DCCP Open Issues

Eddie Kohler
UCLA/ICIR
IETF 58 DCCP Meeting
November 10, 2003
Overview

- # NDP
- Identification and Challenge
- Data Dropped requirements in CCID 3
- Packet sizes
- Payload Checksum
- Service Code
- VoIP issues
# NDP goal

- Make DCCP sequence numbers useful for the application
- Problem: DCCP sequence numbers advance on non-data packets, such as acks
  - This is a good thing
    - Can detect ack loss, simplifies feature negotiation and ack state cleanup
- App doesn't care if an NDP gets lost
• **Solution:** Include a count of the number of non-data packets sent so far on every packet
  
  \[ \text{App seqno} = \text{DCCP seqno} - \# \text{NDP} \]

• **Problem:** No space for a precise count

• So use 4 bits, now reduced to 3
# N Dumb P

- No expansion space in the header
- Losses of $\geq 8$ packets in a row are ambiguous
- Does anyone care about # NDP anyway?
# NDP recommendation

- Remove # NDP from the header
- Either specify NDP options
  - Use NDP feature
    - NDP Count option included on every NDP, and the first DP after a string of one or more NDPs
- … or just punt totally
  - Apps must include their own sequence numbers if they want to detect data loss
  - RTP already does
Identification and Challenge

- Four components: Identification, Challenge, ID Regime, Connection Nonce
- Mechanism for confirming that a packet is part of the connection
  MD5 hash of some packet contents and Connection Nonces (shared secrets between endpoints)
- Used in resynchronization and mobility
I-Dumb-tification

- Not particularly secure
  - Connection Nonces usually exchanged in the clear at connection initiation
  - False sense of security [ekr]
- Resync doesn’t need it
  - DCCP-Sync mechanism much better
- Mobility may not need it
  - Mobility ID, used to avoid NAT issues, serves the same function
Identification recommendation

• Remove Identification, Challenge, ID Regime, and Connection Nonce from main draft

• Perhaps move them to another draft

  “Sequence number security is depressing”, and some variant on this mechanism might help
• Data Dropped distinguishes network losses from endpoint losses
  “I dropped this packet because my receive buffer is full”

• Some Data Dropped states demand that the sender slow down
  “Every packet newly acknowledged as Drop Code 2 SHOULD reduce the sender’s instantaneous rate by one packet per round trip time”
  See also Slow Receiver

• Problem: How to do this in CCID 3/TFRC?
  Sending rate pops out of an equation
  Not a modifiable parameter like cwnd
Data Dropped recommendation

- Each Data Dropped/Slow Receiver recommends a decrease in rate of $\Delta R$
- Remember the total $\Delta R$ for each loss interval
- Combine the $\Delta R$s for the last 8 loss intervals using TFRC’s loss interval weights
- Subtract that from the equation’s suggested rate

- Alternatively, might be able to work out something with adding a fake loss interval
Packet sizes

• DCCP congestion control mechanisms are specified in terms of packets, not bytes
  - CCID 2: cwnd is measured in packets
  - CCID 3: rate is measured in packets per second
• But application determines how long packets are
• Potential attacks
  - Send small packets, build up large window, suddenly switch to huge packets
Packet sizes recommendation

- Currently limit maximum packet size in both CCIDs
  - 1500 bytes
- But attacks not that worrisome
  - Don’t seem to get more bandwidth in the long run
- Recommend removing limit
  - But describe the problem
  - Add text: implementations MAY check for and prevent packet size gaming
Payload Checksum

• Option contains an Internet checksum for the payload
  Intended for use with low Checksum Coverage (partial checksums)

• Goal: Links don’t drop corrupt packets (because of low Checksum Coverage); endpoint detects whether data is corrupt (Payload Checksum)

• Problem: Internet checksum is weak
  Conventional wisdom: most errors detected by link CRCs
  But low Checksum Coverage might cause links to weaken CRCs
Payload Checksum recommendation

- Keep option, weaken text
  
  “Applications MUST NOT depend only on Payload Checksum. . .”

- Alternatives
  
  Remove option
  
  32-bit CRC
Service Code

- DCCP-Request includes a Service Code
  Names the service the client is contacting
  Examples: “HTTP”, “RTSP”

- Does this open security holes? [Bellovin]
  A firewall allows a connection based on Service Code, but the server inside the firewall ignores the Service Code?
Service Code recommendation

• Drop wildcarding

  The Request’s Service Code MUST match the server’s Service Code

  Add a Service Code to the Response
VoIP

- Complexity → CCID 3-Thin
- Slow start → initial rate of 4 pps
- Rate slows down during idle periods
- Rate does not increase during app-limited period
- Variable rate considered harmful
  
  Apps might have discrete rates
- Rate changes considered harmful
  
  Apps work at fixed rates, hard to switch
VoIP recommendations

- Rate slows down during idle periods
  - Rate does not increase during app-limited period
  - Slow start
    You don’t get to reserve bandwidth
    Investigate costs and benefits of quick increases after idle periods in another draft

- Variable rate considered harmful
  
  Could probably allow sending at faster rate than CC suggests, explore in another draft

- Rate changes considered harmful
  
  Application dependent; can be addressed in application behavior?