Internet Engineering Task Force
INTERNET-DRAFT
draft-kohler-dcp-ccid0-00.ps

Eddie Kohler
Sally Floyd
ACIRI

13 July 2001
Expires: January 2002

Profile for DCP Congestion Control ID 0: Single-Window Congestion Control

Status of this Document

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of [RFC 2026]. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This document contains the profile for Congestion Control Identifier 0, Single-Window Congestion Control, in the Datagram Control Protocol (DCP) [DCP]. DCP implements a congestion-controlled, unreliable flow of datagrams suitable for use by applications such as streaming media. The Single-Window Congestion Control CCID is used by senders that promise to send no data whatsoever, except possibly for a single initial window of data sent at the beginning of the connection.

Table of Contents

1. Introduction 3
1.1. Usage Scenario 3
1.2. Example Half-Connection 3
2. Connection Establishment 3
3. Congestion Control on Data Packets 3
4. Acknowledgements 4
4.1. Congestion Control on Acknowledgements 4
4.2. Quiescence 4
4.3. Acknowledgements of Acknowledgements 4
5. Explicit Congestion Notification 4
6. Relevant Options and Features 4
7. Application Requirements 4
8. Thanks 5
9. References 5
10. Authors' Addresses 5

1. Introduction

This document contains the profile for Congestion Control Identifier 0, Single-Window Congestion Control, in the Datagram Control Protocol (DCP).

DCP uses Congestion Control Identifiers, or CCIDs, to specify the congestion control mechanism in use on a half-connection. (A half-connection might consist of data packets sent from DCP A to DCP B, plus acknowledgements sent from DCP B to DCP A. DCP A is the HC-Sender, and DCP B the HC-Receiver, for this half-connection. In this document, we abbreviate HC-Sender and HC-Receiver as "sender" and "receiver", respectively.)

The Single-Window Congestion Control CCID implies that the sender will send no data packets, except for at most an initial window's worth at the beginning of the connection. (This initial window is calculated as for TCP; currently, it is 2 packets.) It is suitable for senders who have almost no data to send -- for example, for clients in a client-server streaming media connection, where the clients might make an application request to start off the connection, but then keep quiet forever.

We note that this draft is rough and incomplete, and needs considerably more attention. In particular, we have not yet decided how to deal with losses within the initial window of packets.

1.1. Usage Scenario

Single-Window Congestion Control was designed for the potentially common case of a client connecting to a data stream not requiring any application feedback -- for example, a streaming media connection. (Current popular streaming-media protocols include application feedback to report congestion. That's unnecessary in DCP, which reports congestion events itself.) Using CCID 0 for the client's half-connection explicitly informs the server that the client will never have data to send. This can simplify the server's implementation: for example, the server need not keep track of detailed acknowledgement information for the client's packets. Some high-use services might choose to force their clients to use CCID 0, since then they would not have to deal with any client data.

Note, however, that DCP was designed so that a quiescent half-connection causes absolutely no overhead. Any quiescent CCID behaves the same as CCID 0. The use of CCID 0 is entirely optional, and has almost no performance effect in terms of numbers of packets sent, or packet sizes sent, compared to sending the same (small) number of packets with a different CCID.

Compensating for losses within the initial window of data is a question for further research.

1.2. Example Half-Connection

TBA.

2. Connection Establishment

No special options or features are strictly necessary to set up a half-connection using CCID 0. Since half-connections begin in CCID 0, it may not even be necessary to negotiate the CCID. However, if the sender plans to send any data in its allowed initial window, the sender SHOULD negotiate the Use Ack Vector feature.

3. Congestion Control on Data Packets

Since CCID 0 allows the sender to send at most one initial window's worth of data, there is no need for any congestion control mechanism for data packets. The initial window is

defined by TCP; currently, its value is 2 packets. TCP senders may send an initial window's worth of data before receiving any congestion feedback. Therefore, CCID 0's behavior here is no worse than TCP.

In a A-to-B half-connection using CCID 0, DCP A MUST drop every packet the application tries to send beyond the initial window. Furthermore, DCP B SHOULD reset the connection if DCP A sends more than an initial window of data packets.

We have not yet determined how to handle loss events in CCID 0's allowed initial window of data. One solution might be to implement reliable transmission of this window in the kernel, using a simple exponential backoff.

4. Acknowledgements

Any half-connection using CCID 0 is quiescent for most of its lifetime. During this period, no acknowledgements need be sent. During the initial window, DCP B SHOULD send acknowledgements to DCP A using Ack Vector, if Use Ack Vector was negotiated.

4.1. Congestion Control on Acknowledgements

Since there are at most an initial window's worth of acknowledgements, there is no need for any congestion control on acknowledgements.

4.2. Quiescence

This section refers to quiescence in the DCP sense (see section 6.1 of [DCP]): How does a CCID 0 receiver determine that the corresponding sender is not sending any data?

The receiver detects that the sender has gone quiescent after receiving three consecutive DCP-Ack packets from the sender (not counting initial feature negotiation). If the other half-connection has any data to send whatsoever, this should happen almost immediately. CCID 0 half-connections stay quiescent permanently: after going quiescent, they never send data again.

4.3. Acknowledgements of Acknowledgements

There is no need for the sender to acknowledge the receiver's acknowledgements.

5. Explicit Congestion Notification

Senders using CCID 0 perform no worse than TCP, despite their lack of active congestion control, due to the extremely limited amount of data they send. All DCP-Data and DCP-Ack packets for CCID 0 SHOULD set ECN-Capable Transport on their packets, regardless of the value of the ECN Capable feature. There should be at most 2*(initial window) such packets.

There is no need for the receiver to echo the ECN Nonce.

6. Relevant Options and Features

CCID 0 optionally uses the Ack Vector option and the Use Ack Vector feature.

7. Application Requirements

Obviously, an application using CCID 0 cannot send more than an initial window's worth of data.

8. Thanks

We thank Mark Handley and Jitendra Padhye for their help in defining CCID 0.

9. References

[DCP] Eddie Kohler, Mark Handley, Sally Floyd, and Jitendra Padhye. Datagram Control Protocol (DCP). Work in progress.

[RFC 2026] S. Bradner. The Internet Standards Process -- Revision 3. RFC 2026.

10. Authors' Addresses

Eddie Kohler <kohler@aciri.org>
Sally Floyd <floyd@aciri.org>

AT&T Center for Internet Research at ICSI (ACIRI),
ICSI,
1947 Center Street, Suite 600
Berkeley, CA 94704.