Mboned J. Holland
Internet-Draft K. Rose
Intended status: Standards Track Akamai Technologies, Inc.
Expires: September 11, 2020 March 10, 2020

Asymmetric Manifest Based Integrity
draft-ietf-mboned-ambi-00

Abstract

This document defines Asymmetric Manifest-Based Integrity (AMBI). AMBI allows each receiver or forwarder of a stream of multicast packets to check the integrity of the contents of each packet in the data stream. AMBI operates by passing cryptographically verifiable hashes of the data packets inside manifest messages, and sending the manifests over authenticated out-of-band communication channels.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Multicast transport poses security problems that are not easily addressed by the same security mechanisms used for unicast transport.

The "Introduction" sections of the documents describing TESLA [RFC4082], and TESLA in SRTP [RFC4383], and TESLA with ALC and NORM [RFC5776] present excellent overviews of the challenges unique to multicast authentication, briefly summarized here:

- A MAC based on a symmetric shared secret cannot be used because each packet has multiple receivers that do not trust each other, and using a symmetric shared secret exposes the same secret to each receiver.

- Asymmetric per-packet signatures can handle only very low bit-rates because of the computational overhead.

- An asymmetric signature of a larger message comprising multiple packets requires reliable receipt of all such packets, something that cannot be guaranteed in a timely manner even for protocols that do provide reliable delivery, and the retransmission of which may anyway exceed the useful lifetime for data formats that can otherwise tolerate some degree of loss.

Aymmetric Manifest-Based Integrity (AMBI) defines a method for receivers or middle boxes to cryptographically authenticate and verify the integrity of a stream of packets, by communicating packet "manifests" (described in Section 2.4) via an out-of-band communication channel that provides authentication and verifiable integrity.

Each manifest contains a message digest (described in Section 2.3) for each packet in a sequence of packets from the data stream, hereafter called a "packet digest". The packet digest incorporates a cryptographic hash of the packet contents and some identifying data from the packet, according to a defined digest profile for the data stream.

Each manifest MUST be delivered in a way that provides cryptographic integrity guarantees of the authenticity of the manifest. For example, TLS could be used to deliver a stream of manifests over a unicast data stream from a set of trusted senders to each receiver, or a protocol that asymmetrically signs each message could be used to transport authenticated manifests over a multicast channel. Note that a UDP-based protocol might drop or reorder manifests while still providing authentication.

Upon successful verification of a manifest and receipt of any subset of the corresponding data packets, the receiver has proof of the integrity of the contents of the data packets that are listed in the manifest.

Authenticating the integrity of the data packets depends on:

- the authenticity of the manifests; and

- the authenticity of the digest profile used for construction of the packet digests; and

- the difficulty of generating a collision for the packet digests contained in the manifest.

This document defines a YANG [RFC7950] module that augments the DORMS [I-D.draft-jholland-mboned-dorms-02] YANG module to provide a way to communicate a digest profile, described in Section 2.3.1, for construction of the packet digests, described in Section 2.3. When obtaining the digest profile by using DORMS, the authenticity of the data stream relies on a trust relationship with the DORMS server, since that anchors the authenticity of the digest profile for constructing packet digests.

1.1. Comparison with TESLA

AMBI and TESLA [RFC4082] and [RFC5776] attempt to achieve a similar goal of authenticating the integrity of streams of multicast packets. AMBI imposes a higher overhead, as measured in the amount of extra data required, than TESLA imposes. In exchange, AMBI provides non-repudiation (which TESLA does not), and relaxes the requirement for establishing an upper bound on clock synchronization between sender and receiver.

This tradeoff enables new capabilities for AMBI, relative to TESLA. In particular, when receiving multicast traffic from an untrusted transit network, AMBI can be used by a middle box to authenticate packets from a trusted source before forwarding traffic through the network, and the receiver also can separately authenticate the packets it receives.

This use case is not possible with TESLA because the data packets can’t be authenticated until a key is disclosed, so either the middlebox has to forward data packets without first authenticating them, so that the receiver has them prior to key disclosure, or the middlebox has to hold packets until the key is disclosed, at which point the receiver can no longer establish their authenticity.

The other new capability is that because AMBI provides authentication information out of band, authentication can be retrofitted into some pre-existing deployments without changing the protocol of the data packets, under some restrictions outlined in Section 7. By contrast, TESLA requires a MAC to be added to each authenticated message.

1.2. Threat Model

TBD: Summarize the applicable threat model this protects against. A diagram plus a cleaned-up version of the on-list explanation here is probably appropriate: https://mailarchive.ietf.org/arch/msg/mboned/CG9FLjPwuno3MtvYvgNcD5p69I4/

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

2. Protocol Operation

2.1. Overview

In order to authenticate a data packet, AMBI receivers need to hold these three pieces of information at the same time:

- the data packet; and
- an authenticated manifest containing the packet digest for the data packet; and
- a digest profile defining the transformation from the data packet to its packet digest.

The manifests are delivered as a stream of manifests over an authenticated data channel. Manifest contents MUST be authenticated before they can be used to authenticate data packets.

The manifest stream is composed of an ordered sequence of manifests that each contain an ordered sequence of packet digests, corresponding to the original packets as sent from their origin, in the same order.

Note that a manifest contains potentially many packet digests, and its size can be tuned to fit within a convenient PDU (Protocol Data Unit) of the manifest transport stream, so that usually, many packet digests for the multicast data stream can be delivered per packet of the manifest transport. The intent is that even with unicast-based manifest transport, multicast-style efficiencies of scale can still be realized, with only a relatively small unicast overhead, when manifests use a unicast transport.

2.2. Buffering and Validation Windows

Using different communication channels for the manifest stream and the data stream introduces a possibility of desynchronization in the timing of the received data between the different channels, so receivers hold data packets and packet digests from the manifest stream in buffers for some duration while awaiting the arrival of their counterparts.

While holding a data packet, if the corresponding packet digest for that packet arrives in the manifest stream and can be authenticated, the data packet is authenticated.

While holding an authenticated packet digest, if the corresponding data packet arrives with a matching packet digest, the data packet is authenticated.

Once a data packet is authenticated, the corresponding packet digest can be discarded and the data packet can be further processed by the receiving application or forwarded through the receiving network. Authenticating a data packet consumes one packet digest and prevents re-learning, with a hold-down time equal to the hold time for packet digests. A different manifest might provide the same packet digest with the same packet sequence number, but the digest remains consumed if it has been used to authenticate a data packet.

If the receiver's hold duration for a data packet expires without authenticating the packet, the packet SHOULD be dropped as unauthenticated. If the hold duration of a manifest expires, packet digests last received in that manifest SHOULD be discarded. (Note that in some cases, packet digests can be sent redundantly in more than one manifest. In such cases, the latest received time for an authenticated packet digest should be used for the expiration time.)

Since packet digests are usually smaller than the data packets, it's RECOMMENDED that senders generate and send manifests with timing such that the packet digests in a manifest will typically be received by subscribed receivers before the data packets corresponding to those digests are received.

This strategy reduces the buffering requirements at receivers at, the cost of introducing some buffering of data packets at the sender, since data packets are generated before their packet digests can be added to manifests.

The RECOMMENDED default hold times at receivers are:

- o 2 seconds for data packets

- 10 seconds for packet digests

The sender MAY recommend different values for specific data streams, in order to tune different data streams for different performance goals. The YANG model in Section 5 provides a mechanism for senders to communicate the sender's recommendation for buffering durations, when using DORMS.

Receivers SHOULD follow the recommendations for hold times provided by the sender, subject to their capabilities and any administratively configured limits on buffer sizes at the receiver.

However receivers MAY deviate from the values recommended by the sender for a variety of reasons. Decreasing the buffering durations recommended by the server increases the risk of losing packets, but can be an appropriate tradeoff for specific network conditions and hardware constraints on some devices.

TBD: should there be any reordering restrictions above and beyond the timing constraints?

2.2.1. Inter-packet Gap

It's RECOMMENDED that middle boxes forwarding buffered data packets preserve the inter-packet gap between packets, and that receiving libraries provide mechanisms to expose the network arrival times of packets to applications.

The purpose for this recommendation is to preserve the capability of receivers to use techniques for available bandwidth detection or network congestion based on observation of packet times. Examples of such techniques include paced chirping and pathrate.

Note that this recommendation SHOULD NOT prevent the transmission of an authenticated packet because the prior packet is unauthenticated. This recommendation only asks implementations to delay the transmission of an authenticated packet to correspond to the interpacket gap if an authenticated packet was previously transmitted and the authentication of the subsequent packet would otherwise burst the packets more quickly.

This does not prevent the transmission of packets out of order according to their order of authentication, only the timing of packets that are transmitted, after authentication, in the same order they were received.

For receiver applications, the time that the original packet was received from the network SHOULD be made available to the receiving application.

2.3. Packet Digests

2.3.1. Digest Profile

A packet digest is a message digest for a data packet, built according to a digest profile defined by the sender.

The digest profile is defined by the sender, and specifies:

1. A cryptographically secure hash algorithm (REQUIRED)

2. A manifest stream identifier

3. Whether to hash the IP payload or the UDP payload. (see Section 2.3.1.1)

The hash algorithm is applied to a pseudoheader followed by the packet payload, as determined by the digest profile. The computed hash value is the packet digest.

TBD: there should also be a way to specify that only packets to a specific UDP port are applicable. I think this is not quite right today and probably should be done with a grouping in the yang model, so that the profile appears either inside a "protocol" container inside the (S,G) or inside the udp-stream inside the "protocol", but am not sure. Follow-up on this after the first reference implementation...

2.3.1.1. Payload Type

2.3.1.1.1. UDP vs. IP payload validation

When the digest profile indicates that UDP payloads are validated, the IP protocol for the packets MUST be UDP (0x11) and the payload used for calculating the packet digest includes only the UDP payload, with length as the number of UDP payload octets, as calculated by subtracting the size of the UDP header from the UDP payload length.

When the digest profile indicates that IP payloads are validated, the IP payload of the packet is used, using the outermost IP layer that contains the (S,G) corresponding to the (S,G) protected by the manifest. There is no restriction on the IP protocols that can be authenticated. The length field in the pseudoheader is calculated by subtracting the IP Header Length from the IP length, and is equal to the number of octets in the payload for the digest calculation.

2.3.1.1.2. Motivation

Full IP payloads often aren't available to receivers without extra privileges on end user operating systems, so it's useful to provide a way to authenticate only the UDP payload, which is often the only portion of the packet available to many receiving applications.

However, for some use cases a full IP payload is appropriate. For example, when retrofitting some existing protocols, some packets may be predictable or frequently repeated. Use of an IPSec Authentication Header [RFC4302] is one way to disambiguate such packets. Even though the shared secret means the Authentication Header can't itself be used to authenticate the packet contents, the sequence number in the Authentication Header can ensure that specific packets are not repeated at the IP layer, and so it's useful for AMBI to have the capability to authenticate such packets.

Another example: some services might need to authenticate the UDP options [I-D.ietf-tsvwg-udp-options]. When using the UDP payload, the UDP options would not be part of the authenticated payload, but would be included when using the IP payload type.

Lastly, since (S,G) subscription operates at the IP layer, it's possible that some non-UDP protocols will need to be authenticated.

2.3.1.2. TBD: Packet contents?

TBD: Determine whether we need to support packet contents in the packet digest. If so, add to above list in Section 2.3.1:

- A set of bits from the packet contents (potentially empty)

The packet contents are a sequence of bits composed from a sequence of fixed bit (offset, length) pairs, as specified in xxxxxx. A useful choice for packet contents is to use sequence numbers in the application level protocol, such as with RTP [RFC3550], but any contents from the packet with a fixed bit offset and length can be used.

Providing variable packet contents in the packet digest increases the difficulty of attacking the hash by limiting the scope of legitimate data packets that can be matched when attempting to generate a hash collision.

The basic idea is to put an encoding here so that for example the RTP sequence number or the sequence number in an Authentication Header can be provided here in bulk (you give "value starts at bit 80 and is 16 bits long unsigned and increases by 1 per packet for the packets in the manifest with starting value 10", indicating that the 100 packets in the manifest have values 10-110 in their contents at the given location. Now those contents are prepended to the packet digest, and can be verified against the packets, as well as the hash of the contents).

For packet streams without a sequence number, we can instead incorporate a few high-entropy bits from the packet contents and NOT provide the value as a sequence number, but rather incorporate it in the digest values themselves. (Is this useful?)

Before defining this, I want to calculate how much overhead it buys us- how much can we truncate a good hash algorithm if we use this to add collision resistance? Might not be worthwhile, it's a significant increase in complexity. -jake 2019-08-31

If we need it, tentative addition to yang for the data profile looks like:

```
list packet-contents {
  key offset;
  description "contents from the packet for the packet
      digest";
  leaf offset {
    type uint16;
    mandatory true;
    description "offset of the contents, in number of bits";
  }
  leaf length {
    type uint16;
    mandatory true;
    description "length of the contents, in number of bits";
  }
  leaf manifest-delivery {
    type enumeration {
      enum sequence;
      enum digest;
    }
    mandatory true;
    description "the way these content bits are delivered in
        the manifest";
  }
}
```

The manifest-delivery would indicate whether the bits are a sequence number (in which case a section for a manifest with a start+step would be added ahead of the digests), or digest (indicating the bits appear inside each digest, ahead of the hash), and they would prepend in order to the packet digest, with sequence number bits inserted at the right bit location for the digest, based on earlier-appearing values, if any.

2.3.2. Pseudoheader

When calculating the hash for the packet digest, the hash algorithm is applied to a pseudoheader followed by the payload from the packet. The complete sequence of octets used to calculate the hash is structured as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Source Address (32 bits IPv4/128 bits IPv6)           |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Destination Address (32 bits IPv4/128 bits IPv6)        |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Zeroes    |   Protocol    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Manifest Identifier                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Payload Data                          |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.3.2.1. Source Address

The IPv4 or IPv6 source address of the packet.

2.3.2.2. Destination Address

The IPv4 or IPv6 destination address of the packet.

2.3.2.3. Zeroes

All bits set to 0.

2.3.2.4. Protocol

The IP Protocol field from IPv4, or the Next Header field for IPv6. When UDP payload is indicated, this value MUST be UDP (0x11).

2.3.2.5. Length

The length in octets of the Payload Data field, expressed as an unsigned 16-bit integer.

2.3.2.6. Source Port

The source port of the packet. Zeroes if using a protocol that does not use source ports.

2.3.2.7. Destination Port

The destination port of the packet. Zeroes if using a protocol that does not use destination ports.

TBD: there's something I hate about the source and destination ports. Maybe it should only be active in UDP-payload mode, instead of zeroes when not UDP? But I suspect there's a better approach than UDP-or-not, so it's this way for now, with hopes of finding something better in the next version.

2.3.2.8. Manifest Identifier

The 32-bit identifier for the manifest stream.

2.3.2.9. Payload Data

The payload data includes either the IP payload or the UDP payload, as indicated by the digest profile.

The payload type is configurable because when sending UDP, some legacy networks may strip the UDP option space, and it's necessary to provide a manifest stream capable of authentication that can interoperate with these networks. However, for non-UDP traffic or in order to authenticate the UDP options, some use cases may require support for authenticating the full IP payload.

2.4. Manifests

2.4.1. Manifest Layout

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Manifest Stream Identifier                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Manifest sequence number                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 First packet sequence number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Refresh Deadline        |      Packet Digest Count      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 ... Packet Content Expansions ...             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      ... Packet Digests ...                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.4.1.1. Manifest Stream Identifier

A 32-bit unsigned integer chosen by the sender.

2.4.1.2. Manifest Sequence Number

A monotonically increasing 32-bit unsigned integer. Each manifest sent by the sender increases this value by 1. On overflow it wraps to 0.

It's RECOMMENDED to expire the manifest stream and start a new stream for the data packets before a sequence number wrap is necessary.

2.4.1.3. First Packet Sequence Number

A monotonically increasing 32-bit unsigned integer. Each packet in the data stream increases this value by 1.

It's RECOMMENDED to expire the manifest stream and start a new stream for the data packets before a sequence number wrap is necessary.

Note: for redundancy, especially if using a manifest stream with unreliable transport, successive manifests MAY provide duplicates of the same packet digest with the same packet sequence number, using overapping sets of packet sequence numbers. When received, these reset the hold timer for the listed packet digests.

2.4.1.4. Refresh Deadline

A 16-bit unsigned integer number of seconds.

A zero value means the current digest profile for the current manifest stream is stable.

A nonzero value means that the authentication is transitioning to a new manifest stream, and the set of digest profiles SHOULD be refreshed by receivers that might stay joined longer than this duration, and a different manifest stream SHOULD be selected, before this many seconds have elapsed, in order to avoid a disruption. See Section 2.5.

2.4.1.5. Packet Digest Count

The count of packet digests in the manifest.

2.4.1.6. Packet Digests

Packet digests appended one after the other, aligned to 8-bit boundaries with zero padding (if the bit length of the digests are not multiples of 8 bits).

2.5. Transitioning to Other Manifest Streams

It's possible for multiple manifest streams authenticating the same data stream to be active at the same time. The different manifest streams can have different hash algorithms, manifest ids, and current packet sequence numbers for the same data stream. These result in different sets of packet digests for the same data packets, one digest per packet per digest profile.

It's necessary sometimes to transition gracefully from one manifest stream to another. The Refresh Deadline field from the manifest is used to signal to receivers the need to transition.

When a receiver gets a nonzero refresh deadline in a manifest the sender SHOULD have an alternate manifest stream ready and available, and the receiver SHOULD learn the alternate manifest stream, join the new one, and leave the old one before the number of seconds given in the refresh deadline. After the refresh deadline has expired, a manifest stream MAY end.

The receivers SHOULD use a random value between now and one half the number of seconds in the deadline field, to spread the spike of load on the DORMS server during a large multicast event.

3. Transport Considerations

3.1. Overview

AMBI manifests MUST be authenticated, but any transport protocol providing authentication can be used. This section discusses several viable options for the use of an authenticating transport, and some associated design considerations.

TBD: extend the 'manifest-transport' in the YANG model to make an extensible mechanism to advertise different transport options for receiving manifest streams.

TBD: add ALTA to the list when and if it gets further along [I-D.draft-krose-mboned-alta-01]. Sending an authenticatable multicast stream (instead of the below unicast-based proposals) is a worthwhile goal, else a 1% unicast authentication overhead becomes a new unicast limit to the scalability.

3.2. HTTPS

This document defines a new media type 'application/ambi' for use with HTTPS.

An HTTPS stream carrying the 'application/ambi' media type is composed of a sequence of binary AMBI manifests. It is RECOMMENDED to use Chunked encoding.

Complete packet Digests from partially received manifests MAY be used by the receiver for authentication, even if the full manifest is not yet delivered.

3.3. DTLS

TBD: DTLS [RFC6347] can provide authentication for datagrams, so if manifests can be constructed to fit within datagrams, it is an appropriate choice. (IPSec is similar-worth adding as an option?).

This option provides no native redundancy or retransmission, but packet digests can be repeated in different manifests to provide some resilience to loss. Lost manifests that result in the loss of blocks of packet digests can be expensive, since they would make received data packets unauthenticatable. TBD: should we therefore not support this case?

3.4. DTLS + FECFRAME

DTLS for manifests that do not fit into single-packet payloads can still be delivered by using FECFRAME [RFC6363], particularly Reed-Solomon [RFC6865] or possibly Raptor [RFC6681]. This has some advantages compared to HTTPS because of the absence of HOL-blocking, while providing for tunable redundancy. This has some advantages relative to DTLS because of overhead reduction and non-integer redundancy tunability (e.g. 1.5 becomes a viable redundancy factor).

TBD: define this method, possibly in another RFC.

4. Examples

TBD: walk through some examples as soon as we have a build running. Likely to need some touching up.

5. YANG Module

5.1. Tree Diagram

```
module: ietf-ambi
  augment /dorms:metadata/dorms:sender/dorms:group/dorms:udp-stream:
    +--rw manifest-stream* [id]
       +--rw id                        uint32
       +--rw manifest-transport*       inet:uri
       +--rw hash-algorithm            iha:hash-algorithm-type
       +--rw payload-type              enumeration
       +--rw data-hold-time-ms?        uint32
       +--rw digest-hold-time-ms?      uint32
```

5.2. Module

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

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

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

    import ietf-inet-types {
        prefix "inet";
```

```
        reference "RFC6991 Section 4";
    }

    import iana-hash-algs {
        prefix "iha";
        reference "draft-ietf-netconf-crypto-types";
    }

    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 RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     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 the AMBI data types.
     It provides metadata for authenticating SSM channels as an
     augmentation to DORMS.";

    revision 2019-08-25 {
        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";

        list manifest-stream {
            key id;
            description "Definition of a manifest stream.";
            leaf id {
                type uint32;
                mandatory true;
                description "The Manifest ID referenced in a manifest.";
            }
            leaf-list manifest-transport {
                type inet:uri;
                description "A URI that provides a location for the
                    manifest stream";
            }
            leaf hash-algorithm {
                type iha:hash-algorithm-type;
                mandatory true;
                description
                    "The hash algorithm for the packet hashes within
                     manifests in this stream.";
            }
            leaf payload-type {
                type enumeration {
                    enum udp {
                      description "The hash includes only the UDP
                          payload.";
                    }
                    enum ip {
                      description "The hash includes the full IP
                          payload.";
                    }
                }
                mandatory true;
                description "The contents of the payload for the
                    digest profile";
            }
            leaf data-hold-time-ms {
                type uint32;
                default 2000;
                description "The number of milliseconds to hold data
                    packets waiting for a corresponding digest before
                    discarding";
            }
            leaf digest-hold-time-ms {
                type uint32;
                default 10000;
```

```
                description "The number of milliseconds to hold packet
                    digests waiting for a corresponding data packet
                    before discarding";
            }
        }
    }
}
```

<CODE ENDS>

6. IANA Considerations

6.1. The 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-ambi
namespace: urn:ietf:params:xml:ns:yang:ietf-ambi
prefix:    ambi
reference: I-D.draft-jholland-mboned-ambi
```

6.2. Media Type

TBD: Register 'application/ambi' according to advice from: https://www.iana.org/form/media-types

TBD: check guidelines in https://tools.ietf.org/html/rfc5226

7. Security Considerations

7.1. Predictable Packets

Protocols that have predictable packets run the risk of offline attacks for hash collisions against those packets. When authenticating a protocol that might have predictable packets, it's RECOMMENDED to use a hash function secure against such attacks or to add content to the packets to make them unpredictable, such as an Authentication Header ([RFC4302]), or the addition of an ignored field with random content to the packet payload.

TBD: explain attack from generating malicious packets and then looking for collisions, as opposed to having to generate a collision on packet contents that include a sequence number and then hitting a match (especially expand on this if we do add Section 2.3.1.2).

TBD: follow the rest of the guidelines: https://tools.ietf.org/html/rfc3552

8. Acknowledgements

Many thanks to Daniel Franke, Eric Rescorla, Christian Worm Mortensen, Max Franke, and Albert Manfredi for their very helpful comments and suggestions.

9. References

9.1. Normative References

[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>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, DOI 10.17487/RFC6363, October 2011, <https://www.rfc-editor.org/info/rfc6363>.

[RFC6681] Watson, M., Stockhammer, T., and M. Luby, "Raptor Forward Error Correction (FEC) Schemes for FECFRAME", RFC 6681, DOI 10.17487/RFC6681, August 2012, <https://www.rfc-editor.org/info/rfc6681>.

[RFC6865] Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K. Matsuzono, "Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME", RFC 6865, DOI 10.17487/RFC6865, February 2013, <https://www.rfc-editor.org/info/rfc6865>.

[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>.

[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>.

9.2. Informative References

[I-D.draft-krose-mboned-alta-01]
Rose, K. and J. Holland, "Asymmetric Loss-Tolerant Authentication", draft-krose-mboned-alta-01 (work in progress), July 2019.

[I-D.ietf-tsvwg-udp-options]
Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-08 (work in progress), September 2019.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <https://www.rfc-editor.org/info/rfc3550>.

[RFC4082] Perrig, A., Song, D., Canetti, R., Tygar, J., and B. Briscoe, "Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction", RFC 4082, DOI 10.17487/RFC4082, June 2005, <https://www.rfc-editor.org/info/rfc4082>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4383] Baugher, M. and E. Carrara, "The Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Secure Real-time Transport Protocol (SRTP)", RFC 4383, DOI 10.17487/RFC4383, February 2006, <https://www.rfc-editor.org/info/rfc4383>.

[RFC5776] Roca, V., Francillon, A., and S. Faurite, "Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Asynchronous Layered Coding (ALC) and NACK-Oriented Reliable Multicast (NORM) Protocols", RFC 5776, DOI 10.17487/RFC5776, April 2010, <https://www.rfc-editor.org/info/rfc5776>.

[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>.

Authors' Addresses

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

Email: jakeholland.net@gmail.com

Kyle Rose
Akamai Technologies, Inc.
150 Broadway
Cambridge, MA 02144
United States of America

Email: krose@krose.org

Mboned J. Holland
Internet-Draft Akamai Technologies, Inc.
Intended status: Standards Track March 10, 2020
Expires: September 11, 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.

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:

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

- o 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:

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

- o '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:

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

- o '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

Mboned J. Holland
Internet-Draft Akamai Technologies, Inc.
Intended status: Standards Track March 10, 2020
Expires: September 11, 2020

Discovery Of Restconf Metadata for Source-specific multicast
draft-ietf-mboned-dorms-00

Abstract

This document defines DORMS (Discovery Of Restconf Metadata for Source-specific multicast), a method to discover and retrieve extensible metadata about source-specific multicast channels using RESTCONF. The reverse IP DNS zone for a multicast sender's IP address is configured to use SRV resource records to advertise the hostname of a RESTCONF server that publishes metadata according to a new YANG module with support for extensions. A new service name and the new YANG module are defined.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines DORMS (Discovery Of Restconf Metadata for Source-specific multicast).

A DORMS service is a RESTCONF [RFC8040] service that provides read access to data in the "ietf-dorms" YANG [RFC7950] model defined in Section 4. This model, along with optional extensions defined in other documents, provide an extensible set of information about multicast data streams.

This document defines the "dorms" service name for use with the SRV DNS Resource Record (RR) type [RFC2782]. A sender offering a DORMS service to publish metadata SHOULD configure at least one SRV RR for the "_dorms._tcp" subdomain in the reverse IP DNS zone for the source IP of its multicast channel to advertise a hostname for a DORMS server that can provide metadata for the sender's source-specific multicast traffic. Doing so enables receivers and middleboxes to discover and query a DORMS server as described in Section 2.

The goal is to provide an extensible framework for attaching information necessary for the correct processing of multicast data channels, both for middle boxes forwarding the traffic, and for receivers subscribing to traffic (hereafter called "clients").

1.1. Background

The reader is assumed to be familiar with the basic DNS concepts described in [RFC1034], [RFC1035], and the subsequent documents that update them, as well as the use of the SRV Resource Record type as described in [RFC2782].

The reader is also assumed to be familiar with the concepts and terminology regarding source-specific multicast as described in [RFC4607] and the use of IGMPv3 [RFC3376] and MLDv2 [RFC3810] for group management of source-specific multicast channels, as described in [RFC4604].

The reader is also assumed to be familiar with the concepts and terminology for RESTCONF [RFC8040] and YANG [RFC7950].

1.2. Terminology

Term	Definition
(S,G)	A source-specific multicast channel, as described in [RFC4607]. A pair of IP addresses with a source host IP and destination group IP.
RR	A DNS Resource Record, as described in [RFC1034]
RRType	A DNS Resource Record Type, as described in [RFC1034]
SSM	Source-specific multicast, as described in [RFC4607]

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

2. Discovery and Metdata Retrieval

A client that needs metadata about a (S,G) MAY attempt to discover metadata for the (S,G) using the mechanisms defined here, and MAY use the metadata received to manage the forwarding or processing of the packets in the channel.

2.1. DNS Bootstrap

The DNS Bootstrap step is how a client discovers an appropriate RESTCONF server, given the source address of an (S,G). Use of the DNS Bootstrap is OPTIONAL for clients with an alternate method of obtaining a RESTCONF hostname for a DORMS server with metadata for an (S,G).

This mechanism only works for source-specific multicast (SSM) channels. The source address of the (S,G) is reversed and used as an index into one of the reverse mapping trees (in-addr.arpa for IPv4, as described in Section 3.5 of [RFC1035], or ip6.arpa for IPv6, as described in Section 2.5 of [RFC3596]).

When a receiver or middle box needs metadata for an (S,G), for example when handling a new join for that (S,G) and looking up authentication methods available, a receiver or middlebox can issue a DNS query for a SRV RR using the "dorms" service name with the domain from the reverse mapping tree, combining them as described in [RFC2782].

For example, while handling a join for (203.0.113.15, 232.1.1.1), a receiver would perform a DNS query for the SRV RRType for the domain:

```
_dorms._tcp.15.113.0.203.in-addr.arpa.
```

The DNS response for this domain might return a record such as:

```
SRV 0 1 443 dorms-restconf.example.com.
```

This response informs the receiver that a DORMS server SHOULD be reachable at dorms-restconf.example.com on port 443. Multiple SRV records are handled as described by [RFC2782].

A sender providing DORMS discovery SHOULD publish at least one SRV record in the reverse DNS zone for each source address of the multicast channels it is sending, in order to advertise the hostname of the DORMS server to receivers and middle boxes. The DORMS servers advertised SHOULD be configured with metadata for all the groups sent from the same source IP address that have metadata published with DORMS.

2.2. RESTCONF Bootstrap

Once a DORMS host has been chosen (whether via an SRV RR from a DNS response or via some other method), RESTCONF provides all the information necessary to determine the versions and url paths for metadata from the server. A walkthrough is provided here for a sequence of example requests and responses from a receiver connecting to a new DORMS server.

2.2.1. Root Resource Discovery

As described in Section 3.1 of [RFC8040] and [RFC6415], the RESTCONF server provides the link to the RESTCONF api entry point via the "/.well-known/host-meta" or "/.well-known/host-meta.json" resource.

Example:

The receiver might send:

```
GET /.well-known/host-meta.json HTTP/1.1
Host: dorms-restconf.example.com
Accept: application/json
```

The server might respond as follows:

```
HTTP/1.1 200 OK
Date: Tue, 27 Aug 2019 20:56:00 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/json

{
  "links":[
    {
      "rel":"restconf",
      "href":"/top/restconf"
    }
  ]
}
```

2.2.2. Yang Library Version

As described in Section 3.3.3 of [RFC8040], the yang-library-version leaf is required by RESTCONF, and can be used to determine the schema of the ietf-yang-library module:

Example:

The receiver might send:

```
GET /top/restconf/yang-library-version HTTP/1.1
Host: dorms-restconf.example.com
Accept: application/yang-data+json
```

The server might respond as follows:

```
HTTP/1.1 200 OK
Date: Tue, 27 Aug 2019 20:56:01 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/yang-data+json

{
    "ietf-restconf:yang-library-version": "2016-06-21"
}
```

TBD: We might need a method for learning a specific restconf server or resource path that supports a version the client knows how to use, in the case the client is older than the server after a new yang-library version is released... Can this be just retry with a hold-down on specific hostnames, so that you can find a lower priority older server from the SRV records, or is signaling that can find or negotiate an explicit version as part of the lookup going to be necessary? -jake 2019-08-26

2.2.3. Yang Library Contents

After checking that the version of the yang-library module will be understood by the receiver, the client can check that the desired metadata module is available on the DORMS server by fetching the module-state resource from the ietf-yang-library module.

Example:

The receiver might send:

```
GET /top/restconf/data/ietf-yang-library:modules-state/\
    module=ietf-dorms,2016-08-15
Host: dorms-restconf.example.com
Accept: application/yang-data+json
```

The server might respond as follows:

```
HTTP/1.1 200 OK
Date: Tue, 27 Aug 2019 20:56:02 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/yang-data+json
```


```
{
  "ietf-yang-library:module": [
    {
      "conformance-type": "implement",
      "name": "ietf-dorms",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-dorms",
      "revision": "2019-08-25",
      "schema":
          "https://example.com/yang/ietf-dorms@2019-08-25.yang"
    }
  ]
}
```


Other modules required or desired by the client also can be checked in a similar way, or the full set of available modules can be retrieved by not providing a key for the "module" list.

2.2.4. Metadata Retrieval

Once the expected DORMS version is confirmed, the client can retrieve the metadata specific to the desired (S,G).

Example:

The receiver might send:

```
GET /top/restconf/data/ietf-dorms:metadata/\
    sender=203.0.113.15/group=232.1.1.1
Host: dorms-restconf.example.com
Accept: application/yang-data+json
```

The server might respond as follows:

```
HTTP/1.1 200 OK
Date: Tue, 27 Aug 2019 20:56:02 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/yang-data+json
```


```
{
  "ietf-dorms:group": [
    {
      "group-address":"232.1.1.1",
      "udp-stream":[
        {
          "port":"5001"
        }
      ]
    }
  ]
}
```


Note that when other modules are installed on the DORMS server that extend the ietf-dorms module, other fields MAY appear inside the response. This is the primary mechanism for providing extensible metadata for an (S,G), so clients SHOULD ignore fields they do not understand.

As mentioned in Section 3.2, most clients SHOULD use data resource identifiers in the request URI as in the above example, in order to retrieve metadata for only the targeted (S,G)s.

2.2.5. Cross Origin Resource Sharing (CORS)

It is RECOMMENDED that DORMS servers use the Access-Control-Allow-Origin header field, as specified by [W3C.REC-cors-20140116], and that they respond appropriately to Preflight requests.

Providing '*' for the allowed origins exposes the DORMS-based metadata to all web pages. When access to the metadata is used as a prerequisite to permitting the joining of the multicast flows, this would permit scripts from arbitrary web pages to issue joins for the multicast flows, which could allow e.g. malicious advertisements to participate in overjoining attacks (see Appendix A of [I-D.draft-jholland-cb-assisted-cc-01]) using multicast flows not controlled by the ad's senders. Therefore the use of '*' for allowed origins is NOT RECOMMENDED. (TBD: this probably deserves a security considerations section.)

3. Scalability Considerations

3.1. Provisioning

In contrast to many common RESTCONF deployments that are intended to provide configuration management for a service to a narrow set of authenticated administrators, DORMS servers often provide read-only metadata for public access, or for a very large set of end receivers, since it provides metadata in support of multicast data streams and multicast can scale to very large audiences.

Operators are advised to provision the DORMS service in a way that will scale appropriately to the size of the expected audience. Specific advice on such scaling is out of scope for this document, but some of the mechanisms outlined in [RFC3040] or other online resources might be useful, depending on the expected number of receivers.

3.2. Data Scoping

In the absence of contextual information, clients SHOULD issue narrowed requests for DORMS resources by following the format from Section 3.5.3 of [RFC8040] to encode data resource identifiers in the request URI. This avoids downloading excessive data, since the DORMS server may provide metadata for many (S,G)s, possibly from many different senders.

However, clients MAY use heuristics or out of band information about the service to issue requests for (S,G) metadata narrowed only by the source-address, or not narrowed at all. Depending on the request patterns and the contents of the data store, this may result in fewer round trips or less overhead, and can therefore be helpful behavior for scaling purposes. Servers MAY restrict or throttle client access based on the client certificate presented (if any), or based on heuristics that take note of client request patterns.

A complete description of the heuristics for clients and servers to meet their scalability goals is out of scope for this document.

4. YANG Model

The primary purpose of the YANG model defined here is to serve as a scaffold for the more useful metadata that will extend it. Currently known use cases include providing authentication information and bit-rate information for use by receivers and middle boxes, but more use cases are anticipated.

4.1. Yang Tree

```
module: ietf-dorms
  +--rw metadata
     +--rw sender* [source-address]
        +--rw source-address    inet:ip-address
        +--rw group* [group-address]
           +--rw group-address    rt-types:ip-multicast-group-address
           +--rw udp-stream* [port]
              +--rw port    inet:port-number
```

DORMS Tree Diagram

4.2. Yang Module

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

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

    import ietf-inet-types {
        prefix "inet";
        reference "RFC 6991 Section 4";
    }

    import ietf-routing-types {
        prefix "rt-types";
        reference "RFC 8294";
    }

    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 RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     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 the DORMS data type.
     It provides out of band metadata about SSM channels.";

    revision 2019-08-25 {
        description "Initial revision.";
        reference
            "";
            // "I-D.draft-jholland-mboned-dorms";
    }

    container metadata {
        description "Metadata scaffold for source-specific multicast
            channels.";
        list sender {
            key source-address;
            description "Sender for DORMS";

            leaf source-address {
                type inet:ip-address;
                mandatory true;
                description
                    "The source IP address of a multicast sender.";
            }

            list group {
                key group-address;
                description "Metadata for a DORMS (S,G).";

                leaf group-address {
                    type rt-types:ip-multicast-group-address;
                    mandatory true;
                    description "The group IP address for an (S,G).";
                }

                list udp-stream {
```

```
                   key "port";
                   description
                       "Metadata for UDP traffic on a specific port.";
                   leaf port {
                       type inet:port-number;
                       mandatory true;
                       description
                           "The UDP port of a data stream in an (S,G).";
                   }
                 }
             }
         }
     }
 }
 <CODE ENDS>
```

5. Privacy Considerations

5.1. Linking Content to Traffic Streams

In the typical case, the mechanisms defined in this document provide a standardized way to discover information that is already available in other ways.

However, depending on the metadata provided by the server, observers may be able to more easily associate traffic from an (S,G) with the content contained within the (S,G). At the subscriber edge of a multicast-capable network, where the network operator has the capability to localize an IGMP [RFC3376] or MLD [RFC3810] channel subscription to a specific user or location by MAC address or source IP address, the structured publishing of metadata may make it easier to automate collection of data about the content a receiver is consuming.

5.2. Linking Multicast Subscribers to Unicast Connections

Subscription to a multicast channel generally only exposes the IGMP or MLD membership report to others on the same LAN, and as the membership propagates through a multicast-capable network, it ordinarily gets aggregated with other end users.

However, a RESTCONF connection is a unicast connection, and exposes a different set of information to the operator of the RESTCONF server, including IP address and timing about the requests made. Where DORMS access becomes required to succeed a multicast join, as expected in a browser deployment, this can expose new information about end users relative to services based solely on multicast streams.

In some deployments it may be possible to use a proxy that aggregates many end users when the aggregate privacy characteristics are needed by end users.

6. IANA Considerations

6.1. The 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-dorms
namespace: urn:ietf:params:xml:ns:yang:ietf-dorms
prefix:    dorms
reference: I-D.draft-jholland-mboned-dorms
```

6.2. The Service Name and Transport Protocol Port Number Registry

This document adds one service name to the "Service Name and Transport Protocol Port Number Registry" maintained at <https://www.iana.org/assignments/service-names-port-numbers>. The following registrations are made, per the format in Section 8.1.1 of [RFC6335]:

```
Service Name:            dorms
Transport Protocol(s):   TCP
Assignee:                IESG <iesg@ietf.org>
Contact:                 IETF Chair <chair@ietf.org>
Description:             This service name is used to construct the
                         SRV service label "_dorms" for discovering
                         DORMS servers.
Reference:               I-D.draft-jholland-mboned-dorms
Port Number:             N/A
Service Code:            N/A
Known Unauthorized Uses: N/A
Assignment Notes:        This protocol uses HTTPS as a substrate.
```

7. Security Considerations

7.1. Secure Communications

It is intended that security related metadata about the SSM channels will be delivered over the RESTCONF connection, and that information available from this connection can be used as a trust anchor.

The provisions of Section 2 of [RFC8040] provide secure communication requirements that are already required of DORMS servers, since they are RESTCONF servers. All RESTCONF requirements and security considerations remain in force for DORMS servers.

7.2. Exposure of Metadata

Although some DORMS servers MAY restrict access based on client identity, as described in Section 2.5 of [RFC8040], many DORMS servers will use the ietf-dorms YANG model to publish information without restriction, and even DORMS servers requiring client authentication will inherently, because of the purpose of DORMS, be providing the DORMS metadata to potentially many receivers.

Accordingly, future YANG modules that augment data paths under "ietf-dorms:metadata" MUST NOT include any sensitive data unsuitable for public dissemination in those data paths. Because of the possibility that scalable read-only access might be necessary to fulfill the scalability goals for a DORMS server, data under these paths MAY be cached or replicated by numerous external entities, so owners of such data SHOULD NOT assume it can be kept secret when provided by DORMS servers anywhere under the "ietf-dorms:metadata" path, even if they are authenticating clients.

7.3. DNS Bootstrapping

The DNS bootstrap phase relies on DNS for the reverse IP tree. When using DNS to discover a DORMS server's domain name, there must be a trust relationship between the end consumer of this resource record and the DNS server. This relationship may be end-to-end DNSSEC validation, a TSIG [RFC2845] or SIG(0) [RFC2931] channel to another secure source, a secure local channel on the host, DNS over TLS [RFC7858] or HTTPS [RFC8484], or some other secure mechanism.

If the SRV Resource Record cannot be authenticated, it may be possible for an attacker who can spoof the resource record to perform a denial of service for the receiver by providing wrong or missing authentication metadata. An attacker who can also inject traffic for (S,G)s, would also be able to provide false content in the data stream, so an attacker who can perform both could provide authenticated false content by authenticating with a trust anchor from an attacker-controlled DORMS server.

Clients MAY use other secure methods to explicitly associate an (S,G) with a set of DORMS server hostnames, such as a configured mapping or an alternative trusted lookup service.

8. Acknowledgements

Thanks to Christian Worm Mortensen for some very helpful comments and review.

9. References

9.1. Normative References

[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>.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <https://www.rfc-editor.org/info/rfc2782>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <https://www.rfc-editor.org/info/rfc3596>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[W3C.REC-cors-20140116]
Kesteren, A., "Cross-Origin Resource Sharing", World Wide Web Consortium Recommendation REC-cors-20140116, January 2014, <http://www.w3.org/TR/2014/REC-cors-20140116>.

9.2. Informative References

[I-D.draft-jholland-cb-assisted-cc-01]
Holland, J., "Circuit Breaker Assisted Congestion Control (CBACC): Protocol Specification", draft-jholland-cb-assisted-cc-01 (work in progress), April 2017.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <https://www.rfc-editor.org/info/rfc2845>.

[RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, DOI 10.17487/RFC2931, September 2000, <https://www.rfc-editor.org/info/rfc2931>.

[RFC3040] Cooper, I., Melve, I., and G. Tomlinson, "Internet Web Replication and Caching Taxonomy", RFC 3040, DOI 10.17487/RFC3040, January 2001, <https://www.rfc-editor.org/info/rfc3040>.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, DOI 10.17487/RFC3376, October 2002, <https://www.rfc-editor.org/info/rfc3376>.

[RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <https://www.rfc-editor.org/info/rfc3810>.

[RFC4604] Holbrook, H., Cain, B., and B. Haberman, "Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast", RFC 4604, DOI 10.17487/RFC4604, August 2006, <https://www.rfc-editor.org/info/rfc4604>.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <https://www.rfc-editor.org/info/rfc4607>.

[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>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC6415] Hammer-Lahav, E., Ed. and B. Cook, "Web Host Metadata", RFC 6415, DOI 10.17487/RFC6415, October 2011, <https://www.rfc-editor.org/info/rfc6415>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[whatwg-fetch]
Kesteren, A., "WHATWG Fetch Living Standard", August 2019, <https://fetch.spec.whatwg.org/>.

Author's Address

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

Email: jakeholland.net@gmail.com