15-744: Computer Networking

L-18 Mobile Transport

Mobile Transport

- TCP on wireless links
- Assigned reading
  - [BPSK97] A Comparison of Mechanism for Improving TCP Performance over Wireless Links

Wireless Challenges

- Force us to rethink many assumptions
- Need to share airwaves rather than wire
  - Don't know what hosts are involved
  - Host may not be using same link technology
- Mobility
- Other characteristics of wireless
  - Noisy → lots of losses
  - Slow
  - Interaction of multiple transmitters at receiver
    - Collisions, capture, interference
  - Multipath interference

Overview

- TCP Over Noisy Links

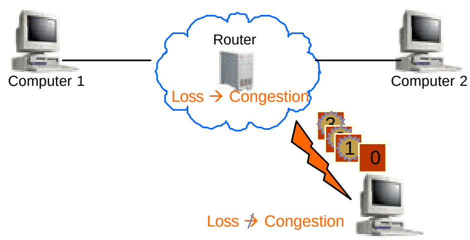
TCP Problems Over Noisy Links

- Wireless links are inherently error-prone
  - Fades, interference, attenuation
  - Errors often happen in bursts
- TCP cannot distinguish between corruption and congestion
  - TCP unnecessarily reduces window, resulting in low throughput and high latency
- Burst losses often result in timeouts
- Sender retransmission is the only option
  - Inefficient use of bandwidth

Constraints & Requirements

- Incremental deployment
  - Solution should not require modifications to fixed hosts
  - If possible, avoid modifying mobile hosts
- Probably more data to mobile than from mobile
  - Attempt to solve this first

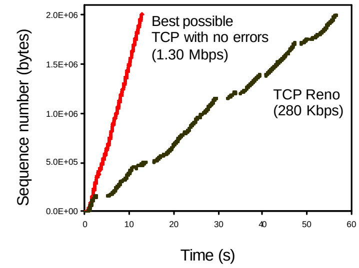
Challenge #1: Wireless Bit-Errors



Burst losses lead to coarse-grained timeouts

Result: Low throughput

Performance Degradation



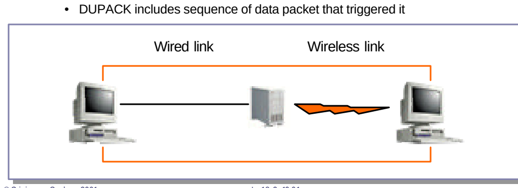
2 MB wide-area TCP transfer over 2 Mbps Lucent WaveLAN

Proposed Solutions

- End-to-end protocols
  - Selective ACKs, Explicit loss notification
- Split-connection protocols
  - Separate connections for wired path and wireless hop
- Reliable link-layer protocols
  - Error-correcting codes
  - Local retransmission

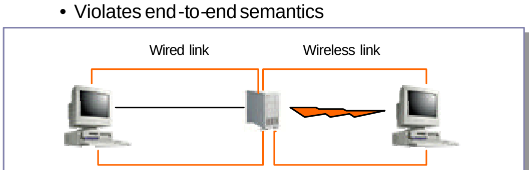
Approach Styles (End-to-End)

- Improve TCP implementations
  - Not incrementally deployable
  - Improve loss recovery (SACK, NewReno)
  - Help it identify congestion (ELN, ECN)
    - ACKs include flag indicating wireless loss
  - Trick TCP into doing right thing → E.g. send extra dupacks
  - What is SMART?
    - DUPACK includes sequence of data packet that triggered it

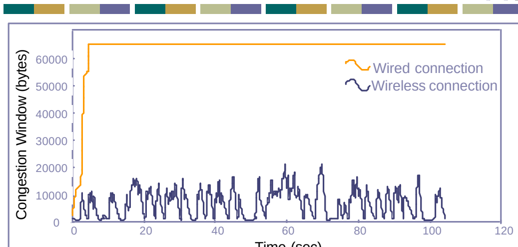


Approach Styles (Split Connection)

- Split connections
  - Wireless connection need not be TCP
  - Hard state at base station
    - Complicates mobility
    - Vulnerable to failures
    - Violates end-to-end semantics



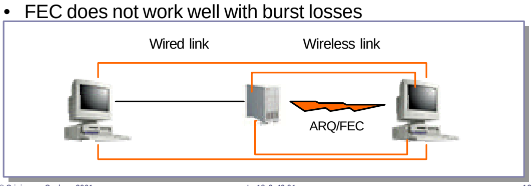
Split-Connection Congestion Window



- Wired connection does not shrink congestion window
- But wireless connection times out often, causing sender to stall

Approach Styles (Link Layer)

- More aggressive local rexmit than TCP
  - Bandwidth not wasted on wired links
- Adverse interactions with transport layer
  - Timer interactions
  - Interactions with fast retransmissions
  - Large end-to-end round-trip time variation
- FEC does not work well with burst losses



Hybrid Approach: Snoop Protocol

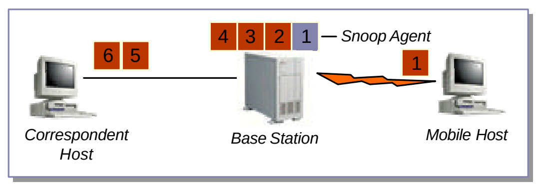
- Shield TCP sender from wireless vagaries
  - Eliminate adverse interactions between protocol layers
  - Congestion control only when congestion occurs
- The End-to-End Argument [SRC84]
  - Preserve TCP/IP service model: end-to-end semantics
  - *Is connection splitting fundamentally important?*
- Eliminate non-TCP protocol messages
  - *Is link-layer messaging fundamentally important?*

Fixed to mobile: transport-aware link protocol
Mobile to fixed: link-aware transport protocol

Snoop Overview

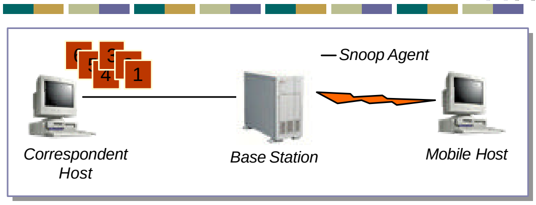
- Modify base station
  - to cache un-acked TCP packets
  - … and perform local retransmissions
- Key ideas
  - No transport level code in base station
  - When node moves to different base station, state eventually recreated there

Snoop Protocol: CH to MH



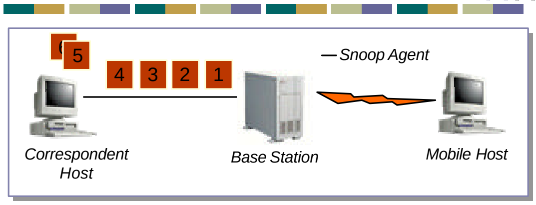
- Snoop agent: *active interposition agent*
  - Snoops on TCP segments and ACKs
  - Detects losses by duplicate ACKs and timers
  - Suppresses duplicate ACKs from FH sender

Snoop Protocol: CH to MH



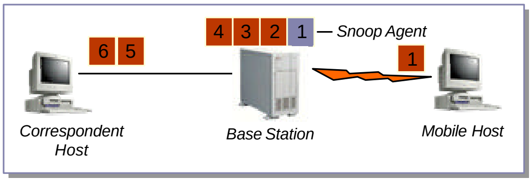
- Transfer of file from CH to MH
- Current window = 6 packets

Snoop Protocol: CH to MH



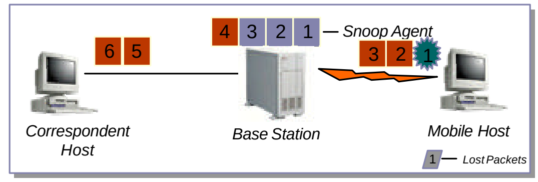
- Transfer begins

Snoop Protocol: CH to MH



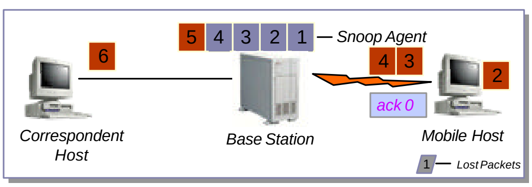
- Snoop agent caches segments that pass by

Snoop Protocol: CH to MH



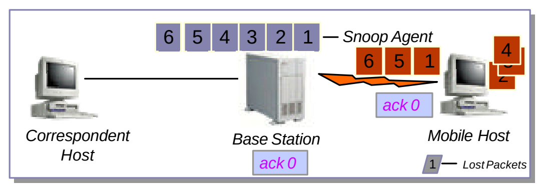
- Packet 1 is Lost

Snoop Protocol: CH to MH



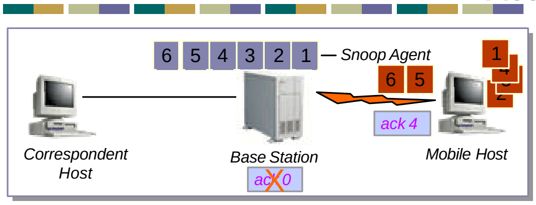
- Packet 1 is Lost
  - Duplicate ACKs generated

Snoop Protocol: CH to MH



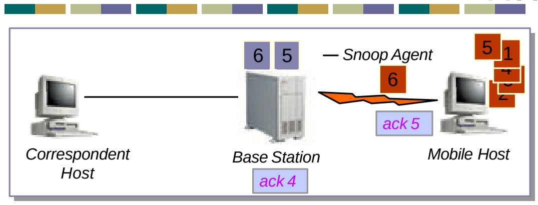
- Packet 1 is Lost
  - Duplicate ACKs generated
- Packet 1 retransmitted from cache at higher priority

Snoop Protocol: CH to MH



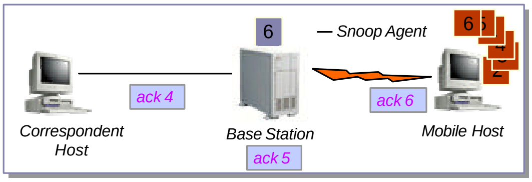
- Duplicate ACKs suppressed

Snoop Protocol: CH to MH



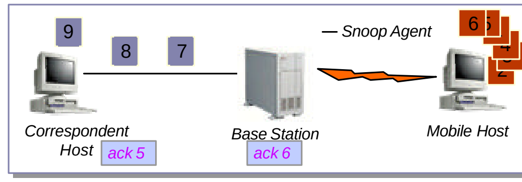
- Clean cache on new ACK

Snoop Protocol: CH to MH



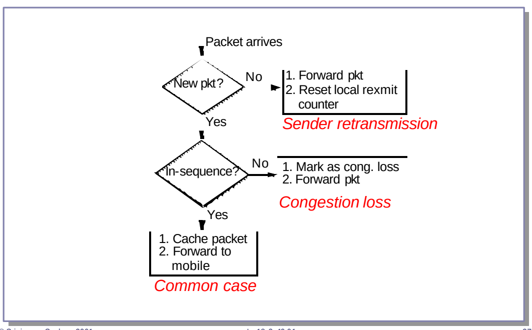
- Clean cache on new ACK

Snoop Protocol: CH to MH

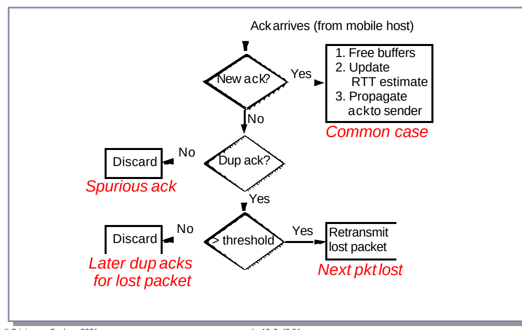


- Active soft state agent at base station
- Transport-aware reliable link protocol
- Preserves end-to-end semantics

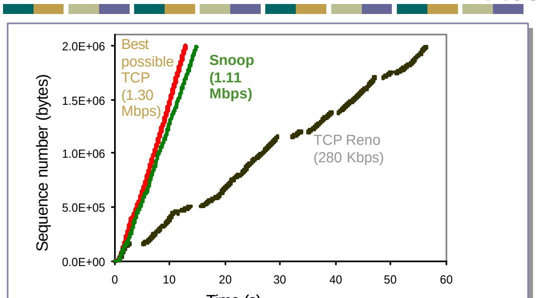
Snoop Data Processing



Snoop ACK Processing

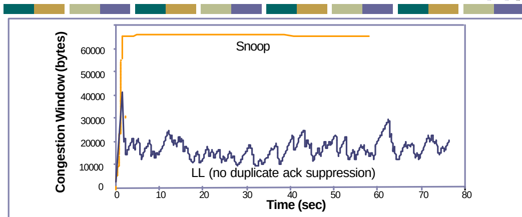


Snoop Performance Improvement



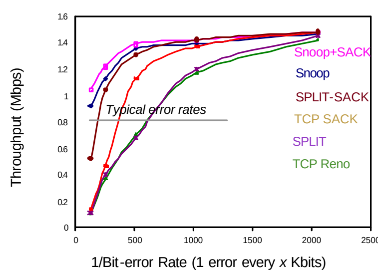
2 MB wide-area TCP transfer over 2 Mbps Lucent WaveLAN

Benefits of TCP-Awareness



- 30-35% improvement for Snoop: LL congestion window is small (but no coarse timeouts occur)
- Connection bandwidth-delay product = 25 KB
- Suppressing duplicate acknowledgments and TCP -awareness leads to better utilization of link bandwidth and performance

Performance: FH to MH



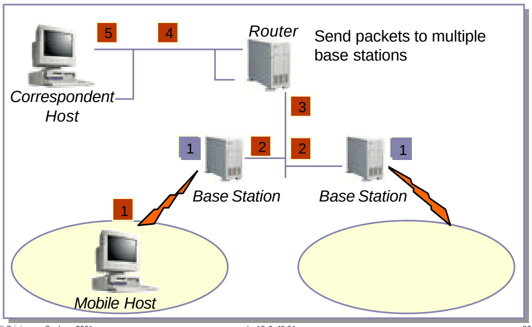
- Snoop+SACK and Snoop perform best
- Connection splitting *not* essential
- TCP SACK performance disappointing

2 MB local-area TCP transfer over 2 Mbps Lucent WaveLAN

Other Issues

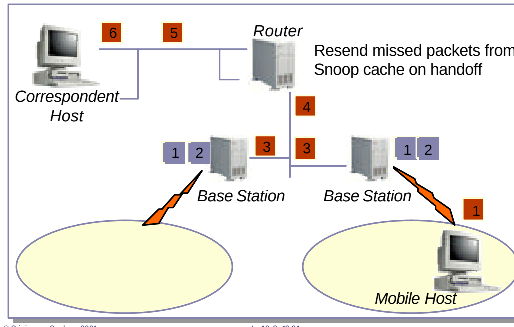
- What about mobility?
- What about mobile-to-fixed communication?

Handling Mobility



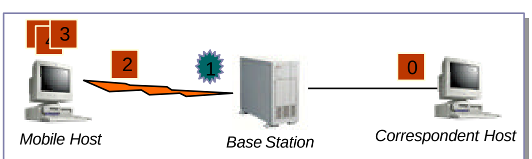
Send packets to multiple base stations

Handling Mobility



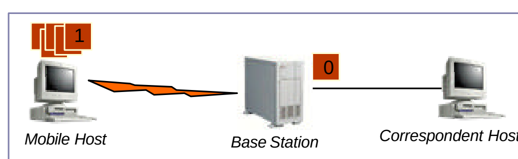
Resend missed packets from Snoop cache on handoff

Snoop Protocol: MH to CH



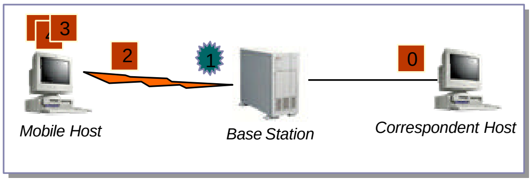
- Caching and retransmission will not work
  - Losses occur before packet reaches BS
  - Congestion losses should not be hidden
- Solution: *Explicit Loss Notifications (ELN)*
  - In-band message to TCP sender

Snoop Protocol: MH to CH



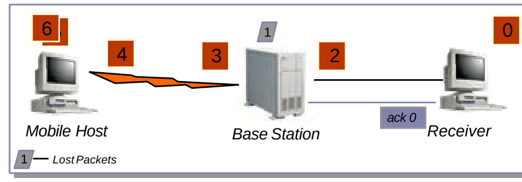
- MH begins transfer to CH

Snoop Protocol: MH to CH



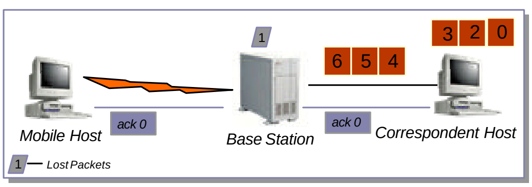
- Packet 1 lost on wireless link

Snoop Protocol: MH to CH



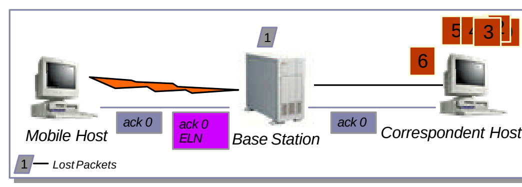
- Add 1 to list of holes after checking for congestion

Snoop Protocol: MH to CH



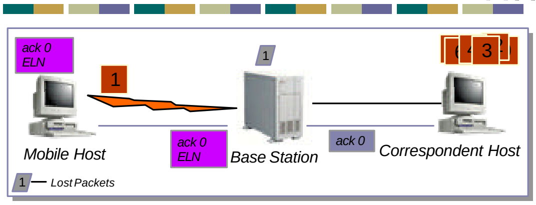
- Duplicate ACKs sent

Snoop Protocol: MH to CH



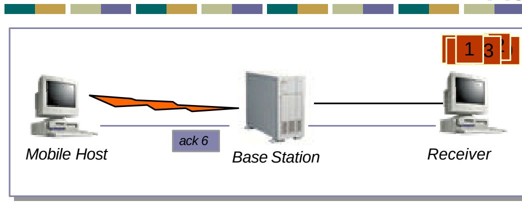
- ELN information added to duplicate ACKs

Snoop Protocol: MH to CH



- ELN information on duplicate ACKs
- Retransmit on Packet 1 on dup ACK + ELN
- No congestion control now

Snoop Protocol: MH to CH



- Clean holes on new ACK
- Link-aware transport decouples congestion control from loss recovery
- Technique generalizes nicely to wireless transit links

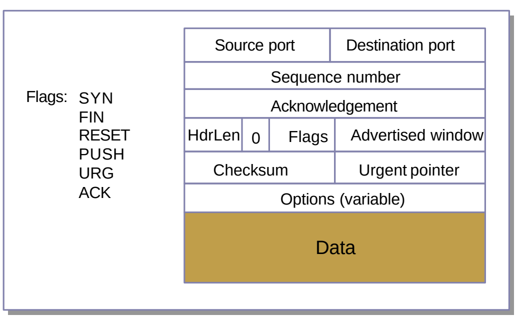
Overview

- Header Compression

Low Bandwidth Links

- Efficiency for interactive
  - 40byte headers vs payload size – 1 byte payload for telnet
- Header compression
  - What fields change between packets?
  - 3 types – fixed, random, differential

TCP Header



Header Compression

- What happens if packets are lost or corrupted?
  - Packets created with incorrect fields
  - Checksum makes it possible to identify
  - How is this state recovered from?
- TCP retransmissions are sent with complete headers
  - Large performance penalty – must take a timeout, no data-driven loss recovery
  - How do you handle other protocols?

Non-reliable Protocols

- IPv6 and other protocols are adding large headers
  - However, these protocols don't have loss recovery
  - How to recovery compression state
- Decaying refresh of compression state
  - Suppose compression state is installed by packet X
  - Send full state with X+2, X+4, X+8 until next state
  - Prevents large number of packets being corrupted
- Heuristics to correct packet
  - Apply differencing fields multiple times
- Do we need to define new formats for each protocol?
  - Not really – can define packet description language [mobicom99]

Next Lecture: Queue Management

- RED
- Blue
- Assigned reading
  - [FJ93] Random Early Detection Gateways for Congestion Avoidance
  - [Fen99] Blue: A New Class of Active Queue Management Algorithms