Mboned
Internet-Draft
Intended status: Standards Track
Expires: September 11, 2020

J. Holland
Akamai Technologies, Inc.

March 10, 2020

Circuit Breaker Assisted Congestion Control
draft-ietf-mboned-cbacc-00

Abstract

This document specifies Circuit Breaker Assisted Congestion Control (CBACC). CBACC enables fast-trip Circuit Breakers by publishing rate metadata about multicast channels from senders to intermediate network nodes or receivers. The circuit breaker behavior is defined as a supplement to receiver driven congestion control systems, to preserve network health if receivers subscribe to a volume of traffic that exceeds capacity policies or capability for a network or receiver.

Status of This Memo

This Internet-Draft will expire on September 11, 2020.

Copyright Notice

Table of Contents

1. Introduction

This document defines Circuit Breaker Assisted Congestion Control (CBACC). CBACC defines a Network Transport Circuit Breaker (CB), as described by [RFC8084].

The CB behavior defined in this document uses bit-rate metadata about multicast data streams, coupled with policy, capacity, and load information at a network node, to prune multicast channels so that

the node's aggregate capacity is not exceeded by the subscribed channels.

To communicate the required metadata, this document defines a YANG [RFC7950] module that augments the DORMS [I-D.draft-jholland-mboned-dorms-02] YANG module. DORMS provides a mechanism for senders to publish metadata about the multicast streams they're sending through a RESTCONF service, so that receivers or forwarding nodes can discover and consume the metadata with a set of standard methods. The metadata MAY be communicated to receivers or forwarding nodes by some other method, but the definition of any alternative methods is out of scope for this document.

The CB behavior defined in this document matches the description provided in Section 3.2.3 of [RFC8084] of a unidirectional CB over a controlled path. The control messages from that description are composed of the messages containing the metadata required for operation of the CB.

CBACC is designed to supplement protocols that use multicast IP and rely on well-behaved receivers to achieve congestion control. Examples of congestion control systems fitting this description include [PLM], [RLM], [RLC], [FLID-DL], [SMCC], and WEBRC [RFC3738].

CBACC addresses a problem with "overjoining" by untrusted receivers.

In an overjoining condition, receivers (either malicious, misconfigured, or with implementation errors) subscribe to multicast channels but do not respond appropriately to congestion. When sufficient multicast traffic is available for subscription by such receivers, this can overload any network.

The overjoining problem is relevant to misbehaving receivers for both receiver-driven and feedback-driven congestion control strategies, as described in Section 4.1 of [RFC8085].

Overjoining attacks and the challenges they present are discussed in more detail in Appendix A.

CBACC offers a solution for the recommendation in Section 4 of [RFC8085] that circuit breaker solutions be used even where congestion control is optional.

1.1. Background and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP

14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Circuit Breaker Behavior

2.1. Functional Components

This section maps the functional components described in Section 3.1 of [RFC8084] to the operational components of the CBACC CB defined by this document.

2.1.1. Ingress Meter

The metadata provides an ingress meter in the form of an advertised maximum data bit-rate, namely the "max-bits-per-second" field in the YANG model in Section 3. This is a self-report by the sender about the maximum amount of traffic a sender will send within the interval given by the "data-rate-window" field, which is the measurement interval.

The sender MUST NOT send more data for a data stream than the amount of data implied by its advertised data rate within any measurement window, and it's RECOMMENDED for the sender to provide some margin to account for forwarding bursts. If an egress node observes a higher data rate within any measurement window, it MAY circuit-break that flow immediately.

2.1.2. Egress Meter

The node implementing the CB behavior has access to several pieces of information that can be used as relevant egress metrics:

1. Physical capacity limits on each interface.

2. Configured capacity limits for multicast traffic for each interface, if any.

3. The observed received data rates of subscribed multicast channels with CBACC metadata.

4. The observed received data rates of subscribed multicast channels without CBACC metadata.

5. The observed received data rates of competing non-multicast traffic.

6. The loss rate for subscribed multicast channels, when available. The loss rate is only sometimes observable at an egress node; for

example, when using AMBI [I-D.draft-jholland-mboned-ambi-05], or when the data stream carries a protocol that is known to the egress node by some out of band means, and whose traffic can be monitored for loss. When available, the loss rates may be used.

Note that any on-path router can be considered an egress node for purposes of this CB, even though it may be forwarding traffic downstream, and even though other egress points may also be operating a downstream CB that covers the same data stream. Components in the receiving devices, such as an operating system or browser can also act as an egress node, as can a receiving application.

2.1.3. Communication Method

CBACC operates at an egress node, so the egress metrics in Section 2.1.2 are available through system calls, or by communication with various locally deployable system monitoring applications. Any suitable application that provides the necessary egress meter is appropriate.

The communication path defined in this document for the information from the ingress meter is the use of DORMS [I-D.draft-jholland-mboned-dorms-02]. Other methods MAY be used as well, but are out of scope for this document.

2.1.4. Measurement Function

The measurement function maintains a few values for each interface, computed from the egress and ingress meter values:

1. The aggregate advertised maximum bit-rate capacity consumed by CBACC data streams. This is the sum of the max-bit-rate values in the CBACC metadata for all data streams subscribed through an interface

2. An oversubscription threshold for each interface. The oversubscription threshold will be determined differently for CBs in different contexts. In some network devices, it might be as simple as an administratively configured absolute value or proportion of an interface's capacity. For other situations, like a CB operating in a context with loss visibility, it could be a dynamically changing value that grows when data streams are successfully subscribed and receiving data without loss, and shrinks as loss is observed across subscribed data streams. The oversubscription threshold calculation could also incorporate other information like out-of-band path capacity measurements with [PathChirp], if available.

This document covers some non-normative examples of valid oversubscription threshold functions in Section 2.3.1, but in general, the oversubscription threshold is the primary parameter that different CBs in different contexts can tune to provide the safety guarantees necessary for their context.

2.1.5. Trigger Function

The trigger function fires when the aggregate advertised maximum bit-rate exceeds the oversubscription threshold for any interface.

When oversubscribed, the trigger function changes the states of subscribed channels to "blocked" until the aggregate subscribed bit-rate is below the oversubscription threshold again.

2.1.5.1. Fairness and Inter-flow Ordering

The trigger function orders the monitored flows according to a fairness function, and blocks flows in order as needed to ensure that only a safe level of bandwidth can be consumed by subscribed flows. The fairness function can be different for CBs in different contexts.

Flows from a single sender MUST be ordered according to their priority field from the CBACC metadata when compared with each other. Between-sender flows and flows from the same sender with the same priority are ordered according to the fairness function. Where flows from the same sender have a priority order that conflicts with the ordering the fairness function would use, it's appropriate to treat those out of order flows from the sender as an aggregate flow for between-sender flow comparisons. (TBD: the aggregation algorithm probably needs more explaining and good examples.)

A CB implementation SHOULD provide mechanisms for administrative controls to configure explicit biases, as this may be necessary to support Service Level Agreements for specific events or providers, or to blacklist or de-prioritize channels with historically known misbehavior.

Subject to the above constraints, where possible the default fairness behavior SHOULD favor streams with many receivers over streams with few receivers, and streams with a low bit-rate over streams with a high bit-rate. For example, when receiver count is known, a good fairness metric is max-bandwidth divided by receiver-count. (Receiver count in some networks can be known through technologies such as the experimental PIM extension for population count described in [RFC6807], or other custom signaling methods.)

An overview of some other approaches to appropriate fairness metrics is given in Section 2.3 of [RFC5166].

2.1.6. Reaction

When the trigger function fires and a subscribed channel becomes blocked, the reaction depends on whether it's an upstream interface or a downstream interface.

If a channel is blocked on one downstream interface, it may still be unblocked on other downstream interfaces. When this is the case, traffic is simply not forwarded along blocked interfaces, even though clients might still be joined.

When a channel is blocked on all downstream interfaces, or when the upstream interface is oversubscribed, the channel is pruned so that data no longer arrives from the network on the upstream interface, by a PIM prune (Section 3.5 of [RFC7761]), or a "leave" operation with IGMP, MLD, or another multicast signaling mechanism.

Once initially circuit-broken, a flow SHOULD remain circuit-broken for no less than 3 minutes, even if space clears up, to ensure downstream subscriptions will notice and respond. (3 minutes is chosen to exceed the default maximum lifetime of 2 minutes that can occur if an IGMP responder suddenly stops operation, and ceases responding to IGMP queries with membership reports.)

When enough capacity is available for a circuit-broken stream to be unblocked and the circuit-breaker hold-down time is expired, the flows SHOULD be unblocked according to the priority order.

2.1.7. Feedback Control Mechanism

The metadata should be refreshed as needed to maintain up to date values. When using DORMS and RESTCONF, the HTTP Cache Control headers provide valid refresh time properties from the server, and SHOULD be used if present. If No-Cache is used, the default refresh timing SHOULD be 30 seconds plus a random value between 0 and 10 seconds.

2.2. States

2.2.1. Interface State

A CB holds the following state for each interface, for both the inbound and outbound directions on that interface:

- aggregate bandwidth: The sum of the bandwidths of all non-circuit-broken CBACC flows which transit this interface in this direction.

- bandwidth limit: The maximum aggregate CBACC advertised bandwidth allowed, not including circuit-broken flows.

  When reducing the bandwidth limit due to congestion, the circuit breaker SHOULD NOT reduce the limit by more than half its value in 10 seconds, and SHOULD use a smoothing function to reduce the limit gradually over time.

  It is RECOMMENDED that no more than half the capacity for a link be allocated to CBACC flows if the link might be shared with TCP or other traffic that is responsive to congestion.

2.2.2. Flow State

Data streams with CBACC metadata have a state for the upstream interface through which the stream is joined:

- 'subscribed' Indicates that the circuit breaker is subscribed upstream to the flow and forwarding packets through zero or more egress interfaces.

- 'pruned' Indicates that the flow has been circuit-broken. A request to unsubscribe from the flow has been sent upstream, e.g. a PIM prune (Section 3.5 of [RFC7761]) or a "leave" operation via IGMP, MLD, or another group membership management mechanism.

Data streams also have a per-interface state for downstream interfaces with subscribers, where the data is being forwarded. It's one of:

- 'forwarding' Indicates that the flow is a non-circuit-broken flow in steady state, forwarding packets downstream.

- 'blocked' Indicates that data packets for this flow are NOT forwarded downstream via this interface.

2.3. Implementation Design Considerations

2.3.1. Oversubscription Thresholds

TBD.

2.3.2. Fairness Functions

TBD.

3. YANG Module

3.1. Tree Diagram

```
   module: ietf-cbacc
     augment /dorms:metadata/dorms:sender/dorms:group/dorms:udp-stream:
       +--rw cbacc!
          +--rw max-bits-per-second    uint32
          +--rw max-mss?               uint16
          +--rw data-rate-window?      uint32
          +--rw priority?              uint16
```

3.2. Module

```
<CODE BEGINS> file ietf-cbacc@2020-03-10.yang
module ietf-cbacc {
    yang-version 1.1;

    namespace "urn:ietf:params:xml:ns:yang:ietf-cbacc";
    prefix "ambi";

    import ietf-dorms {
        prefix "dorms";
        reference "I-D.jholland-mboned-dorms";
    }

    organization "IETF";

    contact
        "Author:   Jake Holland
                   <mailto:jholland@akamai.com>
        ";

    description
    "Copyright (c) 2019 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject to
     the license terms contained in, the Simplified BSD License set
     forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (https://trustee.ietf.org/license-info).
```

```
     This version of this YANG module is part of
     draft-jholland-mboned-cbacc.  See the internet draft for full
     legal notices.

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
     NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
     'MAY', and 'OPTIONAL' in this document are to be interpreted as
     described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
     they appear in all capitals, as shown here.

     This module contains the definition for bandwidth consumption
     metadata for SSM channels, as an extension to DORMS
     (draft-jholland-mboned-dorms).";

    revision 2019-09-26 {
        description "Initial revision as an extension.";
        reference
          "";
    }

    augment
      "/dorms:metadata/dorms:sender/dorms:group/dorms:udp-stream" {
        description "Definition of the manifest stream providing
            integrity info for the data stream";

      container cbacc {
        presence "cbacc-enabled flow";
        description "Information to enable fast-trip circuit breakers";
        leaf max-bits-per-second {
            type uint32;
            mandatory true;
            description "Maximum bitrate for this stream, in Kilobits
                of IP packet data (including headers) of native
                multicast traffic per second";
        }
        leaf max-mss {
            type uint16;
            default 1400;
            description "Maximum payload size, in bytes";
        }
        leaf data-rate-window {
            type uint32;
            default 2000;
            description "Time window over which data rate is guaranteed,
                in milliseconds.";
            /* TBD: range limits? */
        }
        leaf priority {
```

```
            type uint16;
            default 256;
            description "The relative preference level for keeping this
                flow compared to other flows from this sender (higher
                value is more preferred to keep)";
        }
      }
    }
}

<CODE ENDS>
```

4. IANA Considerations

4.1. YANG Module Names Registry

This document adds one YANG module to the "YANG Module Names" registry maintained at <https://www.iana.org/assignments/yang-parameters>. The following registrations are made, per the format in Section 14 of [RFC6020]:

```
         name:      ietf-cbacc
         namespace: urn:ietf:params:xml:ns:yang:ietf-cbacc
         prefix:    cbacc
         reference: I-D.draft-jholland-mboned-cbacc
```

5. Security Considerations

5.1. Metadata Security

Be sure to authenticate the metadata. See DORMS security considerations, and don't accept unauthenticated metadata if using an alternative means.

5.2. Denial of Service

5.2.1. State Overload

Since CBACC flows require state, it may be possible for a set of receivers and/or senders, possibly acting in concert, to generate many flows in an attempt to overflow the circuit breakers' state tables.

It is permissible for a network node to behave as a CBACC circuit breaker for some CBACC flows while treating other CBACC flows as non-CBACC, as part of a load balancing strategy for the network as a whole, or simply as defense against this concern when the number of monitored flows exceeds some threshold.

The same techniques described in Section 3.1 of [RFC4609] can be used to help mitigate this attack, for much the same reasons. It is RECOMMENDED that network operators implement measures to mitigate such attacks.

6. Acknowledgements

Many thanks to Devin Anderson, Ben Kaduk, Cheng Jin, Scott Brown, Miroslav Ponec, Bob Briscoe, Lenny Giuliani, and Christian Worm Mortensen for their thoughtful comments and contributions.

7. References

7.1. Normative References

[I-D.draft-jholland-mboned-ambi-05]
Holland, J. and K. Rose, "Asymmetric Manifest Based Integrity", draft-jholland-mboned-ambi-05 (work in progress), March 2020.

[I-D.draft-jholland-mboned-dorms-02]
Holland, J., "Discovery Of Restconf Metadata for Source-specific multicast", draft-jholland-mboned-dorms-02 (work in progress), March 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <https://www.rfc-editor.org/info/rfc8084>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[FLID-DL] Byers, J., Horn, G., Luby, M., Mitzenmacher, M., Shaver, W., and IEEE, "FLID-DL: congestion control for layered multicast", DOI 10.1109/JSAC.2002.803998, n.d., <https://ieeexplore.ieee.org/document/1038584>.

[PathChirp] Ribeiro, V., Riedi, R., Baraniuk, R., Navratil, J., Cottrell, L., Department of Electrical and Computer Engineering Rice University, and SLAC/SCS-Network Monitoring, Stanford University, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths", 2003.

[PLM] Biersack, Institut EURECOM, A., "PLM: Fast Convergence for Cumulative Layered Multicast Transmission Schemes", 1999.

[RFC3738] Luby, M. and V. Goyal, "Wave and Equation Based Rate Control (WEBRC) Building Block", RFC 3738, DOI 10.17487/RFC3738, April 2004, <https://www.rfc-editor.org/info/rfc3738>.

[RFC4609] Savola, P., Lehtonen, R., and D. Meyer, "Protocol Independent Multicast - Sparse Mode (PIM-SM) Multicast Routing Security Issues and Enhancements", RFC 4609, DOI 10.17487/RFC4609, October 2006, <https://www.rfc-editor.org/info/rfc4609>.

[RFC5166] Floyd, S., Ed., "Metrics for the Evaluation of Congestion Control Mechanisms", RFC 5166, DOI 10.17487/RFC5166, March 2008, <https://www.rfc-editor.org/info/rfc5166>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6807] Farinacci, D., Shepherd, G., Venaas, S., and Y. Cai, "Population Count Extensions to Protocol Independent Multicast (PIM)", RFC 6807, DOI 10.17487/RFC6807, December 2012, <https://www.rfc-editor.org/info/rfc6807>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RLC] Rizzo, L., Vicisano, L., and J. Crowcroft, "The RLC multicast congestion control algorithm", 1999.

[RLM] McCanne, S., Jacobson, V., Vetterli, M., University of California, Berkeley, and Lawrence Berkeley National Laboratory, "Receiver-driven Layered Multicast", 1995.

[SMCC] Kwon, G., Byers, J., and Computer Science Department, Boston University, "Smooth Multirate Multicast Congestion Control", 2002.

Appendix A. Overjoining

[RFC8085] describes several remedies for unicast congestion control under UDP, even though UDP does not itself provide congestion control. In general, any network node under congestion could in theory collect evidence that a unicast flow's sending rate is not responding to congestion, and would then be justified in circuit-breaking it.

With multicast IP, the situation is different, especially in the presence of malicious receivers. A well-behaved sender using a receiver-controlled congestion scheme such as WEBRC does not reduce its send rate in response to congestion, instead relying on receivers to leave the appropriate multicast groups.

This leads to a situation where, when a network accepts inter-domain multicast traffic, as long as there are senders somewhere in the world with aggregate bandwidth that exceeds a network's capacity, receivers in that network can join the flows and overflow the network capacity. A receiver controlled by an attacker could do this at the IGMP/MLD level without running the application layer protocol that participates in the receiver-controlled congestion control.

A network might be able to detect and defend against the most naive version of such an attack by blocking end users that try to join too many flows at once. However, an attacker can achieve the same effect by joining a few high-bandwidth flows, if those exist anywhere, and an attacker that controls a few machines in a network can coordinate the receivers so they join disjoint sets of non-responsive sending flows.

This scenario will produce congestion in a middle node in the network that can't be easily detected at the edge where the IGMP/MLD join is accepted. Thus, an attacker with a small set of machines in a target network can always trip a circuit breaker if present, or can induce excessive congestion among the bandwidth allocated to multicast. This problem gets worse as more multicast flows become available.

Although the same can apply to non-responsive unicast traffic, network operators can assume that non-responsive sending flows are in violation of congestion control best practices, and can therefore cut off flows associated with the misbehaving senders. By contrast, non-responsive multicast senders are likely to be well-behaved participants in receiver-controlled congestion control schemes.

However, receiver controlled congestion control schemes also show the most promise for efficient massive scale content distribution via multicast, provided network health can be ensured. Therefore, mechanisms to mitigate overjoining attacks while still permitting receiver-controlled congestion control are necessary.

Author's Address

Jake Holland
Akamai Technologies, Inc.
150 Broadway
Cambridge, MA 02144
United States of America

Email: jakeholland.net@gmail.com