Workgroup: Mboned
Internet-Draft: draft-ietf-mboned-ambi-03
Published: 7 March 2022
Intended Status: Standards Track
Expires: 8 September 2022
Authors: J. Holland
Akamai Technologies, Inc.
K. Rose
Akamai Technologies, Inc.

Asymmetric Manifest Based Integrity

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on 8 September 2022.

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 for use cases like wide scale software or video distribution with a high data transfer rate. The challenges are 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 transport and computational overhead associated with signature transmission and verification.

- 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 comparing the data packets to a stream of packet "manifests" (described in Section 3.4) received via an out-of-band communication channel that provides authentication and verifiable integrity.

Each manifest contains a message digest (described in Section 3.3) 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.

Upon receipt of a packet digest inside a manifest conveyed in a secure channel and verification that the packet digest of a received data packet matches, the receiver has proof of the integrity of the contents of the data packet corresponding to that digest.

This document defines the "ietf-ambi" YANG [RFC7950] model in Section 6 as an extension of the "ietf-dorms" model defined in [I-D.draft-ietf-mboned-dorms]. Also defined are new URI schemes for transport of manifests over TLS or DTLS, and a new media type for transport of manifests over HTTPS. The encodings for these are defined in Section 4.

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 than TESLA imposes, as measured in the amount of extra data required. In exchange, AMBI relaxes the requirement for establishing an upper bound on clock synchronization between sender and receiver, and allows for the use case of authenticating multicast traffic before forwarding it through the network, while also allowing receivers to authenticate the same traffic. By contrast, this 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 8. By contrast, TESLA requires a MAC to be added to each authenticated message.

1.2. 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.

1.3. Notes for Contributors and Reviewers

Note to RFC Editor: Please remove this section and its subsections before publication.

This section is to provide references to make it easier to review the development and discussion on the draft so far.

1.3.1. Venues for Contribution and Discussion

This document is in the Github repository at:

https://github.com/GrumpyOldTroll/ietf-dorms-cluster

Readers are welcome to open issues and send pull requests for this document.

Please note that contributions may be merged and substantially edited, and as a reminder, please carefully consider the Note Well before contributing: https://datatracker.ietf.org/submit/note-well/

Substantial discussion of this document should take place on the MBONED working group mailing list (mboned@ietf.org).

*Join: https://www.ietf.org/mailman/listinfo/mboned

*Search: https://mailarchive.ietf.org/arch/browse/mboned/

1.3.2. Non-obvious doc choices

*TBD: we need a way to assert that we provide the full set of packets for an (S,G) on all UDP ports and non-UDP protocols. Naively authenticating UDP for specified ports and ignoring other ports means that an attacker could attack a separate UDP port by injecting traffic directed at it, potentially hitting a different application that listens on 0.0.0.0, so an (S,G) with legitimately authenticated UDP traffic on one port could be used to transport UDP-based attacks to apps on another port or protocol unless they are firewalled. Passing traffic for an (S,G) subscription would open a new channel to such targets that otherwise would not be reachable from the internet for users behind e.g. a CPE with nat or connection-state-based firewalling.

*Dropped intent to support DTLS+FECFRAME in this spec because RFC 6363 seems incomprehensible on a few points, most notably demux strategy between repair and source ADUs, which as written seems to require specifying another layer. So support for this will have to be a later separate RFC. However, for future extensibility made manifest-stream into a list instead of a leaf-list so that it can be an augment target for a later YANG extension with FEC selection from the likewise-very-confusing semi-overlapping registries at https://www.iana.org/assignments/rmt-fec-parameters/rmt-fec-parameters.xhtml defined by RFCs 5052 and 6363. See also RFC 6363, RFC 6681, and RFC 6865

2. Threat Model

AMBI is designed to operate over the internet, under the Internet Threat Model described in [RFC3552].

AMBI aims to provide Data Integrity for a multicast data stream, building on the security anchors described in Section 2.1 to do so. The aim is to enable receivers to subscribe to and receive multicast packets from a trusted sender without damage to the Systems Security (Section 2.3 of [RFC3552]) for those receivers or other entities.

Thus, we assume there might be attackers on-path or off-path with the capability to inject or modify packets, but that the attackers have not compromised the sender or discovered any of the sender's secret keys. We assume that an attacker may have compromised some receivers of the multicast traffic, but still aim to provide the above security properties for receivers that have not been compromised.

Those sending multicast traffic to receivers that include untrusted receivers should avoid transmitting sensitive information that requires strong confidentiality guarantees, due to the risk of compromise from those receivers. Since multicast transmits the same packets to potentially many receivers, in the presence of potentially compromised receivers confidentiality of the content cannot be assured.

However, any protocol that provides encryption of the packet data before generating the packet digest can provide confidentiality against on-path passive observers who do not possess the decryption key. This level of confidentiality can be provided by any such protocols without impact on AMBI's operation.

2.1. Security Anchors

Establishing the desired security properties for the multicast data packets relies on secure delivery of some other information:

* Secured unicast connections (providing Data Integrity) to one or more trusted DORMS [I-D.draft-ietf-mboned-dorms] servers that use the AMBI extensions to the DORMS YANG model as defined in Section 6

* Secure delivery (providing Data Integrity) of a stream of manifests (Section 3.4)

The secured unicast connection to the DORMS server provides the Peer Entity authentication of the DORMS server that's needed to establish the Data Integrity of the data it sends.

Note that DORMS provides a method for using DNS to bootstrap discovery of the DORMS server. In contexts where secure DNS lookup cannot be provided, it's still possible to establish a secure connection to a trusted DORMS server as long as the trusted DORMS server's hostname is known to the receivers (removing the need to use DNS for that discovery). Once the server name is known, the ordinary certificate verification of that hostname while establishing a secure https connection provides the needed security properties to anchor the rest.

Receiving unauthenticated data packets and knowing how to generate packet digests from the manifest profile provided by the AMBI extensions in the DORMS metadata allows the receiver to generate packet digests based on the contents of the received packet, which can be compared against the packet digests that were securely received.

Comparing the digests and finding the same answer then provides Data Integrity for the data packets that relies on one more property of the digest generation algorithm:

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

Taken together, successful validation of the multicast data packets proves within the above constraints that someone with control of the manifest URI streams provided by the DORMS server has verified the sending of the packets corresponding to the digests sent in that stream of manifests.

2.1.1. Alternatives and Their Requirements

Other protocols that can provide authentication could also be used for manifest delivery if defined later in another specification. For example a protocol that asymmetrically signs each packet, as the one defined in Section 3 of [RFC6584] does, would be a viable candidate for a delivery protocol for manifests that could be delivered over a multicast transport, which could have some important scalability benefits.

Other methods of securely transmitting metadata equivalent to the metadata provided by the "ietf-ambi" YANG model could also be used to provide the same security guarantees with the manifest channels. Defining other such possibilities is out of scope for this document.

2.2. System Security

By providing the means to authenticate multicast packets, AMBI aims to avoid giving attackers who can inject or modify packets the ability to attack application vulnerabilities that might be possible

to exercise if those applications process the attack traffic. Many of the entries in the Common Vulnerabilities and Exposures (CVE) list at [CVE] (an extensive industry-wide database of software vulnerabilities) have documented a variety of system security problems that can result from maliciously generated UDP packets.

TBD: Fold in a mention of how off-path attacks are possible from most places on the internet for interdomain multicast over AMT at an ingest point, and how the multicast fanout downstream of that can make it a good target if multicast sees more use. A diagram plus a cleaned-up version of the on-list explanation here is probably appropriate: https://mailarchive.ietf.org/arch/msg/mboned/CG9FLjPwuno3MtvYvgNcD5p69I4/. Nightmare scenario is zero-day RCE by off-path attacker that takes over a significant number of the devices watching a major sports event.

See also work-in-progress: https://squarooticus.github.io/draft-krose-multicast-security/draft-krose-multicast-security.html

3. Protocol Operation

3.1. 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

* an authenticated manifest containing the packet digest for the data packet

* 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. By doing so, 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.

3.2. Buffering of Packets and Digests

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 secured manifest stream, 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.

Authenticating a data packet consumes one packet digest and prevents re-learning a digest for the same sequence number with a hold-down time equal to the hold time for packet digests. The hold-down is necessary because a different manifest can send a duplicate packet digest for the same packet sequence number, either when repeating of packet digests is used for resilience to loss or when rotating authentication keys, so re-learning the packet digest could allow a replay of a data packet. After authenticating a packet, the digest and any future digests for the same data packet remain consumed if it has been used to authenticate a data packet, ignoring repeated digests for the same sequence number until after the holddown timer expires.

Once the data packet is authenticated it can be further processed by the receiving application or forwarded through the receiving network.

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 MUST be discarded.

When multiple digests for the same packet sequence number are received, the latest received time for an authenticated packet digest should be used for the expiration time.

3.2.1. Validation Windows

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:

*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 6 provides a mechanism for senders to communicate the sender's recommendation for buffering durations. These parameters are "data-hold-time" and "digest-hold-time", expressed in milliseconds.

Receivers MAY deviate from the values recommended by the sender for a variety of reasons, including their own memory constraints or local administrative configuration (for example, it might improve user experience in some situations to hold packets longer than the server recommended when there are receiver-specific delays in the manifest stream that exceed the server's expectations). 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 or memory constraints on some devices.

Receivers SHOULD follow the recommendations for hold times provided by the sender (including the default values from the YANG model when unspecified), subject to their capabilities and any administratively configured overrides at the receiver.

3.2.2. Preserving Inter-packet Gap

It's RECOMMENDED that middle boxes forwarding buffered data packets preserve the inter-packet gap between packets in the same data stream, and that receiving libraries that perform AMBI-based authentication 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 and packet dispersal, making use of known patterns in the sending. Examples of

such techniques include those described in [PathChirp], [PathRate], and [WEBRC].

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.

3.3. Packet Digests

3.3.1. 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 3.3.1.1)

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: As recommended by https://tools.ietf.org/html/rfc7696#section-2.2, a companion document containing the mandatory-to-implement cipher suite should also be published separately and referenced by this document.

3.3.1.1. Payload Type

3.3.1.1.1. UDP vs. IP payload validation

When the manifest definition is at the UDP layer, it applies only to packets with IP protocol of 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 manifest definition is at the IP layer, the payload used for calculating the packet digest includes the full IP payload of the data packets in the (S,G). 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.

3.3.1.1.2. 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.

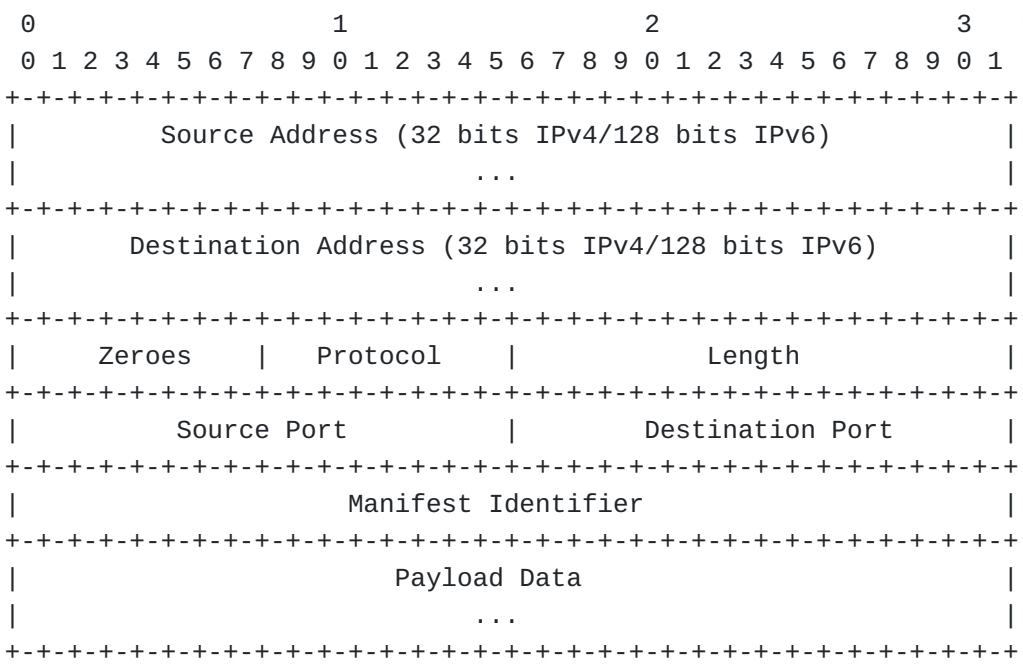
Another example: some services might need to authenticate the UDP options [I-D.ietf-tsvwg-udp-options]. When using the UDP payload, the UDP options would not be part of the authenticated payload, but would be included when using the IP payload type.

Lastly, since (S,G) subscription operates at the IP layer, it's possible that some non-UDP protocols will need to be authenticated, and the IP layer allows for this. However, most user-space transport applications are expected to use the UDP layer authentication.

3.3.2. 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:



3.3.2.1. Source Address

The IPv4 or IPv6 source address of the packet.

3.3.2.2. Destination Address

The IPv4 or IPv6 destination address of the packet.

3.3.2.3. Zeroes

All bits set to 0.

3.3.2.4. Protocol

The IP Protocol field from IPv4, or the Next Header field for IPv6. When using UDP-layer authentication, this value is always UDP (0x11) but for IP-layer authentication it can vary per-packet.

3.3.2.5. Length

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

3.3.2.6. Source Port

The source port of the packet. Zeroes if using IP-layer authentication for a non-UDP protocol.

3.3.2.7. Destination Port

The UDP destination port of the packet. Zeroes if using IP-layer authentication for a non-UDP protocol.

3.3.2.8. Manifest Identifier

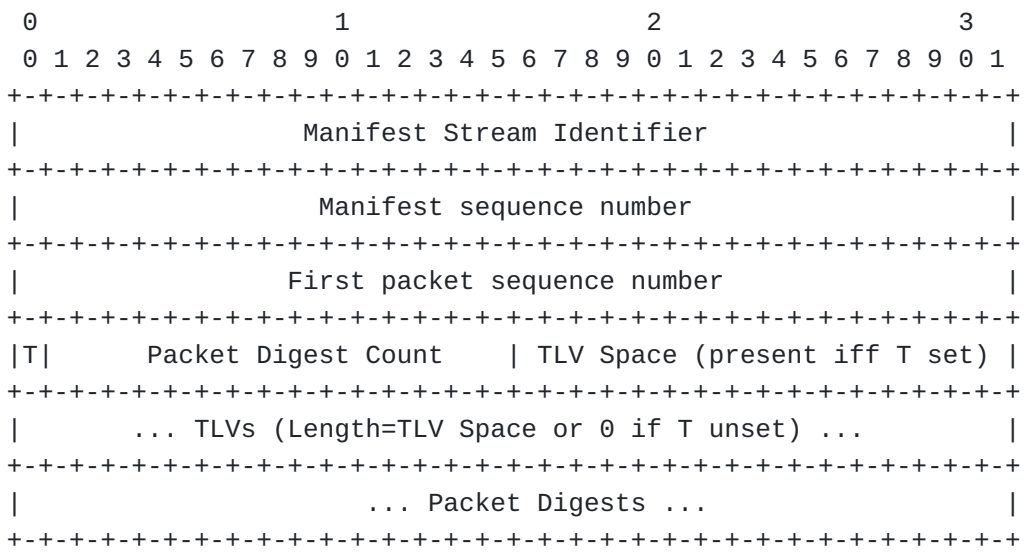
The 32-bit identifier for the manifest stream.

3.3.2.9. Payload Data

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

3.4. Manifests

3.4.1. Manifest Layout



3.4.1.1. Manifest Stream Identifier

A 32-bit unsigned integer chosen by the sender. This value MUST be equal to the "id" field in the manifest-stream in the "ietf-ambi" model. If a manifest is seen that does not have the expected value from the metadata provided for the manifest, the receiver MUST stop processing this manifest and disconnect from this manifest stream. It MAY reconnect with an exponential backoff starting at 1s, or it MAY connect to an alternative manifest stream if one is known.

3.4.1.2. 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.

3.4.1.3. 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.

3.4.1.4. T bit (TLVs Present)

If 1, this indicates the TLV Length and TLV space fields are present. If 0, this indicates neither field is present.

3.4.1.5. Packet Digest Count

A 15-bit unsigned integer equal to the count of packet digests in the manifest.

3.4.1.6. TLV Space

A 16-bit unsigned integer with the length of the TLVs section.

3.4.1.7. TLVs

These are Type-Length-Value blocks, back to back. These may be extended by future specifications.

These are composed of 3 fields:

- *Type: an 8-bit unsigned integer indicating the type. Type values in 0-127 have an 8-bit length, and type values in 128-255 have a 16-bit length.

- *Length: a 8-bit or 16-bit unsigned integer indicating the length of the value

- *Value: a value with semantics defined by the Type field.

Defined values:

Type	Name	Value
0	Pad	Length can be 0-255. Value is filled with 0 and ignored by receiver.
128	Refresh Deadline	Length MUST be 2. Value is a 16-bit unsigned integer number of seconds. When this field is absent or zero, it means the current digest profile for the current manifest stream is stable. A nonzero value means the authentication is transitioning to a new manifest stream, and the set of digest profiles SHOULD be refreshed by receivers before this much time has elapsed in order to avoid a disruption. See Section 3.5.

Table 1

1-120 and 129-248 are unassigned 121-127 and 249-255 are reserved for experiments

Any unknown values MUST be skipped and ignored by the receiver, using the Length field to skip.

The total size of the manifest in octets is exactly equal to:

Size of digests * packet count + 14 if T is 0 Size of digests * packet count + 16 + TLV Length if T is 1

The total size of the TLV space is exactly equal to:

(2 + Length) summed for each TLV

The total size of the TLV space MUST exactly equal TLV Length. If the TLV space exceeds the TLV Length, the receiver MUST disconnect, and behave as if the Manifest Stream Identifier was wrong. This state indicates a failed decoding of the TLV space.

3.4.1.8. Packet Digests

Packet digests appended one after the other, aligned to 8-bit boundaries with 0-bit padding at the end if the bit length of the digests are not multiples of 8 bits.

3.5. 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 TLV 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 stop transmitting and close connections from the server side. When multiple manifest-streams are provided in the metadata, all or all but one SHOULD contain an expire-time, and new or refreshing receivers SHOULD choose a manifest stream without an expire-time, or with the latest expire-time if all manifests have an expire-time.

The receivers SHOULD start the refresh after a random time delay between now and one half the number of seconds in the deadline field after the first manifest they receive containing a nonzero refresh deadline. This time delay is to desynchronize the refresh attempts in order to spread the spike of load on the DORMS server while changing manifest profiles during a large multicast event.

4. Transport Considerations

4.1. 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: add ALTA to the list when and if it gets further along [I-D.draft-krose-mboned-alta]. 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.

TBD: probably should add quic also? Or maybe https is sufficient?

TBD: add a recommendation about scalability, like with DORMS, when using a unicast hash stream. CDN or other kind of fanout solution that can scale the delivery, and still generally hit the time window.

4.2. HTTPS

This document defines a new media type 'application/ambi' for use with HTTPS. URIs in the manifest-transport list with the scheme 'https' use this transport.

An HTTPS stream carrying the 'application/ambi' media type is composed of a sequence of binary AMBI manifests, sent back to back in the payload body (payload body is defined in Section 3.3 of [RFC7230]).

Complete packet digests from partially received manifests MAY be used by the receiver for authentication of data packets from the multicast channel, even if the full manifest is not yet delivered.

4.3. TLS

This document defines the new uri scheme 'ambi+tls' for use with TLS [RFC8446]. URIs in the manifest-transport list with the scheme 'ambi+tls' use this transport.

A TLS stream carrying AMBI manifests is composed of a sequence of binary AMBI manifests, transmitted back to back.

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

4.4. DTLS

This document defines the new uri scheme 'ambi+dtls' for use with DTLS [RFC6347].

Manifests transported with DTLS have the tradeoff (relative to TLS or HTTPS) that they might be lost and not retransmitted or reordered, but they will not cause head-of-line blocking or delay in processing data packets that arrived later. For some applications this is a worthwhile tradeoff.

Note that loss of a single DTLS packet can result in the loss of multiple packet digests, which can mean failure to authenticate multiple data packets.

DTLS transport for manifests supports one manifest per packet. It's OPTIONAL to provides for some redundancy in packet digests by providing overlap in the packet sequence numbers across different manifests, thereby sending some or all packet digests multiple times to avoid loss.

Future extensions might define extensions that can provide more efficient redundancy via FEC. Those future extensions will require a different URI scheme.

5. Examples

TBD: walk through some examples as soon as we have a build running.
Likely to need some touching up of the spec along the way...

6. YANG Module

6.1. Tree Diagram

The tree diagram below follows the notation defined in [RFC8340].

```
module: ietf-ambi

  augment /dorms:dorms/dorms:metadata/dorms:sender/dorms:group
            /dorms:udp-stream:
    +--rw ambi
       +--rw manifest-stream* [id]
          +--rw id                  uint32
          +--rw manifest-stream* [uri]
          |  +--rw uri    inet:uri
          +--rw hash-algorithm      iha:hash-algorithm-type
          +--rw data-hold-time?     uint32
          +--rw digest-hold-time?   uint32
          +--rw expiration?         yang:date-and-time
  augment /dorms:dorms/dorms:metadata/dorms:sender/dorms:group:
    +--rw ambi
       +--rw manifest-stream* [id]
          +--rw id                  uint32
          +--rw manifest-stream* [uri]
          |  +--rw uri    inet:uri
          +--rw hash-algorithm      iha:hash-algorithm-type
          +--rw data-hold-time?     uint32
          +--rw digest-hold-time?   uint32
          +--rw expiration?         yang:date-and-time
```

6.2. Module

```
<CODE BEGINS> file ietf-ambi@2022-03-07.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";
  }

  import ietf-yang-types {
    prefix "yang";
    reference "RFC 6991: Common YANG Data 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 2021-07-08 {
  description "Draft version.";
  reference
    "draft-ietf-mboned-ambi";
}

grouping manifest-stream-definition {
  description
      "This grouping specifies a manifest stream for
       authenticating a multicast data stream with AMBI";
  leaf id {
    type uint32;
    mandatory true;
    description
        "The Manifest ID referenced in a manifest.";
  }
  list manifest-stream {
    key uri;
    leaf uri {
      type inet:uri;
      mandatory true;
      description
          "The URI for a stream of manifests.";
    }
    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 data-hold-time {
    type uint32;
    default 2000;
    units "milliseconds";
    description
        "The number of milliseconds to hold data packets
```

```
          waiting for a corresponding digest before
          discarding";
    }
    leaf digest-hold-time {
      type uint32;
      default 10000;
      units "milliseconds";
      description
          "The number of milliseconds to hold packet
           digests waiting for a corresponding data packet
           before discarding";
    }
    leaf expiration {
      type yang:date-and-time;
      description
          "The time after which this manifest stream may
           stop providing authentication for the data stream.
           When not present or empty there is no known expiration.";
    }
  }

  augment
      "/dorms:dorms/dorms:metadata/dorms:sender/dorms:group/"+
      "dorms:udp-stream" {
    description "AMBI extensions for securing UDP multicast.";

    container ambi {
      description "UDP-layer AMBI container for DORMS extension.";
      list manifest-stream {
        key id;
        description "Manifest stream definition list.";
        uses manifest-stream-definition;
      }
    }
  }

  augment
      "/dorms:dorms/dorms:metadata/dorms:sender/dorms:group" {
    description "AMBI extensions for securing IP multicast.";

    container ambi {
      description "IP-layer AMBI container for DORMS extension.";
      list manifest-stream {
        key id;
        description "Definition of a manifest stream.";
        uses manifest-stream-definition;
      }
    }
  }
```

```
}

<CODE ENDS>
```

7. IANA Considerations

7.1. 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
```

7.2. The XML Registry

This document adds the following registration to the "ns" subregistry of the "IETF XML Registry" defined in [RFC3688], referencing this document.

```
URI: urn:ietf:params:xml:ns:yang:ietf-ambi
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
```

7.3. 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/rfc8126

TBD: comments from Amanda: The first is that the current IANA Considerations RFC is RFC 8126 rather than 5226. The other point, which you may be aware of, is that while https://www.iana.org/form/media-types provides guidance, standards-tree registrations submitted through RFCs shouldn't be submitted through that form and (unlike vendor-tree subtypes and standards-tree subtypes documented in other standards organization specs) won't need to be explicitly approved by the IESG-designated experts. Instead, the advice in RFC 6838 is that media type registrations requested by IETF-stream I-Ds be informally reviewed on the media-types@iana.org mailing list, which the IESG-designated experts participate in.

7.4. URI Schemes

7.4.1. TLS

TBD: register 'ambi+tls' as a uri scheme according to advice from: https://datatracker.ietf.org/doc/html/rfc7595

7.4.2. DTLS

TBD: register 'ambi+dtls' as a uri scheme according to advice from: https://datatracker.ietf.org/doc/html/rfc7595

8. Security Considerations

8.1. 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.

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

8.2. Attacks on Side Applications

A multicast receiver subscribes to an (S,G) and if it's a UDP application, listens on a socket with a port number for packets to arrive.

UDP applications sometimes bind to an "unspecified" address ("::" or "0.0.0.0") for a particular UDP port, which will make the appliction receive and process any packet that arrives on said port.

Forwarding multicast traffic opens a new practical attack surface against receivers that have bound sockets using the "unspecified" address and were operating behind a firewall and/or NAT. Such applications will receive traffic from the internet only after sending an outbound packet, and usually only for return packets with the reversed source and destination port and IP addresses.

Multicast subscription and routing operates at the IP layer, so when a multicst receive application subscribes to a channel, traffic with the IP addresses for that channel will start arriving. There is no selection for the UDP port at the routing layer that prevents multicast IP traffic from arriving.

When an insecure application with a vulnerability is listening to a UDP port on an unspecified address, it will receive multicast packets arriving at the device and with that destination UDP port. Although the primary problem lies in the insecure application, accepting multicast subscriptions increases the attack scope against those applications to include attackers who can inject a packet into a properly subscribed multicast stream.

It's RECOMMENDED that senders using AMBI to secure their traffic include all IP traffic that they send in their DORMS metadata information, and that firewalls using AMBI to provide secure access to multicast traffic block multicast traffic destined to unsecured UDP ports on (S,G)s that have AMBI-based security for any traffic. This mitigation prevents new forwarding of multicast traffic from providing attackers with a packet inject capability access to new attack surfaces from pre-existing insecure apps.

9. Acknowledgements

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

10. References

10.1. Normative References

[I-D.draft-ietf-mboned-dorms] Holland, J., "Discovery Of Restconf Metadata for Source-specific multicast", Work in Progress, Internet-Draft, draft-ietf-mboned-dorms-01, 31 October 2020, <https://www.ietf.org/archive/id/draft-ietf-mboned-dorms-01.txt>.

[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>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and

Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

10.2. Informative References

[CVE] MITRE, "Common Vulnerabilities and Exposures", September 1999, <https://cve.mitre.org/>.

[I-D.draft-krose-mboned-alta] Rose, K. and J. Holland, "Asymmetric Loss-Tolerant Authentication", Work in Progress, Internet-Draft, draft-krose-mboned-alta-01, 8 July 2019, <https://www.ietf.org/archive/id/draft-krose-mboned-alta-01.txt>.

[I-D.ietf-tsvwg-udp-options]
Touch, J., "Transport Options for UDP", Work in Progress, Internet-Draft, draft-ietf-tsvwg-udp-options-15, 3 March 2022, <https://www.ietf.org/archive/id/draft-ietf-tsvwg-udp-options-15.txt>.

[PathChirp]
Ribeiro, V.J., Riedi, R.H., Baraniuk, R.G., 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.

[PathRate] Dovrolis, C., Ramanathan, P., and D. Moore, "Packet dispersion techniques and a capacity estimation methodology", IEEE/ACM Transactions on Networking, Volume 12, Issue 6, pp. 963-977. , December 2004.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI

10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC4082] Perrig, A., Song, D., Canetti, R., Tygar, J. D., 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>.

[RFC6584] Roca, V., "Simple Authentication Schemes for the Asynchronous Layered Coding (ALC) and NACK-Oriented Reliable Multicast (NORM) Protocols", RFC 6584, DOI 10.17487/RFC6584, April 2012, <https://www.rfc-editor.org/info/rfc6584>.

[WEBRC] Luby, M. and V. Goyal, "Wave and Equation Based Rate Control Using Multicast Round Trip Time: Extended Report", Digital Fountain Technical Report no. DF2002-07-001 , September 2002.

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