

DCCP Written Reviewer Comments



Eddie Kohler
International Computer Science Institute
IETF 57 DCCP Meeting
July 16, 2003

Overall note



- I personally, and the authors in general, cannot thank the reviewers enough

Non-outstanding comments: Minshall



- Half-open connections
 - Tough problem, fixed
- Middlebox Considerations section too normative (also Rescorla)
 - Toned down language
- Ack Ratio dangerous
 - Made it explicitly more TCPLike (delayed ack timer)
- Many extremely useful comments
 - Fixed

Non-outstanding comments: Rescorla



- Requirements too weak (SHOULDs instead of MUSTs)
 - Will add occasional text describing what will happen if you don't do a SHOULD
- Presentation issues (documents hard to track)
 - Fixes in progress
- Again, many extremely useful comments

Non-outstanding comments: Westerlund



- RTP sequence number interpolation a bad idea

Removed

- Again, many extremely useful comments

Do we need multiple CCIDs? (Rescorla)



? Non-real-time traffic doesn't care (so just pick one); real-time traffic doesn't want congestion control, it wants congestion *indications*

- Authors believe multiple CCIDs essential

Added use cases to drafts

Will CCID 1 be useful? (Minshall)



- ? Seems hard to provide the feedback required by a protocol you don't understand. (Minshall)
- Ack Vector has stable semantics regardless of CCID
 - All required congestion information is in Ack Vector

PAWS (Minshall)



? Sequence number space is too small

- Tough problem

Security issue as well as classical PAWS old-segment issue

Non-recommended solutions to PAWS



0. Declare nonissue: DCCP only suitable for connections with congestion windows up to a couple hundred thousand packets

Even those connections will have 2MSL problems

1. TCP-like Timestamp option

Verbose and does not solve security problem

2. Globally extend sequence numbers to 48 bits

Header becomes pointlessly verbose for vast majority of connections

Recommended solutions to PAWS



3. Add an alternate header format with long sequence numbers

Want to differentiate formats with a header bit (middleboxes)—take a bit from Cslen? # NDP?

Sequence numbers embedded in options?—Recommend that all options use deltas off the Sequence Number or Acknowledgement Number

Cost: all sequence number calculations use the extended sequence number

4. Add options that extend the sequence numbers

5. Add an epoch option

Sequence epoch and acknowledgement epoch, 24 bits each

Reject packet if $|\text{cur-seq-epoch} - \text{last-seq-epoch}| > 1$ (and similarly for acknowledgement epoch)

Partial checksums (Minshall, Rescorla)



? How useful are partial checksums actually going to be? Link layers will not deliver bad data.

- Positive experiences with delivery of corrupt packets on some link layers

- A lot like UDP-Lite

Only even more compelling because of congestion control

Wouldn't make sense to allow one but not the other

Will be discussed at IAB plenary

When is a packet received? (Minshall)

- ? It is a mistake to define packet receipt as “options processed”. Should define it as “will make best effort to give data to application”. Congestion in the endpoint is still congestion.
- Authors all agree that it is essential to decouple non-network loss from network loss
 - Cost of leaving them coupled is too high for high-volume flows
 - Example: receive buffer drop on a 1 Gbps, 100ms RTT flow leads to 5 minutes of bad performance
 - “Congestion control serves several separate functions: preventing congestion collapse; fairness with competing traffic; and optimizing performance for the individual flow.” [RFC 2914]
 - A receive buffer drop is not causing congestion collapse and has no fairness implications; halving the congestion window pessimizes performance for the individual flow. This is not congestion

When is a packet received? (2)



- What does “will make best effort to give data to application” mean in the presence of unreliability?
 - Applications will often want drop-from-head receive buffers to drop untimely data
- Passing acknowledgements to the application
 - Avoid duplicating acknowledgement information at app level
 - App wants to know if something was really delivered, “best effort” insufficient
- This is a tradeoff: complexity vs. performance
 - Complexity: report different kinds of drops
 - Performance: avoid congestion response to non-congestion loss
 - We believe the tradeoff is clear

The Data Dropped option (Minshall)



- ? However you define packet receipt, the Data Dropped option is a bad way to signal receive buffer loss and/or that the app is no longer listening.
- Seems clean to us
 - Ack Vector: Which packets were delivered to DCCP?
 - Data Dropped: Which packets were delivered to the application?
- Alternatives: Receive window?
 - Not clear how to define for an unreliable protocol
 - Not useful for applications
- Report a count of receive buffer drops?
 - ECN Nonce verification, bad match with apps

Data Dropped possible alternative



- Reduce Data Dropped to two Drop Codes, “delivered” or “not delivered”

Currently 6 code points:

delivered

dropped: protocol constraints

dropped: receive buffer loss

delivered but corrupt

dropped: app not listening

dropped: corrupt

- Not clear how much complexity you lose

Still need different responses to “app not listening”, “corrupt”;
would have to add separate options

- Author consensus: keep Data Dropped

Mobility (Rescorla)



- ? Why include it? Belongs at the IP layer.
- Depends on how important you think it will be for devices to change providers (cell phone to WiFi?)
 - Mobile IP might not be there
 - DCCP might be able to do better than Mobile IP
- Not clear agreement

Mobility (Westerlund)



- ? What if the moving endpoint was behind a NAT? (Old Address, Old Port will be wrong.)
- Possible solutions
 - Mobility ID (extend Mobility Capable to an ID?) instead of Old Address and Old Port
 - Don't move from behind a NAT
 - Drop mobility
- Again, authors +0.1 on mobility

Underspecification (all)



? Is it safe to silently track changes in other specs?

From CCID 2: “Conformant CCID 2 implementations MAY track TCP’s evolution directly, as updates are standardized in the IETF, rather than waiting for revisions of this document.”

- Probably need to specify exactly what changes may be tracked

? Timeout processes are not specified, or are specified by reference to TCP. The feature negotiation diagrams are ambiguous. CCID 3 Loss Event Rate calculation specified by reference. . . .

- Need at least to add more language

Don't want to overspecify timeouts in particular

Security (Rescorla)



- ? Sequence number security is depressing. Replace checksum with a MAC keyed by Connection Nonce?
- Key question: transport security vs. app or network-level security
- Tradeoff: incremental security vs. middlebox friendliness
 - Not clear how much security gained by assuming attacker can see packets *now*, but couldn't at connection startup

Optional ECN? (Minshall)



- ? Why make ECN optional? Middlebox traversal not a good argument.
- An endpoint that doesn't want to verify ECN Nonces should turn off ECN capability
- Authors don't feel too strongly about this issue

Small fields (all)




- ? Reset Reasons and Service Names are too small, strings should be a possibility. (Rescorla)
- Sense of the room?
- ? Want an extension mechanism for things like options and Reset Reasons, with a way of knowing what can be ignored if not understood. (Minshall)
- Case-by-case?
 - For example, Ignored sufficient for options?
- ? How are Service Names allocated? (Westerlund, Rescorla)
- Current text more explicit, refer to recent email

Complexity



- Most complexity derives from in-band signalling
 - Connection setup and teardown
 - Feature negotiation
 - Reliable transmission of acks
- Alternative: Assume a signalling channel
 - Unfriendly to middleboxes
 - Unfriendly to applications
 - Just moves complexity, doesn't remove it (?)
 - But the data-carrying protocol would look much simpler

Authors' feature prioritization



TCP-like CCID	E	E	E	E	Essential
TFRC CCID	E	E	E	H	High priority
ECN nonce	H	H	E	M	Medium priority
current Ack Vector format	H	H	M	L	Low priority
optional Ack Vector	H	L	H	-	Don't care
current Data Dropped	H	M	M		
DataAck piggybacking	H	M	M		
different CCIDs on HCs	M	H	L		
partial checksums	M	L	M		
ack congestion control	M	L	M		
quiescence	M	L	M		
CCval	M	M	L		
# NDP	-	H	-		
optional ECN	L	-	L		
mobility	-	L	-		