1 Performance at Transport Layer

1.1 TCP

Characteristics

- Connection-oriented reliable byte-stream service
  - Two comms end points must establish, maintain connection
  - Byte stream data exchanged between end points
  - Reliable data delivery ensured

1.1.1 Principles

- Data broken into segments of certain size
  - Max. Segment Size (MSS) set (negotiated) at conn. estab.
  - MSS val carried by SYN segment
  - End point never TX segment larger than MSS
  - Higher MSS, better performance (min. OH vs payload ratio)
  - If not indicated, default val of 536 bytes is used for MSS
  - In reality, seg size 40 byteslarger as also includes
    - 20 bytes TCO header
    - 20 bytes IP header

1.1.2 Problem

- Data broken into segments of certain size
  - COmpute efficiency considering OH and payload when TX 4GB video and MSS of 1460and 536 used:
    - Case 1: No pkts: $4 * \frac{1024}{1460} = 2873; Headers : 40 * 2873 = 114920$
      - Efficiency = useful data/total data = $\frac{4194304}{4309224} = 97.33\%$
    - Case 2: No pkts: $4 * \frac{1024}{536} = 7826; Headers : 40 * 7826 = 313040$
      - Efficiency=useful data/total data= $\frac{419304}{4507344} = 93.05\%$

1.1.3 Principles

- Reliability through ack and reTX
- When TX seg, timer set
- RX expected to ack seg
- If ack not RX, seg is reTX

1.1.4 Issues

- ReTX TimeOut (RTO) period is variable
  - RTO set in relation to RTT
  - RTT estimated using smoothed estimator (SRTT) using a low-pass filter
  - After unsuccessful reTX, TO period doubled (exponential backoff) with an upper limit
- Adter series of unsuccessful reTX, conn. is reset
  - TCP implementation have Keepalive timer, not present in TCP standard

1.1.5 Problem

- Exponential backoff reTX
  1. List times at which reTX occurs if there is a need for 6 reTX and the first two are 2s apart
     - ReTX: 1,3,7,15,31,63
  2. List times at which reTX occur if there is need for 10 reTX and the first two are 1.5s apart
     - ReTX: 1,2.5,6.5,11.5,23.5,47.5,96.5,159.5,223.5,287.5

1.1.6 RTT Estimation

- Calculated every time new measurement performed
- $STT = \alpha * SRTT + (1 - \alpha) * MRTT$, where
  - SRTT is RTT estimator
  - α is smoothing factor with rec. val between 0.8 and 0.9
  - MRTT is measured RTT

1.1.7 RTO Value

- Calculated every time there is need for reTX
- $RTO = min(RTOMax, max(RTOMin, (\beta * SRTT)))$, where
  - RTOMax is upper bound on timeout (e.g. 1m, 64sec)
  - RTOMin is lower bound on timeout (e.g. 1s)
  - β is delay variance factor with dixed val between 1.3 and 2
  - If ack not RX, seg is reTX

1.1.8 ISsues

- RTT estimation accuracy problem
  - RTT estimation alg assumes RTT variations are small, constant
  - Loses accuracy with wide fluctuations in RTT, causing unnecessary reTX
  - ReTX add traffic to already loaded net.
- Jacobson's soln.
  - Keep track of both mean and variance of RTT, compute RTO based on both

1.1.9 RTT Average and Mean Deviation

- Calc every time new measurement performed
- $Err = MRTT - ARTT$
  - Err is error between measured val and smothered value for RTT
  - ARTT is smothered RTT average
  - g is gain factor with rec. value of 1/8
  - MRTT is measured RTT

1.1.10 RTT MEan Deviation

- Calc. every time new measurement performed
- $DRTT = DRTT * h * (|Err| - DRTT)$
  - DRTT is smothered mean deviation
  - h is difference factor set to 0.25

1.1.11 RTO value

- Calc. every time need for reTX
- $RTO = ARTT + r * DRTT$, where
  - r is constant set to 4
  - Initial val set for r was 2, later changed to 4

1.1.12 Issues

- RTT measurement accuracy problem
  - RTT measured between sending of data pkt and its ack
  - When Large delays occur, timer goes off, reTX takes place
  - When RX ack, no way to know if it was delayed res. to orig data seg or res. to reTX seg

- Karn's soln
  - Not to update RTT estimator with info on reTX seg's performance
- Limitations of Karn;s soln
  - When RTT increases sharply, normal reTX resumed after series of reTX and TCP does to receive ack, for a while RTT not updated, reTX would happen considering old RTT val
- Limitations soln
  - Exponential backoff timer timout val employed: timout=2*timout
- Solution isse
  - If to increases too much, delays added with no correlation with actual net. delay
- Solution fix
  - Upper limitations added to to value > 1 min: 64s

1.1.13 Computation of RTT estimation

1.2 Congestion control

- Modern TCP std include 4 major alg
  - Slow start
  - COngestion avoidance
  - Fast reTX
  - Fast recovery

1.3 Slow Start

1.3.1 Principle

- Slowly probes net in order to determine available capacity
- Employed at beginning of transfer/after loss detected by TX timer
- Uses following var.
  - COngestion window (cwnd) - sender side limit on amount which can be TX before receiving ack
  - Receiver window (rwnd) - Receiver-side limit on outstanding data
  - Slow start threshold (ssthresh) - limit to decide using slow start or congestion avoidance
  - Sender Maximum Segment Size (SMSS) - Max seg size at sender
  - Flight Size - amount of unack data in TX between sender, receiver
- TX should exchange min of cwnd and rwnd amount of data
- Slow start used when cwnd > ssthresh, Congestion Avoidance emploued for cwnd < ssthresh and either alg when cwnd = ssthresh

1.3.2 MEchanism

-
-
-
-
-

1.3.3 Note

-

1.3.4 Problem: ACK Division

- ‘
-
-

1.3.5 UPdated Mechanism

-
-
-

1.3.6 Performance Issues

- - *
- - *
  - *
- - *
  - *
  - *
    *
    *
    *
    *

1.4 Congestion Avoidance

1.4.1 Principle

1.4.2 Mechanism

1.4.3 Updated MEchanism

1.4.4 Note

1.5 Fast Retransmit

1.5.1 Principle

1.5.2 Mechanism

1.6 Fast Recovery

1.6.1 Principle

1.6.2 Mechanism

1.6.3 Note

1.7 Fast Retransmit and Fast Recovery

1.7.1 PRinciple

- Two major alg types that improve Fast reTX and Fast recovery
- Based on TCP selective Ack
  -

2 TCP Tahoe

2.1 Characteristics

- Fast recovery not included
- Fast reTX not included
- TXP old tahoe did not have fast reTX either
- Implemented in Unix 4.3 BSD Tahoe

2.2 ISsues

- ONly mechanism to detect loss is through reTX timer timeout
  - Introduces potential delays
- TCP old tahoe by not emplying dast reTX alg, slow start has to be used
  - Rates kept low
- Every lost pkt determines cwnd reste to min
  - Severe reduction in rates

3 TCP Reno

3.1 Characteristics

3.2 Issues

4 TCP NEw Reno, SACK and Vegas

4.1 TCP New Reno

4.1.1 Characteristics

4.1.2 ISsues

4.2 TCP SACK

4.2.1 Characteristics

4.2.2 Issues

4.3 TCP Vegas

4.3.1 Characteristics

4.3.2 Issues

5 SCTP

5.1 Motivation

- TCP limitations with wireless and mobile comms
- Need for multi-streaming
- Need for multi-homing

5.2 OVerview

- Series of IETF 2960 (2000), IETF RFC 3286 (2002)

5.3 Features

- Reliable transport protocol
- Uses association instead of conn.
- Designed for message oriented applications
  - Framing (preserve message boundaries)
- Ack error free transfer of msg
- Detection of data corruption, data loss and data duplication
- Selective reTX to corect lost or corrupted data
- Active monitoring of session conn. via heartbeat

- Resistance to DOS attacks
  - 4-way handshake
- Supports multi-streaming
  - Up to 64K indp. ordered streams
- Supports multi-homing
  - Set of IP addresses per endpoint

5.4 Message Format - 1:HEader

- Src Port and Dest Port (2+2 bytes)
  - Same port concept as TCP and UDP
- Verification Tag (4 bytes)
  - Exchanged etween endpoints at startup to validate the sender
- Checksum (4 bytes)
  - Uses CRC32 alg

5.5 MEssage Format - 2: Chuncks

- Type (1 byte)
  - Control or Data: e.g. Data, Init, SACK
- Flags (1 byte)
  - Carry info depending on type
- Length (2 bytes)
  - Chunk length, including data payload length
- Data (N bytes)
  - Variable length payload

5.6 Message Format - 3: Important Chunk Types

- DATA
  - IDs chunks carrying data
- INIT, INIT-ACK, COOKIE-ECHO, COOKIE-ACK
  - USedfor association establishment
- HEARTBEAT, HEARTBEAT-ACK
  - Used for keep-alive chacking
- SHUTDOWN, SHUTDOWN-ACK
  - USed for graceful disconn.

5.7 Establishing an Association

Association Establishment Procedure

-

5.8 INIT Chunk

- Initiate Tag
  - Receiver stores Initiate Tag Value
  - Must be placed in VErification Tag field of every SCTP pkt receiver sends
- Advertised Receiver Window Credit (a_rwnd)
  - Indicates dedicated buffer space sender reserved for this association
- Number of Outbound Streams (OS)
  - Number of outbound streams sender wishes to create in this association (max 64k)
- Number of Inbound Streams (I-TSN)
  - Defines initial TX seq number the sender will use
  - Field may be set to value of Initiate Tag Field
- –

5.9 INIT-ACK Chunk

5.10 CECHO and CACK Chunks

5.11 DATA Chunk

5.12 Terminating an Association

5.13 SHUTDOWN Chunks

5.14 Multihoming

SCTP Association:

- Comm. hosts use set of IP addr. instead of single one each
- Multi comms path may be set up
  - One primary path
  - No. of secondary paths
- Lists of IP addrs exchanged between hosts during init of assoc.
  - Both INIT and INITACK msgs include list of IP addrs
- Source of INIT msg is dest of INIT-ACK

    - In general, determine primary path

SCTP Operation:

- HOsts monitor data TOs and No. of reTX to determine path's transmission quality

- ReTX Data chunks may be sent over multipaths if status of one path is suspect

- Faulty paths marked "Out of Service"

- HEartbeat chunks sent periodically to all inactive IP addr

- Non-responding IP addrs will be marked "Out of Service"

6 mSCTP

Mobile SCTP

- Extends SCTP
    - Adds Dynamic Address Reconfiguration (ADDIP)
    - Enables SCTO to add, delete, and change existing IP addrs attached to an assoc. during an acrive conn.
    - Enables support for seamless handover for mobile hosts that are moving between IP networks
    - Uses ASCONF and ASCONF-ACK
        * Add new IP addrs to assoc
        * Change primary IP of assoc
        * Delete old IP addr from assoc

7 DCCP

Motivation

- UPD and TCP limitations with realtime transport of data

- Need to support real-time data transfers over wireless links

Overview

- DCCP is novel non-reliable transport layer protocol

-

-

Features

-

Datagram Format 1:

- Headers - Long Sequence No.

Datagram Format 2:

- Headers - Short seq no.
- Acknowledgements - Short seq no.
- Options and Data

Datagram Fields

-

Packet Types

-

Connection Setup

- 3-way handshake

Data exchange

- Two endpoints exchange Data pkts and ack pkts acking data
- Optionally DataAck pkts containing data and acks can be exchanged
- If one endpoint has no data to send it will send ack pkts exclusively

Connection Close

- 3-way handshake

8 Congestion Control-Related Schemes

Drop Tail

- Involves default queue mechanisms
- Drops all pkts exceeding queue length
  - Any TCP-based receiver reports loss in ACK pkt
  - Most often sender adapts to loss by multiplicatively decreasing
- One loss event is very likely to be dollowed by series of loss events
  - Little or no space in queue
- If adaptive senders need some time to respond

Random Early Detection (RED)

- Uses active queue management
- Drops pkt in intermediate node based on av. queue legnth exceeding a thresh

- Any TCP receiver reports loss in ACK pkt
  - Most often sender adapts to loss by multiplicatively decreasing rate

- RED experiences mostly singular loss events

- Gives time to adaptive senders to respond

Early Congestion Notifications

- End-End congestion avoidance mechanism
  - Implementedin routers and supported by end-systems
  - Not multimedia-specific, very TCP-specific

- Uses two IP header bits
  - ECT - ECN Capable Transport, set by sender
  - CE - COngestion Experienced, may be set by routers

- If pkt has ECT bit 0,
  - ECN acts as RED

- If pkt has ECT bit 1:
  - ECN node sets CE bit
  - Any TCP receiver sets ECN bit in ACK
  - As result most often sender applies multi decrease

- EXN-pkts never lost on un-congested links

- Distinction between loss and marked pkts
  - TX window can decrease
  - No pkt loss and no reTX

Early Congestion Notification (ECN) Nonce

- Optional addition to ECN

- Improvies robustness of congestion control

- Prevents receivers from ecploiting ECN to gain unfair share of net. BW

- Protects against accisdental/malicious concealment of marked pkts from sender

eXplicit Congestion Notiication (XCN)

- Protocol for Connections with high BW-delay product

- Routers return explicit feedback to host

- Hosts use feedback from routers to change their congestion window

9 TCP over Wireless

Motivation

- LArge percentage of traffic is reliable:
  - File Trandfer (FTP)
  - Web Traffic (HTTP)
  - Command Based (TELNET, SSH)
- TCP very popular in wired networks
  - Very good congestion control
  - Very good congestion avoidanve

TCP in Wireless Networks

- Pkt loss in wireless networks occurs due to:
  - Bit errors due to wireless channel impairments
  - Handovers due to mobility
  - Congestion (rarely)
  - Pkt reordering (rarely)
- TCP assumes pkt loss is due to:
  - COngestion in the net.
  - Pkt reordering (rarely)

Problems with TCP over Wireless NEtworks

- Congestion avoidance can be triggered by pkt loss
  - TCPs mechanisms do not respond well to pkt loss due to bit errors and handoffs
  - Efficiency of TCP-based transfers suffer
- Error bursts may occur due to low signal stregth or longer period of noise
  - More than one pkt lost in TCP
  - More likely to be detected as TO -> TCP enters slow start
- Delay is often very high and variable
  - RTT can be very long and variable
  - TCPs TO mechanisms may not work well
  - Problem exacerbated by link-level reTX
- Links may be asymmetric
  - Delayed ACKs in slow dirn. limit throughput in fast dirn

Solutions for TCP over Wireless Networks

- Link-Layer approaches (A)
  - Hide losses not caused by congestion from the transport-layer sender
  - Makes link appear to be more reliable than it is in reality
  - Solns:
    * Use frame reTX
      · Link-level automatic reTX request (ARQ)
    * USe error correction codes
      · Forward Error Correciton (FEC)
    * Use hybrid solutions
      · ARQ and FEC
- Advantages
  - Requires no change to existing sender behaviour
  - Matches layered protocol stack model
- Disadvantages
  - Negative TCP effect:
    * Delays due to link-level TO and reTX may trigger TCP fast reTX
    * TX efficiency decreases
  - Soln to negative TCP effect
    * Make link-level protocol TCP-aware
- Example: Snoop TCP
  - Advantages
    * Attempts to reTX locally, suppress duplicate ack
    * State is soft, handoff simplified
  - Disadvantege
    * May not completely shield TCP from effects of mobility and wireless loss

Split Connection Approaches (B)

- Divide single TCO conn. into two conn.
- Isolate wired net. from wireless net
- OFten split at base station or access point
- soln
  - Use TCP on wired net
  - Enhanced protocol over wireless net
- Advantages
  - Clarity of approach

  - Each of the protocols performs best in its setup
- Disadvantages
  - Extra protocol OH
  - Violates end-end semantics of TCP
  - Complicates handoff due to state info at access point or base station where protocol is "split"
- Example
  - Indirect TCP

End-to-End Approaches (C)

- Make sender aware that some losses are not due to congestion
- Avoid congestion control when not needed
- Solns
  - Selective ack (SACKs)
  - Explicit loss notificaiton (ELN) distinguishes between congestion and other losses
- Advantages
  - Maintains end-end semantics of TCP
  - Intros no extra OH at base stations for protocol processing or handoff
- Disadvantages
  - Requires modified TCP
  - May not operate efficiently e.g. for pkt reordering versus pkt loss
- Example
  - SMART

10 Snoop TCP

OVerview

- Link-layer protocol that snoops passing TCP data and acks
- Employs Snoop agent between two endpoints
- Data from Fixed Host to Mobile Host
  - Cache unack'd TCP data
  - Perform local reTX
- Data from Mobile Host to Fixed Host
  - Detect missing pkts

- Perform negative ack

Architecture

-

Fixed Host to Mobile Host Operation

- If new pt rec. in normal TCP seq
  - Add to snoop cache
  - Forward to Mobile Host
- If out of seq pkt cached earlier arrives
  - Fast reTX/TO at send due to:
  - if last_ACK $<$ crt_seq_no
    * Loss in wireless link - Forward to Mobile Host
  - if last_ACK $>$ crt_seq_no
    * Loss of previous ACK - send ACK to Fixed Host with Mobile Host addr and port
- If out of seq pkt not cached earlier arrives
  - if seq_no far from last_seq_no
    * Congestion in fixed network
      · Forward to Mobile Host
      · Mark as reTX by sender
  - if seq_no close to last_seq_no
    * Out of order delivery

Mobile Host to Fixed Host Operation

- If new ack rec. in normal TCP operation
  - Normal Case
    * Clean snoop cache
    * Update RT estimate
    * Forward ack to Fixed Host
- if spurious ack rec.
  - Discard
- If duplicate ack rec.
  - If pkt not in snoop cache
    * Lost in fixed net.
      · Forwared to fixed host
  - If pkt marked as sender reTX
    * Forward to Fixed Host

- If unexpected (first after a pkt loss)
  - Lost pjt on wireless link
    - ReTX ar higher priority
- If expected (subsequent adter one lost)
  - Discard

Advantages

- Improved performance in wireless net.
- No change to TCP at fixed host
- No violaiton of end-end TCP semantics
- No recompiling/re-linking of existing applications
- Automatic fallback to standard TCP
  - No need to ensure all foreign net. provide Snoop agent

Disadvantages

- Does not fully isolate wireless link errors from the fixed net.
- Mobile host must be modified to handle NACKs for reverse traffic
- Cannot snoop encrypted datagrams
- Cannot be used with authentication

11 Indirect TCP (I-TCP)

Overview

- Hides pkt loss due to wireless from sender
- Wireless TCP can be independently optimized
- Good performance in case of wide-area net.
- reTX occurs only on bad link
- Faster recovery due to relativelty shortRTT for wireless link
- Handoff requires state transfer
- Buffer space needed, extra copying at proxy
- End-end semantics violation needs to be augmented by apllication level

Architecture

-

Advantages

- No changes to TCP at fixed hosts

- Wireless link errors are corrected at the TCP proxy and do not propagate to the fixed net.
- New "wireless" protocol affects only limited part of internet
- Possible further optimizaitons over wireless link
- Delay variane between proxy and mobile host is small -> optimised TCP
- Opportunity for header compression
- Opportunity for different transport protocol

Disadvantages

- Loss of TCPs end-end semantics
- Addition of third point of failure (proxy) apart from fixed, mobile hosts
- Handover can be significant
- OH at proxy for per pkt processing
- TCP proxy must be trusted
- Opportunities for snooping and DOS attacks
- End-end IP-level privacy and auth. must terminate at proxy
- Proxy failure may cause loss of TCP state

12 Other Approaches