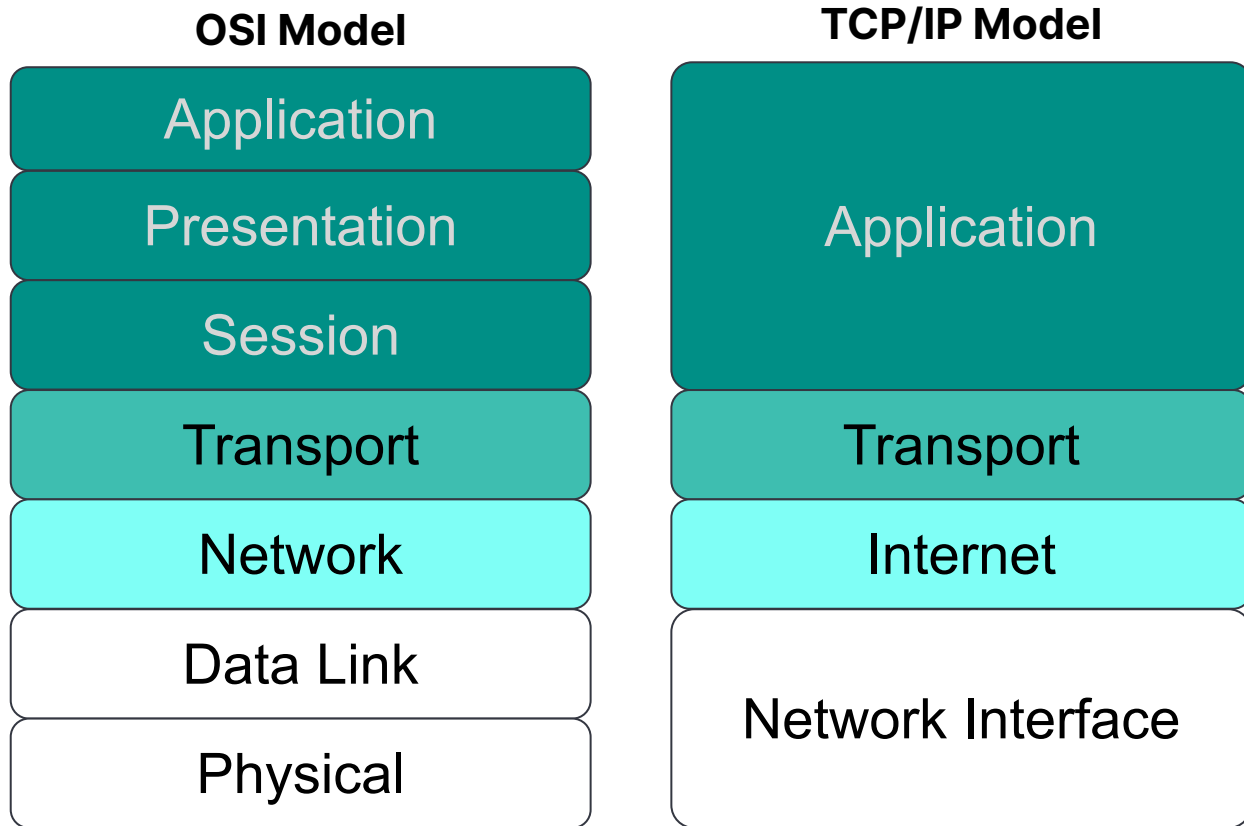
Packet Analysis

1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

OSI Model Conceptual Layer



TCP/IP Conceptual Layer



Network Interface

- Manages the electrical signal that represents data transfer
- Sends data from one machine to another based physical (MAC) address

Protocols:

Ethernet (Physical), Ethernet (Logical), Bluetooth, USB, Wifi, MAC, LLDP

Devices:

CAT5 Cable, Hubs, Repeaters, Switches

TCP/IP Model

Application

Transport

Internet

Network Interface

Internet Layer

- Routes packets to the next-hop host and passes link layer info
- Manages data routing paths and protocols

Protocols: IPv4, IPv6, ICMP, RIP, OSPF

Devices: Routers, Gateways

TCP/IP Model

Application

Transport

Internet

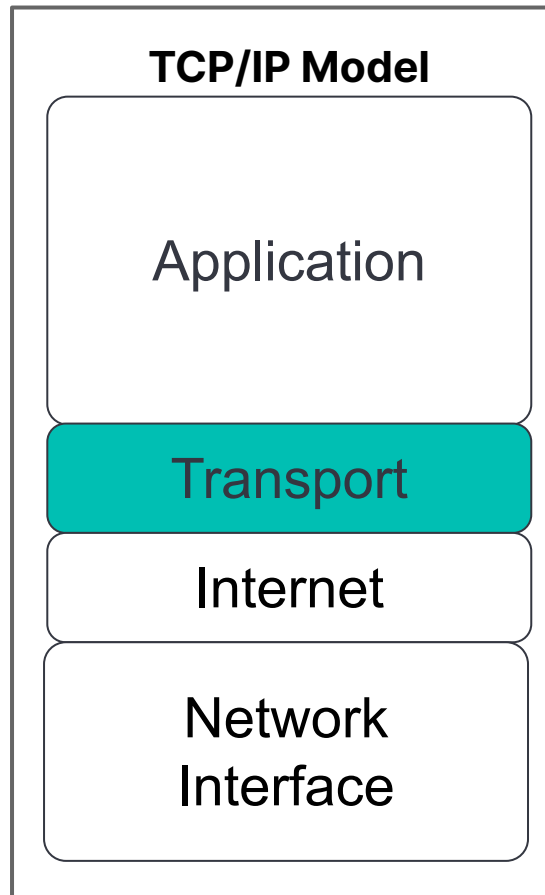
Network
Interface

Transport Layer (4)

- Operates over port numbers
- Connection-oriented
- Flow control

Protocols:

TCP, UDP, iSCSI



Application

- Data intended for final destination software
- Users are most likely to interact with this layer

Protocols:

HTTP, IMAP, DNS, DHCP, SSL/TLS, SMB

Devices:

Web Application Firewall, Layer 7 Switch

TCP/IP Model

Application

Transport

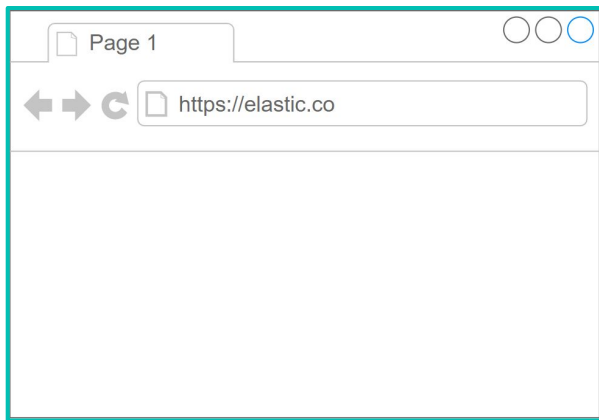
Internet

Network
Interface

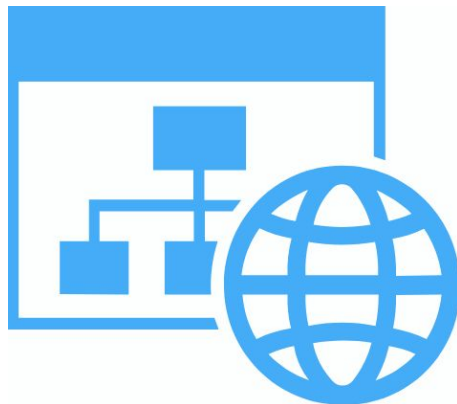
Journey of a Packet

- Primary goal is to get data from one application to another
- These two pieces of software know how to communicate with each other

**Browser
Software**

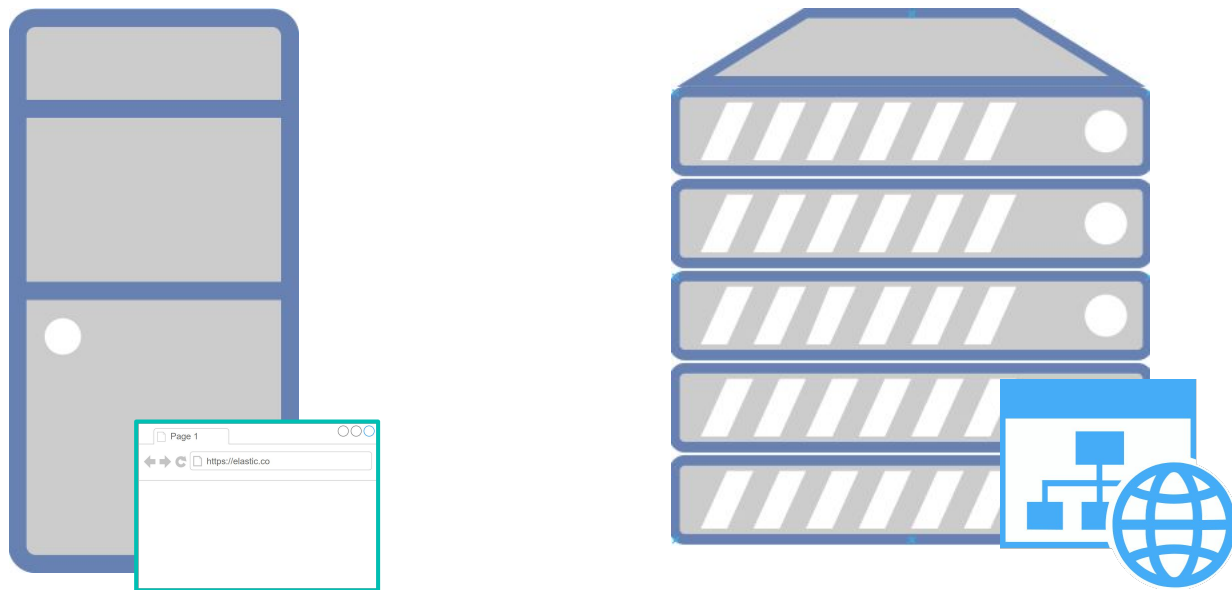


Web Server Software



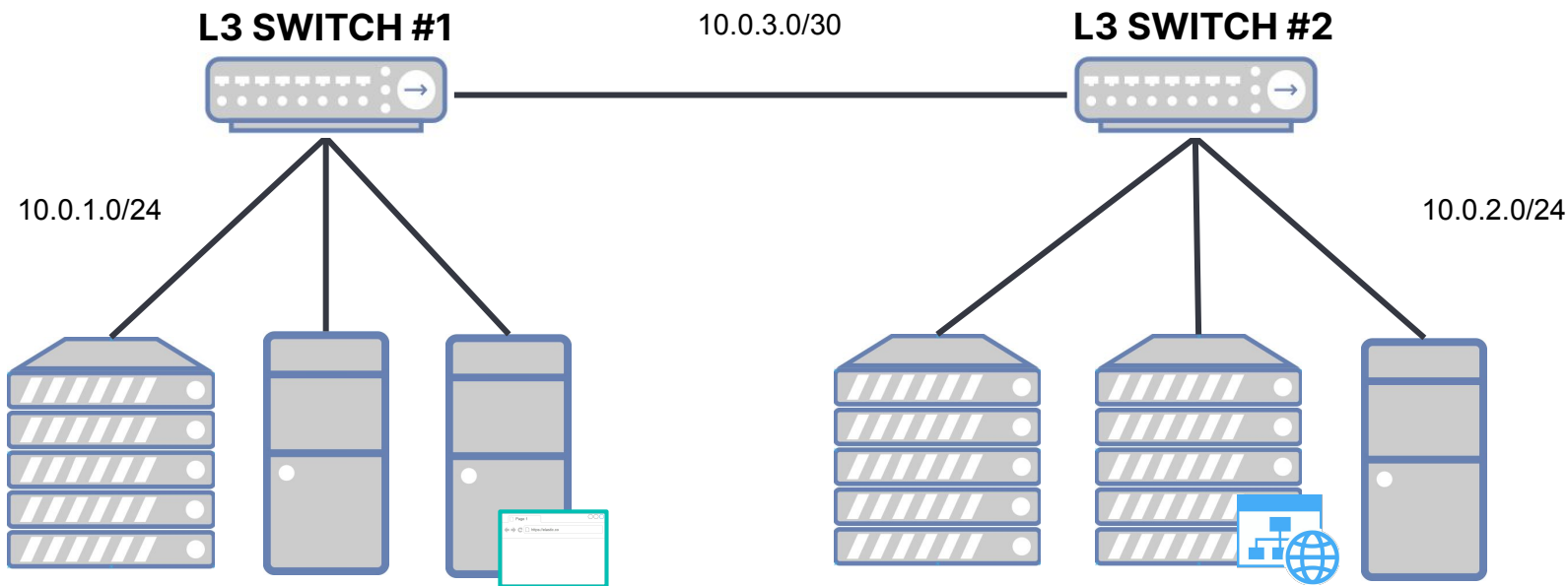
Journey of a Packet

Unfortunately the software is usually not on the same machine



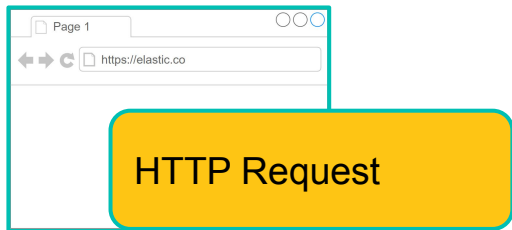
Journey of a Packet

The systems also aren't directly connected to one another



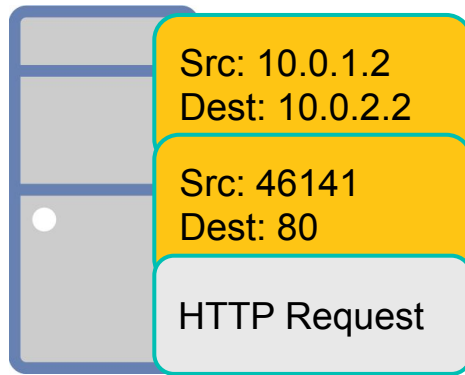
Journey of a Packet

1



The application (browser) creates a payload intended for the web server

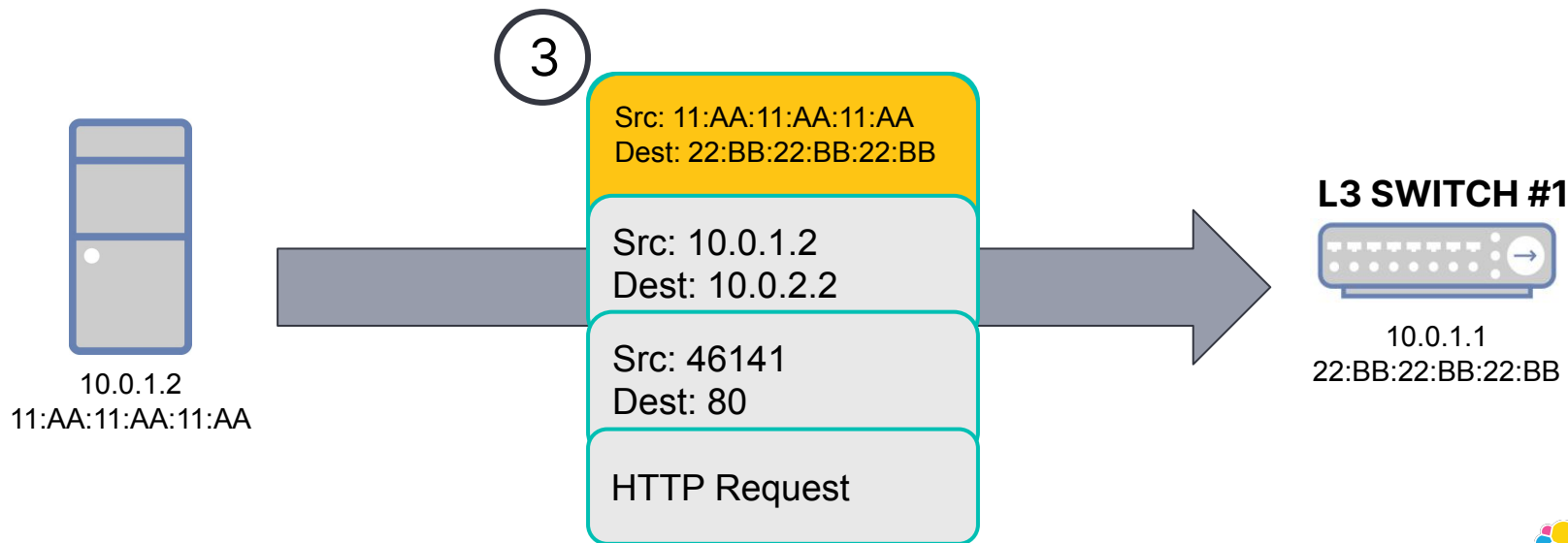
2



The host then adds port and IP information indicating the final destination and where to send data back

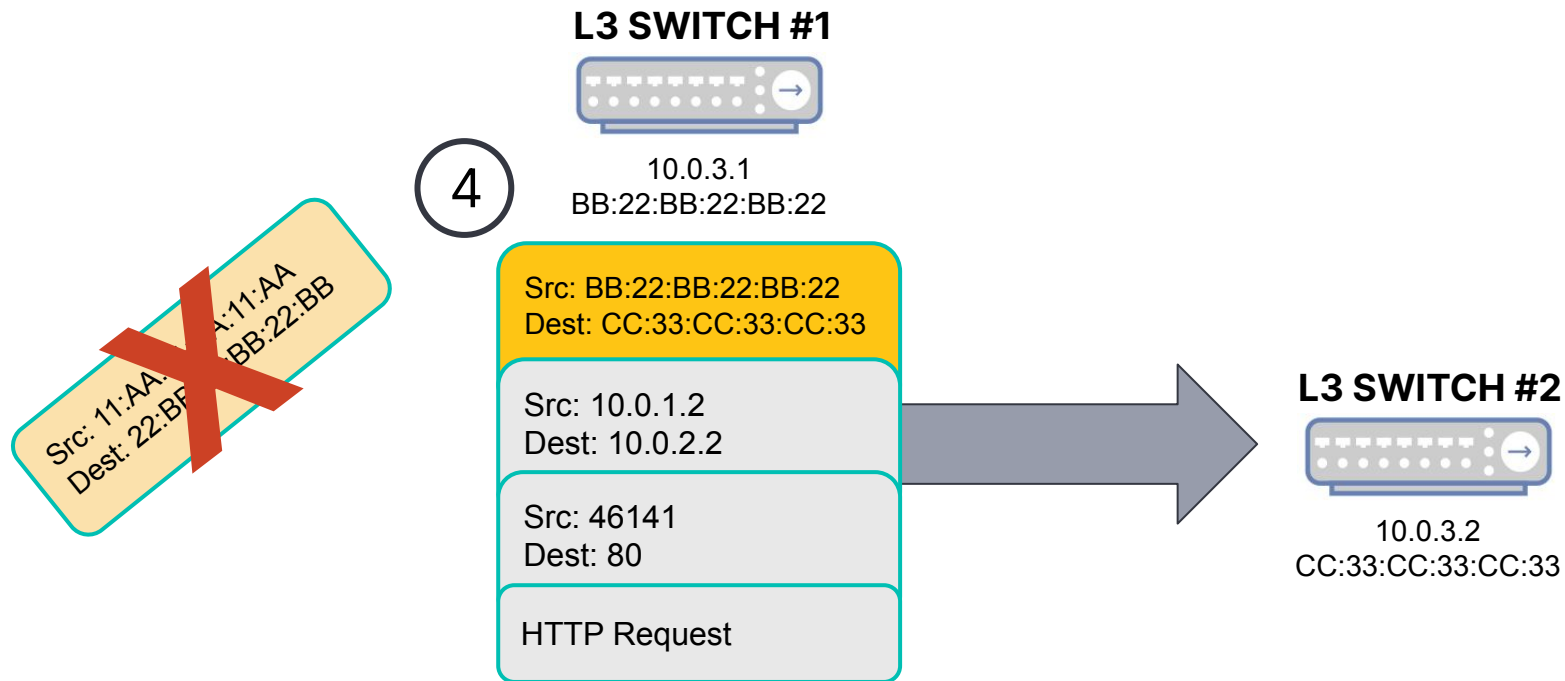
Journey of a Packet

- The packet can't skip other devices and magically appear on the web server so we use the network interface layer to help it hop through devices
- Because the destination host exists outside the client's subnet, the MAC address of the default gateway is used for the destination MAC



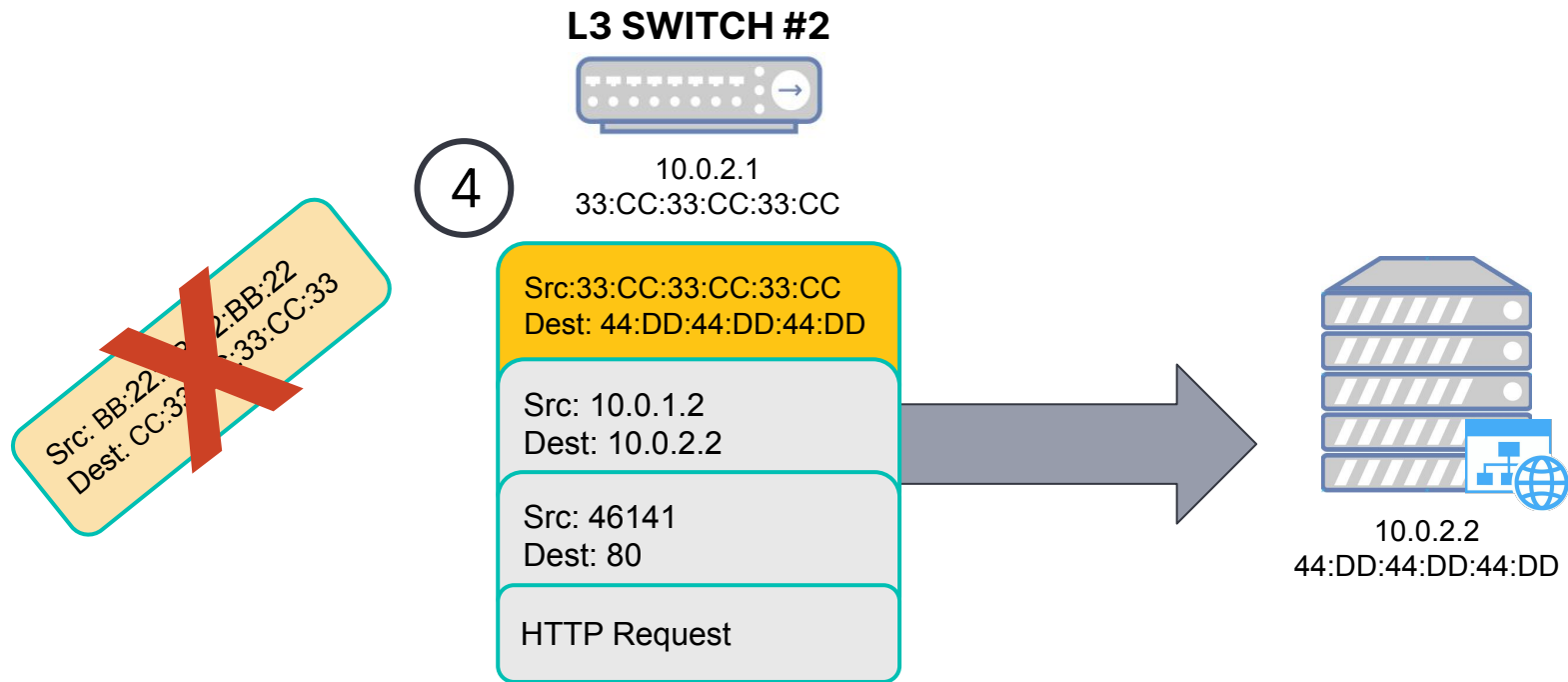
Journey of a Packet

- The first L3 switch checks its route table for the destination 10.0.2.2
- Then the switch replaces the Ethernet Header for the next hop L3 switch



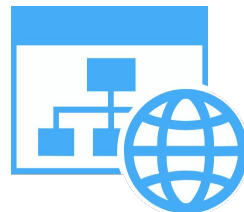
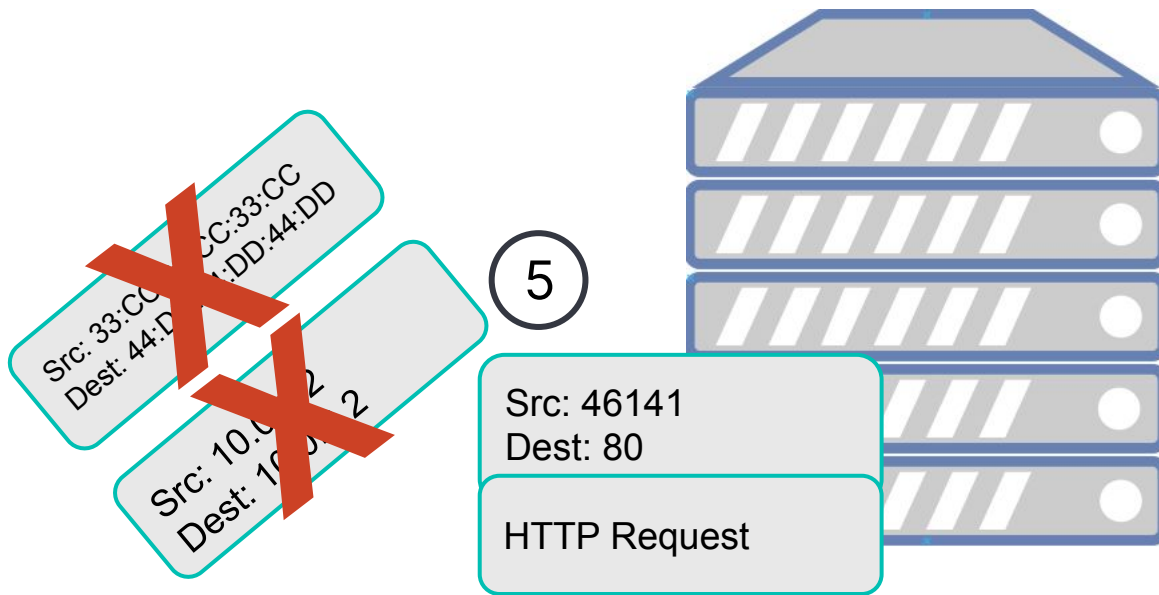
Journey of a Packet

- The switch replaces the Ethernet Header with the correct info for the next hop
- The destination is the MAC of the destination web server



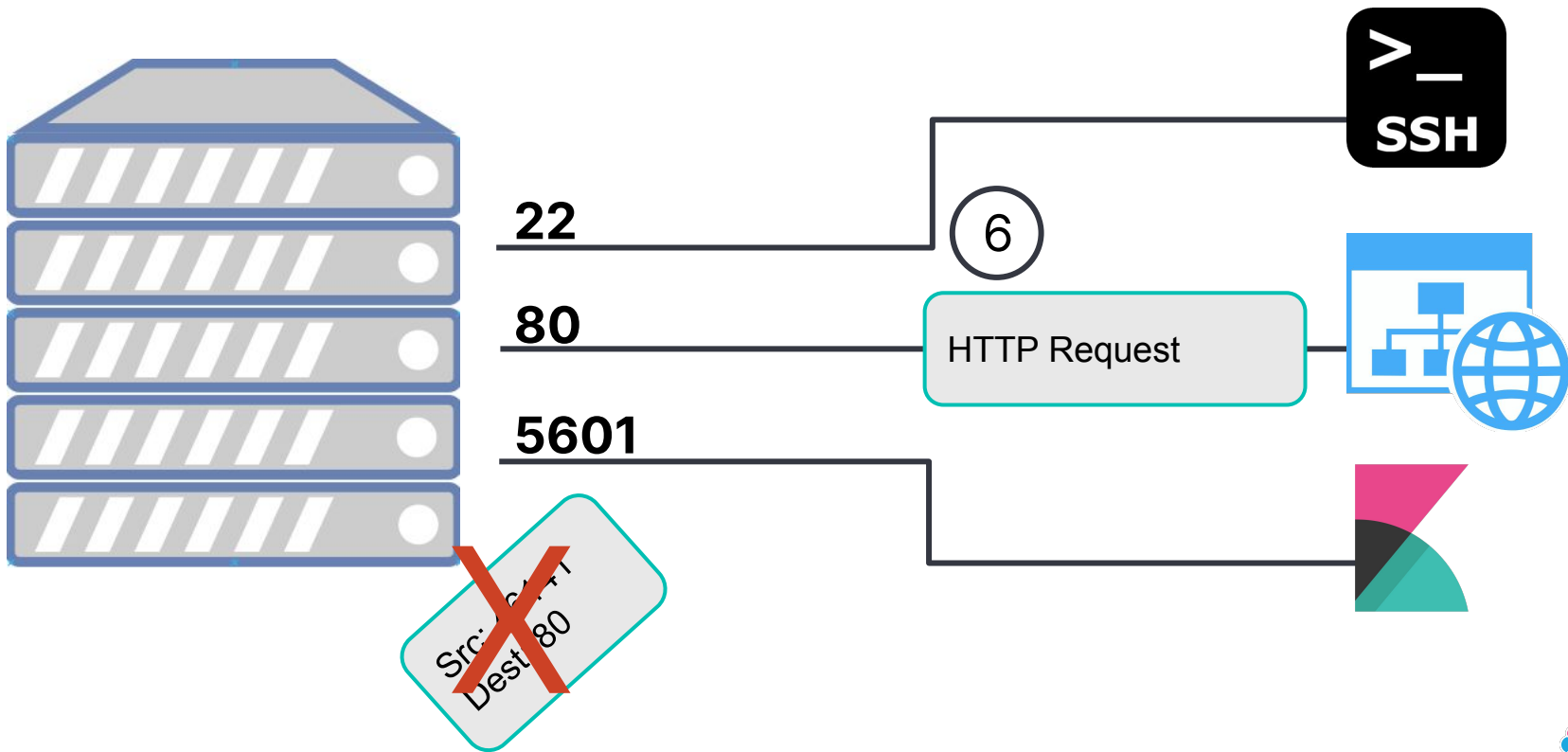
Journey of a Packet

- The packet has now reached the host and can strip the previous layers
- A response packet would follow this same process in reverse

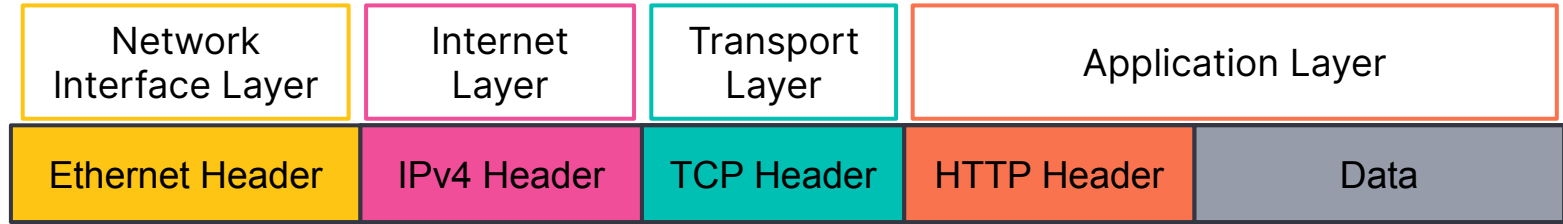


Journey of a Packet

- The port number within the transport layer directs packets to the correct application



Packet Encapsulation - Layer Perspective



CTF: TCP/IP

Packet Analysis

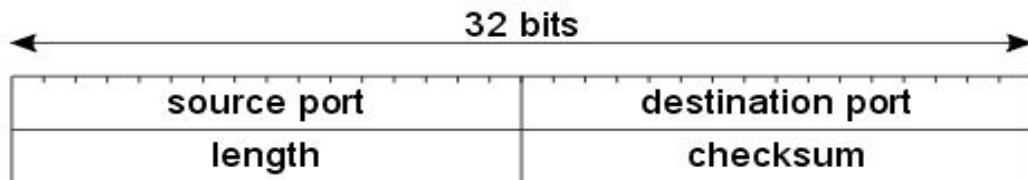
1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

Header Diagrams Explained

Raw Packet (Binary Data)

010011011010100000000000011011111011101011010000100110110101000

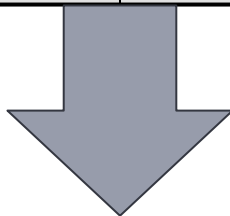
Protocol Header Diagram (Decoder Key)



Header Diagrams Explained

Apply Protocol Header to Raw Data

Source Port	Destination Port
0100 1101 1010 1000	0000 0000 0011 0111
Length	Checksum
1101 1101 0110 1000	1100 1101 1010 1000



Convert from Binary

Source Port	Destination Port
19880	53
Length	Checksum
56680	52648

Ethernet II Frame

Net.

Int.

Transp.

App

	0	1	2	3	4	5	6	7
0	MAC Destination						MAC -->	
8	--- Source				VLAN Tag (optional)			
16	Type/Length				**DATA (Payload)**			
--	Frame Check Sequence							

0x0800	Internet Protocol version 4 (IPv4)
0x0806	Address Resolution Protocol (ARP)
0x86DD	Internet Protocol version 6 (IPv6)

MAC (Hardware) Address



Network Interface Card

00 : 14 : 22 : 01 : 23 : 45

**Organizationally
Unique
Identifier**

NIC Specific

OUI Examples

FC:FC:48 Apple

B4:E9:B0 Cisco

00:50:56 VMware

IPv4 Header

Net.

Int.

Transp.

App

	0	1	2	3	4	5	6	7
0	Ver IHL	TOS	Total Length		IP Identification		X/D/M Offset	
8	TTL	Protocol	Checksum		Source Address			
16	Destination Address				Options (optional)			

Time To Live

- Value that decrements at each router hop.
- Keeps packets from looping forever
- Can be used to fingerprint OS

1	ICMP
6	TCP
8	EGP
17	UDP

ICMP(v4) Header

Net.

Int.

Transp.

App

	0	1	2	3
0	Type	Code	Checksum	
4	Additional Information (dependent of type/code)			

Type		Code	
0	Echo Reply	0	No Code
3	Destination Unreachable	0	Net Unreachable
		1	Host Unreachable
		2	Protocol Unreachable
		3	Port Unreachable
	
8	Echo	0	No Code
11	Time Exceeded	0	Time to Live exceeded in Transit
		1	Fragment Reassembly Time Exceeded

TCP vs UDP



TCP

	0	1	2	3	4	5	6	7
0	Source Port		Destination Port		Sequence Number			
8	Acknowledgement Number				HL R	Flags	WindowSize	
16	Checksum		Urgent Pointer		Options (up to 40 bytes)			

- Connection based
- Flow Control
- File transmission, Email

UDP

	0	1	2	3	4	5	6	7
0	Source Port		Destination Port		Length		Checksum	

- Connection-less
- Fire and forget
- Streaming A/V, Gaming, Small Request/Response

TCP Header

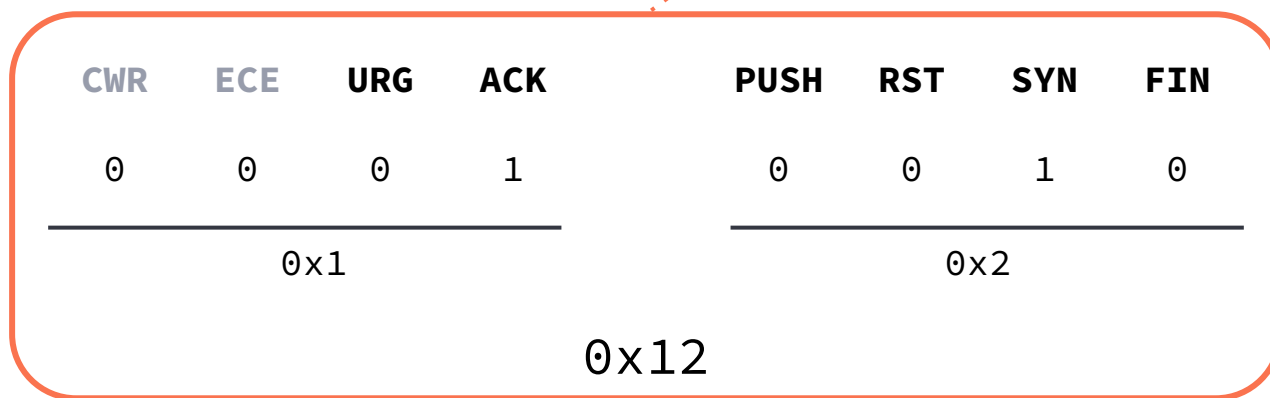
Net.

Int.

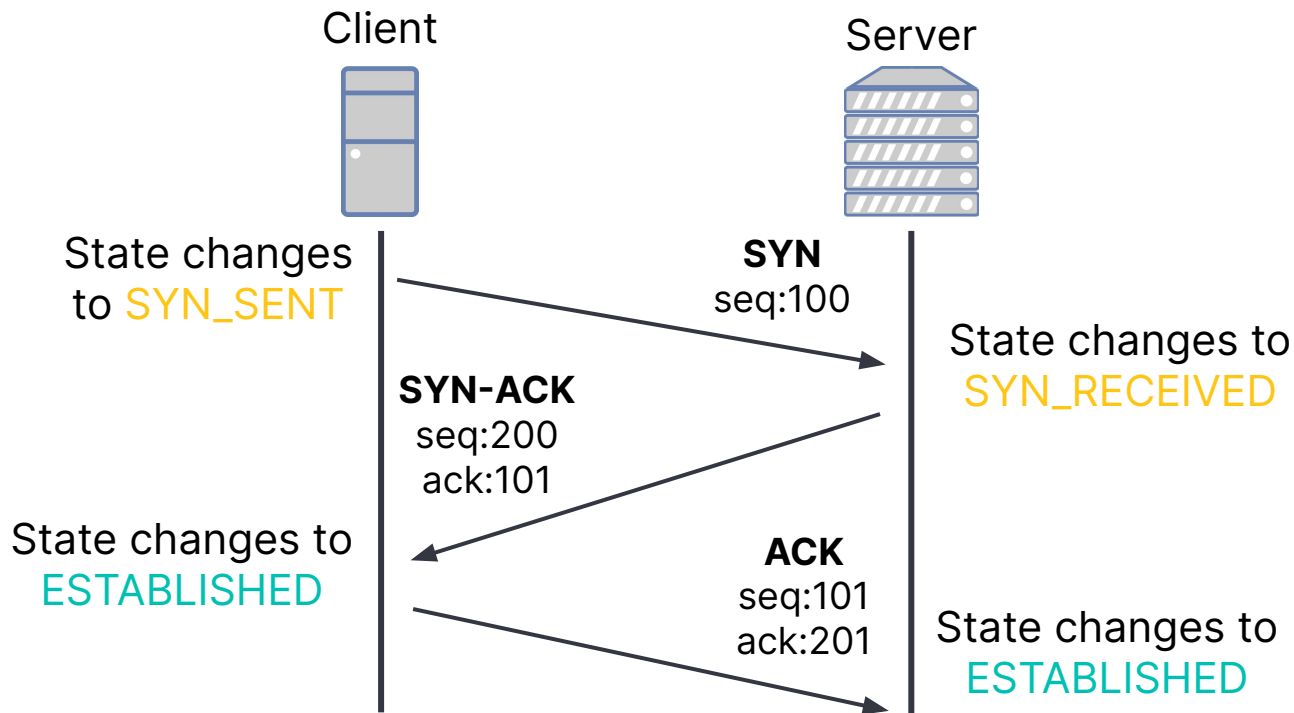
Transp.

App

	0	1	2	3	4	5	6	7
0	Source Port		Destination Port		Sequence Number			
8	Acknowledgement Number				HL R	Flags	Window Size	
16	Checksum		Urgent Pointer		Options (up to 40 bytes)			



TCP 3-Way Handshake



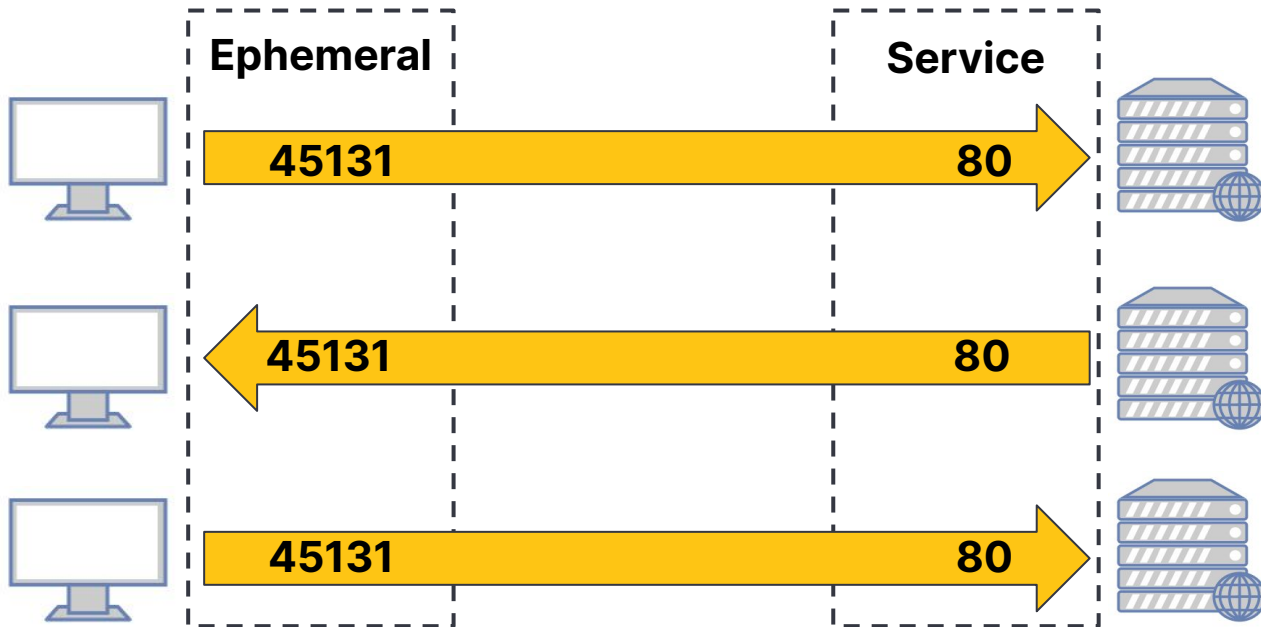
UDP Header



	0	1	2	3
0	Source Port		Destination Port	
4	Length		Checksum	

Ephemeral Ports

- Ephemeral ports are used by clients as source ports as they reach out to server ports
- Higher numbers are set aside to be used as ephemeral ports



Ephemeral Ports

- Short-lived transport protocol port
- Ephemeral Port \longleftrightarrow Service Port

Standard?	Ports
IANA Suggestion	49152 - 65535
“Modern” Windows (Vista, 7, Server 2008 and up)	49152 - 65535
Windows Server 2008	1025 - 60000
“Older” Windows (XP and older)	1025 - 5000
Most Linux kernels	32768 - 61000

Packet Analysis

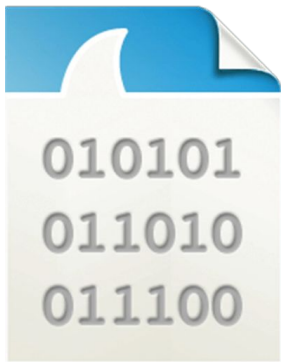
1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

Wireshark

- Great GUI tool to deep dive into packets
- Limited usability on larger pcap files
- Parses over 2,000 network protocols
- Contains 180,000+ fields

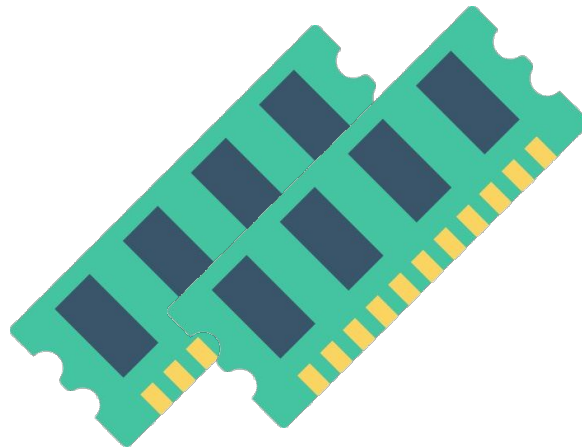


Wireshark Resource Cost



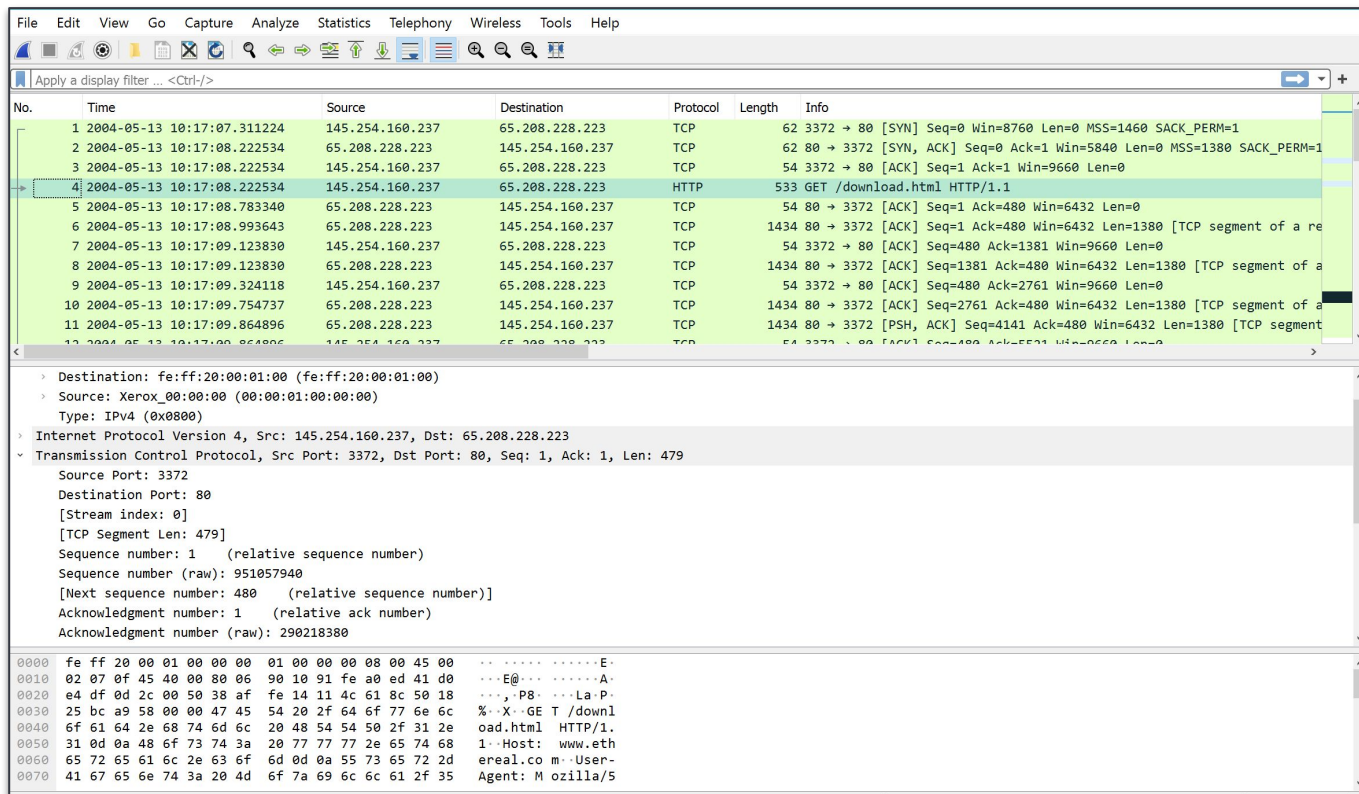
1 GB PCAP File

x4



**4GB RAM
Utilization**

Wireshark Basic Interface



The screenshot displays the Wireshark Basic Interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. The main window is divided into three sections: Packet List, Packet Details, and Packet Bytes.

Packet List: This section shows a list of captured packets. The columns are No., Time, Source, Destination, Protocol, Length, and Info. The selected packet is packet 4, which is an HTTP GET request to /download.html.

No.	Time	Source	Destination	Protocol	Length	Info
1	2004-05-13 10:17:07.311224	145.254.160.237	65.208.228.223	TCP	62	3372 → 80 [SYN] Seq=0 Win=8760 Len=0 MSS=1460 SACK_PERM=1
2	2004-05-13 10:17:08.222534	65.208.228.223	145.254.160.237	TCP	62	80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380 SACK_PERM=1
3	2004-05-13 10:17:08.222534	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=1 Ack=1 Win=9660 Len=0
4	2004-05-13 10:17:08.222534	145.254.160.237	65.208.228.223	HTTP	533	GET /download.html HTTP/1.1
5	2004-05-13 10:17:08.783340	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=0
6	2004-05-13 10:17:08.993643	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=1380 [TCP segment of a re
7	2004-05-13 10:17:09.123830	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=1381 Win=9660 Len=0
8	2004-05-13 10:17:09.123830	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1381 Ack=480 Win=6432 Len=1380 [TCP segment of a
9	2004-05-13 10:17:09.324118	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=2761 Win=9660 Len=0
10	2004-05-13 10:17:09.754737	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=2761 Ack=480 Win=6432 Len=1380 [TCP segment of a
11	2004-05-13 10:17:09.864896	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=4141 Ack=480 Win=6432 Len=1380 [TCP segment
12	2004-05-13 10:17:09.864896	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=5571 Win=9660 Len=0

Packet Details: This section shows the details of the selected packet (packet 4). It includes the destination IP address (fe:ff:20:00:01:00), source IP address (Xerox_00:00:00 (00:00:01:00:00:00)), and the Transmission Control Protocol (TCP) details, including the source port (3372), destination port (80), sequence number (1), and acknowledgment number (1).

Packet Bytes: This section shows the raw bytes of the selected packet. It displays the hexadecimal representation of the packet data, along with the corresponding ASCII representation. The packet data starts with the Ethernet II header, followed by the Internet Protocol Version 4 header, and the Transmission Control Protocol header.

Packet List

Packet Details

Packet Bytes

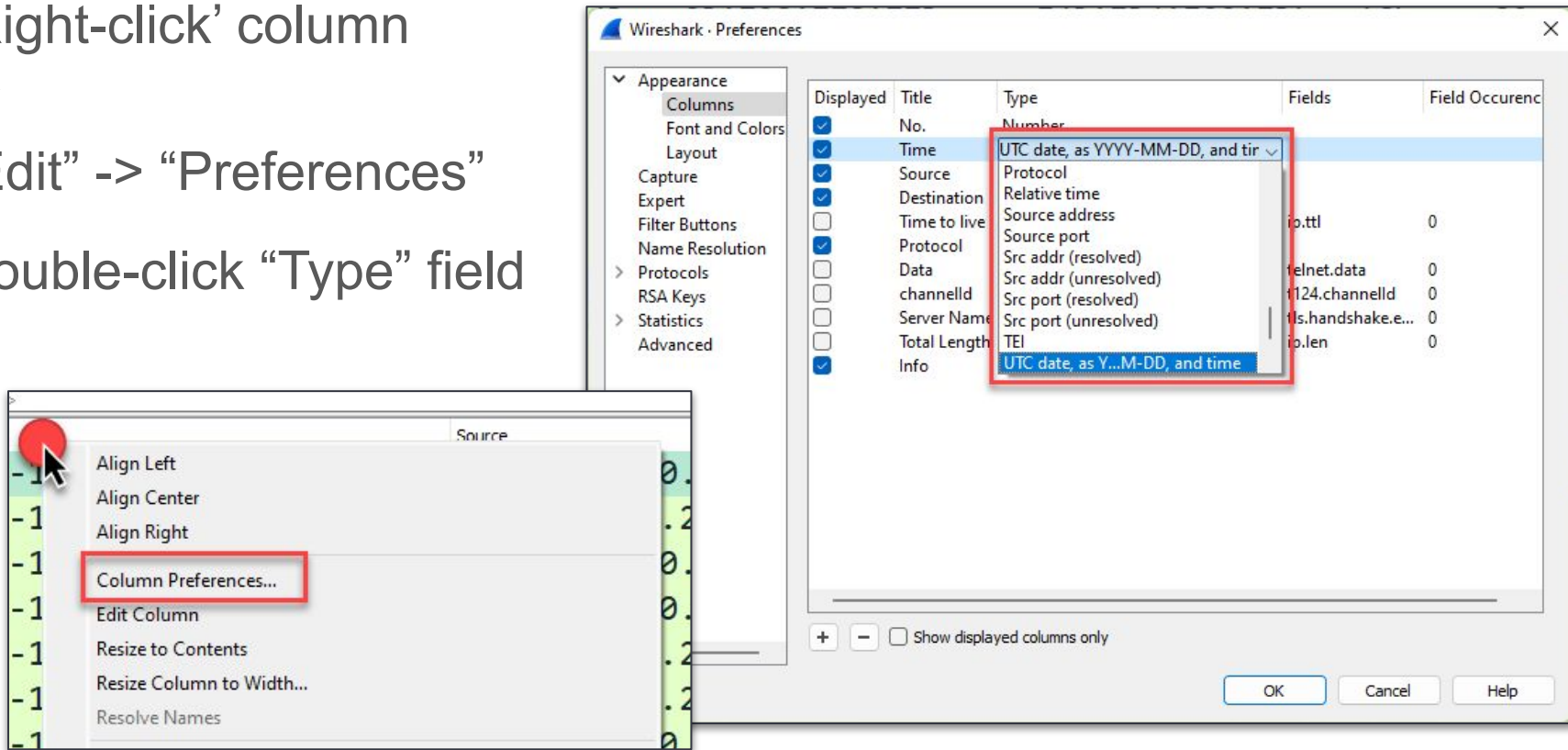
Packet List Frame

- Displays an overview list of all the packets

No.	Time	Source	Destination	Protocol	Info
1	2004-05-13 10:17:07.311224	145.254.160.237	65.208.228.223	TCP	3372 → 80 [SYN] Seq=0 Win=8760 Len=0 MSS=1460 SACK_PERM=1
2	2004-05-13 10:17:08.222534	65.208.228.223	145.254.160.237	TCP	80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380 SACK_PERM=1
3	2004-05-13 10:17:08.222534	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=1 Ack=1 Win=9660 Len=0
4	2004-05-13 10:17:08.222534	145.254.160.237	65.208.228.223	HTTP	GET /download.html HTTP/1.1
5	2004-05-13 10:17:08.783340	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=0
6	2004-05-13 10:17:08.993643	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=1380 [TCP segment of a re
7	2004-05-13 10:17:09.123830	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=1381 Win=9660 Len=0
8	2004-05-13 10:17:09.123830	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=1381 Ack=480 Win=6432 Len=1380 [TCP segment of a
9	2004-05-13 10:17:09.324118	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=2761 Win=9660 Len=0
10	2004-05-13 10:17:09.754737	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=2761 Ack=480 Win=6432 Len=1380 [TCP segment of a
11	2004-05-13 10:17:09.864896	65.208.228.223	145.254.160.237	TCP	80 → 3372 [PSH, ACK] Seq=4141 Ack=480 Win=6432 Len=1380 [TCP segment
12	2004-05-13 10:17:09.864896	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=5521 Win=9660 Len=0
13	2004-05-13 10:17:09.864896	145.254.160.237	145.253.2.203	DNS	Standard query 0x0023 A pagead2.google syndication.com
14	2004-05-13 10:17:09.945011	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=5521 Ack=480 Win=6432 Len=1380 [TCP segment of a
15	2004-05-13 10:17:10.125270	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=6901 Win=9660 Len=0
16	2004-05-13 10:17:10.205385	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=6901 Ack=480 Win=6432 Len=1380 [TCP segment of a
17	2004-05-13 10:17:10.225414	145.253.2.203	145.254.160.237	DNS	Standard query response 0x0023 A pagead2.google syndication.com CNAME
18	2004-05-13 10:17:10.295515	145.254.160.237	216.239.59.99	HTTP	GET /pagead/ads?client=ca-pub-2309191948673629&random=10844434302858
19	2004-05-13 10:17:10.325558	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=8281 Win=9660 Len=0
20	2004-05-13 10:17:10.686076	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=8281 Ack=480 Win=6432 Len=1380 [TCP segment of a
21	2004-05-13 10:17:10.806249	65.208.228.223	145.254.160.237	TCP	80 → 3372 [PSH, ACK] Seq=9661 Ack=480 Win=6432 Len=1380 [TCP segment
22	2004-05-13 10:17:10.806249	145.254.160.237	65.208.228.223	TCP	3372 → 80 [ACK] Seq=480 Ack=11041 Win=9660 Len=0
23	2004-05-13 10:17:10.946451	65.208.228.223	145.254.160.237	TCP	80 → 3372 [ACK] Seq=11041 Ack=480 Win=6432 Len=1380 [TCP segment of
24	2004-05-13 10:17:10.956465	216.239.59.99	145.254.160.237	TCP	80 → 3371 [ACK] Seq=1 Ack=722 Win=31460 Len=0

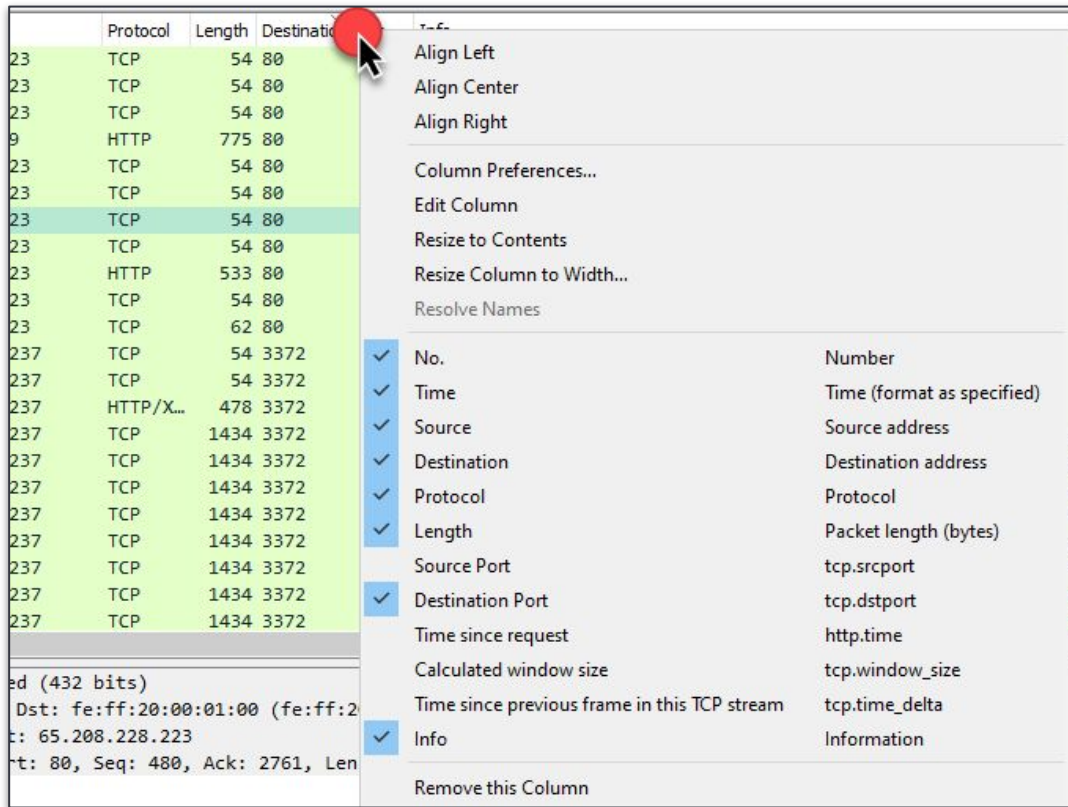
Time Preferences

- ‘Right-click’ column or
or
“Edit” -> “Preferences”
- Double-click “Type” field



Changing available columns

- Quickly add or remove columns from display



The screenshot shows a network packet capture viewer with a table of packets. A right-click context menu is open over the 'Destination' column header. The menu includes options for alignment, column preferences, editing, and resizing. A secondary menu is also visible, listing various fields that can be added to the display, each with a checkbox and a description.

	Protocol	Length	Destination
23	TCP	54	80
23	TCP	54	80
23	TCP	54	80
9	HTTP	775	80
23	TCP	54	80
23	TCP	54	80
23	TCP	54	80
23	TCP	54	80
23	TCP	54	80
23	HTTP	533	80
23	TCP	54	80
23	TCP	62	80
237	TCP	54	3372
237	TCP	54	3372
237	HTTP/X...	478	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372
237	TCP	1434	3372

ed (432 bits)
Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
Src: 65.208.228.223
Port: 80, Seq: 480, Ack: 2761, Len: 54

Context Menu Options:

- Align Left
- Align Center
- Align Right
- Column Preferences...
- Edit Column
- Resize to Contents
- Resize Column to Width...
- Resolve Names

Available Fields:

- ☒ No. Number
- ☒ Time Time (format as specified)
- ☒ Source Source address
- ☒ Destination Destination address
- ☒ Protocol Protocol
- ☒ Length Packet length (bytes)
- Source Port tcp.srcport
- ☒ Destination Port tcp.dstport
- Time since request http.time
- Calculated window size tcp.window_size
- Time since previous frame in this TCP stream tcp.time_delta
- ☒ Info Information
- Remove this Column

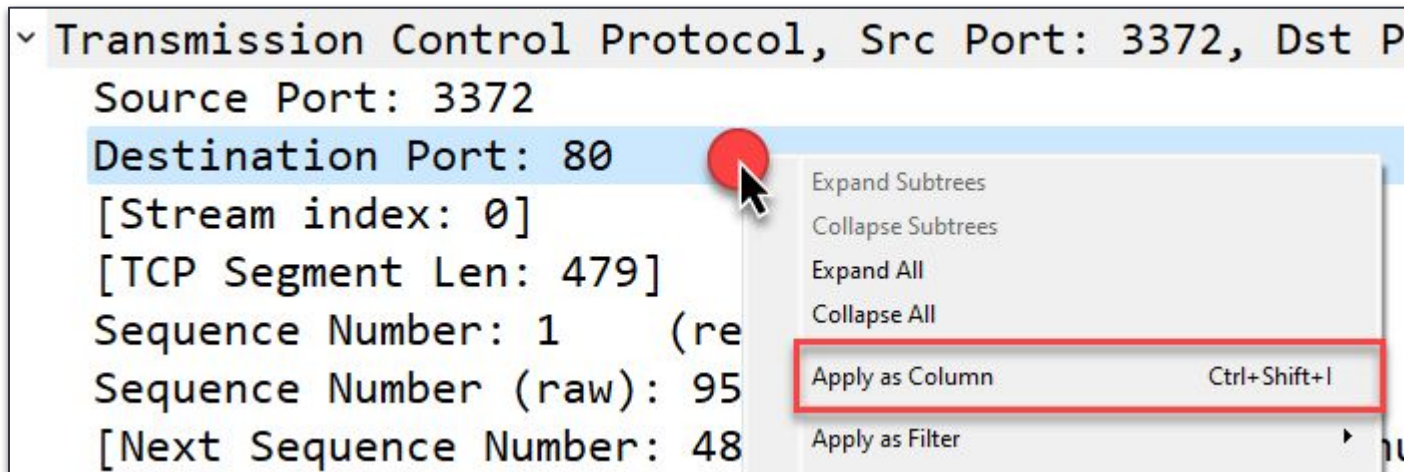
Packet Details Frame

- Displays full details grouped by layers

```
> Frame 4: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits)
> Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
> Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 519
  Identification: 0x0f45 (3909)
> Flags: 0x40, Don't fragment
  Fragment Offset: 0
  Time to Live: 128
  Protocol: TCP (6)
  Header Checksum: 0x9010 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 145.254.160.237
  Destination Address: 65.208.228.223
> Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 1, Ack: 1, Len: 479
> Hypertext Transfer Protocol
```

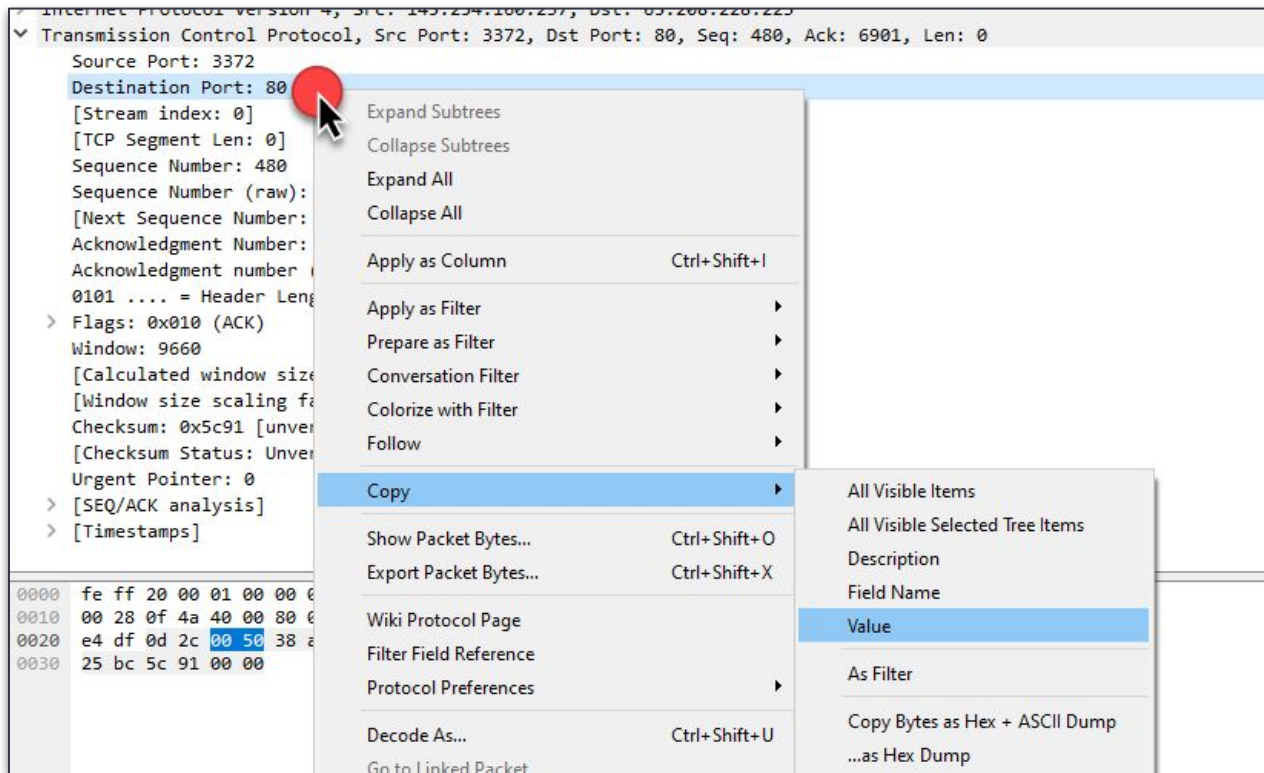
Apply as Column

- 'Right-click' any field
- Adds the column as a quick-view to the packet list



Copying Values

- Copies field values into clipboard for documenting



Packet Bytes Frame

Hexadecimal

ASCII

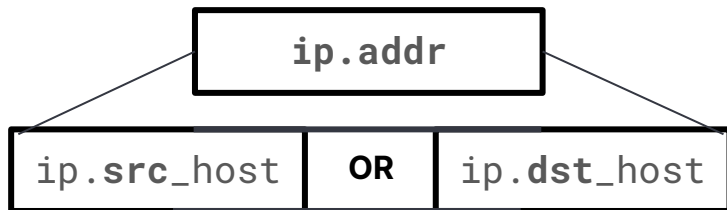
	Hexadecimal																ASCII														
0020	e4	df	0d	2c	00	50	38	af	fe	14	11	4c	61	8c	50	18	...	,	P8	...	La	P									
0030	25	bc	a9	58	00	00	47	45	54	20	2f	64	6f	77	6e	6c	%	X	GE	T	/downl										
0040	6f	61	64	2e	68	74	6d	6c	20	48	54	54	50	2f	31	2e	oad.html				HTTP/1.										
0050	31	0d	0a	48	6f	73	74	3a	20	77	77	77	2e	65	74	68	1	Host:			www.eth										
0060	65	72	65	61	6c	2e	63	6f	6d	0d	0a	55	73	65	72	2d	ereal.co	m	User-												
0070	41	67	65	6e	74	3a	20	4d	6f	7a	69	6c	6c	61	2f	35	Agent: M	ozilla/5													
0080	2e	30	20	28	57	69	6e	64	6f	77	73	3b	20	55	3b	20	.0 (Wind	ows; U;													
0090	57	69	6e	64	6f	77	73	20	4e	54	20	35	2e	31	3b	20	Windows		NT 5.1;												
00a0	65	6e	2d	55	53	3b	20	72	76	3a	31	2e	36	29	20	47	en-US; r	v:1.6) G													
00b0	65	63	6b	6f	2f	32	30	30	34	30	31	31	33	0d	0a	41	ecko/200		40113	A											
00c0	63	63	65	70	74	3a	20	74	65	78	74	2f	78	6d	6c	2c	ccept: t		ext/xml,												

CTF: Packet Encapsulation

Wireshark Syntax

Basic Fields (Too many to cover here!)

- ip.addr
- ip.src_host
- ip.dst_host
- tcp.dstport
- tcp.srcport
- http.request.method



Operators

- == eq
- contains
- <
- >
- ! not
- && and
- || or

https://packetlife.net/media/library/13/Wireshark_Display_Filters.pdf

Wireshark examples

```
# HTTP packets (Wireshark does port-association for protocols)
```

```
http
```

```
# DNS packets using TCP for transport (DNS typically uses UDP)
```

```
dns and tcp
```

```
# Packets that have a source or destination IP of 1.2.3.4 and not TCP port 53
```

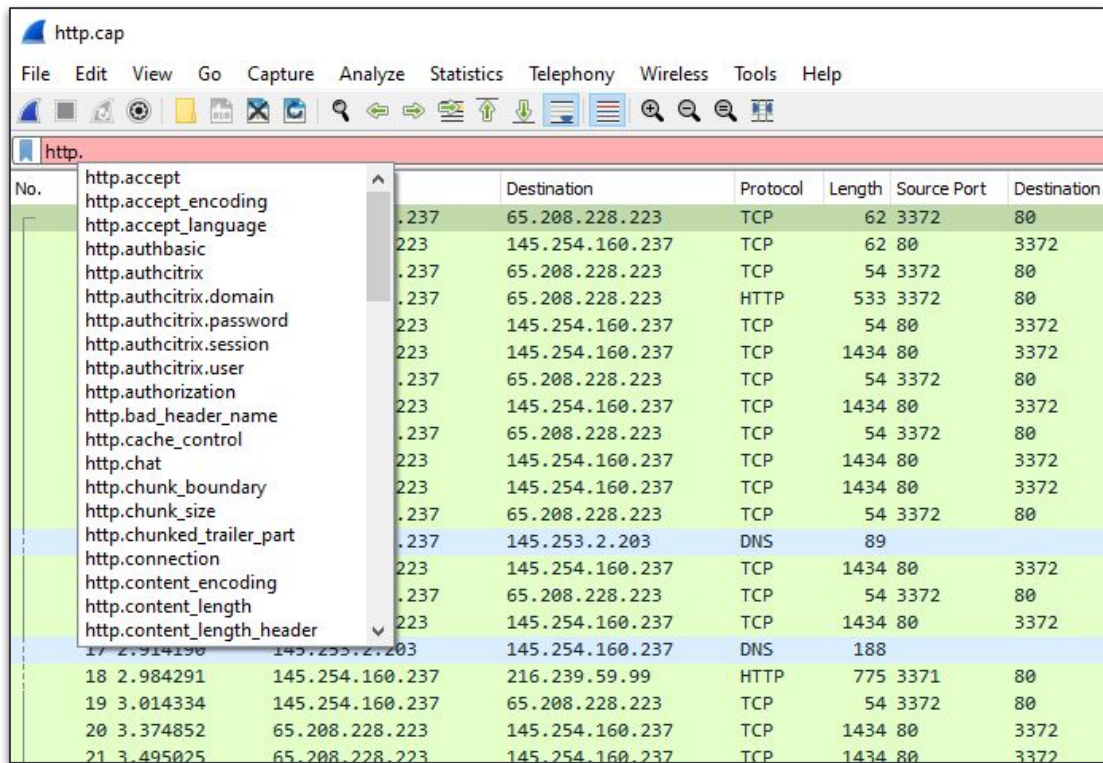
```
ip.addr == 1.2.3.4 and not tcp.port == 53
```

```
# Packets coming from external to internal IPs
```

```
not ip.src_host == 10.0.0.0/8 and ip.dst_host == 10.0.0.0/8
```

Auto-fill Syntax

- Fields are “nested” and will auto-populate with available options

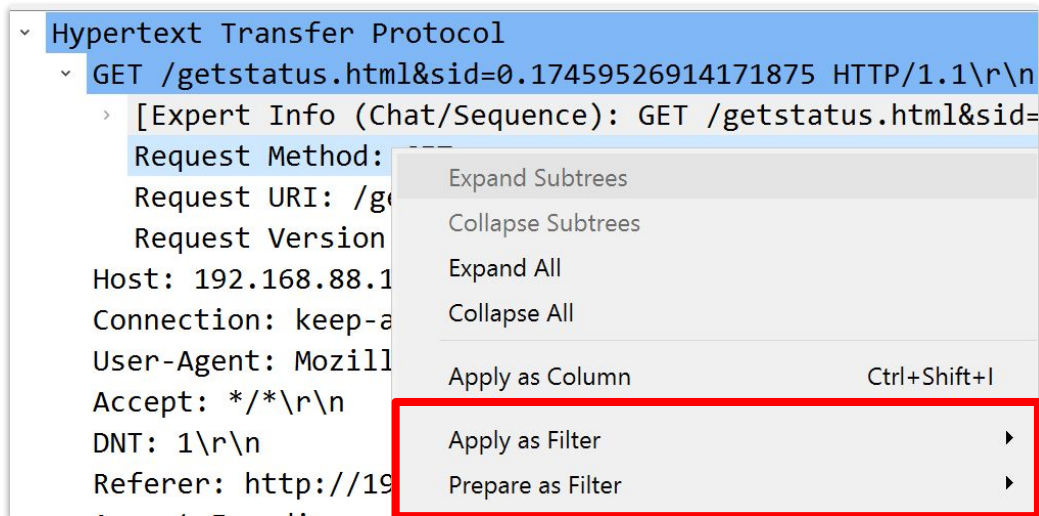


The screenshot shows the Wireshark interface with the 'http.' field list on the left and the packet list on the right. The field list shows a tree structure of HTTP fields, including 'http.accept', 'http.accept_encoding', 'http.accept_language', 'http.authbasic', 'http.authcitrix', 'http.authcitrix.domain', 'http.authcitrix.password', 'http.authcitrix.session', 'http.authcitrix.user', 'http.authorization', 'http.bad_header_name', 'http.cache_control', 'http.chat', 'http.chunk_boundary', 'http.chunk_size', 'http.chunked_trailer_part', 'http.connection', 'http.content_encoding', 'http.content_length', and 'http.content_length_header'. The packet list shows a table of captured packets with columns for No., Destination, Protocol, Length, Source Port, and Destination.

No.	Destination	Protocol	Length	Source Port	Destination
17	65.208.228.223	TCP	62	3372	80
223	145.254.160.237	TCP	62	80	3372
237	65.208.228.223	TCP	54	3372	80
237	65.208.228.223	HTTP	533	3372	80
223	145.254.160.237	TCP	54	80	3372
223	145.254.160.237	TCP	1434	80	3372
237	65.208.228.223	TCP	54	3372	80
223	145.254.160.237	TCP	1434	80	3372
237	65.208.228.223	TCP	54	3372	80
223	145.254.160.237	TCP	1434	80	3372
237	65.208.228.223	TCP	54	3372	80
237	145.253.2.203	DNS	89		
223	145.254.160.237	TCP	1434	80	3372
237	65.208.228.223	TCP	54	3372	80
223	145.254.160.237	TCP	1434	80	3372
17	145.253.2.203	DNS	188		
18	216.239.59.99	HTTP	775	3371	80
19	65.208.228.223	TCP	54	3372	80
20	145.254.160.237	TCP	1434	80	3372
21	145.254.160.237	TCP	1434	80	3372

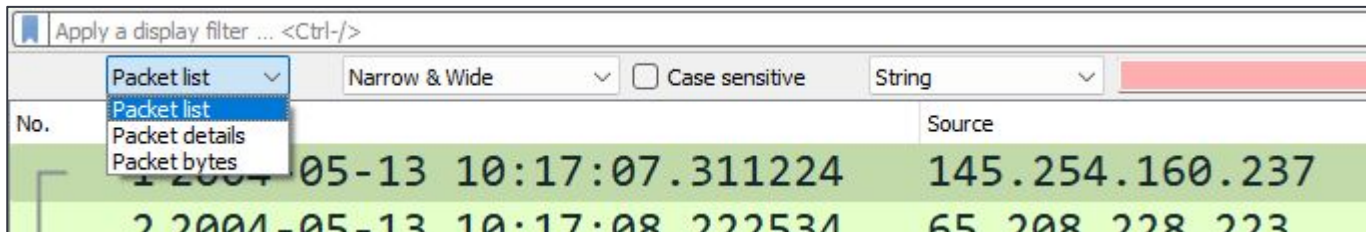
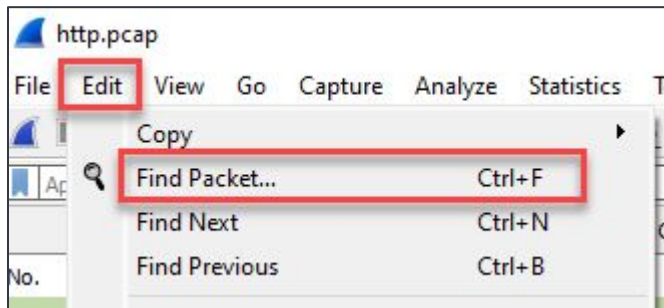
Wireshark: Right Click → Apply/Prepare As Filter

- **Apply as Filter** - Create and apply the a query in search bar
- **Prepare as Filter** - Create query in search bar but do not apply it



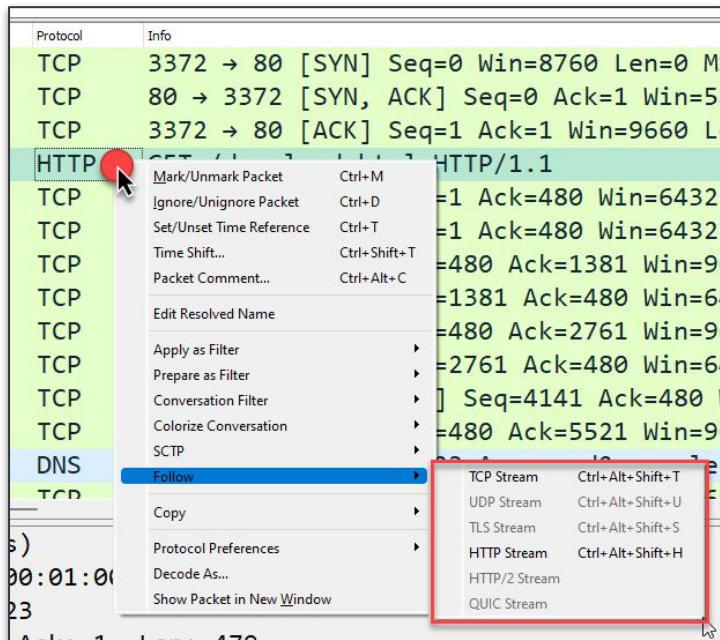
Find

- “Standard” search
 - Search all three panes
 - String or Regex options available

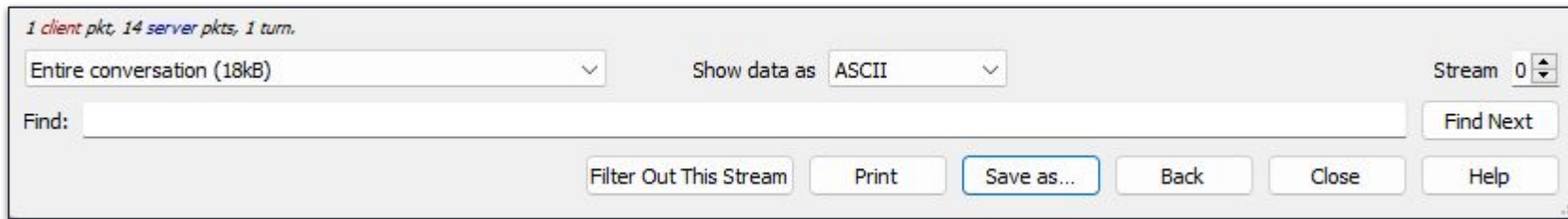


Wireshark: Right Click → Follow (TCP/UDP/HTTP)

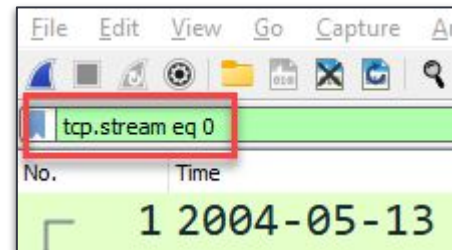
- Follows a conversation across multiple packets
- Displays the application payload



TCP Stream - Continued



- Extra Options!
 - Display format
 - Find
 - Moving through streams
- Automatically applies a filter



Misc

- Statistics Menu
 - High-level metadata about protocols and endpoints
- Live Capture
 - Live capture from any interface (ethernet, Wi-Fi, Bluetooth, or USB)
- File Extraction
 - For certain protocols (http, smb, tftp, etc.)
- Protocol Decryption
 - Can decrypt protocols if you load in the appropriate key file
- 'tshark'
 - CLI companion utility (same filter syntax - much lower memory requirements)

CTF: More PCAP!

Packet Analysis

1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

TCPdump

- Created in 1988 by researchers at Lawrence Berkeley Labs
- Literally dumps TCP packets
- Can **capture** from interface or **read** from file
- Can write **summary data** to screen or full PCAP to file

TCPdump Options Flags

Option flags are mainly used to change the output of TCPdump

#Basic TCPdump Syntax

tcpdump <OPTIONS> -r <FILE>

- r read file
- c X limit the results to the first X number of packets that match the filter
- v(vv) verbose, adds more information to the output of tcpdump (has three levels of verbosity v, vv, and vvv)
- n no name resolution on IP addresses
- nn no name resolution on IPs or port numbers

TCPdump - options examples

#Read the file traffic.pcap

```
tcpdump -r traffic.pcap
```

```
04:17:07.311224 IP dialin-145-254-160-237.pools.arcor-ip.net.3372 > 65.208.228.223.http: Flags [S], seq 951057939, win 8760, options [mss 1460,nop,nop,sackOK], length 0
```

#Read the file traffic.pcap but limit it to the first 10 packets. Do not resolve IPs or ports.

```
tcpdump -nn -c 10 -r traffic.pcap
```

```
04:17:07.311224 IP 145.254.160.237.3372 > 65.208.228.223.80: Flags [S], seq 951057939, win 8760, options [mss 1460,nop,nop,sackOK], length 0
```

#Read the file traffic.pcap and limit it to the first 10 packets at the highest level of verbosity.

```
tcpdump -c 10 -vvv -r traffic.pcap
```

```
04:17:07.311224 IP (tos 0x0, ttl 128, id 3905, offset 0, flags [DF], proto TCP (6), length 48)  
    dialin-145-254-160-237.pools.arcor-ip.net.3372 > 65.208.228.223.http: Flags [S], cksum 0xc30c (correct), seq 951057939, win 8760, options [mss 1460,nop,nop,sackOK], length 0
```

TCPdump filtering - BPF

- Simple query language to filter on network traffic
- Provides raw interface to data link layers
- Extremely efficient when run in kernel space - Fast!
- Can be used along TCPdump Option Flags

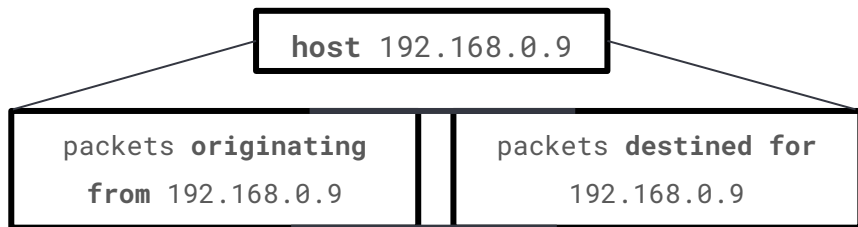
BPF Syntax

- Basic Tokens

- Host IPv4 Address `'host 192.168.0.9'`
- Net CIDR Address `'net 192.168.0.0/16'`
- Port TCP/UDP Port `'port 53'`

- Protocol Tokens

- icmp `'icmp'`
- tcp `'tcp'`
- udp `'udp'`



BPF Syntax

- Direction Operators

- src Originating host/net/port '**src** host 10.0.0.32'
- dst Receiving host/net/port '**dst** port 53'

- Logical Tokens

- or || 'dst port 80 **or** dst port 443'
- and && 'host 10.0.0.3 **and** src port 53'
- not ! '**not** net 192.168.0.0/16'

TCPdump + BPF: Basic Query Function

#Basic TCPdump/BPF Syntax

tcpdump <OPTIONS> -r <FILE> '<BPF FILTER>'

#All TCP packets destined for 8.8.8.8 to port 53

tcpdump -r file.pcap 'tcp and dst host 8.8.8.8 and dst port 53'

#All non-ICMP packets heading from 10.0.0.7 to 10.0.0.3

tcpdump -r traffic.pcap 'src host 10.0.0.7 and dst host 10.0.0.3 and not icmp'

#All packets originating from outside the internal network and do not resolve IP/ports

tcpdump -nn -r traffic.pcap 'not src net 192.168.1.0/24'

#Query functions:

host 8.8.8.8
net 1.0.0.0/8
port 80
icmp
tcp
udp

#Direction operators:

src
dst

#Query operators:

or ||
and &&
not !

CTF: TCPdump & Berkeley Packet Filters

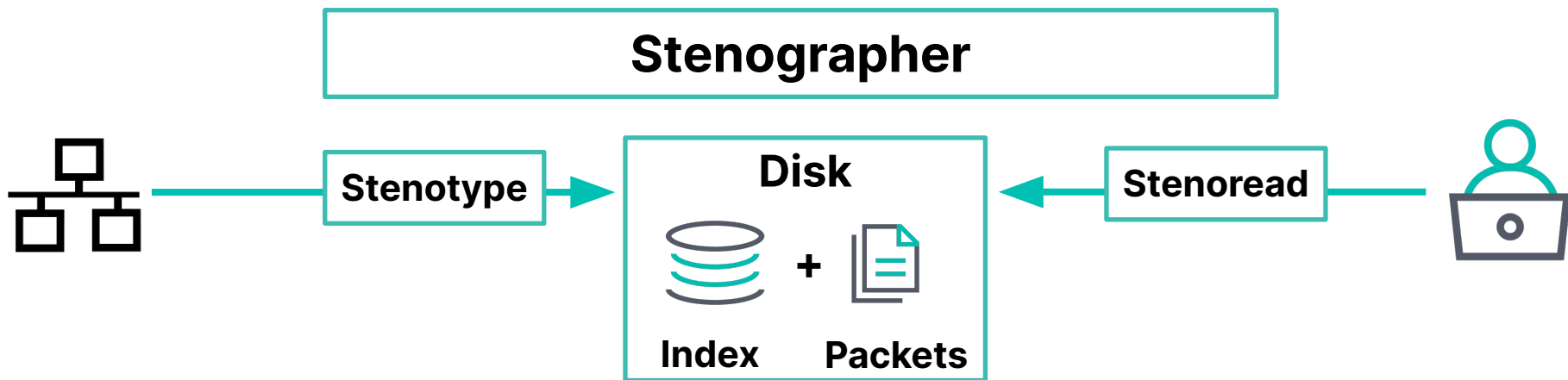
Packet Analysis

1. TCP/IP Model
2. Protocol Basics
3. Wireshark
4. TCPdump/BPF
5. Stenographer/Docket

Google Stenographer

- 80/20 project by Graeme Connell
- High speed packet capture and index
- Highly Reliable
- Automatically Manages Disk space
- Written in Go

Process Architecture - Simplified



Stenoread

Stenoread is the command used to query Stenographer for packets via the Application Programming Interface (API)

- Uses subset of BPF, **CANNOT specify direction (src/dst)**
- After retrieving packets, stenoread passes them over to tcpdump to display to user
- Supports tcpdump options

Stenoread - Basic Query Function

#Basic Syntax

stenoread '<STENO OPTIONS>'

#Return all packets involving host 8.8.8.8

stenoread 'host 8.8.8.8'

#Return all TCP packets involving hosts in 10.0.0.0/8

stenoread 'tcp and net 10.0.0.0/8'

#Return all UDP packets involving host 1.2.3.4 and 5.6.7.8

stenoread 'udp and host 1.2.3.4 and host 5.6.7.8'

#Basic query functions:

host 8.8.8.8
net 1.0.0.0/8
port 80
icmp
tcp
udp

#Basic query operators:

or ||
and &&

Stenoread - Time Function

#Stenographer-specific time additions:

<code>before 2012-11-03T11:05:00Z</code>	# Packets before a specific time (UTC)
<code>after 2012-11-03T11:05:00-07:00</code>	# Packets after a specific time (with TZ)
<code>before 45m ago</code>	# Packets before a relative time
<code>after 3h ago</code>	# Packets after a relative time

#Return all packets involving host 8.8.8.8 from the last 2 hours

stenoread 'host 8.8.8.8 and after 2h ago'

#Return all packets from a specific 6 hours

stenoread 'before 2020-11-03T11:05:00Z and after 2020-11-03T05:05:00Z'

#Return all UDP packets involving host 1.2.3.4 and 5.6.7.8

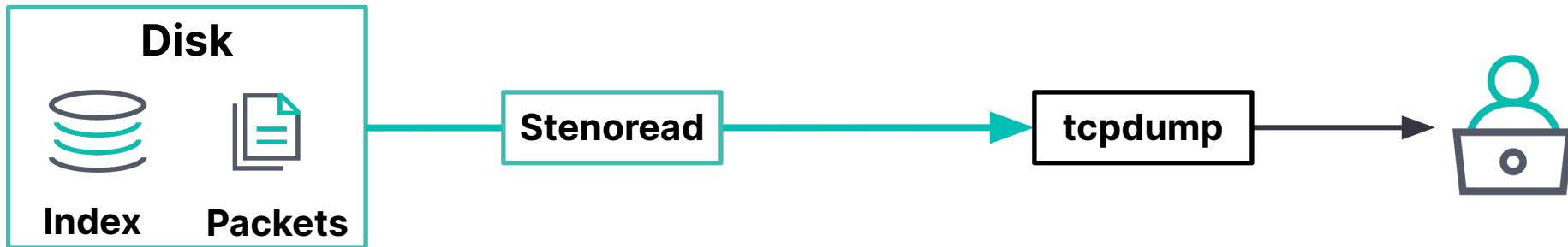
stenoread 'udp and host 1.2.3.4 and host 5.6.7.8'

Stenographer + TCPdump Options

- The results from stenoread are given to tcpdump to display to the user
- Able to use filters and display flags
- Allows to specify direction

```
stenoread 'host 8.8.8.8 and tcp' -nn 'src host 8.8.8.8'
```

stenoread query + tcpdump options/filters



Stenographer + TCPdump Examples

All packets originating from 1.1.1.1 without DNS resolution (-n).

```
stenoread 'host 1.1.1.1' -n 'src host 1.1.1.1'
```

UDP packets from 10.0.0.7 to 2.2.2.2 without DNS or port resolution (-nn):

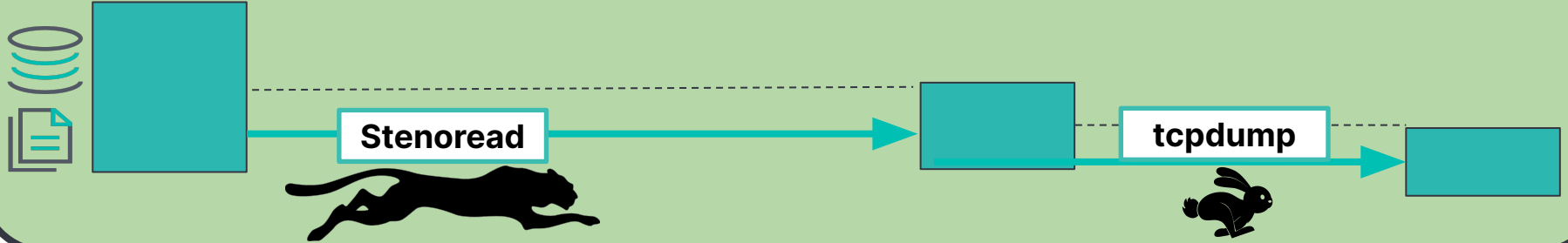
```
stenoread 'udp and host 10.0.0.7 and host 2.2.2.2' -nn 'src host 10.0.0.7 and  
dst host 2.2.2.2'
```

Write all packets involving 1.1.1.1 from the last three hours to a pcap.

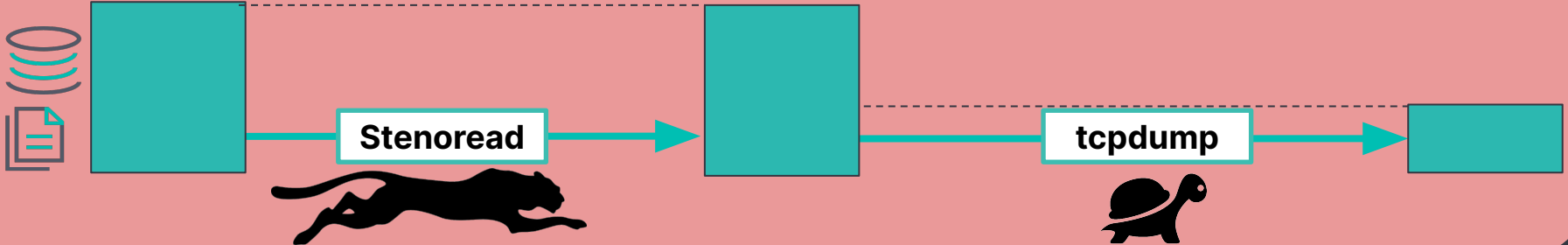
```
stenoread 'host 1.1.1.1 and after 3h ago' -w /tmp/out.pcap
```

Stenographer + TCPdump Efficiency

```
stenoread 'host 8.8.8.8 and tcp' -nn 'src host 8.8.8.8'
```



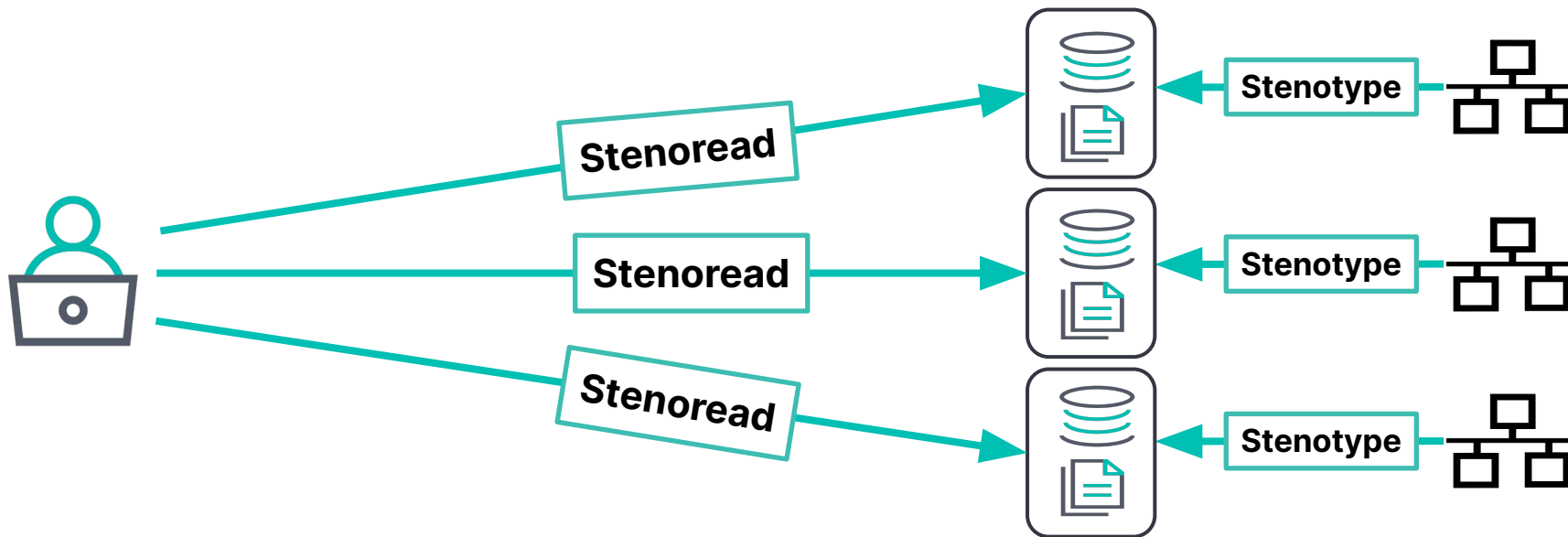
```
stenoread 'net 0.0.0.0/0' -nn 'src host 8.8.8.8 and tcp'
```



CTF: Stenographer

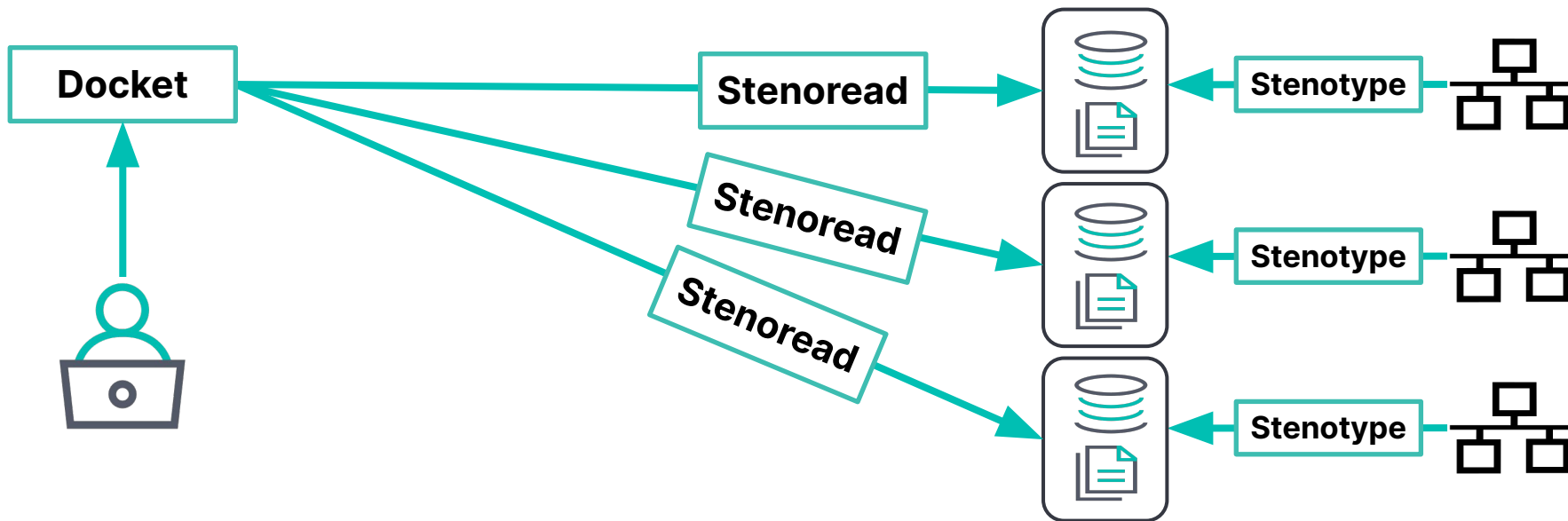
Stenoread Across Multiple Sensors

- Time Intensive - Logging on to each individual machine
- Not scalable - Returning multiple PCAPS over possibly 100's of machines



Docket

- Sends stenoread API calls to multiple machines
- Returns single PCAP to user
- Reduces Analyst Overhead



Docket -> Stenoread

```
curl http://docket/api/uri/host/1.2.3.4/udp/port/53/  
stenoread 'host 1.2.3.4 and udp and port 53'
```



Docket Web Front

Absolute Time

Relative Time



After

Date/Time

Before

Date/Time



Host Query

Network Query



First Host

Second Host


First Port

Second Port

Protocol



Stenoread Time Options

- Tab for absolute and relative time

Stenoread Host Options

- Still cannot specify src/dst
- Tab for CIDR range

Stenoread Port/Protocol Options

- Still cannot specify src/dst

CTF: Docket

Quiz - Part 1

1. Stenographer is written in what language?
2. Which command line tool allows you to query Stenographer's indexed packets?
3. True or False: When querying Stenographer for packets, you can use tcpdump filters.

Quiz - Part 2

4. Which of the following is an additional query option that is available in Stenographer but not available in tcpdump?

- a before 2012-11-03T11:05:00Z
- b host 1.1.1.1
- c tcp[tcpflags] & (tcp-syn|tcp-fin)
- d net 2.2.2.2/24

5. Which of the following is NOT true about PCAP?

- a allows the analyst to reconstruct what actually happened
- b PCAP scales well... the more the better!
- c contains all the raw data

Quiz - Part 3

6. True or False: Stenographers indexed packets on disk are not in tcpdump format because they have an additional 4 byte header.
7. Analyzing PCAP is a memory intensive task. What is the general memory usage multiplier?
8. What does BPF stand for?
9. What is Docket?
10. True or False: BPF is more efficient than text filtering tools, like grep?