MPLS Generic Associated Channel (G-ACh) Advertisement Protocol
draft-ietf-mpls-gach-adv-06

Abstract

   The MPLS Generic Associated Channel (G-ACh) provides an auxiliary
   logical data channel associated with a Label Switched Path (LSP), a
   pseudowire, or a section (link) over which a variety of protocols may
   flow. These protocols are commonly used to provide Operations,
   Administration, and Maintenance (OAM) mechanisms associated with the
   primary data channel. This document specifies simple procedures by
   which an endpoint of an LSP, pseudowire, or section may inform the
   other endpoints of its capabilities and configuration parameters, or
   other application-specific information. This information may then be
   used by the receiver to validate or adjust its local configuration,
   and by the network operator for diagnostic purposes.

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 http://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 August 16, 2013.

Copyright Notice

Table of Contents

1. Introduction

   The MPLS Generic Associated Channel (G-ACh) is defined and described
   in [RFC5586]. It provides an auxiliary logical data channel over
   which a variety of protocols may flow. Each such data channel is
   associated with an MPLS Label Switched Path (LSP), a pseudowire, or a
   section (link). An important use of the G-ACh and the protocols it
   supports is to provide Operations, Administration, and Maintenance
   (OAM) capabilities for the associated LSP, pseudowire, or section.
   Examples of such capabilities include Pseudowire Virtual Circuit
   Connectivity Verification (VCCV) [RFC5085], Bidirectional Forwarding
   Detection (BFD) for MPLS [RFC5884], and MPLS packet loss, delay, and
   throughput measurement [RFC6374], as well as OAM functions developed
   for the MPLS Transport Profile (MPLS-TP) [RFC5921].

   This document specifies procedures for an MPLS Label Switching Router
   (LSR) to advertise its capabilities and configuration parameters, or
   other application-specific information, to its peers over LSPs,
   pseudowires, and sections. Receivers can then make use of this
   information to validate or adjust their own configurations, and
   network operators can make use of it to diagnose faults and
   configuration inconsistencies between endpoints.

   The main principle guiding the design of the MPLS G-ACh Advertisement
   Protocol (GAP) is simplicity. The protocol provides a one-way method
   of distributing information about the sender. How this information
   is used by a given receiver is a local matter. The data elements
   distributed by the GAP are application-specific and, except for those
   associated with the GAP itself, are outside the scope of this
   document. An IANA registry is created to allow GAP applications to
   be defined as needed.

1.1. Motivation

   It is frequently useful in a network for a node to have general
   information about its adjacent nodes, i.e., those nodes to which it
   has links. At a minimum this allows a human operator or management
   application with access to the node to determine which adjacent nodes
   this node can see, which is helpful when troubleshooting connectivity
   problems. A typical example of an "adjacency awareness protocol" is
   the Link Layer Discovery Protocol [LLDP], which can provide various
   pieces of information about adjacent nodes in Ethernet networks, such
   as system name, basic functional capabilities, link speed/duplex
   settings, and maximum supported frame size. Such data is useful both
   for human diagnostics and for automated detection of configuration
   inconsistencies.

   In MPLS networks, the G-ACh provides a convenient link-layer-agnostic

means for communication between LSRs that are adjacent at the link
layer. The G-ACh advertisement protocol presented in this document
thus allows LSRs to exchange information of a similar sort to that
supported by LLDP for Ethernet links. The GAP, however, does not
depend on the specific link-layer protocol in use, and can be used to
advertise information on behalf of any MPLS application.

In networks based on the MPLS Transport Profile (MPLS-TP) [RFC5921]
that do not also support IP, the normal protocols used to determine
the Ethernet address of an adjacent MPLS node, such as the Address
Resolution Protocol [RFC0826] and IP version 6 Neighbor Discovery
[RFC4861], are not available. One possible use of the G-ACh
advertisement protocol is to discover the Ethernet MAC addresses of
MPLS-TP nodes lacking IP capability
[I-D.ietf-mpls-tp-ethernet-addressing]. However, where it is
anticipated that the only data that needs to be exchanged between
LSRs over an Ethernet link are their Ethernet addresses, then the
operator may instead choose to use LLDP for that purpose.

The applicability of the G-ACh advertisement protocol is not limited
to link-layer adjacency, either in terms of message distribution or
message content. The G-ACh exists for any MPLS LSP or pseudowire, so
GAP messages can be exchanged with remote LSP or pseudowire
endpoints. The content of GAP messages is extensible in a simple
manner, and can include any kind of information that might be useful
to MPLS LSRs connected by links, LSPs, or pseudowires. For example,
in networks that rely on the G-ACh for OAM functions, GAP messages
might be used to inform adjacent LSRs of a node's OAM capabilities
and configuration parameters.

1.2. Terminology

```
Term   Definition
-----  ---------------------------------------------
G-ACh  Generic Associated Channel
GAL    G-ACh Label
GAP    G-ACh Advertisement Protocol
LSP    Label Switched Path
OAM    Operations, Administration, and Maintenance
```

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in [RFC2119].

2. Overview

   The G-ACh Advertisement Protocol has a simple one-way mode of
   operation: a device configured to send information for a particular
   data channel (MPLS LSP, pseudowire, or section) transmits GAP
   messages over the G-ACh associated with the data channel. The
   payload of a GAP message is a collection of Type-Length-Value (TLV)
   objects, organized on a per-application basis. An IANA registry is
   created to identify specific applications. Application TLV objects
   primarily contain static data that the receiver is meant to retain
   for a period of time, but may also represent metadata or special
   processing instructions.

   Each GAP message can contain data for several applications. A sender
   may transmit a targeted update that refreshes the data for a subset
   of applications without affecting the data of other applications sent
   on a previous message.

   For example, a GAP message might be sent containing the following
   data:

      Application A: A-TLV4, A-TLV15, A-TLV9

      Application B: B-TLV1, B-TLV3

      Application C: C-TLV6,

   where the numbers are specific Type values.

   A second message might then be sent containing:

      Application B: B-TLV7, B-TLV3

   Upon receiving the second message, the receiver retains B-TLV1 from
   the first message and adds B-TLV7 to its B-database. How it handles
   the new B-TLV3 depends on the rules B has specified for this object
   type; this object could replace the old one or be combined with it in
   some way. The second message has no effect on the databases
   maintained by the receiver for Applications A and C.

   The rate at which GAP messages are transmitted is at the discretion
   of the sender, and may fluctuate over time as well as differ per
   application. Each message contains, for each application it
   describes, a lifetime that informs the receiver how long to wait
   before discarding the data for that application.

   The GAP itself provides no fragmentation and reassembly mechanisms.
   In the event that an application wishes to send larger chunks of data

via GAP messages than fall within the limits of packet size, it is
the responsibility of the application to fragment its data
accordingly.

Note that for bidirectional channels communication may optimised
through the use of a number of messages defined for transmission from
the receiver back to the sender. These are optimizations and are not
required for protocol operation.

3. **Message Format**

An Associated Channel Header (ACH) Channel Type has been allocated
for the GAP as follows:

Protocol   Channel Type
---------------------------------- --------------------
G-ACh Advertisement Protocol   0xXXXX (TBD by IANA)

For this Channel Type, the ACH SHALL NOT be followed by the ACH TLV
Header defined in [RFC5586].

Fields in this document shown as Reserved or Resv are reserved for
future specification and MUST be set to zero. All integer values for
fields defined in this document SHALL be encoded in network byte
order.

A GAP message consists of a fixed header followed by a GAP payload.
The payload of a GAP message is an Application Data Block (ADB)
consisting of one or more block elements. Each block element
contains an application identifier, a lifetime, and a series of zero
or more TLV objects for the application it describes.

Malformed GAP messages MUST be discarded by the receiver, although an
error MAY be logged.

The following figure shows the format of a G-ACh Advertisement
Protocol message, which follows the Associated Channel Header (ACH):

```
  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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |Version|      Reserved        |          Message Length        |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                       Message Identifier                      |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                         Timestamp                             |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ~                   Application Data Block (ADB)                ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
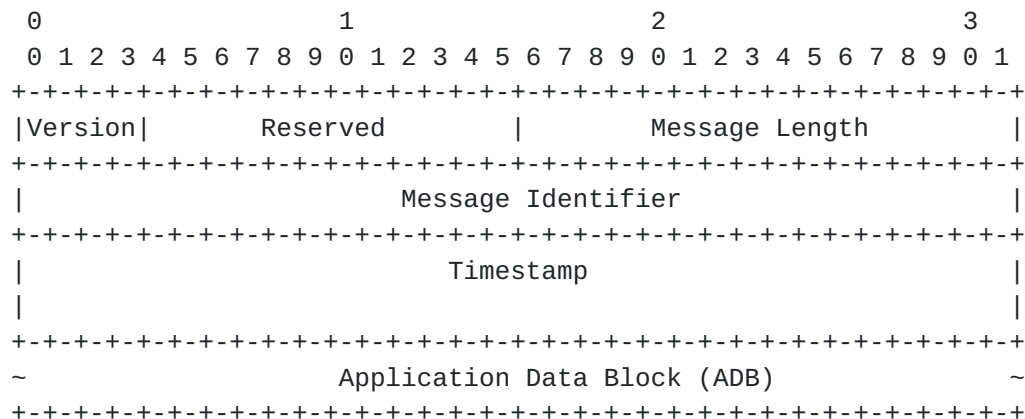
                   Figure 1: GAP Message Format

The meanings of the fields are:

   Version: Protocol version, currently set to 0

   Message Length: Size in octets of this message, i.e. of the
   portion of the packet following the Associated Channel Header

   Message Identifier (MI): Unique identifier of this message. A
   sender MUST NOT re-use an MI over a given channel until the
   message lifetime has expired. The sole purpose of this field is
   duplicate detection in the event of a message burst (Section 5.1).

   Timestamp: 64-bit Network Time Protocol (NTP) transmit timestamp,
   as specified in Section 6 of [RFC5905]

An ADB consists of one or more elements of the following format:

```
  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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |        Application ID       |          Element Length         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |            Lifetime          |            Reserved             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ~                          TLV Object                           ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ~                          TLV Object                           ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 .                                                               .
 .                                                               .
 .                                                               .
```

              Figure 2: Application Data Block Element

In this format, the Application ID identifies the application this
element describes; an IANA registry has been created to track the
values for this field. More than one block element with the same
Application ID may be present in the same ADB, and block elements
with different Application IDs may also be present in the same ADB.
The protocol rules for the mechanism, including what ADB elements are
present and which TLVs are contained in an ADB element, are to be
defined in the document that specifies the application-specific
usage.

Editors note we prefer ", are to be defined in the application's
specification."

The Element Length field specifies the total length in octets of this
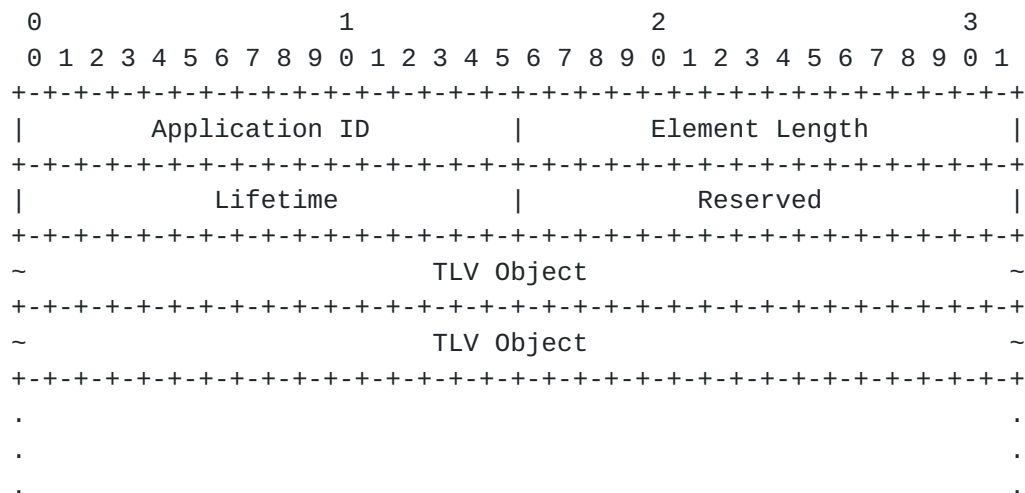block element (including the Application ID and Element Length
fields).

The Lifetime field specifies how long, in seconds, the receiver
should retain the data in this message (i.e. it specifies the
lifetime of the static data carried in the TLV set of this ADB). For
TLVs not carrying static data the Lifetime is of no significance. If
the Lifetime is zero, TLVs in this ADB are processed by the receiver
and the data associated with these TLV types is immediately marked as
expired. If the ADB contains no TLVs, the receiver expires all data
associated TLVs previously sent to this application. The scope of
the Lifetime is the source-channel-application tuple.

The remainder of the Application Data Block element consists of a
sequence of zero or more TLV objects, which are of the form:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Reserved    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             Value                             ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: TLV Object Format

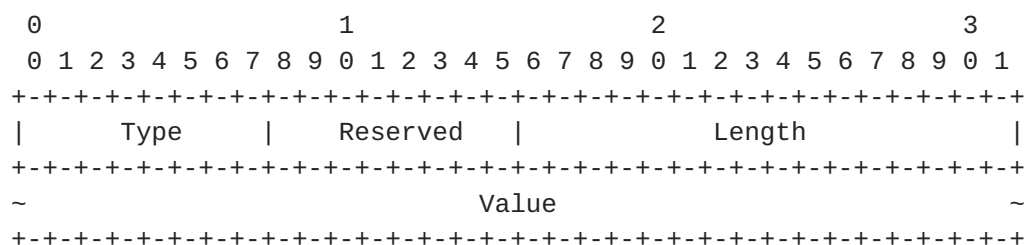The Type field identifies the TLV Object and is scoped to a specific
application; each application creates an IANA registry to track its
Type values. The Length field specifies the length in octets of the
Value field. The value field need not be padded to provide
alignment.

GAP messages do not contain a checksum. If validation of message
integrity is desired, the authentication procedures in Section 6

should be used.

4. G-ACh Advertisement Protocol TLVs

The GAP supports several TLV objects related to its own operation via
the Application ID 0x0000. These objects represent metadata and
processing instructions rather than static data that is meant to be
retained. When an ADB element for the GAP is present in a GAP
message, it MUST precede other elements.

Any application using the GAP inherits the ability to use facilities
provide by Application 0x0000.

4.1. Source Address TLV

The Source Address object identifies the sending device and possibly
the transmitting interface and the channel; it has the following
format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type=0     |    Reserved   |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Reserved           |         Address Family        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                            Address                            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
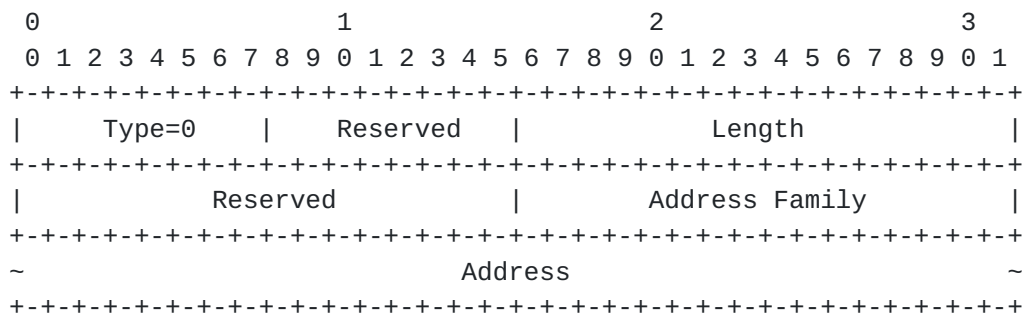
Figure 4: Source Address TLV Format

The Address Family field indicates the type of the address; it SHALL
be set to one of the assigned values in the IANA "Address Family
Numbers" registry.

In IP networks a Source Address SHOULD be included in GAP messages
and set to an IP address of the sending device; when the channel is a
link, this address SHOULD be an address of the transmitting
interface.

In non-IP MPLS-TP networks a Source Address SHOULD be included in GAP
messages and set to the endpoint identifier of the channel. The
formats of these channel identifiers SHALL be as given in Sections
3.5.1, 3.5.2, and 3.5.3 of [RFC6428] (excluding the initial Type and
Length fields shown in those sections). IANA has allocated Address
Family Numbers for these identifiers; see Section 10.2.

On multipoint channels a Source Address TLV is REQUIRED.

4.2. GAP Request TLV

This object is a request by the sender for the receiver to transmit
an immediate unicast GAP update to the sender. If the Length field
is zero, this signifies that an update for all applications is
requested. Otherwise, the Value field specifies the applications for
which an update is requested, in the form of a sequence of
Application IDs:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type=1     |    Reserved   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Application ID 1       |       Application ID 2        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Application ID N-1      |       Application ID N        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
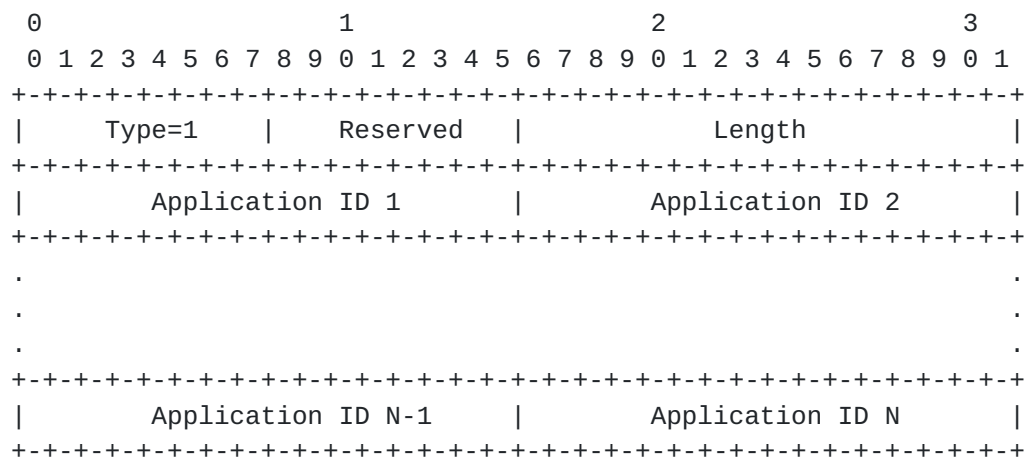
Figure 5: GAP Request TLV Format

The intent of this TLV is to request the immediate transmission of
data following a local event such as a restart rather than waiting
for a periodic update. Applications need to determine what
information is meaningful to send in response to such a request.

For an application 0x0000 GAP Request it is meaningful to respond
with the Source Address.

4.3. GAP Flush TLV

This object is an instruction to the receiver to flush the GAP data
for all applications associated with this (sender, channel) pair. It
is a null object, i.e. its Length is set to zero.

The GAP Flush instruction does not apply to data contained in the
message carrying the GAP Flush TLV object itself. Any application
data contained in the same message SHALL be processed and retained by
the receiver as usual.

The flush TLV type is 2.

4.4. GAP Suppress TLV

   This object is a request to the receiver to cease sending GAP updates
   to the transmitter over the current channel for the specified
   duration (in seconds). The receiver MAY accept and act on the
   request, MAY ignore the request, or MAY resume transmissions at any
   time according to implementation or configuration choices, and
   depending on local pragmatics. The format of this object is 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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Type=3     |    Reserved   |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Duration           |        Application ID 1       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   .                                                               .
   .                                                               .
   .                                                               .
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       Application ID N-1      |        Application ID N       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
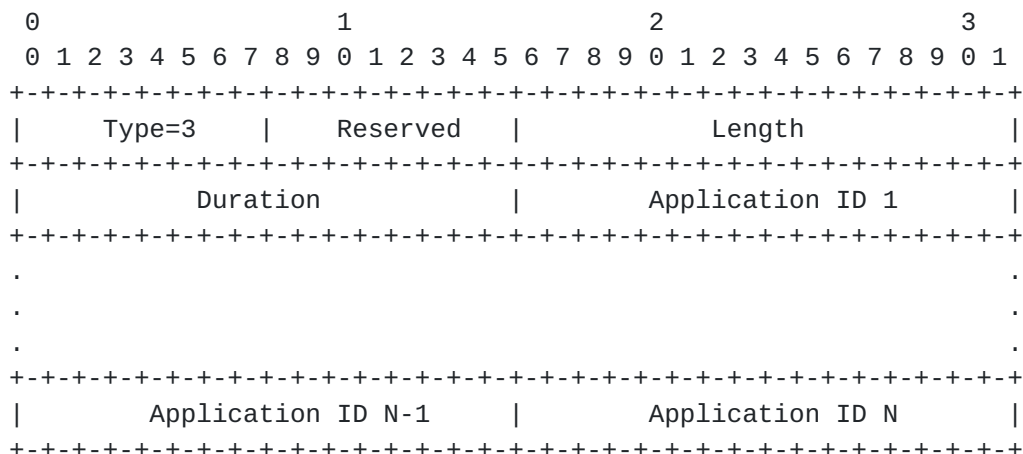
                    Figure 6: GAP Suppress TLV Format

   If the Length is set to 2, i.e. if the list of Application IDs is
   empty, then suppression of all GAP messages is requested; otherwise
   suppression of only those updates pertaining to the listed
   applications is requested. A duration of zero cancels any existing
   suppress requests for the listed applications.

   This object makes sense only for point-to-point channels or when the
   sender is receiving unicast GAP updates.

4.5. GAP Authentication TLV

   This object is used to provide authentication and integrity
   validation for a GAP message. It has the following format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type=4     |    Reserved   |              Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Reserved           |            Key ID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                      Authentication Data                      ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: GAP Authentication TLV Format

The data and procedures associated with this object are explained in
Section 6.

5. Operation

5.1. Message Transmission

G-ACh Advertisement Protocol message transmission SHALL operate on a
per-data-channel basis and be configurable by the operator
accordingly.

Because GAP message transmission may be active for many logical
channels on the same physical interface, message transmission timers
SHOULD be randomized across the channels supported by a given
interface so as to reduce the likelihood of large synchronized
message bursts.

The Message Identifier (MI) uniquely identifies this message and its
value is set at the sender's discretion.

The Timestamp field SHALL be set to the time at which this message is
transmitted.

The Lifetime field of each Application Data Block element SHALL be
set to the number of seconds the receiver is advised to retain the
data associated with this message and application.

When the transmitter wishes the data previously sent in an ADB
element to persist then it must refresh the ADB element by sending
another update. Refresh times SHOULD be set in such a way that at
least three updates will be sent prior to Lifetime expiration. For
example, if the Lifetime is set to 210 seconds, then updates should
be sent at least once every 60 seconds.

A sender may signal that previously sent data SHOULD be marked as

expired by setting the ADB element lifetime to zero as previously
described in Section 3 .

In some cases an application may desire additional reliability for
the delivery of some of its data. When this is the case, the
transmitter MAY send several (for example three) instances of the
message in succession, separated by a delay appropriate to, or
specified by, the application. For example this procedure might be
invoked when sending a flush instruction following device reset. The
expectation is that the receiver will detect duplicate messages using
the MI.

5.2. Message Reception

G-ACh Advertisement Protocol message reception SHALL operate on a
per-data-channel basis and be configurable by the operator
accordingly.

Upon receiving a G-ACh Advertisement Protocol message that contains
data for some application X, the receiver determines whether it can
interpret X-data. If it cannot, then the receiver MAY retain this
data for the number of seconds specified by the Lifetime field;
although it cannot parse this data, it may still be of use to the
operator.

If the receiver can interpret X-data, then it processes the data
objects accordingly, retaining the data associated with those that
represent static data for the number of seconds specified by the
Lifetime field. If the lifetime is zero, such data is immediately
marked as expired, and if no TLVs are specified all data associated
with previously received TLVs is marked as expired Section 3. If one
of the received TLV objects has the same Type as a previously
received TLV then the data from the new object SHALL replace the data
associated with that Type unless the X specification dictates a
different behavior.

The receiver MAY make use of the application data contained in a GAP
message to perform some level of auto-configuration, for example if
the application is an OAM protocol. The application SHOULD, however,
take care to prevent cases of oscillation resulting from each
endpoint attempting to adjust its configuration to match the other.
Any such auto-configuration based on GAP information MUST be disabled
by default.

The MI may be used to detect and discard duplicate messages.

6. **Message Authentication**

   The GAP provides a means of authenticating messages and ensuring
   their integrity. This is accomplished by attaching a GAP
   Authentication TLV and including, in the Authentication Data field,
   the output of a cryptographic hash function, the input to which is
   the message together with a secret key known only to the sender and
   receiver. Upon receipt of the message, the receiver computes the
   same hash and compares the result with the hash value in the message;
   if the hash values are not equal, the message is discarded.

   The remainder of this section gives the details of this procedure,
   which is based on the procedures for generic cryptographic
   authentication for the Intermediate System to Intermediate System
   (IS-IS) routing protocol as described in [RFC5310].

6.1. **Authentication Key Identifiers**

   An Authentication Key Identifier (Key ID) is a 16-bit tag shared by
   the sender and receiver that identifies a set of authentication
   parameters. These parameters are not sent over the wire; they are
   assumed to be associated, on each node, with the Key ID by external
   means, such as via explicit operator configuration or a separate key-
   exchange protocol. Multiple Key IDs may be active on the sending and
   receiving nodes simultaneously, in which case the sender locally
   selects a Key ID from this set to use in an outbound message. This
   capability facilitates key migration in the network.

   The parameters associated with a Key ID are:

   o Authentication Algorithm: This signifies the authentication
      algorithm to use to generate or interpret authentication data. At
      present, the following values are possible: HMAC-SHA-1, HMAC-SHA-
      224, HMAC-SHA- 256, HMAC-SHA-384, and HMAC-SHA-512.

   o Authentication Keystring: A secret string that forms the basis for
      the cryptographic key used by the Authentication Algorithm.

   Implementors SHOULD consider the use of
   [I-D.ietf-karp-crypto-key-table] for key management.

   At the time of this writing, mechanisms for dynamic key management in
   the absence of IP are not available. Key management in such
   environments therefore needs to take place via the equipment
   management system or some other out of band service. The MPLS layer
   in a network is normally isolated from direct access by users and
   thus is a relatively protected environment. Thus key turnover is a
   relatively infrequent event.

6.2. Authentication Process

   The authentication process for GAP messages is straightforward.
   First, a Key ID is associated on both the sending and receiving nodes
   with a set of authentication parameters. Following this, when the
   sender generates a GAP message, it sets the Key ID field of the GAP
   Authentication TLV accordingly. (The length of the Authentication
   Data field is also known at this point, because it is a function of
   the Authentication Algorithm.) The sender then computes a hash for
   the message as described in Section 6.3 , and fills the
   Authentication Data field of the GAP Authentication TLV with the hash
   value. The message is then sent.

   When the message is received, the receiver computes a hash for it as
   described below. The receiver compares its computed value to the
   hash value received in the Authentication Data field. If the two
   hash values are equal, authentication of the message is considered to
   have succeeded; otherwise it is considered to have failed.

   This process suffices to ensure the authenticity and integrity of
   messages, but is still vulnerable to a replay attack, in which a
   third party captures a message and sends it on to the receiver at
   some later time. The GAP message header contains a Timestamp field
   which can be used to protect against replay attacks. To achieve this
   protection, the receiver checks that the time recorded in the
   timestamp field of a received and authenticated GAP message
   corresponds to the current time, within a reasonable tolerance that
   allows for message propagation delay, and accepts or rejects the
   message accordingly. Clock corrections SHOULD be monotonic to avoid
   replay attack unless operator intervention overrides this to achieve
   a faster convergence with current time.

   If the clocks of the sender and receiver are not synchronized with
   one another, then the receiver must perform the replay check against
   its best estimate of the current time according to the sender's
   clock. The timestamps that appear in GAP messages can be used to
   infer the approximate clock offsets of senders and, while this does
   not yield high-precision clock synchronization, it suffices for
   purposes of the replay check with an appropriately chosen tolerance.

6.3. Hash Computation

   In the algorithm description below, the following nomenclature, which
   is consistent with [FIPS-198], is used:

```
   Symbol        Definition
   -------------- --------------------------------------------------------
   H             The specific hash algorithm, e.g.  SHA-256
   K             The Authentication Keystring
   Ko            The cryptographic key used with the hash algorithm
   B             The block size of H, measured in octets rather than in
                 bits.  Note that B is the internal block size, not the
                 hash size.  This is equal to 64 for SHA-1 and SHA-256,
                 and to 128 for SHA-384 and SHA-512.
   L             The length of the hash, measured in octets rather than
                 in bits
   XOR           The exclusive-or operation
   Opad          The hexadecimal value 0x5c repeated B times
   Ipad          The hexadecimal value 0x36 repeated B times
   Apad          hexadecimal value 0x878FE1F3 repeated (L/4) times
```

1. Preparation of the Key

        In this application, Ko is always L octets long.

        If the Authentication Keystring (K) is L octets long, then Ko
        is equal to K. If the Authentication Keystring (K) is more
        than L octets long, then Ko is set to H(K). If the
        Authentication Keystring (K) is less than L octets long, then
        Ko is set to the Authentication Keystring (K) with zeros
        appended to the end of the Authentication Keystring (K) such
        that Ko is L octets long.

2. First Hash

        First, the Authentication Data field is filled with the value
        Apad.

        Then, a first hash, also known as the inner hash, is computed
        as follows:

            First-Hash = H(Ko XOR Ipad || (GAP Message))

        Here the GAP Message is the portion of the packet that follows
        the Associated Channel Header.

3. Second Hash

        Then a second hash, also known as the outer hash, is computed
        as follows:

            Second-Hash = H(Ko XOR Opad || First-Hash)

4. Result

>   The resulting second hash becomes the authentication data that
>   is sent in the Authentication Data field of the GAP
>   Authentication TLV. The length of the Authentication Data
>   field is always identical to the message digest size of the
>   specific hash function H that is being used.
>
>   This also means that the use of hash functions with larger
>   output sizes will increase the size of the GAP message as
>   transmitted on the wire.

7. Link-Layer Considerations

When the GAP is used to support device discovery on a data link, GAP
messages must be sent in such a way that they can be received by
other listeners on the link without the sender first knowing the
link-layer addresses of the listeners. In short, they must be
multicast. Considerations for multicast MPLS encapsulation are
discussed in [RFC5332]. For example, Section 8 of [RFC5332]
describes how destination Ethernet MAC addresses are selected for
multicast MPLS packets. Since a GAP packet transmitted over a data
link contains just one label, the G-ACh Label (GAL) with label value
13, the correct destination Ethernet address for frames carrying GAP
packets intended for device discovery, according to these selection
procedures, is 01-00-5e-80-00-0d.

8. Managability Considerations

The data sent and received by this protocol MUST be made accessible
for inspection by network operators, and where local configuration is
updated by the received information, it MUST be clear why the
configured value has been changed. The persistence of data
advertised by this protocol is applications specific, but in general
SHOULD be persistent across restarts. Received advertisements MUST
be discarded across restarts. If the received values change, the new
values MUST be used and the change made visible to the network
operators.

All applications MUST be disabled by default and need be enabled by
the operator if required.

9. Security Considerations

G-ACh Advertisement Protocol messages contain information about the

sending device and its configuration, which is sent in cleartext over
the wire. If an unauthorized third party gains access to the MPLS
data plane or the lower network layers between the sender and
receiver, it can observe this information. In general, however, the
information contained in GAP messages is no more sensitive than that
contained in other protocol messages, such as routing updates, which
are commonly sent in cleartext. No attempt is therefore made to
guarantee confidentiality of GAP messages.

A more significant potential threat is the transmission of GAP
messages by unauthorized sources, or the unauthorized manipulation of
messages in transit; this can disrupt the information receivers hold
about legitimate senders. To protect against this threat, message
authentication procedures are specified in Section 6 of this document
that enable receivers to ensure the authenticity and integrity of GAP
messages. These procedures include the means to protect against
replay attacks, in which a third party captures a legitimate message
and "replays" it to a receiver at some later time.

10. IANA Considerations

10.1. Associated Channel Type Allocation

This document requests that IANA allocate an entry in the "Pseudowire
Associated Channel Types" registry [RFC5586] (currently located
within the "Pseudowire Name Spaces (PWE3)" registry) for the "G-ACh
Advertisement Protocol", as follows:

Value	Description	TLV Follows	Reference
XXXX(TBD)	G-ACh Advertisement Protocol	No	(this draft)

10.2. Allocation of Address Family Numbers

IANA is requested to allocate three entries from the Standards Track
range in the "Address Family Numbers" registry for MPLS-TP Section,
LSP, and Pseudowire endpoint identifiers, per Section 4.1. The
allocations are:

Number	Description	Reference
(TBD)	MPLS-TP Section Endpoint Identifier	(this draft)
(TBD)	MPLS-TP LSP Endpoint Identifier	(this draft)
(TBD)	MPLS-TP Pseudowire Endpoint Identifier	(this draft)

10.3. Creation of G-ACh Advertisement Protocol Application Registry

   This document requests that IANA create a new registry, "G-ACh
   Advertisement Protocol Applications" in the "Pseudowire Name Spaces
   (PWE3)" registry, with fields and initial allocations as follows:

   Application ID Description   Reference
   -------------- ---------------------------- ------------
   0x0000   G-ACh Advertisement Protocol (this draft)

   The range of the Application ID field is 0x0000 - 0xFFFF.

   The allocation policy for this registry is IETF Review.

10.4. Creation of G-ACh Advertisement Protocol TLV Registry

   This document requests that IANA create a new registry, "G-ACh
   Advertisement Protocol: GAP TLV Objects (Application ID 0)" in the
   "Pseudowire Name Spaces (PWE3)" registry, with fields and initial
   allocations as follows:

   Type Name   Type ID Reference
   ------------------ ------- ------------
   Source Address   0   (this draft)
   GAP Request   1   (this draft)
   GAP Flush   2   (this draft)
   GAP Suppress   3   (this draft)
   GAP Authentication 4   (this draft)

   The range of the Type ID field is 0 - 255.

   The allocation policy for this registry is IETF Review.

11. Acknowledgements

   We thank Adrian Farrel for his valuable review comments on this
   document.

12. References

12.1. Normative References

   [FIPS-198]
             US National Institute of Standards and Technology, "The
             Keyed-Hash Message Authentication Code (HMAC)", FIPS PUB
             198, March 2002.

[I-D.ietf-karp-crypto-key-table]
          Housley, R., Polk, T., Hartman, S., and D. Zhang,
          "Database of Long-Lived Symmetric Cryptographic Keys",
          draft-ietf-karp-crypto-key-table-04 (work in progress),
          October 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
          Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5332] Eckert, T., Rosen, E., Aggarwal, R., and Y. Rekhter, "MPLS
          Multicast Encapsulations", RFC 5332, August 2008.

[RFC5586] Bocci, M., Vigoureux, M., and S. Bryant, "MPLS Generic
          Associated Channel", RFC 5586, June 2009.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network
          Time Protocol Version 4: Protocol and Algorithms
          Specification", RFC 5905, June 2010.

[RFC6428] Allan, D., Swallow Ed. , G., and J. Drake Ed. , "Proactive
          Connectivity Verification, Continuity Check, and Remote
          Defect Indication for the MPLS Transport Profile",
          RFC 6428, November 2011.

12.2. Informative References

[I-D.ietf-mpls-tp-ethernet-addressing]
          Frost, D., Bryant, S., and M. Bocci, "MPLS-TP Next-Hop
          Ethernet Addressing",
          draft-ietf-mpls-tp-ethernet-addressing-05 (work in
          progress), February 2013.

[LLDP]   IEEE, "Station and Media Access Control Connectivity
          Discovery (802.1AB)", September 2009.

[RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or
          converting network protocol addresses to 48.bit Ethernet
          address for transmission on Ethernet hardware", STD 37,
          RFC 826, November 1982.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
          "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
          September 2007.

[RFC5085] Nadeau, T. and C. Pignataro, "Pseudowire Virtual Circuit
          Connectivity Verification (VCCV): A Control Channel for
          Pseudowires", RFC 5085, December 2007.

   [RFC5310]   Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R.,
               and M. Fanto, "IS-IS Generic Cryptographic
               Authentication", RFC 5310, February 2009.

   [RFC5884]   Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow,
               "Bidirectional Forwarding Detection (BFD) for MPLS Label
               Switched Paths (LSPs)", RFC 5884, June 2010.

   [RFC5921]   Bocci, M., Bryant, S., Frost, D., Levrau, L., and L.
               Berger, "A Framework for MPLS in Transport Networks",
               RFC 5921, July 2010.

   [RFC6374]   Frost, D. and S. Bryant, "Packet Loss and Delay
               Measurement for MPLS Networks", RFC 6374, September 2011.

Authors' Addresses

   Dan Frost
   Cisco Systems

   Email: danfrost@cisco.com

   Stewart Bryant
   Cisco Systems

   Email: stbryant@cisco.com

   Matthew Bocci
   Alcatel-Lucent

   Email: matthew.bocci@alcatel-lucent.com