Master Course
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle

Chair for Network Architectures and Services
Department of Computer Science
Technische Universität München
http://www.net.in.tum.de

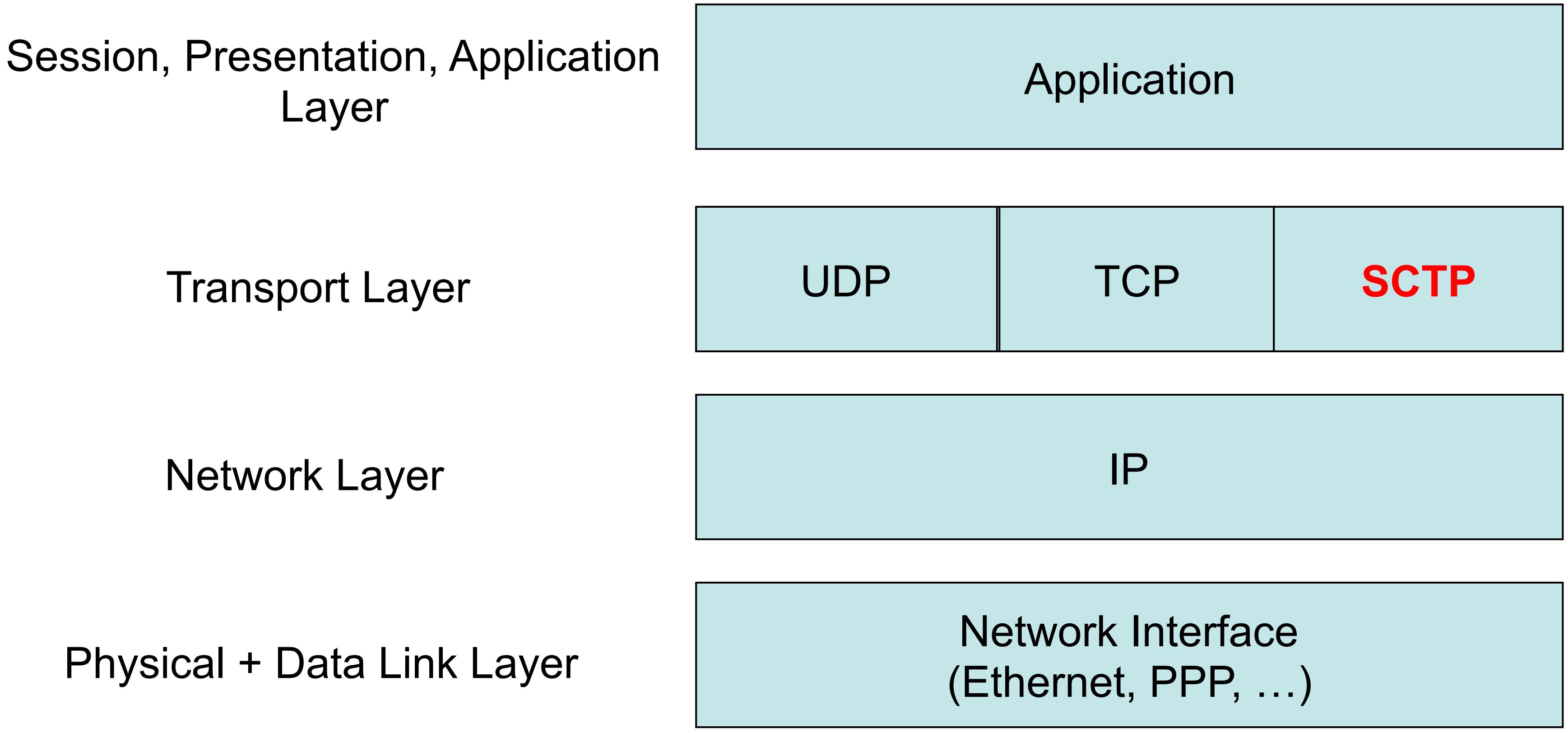
Outline

- Transport Layer
  - SCTP
  - TCP congestion control
  - Reliable Multicast Transport

Stream Control Transmission Protocol (SCTP)

Internet Protocol Stack

- The Internet Protocol Stack



Session, Presentation, Application Layer — Application

Transport Layer — UDP | TCP | **SCTP**

Network Layer — IP

Physical + Data Link Layer — Network Interface (Ethernet, PPP, …)

- Why another transport layer protocol?

Contents

- Limitations of UDP and TCP

- The Stream Control Transmission Protocol (SCTP)
  - Association setup / stream setup
  - Message types
  - Partial reliability
  - Multi-homing support
  - Congestion control

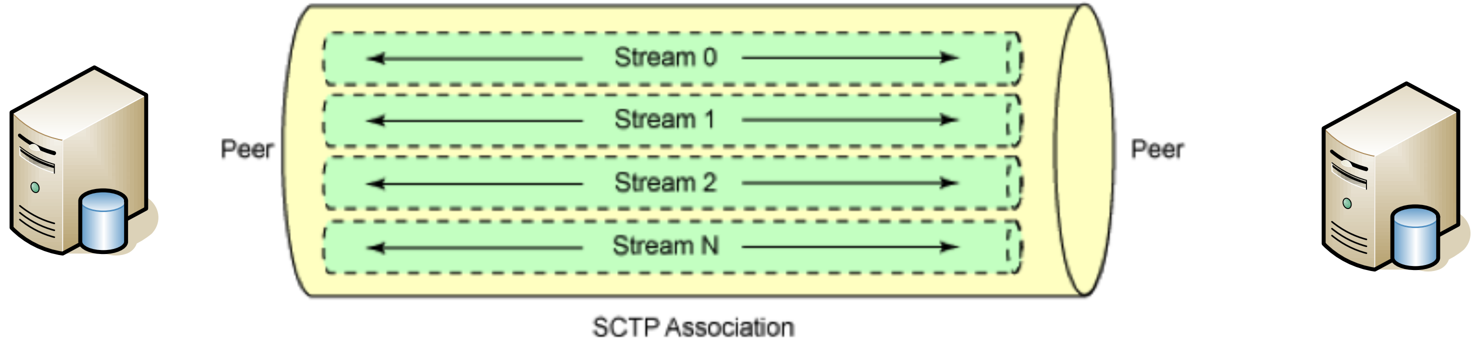
Limitations of UDP and TCP

- ❑ Certain applications have problems with UDP and TCP
- ❑ TCP: Head-of-line blocking with video streaming
  - ▪ Frames 2,3,4 arrived but cannot be shown because frame 1 is missing
  - ⇨ Video will stop until frame 1 is delivered
- ❑ UDP:
  - ▪ Out-of-order delivery possible
  - ▪ Lost packets neither detected nor corrected
  - ▪ No congestion control
- ❑ Example: Internet-Telephony
  - ▪ Two types of traffic:
    - • Signalling traffic: should be delivered reliable + in-order (TCP)
    - • Voice traffic: should not suffer from head-of-line blocking (UDP)
  - ▪ Need to manage two sockets

⇨ SCTP can deal with these problems

SCTP Features at a glance

- Connection and message oriented
  - SCTP builds an “association” between two peers
  - Association can contain multiple “streams”
  - Messages are sent over one of the streams

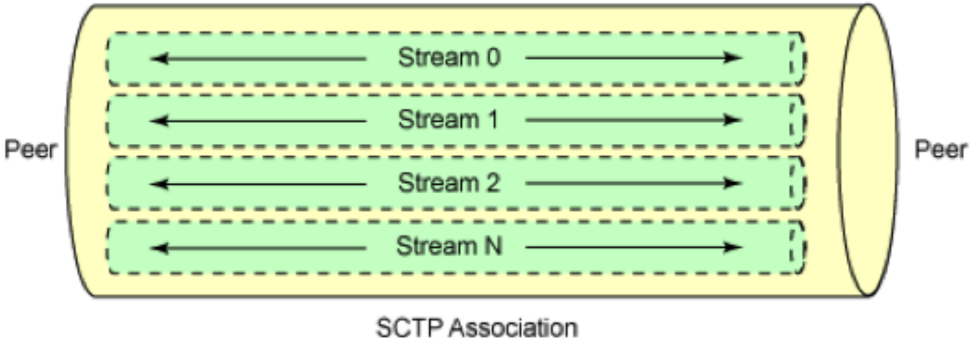


- Partial reliability
  - “Lifetime” defined for each message
    - Retransmission of a message performed during its lifetime
  - Messages delivery can be unreliable, fully reliable or partially reliable

- Multi-Homing
  - SCTP can use multiple IP addresses

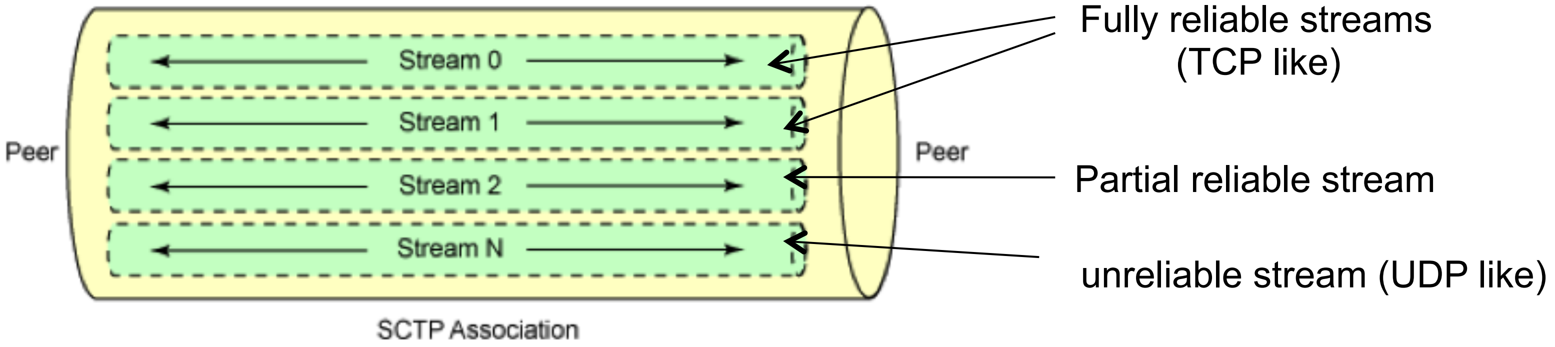
Transmission reliability (1)

- TCP
  - Segments transmitted fully reliably
  - In-order delivery to the application
  - Slow start and congestion avoidance for congestion control
- UDP
  - Error control limited to bit error detection
    ⇨ packets not retransmitted (by transport protocol)
  - No in-order delivery enforced
  - No congestion control
- SCTP can provide service of TCP and UDP and more, in a stream-specific way



Transmission reliability (2)

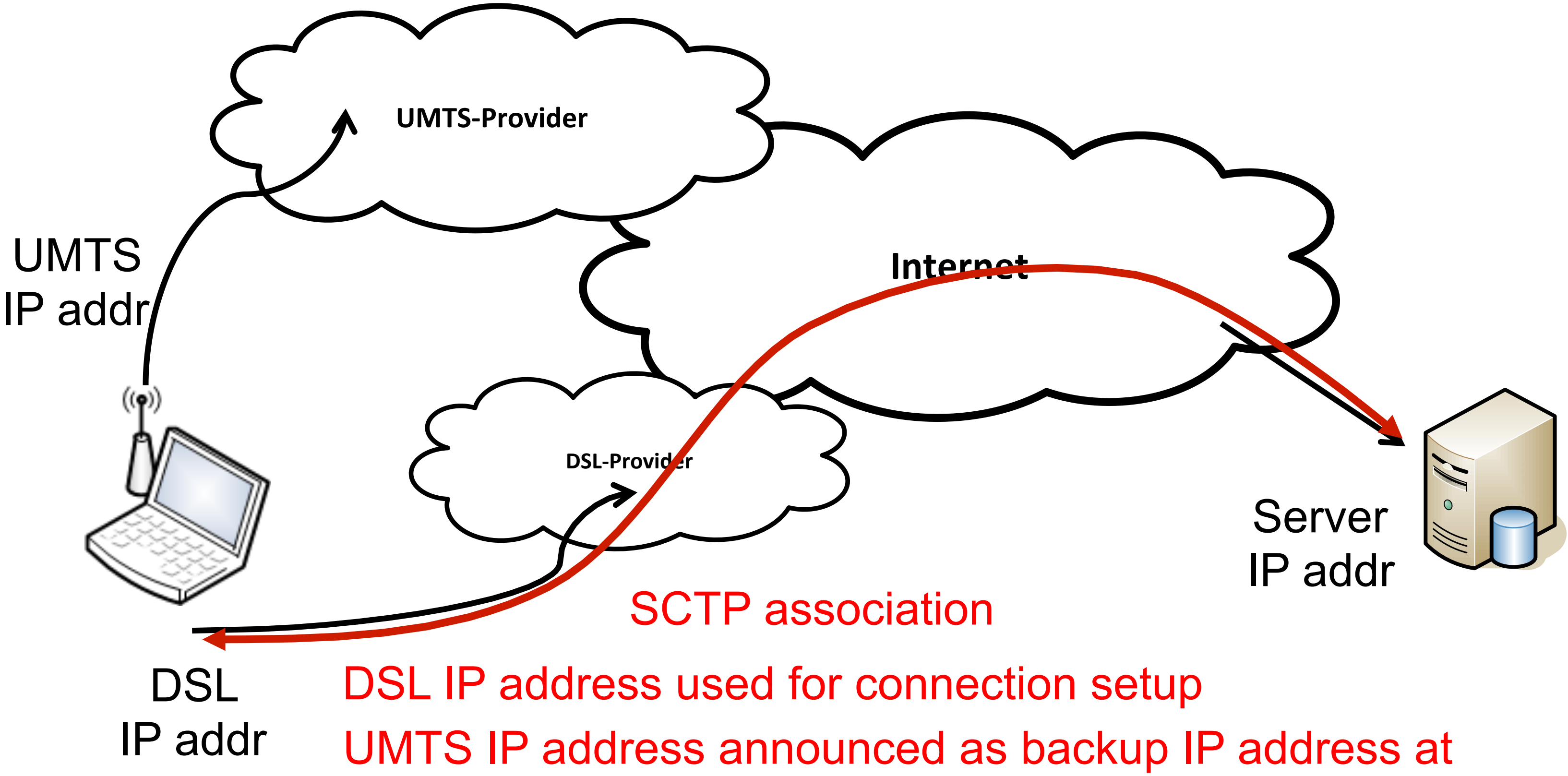
- Why multiple streams?
  - Solves head of line blocking
  - Simpler firewall rules (only one port for several streams)
  - Partial Reliability Extension (PR-SCTP) for different reliability levels
- PR-SCTP
  - Allows to set a lifetime parameter for each stream
  - Lifetime specifies how long the sender should try to retransmit a packet
  - Allows to mix reliable and unreliable streams



Multi-Homing: Association setup

- SCTP allows to choose one IP address at association setup

- Example scenario:

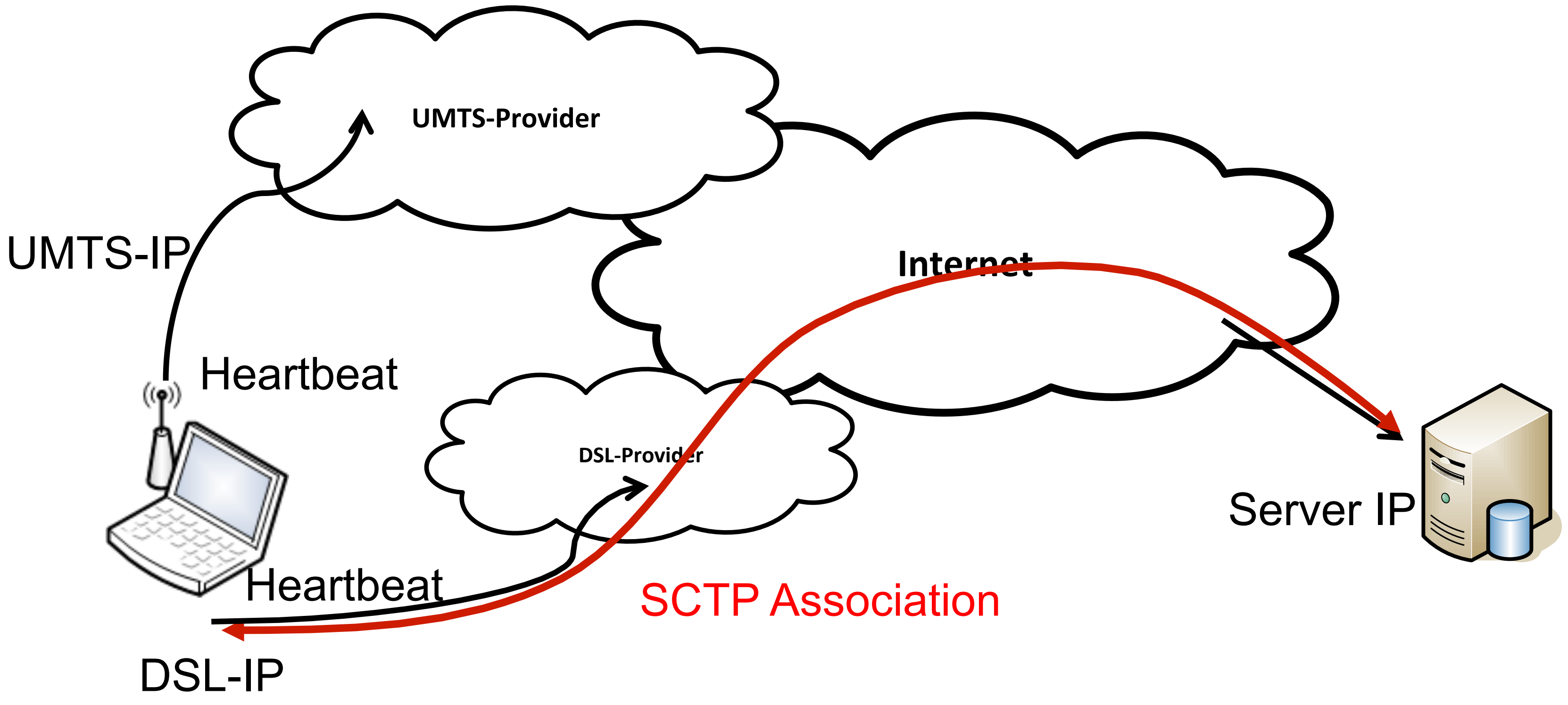


DSL IP address used for connection setup

UMTS IP address announced as backup IP address at association setup

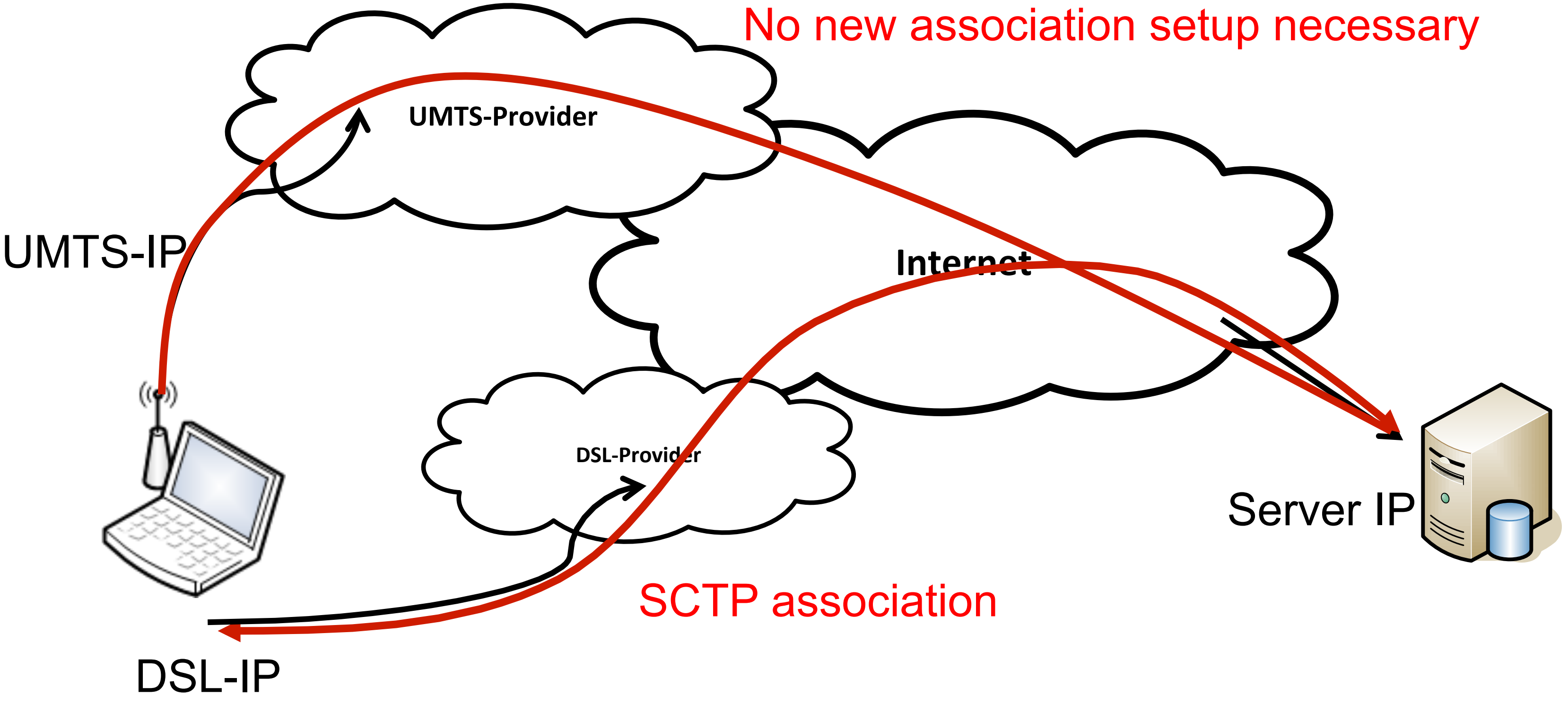
Multi-Homing

- Heartbeat messages periodically sent to check link availability



Multi-Homing

- Address Changes occur when default link is found to be broken
  - Identified by packet loss (data or heartbeat)
  - Consequence: SCTP resumes on the backup link

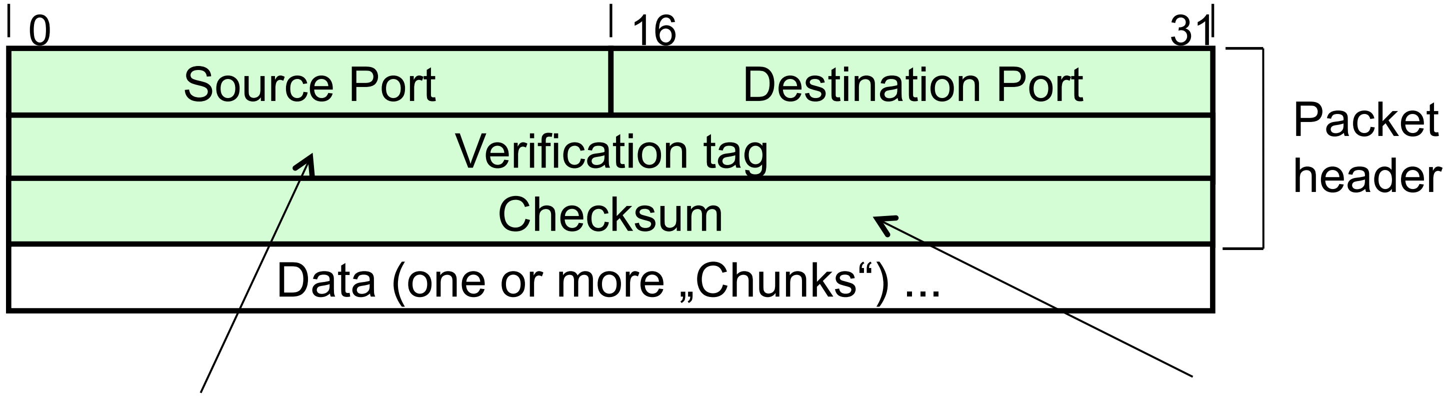


No new association setup necessary

SCTP association

SCTP Message Format

- Common header format
  - 12 byte header
  - included in every SCTP message

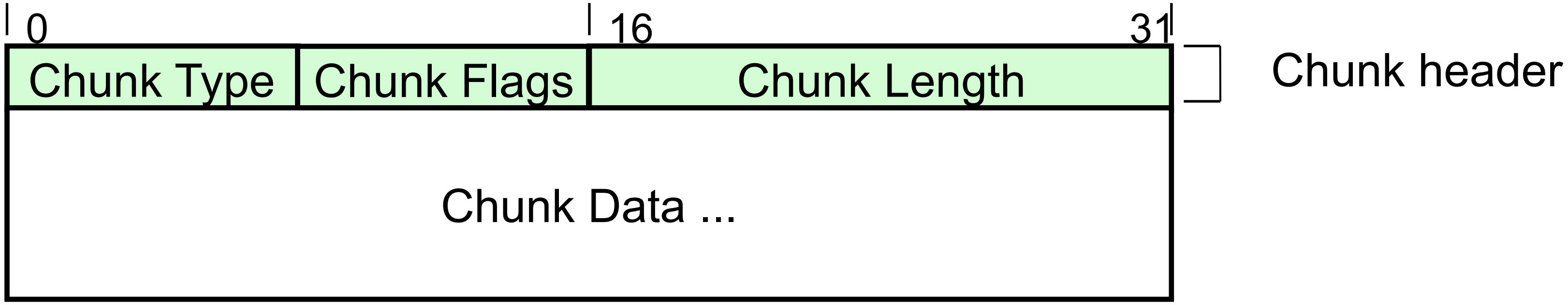


Random number that identifies association:
Used to distinguish new from old connections

Checksum of complete SCTP message (common header and “chunks”)

SCTP Chunk Format

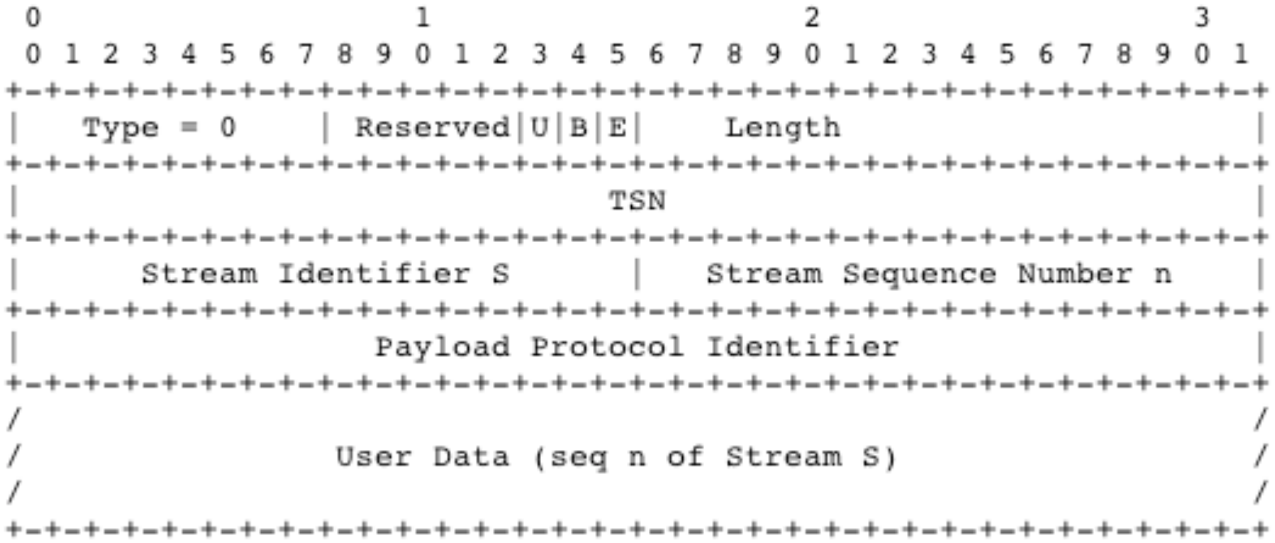
- Signaling information and data are transported in chunks
  - One or more chunks in a SCTP message
  - Each chunk type has a special meaning:
    - INIT, INIT-ACK, COOKIE, COOKIE-ACK ⇨ Connection setup
    - DATA ⇨ Transports user data
    - SACK ⇨ Acknowledge Data
    - SHUTDOWN, SHUTDOWN-ACK, SHUTDOWN-COMPLETION ⇨ Connection teardown
- Common chunk format



- Additional chunk formats are defined for specific chunk types

Data Transmission

- Application data is transmitted in Data Chunks
  - A data chunk is associated to a stream (Stream Identifier S)

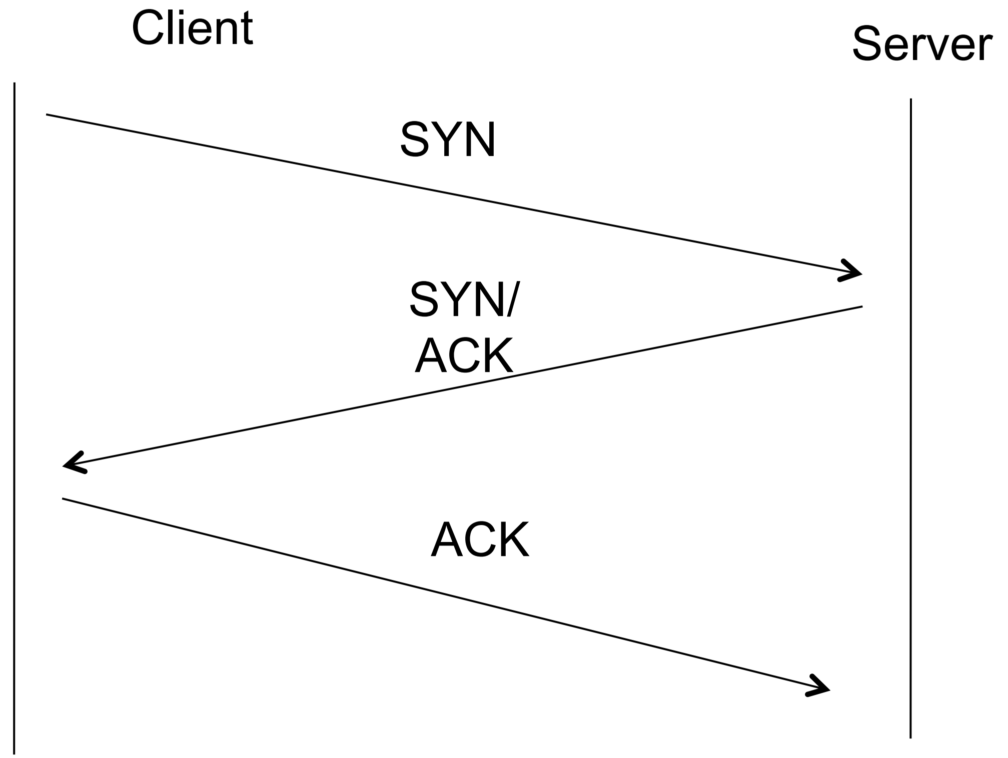


```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type = 0    | Reserved|U|B|E|   Length                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              TSN                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Stream Identifier S      |   Stream Sequence Number n    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Payload Protocol Identifier                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                 User Data (seq n of Stream S)                 /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- TSN (Transport Sequence Number)
  - Global Sequence Number
  - Similar to TCP sequence number, used for retransmissions
- Stream sequence number
  - Necessary for per-stream transmission reliability

Connection Setup – Example: TCP

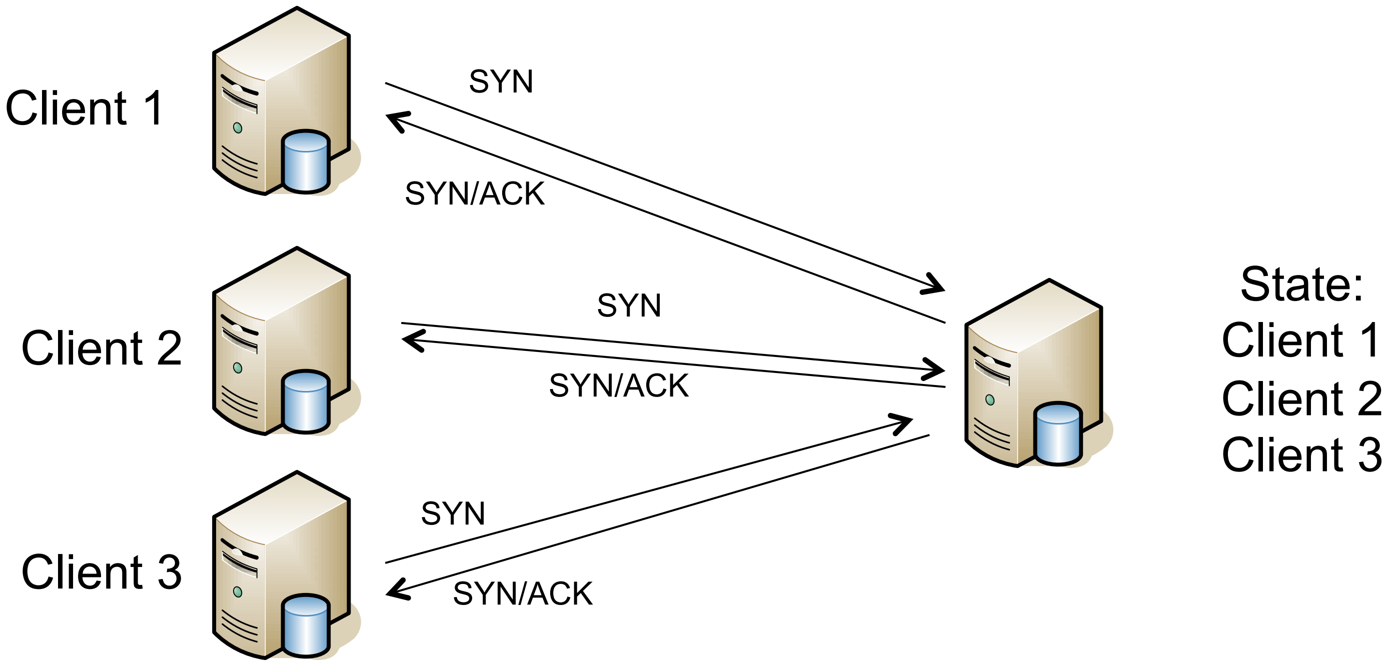
- TCP connection setup



Create State for
TCP connection: Store
client information

- Known Problem: TCP SYN-Flooding

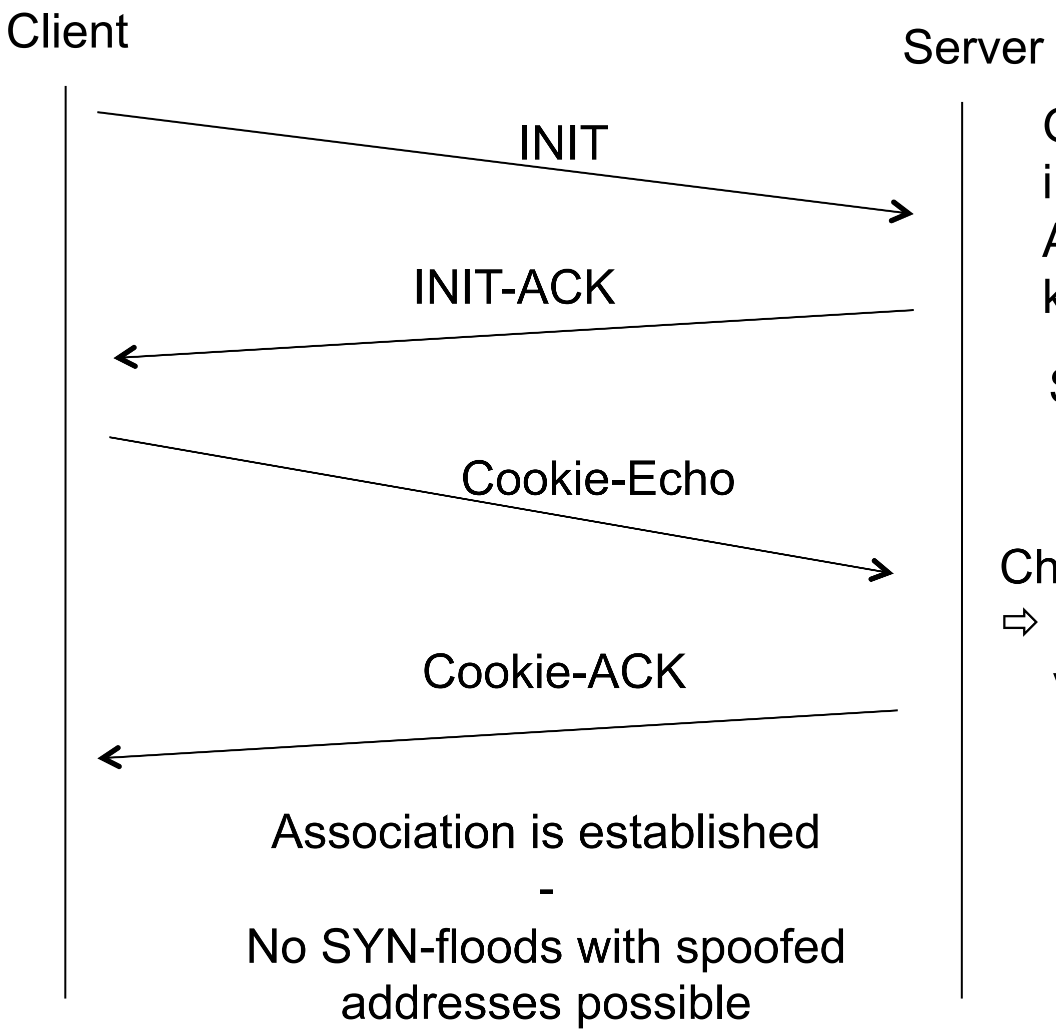
SYN Flodding



- ❑ Clients send SYN-Packets but do not respond to SYN-ACK
  - Usually done by a single client that performs IP address spoofing
  - Works because sending of single forged packet causes server to block valuable resources (TCB – Transmission Control Block)
- ⇨ Server has to store state until a TCP timeout occurs
  - May lead to resource exhaustion, during which server cannot accept new connections

SCTP Association Setup

- Solution to SYN-Flood problem (c.f. RFC 4960)



Generate client specific cookie: i.e. specific hash (Message Authentication Code - MAC) using key, TCB info (timestamp,…)

Send cookie ⇨ forget client

Check if cookie is valid
⇨ Create state only on valid cookie

Association is established
-
No SYN-floods with spoofed addresses possible

SCTP Origins and Design Philosophy

- Designed within IETF SIGTRAN (Signaling Transport) WG
- Goals include
  - Reliable service for Signaling System 7 (SS7); ISDN protocols
  - Support call management of SS7 by IP
  - Adapting Voice over IP (VoIP) to Public Switched telephone network (PSTN)
- RFC 2719: “Framework Architecture for Signaling Transport“
  - signaling gateway (SG): converting Common Channel Signaling (CCS) messages from SS7 to SIGTRAN; softswitch
- Ambition
  - Find applications for SCTP beyond original purpose within scope of IP-based telephony services
- SCTP APIs
  - APIs defined in RFC 6458: UDP-style API; TCP-style API

SCTP Standardisation

- RFC 2960 Stream Control Transmission Protocol (updated by RFC 3309 and obsoleted by RFC 4960)
- RFC 3257 Stream Control Transmission Protocol Applicability Statement
- RFC 3286 An Introduction to the Stream Control Transmission Protocol
- RFC 3309 Stream Control Transmission Protocol (SCTP) Checksum Change (obsoleted by RFC 4960)
- RFC 3436 Transport Layer Security over Stream Control Transmission Protocol
- RFC 3554 On the Use of Stream Control Transmission Protocol (SCTP) with IPsec
- RFC 3758 Stream Control Transmission Protocol (SCTP) Partial Reliability Extension
- RFC 3873 Stream Control Transmission Protocol (SCTP) Management Information Base (MIB)
- RFC 4460 Stream Control Transmission Protocol (SCTP) Specification Errata and Issues
- RFC 4820 Padding Chunk and Parameter for the Stream Control Transmission Protocol (SCTP)
- RFC 4895 Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)
- RFC 4960 Stream Control Transmission Protocol
- RFC 5043 Stream Control Transmission Protocol (SCTP) Direct Data Placement (DDP) Adaptation
- RFC 5061 Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration
- RFC 5062 Security Attacks Found Against the Stream Control Transmission Protocol (SCTP) and Current Countermeasures
- RFC 6096 Stream Control Transmission Protocol (SCTP) Chunk Flags Registration (updates RFC 4960)
- RFC 6458 Sockets API Extensions for the Stream Control Transmission Protocol (SCTP)

SCTP Congestion Control

- Based on Congestion Control in TCP
  - SCTP can ACK out-of-order blocks
  - out-of-order blocks do not count for congestion control
- Separate CC parameters kept for each destination address
- Parameters for unused destinations “decay” over time
- Each destination address begins with slow-start

SCTP Availability

- Availability
  - Linux, BSD
  - Solaris, HP-UX, AIX
  - VxWorks, QNX
  - Windows: SctpDrv
- Caveat
  - http://en.wikipedia.org/wiki/Stream_Control_Transmission_Protocol
  - „SCTP is sometimes a good fingerprinting candidate. Some operating systems ship with SCTP support enabled, and, as it is not as well known as TCP or UDP, it is sometimes overlooked in firewall and intrusion detection configurations, thus often permitting probing traffic. “

SCTP Discussion

- SCTP has attractive features
  - but to which extent is it used?
- Why do we use HTTP over TCP for Video Streaming?
- Firewall and NAT issues
  - many home routers / NAT middleboxes do no support SCTP
- Implementations
  - Not supported / by default included ubiquitously
- BUT: mandatory for some newly developed protocols such as IPFIX (IP Flow Information Export)
- Alternatives? (c.f. Multipath TCP)

Congestion Control

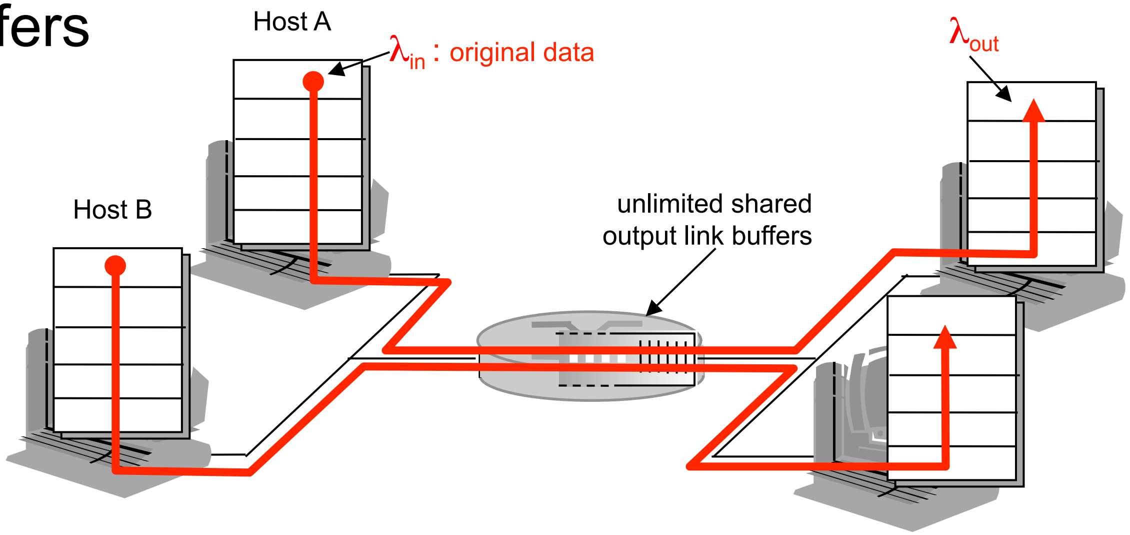
Principles of Congestion Control

Congestion:

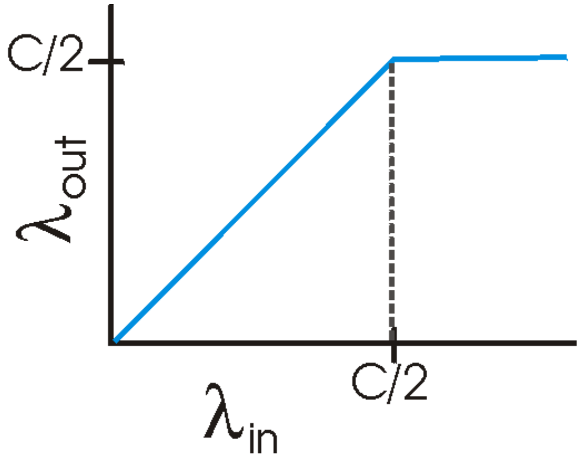
- informally: "too many sources sending too much data too fast for *network* to handle"
- different from flow control!
- manifestations:
  - lost packets (buffer overflow at routers)
  - long delays (queueing in router buffers)

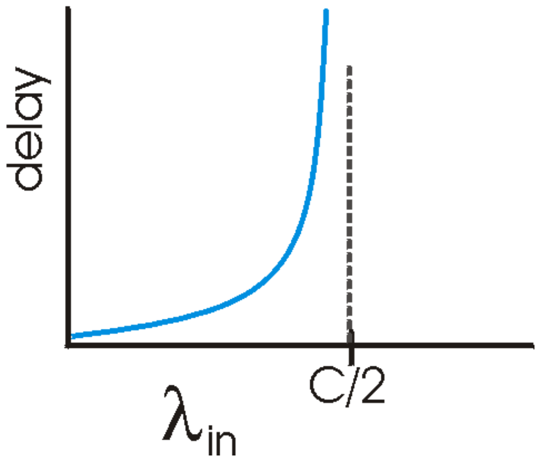
Causes/Costs of Congestion: Scenario 1

- two senders, two receivers
- one router, infinite buffers
- no retransmission


Host A
λ_{in} : original data
λ_{out}
Host B
unlimited shared output link buffers

λ_{in} : av. incoming load (variable interarrival times)

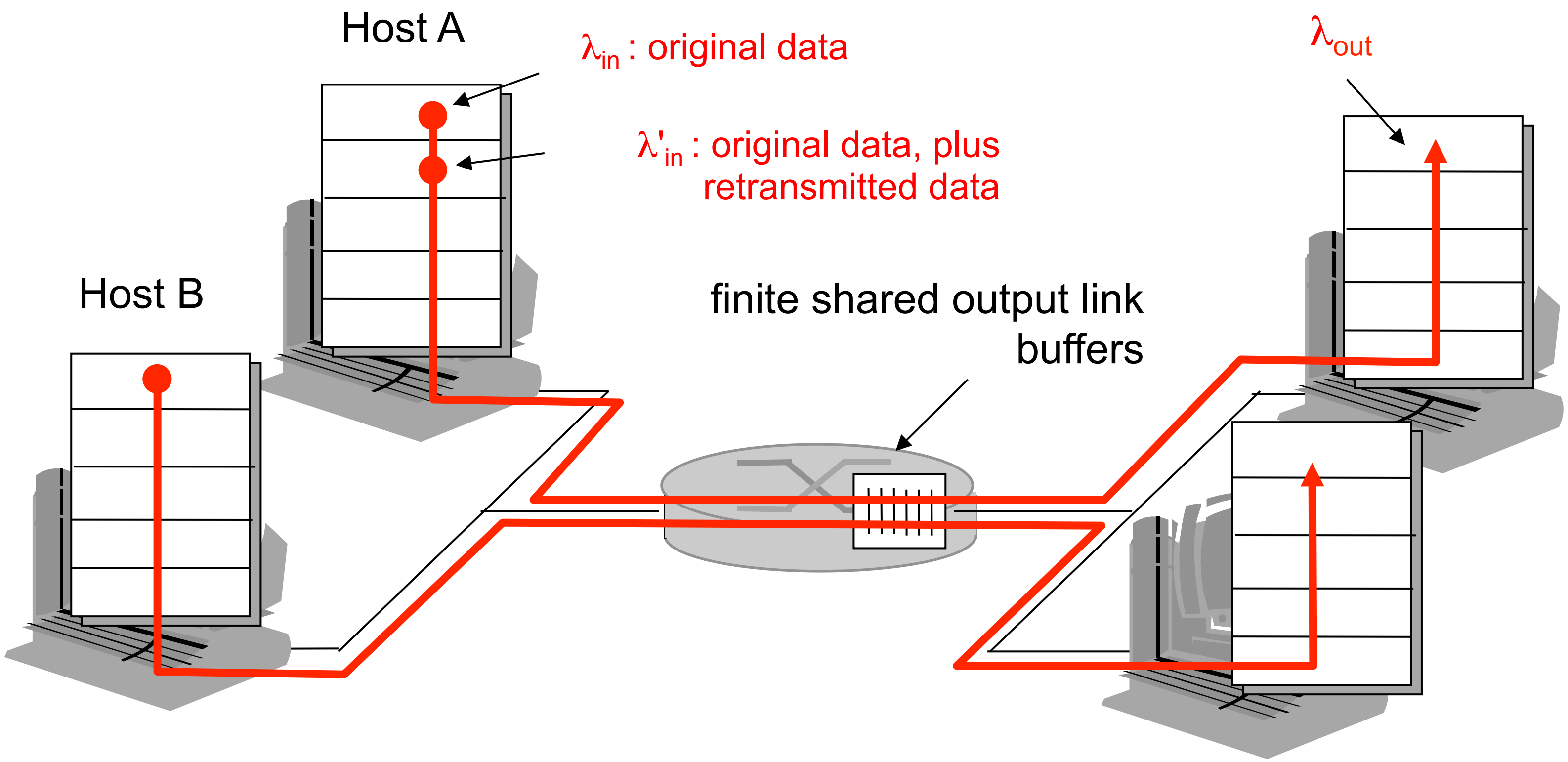




- large delays when congested
- maximum achievable throughput: C/2

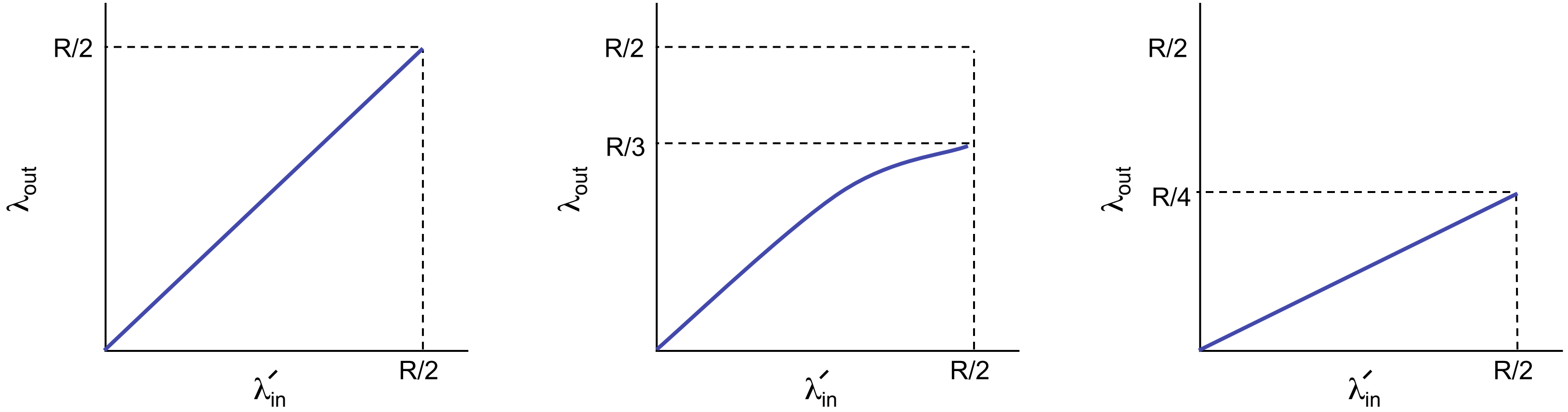
Causes/Costs of Congestion: Scenario 2

- one router, *finite* buffers
- sender retransmission of lost packets


Host A
λ_{in} : original data
λ_{out}
λ'_{in} : original data, plus retransmitted data
Host B
finite shared output link buffers

Causes/Costs of Congestion: Scenario 2

- a) always: $\lambda_{in} = \lambda_{out}$ (goodput)
- b) "perfect" retransmission when loss: $\lambda'_{in} > \lambda_{out}$
- c) retransmission of delayed (not lost) packet makes λ'_{in} larger (than perfect case) for same λ_{out}



a) no loss in router

b) retransmission of needed packets

c) with unnecessary retransmissions

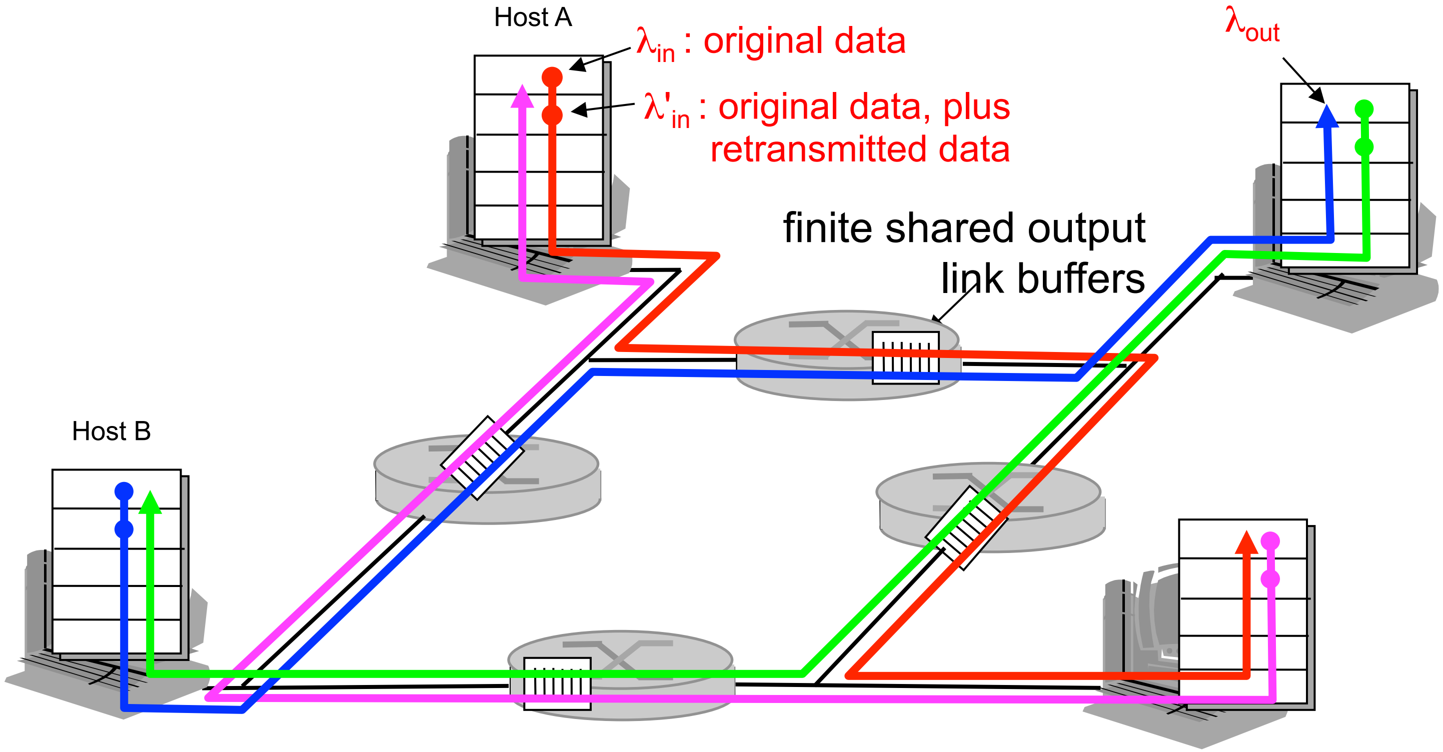
"costs" of congestion:

- more work (retransmissions) for given "goodput"
- unneeded retransmissions: link carries multiple copies of packet

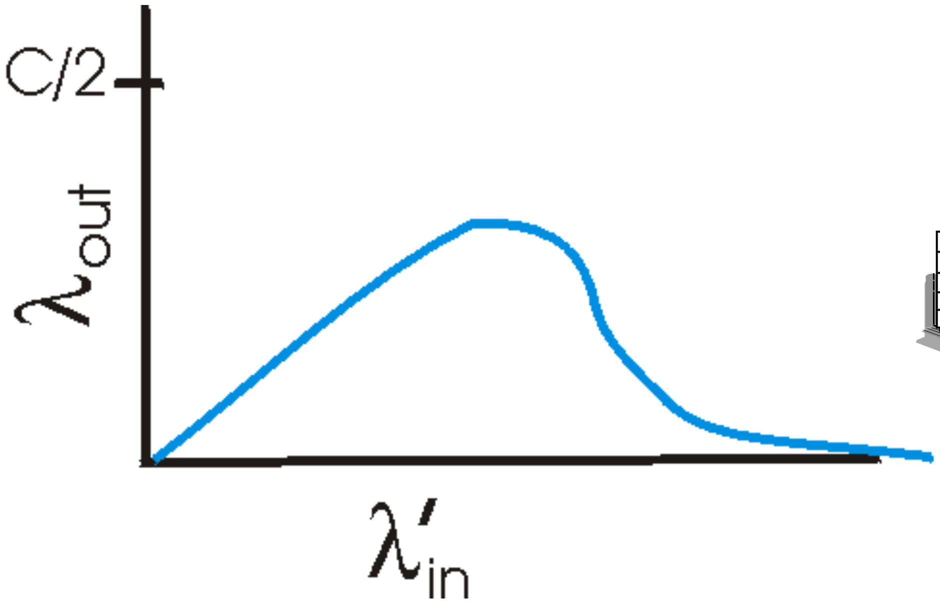
Causes/Costs of Congestion: Scenario 3

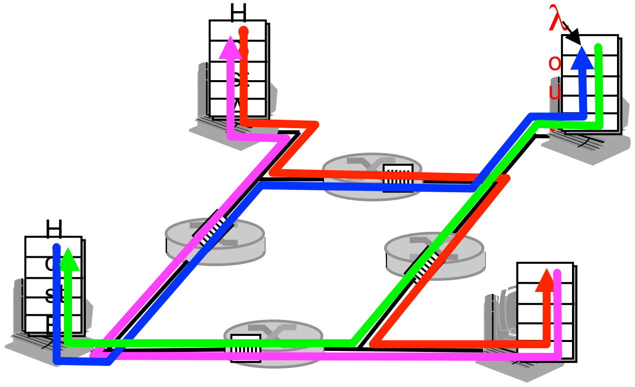
- four senders
- multihop paths
- timeout/retransmit

Q: what happens as λ_{in} and λ'_{in} increase ?


Host A
λ_{in} : original data
λ'_{in} : original data, plus retransmitted data
λ_{out}
finite shared output link buffers
Host B

Causes/Costs of Congestion: Scenario 3



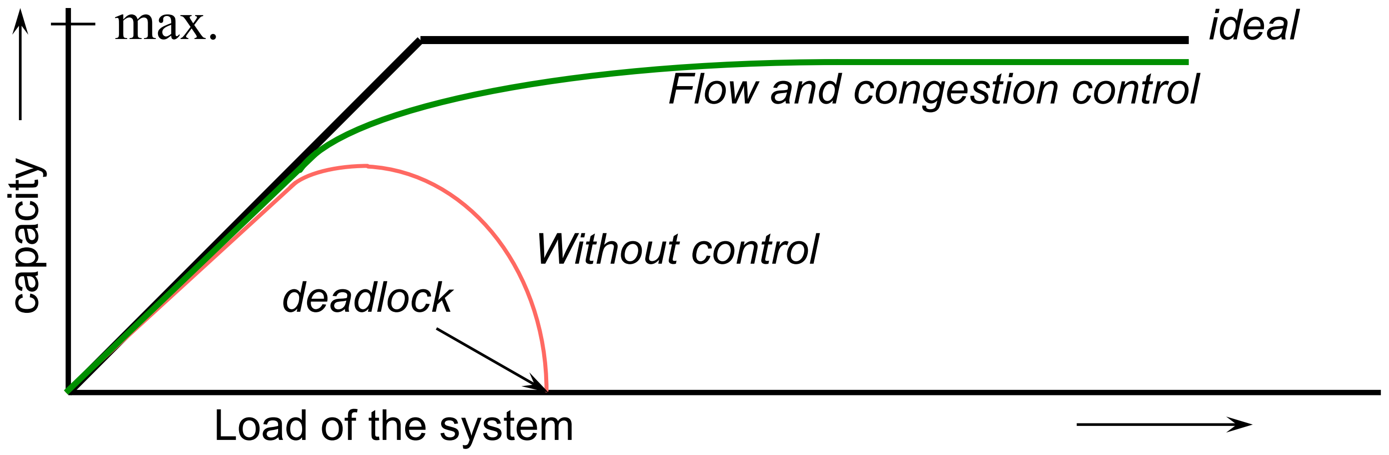


Another "cost" of congestion:

- ❑ when packet dropped, any "upstream transmission capacity used for that packet was wasted!

Congestion Control

- Goals and problems hereby
  - Reasonable behavior in case of network (over)load
  - Without controlling the outgoing amount of data, the capacity may drop to zero because of deadlocks
  - Fair ressource sharing
  - Criteria: effective, simple, robust, end-host driven



Approaches Towards Congestion Control

Two broad approaches towards congestion control:

End-end congestion control:

- no explicit feedback from network
- congestion inferred from end-system observed loss, delay
- approach taken by TCP

Network-assisted congestion control:

- routers provide feedback to end systems
  - single bit indicating congestion (SNA, DECbit, TCP/IP ECN, ATM)
  - explicit rate sender should send at

Congestion Control (Van Jacobson)

- Problem: the end host does not know a lot about the network.
  - It only knows if a packet has been delivered successfully or not
- Self clocking:
  - for every segment that leaves the network we can send a new one
- Assumption:
  - packet loss only because of congestion
  - Not true for wireless networks