Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: March 6, 2015

N. Akiya
M. Binderberger
Cisco Systems
G. Mirsky
Ericsson

September 2, 2014

Common Interval Support in Bidirectional Forwarding Detection
draft-ietf-bfd-intervals-04

Abstract

Bidirectional Forwarding Detection (BFD) requires that messages are transmitted at regular intervals and provides a way to negotiate the interval used by BFD peers. Some BFD implementations may be restricted to only support several interval values. When such BFD implementations speak to each other, there is a possibility of two sides not being able to find a common value for the interval to run BFD sessions.

This document defines a small set of interval values for BFD that we call "Common Intervals", and recommends implementations to support the defined intervals. This solves the problem of finding an interval value that both BFD speakers can support while allowing a simplified implementation as seen for hardware-based BFD. It does not restrict an implementation from supporting more intervals in addition to the Common Intervals.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Table of Contents

1. Introduction

The Bidirectional Forwarding Detection (BFD) standard [RFC5880] describes how to calculate the transmission interval and the detection time. It does not make any statement though how to solve a situation where one BFD speaker cannot support the calculated value. In practice this may not been a problem as long as software-implemented timers have been used and as long as the granularity of such timers was small compared to the interval values being supported, i.e. as long as the error in the timer interval was small compared to 25 percent jitter.

In the meantime requests exist for very fast interval values, down to 3.3msec for MPLS-TP. At the same time the requested scale for the number of BFD sessions is increasing. Both requirements have driven vendors to use Network Processors (NP), FPGAs or other hardware-based solutions to offload the periodic packet transmission and the timeout detection in the receive direction. A potential problem with this hardware-based BFD is the granularity of the interval timers. Depending on the implementation only a few intervals may be supported, which can cause interoperability problems. This document proposes a set of interval values that should be supported by all implementations. Details are laid out in the following sections.

2. The problem with few supported intervals

Let's assume vendor "A" supports 10msec, 100msec and 1sec interval timers in hardware. Vendor "B" supports every value from 20msec onward, with a granularity of 1msec. For a BFD session "A" tries to set up the session with 10msec while "B" uses 20msec as the value for RequiredMinRxInterval and DesiredMinTxInterval. [RFC5880] describes that the negotiated value for Rx and Tx is 20msec. But system "A" is not able to support this. Multiple ways exist to resolve the dilemma but none of them is without problems.

a. Realizing that it cannot support 20msec, system "A" sends out a new BFD packet, advertising the next larger interval of 100msec with RequiredMinRxInterval and DesiredMinTxInterval. The new negotiated interval between "A" and "B" then is 100msec, which is supported by both systems. The problem though is that we moved from the 10/20msec range to 100msec, which has far deviated from operator expectations.

b. System "A" could violate [RFC5880] and use the 10msec interval for the Tx direction. In the receive direction it could use an adjusted multiplier value M' = 2 * M to match the correct detection time. Now beside the fact that we explicitly violate [RFC5880] there may be the problem that system "B" drops up to 50% of the packets; this could be the case when "B" uses an ingress rate policer to protect itself and the policer would be programmed with an expectation of 20msec receive intervals.

The example above could be worse when we assume that system "B" can only support a few timer values itself. Let's assume "B" supports "20msec", "300msec" and "1sec". If both systems would adjust their advertised intervals, then the adjustment ends at 1sec. The example above could even be worse when we assume that system "B" can only support "50msec", "500msec" and "2sec". Even if both systems walk their supported intervals, the two systems will never be able to agree on a interval to run any BFD sessions.

3. Well-defined, Common Intervals

The problem can be reduced by defining interval values that are supported by all implementations. Then the adjustment mechanism could find a commonly supported interval without deviating too much from the original request.

In technical terms the requirement is as follows: a BFD implementation SHOULD support all values in the set of Common Interval values which are equal to or larger than the fastest, i.e. lowest, interval the particular BFD implementation supports.

The proposed set of Common Interval values is: 3.3msec, 10msec, 20msec, 50msec, 100msec and 1sec.

In addition support for 10sec interval together with multiplier values up to 255 is recommended to support graceful restart.

The adjustment is always towards larger, i.e. slower, interval values when the initial interval proposed by the peer is not supported.

This document is not adding new requirements with respect to the precision with which a timer value must be implemented. Supporting an interval value means to advertise this value in the DesiredMinTxInterval and/or RequiredMinRxInterval field of the BFD packets and to provide timers that are reasonably close. [RFC5880] defines safety margins for the timers by defining a jitter range.

How is the "Common Interval" set used exactly? In the example above, vendor "A" has a fastest interval of 10msec and thus would be required to support all intervals in the Common Interval set that are equal or larger than 10msec, i.e. it would support 10msec, 20msec, 50msec, 100msec, 1sec. Vendor "B" has a fastest interval of 20msec and thus would need to support 20msec, 50msec, 100msec and 1sec. As long as this requirement is met for the common set of values, then both vendor "A" and "B" are free to support additional values outside of the Common Interval set.

4. IANA Considerations

RFC Ed.: RFC-editor please remove this section

No request to IANA.

5. Security Considerations

This document does not introduce any additional security concerns. The security considerations described in the BFD documents, [RFC5880] and others, apply to devices implementing the BFD protocol, regardless of whether or not the Common Interval set is implemented.

6. Acknowledgements

We would like to thank Sylvain Masse and Anca Zamfir for bringing up the discussion about the Poll sequence, and Jeffrey Haas helped finding the fine line between "exact" and "pedantic".

7. References

7.1. Normative References

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

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, June 2010.

7.2. Informative References

[G.8013_Y.1731]
ITU-T G.8013/Y.1731, "ITU-T OAM functions and mechanisms for Ethernet based network", November 2013.

[GR-253-CORE]
Telcordia Technologies, Inc., "Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria", October 2009.

Appendix A. Why some values are in the Common Interval set

The list of Common Interval values is trying to balance various objectives. The list should not contain too many values as more timers may increase the implementation costs. On the other hand less values produces larger gaps and adjustment jumps. More values in the lower interval range is thus seen as critical to support customer needs for fast detection in setups with multiple vendors.

- o 3.3msec: required by MPLS-TP, adopting the detection time of 10msec from [GR-253-CORE].

- o 10msec: general consensus is to support 10msec. Multiple vendors plan to or do already implement 10msec.

- o 20msec: basically avoids a larger gap in this critical interval region. Still allows 50-60msec detect and restore (with multiplier of 2) and covers existing software-based implementations.

- o 50msec: widely deployed interval. Supporting this value reflects reality of many BFD implementations today.

- o 100msec: similar to 10msec this value allows the reuse of [G.8013_Y.1731] implementations, especially hardware. It allows to support large scale of 9 x 100msec setups and would be a replacement for 3 x 300msec configurations used by customers to have a detection time slightly below 1sec for VoIP setups.

- o 1sec: as mentioned in [RFC5880]. While the interval for Down packets can be 1sec or larger this draft proposes to use exactly 1sec to avoid interoperability issues.

The proposed value for large intervals is 10sec, allowing for a timeout of 42.5 minutes with a multiplier of 255. This value is kept outside the Common Interval set as it is not required for normal BFD operations, which occur in the sub-second range. Instead the expected usage is for graceful restart, if needed.

Appendix B. Timer adjustment with non-identical interval sets

[RFC5880] implicitly assumes that a BFD implementation can support any timer value equal or above the advertised value. When a BFD speaker starts a poll sequence then the peer must reply with the Final (F) bit set and adjust the transmit and detection timers accordingly. With contiguous software-based timers this is a valid assumption. Even in the case of a small number of supported interval values this assumption holds when both BFD speakers support exactly the same interval values.

But what happens when both speakers support intervals that are not supported by the peer? An example is router "A" supporting the Common Interval set plus 200msec while router "B" support the Common Intervals plus 300msec. Assume both routers are configured and run at 50msec. Now router A is configured for 200msec. We know the result must be that both BFD speaker use 1sec timers but how do they reach this endpoint?

First router A is sending a packet with 200msec. The P bit is set according to [RFC5880]. The Tx timer stays at 50msec, the detection timer is 3 * 200msec:

    (A) DesiredTx: 200msec, MinimumRx: 200msec, P-bit

   Tx: 50msec , Detect: 3 * 200msec

Router B now must reply with an F bit. The problem is B is confirming timer values which it cannot support. The only setting to avoid a session flap would be

   (B) DesiredTx: 300msec, MinimumRx: 300msec, F-bit
   Tx: 50msec , Detect: 3 * 300msec

immediately followed by a P-bit packet as the advertised timer values have been changed:

   (B) DesiredTx: 300msec, MinimumRx: 300msec, P-bit
   Tx: 50msec , Detect: 3 * 300msec

This is not exactly what [RFC5880] states in section 6.8.7 about the transmission rate. On the other hand as we will see this state does not last for long. Router A would adjust its timers based on the received Final bit

   (A) Tx: 200msec , Detect: 3 * 1sec

Router A is not supporting the proposed 300msec and would use 1sec instead for the detection time. It would then respond to the received Poll sequence from router B, using 1sec as router A does not support the Max(200msec, 300msec):

   (A) DesiredTx: 1sec, MinimumRx: 1sec, F-bit
   Tx: 200msec , Detect: 3 * 1sec

followed by its own Poll sequence as the advertised timer values have been changed:

   (A) DesiredTx: 1sec, MinimumRx: 1sec, P-bit
   Tx: 200msec , Detect: 3 * 1sec

Router B would adjust its timers based on the received Final

   (B) Tx: 300msec , Detect: 3 * 1sec

and would then reply to the Poll sequence from router A:

   (B) DesiredTx: 300msec, MinimumRx: 300msec, F-bit
   Tx: 1sec , Detect: 3 * 1sec

which finally makes router A adjusting its timers:

   (A) Tx: 1sec , Detect: 3 * 1sec

In other words router A and B go through multiple poll sequences until they reach a commonly supported interval value. Reaching such a value is guaranteed by this draft.

Authors' Addresses

Nobo Akiya
Cisco Systems

Email: nobo@cisco.com

Marc Binderberger
Cisco Systems

Email: mbinderb@cisco.com

Greg Mirsky
Ericsson

Email: gregory.mirsky@ericsson.com