

Introduction to the Transport Layer

Networked Systems 3
Lecture 11

Lecture Outline

- Role of the transport layer
- Transport layer functions
- Transport protocols in the Internet
  - TCP, UDP, DCCP, and SCTP
  - Deployment considerations

Role of the Transport Layer

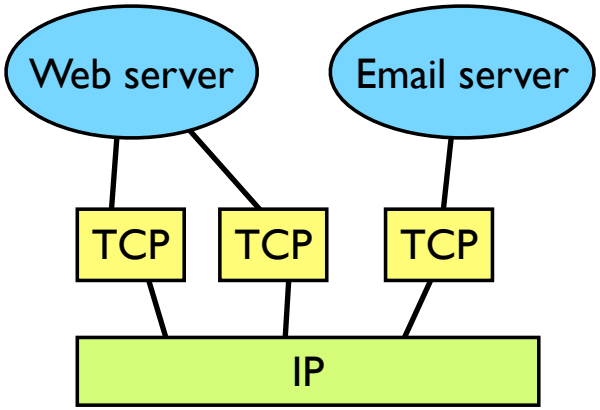
- Isolate upper layers from the network layer
  - Hide network complexity; make unreliable network appear reliable; enhance QoS of network layer
- Provide a useful, convenient, easy to use service
  - An easy to understand service model
  - An easy to use programming API
    - The Berkeley sockets API – very widely used by application programmers
    - Compare to network layer API – usually hidden in operating system internals

Transport Layer Functions

- Transport layer provides the following functions:
  - Addressing and multiplexing
  - Reliability
  - Framing
  - Congestion control
- Operates process-to-process, not host-to-host

Addressing and Multiplexing

- The network layer address identifies a host
- The transport layer address identifies a user process – a *service* – running on a host
- Provides a demultiplexing point
  - Each service has a unique transport layer address



Reliability

- Network layer *is* unreliable
  - Best effort packet switching in the Internet
  - But even nominally reliable circuits may fail
- Transport layer enhances the quality of service provided by the network, to match application needs
  - Appropriate *end-to-end* reliability

The End-to-End Argument

- Is it better to place functionality within the network or at the end points?
  - Only put functions that are absolutely necessary within the network, leave everything else to end systems
    - Example: put reliability in the transport layer, rather than the network
    - If the network is not guaranteed 100% reliable, the application will have to check the data anyway → don't check in the network, leave to the end-to-end transport protocol, where the check is visible to the application
  - One of the defining principles of the Internet

Transport Layer Reliability

- Different applications need different reliability
  - Email and file transfer → all data must arrive, in the order sent, but no strict timeliness requirement
  - Voice or streaming video → can tolerate a small amount of data loss, but requires timely delivery
- Implication for network architecture:
  - Network layer provides timely but unreliable service
  - Transport layer protocols add reliability, if needed

Framing

- Applications may wish to send structured data
- Transport layer responsible for maintaining the boundaries
  - Transport must *frame* the original data, if this is part of the service model

Congestion and Flow Control

- Transport layer controls the application sending rate
  - To match the rate at which network layer can deliver data – *congestion control*
  - To match the rate at which receiving application can process the data – *flow control*
- Must be performed end-to-end, since only end points know characteristics of entire path

Congestion and Flow Control

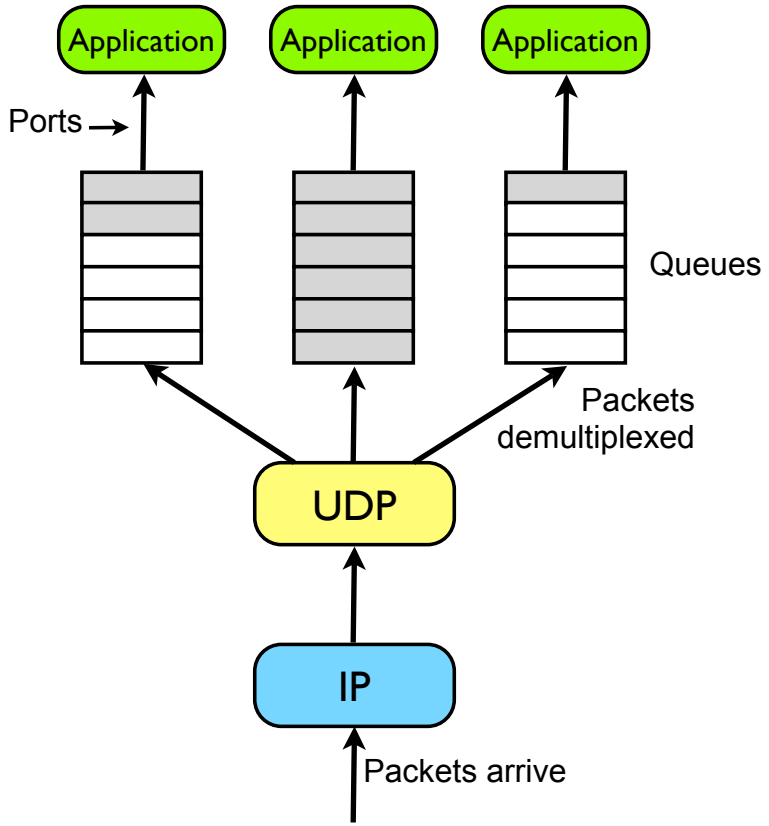
- Different applications have different needs for congestion control
  - Email and file transfer → *elastic* applications; faster is better, but don't care about actual sending rate
  - Voice or streaming video → *inelastic* applications; have minimum and maximum sending rates, and care about the actual sending rate
- Want range of congestion control algorithms at transport layer; within the network constraints

Internet Transport Protocols

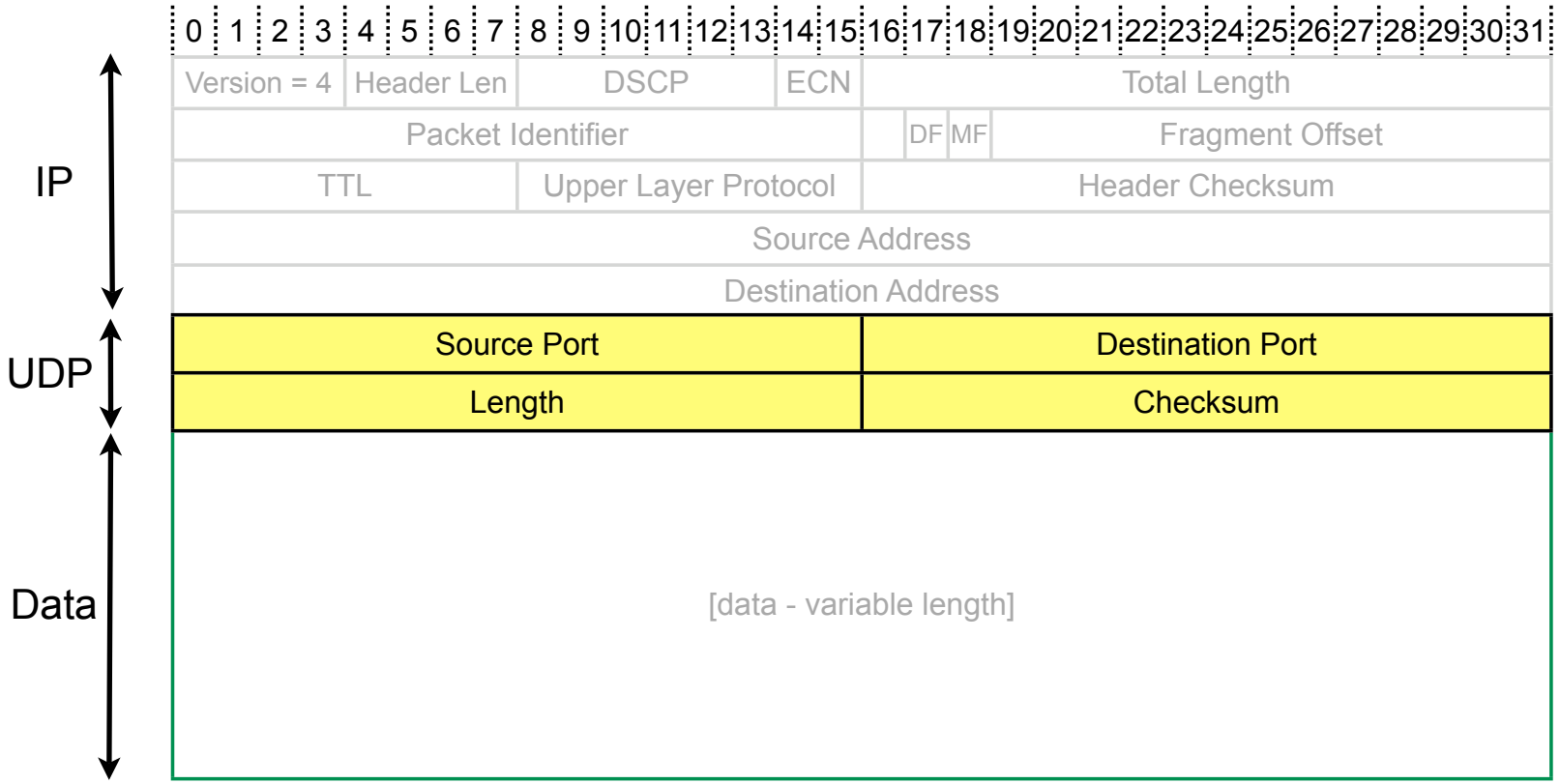
- The Internet Protocol provides a common base for various transports
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)
  - Datagram Congestion Control Protocol (DCCP)
  - Stream Control Transmission Protocol (SCTP)
- Each makes different design choices

UDP

- Simplest transport protocol
- Exposes raw IP service model to applications
  - Connectionless, best effort packet delivery: framed, but unreliable
  - No congestion control
- Adds a 16 bit *port* number to identify services



UDP Packet Format



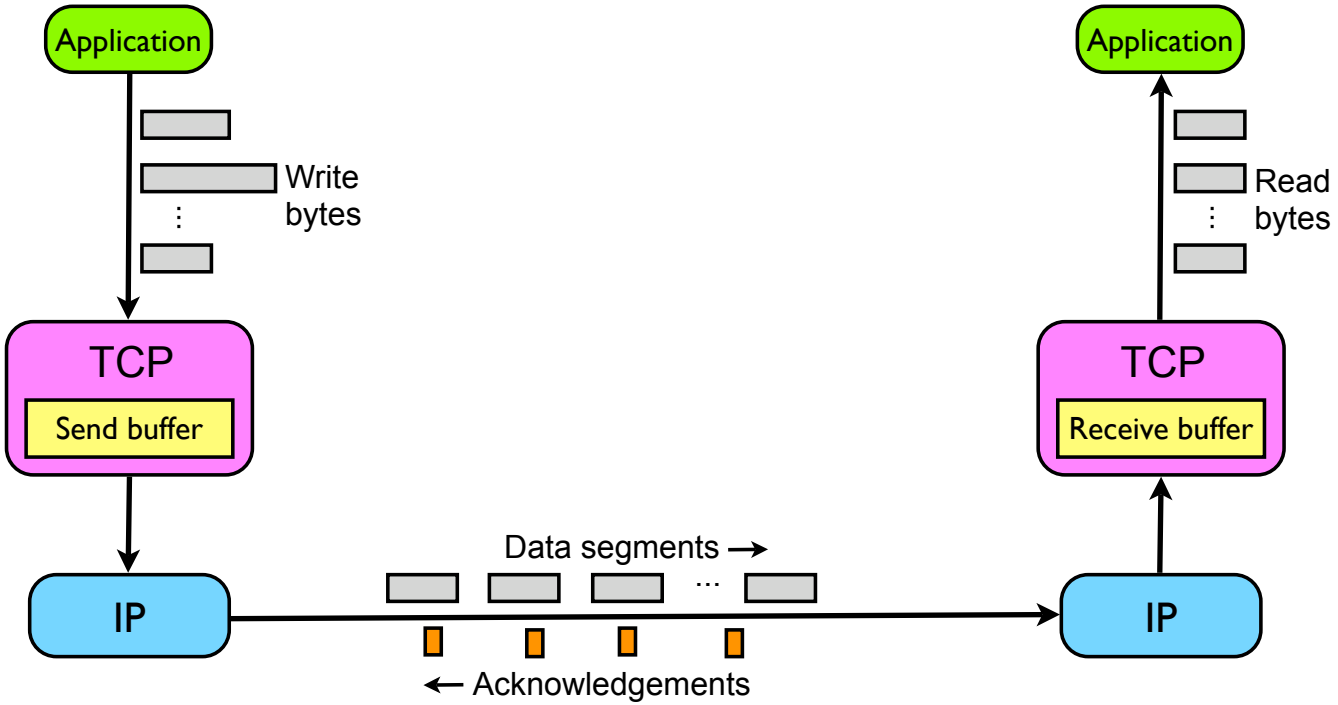
UDP Applications

- Useful for applications that prefer timeliness to reliability
  - Voice-over-IP
  - Streaming video
- Must be able to tolerate some loss of data
- Must be able to adapt to congestion in the application layer

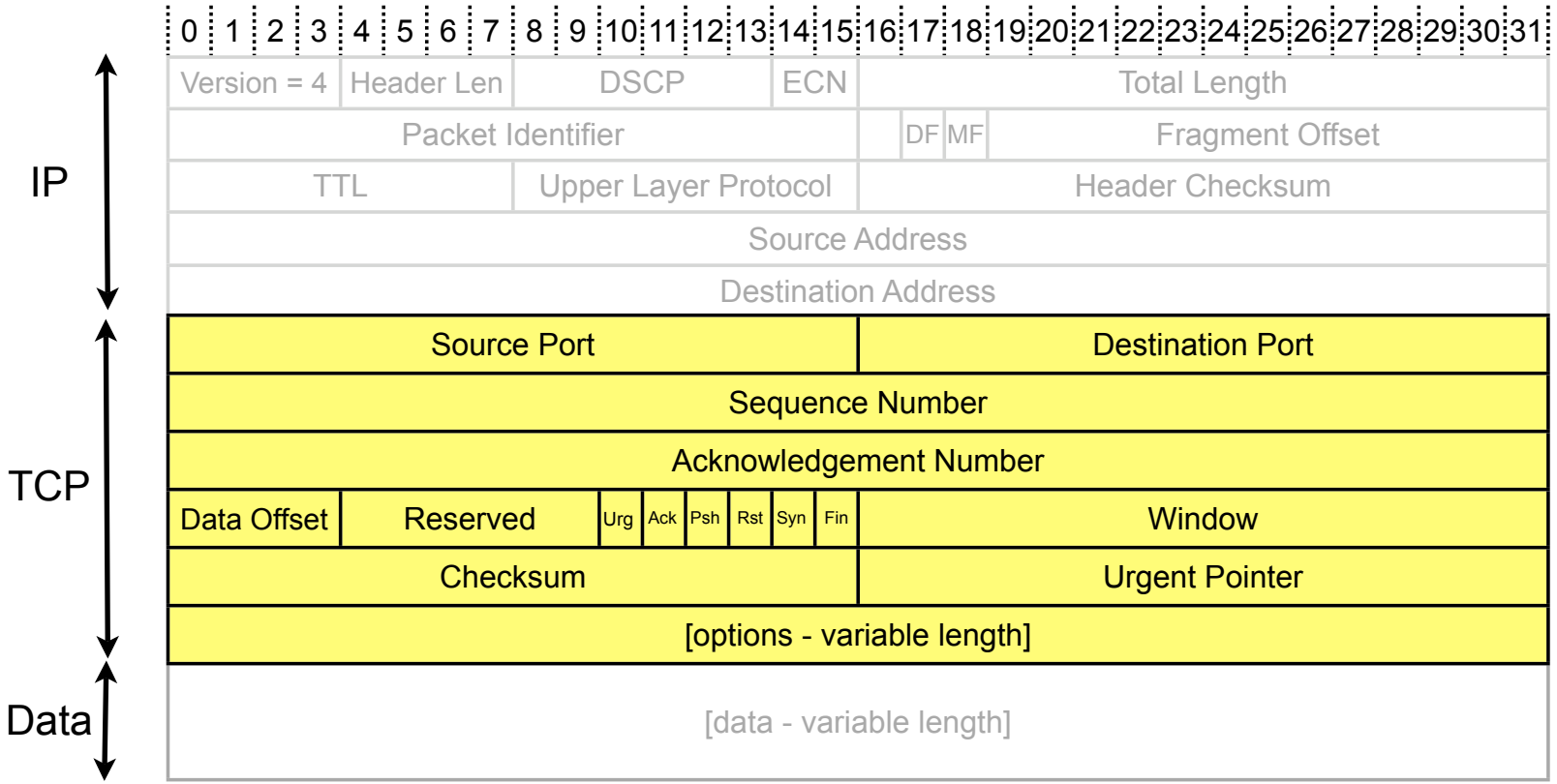
TCP

- Reliable byte stream protocol running over IP
  - Adds reliability
    - Packets contain sequence number to detect loss; any lost packets are retransmitted; data is delivered to higher-layers in order, without gaps
  - Adds congestion control – details in lecture 14
  - Adds 16 bit *port* number as a service identifier
  - Doesn't provide framing
    - Delivers an ordered byte stream, the application must impose structure

TCP


Application
Write bytes
TCP
Send buffer
IP
Data segments →
Acknowledgements ←
Application
Read bytes
TCP
Receive buffer
IP

TCP Packet Format



TCP Applications

- Useful for applications that require reliable data delivery, and can tolerate some timing variation
  - File transfer and web downloads
  - Email
  - Instant messaging
- Default choice for most applications - you've been using TCP for your lab work

TCP and UDP Ports

- TCP and UDP use a 16 bit port number to identify services
- Servers listen on *well known ports*
  - E.g. HTTP = 80, ssh = 22, jabber = 5269
  - http://www.iana.org/assignments/port-numbers
- Clients connect from an *ephemeral port*
  - Unused port, randomly allocated by operating system

Other Transport Protocols

- The IP network layer also supports two new transport protocols:
  - DCCP
  - SCTP

- Not widely used at this time, but potentially useful in future

DCCP

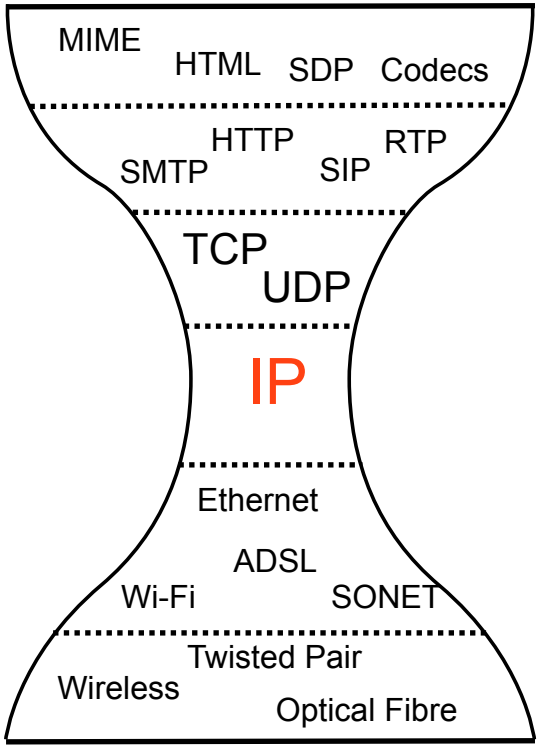
- Datagram Congestion Control Protocol
  - Unreliable, connection oriented, congestion controlled datagram service
    - “TCP without reliability” or “UDP with connections and congestion control”
    - Potentially easier for NAT boxes and firewalls than UDP
    - Congestion control algorithm (“CCID”) negotiated at connection setup – range of algorithms supported
  - Adds 32 bit service code in addition to port number
- Use case: streaming multimedia and IPTV

SCTP

- Stream Control Transmission Protocol
  - Reliable datagram service, ordered per stream
  - Multiple streams within a single association
  - Multiple connection management
    - Fail-over from one IP address to another, for reliable multi-homing
  - TCP-like congestion control
- Use case: telephony signalling; “a better TCP”

Deployment Considerations

- IP is agnostic of the transport layer protocol
- But, firewalls perform “deep packet inspection” and look beyond the IP header to make policy decisions
  - The only secure policy is to disallow anything not understood
  - Implication: very difficult to deploy new transport protocols (i.e. DCCP and SCTP) in the Internet
  - Implication: limits future evolution of the network



Service Models

Protocol	Addressing	Reliable?	Framed?	Congestion Controlled?
UDP	16 bit port number	Unreliable packet delivery	Yes – uses explicit datagrams	No – handled by application layer
TCP	16 bit port number	Reliable ordered byte stream	No – handled by application layer	Yes – suitable for elastic applications
DCCP	16 bit port number plus service code	Unreliable packet stream	Yes – uses explicit datagrams	Yes – wide range of algorithms possible
SCTP	16 bit port number	Reliable ordered byte stream	Yes – explicit *chunk* boundaries	Yes – suitable for elastic applications

Questions?