Network Working Group
Internet Draft

D. Katz
Juniper Networks
D. Ward
Cisco Systems

Expires: December, 2006

June, 2006

Generic Application of BFD
draft-ietf-bfd-generic-02.txt

Status of this Memo

Copyright Notice

Abstract

This document describes the generic application of the Bidirectional Forwarding Detection (BFD) protocol. 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].

1. Introduction

The Bidirectional Forwarding Detection protocol [BFD] provides a liveness detection mechanism that can be utilized by other network components for which their integral liveness mechanisms are either too slow, inappropriate, or nonexistent. Other drafts have detailed the use of BFD with specific encapsulations ([BFD-1HOP], [BFD-MULTI], [BFD-MPLS]). As the utility of BFD has become understood, there have been calls to specify BFD interactions with a growing list of network functions. Rather than producing a long series of short documents on the application of BFD, it seemed worthwhile to describe the interactions between BFD and other network functions in a broad way.

This document describes the generic application of BFD. Specific protocol applications are provided for illustrative purposes.

2. Overview

The Bidirectional Forwarding Detection (BFD) specification defines a protocol with simple and specific semantics. Its sole purpose is to verify connectivity between a pair of systems, for a particular data protocol across a path (which may be of any technology, length, or OSI layer). The promptness of the detection of a path failure can be controlled by trading off protocol overhead and system load with detection times.

BFD is *not* intended to directly provide control protocol liveness information; those protocols have their own means and vagaries. Rather, control protocols can use the services provided by BFD to inform their operation. BFD can be viewed as a service provided by

the layer in which it is running.

The service interface with BFD is straightforward. The application supplies session parameters (neighbor address, time parameters, protocol options), and BFD provides the session state, of which the most interesting transitions are to and from the Up state. The application is expected to bootstrap the BFD session, as BFD has no discovery mechanism.

An implementation SHOULD establish only a single BFD session per data protocol path, regardless of the number of applications that wish to utilize it. There is no additional value in having multiple BFD sessions to the same endpoints. If multiple applications request different session parameters, it is a local issue as to how to resolve the parameter conflicts. BFD in turn will notify all applications bound to a session when a session state change occurs.

BFD should be viewed as having an advisory role to the protocol or protocols or other network functions with which it is interacting, which will then use their own mechanisms to effect any state transitions. The interaction is very much at arm's length, which keeps things simple and decoupled. In particular, BFD explicitly does not carry application-specific information, partly for architectural reasons, and partly because BFD may have curious and unpredictable latency characteristics and as such makes a poor transport mechanism.

It is important to remember that the interaction between BFD and its client applications has essentially no interoperability issues, because BFD is acting in an advisory nature (similar to hardware signaling the loss of light on a fiber optic circuit, for example) and existing mechanisms in the client applications are used in reaction to BFD events. In fact, BFD may interact with only one of a pair of systems for a particular client application without any ill effect.

3. Control Protocol Interactions

Very common client applications of BFD are control protocols, such as routing protocols. The object when BFD interacts with a control protocol is to advise the control protocol of the connectivity of the data protocol. In the case of routing protocols, for example, this allows the connectivity failure to trigger the rerouting of traffic around the failed path more quickly than the native detection mechanisms.

3.1. Session Establishment

BFD sessions are typically bootstrapped by the control protocol, using the mechanism (discovery, configuration) used by the control protocol to find neighbors. In most cases it is not desirable to preclude the control protocol from establishing an adjacency if the BFD session cannot be established (usually because the neighbor does not support BFD.)

If the control protocol carries signaling that indicates the willingness of each system to establish a BFD session, the lack of a BFD session MAY be used to block establishment of a control protocol adjacency.

If it appears that the neighboring system does not support BFD (because the control protocol adjacency was established but no BFD Control packets have been received from the neighbor) a system MAY increase the interval between transmitted BFD Control packets beyond

the minimum specified in [BFD]. This will have negligible impact on BFD session establishment if the neighbor decides to run BFD after all, since BFD Control packets will be sent on an event-driven basis once the first packet is seen from the neighbor.

The setting of BFD's various timing parameters and modes are not subject to standardization. Note that all protocols sharing a session will operate using the same parameters. The mechanism for choosing the parameters among those desired by the various protocols are outside the scope of this specification. It is generally useful to choose the parameters resulting in the shortest detection time; a particular client application can always apply hysteresis to the notifications from BFD if it desires longer detection times.

3.2. Reaction to BFD Session State Changes

The mechanism by which the control protocol reacts to a path failure signaled by BFD depends on the capabilities of the protocol.

3.2.1. Control Protocols with a Single Data Protocol

A control protocol that is tightly bound to a single failing data protocol SHOULD take action to ensure that data traffic is no longer directed to the failing path. Note that this should not be interpreted as BFD replacing the control protocol liveness mechanism, if any, as the control protocol may rely on mechanisms not verified by BFD (multicast, for instance) so BFD most likely cannot detect all failures that would impact the control protocol. However, a control protocol MAY choose to use BFD session state information to more rapidly detect an impending control protocol failure, particularly if the control protocol operates in band (over the data protocol.)

Therefore, when a BFD session transitions from Up to Down, action SHOULD be taken in the control protocol to signal the lack of connectivity for the data protocol over which BFD is running. If the

control protocol has an explicit mechanism for announcing path state, a system SHOULD use that mechanism rather than impacting the connectivity of the control protocol, particularly if the control protocol operates out of band from the failed data protocol. However, if such a mechanism is not available, a control protocol timeout SHOULD be emulated for the associated neighbor.

3.2.2. Control Protocols with Multiple Data Protocols

Slightly different mechanisms are used if the control protocol supports the routing of multiple data protocols, depending on whether the control protocol supports separate topologies for each data protocol.

3.2.2.1. Shared Topologies

With a shared topology, if one of the data protocols fails (as signaled by the associated BFD session), it is necessary to consider the path to have failed for all data protocols. Otherwise, there is no way for the control protocol to turn away traffic for the failed data protocol (and such traffic would be black holed indefinitely.)

Therefore, when a BFD session transitions from Up to Down, action SHOULD be taken in the control protocol to signal the lack of

connectivity for all data protocols sharing the topology. If this cannot be signaled otherwise, a control protocol timeout SHOULD be emulated for the associated neighbor.

3.2.2.2. Independent Topologies

With individual routing topologies for each data protocol, only the failed data protocol needs to be rerouted around the failed path.

Therefore, when a BFD session transitions from Up to Down, action SHOULD be taken in the control protocol to signal the lack of connectivity for the data protocol over which BFD is running. Generally this can be done without impacting the connectivity of other data protocols (since otherwise it is very difficult to support

separate topologies for multiple data protocols.)

3.2.3. Partial BFD Deployments

Note that it is possible in some failure scenarios for the network to be in a state such that the control protocol comes up, but the BFD session cannot be established, and, more particularly, data cannot be forwarded. To avoid this situation, it would be beneficial to not allow the control protocol to establish a neighbor adjacency. However, this would preclude the operation of the control protocol in an environment in which not all systems support BFD.

Therefore, if a BFD session is not in Up state (possibly because the remote system does not support BFD), it is OPTIONAL to preclude the establishment of a control protocol neighbor adjacency. The choice of whether to do so SHOULD be controlled by means outside the scope of this specification, such as configuration or other mechanisms. If a neighbor adjacency is established for the control protocol but the corresponding BFD session is not in Up state (implying that the neighbor does not support BFD) implementations MAY raise the BFD transmit and receive intervals beyond the minimum of one second specified in [BFD] in order to minimize extraneous traffic.

3.3. Interactions with Graceful Restart Mechanisms

A number of control protocols support Graceful Restart mechanisms. These mechanisms are designed to allow a control protocol to restart without perturbing network connectivity state (lest it appear that the system and/or all of its links had failed.) They are predicated on the existence of a separate forwarding plane that does not necessarily share fate with the control plane in which the routing

protocols operate. In particular, the assumption is that the forwarding plane can continue to function while the protocols restart and sort things out.

BFD implementations announce via the Control Plane Independent (C) bit whether or not BFD shares fate with the control plane. This information is used to determine the actions to be taken in conjunction with Graceful Restart. If BFD does not share its fate

with the control plane on either system, it can be used to determine whether Graceful Restart in a control protocol is NOT viable (the forwarding plane is not operating.)

If the control protocol has a Graceful Restart mechanism, BFD may be used in conjunction with this mechanism. The interaction between BFD and the control protocol depends on the capabilities of the control protocol, and whether or not BFD shares fate with the control plane. In particular, it may be desirable for a BFD session failure to abort the Graceful Restart process and allow the failure to be visible to the network.

3.3.1. BFD Fate Independent of the Control Plane

If BFD is implemented in the forwarding plane and does not share fate with the control plane on either system (the "C" bit is set in the BFD Control packets in both directions), control protocol restarts should not affect the BFD Session. In this case, a BFD session failure implies that data can no longer be forwarded, so any Graceful Restart in progress at the time of the BFD session failure SHOULD be aborted in order to avoid black holes, and a topology change SHOULD be signaled in the control protocol.

3.3.2. BFD Shares Fate with the Control Plane

If BFD shares fate with the control plane on either system (the "C" bit is clear in either direction), a BFD session failure cannot be disentangled from other events taking place in the control plane. In many cases, the BFD session will fail as a side effect of the restart taking place. As such, it would be best to avoid aborting any Graceful Restart taking place, if possible (since otherwise BFD and Graceful Restart cannot coexist.)

There is some risk in doing so, since a simultaneous failure or restart of the forwarding plane will not be detected, but this is always an issue when BFD shares fate with the control plane.

3.3.2.1. Control Protocols with Planned Restart Signaling

Some control protocols can signal a planned restart prior to the restart taking place. In this case, if a BFD session failure occurs during the restart, such a planned restart SHOULD NOT be aborted and the session failure SHOULD NOT result in a topology change being signaled in the control protocol.

3.3.2.2. Control Protocols Without Planned Restart Signaling

Control protocols that cannot signal a planned restart depend on the recently restarted system to signal the Graceful Restart prior to the control protocol adjacency timeout. In most cases, whether the restart is planned or unplanned, it is likely that the BFD session will time out prior to the onset of Graceful Restart, in which case a topology change SHOULD be signaled in the control protocol as specified in section 3.2.

However, if the restart is in fact planned, an implementation MAY adjust the BFD session timing parameters prior to restarting in such a way that the detection time in each direction is longer than the restart period of the control protocol, providing the restarting system the same opportunity to enter Graceful Restart as it would have without BFD. The restarting system SHOULD NOT send any BFD Control packets until there is a high likelihood that its neighbors know a Graceful Restart is taking place, as the first BFD Control packet will cause the BFD session to fail.

3.4. Interactions with Multiple Control Protocols

If multiple control protocols wish to establish BFD sessions with the same remote system for the same data protocol, all MUST share a single BFD session.

If hierarchical or dependent layers of control protocols are in use (say, OSPF and IBGP), it may not be useful for more than one of them to interact with BFD. In this example, because IBGP is dependent on OSPF for its routing information, the faster failure detection relayed to IBGP may actually be detrimental. The cost of a peer state transition is high in BGP, and OSPF will naturally heal the path through the network if it were to receive the failure detection.

In general, it is best for the protocol at the lowest point in the hierarchy to interact with BFD, and then to use existing interactions between the control protocols to effect changes as necessary. This will provide the fastest possible failure detection and recovery in a

network.

4. Interactions With Non-Protocol Functions

BFD session status may be used to affect other system functions that are not protocol-based (for example, static routes.) If the path to a remote system fails, it may be desirable to avoid passing traffic to that remote system, so the local system may wish to take internal measures to accomplish this (such as withdrawing a static route and withdrawing that route from routing protocols.)

Bootstrapping of the BFD session in the non-protocol case is likely to be derived from configuration information.

There is no need to exchange endpoints or discriminator values via any mechanism other than configuration (via Operational Support Systems or any other means) as the endpoints must be known and configured by the same means.

5. Data Protocols and Demultiplexing

BFD is intended to protect a single "data protocol" and is encapsulated within that protocol. A pair of systems may have multiple BFD sessions over the same topology if they support (and are encapsulated by) different protocols. For example, if two systems have IPv4 and IPv6 running across the same link between them, these are considered two separate paths and require two separate BFD sessions.

This same technique is used for more fine-grained paths. For example, if multiple differentiated services [DIFFSERV] are being operated on over IPv4, an independent BFD session may be run for each service level. The BFD Control packets must be marked in the same way as the data packets, partly to ensure as much fate sharing as possible between BFD and data traffic, and also to demultiplex the initial packet if the discriminator values have not been exchanged.

6. Other Application Issues

BFD can provide liveness detection for OAM-like functions in tunneling and pseudowire protocols. Running BFD inside the tunnel is recommended, as it exercises more aspects of the path. One way to accommodate this is to address BFD packets based on the tunnel endpoints, assuming that they are numbered.

If a planned outage is to take place on a path over which BFD is run, it is preferable to take down the BFD session by going into ADMIN DOWN state prior to the outage.

7. Interoperability Issues

The BFD protocol itself is designed so that it will always interoperate at a basic level; asynchronous mode is mandatory and is always available, and other modes and functions are negotiated at run time. Since the service provided by BFD is identical regardless of the variants used, the particular choice of BFD options has no bearing on interoperability.

The interaction between BFD and other protocols and control functions is very loosely coupled. The actions taken are based on existing mechanisms in those protocols and functions, so interoperability problems are very unlikely unless BFD is applied in contradictory ways (such as a BFD session failure causing one implementation to go down and another implementation to come up.) In fact, BFD may be advising one system for a particular control function but not the other; the only impact of this would be potentially asymmetric control protocol failure detection.

8. Specific Protocol Interactions (Non-Normative)

As noted above, there are no interoperability concerns regarding interactions between BFD and control protocols. However, there is

enough concern and confusion in this area so that it is worthwhile to provide examples of interactions with specific protocols.

Since the interactions do not affect interoperability, they are non-normative.

8.1. BFD Interactions with OSPFv2, OSPFv3, and IS-IS

The two versions of OSPF ([OSPFv2] and [OSPFv3]), as well as IS-IS [ISIS], all suffer from an architectural limitation, namely that their Hello protocols are limited in the granularity of their failure detection times. In particular, OSPF has a minimum detection time of two seconds, and IS-IS has a minimum detection time of one second.

BFD may be used to achieve arbitrarily small detection times for these protocols by supplementing the Hello protocols used in each case.

8.1.1. Session Establishment

The most obvious choice for triggering BFD session establishment with these protocols would be to use the discovery mechanism inherent in the Hello protocols in OSPF and IS-IS to bootstrap the establishment of the BFD session. Any BFD sessions established to support OSPF and IS-IS across a single IP hop must operate in accordance with [BFD-1HOP].

8.1.2. Reaction to BFD State Changes

The basic mechanisms are covered in section 3 above. At this time, OSPFv2 and OSPFv3 carry routing information for a single data protocol (IPv4 and IPv6, respectively) so when it is desired to signal a topology change after a BFD session failure, this should be done by tearing down the corresponding OSPF neighbor.

ISIS may be used to support only one data protocol, or multiple data

protocols. [ISIS] specifies a common topology for multiple data protocols, but work is underway to support multiple topologies. If multiple data protocols are advertised in the ISIS Hello, and independent topologies are in use, the failing data protocol should no longer be advertised in ISIS Hello packets in order to signal a lack of connectivity for that protocol. Otherwise, a failing BFD session should be signaled by simulating an ISIS adjacency failure.

OSPF has a planned restart signaling mechanism, whereas ISIS does not. The appropriate mechanisms outlined in section 3.3 should be used.

8.1.3. OSPF Virtual Links

If it is desired to use BFD for failure detction of OSPF Virtual Links, the mechanism described in [BFD-MULTI] MUST be used, since OSPF Virtual Links may traverse an arbitrary number of hops. BFD Authentication SHOULD be used and is strongly encouraged.

8.2. Interactions with BGP

BFD may be useful with EBGP sessions [BGP] in order to more rapidly trigger topology changes in the face of path failure. As noted in section 3.4, it is generally unwise for IBGP sessions to interact with BFD if the underlying IGP is already doing so.

EBGP sessions being advised by BFD may establish either a one hop [BFD-1HOP] or a multihop [BFD-MULTIHOP] session, depending on whether the neighbor is immediately adjacent or not. The BFD session should be established to the BGP neighbor (as opposed to any other Next Hop advertised in BGP.)

[BGP-GRACE] describes a Graceful Restart mechanism for BGP. If Graceful Restart is not taking place on an EBGP session, and the corresponding BFD session fails, the EBGP session should be torn down in accordance with section 3.2. If Graceful Restart is taking place,

the basic procedures in section 3.3 applies. BGP Graceful Restart does not signal planned restarts, so section 3.3.2.2 applies. If Graceful Restart is aborted due to the rules described in section 3.3, the "receiving speaker" should act as if the "restart timer" expired (as described in [BGP-GRACE].)

8.3. Interactions with RIP

The RIP protocol [RIP] is somewhat unique in that, at least as specified, neighbor adjacency state is not stored per se. Rather, installed routes contain a next hop address, which in most cases is the address of the advertising neighbor (but may not be.)

In the case of RIP, when the BFD session associated with a neighbor fails, an expiration of the "timeout" timer for each route installed from the neighbor (for which the neighbor is the next hop) should be simulated.

Note that if a BFD session fails, and a route is received from that neighbor with a next hop address that is not the address of the neighbor itself, the route will linger until it naturally times out (after 180 seconds. However, if an implementation keeps track of all

of the routes received from each neighbor, all of the routes from the neighbor corresponding to the failed BFD session should be timed out, regardless of the next hop specified therein, and thereby avoiding the lingering route problem.

Normative References

[BFD] Katz, D., and Ward, D., "Bidirectional Forwarding Detection", draft-ietf-bfd-base-05.txt, June, 2006.

[BFD-1HOP] Katz, D., and Ward, D., "BFD for IPv4 and IPv6 (Single Hop)", draft-ietf-bfd-v4v6-1hop-05.txt, June, 2006.

[BFD-MPLS] Aggarwal, R., and Kompella, K., "BFD for MPLS LSPs", draft-ietf-bfd-mpls-03.txt, June, 2006.

[BFD-MULTI] Katz, D., and Ward, D., "BFD for Multihop Paths", draft-ietf-bfd-multihop-04.txt, June, 2006.

[BGP] Rekhter, Y., Li, T. et al, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January, 2006.

[BGP-GRACE] Sangli, S., Rekhter, Y., et al, "Graceful Restart Mechanism for BGP", draft-ietf-idr-restart-12.txt, June, 2006.

[DIFFSERV] Nichols, K. et al, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December, 1998.

[ISIS] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, December 1990.

[ISIS-GRACE] Shand, M., and Ginsberg, L., "Restart signaling for IS-IS", RFC 3847, July 2004.

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

[OSPFv2] Moy, J., "OSPF Version 2", RFC 2328, April 1998.

[OSPFv3] Coltun, R., et al, "OSPF for IPv6", RFC 2740, December 1999.

[OSPF-GRACE] Moy, J., et al, "Graceful OSPF Restart", RFC 3623, November 2003.

[RIP] Malkin, G., "RIP Version 2", RFC 2453, November, 1998.

Security Considerations

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

IANA Considerations

This document has no actions for IANA.

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 Systems
170 W. Tasman Dr.
San Jose, CA 95134 USA
Phone: +1-408-526-4000
Email: dward@cisco.com

Changes from the previous draft

Control protocol-specific interactions were moved from [BFD-1HOP] to this document, and brief mentions of BGP and RIP were added.

IPR Disclaimer

pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Full Copyright Notice

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

This document expires in December, 2006.