          Seamless Bidirectional Forwarding Detection (S-BFD) Use Cases

Abstract

   This document describes various use cases for Seamless Bidirectional
   Forwarding Detection (S-BFD) and provides requirements such that
   protocol mechanisms allow for simplified detection of forwarding
   failures.

   These use cases support S-BFD, which is a simplified mechanism for
   using BFD with a large proportion of negotiation aspects eliminated,
   accelerating the establishment of a BFD session. The benefits of
   S-BFD include quick provisioning, as well as improved control and
   flexibility for network nodes initiating path monitoring.

Copyright Notice

Table of Contents

1. Introduction

   Bidirectional Forwarding Detection (BFD), as defined in [RFC5880], is
   a lightweight protocol used to detect forwarding failures. Various
   protocols, applications, and clients rely on BFD for failure
   detection. Even though the protocol is lightweight and simple, there
   are certain use cases where faster setup of sessions and faster
   continuity checks of the data-forwarding paths are necessary. This
   document identifies these use cases and consequent requirements, such
   that enhancements and extensions result in a Seamless BFD (S-BFD)
   protocol.

   BFD is a simple and lightweight "Hello" protocol to detect data-plane
   failures. With dynamic provisioning of forwarding paths on a large
   scale, establishing BFD sessions for each of those paths not only
   creates operational complexity but also causes undesirable delay in
   establishing or deleting sessions. The existing session
   establishment mechanism of the BFD protocol has to be enhanced in
   order to minimize the time for the session to come up to validate the
   forwarding path.

   This document specifically identifies various use cases and
   corresponding requirements in order to enhance BFD and other
   supporting protocols. Specifically, one key goal is removing the
   time delay (i.e., the "seam") between when a network node wants to
   perform a continuity test and when the node completes that continuity
   test. Consequently, "Seamless BFD" (S-BFD) has been chosen as the
   name for this mechanism.

   While the identified requirements could meet various use cases, it is
   outside the scope of this document to identify all of the possible
   and necessary requirements. Solutions related to the identified use
   cases and protocol-specific enhancements or proposals are outside the
   scope of this document as well. Protocol definitions to support
   these use cases can be found in [RFC7880] and [RFC7881].

1.1. Terminology

   The reader is expected to be familiar with the BFD [RFC5880], IP
   [RFC791] [RFC2460], MPLS [RFC3031], and Segment Routing [SR-ARCH]
   terms and protocol constructs.

   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].

2. Introduction to Seamless BFD

   BFD, as defined in [RFC5880], requires two network nodes to exchange
   locally allocated discriminators. These discriminators enable the
   identification of the sender and the receiver of BFD packets over the
   particular session. Subsequently, BFD performs proactive continuity
   monitoring of the forwarding path between the two. Several
   specifications describe BFD's multiple deployment uses:

   o [RFC5881] defines BFD over IPv4 and IPv6 for single IP hops.

   o [RFC5883] defines BFD over multi-hop paths.

   o [RFC5884] defines BFD for MPLS Label Switched Paths (LSPs).

   o [RFC5885] defines BFD for MPLS Pseudowires (PWs).

   Currently, BFD is best suited for verifying that two endpoints are
   mutually reachable or that an existing connection continues to be up
   and alive. In order for BFD to be able to initially verify that a
   connection is valid and that it connects the expected set of
   endpoints, it is necessary to provide each endpoint with the
   discriminators associated with the connection at each endpoint prior
   to initiating BFD sessions. The discriminators are used to verify
   that the connection is up and valid. Currently, the exchange of
   discriminators and the demultiplexing of the initial BFD packets are
   application dependent.

   If this information is already known to the endpoints of a potential
   BFD session, the initial handshake including an exchange of
   discriminators is unnecessary, and it is possible for the endpoints
   to begin BFD messaging seamlessly. A key objective of the S-BFD use
   cases described in this document is to avoid needing to exchange the
   initial packets before the BFD session can be established, with the
   goal of getting to the established state more quickly; in other
   words, the initial exchange of discriminator information is an
   unnecessary extra step that may be avoided for these cases.

   In a given scenario, an entity (such as an operator or a centralized
   controller) determines a set of network entities to which BFD
   sessions might need to be established. In traditional BFD, each of
   those network entities chooses a BFD Discriminator for each BFD
   session that the entity will participate in (see Section 6.3 of
   [RFC5880]). However, a key goal of S-BFD is to provide operational
   simplification. In this context, for S-BFD, each of those network
   entities is assigned one or more BFD Discriminators, and those
   network entities are allowed to use one Discriminator value for
   multiple sessions. Therefore, there may be only one or a few

discriminators assigned to a node. These network entities will
create an S-BFD listener session instance that listens for incoming
BFD Control packets. When the mappings between specific network
entities and their corresponding BFD Discriminators are known to
other network nodes belonging to the same administrative domain,
then, without having received any BFD packets from a particular
target, a network entity in this network is able to send a BFD
Control packet to the target's assigned discriminator in the
Your Discriminator field. The target network node, upon reception of
such a BFD Control packet, will transmit a response BFD Control
packet back to the sender.

3. Use Cases

   As per the BFD protocol [RFC5880], BFD sessions are established using
   a handshake mechanism prior to validating the forwarding path. This
   section outlines some use cases where the existing mechanism may not
   be able to satisfy the requirements identified. In addition, some of
   the use cases also stress the need for expedited BFD session
   establishment while preserving the benefits of forwarding failure
   detection using existing BFD mechanisms. Both of these high-level
   goals result in the S-BFD use cases outlined in this document.

3.1. Unidirectional Forwarding Path Validation

   Even though bidirectional verification of forwarding paths is useful,
   there are scenarios where verification is only required in one
   direction between a pair of nodes. One such case is when a static
   route uses BFD to validate reachability to the next-hop IP router.
   In this case, the static route is established from one network entity
   to another. The requirement in this case is only to validate the
   forwarding path for that statically established unidirectional path.
   Validation of the forwarding path in the direction of the target
   entity to the originating entity is not required in this scenario.
   Many LSPs have the same unidirectional characteristics and
   unidirectional validation requirements. Such LSPs are common in
   Segment Routing and LDP-based MPLS networks. A final example is when
   a unidirectional tunnel uses BFD to validate the reachability of an
   egress node.

   Additionally, validation of the unidirectional path has operational
   implications. If traditional BFD is to be used, the target network
   entity, as well as an initiator, has to be provisioned, even though
   reverse-path validation with the BFD session is not required.
   However, in the case of unidirectional BFD, there is no need for
   provisioning on the target network entity -- only on the source
   entity.

In this use case, a BFD session could be established in a single
direction. When the target network entity receives the packet, it
identifies the packet as BFD in an application-specific manner (for
example, a destination UDP port number). Subsequently, the BFD
module processes the packet, using the Your Discriminator value as
context. Then, the network entity sends a response to the
originator. This does not necessitate the requirement for
establishment of a bidirectional session; hence, the two-way
handshake to exchange discriminators is not needed. The target node
does not need to know the My Discriminator value of the source node.

Thus, in this use case a requirement for BFD is to enable session
establishment from the source network entity to the target network
entity without the need to have a session (and state) for the reverse
direction. Further, another requirement is that the BFD response
from the target back to the sender can take any (in-band or
out-of-band) path. The BFD module in the target network entity (for
the BFD session), upon receipt of a BFD packet, starts processing the
BFD packet based on the discriminator received. The source network
entity can therefore establish a unidirectional BFD session without
the bidirectional handshake and exchange of discriminators for
session establishment.

3.2. Validation of the Forwarding Path prior to Switching Traffic

In this use case, BFD is used to verify reachability before sending
traffic via a path/LSP. This comes at a cost: traffic is prevented
from using the path/LSP until BFD is able to validate reachability;
this could take seconds due to BFD session bring-up sequences
[RFC5880], LSP Ping bootstrapping [RFC5884], etc. This use case
would be better supported by eliminating the need for the initial BFD
session negotiation.

All it takes to be able to send BFD packets to a target and for the
target to properly demultiplex these packets is for the source
network entities to know what Discriminator values will be used for
the session. This is also the case for S-BFD: the three-way
handshake mechanism is eliminated during the bootstrapping of BFD
sessions. However, this information is required at each entity to
verify that BFD messages are being received from the expected
endpoints; hence, the handshake mechanism serves no purpose.
Elimination of the unnecessary handshake mechanism allows for faster
reachability validation of BFD provisioned paths/LSPs.

In addition, it is expected that some MPLS technologies will require
traffic-engineered LSPs to be created dynamically, perhaps driven by
external applications, as, for example, in Software-Defined
Networking (SDN). It will be desirable to perform BFD validation as
soon as the LSPs are created, so as to use them.

In order to support this use case, an S-BFD session is established
without the need for session negotiation and exchange of
discriminators.

3.3. Centralized Traffic Engineering

Various technologies in the SDN domain that involve controller-based
networks have evolved such that the intelligence, traditionally
placed in a distributed and dynamic control plane, is separated from
the networking entities themselves; instead, it resides in a
(logically) centralized place. There are various controllers that
perform the function of establishing forwarding paths for the data
flow. Traffic engineering is one important function, where the path
of the traffic flow is engineered, depending upon various attributes
and constraints of the traffic paths as well as the network state.

When the intelligence of the network resides in a centralized entity,
the ability to manage and maintain the dynamic network, and its
multiple data paths and node reachability, becomes a challenge. One
way to ensure that the forwarding paths are valid and working is done
by validation using BFD. When traffic-engineered tunnels are
created, it is operationally critical to ensure that the forwarding
paths are working, prior to switching the traffic onto the engineered
tunnels. In the absence of distributed control-plane protocols, it
may be desirable to verify any arbitrary forwarding path in the
network. With tunnels being engineered by a centralized entity, when
the network state changes, traffic has to be switched with minimum
latency and without black-holing of the data.

It is highly desirable in this centralized traffic-engineering use
case that the traditional BFD session establishment and validation of
the forwarding path do not become a bottleneck. If the controller or
other centralized entity is able to very rapidly verify the
forwarding path of a traffic-engineered tunnel, it could steer the
traffic onto the traffic-engineered tunnel very quickly, thus
minimizing adverse effects on a service. This is even more useful
and necessary when the scale of the network and the number of
traffic-engineered tunnels grow.

The cost associated with the time required for BFD session
negotiation and establishment of BFD sessions to identify valid paths
is very high when providing network redundancy is a critical issue.

3.4. BFD in Centralized Segment Routing

   A monitoring technique for a Segment Routing network based on a
   centralized controller is described in [SR-MPLS]. Specific
   Operations, Administration, and Maintenance (OAM) requirements for
   Segment Routing are captured in [SR-OAM-REQS]. In validating this
   use case, one of the requirements is to ensure that the BFD packet's
   behavior is according to the monitoring specified for the segment and
   that the packet is U-turned at the expected node. This criterion
   ensures the continuity check to the adjacent Segment Identifier.

   To support this use case, the operational requirement is for BFD,
   initiated from a centralized controller, to perform liveness
   detection for any given segment in its domain.

3.5. Efficient BFD Operation under Resource Constraints

   When BFD sessions are being set up, torn down, or modified (i.e.,
   when parameters such as intervals and multipliers are being
   modified), BFD requires additional packets, other than scheduled
   packet transmissions, to complete the negotiation procedures (i.e.,
   Poll (P) bits and Final (F) bits; see Section 4.1 of [RFC5880]).
   There are scenarios where network resources are constrained: a node
   may require BFD to monitor a very large number of paths, or BFD may
   need to operate in low-powered and traffic-sensitive networks; these
   include microwave systems, low-powered nanocells, and others. In
   these scenarios, it is desirable for BFD to slow down, speed up,
   stop, or resume at will and with a minimal number of additional BFD
   packets exchanged to modify the session or establish a new one.

   The established BFD session parameters, and such attributes as
   transmission interval and receiver interval, need to be modifiable
   without changing the state of the session.

3.6. BFD for Anycast Addresses

   The BFD protocol requires two endpoints to host BFD sessions, both
   sending packets to each other. This BFD model does not fit well with
   anycast address monitoring, as BFD packets transmitted from a network
   node to an anycast address will reach only one of potentially many
   network nodes hosting the anycast address.

   This use case verifies that a source node can send a packet to an
   anycast address and that the target node to which the packet is
   delivered can send a response packet to the source node. Traditional
   BFD cannot fulfill this requirement, since it does not provide for a

set of BFD agents to collectively form one endpoint of a BFD session.
The concept of a "target listener" in S-BFD fulfills this
requirement.

To support this use case, the BFD sender transmits BFD packets, which
are received by any of the nodes hosting the anycast address to which
the BFD packets are being sent. The anycast target that receives the
BFD packet responds. This use case does not imply BFD session
establishment with every node hosting the anycast address.
Consequently, in this anycast use case, target nodes that do not
happen to receive any of the BFD packets do not need to maintain any
state, and the source node does not need to maintain separate state
for each target node.

3.7. BFD Fault Isolation

BFD for multi-hop paths [RFC5883] and BFD for MPLS LSPs [RFC5884]
perform end-to-end validation, traversing multiple network nodes.
BFD has been designed to declare a failure to receive some number of
consecutive packets. This failure can be caused by a fault anywhere
along these paths. Fast failure detection allows for rapid fault
detection and consequent rapid path recovery procedures. However,
operators often have to follow up, manually or automatically, to
attempt to identify and localize the fault that caused BFD sessions
to fail (i.e., fault isolation). If Equal-Cost Multipath (ECMP) is
used, the usage of other tools to isolate the fault (e.g.,
traceroute) may cause the packets to traverse a different path
through the network. In addition, the longer it takes from the time
of BFD session failure to the time that fault isolation begins, the
more likely the fault will not be isolated (e.g., a fault may be
corrected via rerouting or some other means during that time). If
BFD had built-in fault-isolation capability, fault isolation would be
triggered when the fault was first detected. This embedded fault
isolation would be more effective (i.e., faults would be detected
sooner) if those BFD fault-isolation packets were load-balanced in
the same way as the BFD packets that were dropped.

This use case describes S-BFD fault-isolation capabilities, utilizing
a TTL field (e.g., as described in Section 5.1.1 of [RFC7881]) or
using fields that indicate status.

3.8. Multiple BFD Sessions to the Same Target Node

BFD is capable of providing very fast failure detection, as relevant
network nodes continuously transmit BFD packets at the negotiated
rate. If BFD packet transmission is interrupted, even for a very
short period of time, BFD can declare a failure irrespective of path
liveness. On a system where BFD is running, it is possible for

certain events to (intentionally or unintentionally) cause a brief
interruption of BFD packet transmissions. With distributed
architectures of BFD implementations, this case can be prevented.
This use case is for an S-BFD node running multiple BFD sessions to
the same target node, with those sessions hosted on different system
modules (e.g., in different CPU instances). This can reduce false
failures, resulting in a more stable network.

To support this use case, a mapping between the multiple
discriminators on a single system and the specific entity within that
system is required.

3.9. An MPLS BFD Session per ECMP Path

BFD for MPLS LSPs, defined in [RFC5884], describes procedures for
running BFD as an LSP in-band continuity check mechanism by using
MPLS Echo Request messages [RFC4379] to bootstrap the BFD session on
the target (i.e., egress) node. Section 4 of [RFC5884] also
describes the possibility of running multiple BFD sessions per
alternative of LSPs. [RFC7726] further clarifies the procedures, for
both ingress and egress nodes, regarding how to bootstrap, maintain,
and remove multiple BFD sessions for the same <MPLS LSP, FEC> tuple
("FEC" means Forwarding Equivalence Class). However, this mechanism
still requires the use of MPLS LSP Ping for bootstrapping,
round trips for initialization, and keeping state at the receiver.

In the presence of ECMP within an MPLS LSP, it may be desirable to
run in-band monitoring that exercises every path of this ECMP.
Otherwise, there will be scenarios where an in-band BFD session
remains up through one path but traffic is black-holing over another
path. A BFD session per ECMP path of an LSP requires the definition
of procedures that update [RFC5884] in terms of how to bootstrap and
maintain the correct set of BFD sessions on the egress node.
However, for traditional BFD, that requires the constant use of MPLS
Echo Request messages to create and delete BFD sessions on the egress
node when ECMP paths and/or corresponding load-balance hash keys
change. If a BFD session over any paths of the LSP can be
instantiated, stopped, and resumed without requiring additional
procedures for bootstrapping via an MPLS Echo Request message, it
would greatly simplify both implementations and operations and
would benefit network devices, as less processing would be required
by them.

To support this requirement, multiple S-BFD sessions need to be
established over different ECMP paths between the same pair of source
and target nodes.

4. Detailed Requirements for Seamless BFD

   REQ 1:   Upon receipt of an S-BFD packet, a target network entity
             (for the S-BFD session) MUST process the packet based on the
             discriminator received in the BFD packet. If the S-BFD
             context is found, the target network entity MUST be able to
             send a response.

   REQ 2:   The source network entity MUST be able to establish a
             unidirectional S-BFD session without the bidirectional
             handshake of discriminators for session establishment.

   REQ 3:   The S-BFD session MUST be able to be established without the
             need for the exchange of discriminators during session
             negotiation.

   REQ 4:   In a Segment Routed network, S-BFD MUST be able to perform
             liveness detection initiated from a centralized controller
             for any given segment in its domain.

   REQ 5:   The established S-BFD session parameters and attributes,
             such as transmission interval and reception interval, MUST
             be modifiable without changing the state of the session.

   REQ 6:   An S-BFD source network entity MUST be able to send Control
             packets to an anycast address. These packets are received
             and processed by any node hosting the anycast address. The
             S-BFD entity MUST be able to receive responses to S-BFD
             Control packets from any of these anycast nodes, without
             establishing a separate S-BFD session with every node
             hosting the anycast address.

   REQ 7:   S-BFD SHOULD support fault-isolation capability, which MAY
             be triggered when a fault is encountered.

   REQ 8:   S-BFD SHOULD be able to establish multiple sessions between
             the same pair of source and target nodes. This requirement
             enables but does not guarantee the ability to monitor
             divergent paths in ECMP environments. It also provides
             resiliency in distributed router architectures. The mapping
             between BFD Discriminators and particular entities (e.g.,
             ECMP paths, line cards) is out of scope for the S-BFD
             protocol.

   REQ 9:   The S-BFD protocol MUST provide mechanisms for loop
             detection and prevention, protecting against malicious
             attacks attempting to create packet loops.

   REQ 10:   S-BFD MUST incorporate robust security protections against
             impersonators, malicious actors, and various active and
             passive attacks. The simple and accelerated establishment
             of an S-BFD session should not negatively affect security.

5. Security Considerations

   This document details use cases for S-BFD and identifies various
   associated requirements. Some of these requirements are security
   related. The use cases described herein do not expose a system to
   abuse or additional security risks. Since some negotiation aspects
   are eliminated, a misconfiguration can result in S-BFD packets being
   sent to an incorrect node. If this receiving node runs S-BFD, the
   packet will be discarded due to discriminator mismatch. If the node
   does not run S-BFD, the packet will be naturally discarded.

   The proposed new protocols, extensions, and enhancements for S-BFD
   supporting these use cases and realizing these requirements will
   address associated security considerations. S-BFD should not have
   reduced security capabilities as compared to traditional BFD.

6. References

6.1. Normative References

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

   [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection
              (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010,
              <http://www.rfc-editor.org/info/rfc5880>.

   [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection
              (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881,
              DOI 10.17487/RFC5881, June 2010,
              <http://www.rfc-editor.org/info/rfc5881>.

   [RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection
              (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883,
              June 2010, <http://www.rfc-editor.org/info/rfc5883>.

   [RFC5884] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow,
              "Bidirectional Forwarding Detection (BFD) for MPLS Label
              Switched Paths (LSPs)", RFC 5884, DOI 10.17487/RFC5884,
              June 2010, <http://www.rfc-editor.org/info/rfc5884>.

   [RFC5885] Nadeau, T., Ed., and C. Pignataro, Ed., "Bidirectional
              Forwarding Detection (BFD) for the Pseudowire Virtual
              Circuit Connectivity Verification (VCCV)", RFC 5885,
              DOI 10.17487/RFC5885, June 2010,
              <http://www.rfc-editor.org/info/rfc5885>.

6.2. Informative References

   [RFC791]   Postel, J., "Internet Protocol", STD 5, RFC 791,
              DOI 10.17487/RFC791, September 1981,
              <http://www.rfc-editor.org/info/rfc791>.

   [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6
              (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460,
              December 1998, <http://www.rfc-editor.org/info/rfc2460>.

   [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol
              Label Switching Architecture", RFC 3031,
              DOI 10.17487/RFC3031, January 2001,
              <http://www.rfc-editor.org/info/rfc3031>.

   [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol
              Label Switched (MPLS) Data Plane Failures", RFC 4379,
              DOI 10.17487/RFC4379, February 2006,
              <http://www.rfc-editor.org/info/rfc4379>.

   [RFC7726] Govindan, V., Rajaraman, K., Mirsky, G., Akiya, N., and S.
              Aldrin, "Clarifying Procedures for Establishing BFD
              Sessions for MPLS Label Switched Paths (LSPs)", RFC 7726,
              DOI 10.17487/RFC7726, January 2016,
              <http://www.rfc-editor.org/info/rfc7726>.

   [RFC7880] Pignataro, C., Ward, D., Akiya, N., Bhatia, M., and S.
              Pallagatti, "Seamless Bidirectional Forwarding Detection
              (S-BFD)", RFC 7880, DOI 10.17487/RFC7880, July 2016,
              <http://www.rfc-editor.org/info/rfc7880>.

   [RFC7881] Pignataro, C., Ward, D., and N. Akiya, "Seamless
              Bidirectional Forwarding Detection (S-BFD) for IPv4, IPv6,
              and MPLS", RFC 7881, DOI 10.17487/RFC7881, July 2016,
              <http://www.rfc-editor.org/info/rfc7881>.

   [SR-ARCH] Filsfils, C., Ed., Previdi, S., Ed., Decraene, B.,
              Litkowski, S., and R. Shakir, "Segment Routing
              Architecture", Work in Progress,
              draft-ietf-spring-segment-routing-09, July 2016.

   [SR-MPLS] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N.
              Kumar, "A Scalable and Topology-Aware MPLS Dataplane
              Monitoring System", Work in Progress,
              draft-ietf-spring-oam-usecase-03, April 2016.

   [SR-OAM-REQS]
              Kumar, N., Pignataro, C., Akiya, N., Geib, R., Mirsky, G.,
              and S. Litkowski, "OAM Requirements for Segment Routing
              Network", Work in Progress,
              draft-ietf-spring-sr-oam-requirement-02, July 2016.

Contributors

   The following are key contributors to this document:

      Manav Bhatia, Ionos Networks
      Satoru Matsushima, Softbank
      Glenn Hayden, ATT
      Santosh P K
      Mach Chen, Huawei
      Nobo Akiya, Big Switch Networks

Authors' Addresses

   Sam Aldrin
   Google, Inc.

   Email: aldrin.ietf@gmail.com

   Carlos Pignataro
   Cisco Systems, Inc.

   Email: cpignata@cisco.com

   Greg Mirsky
   Ericsson

   Email: gregory.mirsky@ericsson.com

   Nagendra Kumar
   Cisco Systems, Inc.

   Email: naikumar@cisco.com