# Integrating Satellite links within the Internet

Cours DEA-RSD

Jeudi 26 Fevrier 1998

*Walid Dabbous*

INRIA Sophia Antipolis

# Outline

- Introduction
  - Hybride Satellite Networks architecture
  - Satellite link specificities
- TCP over satellite
  - Why TCP (and IP) does not perform well
  - Standard mechanisms to enhance performance
  - Research issues related to TCP over satellite

# Outline (contd)

- Other solutions
  - STP
  - SCPS-TP
- Dynamic routing problem
  - the udlr working group
  - RPM or Tunneling
  - details of the tunneling solution
  - open problems

# Hybrid Satellite network architecture

- voir version .ps

# Satellite link specificities

- Advantages
  - natural broadcast capability
  - reach remote areas or
  - countries with little terrestrial infrastructure
  - reach mobile users

# Satellite link specificities

- Long feedback loops
- Large BDP (bandwidth delay product)
- Error prone links
- Limited bandwidth
- Asymmetric use
- Variable round trip time
- Intermittent connectivity

# Delay and connectivity

- Propoagation time to travel earth-sat-earth
  - between 239.6 and 279.0 ms for GEO
  - RTT no more than 558 ms (if sat is used for the return path). Hurts interactive applications.
- ranges from several ms to 80 ms in LEOs
- Variable path delay
- Handoff in non-GEO can cause packet losses

# Transmission erros

- Strength of signal in $1/d^2$
  - low S/N ratio
- Advanced error control coding
  - Reed Solomon
  - too much overhead
- Many current systems do not provide error free service

# Limited bandwidth & Asym. use

- Limited resource : radio spectrum
  - Satellites will not provide Terrabytes speeds
    - C (6/4)
    - Ku (14/12)
    - Ka (30/20)
- Asymetric satellite networks
  - udlr case
  - less avaialable capacity for uplink than downlink

# TCP over Satellite

- Why TCP/IP does not perform well
- Standard mechanisms to enhance performance
- Research issues related to TCP over satellite

# IP performance issues

- IP TTL
  - to protect against loops
  - avoid to send two datagrams with the same id before TTL expired (TCP assumes 2 min!)
- IP fragmentation
  - different MTUs
  - problem in reassembley if two datagrams have the same id

# TCP performance issues

- Expectations
  - Share the link effectively (aggregation)
  - fill an otherwise idle link (high speed applications)
- TCP sequence numbers
  - cumulative ack, 32 bit sequence space
  - max throughput of 286 Mbps

# TCP performance issues (contd)

- TCP transmission window
  - maximum value of 64 Kbytes (16-bit field)
  - max throughput of 1048,560 bps for 500 ms RTT.
- Ack Strategy
  - Go-Back N takes more time to recover from losses than selective repeat

# TCP performance issues (contd)

- Slow start
  - start with 1, increase exponentialy
  - until cwnd exceeds ssthresh or loss occurs
- Congestion avoidance
  - increase window slower than ss (1 seg/RTT)
  - Long delay satellite forces poor utilisation
  - Path MTU discovery is important

# TCP performance issues (contd)

- Fast retransmit
  - implemented in Reno (April 1990)
  - 3 duplicate ACKs -> packet loss
  - retransmit the packet and
- Fast recovery
  - ssthresh=cwnd;
  - cwnd = cwnd/2; cwnd += 1; (for each dup ack)
  - when «new» ack is received cwnd =ssthresh;

# Enhance with std mechanisms

- Lower level enhancements
  - Path MTU discovery
    - increasing TCP congestion window is segment based
    - can cause long pause before data send (e.g. when the limited size networks are after the satellite link)
    - large packets are not a problem if link level FEC is supported
  - FEC: reduce corruption losses

# Standard TCP mechanisms

- TCP-Large windows (RFC1323)
  - window scale option
  - Protection Against Wrapped Sequence space (PAWS)
  - RTTM (Round-Trip Time Measurements)
- Selective Acknowledgments (RFC2018)
  - inform sender exactly which packets arrived
  - reduces transmission period
  - better evaluate available path bandwidth, and avoid ss.

# Ongoing research proposals

- Connection set-up
- Sharing TCP state among similar connections
- Slow start
  - Larger inital windows
  - Byte counting
  - Disabling delayed ACKs during slow start (*)
  - terminating slow start

# Ongoing research proposals

- Loss recovery (*) (see section on SCPS-TP)
  - Sack and non-Sack mechanisms
  - Explicit Congestion Notification
- Pacing TCP segments
- TCP Spoofing
- TCP header compression
- Multiple data connection

# Connection set-up

- TCP uses 3-way handshake (1 or 1.5 RTT)
- Eliminate by using T/TCP (RFC1644)
  - bypass 3-way handshake
  - send data in the first packet sent
    - use a connection identifier
  - helpful for short request/response traffic
  - use TCB state sharing to provide meaningful RTT

# Sharing TCP state

- Use persistent state to «stay tuned»
- among sequence of connections
- or concurrent connections
- What to share
  - initial window size
  - initial MSS size,...
- Security implications

# Slow start

- Important safe-guard against congestion
- window growth proportional to RTT
- Delayed ACKs
  - ACK every other packet
  - another source of wasted capacity during ss.

# Larger Initial Window

- Start with an initial window of more than 1:
  - value: min (4*MSS, max (2*MSS, 4380 bytes))
    - MSS/w : 1460/ 3, 512/4, 3000/2
  - Reduce transmission time (if size<4K -> 1RTT)
  - receiver sends an Ack even with Delayed Acks
  - may suffer packet loss in FIFO routers
  - Unnecessarily retransmitted packets (not too large)
  - Segements dropped later in the network (only for multiple congested links)

# Byte Counting

- Mitigate delayed ACKs problem
- increase cwnd based on number acked bytes (and not on the number of ACKs)
- reduce the time to increase the window
- Increase burstiness
- Needs further study in a congested environment with competing traffic

# Terminating slow start

- Slow start terminated when
  - the size of the window reaches ssthersh (receiver advertized window)
  - based on the spacing between the first few Acks, determine bottleneck link bandwidth using PP, and set ssthresh accordignly.
  - Reducing ssthersh allow to avoid overwhelming the network with segments

# Pacing TCP segments

- ACK spacing
  - routers seperate ACKs in order to spread bursts
  - seperate each Ack by two segments
  - this allow the sender to transmit at the correct rate
  - not necessary in asymmetric networks
    - inherent slowness of the returning Acks

# TCP spoofing

- Intermediate gateway (uplink) acks packets
- suppress the actual ack or retransmit
- breaks the end-to-end semantics
- cannot be used with IP encryption
- acceptable in many applications
  - WWW browsing through proxies
  - X windows proxies

# TCP header compression

- How
  - replace fized ot slowly changing data by cnx-id
  - use delta encoding after first packet
- Why
  - Reduce header overhead (telnet, limited link capacity)
  - Reduce PER in case of uniform BER (sat)

# Multiple data connections

- Use «N» TCP connections
  - makes applications N times more aggressive
  - may causes congestion during ss/cong. avoi.
  - avoid the use of TCP LW
  - less dramatic reduction in w for single losses
    - W*((2N - 1)/2N) instead of W/2
  - Not too social

# Other solutions

- Split solutions
  - TCP (again!)
  - STP (for the satellite link)
    - large modulus
    - 32-bit aligned, trailer information
    - selective retransmission
    - seperate control flow (reduce per-packet overhead)
    - not based on timers (a la XTP, NETBLT, Datakit, Delta-t, etc..)

# SCPS-TP

- MITRE work on Space Communication Protocol Standards- Transport Protocol
- Goal is to solve space related problems
  - error-prone links (congestion, corruption, link outage)
  - asymmetric channels
  - limited link capacity
  - Intermitted Connectivity

# SCPS-TP

- A set of extensions to TCP
  - rfc1323, Selective NACKs, header compression
  - TCP Vegas congestion control
    - low-loss congestion control mechanims
    - invoked only when errors are congestion-inducted
      - either by source default assumption or ECN message.
  - open-loop token bucket rate control with specific handling in case of corruption
  - coping with asymmetry: drastically delay ACKs

# Partial Conclusion

- A lot of things to study and evaluate....