

Network Traffic Evolution

Prof. Anja Feldmann, Ph.D.

Dr. Nikolaos Chatzis

- ❑ Keelback.net.t-labs.tu-berlin.de

- ❑ Kingsnake

- ❑ Oilsnake

- ❑ Python

- ❑ Treesnake

- ❑ Username: measurement12

- ❑ Password: meter

Example trace

Name	port	% bytes	% packets	bytes per packet
world-wide-web	80	????	????	????
netnews	119	????	????	????
pop-3 mail	110	????	????	????
. . .				

□ How?

Passive measurements

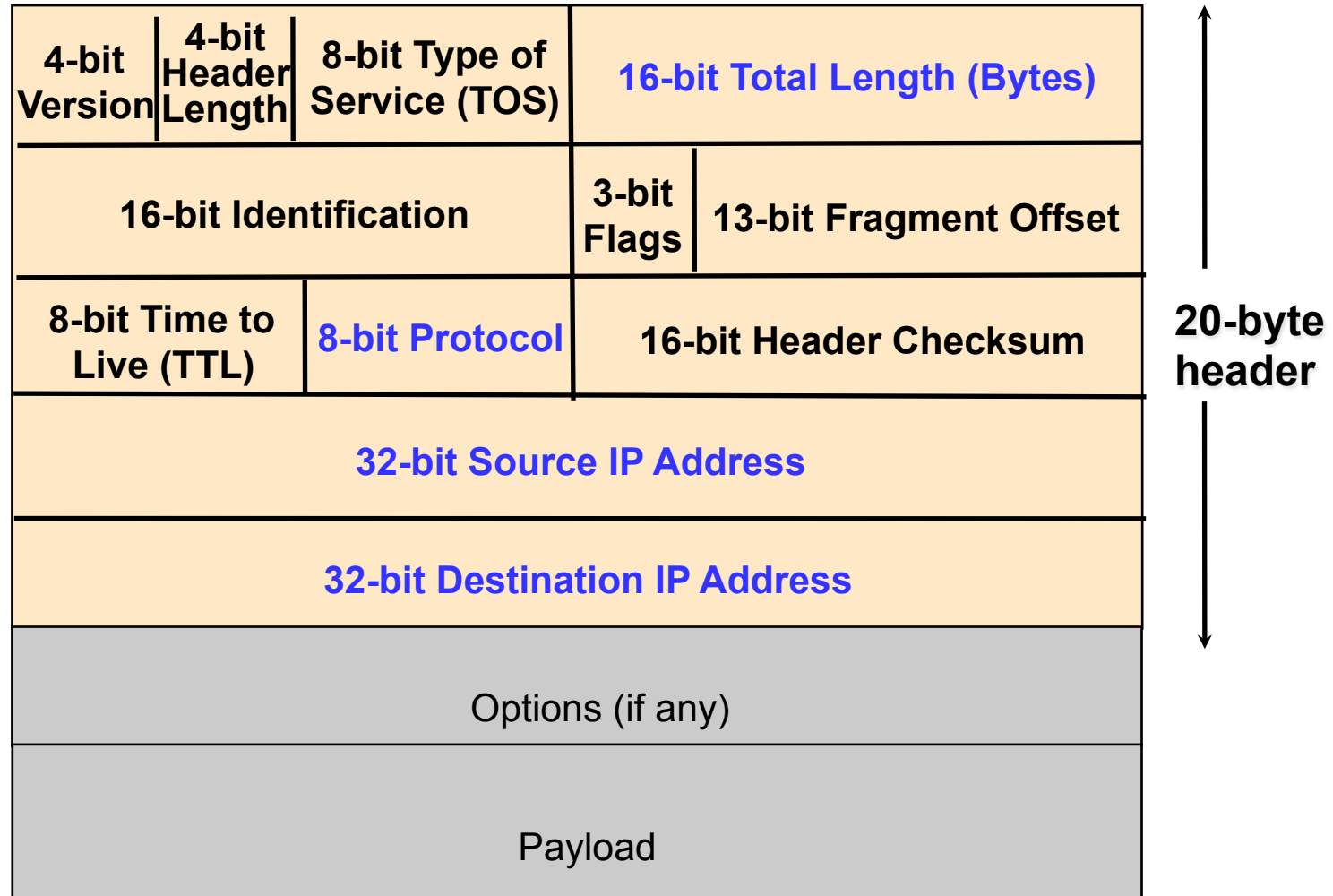
□ Definition:

- Observing traffic into the network
- Computing metrics on the monitored traffic
 - In our case: Application Mix

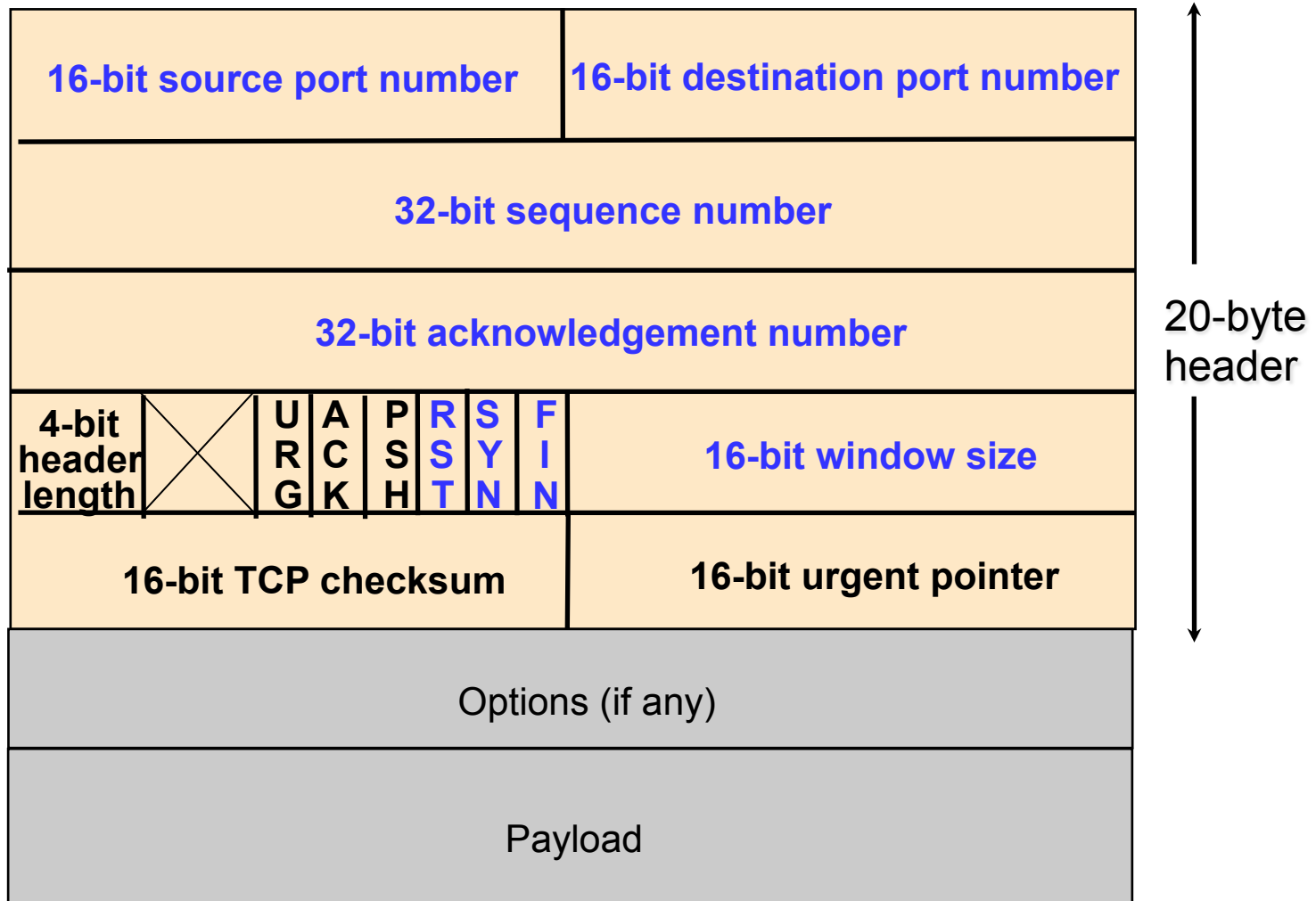
□ Packet monitors

- Available data:
 - All protocol information
 - All content

IP header format



TCP header format



Tools

❑ ipsumdump

- Good for quick summaries

❑ Wireshark

- Good for visual inspection of in depth details

❑ tcpdump

- Good for in depth details
- Basis for Wireshark

❑ Bro

- Good for in depth scripted analysis
- Security analysis
- Application analysis

Ipsumdump (subset)

```
anja% ipsumdump -h
```

'Ipsumdump' reads IP packets from tcpdump(1) files, or network interfaces, and summarizes their contents in an ASCII log.

Usage: `ipsumdump [CONTENT OPTIONS] [-i DEVNAMES | FILES] > LOGFILE`

<code>-t, --timestamp</code>	Include packet timestamps.
<code>-s, --src</code>	Include IP source addresses.
<code>-d, --dst</code>	Include IP destination addresses.
<code>-S, --sport</code>	Include TCP/UDP source ports.
<code>-D, --dport</code>	Include TCP/UDP destination ports.
<code>-l, --length</code>	Include IP lengths.
<code>-p, --protocol</code>	Include IP protocols.
<code>--id</code>	Include IP IDs.
<code>-g, --fragment</code>	Include IP fragment flags ('F' or '.').
<code>-F, --tcp-flags</code>	Include TCP flags word.
<code>-Q, --tcp-seq</code>	Include TCP sequence numbers.
<code>-K, --tcp-ack</code>	Include TCP acknowledgement numbers.
<code>-W, --tcp-window</code>	Include TCP receive window (unscaled).
<code>--udp-length</code>	Include UDP lengths.
<code>-L, --payload-length</code>	Include payload lengths (no IP/UDP/TCP headers).
<code>--payload</code>	Include packet payloads as quoted strings.
<code>--payload-md5</code>	Include MD5 checksum of packet payloads.
<code>--capture-length</code>	Include lengths of captured IP data.

...

Data source options (give exactly one):

<code>-r, --tcpdump</code>	Read tcpdump(1) FILES (default).
<code>-i, --interface</code>	Read network devices DEVNAMES until interrupted.

Tcpdump (subset)

Usage: tcpdump [options] [filter expression]

general options:

[-c packetcount]

input options:

[-i interface] | [-r input dumpfile name
[-P passphrase or - for stdin]]

[-F filterfile] [-s snaplength]

binary-output options:

[-w dumpfile(base) [-W dumpfile slice size]
[-P passphrase or - for stdin]]

ASCII output options:

[-n]: do not resolve hostnames

[-M]: output in machine-readable format

[-v]: increase verbosity (e.g. prints checksums)

[-e]: print linklayer information

[-X]: full-packet output in hex-format

[-A]: print packet payload as ASCII

[-S]: absolute TCP sequence numbers

...

Tcpdump output (three-way TCP handshake and HTTP request message)

23:40:21.008043 eth0 > 135.207.38.125.1043 > lovelace.acm.org.www: 80: S
617756405:617756405(0) win 32120 <mss 1460,sackOK,timestamp 46339
0,nop,wscale 0> (DF)

23:40:21.036758 eth0 < lovelace.acm.org.www > 135.207.38.125.1043: S
2598794605:2598794605(0) ack 617756406 win 16384 <mss 512>

23:40:21.036789 eth0 > 135.207.38.125.1043 > lovelace.acm.org.www: .
1:1(0) ack 1 win 32120 (DF)

23:40:21.037372 eth0 > 135.207.38.125.1043 > lovelace.acm.org.www: P
1:513(512) ack 1 win 32256 (DF)

23:40:21.085106 eth0 < lovelace.acm.org.www > 135.207.38.125.1043: .
1:1(0) ack 513 win 16384

23:40:21.085140 eth0 > 135.207.38.125.1043 > lovelace.acm.org.www: P
513:676(163) ack 1 win 32256 (DF)

23:40:21.124835 eth0 < lovelace.acm.org.www > 135.207.38.125.1043: P
1:179(178) ack 676 win 16384

Wireshark (subset)

Usage: wireshark [options] ... [<infile>]

Capture interface:

-i <interface> name or idx of interface (def: first non-loopback)
-f <capture filter> packet filter in libpcap filter syntax
-s <snaplen> packet snapshot length (def: 65535)
-S update packet display when new packets are captured

...

Capture stop conditions:

-c <packet count> stop after n packets (def: infinite)

...

Input file:

-r <infile> set the filename to read from (no pipes or stdin!)

Processing:

-R <read filter> packet filter in Wireshark display filter syntax
-n disable all name resolutions (def: all enabled)

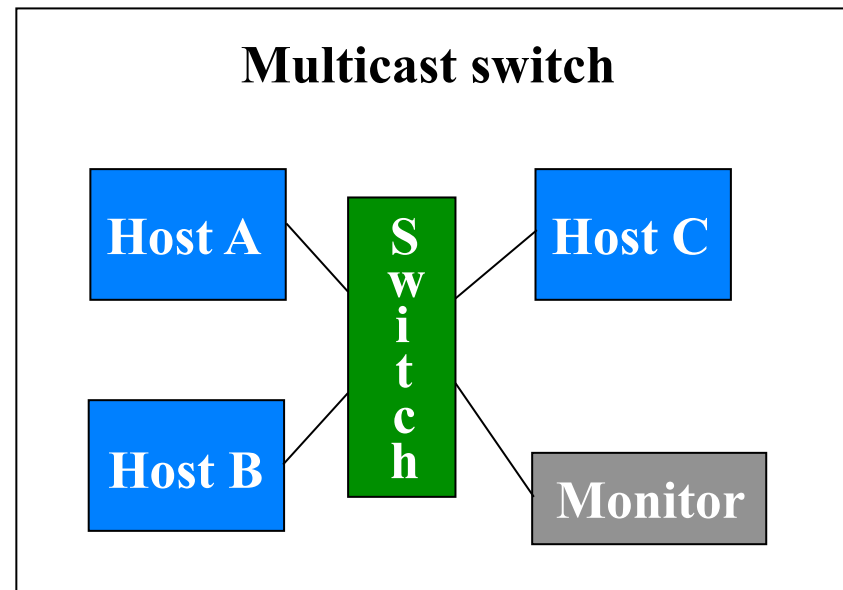
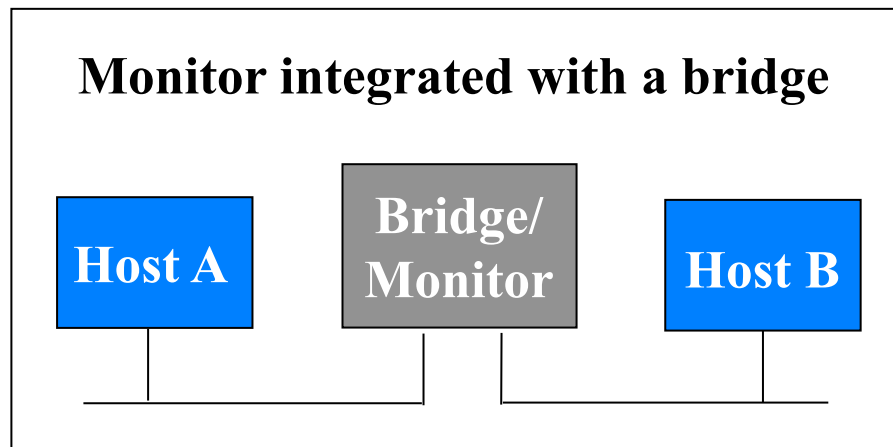
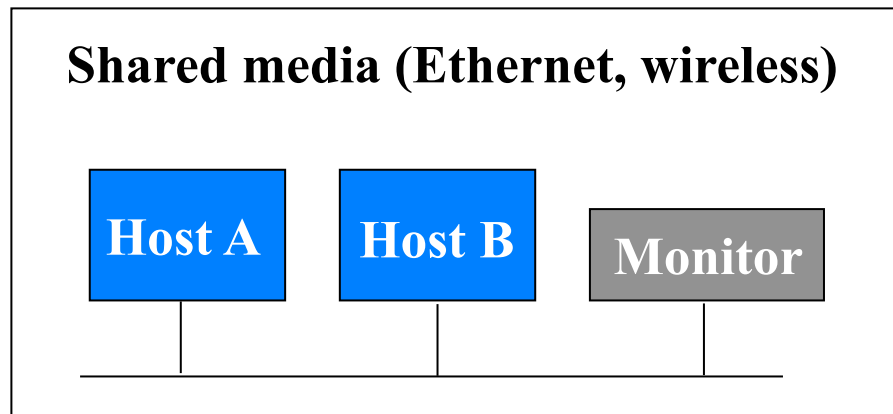
User interface:

-g <packet number> go to specified packet number after "-r"
-m set the font name used for most text
-t ad|a|r|d|dd|e output format of time stamps (def: r: rel. to first)
-X <key>:<value> eXtension options, see man page for details
-z <statistics> show various statistics, see man page for details

Selecting traffic

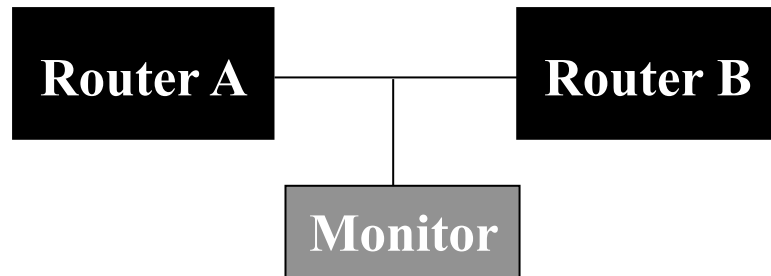
- ❑ Filter to focus on a subset of the packets
 - IP addresses/prefixes (e.g., to/from specific Web sites, client machines, DNS servers, mail servers)
 - Protocol (e.g., TCP, UDP, or ICMP)
 - Port numbers (e.g., HTTP, DNS, BGP, Napster)
- ❑ Collect first n bytes of packet (snap length)
 - Medium access control header (if present)
 - IP header (typically 20 bytes)
 - IP+UDP header (typically 28 bytes)
 - IP+TCP header (typically 40 bytes)
 - Application-layer message (entire packet)

Monitoring a LAN link



Monitoring a WAN link

Splitting a point-to-point link



Line card that does packet sampling

