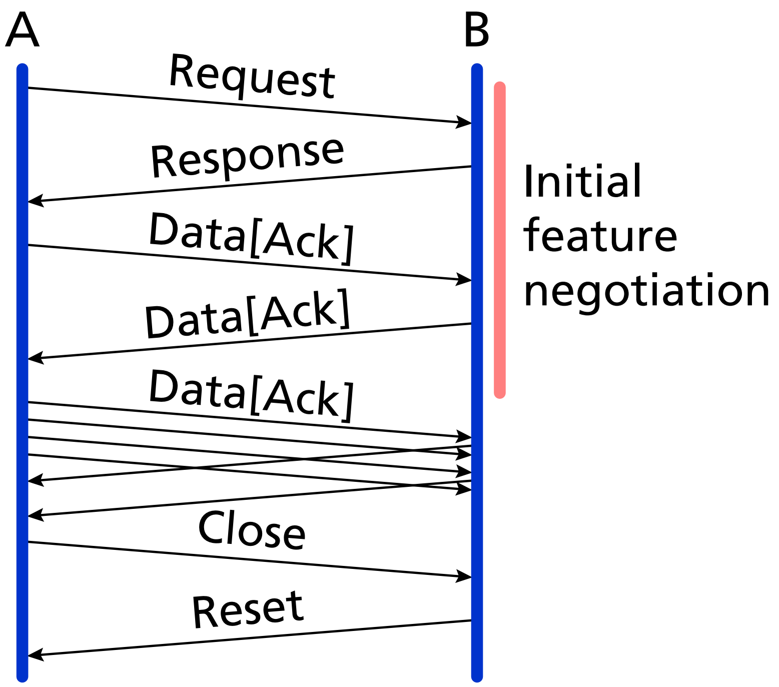
Datagram Congestion Control Protocol Overview

—

Eddie Kohler, Mark Handley, Sally Floyd, Jitendra Padhye
kohler@aciri.org
http://www.aciri.org/kohler/dcp/

Connection overview and packet types

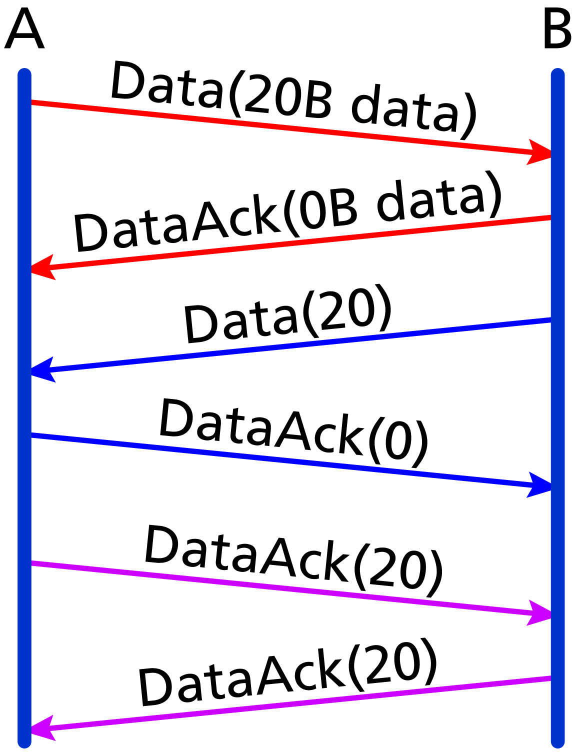


- Two ends, (potentially) two-way data transfer
- Explicit connection setup and teardown

  Request, Response, CloseReq, Close, Reset
- Data transfer and most feature negotiations use Data and DataAck

  Data = data only; DataAck = data + acknowledgement
- Other types: Move

Connection breakdown



- Two *half-connections*

  Data from A to B plus acks from B to A

  Data from B to A plus acks from A to B

  Ack piggybacking: a single packet relevant to both half-connections

DCCP core idea

- Let each half-connection choose a congestion control mechanism

  Possibly different for the two half-connections

  CC is negotiated: both endpoints must agree

  CC determines form of acknowledgements, . . .

- Congestion control mechanisms represented by CCIDs

  CCID 0: Single-Window (not much data to send)

  CCID 1: unspecified sender-based (expect Ack Vector)

  CCID 2: TCP-like

  CCID 3: TFRC (rate-based)

Packet properties

- Every packet has a sequence number

  Includes pure acknowledgements

  Increments every time

- Acknowledgement information

  DataAck acks *largest* received seqno

  Ack Vector option: which packets were received? ECN marked?

  TFRC options: loss event rate, . . .

Reliably negotiated features

- Half-connection properties agreed upon by both ends

  CCID, ECN capability, mobility capable, . . .

- Three options negotiate feature values

  Ask (A → B): “I want to use this value”

  Choose (B → A): “No, try this value instead”

  Answer (B → A): “Value OK”

- Example

  A: Ask CCID(4); B: Choose CCID(3);

  A: Ask CCID(3); B: Answer CCID(3)

Mobility

- Negotiate capability beforehand

  Produces a nonce (shared secret)

- Send a Move packet after you’ve moved

  Includes nonce

- Patent status unclear (!?)

Example half-connection (1)

- A $\longrightarrow$ B 0: Request, Ask(CCID 2), Choose(CCID 0)
- B $\longrightarrow$ A 100: Response[0], Answer(CCID 2), Ask(CCID 0)
- A $\longrightarrow$ B 1: DataAck[100], Answer(CCID 0), [app request]
- B $\longrightarrow$ A 101: Data, [media data]
- B $\longrightarrow$ A 102: Data, [media data]
- A $\longrightarrow$ B 2: DataAck[102], Ack Vector($\surd_{102}\surd_{101}$)

Example half-connection (2)

- B → A 103: Data, [media data]
- B → A 104: Data, [media data] *** LOST ***
- B → A 105: Data, [media data]
- B → A 106: DataAck[2], [media data]
- A → B 3: DataAck[103], Ack Vector($\surd_{103}\surd_{102}\surd_{101}$)
- A → B 4: DataAck[106], Ack Vector($\surd_{106}\surd_{105}\mathrm{X}_{104}\surd_{103}$)

Example half-connection (3)

- B → A 107: DataAck[4], [media data]
- B → A 108: Data, [media data]
- A → B 5: DataAck[108], Ack Vector($\surd_{108}\surd_{107}$)
- . . .
- B → A 200: CloseReq[80]
- A → B 81: Close[200]
- B → A 201: Reset[81]