Increasing TCP's Extensibility Using Second Grade Math

Eric Kaminsky (BWC), Ken Atchinson (BWC),
Mark Allman (ICSI)

Baldwin-Wallace College
October 2011

"I want to put a ding the universe." --Steve Jobs

Increasing TCP's Extensibility Using Second Grade Math

Eric Kaminsky (BWC), Ken Atchinson (BWC),
Mark Allman (ICSI)

Baldwin-Wallace College
October 2011

"I want to put a ding the universe." --Steve Jobs

Background

- The Internet is made up of myriad technologies
- *Network architecture* provides ....
  - abstractions
  - services
    - organizing principles
- I.e., the basic structure of the system

- Primary goal: flexibility and generality

Layering

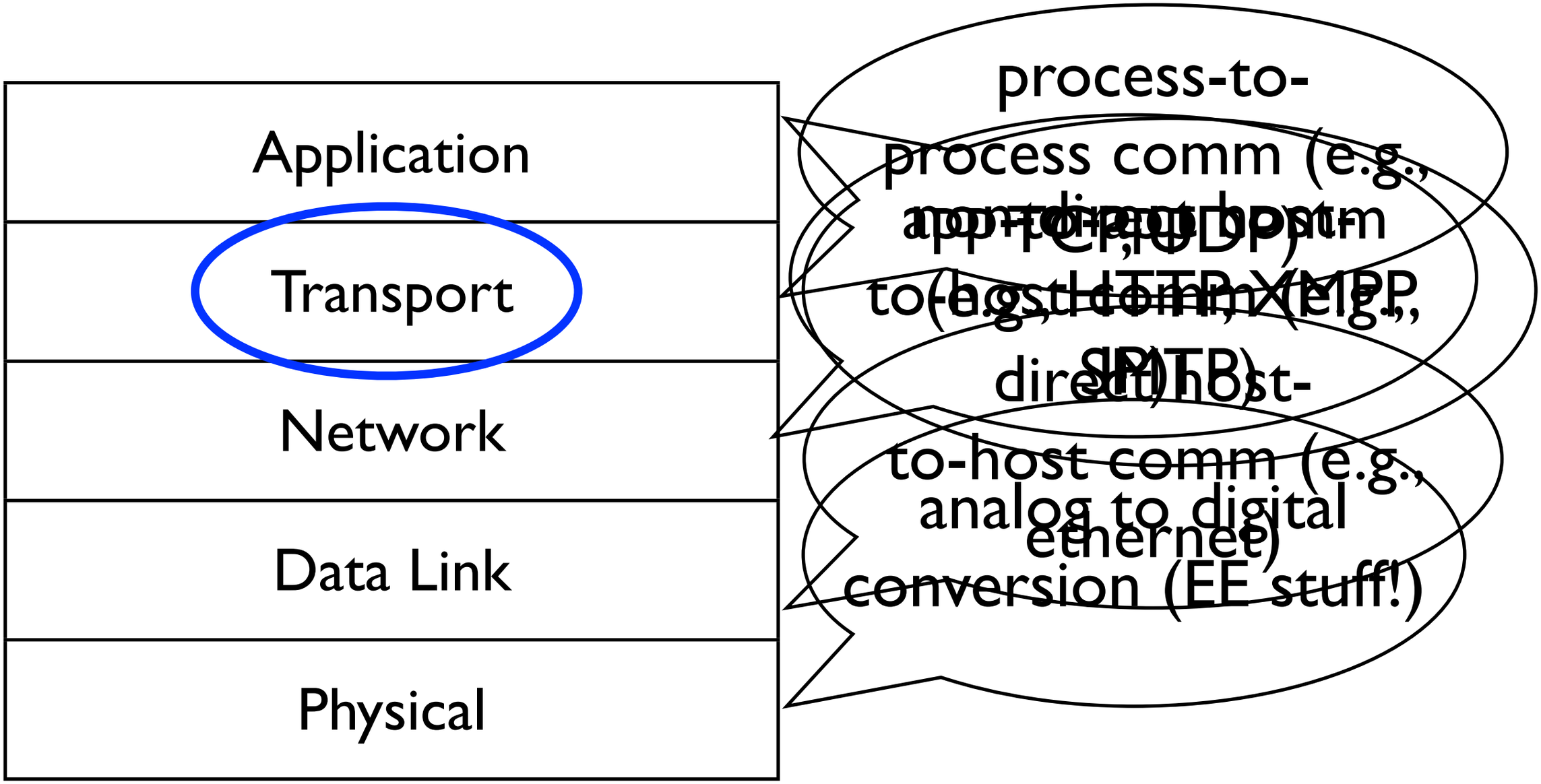
OSI Model

Application
Presentation
Session
Transport
Network
Data Link
Physical

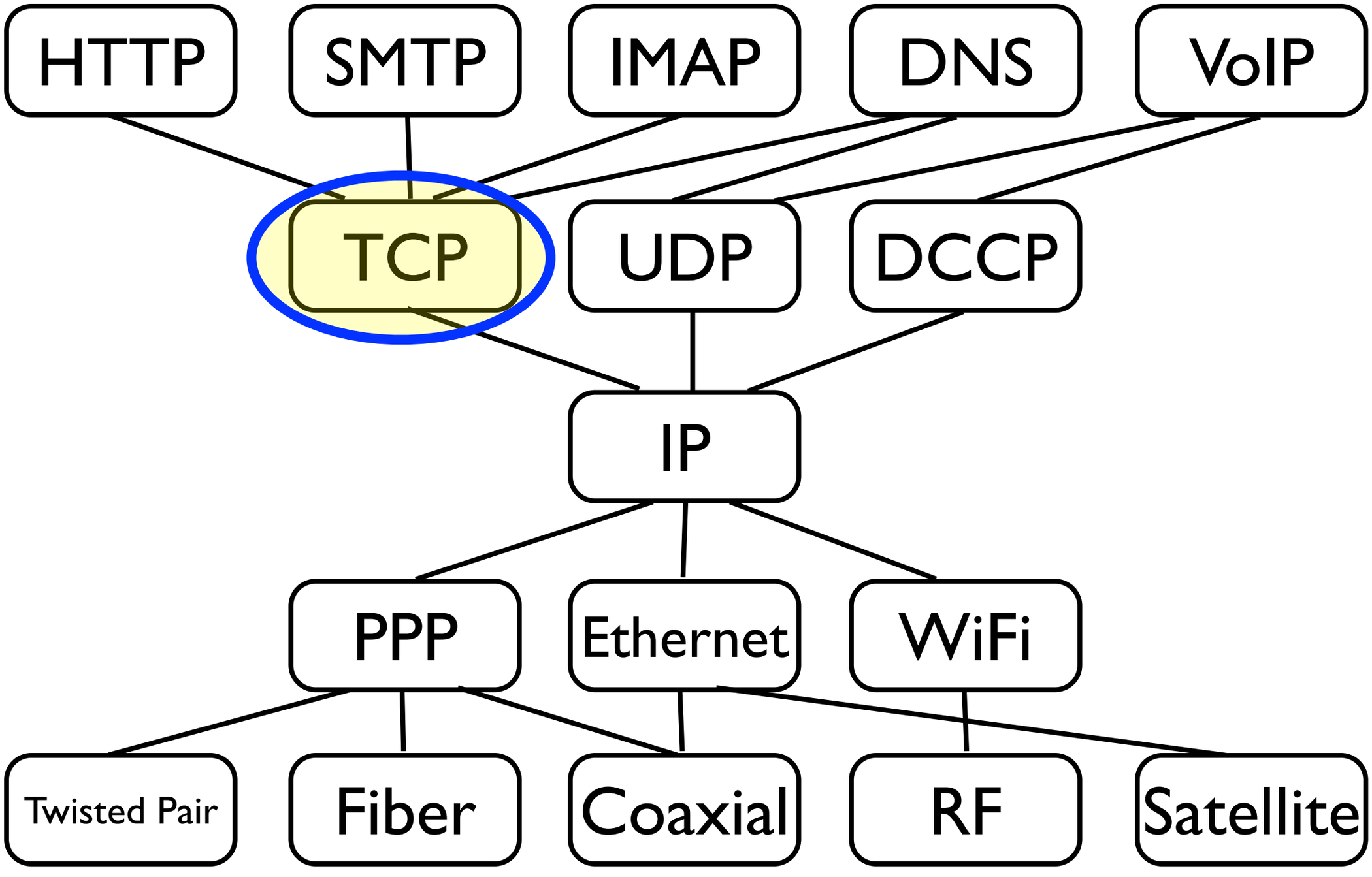
Internet Model

Application
Transport
Network
Data Link
Physical

Layering (cont.)



Layering (cont.)



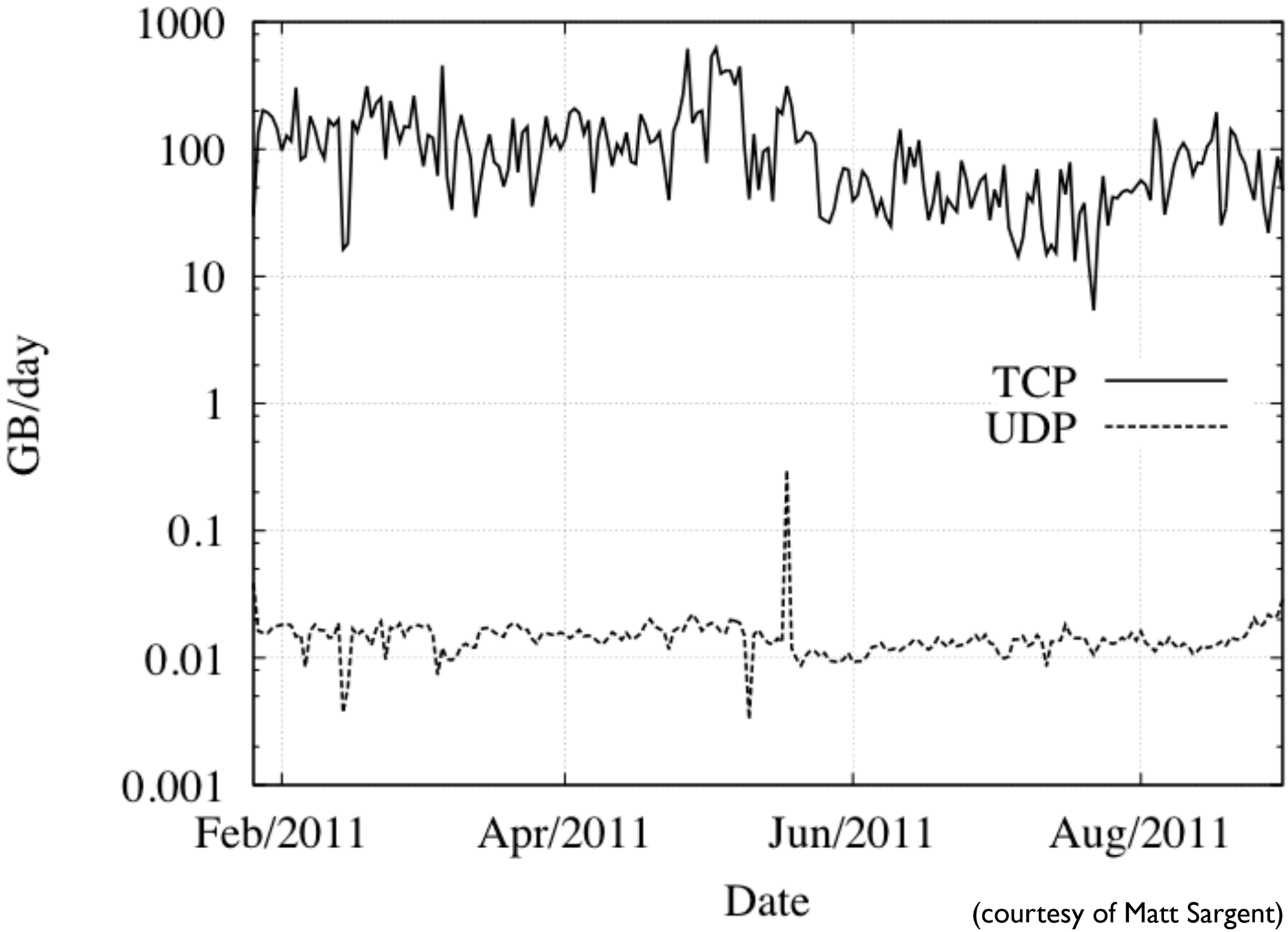
TCP

- Provides reliable in-order byte stream between processes
- Implements congestion control to prevent *collapse*

- Very general
- Usually suboptimal

- Carries most of the traffic on the Internet

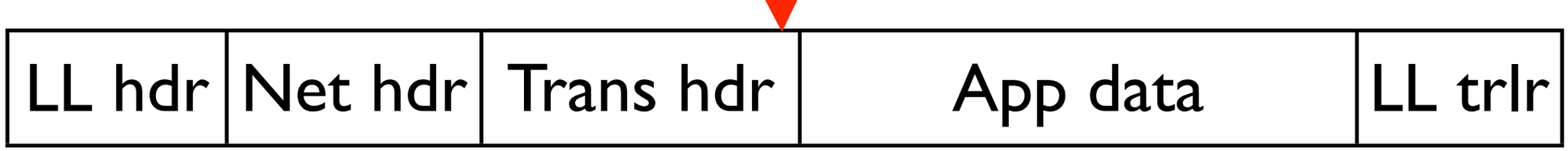
TCP Usage

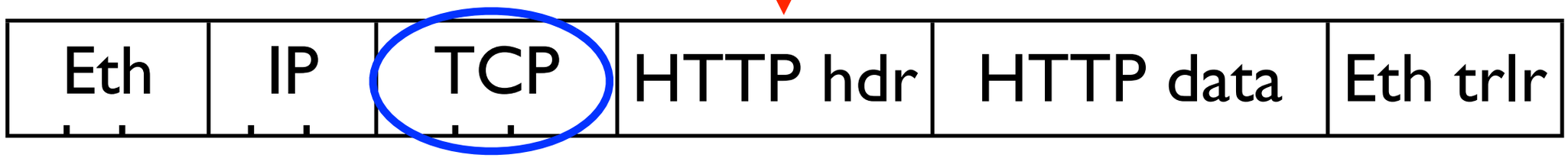


(courtesy of Matt Sargent)

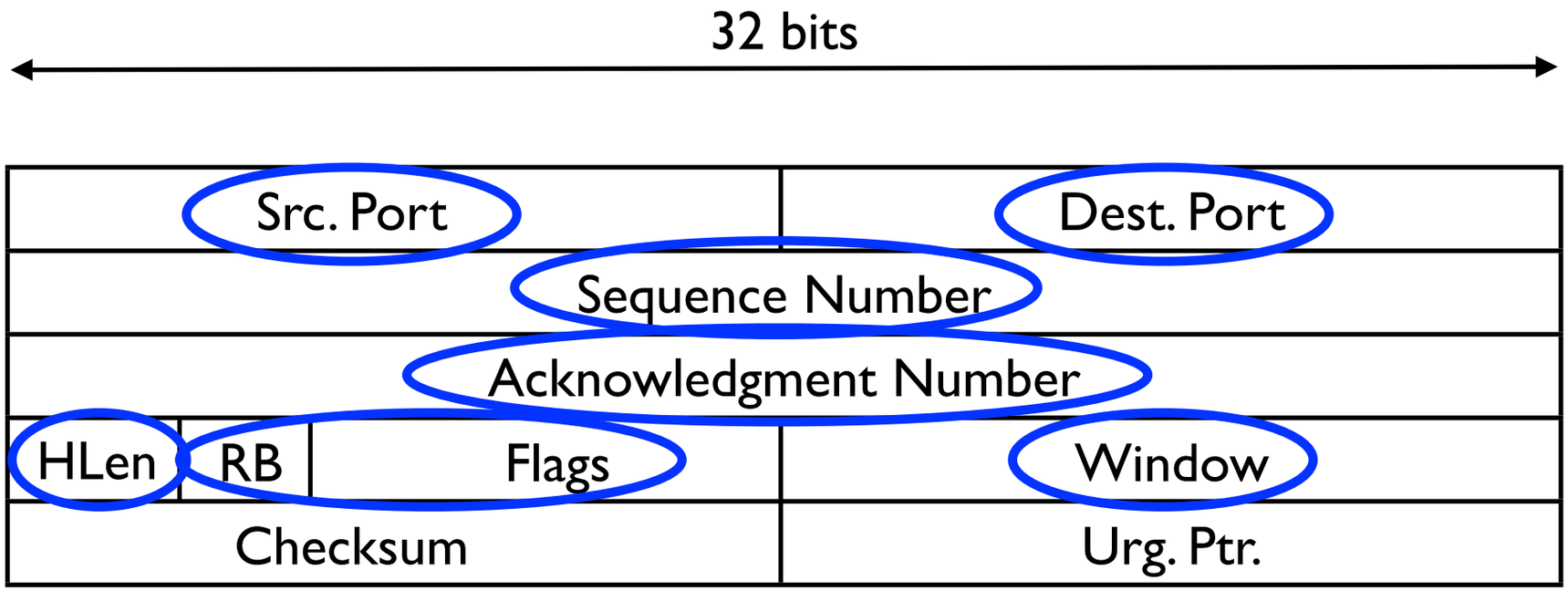
Packets

0000111010101001110101001001111100110I.....



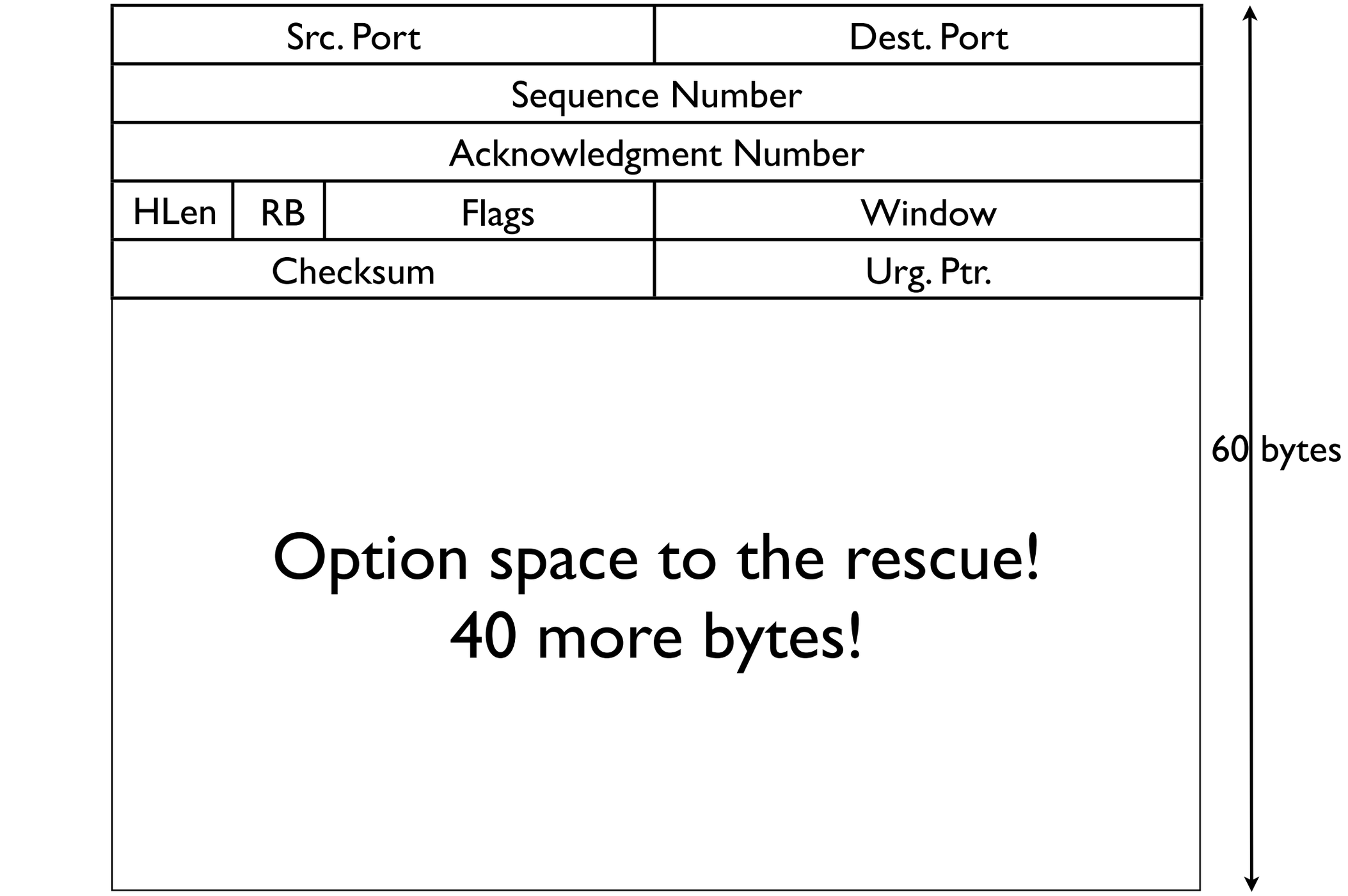


TCP Header



We're out of space!

TCP Header (cont.)



TCP Options

Option	***Size (bytes)***
MSS	3
SACK Permitted	2
Window Scale	3
SACK	2 + multiples of 8
Timestamp	10
Auth	18
.....	

UGH! We're really out of space!

How To Fix TCP?

- Two approaches:
  - TCPng
  - More options

More Options

- Approach #1: Negotiate a different way to convey header length
  - scale factor on the current field
  - new field as the first option

More Options (cont.)

- Approach #2: Introduce a magic "end-of-options" option and walk the options

More Options (cont.)

- Approach #3: Define unused code-points in the current header length field
  - E.g., HLen == 0 may yield a 96 byte header
  - E.g., HLen == 1 may yield a 128 byte header
  - Etc.

More Options (cont.)

- Engineering costs and benefits to all these schemes
- Maybe TCP is just the wrong size!

Our Approach

- The previously voiced two options:
  - TCPng or more options

- Maybe we need a third general approach ....

- Foolishly multiply the TCP header size by two
  - aka, “TCPx2”

TCPx2: Pros

- Conceptually simple
- Extends a known useful protocol
- The protocol semantics don't change; only the syntax
- Code "should be" straightforward
- Solves all issues at once, not piecemeal

TCPx2: Cons

- New header format
  - All hosts have to change
  - All firewalls have to change
  - All NATs have to change
  - .....
- Not backward compatible
  - (a point we address and will return to ....)

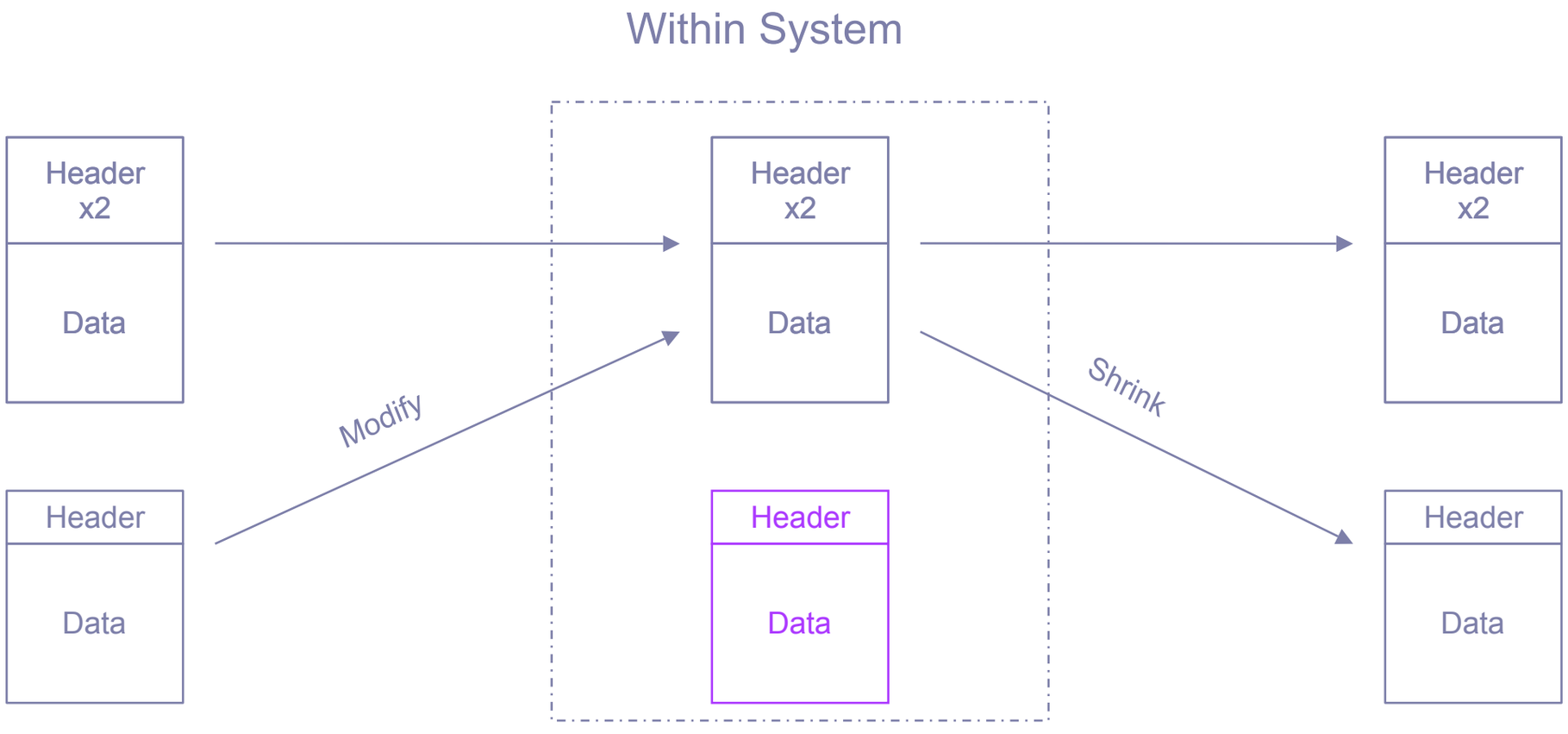
Implementation

- Modifying the Linux kernel
- Kernel: lots of C code
- 18 files modified for TCPx2

	Amount of Code
TCP	23K-30K LOC
TCPx2	2,851 LOC (added or changed)
	10-12% change

Strategy

- Modify TCP packet structure
- Packets in the system will have TCPx2 headers
- Accept both types of packets, but keep one structure internally



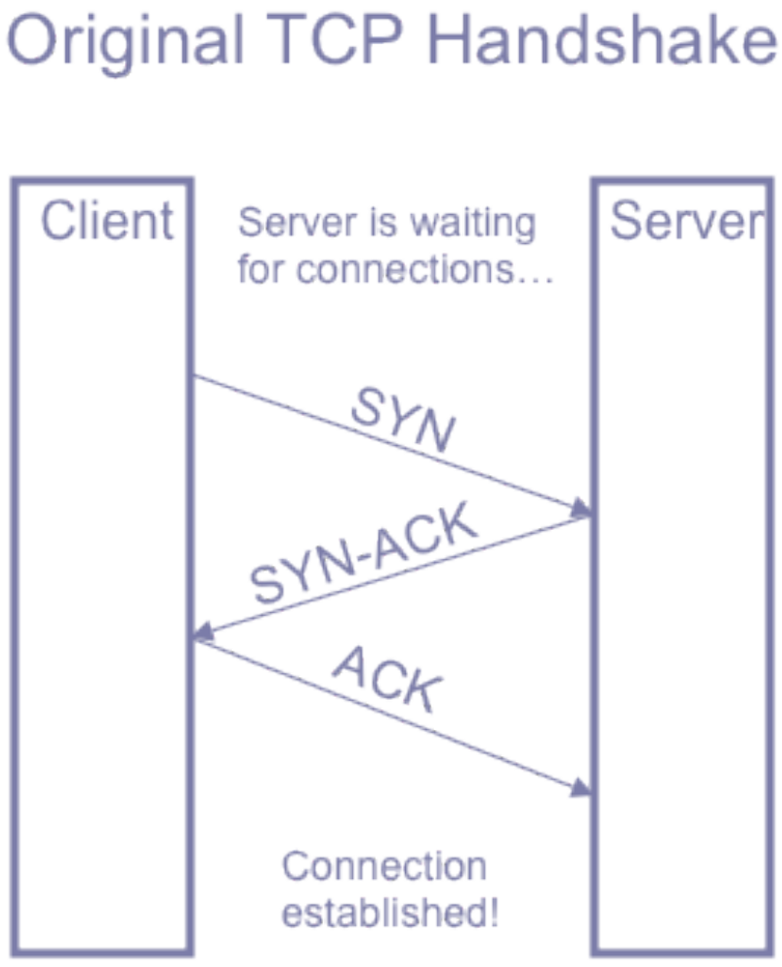
Further Changes

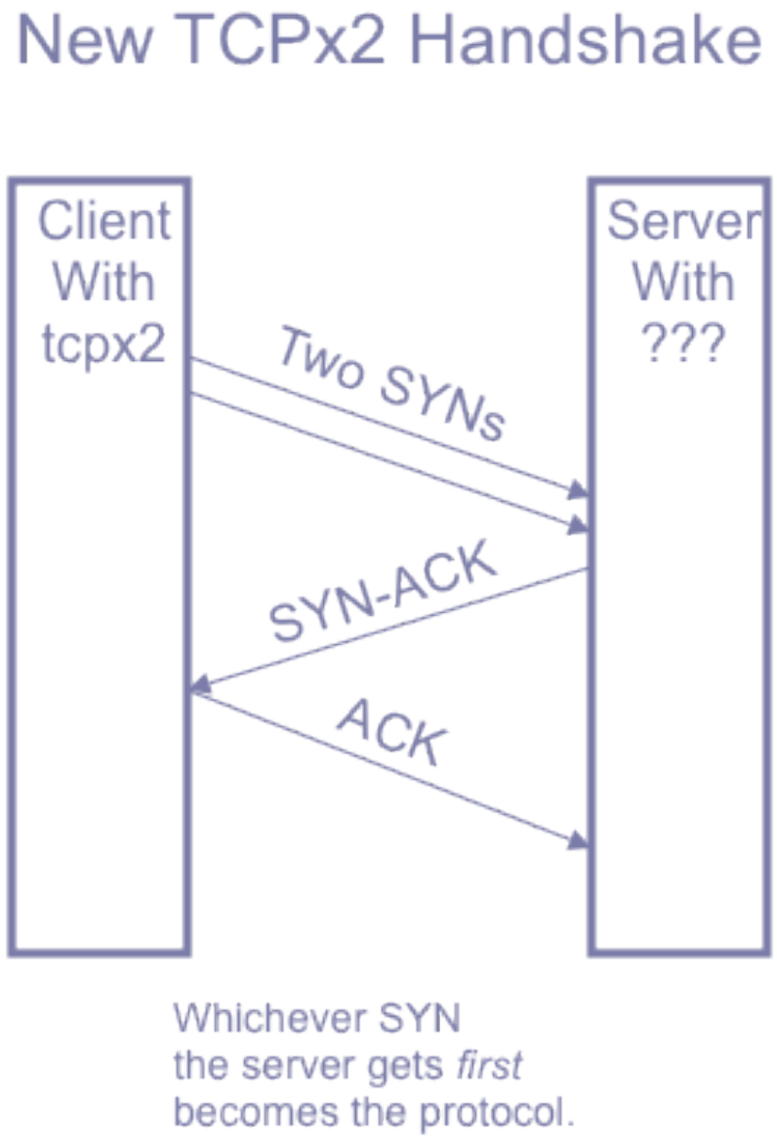
- Needed to fix many bugs caused by struct change
- Migrated a number of TCP features:
  - SACK (Selective Acknowledgement)
  - Expanded SYN cookies
  - Disabled window scaling for TCPx2 packets
    - (unneeded with larger window size)
- Modified *tcpdump* to parse TCPx2 packets

Migration

- How can we transition to the new protocol?
- TCPx2 needs to be able to:
  - Accept incoming connections from TCP and TCPx2 clients
  - Connect to servers, not knowing whether they support TCPx2 or not
  - … without sacrificing any speed

Migration (cont.)





Future Work

- Performance testing
- Modify a NAT/firewall to support TCPx2
- Utilize expanded port space

Questions? Comments?

Mark Allman, Eric Kaminsky, Ken Atchinson

mallman@icir.org
http://www.icir.org/mallman/