Network Working Group   D. Katz
Internet Draft   Juniper Networks
Intended status: Proposed Standard   D. Ward
                                                                 Cisco

Expires: January, 2013   June 1, 2012

                      BFD for Multipoint Networks
                     draft-ietf-bfd-multipoint-01.txt

Status of this Memo

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

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that other
   groups may also distribute working documents as Internet-Drafts.

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

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

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

Abstract

This document describes extensions to the Bidirectional Forwarding
Detection (BFD) protocol for its use in multipoint and multicast
networks. Comments on this draft should be directed to
rtg-bfd@ietf.org.

Conventions used in this document

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 RFC-2119 [KEYWORDS].

Table of Contents

1. Introduction

The Bidirectional Forwarding Detection protocol [BFD] specifies a
method for verifying unicast connectivity between a pair of systems.
This document defines a method for using BFD to provide verification
of multipoint or multicast connectivity between a multipoint sender
(the "head") and a set of one or more multipoint receivers (the
"tails").

As multipoint transmissions are inherently unidirectional, this
mechanism purports only to verify this unidirectional connectivity.
Although this seems in conflict with the "Bidirectional" in BFD, it
is a natural fit for that protocol.

This application of BFD allows for the tails to detect a lack of
connectivity from the head. As an option, the tail may unreliably
notify the head of the lack of multipoint connectivity. As a further
option, this notification can be made reliable. Notification to the
head can be enabled for all tails, or for only a subset of the tails.

Multipoint BFD verifies only the head-to-tail connectivity over the
multipoint path. Although it may use unicast paths in both
directions, Multipoint BFD does not verify those paths (and in fact
it is preferable if unicast paths share as little fate with the
multipoint path as is feasible.)

Virtually all options and timing parameters are controlled by the
head. This is particularly important if head notifications are
enabled, since there are obvious scaling concerns in that case.

Throughout this document, the term "multipoint" is defined as a

mechanism by which one or more systems receive packets sent by a single sender. This specifically includes such things as IP multicast and point-to-multipoint MPLS.

This document effectively modifies and adds to the base BFD specification. It is the intention of the authors to fold these extensions into the base specification at the appropriate time.

2. Goals

The primary goal of this mechanism is to allow tails to rapidly detect the fact that multipoint connectivity from the head has failed. An optional goal is for the head to reasonably rapidly have knowledge of tails that have lost connectivity from the head.

Since scaling is a primary concern (particularly state implosion toward the head), it is a goal that the head be in control of all timing aspects of the mechanism, and that BFD packets from the tails to the head not be synchronized.

Another goal is for the mechanism to work on any multicast or multipoint medium.

A further goal is to support multiple, overlapping multipoint paths, as well as multipoint paths with multiple heads, and to allow point-to-point BFD sessions to operate simultaneously among the systems participating in Multipoint BFD.

A final goal is to integrate multipoint operation into the base specification in such a way as to make it relatively easy to support both multipoint and point-to-point operation in a single implementation.

It is a non-goal for this protocol to verify point-to-point connectivity between the head and any tails. This can be done independently (and with no penalty in protocol overhead) by using point-to-point BFD.

[3](). Overview

   The heart of this protocol is the periodic transmission of BFD
   Control packets along a multipoint path, from the head to all tails
   on the tree. The contents of the BFD packets provide the means for
   the tails to calculate the detection time for path failure. If no
   BFD Control packets are received by a tail for a detection time, the
   tail declares the path to have failed. For some applications this is
   the only mechanism necessary; the head can remain ignorant of the
   tails. In this mode, the tails never send any BFD traffic to the
   head.

   If the head wishes to be alerted to the tails' connectivity (or lack
   thereof), there are a number of options. First, if all that is
   needed is an unreliable failure notification, the head can direct the
   tails to transmit unicast BFD Control packets back to the head when
   the path fails.

   If the head wishes to know the identity of the tails on the
   multipoint path, it may solicit membership by sending a multipoint
   BFD Control packet with the Poll (P) bit set, which will induce the
   tails to return a unicast BFD Control packet with the Final (F) bit
   set. The head can then create BFD session state for each of the
   tails that have multipoint connectivity. If the head sends such a
   packet on occasion, it can keep track of which tails answer, thus
   providing a somewhat reliable mechanism for detecting which tails
   fail to respond (implying a loss of multipoint connectivity.)

   If the head wishes a reliable indication of the tails' connectivity,
   it may do all of the above, but if it detects that a tail did not
   answer the previous multipoint poll, it may initiate a Demand mode
   Poll Sequence as a unicast to the tail. This covers the case where
   either the multipoint poll or the single reply thereto is lost in
   transit. If desired, the head may Poll one or more tails proactively
   to track the tails' connectivity.

   If some tails are more equal than others, in the sense that the head
   needs to detect the lack of multipoint connectivity to a subset of
   tails at a different rate, the head may transmit unicast BFD Polls to
   that subset of tails. In this case, the timing may be independent on
   a tail-by-tail basis.

   Individual tails may be configured so that they never send BFD
   control packets to the head, even when the head wishes notification
   of path failure from the tail. Such tails will never be known to the
   head, but will still be able to detect multipoint path failures from
   the head.

Although this document describes a single head and a set of tails
spanned by a single multipoint path, the protocol is capable of
supporting (and discriminating between) more than one multipoint path
at both heads and tails. Furthermore, the same head and tail may
share multiple multipoint paths, and a multipoint path may have
multiple heads.

4. Protocol Details

This section describes the operation of Multipoint BFD in detail.

4.1. Multipoint BFD Control Packets

Multipoint BFD Control packets (packets sent by the head over a
multipoint path) are explicitly marked as such, via the setting of
the M bit (added to the latest revision of the BFD base
specification. This means that Multipoint BFD does not depend on the
recipient of a packet to know whether the packet was received over a
multipoint path. This can be useful in scenarios where this
information may not be available to the recipient.

4.2. Session Model

Multipoint BFD is modeled as a set of sessions of different types.
The elements of procedure differ slightly for each type.

Point-to-point sessions, as described in [BFD], are of type
PointToPoint.

The head has a session of type MultipointHead that is bound to a
multipoint path. Multipoint BFD Control packets are sent by this
session over the multipoint path, and no BFD Control packets are
received by it.

If the head is keeping track of some or all of the tails, it has a
session of type MultipointClient per tail that it cares about. All
of the MultipointClient sessions for tails on a particular particular
multipoint path are grouped with the MultipointHead session to which
the clients are listening. A BFD Poll Sequence may be sent over such
a session to a tail if the head wishes to verify connectivity. These
sessions receive any BFD Control packets sent by the tails, and never
transmit periodic BFD Control packets other than Poll Sequences
(since periodic transmission is always done by the MultipointHead
session.)

Each tail has a session of type MultipointTail associated with a
multipoint path. These sessions receive BFD Control packets from the
head, both as multipoint packets (the MultipointHead session) and as
unicast packets (the MultipointClient session, if it exists.) Any
BFD Control packets sent to the head are sent over this session.

4.3. Session Failure Semantics

The semantics of session failure are subtle enough to warrant further
explanation.

MultipointHead sessions cannot fail (since they are controlled
administratively.)

If a MultipointTail session fails, it means that the tail definitely
has lost contact with the head (or the head has been administratively
disabled) and the tail should take appropriate action.

If a MultipointClient session receives a BFD Control packet from the
tail with state Down or AdminDown, the head reliably knows that the
tail has lost multipoint connectivity. If the Detection Time expires
on a MultipointClient session, it is ambiguous as to whether the
multipoint connectivity failed or whether there was a unicast path
problem in one direction or the other, so the head does not reliably
know the tail state.

4.4. State Variables

Multipoint BFD introduces some new state variables, and modifies the
usage of a few existing ones.

4.4.1. New State Variables

A number of state variables are added to the base specification in
support of Multipoint BFD.

   bfd.SessionType

      The type of this session. Allowable values are:

         PointToPoint: Classic point-to-point BFD.

         MultipointHead: A session on the head responsible for the
         periodic transmission of multipoint BFD Control packets

along the multipoint path.

MultipointClient: A session on the head that tracks the
state of an individual tail, when desirable.

MultipointTail: A multipoint session on a tail.

This variable MUST be initialized to the appropriate type when
the session is created, according to the rules in [section 4.16](#).

bfd.SilentTail

If 1, a tail will never transmit any BFD Control packets to the
head under any circumstances. If 0, a tail may send packets to
the head according to other parts of this specification. This
allows tails to be provisioned to always be silent, even when
the head is soliciting traffic from the tails. This can be
useful, for example, in deployments of a large number of tails
when the head wishes to track the state of a subset of them.
This variable MUST be initialized based on configuration.

This variable is only pertinent when bfd.SessionType is
MultipointTail.

bfd.ReportTailDown

Set to 1 if the head wishes tails to notify the head, via
periodic BFD Control packets, when they see the BFD session
fail. If 0, the tail will never send periodic BFD Control
packets, and the head will not be notified of session failures
by the tails. This variable MUST be initialized based on
configuration.

This variable is only pertinent when bfd.SessionType is
MultipointHead or MultipointClient.

bfd.UnicastRcvd

Set to 1 if a tail receives a unicast BFD Control packet from
the head. This variable MUST be set to zero if the session
transitions from Up state to some other state.

This variable MUST be initialized to zero.

This variable is only pertinent when Bfd.SessionType is

        MultipointTail.

4.4.2. State Variable Initialization and Maintenance

   Some state variables defined in section 6.8.1 of the BFD base
   specification need to be initialized or manipulated differently
   depending on the session type.

      bfd.LocalDiscr

         For session type MultipointClient, this variable MUST always
         match the value of bfd.LocalDiscr in the associated
         MultipointHead session.

      bfd.DesiredMinTxInterval

         For session type MultipointClient, this variable MUST always
         match the value of bfd.DesiredMinTxInterval in the associated
         MultipointHead session.

      bfd.RequiredMinRxInterval

         This variable MUST always be 0 for session type MultipointHead
         if bfd.ReportTailDown is 0.

         It should be noted that for sessions of type MultipointTail,
         this variable only affects the rate of unicast Polls sent by
         the head; the rate of multipoint packets is necessarily
         unaffected by it.

      bfd.DemandMode

         This variable MUST be initialized to 1 for session types
         MultipointHead and MultipointClient, and MUST be initialized to
         0 for session type MultipointTail.

      bfd.DetectMult

         For session type MultipointClient, this variable MUST always
         match the value of bfd.DetectMult in the associated
         MultipointHead session.

4.5. Controlling Multipoint BFD Options

The state variables defined above are used to choose which
operational options are active.

The most basic form of operation, in which BFD Control packets flow
only from the head and no tracking is desired of tail state at the
head, is accomplished by setting bfd.ReportTailDown to 0 in the
MultipointHead session.

If the head wishes to know the identity of the tails, it sends
multipoint Polls as needed. Previously known tails that don't
respond to the Polls will be detected.

If the head wishes to be notified by the tails when they lose
connectivity, it sets bfd.ReportTailDown to 1 in either the
MultipointHead session (if such notification is desired from all
tails) or in the MultipointClient session (if notification is desired
from a particular tail.) Note that the setting of this variable in a
MultipointClient session for a particular tail overrides the setting
in the MultipointHead session.

If the head wishes to verify the state of a tail on an ongoing basis,
it sends a Poll Sequence from the MultipointClient session associated
with that tail as needed.

If the head wants to more quickly be alerted to a session failure
from a particular tail, it sends a BFD Control packet from the
MultipointClient session associated with that tail. This has the
effect of eliminating the initial delay that the tail would otherwise
insert prior to transmission of the packet.

If a tail wishes to operate silently (sending no BFD Control packets
to the head) it sets bfd.SilentTail to 1 in the MultipointTail
session. This allows a tail to be silent independent of the settings
on the head.

4.6. State Machine

The BFD state machine works slightly differently in the multipoint
application. In particular, since there is a many-to-one mapping,
three-way handshakes for session establishment and teardown are
neither possible nor appropriate. As such there is no Init state.

The following diagram provides an overview of the state machine for
session types MultipointClient and MultipointTail. The notation on
each arc represents the state of the remote system (as received in
the State field in the BFD Control packet) or indicates the
expiration of the Detection Timer.

```
                              DOWN, ADMIN DOWN,
                 +------+    TIMER                +------+
           +----|        |<--------------------|      |----+
      DOWN,|    | DOWN |                        |  UP  |    |UP
 ADMIN DOWN,+--->|      |-------------------->|      |<---+
      TIMER      +------+          UP          +------+
```
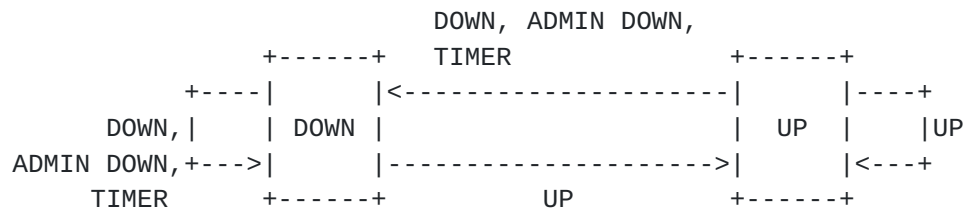
Sessions of type MultipointHead never receive packets and have no
Detection Timer, and as such all state transitions are
administratively driven.

4.7. Session Establishment

Unlike Point-to-point BFD, Multipoint BFD provides a form of
discovery mechanism for tails to discover the head, and vice versa.
The minimum amount of a priori information required both on the head
and tails is the binding to the multipoint path over which BFD is
running. The head transmits Multipoint BFD packets on that tree, and
the tails listen for BFD packets on that tree. All other information
MAY be determined dynamically.

A session of type MultipointHead is created for each multipoint path
over which the head wishes to run BFD. This session runs in the
Active role. Except when terminating BFD service, this session is
always in state Up and always operates in Demand mode. No received
packets are ever demultiplexed to the MultipointHead session. In
this sense it is a degenerate form of a session.

Sessions on the tail MAY be established dynamically, based on the
receipt of a Multipoint BFD Control packet from the head, and are of
type MultipointTail. Tail sessions always take the Passive role.

If BFD Control packets are received at the head, they are
demultiplexed to sessions of type MultipointClient, which represent
the set of tails that the head is interested in tracking. These
sessions will typically also be established dynamically based on the
receipt of BFD Control packets. The head has broad latitude in
choosing which tails to track, if any, without affecting the basic
operation of the protocol. The head directly controls whether or not
tails are allowed to send BFD Control packets back to the head.

4.8. Discriminators and Packet Demultiplexing

The use of Discriminators is somewhat different in Multipoint BFD
than in Point-to-point BFD.

The head sends Multipoint BFD Control packets over the MultipointHead
session with My Discr set to a value bound to the multipoint path,
and with Your Discr set to zero. The tails MUST demultiplex these
packets based on a combination of the source address and My Discr,
which together uniquely identify the head and the multipoint path.

When the tails send BFD Control packets to the head from the
MultipointTail session, the contents of Your Discr (the discriminator
received from the head) will not be sufficient for the head to
demultiplex the packet, since the same value will be received from
all tails on the multicast tree. In this case, the head MUST
demultiplex packets based on the source address and the value of Your
Discr, which together uniquely identify the tail and the multipoint
path.

When the head sends unicast BFD Control packets to a tail from a
MultipointClient session, the value of Your Discr will be valid, and
the tail MUST demultiplex the packet based solely on Your Discr.

Note that, unlike PointToPoint sessions, the discriminator values on
all multipoint session types MUST NOT be changed during the life of a
session. This is a side effect of the more complex demultiplexing
scheme.

4.9. Controlling Tail Packet Transmission

As the fan-in from the tails to the head may be very large, it is
critical that the flow of BFD Control packets from the tails is
controlled.

The head always operates in Demand mode. This means that no tail
will send an asynchronous BFD Control packet as long as the session
is Up.

The value of Required Min Rx Interval received by a tail in a unicast
BFD Control packet, if any, always takes precedence over the value
received in Multipoint BFD Control packets. This allows the packet
rate from individual tails to be controlled separately as desired by
sending a BFD Control packet from the corresponding MultipointClient
session. This also eliminates the random delay prior to transmission
from the tail that would otherwise be inserted, reducing the latency
of reporting a failure to the head.

   If the head wishes to suppress traffic from the tails when they
   detect a session failure, it MAY set bfd.RequiredMinRxInterval to
   zero, which is a reserved value that indicates that the sender wishes
   to receive no periodic traffic. This can be set in the
   MultipointHead session (suppressing traffic from all tails) or it can
   be set in a MultipointClient session (suppressing traffic from only a
   single tail.)

   Any tail may be provisioned to never send *any* BFD Control packets
   to the head by setting bfd.SilentTail to 1. This provides a
   mechanism by which only a subset of tails report their session status
   to the head.

[4.10](). Bringing Up and Shutting Down Multipoint BFD Service

   Because there is no three-way handshake in Multipoint BFD, a newly
   started head (that does not have any previous state information
   available) SHOULD start with bfd.SessionState set to Down and with
   bfd.RequiredMinRxInterval set to zero in the MultipointHead session.
   The session SHOULD remain in this state for a time equal to
   (bfd.DesiredMinTxInterval * bfd.DetectMult). This will ensure that
   all MultipointTail sessions are reset (so long as the restarted head
   is using the same or larger value of bfd.DesiredMinTxInterval than it
   did previously.)

   Multipoint BFD service is brought up by administratively setting
   bfd.SessionState to Up in the MultipointHead session.

   A head may wish to shut down its BFD service in a controlled fashion.
   This is desirable because the tails need not wait a detection time
   prior to declaring the multipoint session to be down (and taking
   whatever action is necessary in that case.)

   To shut down a multipoint session a head MUST administratively set
   bfd.SessionState in the MultipointHead session to either Down or
   AdminDown and SHOULD set bfd.RequiredMinRxInterval to zero (to keep
   the tails from sending any BFD Control packets back.) The session
   SHOULD send BFD Control packets in this state for a period equal to
   (bfd.DesiredMinTxInterval * bfd.DetectMult). The tail SHOULD destroy
   all MultipointClient sessions associated with the MultipointHead
   session.

   The semantic difference between Down and AdminDown state is for
   further discussion.

[4.11](). Soliciting the Tails

   If the head wishes to know the identities of the tails, the
   MultipointHead session MAY send a BFD Control packet as specified in
   [section 4.16.3](), with the Poll (P) bit set to 1. This will cause all
   of the tails to reply with a unicast BFD Control Packet, randomized
   across one packet interval.

   The decision as to when to send a multipoint Poll is outside the
   scope of this specification. However, it must never be sent more
   often than the regular multipoint BFD Control packet. Since the tail
   will treat a multipoint Poll like any other multipoint BFD Control
   packet, Polls may be sent in lieu of non-Poll packets.

   Soliciting the tails also starts the Detection Timer for each
   associated MultipointClient session, which will cause those sessions
   to time out if the associated tails do not respond.

   Note that for this mechanism to work properly, the Detection Time
   (which is equal to bfd.DesiredMinTxInterval) MUST be greater than the
   round trip time of BFD Control packets from the head to the tail (via
   the multipoint path) and back (via a unicast path.) See [section 4.14]()
   for more details.

[4.12](). Verifying Connectivity to Specific Tails

   If the head wishes to verify connectivity to a specific tail, the
   corresponding MultipointClient session MAY send a BFD Poll Sequence
   to said tail. This might be done in reaction to the expiration of
   the Detection Timer (the tail didn't respond to a multipoint Poll),
   or it might be done on a proactive basis.

   The interval between transmitted packets in the Poll Sequence MUST be
   calculated as specified in the base specification (the greater of
   bfd.DesiredMinTxInterval and bfd.RemoteMinRxInterval.)

   The value transmitted in Required Min RX Interval will be used by the
   tail (rather than the value received in any multipoint packet) when
   it transmits BFD Control packets to the head notifying it of a
   session failure, and the transmitted packets will not be delayed.
   This value can potentially be set much lower than in the multipoint
   case, in order to speed up notification to the head, since the value
   will be used only by the single tail. This value (and the lack of
   delay) are "sticky", in that once the tail receives it, it will
   continue to use it indefinitely. Therefore, if the head no longer
   wishes to single out the tail, it SHOULD reset the timer to the
   default by sending a Poll Sequence with the same value of Required

Min Rx Interval as is carried in the multipoint packets, or it MAY
reset the tail session by sending a Poll Sequence with state
AdminDown (after the completion of which the session will come back
up.)

Note that a failure of the head to receive a response to a Poll
Sequence does not necessarily mean that the tail has lost multipoint
connectivity, though a reply to a Poll Sequence does reliably
indicate connectivity or lack thereof (by virtue of the tail's state
not being Up in the BFD Control packet.)

[4.13](). Timer Manipulation

Because of the one-to-many mapping, a session of type MultipointHead
SHOULD NOT initiate a Poll Sequence in conjunction with timer value
changes. As such, such a session cannot wait for a Final before
increasing the transmit interval; such a session SHOULD send
bfd.DetectMult packets at the old transmit interval before using the
higher value in order to avoid false detection timeouts at the tails.

Since MultipointHead sessions do not calculate detection times, the
value of bfd.RequiredMinRxInterval may be changed at any time.

[4.14](). Detection Times

Multipoint BFD is inherently asymmetric. As such, each session type
has a different approach to detection times.

Since the MultipointHead session never receives packets, it does not
calculate a detection time.

MultipointClient sessions at the head are always in Demand mode, and
as such only care about detection time in two cases. First, if a
Poll Sequence is being sent on a MultipointClient session, the
detection time on this session is calculated according to the base
specification, that is, the transmission interval multiplied by
bfd.DetectMult. Second, when a multipoint Poll is sent to solicit
tail replies, the detection time on all associated MultipointClient
sessions that aren't currently sending Poll Sequences is set to a
value greater than or equal to bfd.RequiredMinRxInterval (one packet
time.) This value can be made arbitrarily large in order to ensure
that the detection time is greater than the BFD round trip time
between the head and the tail with no ill effects, other than
delaying the detection of unresponsive tails. Note that a detection
time expiration on a MultipointClient session at the head, while
indicating a BFD session failure, cannot be construed to mean that

   the tail is not hearing multipoint packets from the head.

   MultipointTail sessions cannot influence the transmission rate of the
   MultipointHead session using the Required Min Rx Interval field
   because of its one-to-many nature. As such, the Detection Time
   calculation for a MultipointTail session does not use
   bfd.RequiredMinRxInterval in the calculation. The detection time is
   calculated as the product of the last received values of Desired Min
   TX Interval and Detect Mult.

   The value of bfd.DetectMult may be changed at any time on any session
   type.

4.15. State Maintenance for Down/AdminDown Sessions

   The length of time session state is kept after the session goes down
   determines how long the session will continue to send BFD Control
   packets (since no packets can be sent after the session is
   destroyed.)

4.15.1. MultipointHead Sessions

   When a MultipointHead session transitions to states Down or
   AdminDown, the state SHOULD be maintained for a period equal to
   (bfd.DesiredMinTxInterval * bfd.DetectMult) to ensure that the tails
   more quickly detect the session going down (by continuing to transmit
   BFD Control packets with the new state.)

4.15.2. MultipointTail Sessions

   If bfd.SilentTail is 1, or bfd.RemoteMinRxInterval is zero,
   MultipointTail sessions MAY be destroyed immediately upon leaving Up
   state, since they will transmit no further packets.

   Otherwise, MultipointTail sessions MUST be maintained as long as BFD
   Control packets are being received by it (which by definition will
   indicate that the head is not Up.)

   MultipointTail sessions MUST be maintained after a Detection Time
   expiration for at least the longer of an additional Detection Time
   and the transmission of the first (delayed) BFD Control packet to the
   head. The state MAY be maintained longer than this, but the session
   MUST NOT transmit periodic BFD Control packets for a period longer
   than the negotiated transmit interval multiplied by bfd.DetectMult;
   after this time either the session MUST be destroyed or

bfd.RemoteMinRxInterval MUST be set to zero to suppress packet
transmission.

[4.15.3]. MultipointClient Sessions

If the MultipointHead session is going down (which only happens
administratively), all associated MultipointClient sessions SHOULD be
destroyed as they are superfluous.

If a MultipointClient session goes down due to the receipt of an
unsolicited BFD Control packet from the tail with state Down or
AdminDown (not in response to a Poll), and tail connectivity
verification is not being done, the session MAY be destroyed. If
verification is desired, the session SHOULD send a Poll Sequence and
the session SHOULD be maintained.

If the tail replies to a Poll Sequence with state Down or AdminDown,
it means that the tail session is definitely down. In this case, the
session MAY be destroyed.

If the Detection Time expires on a MultipointClient session (meaning
that the tail did not reply to a Poll Sequence) the session MAY be
destroyed.

[4.16]. Base Specification Text Replacement

The following sections are meant to replace the corresponding
sections in the base specification.

[4.16.1]. Reception of BFD Control Packets

The following procedure replaces section 6.8.6 of [BFD].

When a BFD Control packet is received, the following procedure MUST
be followed, in the order specified. If the packet is discarded
according to these rules, processing of the packet MUST cease at that
point.

   If the version number is not correct (1), the packet MUST be
   discarded.

   If the Length field is less than the minimum correct value (24 if
   the A bit is clear, or 26 if the A bit is set), the packet MUST be
   discarded.

If the Length field is greater than the payload of the
encapsulating protocol, the packet MUST be discarded.

If the Detect Mult field is zero, the packet MUST be discarded.

If the My Discriminator field is zero, the packet MUST be
discarded.

Demultiplex the packet to a session according to section 4.16.2
below. The result is either a session of the proper type, or the
packet is discarded (and packet processing MUST cease.)

If the A bit is set and no authentication is in use (bfd.AuthType
is zero), the packet MUST be discarded.

If the A bit is clear and authentication is in use (bfd.AuthType
is nonzero), the packet MUST be discarded.

If the A bit is set, the packet MUST be authenticated under the
rules of section 6.7, based on the authentication type in use
(bfd.AuthType.) This may cause the packet to be discarded.

Set bfd.RemoteDiscr to the value of My Discriminator.

Set bfd.RemoteState to the value of the State (Sta) field.

Set bfd.RemoteDemandMode to the value of the Demand (D) bit.

If bfd.SessionType is MultipointTail

   If bfd.UnicastRcvd is 0 or the M bit is clear, set
   bfd.RemoteMinRxInterval to the value of Required Min RX
   Interval.

   If the M bit is clear, set bfd.UnicastRcvd to 1.

Else (not MultipointTail)

   Set bfd.RemoteMinRxInterval to the value of Required Min RX
   Interval.

If the Required Min Echo RX Interval field is zero, the
transmission of Echo packets, if any, MUST cease.

If a Poll Sequence is being transmitted by the local system and
the Final (F) bit in the received packet is set, the Poll Sequence
MUST be terminated.

If bfd.SessionType is PointToPoint or MultipointClient, update the
transmit interval as described in [BFD] section 6.8.2.

If bfd.SessionType is PointToPoint, update the Detection Time as
described in [BFD] section 6.8.4. Otherwise, update the Detection
Time as described in section 4.14 above.

If bfd.SessionState is AdminDown
   Discard the packet

If received state is AdminDown
   If bfd.SessionState is not Down
      Set bfd.LocalDiag to 3 (Neighbor signaled session down)
      Set bfd.SessionState to Down

Else

   If bfd.SessionState is Down
      If bfd.SessionType is PointToPoint
         If received State is Down
            Set bfd.SessionState to Init
         Else if received State is Init
            Set bfd.SessionState to Up

      Else (bfd.SessionType is not PointToPoint)
         If received State is Up
            Set bfd.SessionState to Up

   Else if bfd.SessionState is Init
      If received State is Init or Up
         Set bfd.SessionState to Up

   Else (bfd.SessionState is Up)
      If received State is Down
         Set bfd.LocalDiag to 3 (Neighbor signaled session down)
         Set bfd.SessionState to Down

Check to see if Demand mode should become active or not (see [BFD]
section 6.6).

If bfd.RemoteDemandMode is 1, bfd.SessionState is Up, and
bfd.RemoteSessionState is Up, Demand mode is active on the remote
system and the local system MUST cease the periodic transmission
of BFD Control packets (see section 4.16.3.)

If bfd.RemoteDemandMode is 0, or bfd.SessionState is not Up, or
bfd.RemoteSessionState is not Up, Demand mode is not active on the
remote system and the local system MUST send periodic BFD Control

packets (see section 4.16.3.)

If the Poll (P) bit is set, and bfd.SilentTail is zero, send a BFD
Control packet to the remote system with the Poll (P) bit clear,
and the Final (F) bit set (see section 4.16.3.)

If the packet was not discarded, it has been received for purposes
of the Detection Time expiration rules in [BFD] section 6.8.4.

4.16.2. Demultiplexing BFD Control Packets

This section is part of the replacement for [BFD] section 6.8.6,
separated for clarity.

If the Multipoint (M) bit is set

If the Your Discriminator field is nonzero, the packet MUST be
discarded.

Select a session based on the source address and the My
Discriminator field. If a session is found, and
bfd.SessionType is not MultipointTail, the packet MUST be
discarded. If a session is not found, a new session of type
MultipointTail MAY be created, or the packet MAY be discarded.
This choice is outside the scope of this specification.

Else (Multipoint bit is clear)

If the Your Discriminator field is nonzero

Select a session based on the value of Your Discriminator.
If no session is found, the packet MUST be discarded.

If bfd.SessionType is MulticastHead

Select a session based on the source address and the
value of Your Discriminator. If no session is found, a
new session of type MultipointClient MAY be created, or
the packet MAY be discarded. This choice is outside the
scope of this specification.

If bfd.SessionType is not MulticastClient, the packet
MUST be discarded.

Else (Your Discriminator is zero)

If the State field is not Down or AdminDown, the packet MUST

be discarded.

Otherwise, the session MUST be selected based on some
combination of other fields, possibly including source
addressing information, the My Discriminator field, and the
interface over which the packet was received. The exact
method of selection is application-specific and is thus
outside the scope of this specification.

If a matching session is found, and bfd.SessionType is not
PointToPoint, the packet MUST be discarded.

If a matching session is not found, a new session of type
PointToPoint may be created, or the packet may be discarded.
This choice is outside the scope of this specification.

If the State field is Init and bfd.SessionType is not
PointToPoint, the packet MUST be discarded.

4.16.3. Transmitting BFD Control Packets

The following procedure replaces section 6.8.7 of [BFD].

BFD Control packets MUST be transmitted periodically at the rate
determined according to [BFD] section 6.8.2, except as specified in
this section.

A system MUST NOT transmit any BFD Control packets if bfd.RemoteDiscr
is zero and the system is taking the Passive role.

A system MUST NOT transmit any BFD Control packets if bfd.SilentTail
is 1.

A system MUST NOT periodically transmit BFD Control packets if Demand
mode is active on the remote system (bfd.RemoteDemandMode is 1,
bfd.SessionState is Up, and bfd.RemoteSessionState is Up) and a Poll
Sequence is not being transmitted.

A system MUST NOT periodically transmit BFD Control packets if
bfd.RemoteMinRxInterval is zero.

A system MUST NOT periodically transmit BFD Control packets if
bfd.SessionType is MulticastClient and a Poll Sequence is not being
transmitted.

If bfd.SessionType is MultipointHead, the transmit interval MUST be
set to bfd.DesiredMinTxInterval (this should happen automatically, as

bfd.RemoteMinRxInterval will be zero.)

If bfd.SessionType is not MultipointHead, the transmit interval MUST
be recalculated whenever bfd.DesiredMinTxInterval changes, or
whenever bfd.RemoteMinRxInterval changes, and is equal to the greater
of those two values. See [BFD] sections 6.8.2 and 6.8.3 for details
on transmit timers.

If bfd.SessionType is MulticastTail and periodic transmission of BFD
Control packets is just starting (due to Demand mode not being active
on the remote system), the first packet to be transmitted MUST be
delayed by a random amount of time between zero and (0.9 *
bfd.RemoteMinRxInterval).

If a BFD Control packet is received with the Poll (P) bit set to 1,
the receiving system MUST transmit a BFD Control packet with the Poll
(P) bit clear and the Final (F) bit, without respect to the
transmission timer or any other transmission limitations, without
respect to the session state, and without respect to whether Demand
mode is active on either system. A system MAY limit the rate at
which such packets are transmitted. If rate limiting is in effect,
the advertised value of Desired Min TX Interval MUST be greater than
or equal to the interval between transmitted packets imposed by the
rate limiting function. If the Multipoint (M) bit is set in the
received packet, the packet transmission MUST be delayed by a random
amount of time between zero and (0.9 * bfd.RemoteMinRxInterval).
Otherwise, the packet MUST be transmitted as soon as practicable.

A system MUST NOT set the Demand (D) bit if bfd.SessionType is
MultipointTail.

A system MUST NOT set the Demand (D) bit if bfd.SessionType is
MultipointClient or PointToPoint unless bfd.DemandMode is 1,
bfd.SessionState is Up, and bfd.RemoteSessionState is Up.

If bfd.SessionType is PointToPoint or MultipointHead, a BFD Control
packet SHOULD be transmitted during the interval between periodic
Control packet transmissions when the contents of that packet would
differ from that in the previously transmitted packet (other than the
Poll and Final bits) in order to more rapidly communicate a change in
state.

The contents of transmitted BFD Control packets MUST be set as
follows:

Version

   Set to the current version number (1).

Diagnostic (Diag)

   Set to bfd.LocalDiag.

State (Sta)

   Set to the value indicated by bfd.SessionState.

Poll (P)

   Set to 1 if the local system is sending a Poll Sequence or is a
   session of type MultipointHead soliciting the identities of the
   tails, or 0 if not.

Final (F)

   Set to 1 if the local system is responding to a Control packet
   received with the Poll (P) bit set, or 0 if not.

Control Plane Independent (C)

   Set to 1 if the local system's BFD implementation is independent
   of the control plane (it can continue to function through a
   disruption of the control plane.)

Authentication Present (A)

   Set to 1 if authentication is in use on this session (bfd.AuthType
   is nonzero), or 0 if not.

Demand (D)

   Set to bfd.DemandMode if bfd.SessionState is Up and
   bfd.RemoteSessionState is Up. Set to 1 if bfd.SessionType is
   MultipointHead or MultipointClient. Otherwise it is set to 0.

Multipoint (M)

   Set to 1 if bfd.SessionType is MultipointHead. Otherwise it is
   set to 0.

Detect Mult

   Set to bfd.DetectMult.

Length

   Set to the appropriate length, based on the fixed header length
   (24) plus any Authentication Section.

My Discriminator

   Set to bfd.LocalDiscr.

Your Discriminator

   Set to bfd.RemoteDiscr.

Desired Min TX Interval

   Set to bfd.DesiredMinTxInterval.

Required Min RX Interval

   Set to bfd.RequiredMinRxInterval.

Required Min Echo RX Interval

   Set to the minimum required Echo packet receive interval for this
   session. If this field is set to zero, the local system is
   unwilling or unable to loop back BFD Echo packets to the remote
   system, and the remote system will not send Echo packets.

   Authentication Section

      Included and set according to the rules in section 6.7 if
      authentication is in use (bfd.AuthType is nonzero.) Otherwise
      this section is not present.

5. Assumptions

   If head notification is to be used, it is assumed that a multipoint
   BFD packet encapsulation contains enough information so that a tail
   can address a unicast BFD packet to the head.

   If head notification is to be used, it is assumed that is that there
   is bidirectional unicast communication available (at the same
   protocol layer within which BFD is being run) between the tail and
   head.

   For the head to know reliably that a tail has lost multipoint
   connectivity, the unicast paths in both directions between that tail
   and the head must remain operational when the multipoint path fails.
   It is thus desirable that unicast paths not share fate with the
   multipoint path to the extent possible if the head wants reliable
   knowledge of tail state.

   Since the normal BFD three-way handshake is not used in this
   application, a tail transitioning from state Up to Down and back to
   Up again may not be reliably detected at the head.

   If authentication is in use, all tails must be configured to have a
   common authentication key in order to receive the multipoint BFD
   Control packets.

6. Operational Scenarios

   It is worth analyzing how this protocol reacts to various scenarios.
   There are three path components present, namely, the multipoint path,
   the forward unicast path (from head to a particular tail), and the
   reverse unicast path (from a tail to the head.) There are also four
   options as to how the head is notified about failures from the tail.

6.1. No Head Notification

   Since the only path used in this scenario is the multipoint path,
   none of the others matter. A failure in the multipoint path will
   result in the tail noticing the failure within a detection time, and
   the head will remain ignorant of the tail state.

6.2. Unreliable Head Notification

   In this scenario, the tail sends back unsolicicted BFD packets in
   response to the detection of a multipoint path failure. It uses the
   reverse unicast path, but not the forward unicast path.

   If the multipoint path fails but the reverse unicast path stays up,
   the tail will detect the failure within a detection time, and the
   head will know about it within one reverse packet time (since the
   notification is delayed.)

   If both the multipoint path and the reverse unicast paths fail, the
   tail will detect the failure but the head will remain unaware of it.

6.3. Semi-reliable Head Notification and Tail Solicitation

   In this scenario, the head sends occasional multipoint Polls in
   addition to (or in lieu of) non-Poll multipoint BFD Control packets,
   expecting the tails to reply with Final. This also uses the reverse
   unicast path, but not the forward unicast path.

   If the multipoint path fails but the reverse unicast path stays up,
   the tail will detect the failure within a detection time, and the
   head will know about it within one reverse packet time (the
   notification is delayed to avoid synchronization of the tails.)

   If both the multipoint path and the reverse unicast paths fail, the
   tail will detect the failure but the head will remain unaware of this
   fact.

   If the reverse unicast path fails but the multipoint path stays up,
   the head will see the BFD session fail, but the state of the
   multipoint path will be unknown to the head. The tail will continue
   to receive multipoint data traffic.

   If either the multipoint Poll or the unicast reply is lost in
   transit, the head will see the BFD session fail, but the state of the
   multipoint path will be unknown to the head. The tail will continue
   to receive multipoint data traffic.

6.4. Reliable Head Notification

   In this scenario, the head sends occasional multipoint Polls in
   addition to (or in lieu of) non-Poll multipoint BFD control packets,
   expecting the tails to reply with Final. If a tail that had
   previously replied to a multipoint Poll fails to reply (or if the
   head simply wishes to verify tail connectivity,) the head issues a
   unicast Poll Sequence to the tail. This scenario makes use of all
   three paths.

   If the multipoint path fails but the two unicast paths stay up, the
   tail will detect the failure within a detection time, and the head
   will know about it within one reverse packet time (since the
   notification is delayed.) Note that the reverse packet time may be
   smaller in this case if the head has previously issued a unicast Poll
   (since the tail will not delay transmission of the notification in
   this case.)

   If both the multipoint path and the reverse unicast paths fail
   (regardless of the state of the forward unicast path), the tail will
   detect the failure but the head will remain unaware of this fact.
   The head will detect a BFD session failure to the tail but cannot
   make a determination about the state of the tail's multipoint
   connectivity.

   If the forward unicast path fails but the reverse unicast path stays
   up, the head will detect a BFD session failure to the tail if it
   happens to send a unicast Poll sequence, but cannot make a
   determination about the state of the tail's multipoint connectivity.
   If the multipoint path to the tail fails prior to any unicast Poll
   being sent, the tail will detect the failure within a detection time,
   and the head will know about it within one reverse packet time (since
   the notification is delayed.)

   If the multipoint path stays up but the reverse unicast path fails,
   the head will see the BFD session fail if it happens to send a Poll
   Sequence, but the state of the multipoint path will be unknown to the
   head. The tail will continue to receive multipoint data traffic.

   If the multipoint path and the reverse unicast path both stay up but
   the forward unicast path fails, neither side will notice so long as a
   unicast Poll Sequence is never sent by the head. If the head sends a
   unicast Poll Sequence, the head will see the BFD session fail, but
   the state of the multipoint path will be unknown to the head. The
   tail will continue to receive multipoint data traffic.

7. **IANA Considerations**

   This document has no actions for IANA.

8. **Security Considerations**

   This specification does not raise any additional security issues
   beyond those of the specifications referred to in the list of
   normative references.

9. **Normative References**

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

   [BFD] Katz, D., and Ward, D., "Bidirectional Forwarding
       Detection", RFC 5880, June 2010

Contributors

   Rahul Aggarwal of Juniper Networks and George Swallow of Cisco
   Systems provided the initial idea for this specification and
   contributed to its development.

Authors' Addresses

    Dave Katz
    Juniper Networks
    1194 N. Mathilda Ave.
    Sunnyvale, California 94089-1206 USA
    Phone: +1-408-745-2000
    Email: dkatz@juniper.net

    Dave Ward
    Cisco
    170 West Tasman Dr.
    San Jose, CA 95134
    Phone: +1-408-526-4000
    Email: dward@cisco.com

Changes from the previous draft

   This is a reissue of the previous version. There are only minor
   editorial changes.

This document expires in January, 2013.