

Network Working Group
Internet-Draft
Updates: 5880 (if approved)
Intended status: Standards Track
Expires: 30 September 2022

M. Jethanandani
Kloud Services
S. Agarwal
Cisco Systems, Inc
A. Mishra
O3b Networks
A. Saxena
Ciena Corporation
A. Dekok
Network RADIUS SARL
29 March 2022

Secure BFD Sequence Numbers
draft-ietf-bfd-secure-sequence-numbers-09

Abstract

This document describes two new BFD Authentication mechanism, Meticulous Keyed ISAAC, and Meticulous Keyed FNV1A. These mechanisms can be used to authenticate BFD packets, and secure the sequence number exchange, with less CPU time cost than using MD5 or SHA1, with the tradeoff of decreased security. This document updates RFC 5880.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 30 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Meticulous Keyed ISAAC	3
4. Meticulous Keyed FNV1A	5
5. Operation	6
5.1. Seeding and Operation of ISAAC	7
5.2. Secret Key	8
5.3. Seeding ISAAC	8
6. Meticulous Keyed ISAAC Authentication	9
7. Meticulous Keyed FNV1A Authentication	10
7.1. Calculation of the FNV-1a Digest	12
8. IANA Considerations	12
9. Security Considerations	13
9.1. Spoofing	14
9.2. Re-Use of keys	14
10. Acknowledgements	15
11. References	15
11.1. Normative References	15
11.2. Informative References	15
Authors' Addresses	15

1. Introduction

BFD [RFC5880] defines a number of authentication mechanisms, including Simple Password (Section 6.7.2), and various other methods based on MD5 and SHA1 hashes. The benefit of using cryptographic hashes is that they are secure. The downside to cryptographic hashes is that they are expensive and time consuming on resource-constrained hardware.

When BFD packets are unauthenticated, it is possible for an attacker to forge, modify, and/or replay packets on a link. These attacks have a number of side effects. They can cause parties to believe that a link is down, or they can cause parties to believe that the link is up when it is, in fact, down. The goal of these methods is to prevent spoofing of the BFD session by someone who could guess the next sequence number. We therefore define simple and fast Auth Type methods which allow parties to detect and prevent both spoofed sequence numbers, and spoofed packets.

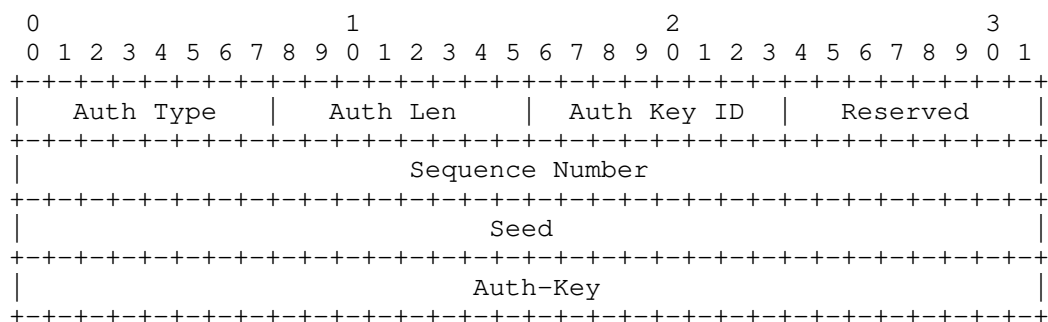
This document proposes the use of Authentication methods which provides meticulous keying, but which have less impact on resource constrained systems. The algorithms chosen are ISAAC [ISAAC], which is a fast cryptographic random number generator, and FNV-1a FNV1A [FNV1A] which is a fast (but non-cryptographic) hash. ISAAC has been subject to significant cryptanalysis in the past thirty years, and has not yet been broken. Similarly, FNV-1a is fast, and while not cryptographically secure, it is has good hashing properties.

2. 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].

3. Meticulous Keyed ISAAC

If the Authentication Present (A) bit is set in the header, and the State (Sta) field equals 3 (Up), and the Authentication Type field contains TB1 (Meticulous Keyed ISAAC), the Authentication Section has the following format:



Auth Type

The Authentication Type, which in this case is TB1 (Meticulous Keyed ISAAC). If the State (Sta) field value is not 3 (Up), then Meticulous Keyed ISAAC MUST NOT be used.

Auth Len

The length of the Authentication Section, in bytes. For Meticulous Keyed ISAAC authentication, the length is 16.

Auth Key ID

The authentication key ID in use for this packet. This allows multiple keys to be active simultaneously.

Reserved

This byte MUST be set to zero on transmit, and ignored on receipt.

Sequence Number

The sequence number for this packet. For Meticulous Keyed ISAAC Authentication, this value is incremented for each successive packet transmitted for a session. This provides protection against replay attacks.

Seed

A 32-bit (4 octet) seed which is used in conjunction with the shared key in order to configure and initialize the ISAAC pseudo-random-number-generator (PRNG). It is used to identify and distinguish "streams" of random numbers which are generated by ISAAC.

Auth-Key

This field carries the 32-bit (4 octet) ISAAC output which is associated with the Sequence Number. The ISAAC PRNG MUST be configured and initialized as given in section TBD.

Note that the Auth-Key here does not include any summary or hash of the packet. The packet itself is completely unauthenticated.

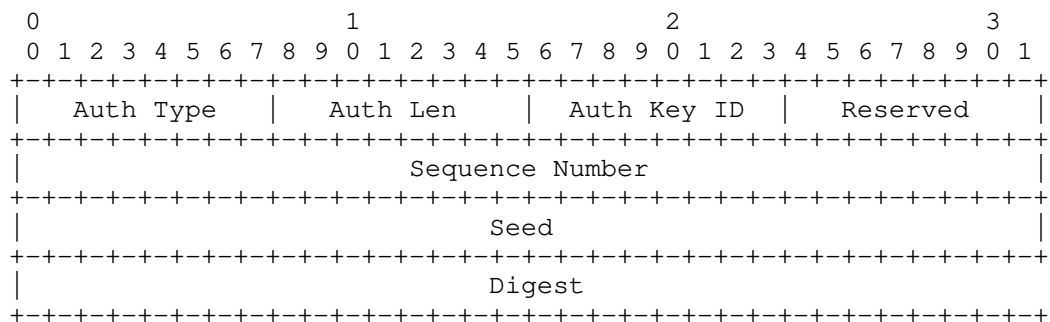
The purpose of this method is to secure the sequence number exchange, and to both detect and prevent spoofing of sequence numbers. In some cases, it is acceptable to not authenticate the entire packet, in which case this method may be used.

When the receiving party receives a BFD packet with an expected sequence number, and the correct corresponding ISAAC output, it knows that only the authentic sending party could have sent that message. The sending party is therefore alive/up, and intended to send the message.

While the rest of the contents of the BFD packet are unauthenticated and may be modified by an attacker, the same is true of stronger Auth Types, such as MD5 or SHA1. The Auth Type methods are not designed to prevent such attacks. Instead, they are designed to prevent an attacker from spoofing identities, and an attacker from artificially keeping a session "Up".

4. Meticulous Keyed FNV1A

If the Authentication Present (A) bit is set in the header, and the State (Sta) field equals 3 (Up), and the Authentication Type field contains TB2 (Meticulous Keyed FNV1A), the Authentication Section has the following format:



Auth Type

The Authentication Type, which in this case is TB2 (Meticulous Keyed FNV1A). If the State (Sta) field value is not 3 (Up), then Meticulous Keyed FNV1A MUST NOT be used.

Auth Len

The length of the Authentication Section, in bytes. For Meticulous Keyed FNV1A authentication, the length is 16.

Auth Key ID

The authentication key ID in use for this packet. This allows multiple keys to be active simultaneously.

Reserved

This byte MUST be set to zero on transmit, and ignored on receipt.

Sequence Number

The sequence number for this packet. For Meticulous Keyed FNV1A Authentication, this value is incremented for each successive packet transmitted for a session. This provides protection against replay attacks.

Seed

A 32-bit (4 octet) seed which is used in conjunction with the shared key in order to configure and initialize the ISAAC PRNG. It is also used to identify and distinguish "streams" of random numbers which are generated by ISAAC.

Digest

This field carries the 32-bit (4 octet) FNV1A digest associated with the Sequence Number. The ISAAC PRNG MUST be configured and initialized as given in section TBD.

Note that the ISAAC PRNG output is still used with this authentication type. The FNV1A hash is fast, but it is not secure. In order to reach an acceptable level of security with FNV1A, we use ISAAC to generate secure per-packet "signing keys". These per-packet keys are then used with FNV1A in order to perform a keyed of hash the packet, and therefore create the Digest.

5. Operation

BFD requires fast and reasonably secure authentication of messages which are exchanged. Methods using MD5 or SHA1 are CPU intensive, and can negatively impact systems with limited CPU power.

We use ISAAC here as a way to generate an infinite stream of pseudo-random numbers. With Meticulous Keyed ISAAC, these numbers are used as a signal that the sending party is authentic. That is, only the sending party can generate the numbers. Therefore if the receiving party sees a correct number, then only the sending party could have generated that number. The sender is therefore authentic, even if the packet contents are not necessarily trusted.

Note that since the packets are not signed with this authentication type, the Meticulous Keyed ISAAC method MUST NOT be used to signal BFD state changes. For BFD state changes, and a more optimized way

to authenticate packets, please refer to BFD Authentication [I-D.ietf-bfd-optimizing-authentication]. Instead, the packets containing Meticulous Keyed ISAAC are only a signal that the sending party is still alive, and that the sending party is authentic. That is, these Auth Type methods must only be used when `bfd.SessionState=Up`, and the State (Sta) field equals 3 (Up).

If slightly more security is desired, the packets can be authenticated via the Meticulous Keyed FNV1A method. This method is similar to the Meticulous Keyed ISAAC authentication type, except that the FNV-1A hash function is used to hash a combination of the packet, and per-packet ISAAC pseudo-random number. If the receiving party is able to validate the hash, then the receiver knows both that the sender is authentic, and that the packet contents have likely not been modified.

As this hash function is not very secure, this method can be used only in situations where the Meticulous Keyed ISAAC method can be used. The Meticulous Keyed FNV1A method MUST NOT be used to signal BFD state changes.

5.1. Seeding and Operation of ISAAC

The ISAAC PRNG state is initialized with the 32-bit Seed, followed by the secret key, and then the rest of the state is filled with zeros. The internal state of ISAAC is 1024 bytes, so the secret key is limited to 1020 bytes in length.

The origin of the Seed field is discussed later in this document. For now, we note that each time a new Seed is used, the `bfd.XmitAuthSeq` value MUST be set to zero.

Once the state has been initialized, the standard ISAAC initial mixing function is run. Once this operation has been performed, ISAAC will be able to produce 256 random numbers at near-zero cost. When all 256 numbers are consumed, the ISAAC mixing function is run, which then results in another set of 256 random numbers

ISAAC can be thought of here as producing an infinite stream of numbers, based on a secret key, where the numbers are produced in "pages" of 256 32-bit values. This property of ISAAC allows for essentially zero-cost "seeking" within a page. The expensive operation of mixing is performed only once per 256 packets, which means that most BFD packet exchanges can be fast and efficient.

The Sequence number is used to "seek" within a the stream of 32-bit numbers produced by ISAAC. The sending party increments the Sequence Number on every packet sent, to indicate to the receiving party where it is in the sequence.

The receiving party can then look at the Sequence Number to determine which particular PRNG value is being used in the packet. The Sequence Number thus permits the two parties to synchronise if/when a packet or packets are lost. Incrementing the Sequence Number for every packet also prevents the re-use of any individual pseudo-random number which was derived from ISAAC.

The Sequence Number can increment without bounds, though it can wrap once it reaches the limit of the 32-bit counter field. ISAAC has a cycle length of 2^{8287} , so there is no issue with using more than 2^{32} values from it.

The result of the above operation is an infinite series of numbers which are unguessable, and which can be used to authenticate the sending party.

5.2. Secret Key

For interoperability, the management interface by which the key is configured MUST accept ASCII strings, and SHOULD also allow for the configuration of any arbitrary binary string in hexadecimal form. Other configuration methods MAY be supported.

The secret Key is mixed with the Seed before being used in ISAAC. If instead ISAAC was initialized without a Seed, then an attacker could pre-compute ISAAC states for many keys, and perform an off-line dictionary attack. The addition of the Seed makes these attacks infeasible.

As a result, it is safe to use the same secret Key for the Auth Types defined here, and also for other Auth Types.

5.3. Seeding ISAAC

The value of the Seed field SHOULD be derived from a secure source. Exactly how this can be done is outside of the scope of this document.

The Seed value SHOULD remain the same for the duration of a BFD session. The Seed value MAY change when the BFD state changes.

If the sending party changes its Seed value, `bfd.XmitAuthSeq` value MUST be set to zero, otherwise the receiving party would be unable to synchronize its sequence of numbers produced by the ISAAC generator. There is no way to signal or negotiate Seed changes. The receiving party MUST remember the current Seed value, and then detect if the Seed changes. Note that the Seed value MUST NOT change unless sending party has signalled a BFD state change with a packet that is authenticated using a more secure Auth Type method.

6. Meticulous Keyed ISAAC Authentication

In this method of authentication, one or more secret keys (with corresponding key IDs) are configured in each system. One of the keys is used to seed the ISAAC PRNG. The output of ISAAC (I) is used to signal that the sender is authentic. To help avoid replay attacks, a sequence number is also carried in each packet. For Meticulous Keyed ISAAC, the sequence number is incremented on every packet.

The receiving system accepts the packet if the key ID matches one of the configured Keys, the Auth-Key derived from the selected Key, Seed, and Sequence Number matches the Auth-Key carried in the packet, and the sequence number is strictly greater than the last sequence number received (modulo wrap at 2^{32})

Transmission Using Meticulous Keyed ISAAC Authentication

The Auth Type field MUST be set to TBD1 (Meticulous Keyed ISAAC). The Auth Len field MUST be set to 16. The Auth Key ID field MUST be set to the ID of the current authentication key. The Sequence Number field MUST be set to `bfd.XmitAuthSeq`.

The Seed field MUST be set to the value of the current seed used for this sequence.

The Auth-Key field MUST be set to the output of ISAAC, which depends on the secret Key, the current Seed, and the Sequence Number.

For Meticulous Keyed ISAAC, `bfd.XmitAuthSeq` MUST be incremented on each packet, in a circular fashion (when treated as an unsigned 32-bit value). The `bfd.XmitAuthSeq` MUST NOT be incremented by more than one for a packet.

Receipt using Meticulous Keyed ISAAC Authentication

If the received BFD Control packet does not contain an Authentication Section, or the Auth Type is not correct (TBD2 for Meticulous Keyed ISAAC), then the received packet MUST be discarded.

If the Auth Key ID field does not match the ID of a configured authentication key, the received packet MUST be discarded.

If the Auth Len field is not equal to 16, the packet MUST be discarded.

If the Seed field does not match the current Seed value, the packet MUST be discarded.

If `bfd.AuthSeqKnown` is 1, examine the Sequence Number field. For Meticulous Keyed FNV1A, if the sequence number lies outside of the range of `bfd.RcvAuthSeq+1` to `bfd.RcvAuthSeq+(3*Detect Mult)` inclusive (when treated as an unsigned 32-bit circular number space) the received packet MUST be discarded.

Calculate the current expected output of ISAAC, which depends on the secret Key, the current Seed, and the Sequence Number. If the value does not matches the Auth-Key field, then the packet MUST be discarded.

Note that in some cases, calculating the expected output of ISAAC will result in the creation of a new "page" of 256 numbers. This process will irreversible, and will destroy the current "page". As a result, if the generation of a new output will create a new "page", the receiving party MUST save a copy of the entire ISAAC state before proceeding with this calculation. If the outputs match, then the saved copy can be discarded, and the new ISAAC state is used. If the outputs do not match, then the saved copy MUST be restored, and the modified copy discarded.

7. Meticulous Keyed FNV1A Authentication

Where slightly more security is needed, the sender can use Meticulous Keyed FNV1A. In this method, each packet is signed with a non-cryptographic hash, FNV-1a [FNV1A]. This hash is reasonably fast, it has good distribution, and collisions are rare. However, it is linear, and potentially reversible. In addition, its output is only 32 bits, and it is not cryptographically strong.

In this methods of authentication, one or more secret keys (with corresponding key IDs) are configured in each system. One of the keys is included in an FNV1A digest calculated over the outgoing BFD Control packet, but the Key itself is not carried in the packet. To

help avoid replay attacks, a sequence number is also carried in each packet. For Meticulous Keyed FNV1A, the sequence number is incremented on every packet.

The receiving system accepts the packet if the key ID matches one of the configured Keys, an FNV-1a digest including the selected key matches the digest carried in the packet, and the sequence number is strictly greater than the last sequence number received (modulo wrap at 2^{32})

Transmission Using Meticulous Keyed FNV1A Authentication

The Auth Type field MUST be set to TBD2 (Meticulous Keyed FNV1A). The Auth Len field MUST be set to 16. The Auth Key ID field MUST be set to the ID of the current authentication key. The Sequence Number field MUST be set to `bfd.XmitAuthSeq`.

The Digest field MUST be set to the value of the FNV-1a digest, as described below.

For Meticulous Keyed FNV1A, `bfd.XmitAuthSeq` MUST be incremented on each packet, in a circular fashion (when treated as an unsigned 32-bit value). The `bfd.XmitAuthSeq` MUST NOT be incremented by more than one for a packet.

Receipt Using Meticulous Keyed FNV1A Authentication

If the received BFD Control packet does not contain an Authentication Section, or the Auth Type is not correct (TBD2 for Meticulous Keyed FNV1A), then the received packet MUST be discarded.

If the Auth Key ID field does not match the ID of a configured authentication key, the received packet MUST be discarded.

If the Auth Len field is not equal to 16, the packet MUST be discarded.

If the Seed field does not match the current Seed value, the packet MUST be discarded.

If `bfd.AuthSeqKnown` is 1, examine the Sequence Number field. For Meticulous Keyed FNV1A, if the sequence number lies outside of the range of `bfd.RcvAuthSeq+1` to `bfd.RcvAuthSeq+(3*Detect Mult)` inclusive (when treated as an unsigned 32-bit circular number space) the received packet MUST be discarded.

Otherwise (bfd.AuthSeqKnown is 0), bfd.AuthSeqKnown MUST be set to 1, and bfd.RcvAuthSeq MUST be set to the value of the received Sequence Number field.

Replace the contents of the Digest field with zeros, and calculate the FNV-1a digest as described below. If the calculated FNV-1a digest is equal to the received value of the Digest field, the received packet MUST be accepted. Otherwise (the digest does not match the Digest field), the received packet MUST be discarded.

7.1. Calculation of the FNV-1a Digest

Unlike other authentication mechanisms, the user-supplied key is not placed into the Auth Key / Digest field, and the packet hashed. As FNV-1a is not a cryptographic hash, such a process would simplify the process for an attacker to "crack" the key.

Instead, for a particular packet "P", and ISAAC pseudo-random number "I", the FNV1A digest "D" is calculated as shown below, where "+" indicates concatenation.

$$D = \text{FNV1A}(I + P + I)$$

Where "+" denotes concatenation. We also note that the Digest field of the packet MUST be initialized to all zeroes before this calculation is performed

The calculated value "D" is then inserted into the packet in the Digest field, and the packet is sent as normal. The receiving party reverses this operation in order to validate the packet.

8. IANA Considerations

This document asks that IANA allocate a new entry in the "BFD Authentication Types" registry.

Address - TBD1

BFD Authentication Type Name - Meticulous Keyed ISAAC

Reference - this document

Address - TBD2

BFD Authentication Type Name - Meticulous Keyed FNV1A

Reference - this document

Note to RFC Editor: this section may be removed on publication as an RFC.

9. Security Considerations

The security of this proposal depends strongly on the length of the secret, and the entropy of the key. It is RECOMMENDED that the key be 16 octets in length or more.

The security of this proposal depends strongly on ISAAC. This generator has been analyzed and has not been broken. Research shows few other CSRNGs which are as simple and as fast as ISAAC. For example, many other generators are based on AES, which is infeasible for resource constrained systems.

The security of this proposal depends on the strength of the FNV-1a hash algorithm. Folding the output of ISAAC into the hash limits the ability of an attacker to reverse the hash, or to perform off-line dictionary attacks. Even if one particular 32-bit per-packet key is found via brute force, that information will be useless, as the next packet will use a different key. And since ISAAC is secure, knowledge of one particular key will give an attacker no ability to predict the next key.

In a keyed algorithm, the key is shared between the two systems. Distribution of this key to all the systems at the same time can be quite a cumbersome task. BFD sessions running a fast rate will require these keys to be refreshed often, which poses a further challenge. Therefore, it is difficult to change the keys during the operation of a BFD session without affecting the stability of the BFD session. Therefore, it is recommended to administratively disable the BFD session before changing the keys.

This method allows the BFD end-points to detect a malicious packet, as the calculated hash value will not match the value found in the packet. The behavior of the session, when such a packet is detected, is based on the implementation. A flood of such malicious packets may cause a BFD session to be operationally down.

As noted earlier with Meticulous Keyed FNV1A, each packet is associated with a unique, per-packet key. This process means that even if an observer sees the Auth-Key, or the FNV-1a hash for one packet, the only information gained will be a key which is never be re-used, and will therefore be useless to an attacker. Further, even if the attacker can "crack" a sequence of packets to obtain a stream of keys, the cryptographic nature of ISAAC makes it impossible for the attacker to derive the input key which is used to "seed" the ISAAC state.

The particular method of hashing was chosen because of the non-cryptographic and reversible nature of the FNV-1a hash. If the digest had been calculated any other way, then an attacker would have significantly less work to do in order to "crack" the hash. In short the per-packet key protects the hash, and the hash protects the per-packet key.

We believe that this construction is reasonably secure, given the constraints. If cryptographic security is desired, then implementors can use MD5 or SHA1 authentication mechanisms

9.1. Spoofing

When Meticulous Keyed ISAAC is used, it is possible for an attacker who can see the packets to observe a particular Auth Key value, and then copy it to a different packet as a "man-in-the-middle" attack. However, the usefulness of such an attack is limited by the requirements that these packets must not signal state changes in the BFD session, and that the key changes on every packet.

Performing such an attack would require an attacker to have the following information and capabilities:

- This is man-in-the-middle active attack.

- The attacker has the contents of a stable packet

- The attacker has managed to deduce the ISAAC key and knows which per-packet key is being used.

The attack is therefore limited to keeping the BFD session up when it would otherwise drop.

However, the usual actual attack which we are protecting BFD from is availability. That is, the attacker is trying to shut down then connection when the attacked parties are trying to keep it up. As a result, the attacks here seem to be irrelevant in practice.

9.2. Re-Use of keys

The strength of the Auth-Type methods is significantly different between the strong one like SHA-1 and ISAAC. While ISAAC has had cryptanalysis, and has not been shown to be broken, that analysis is limited. The question then is whether or not it is safe to use the same key for both Auth Type methods (SHA1 and ISAAC), or should we require different keys for each method?

If we recommend different keys, then it is possible for the two keys to be configured differently on each side of a BFD lin. For example. the strong key can be properly provisioned, which allows to the BFD state machine to advance to Up, Then, when we switch to the weaker Auth Type which uses a different key, that key may not match, and the session will immediatly drop.

We believe that the use of the same key is acceptable, as the Auth Types which use ISAAC also depend on a Seed. The use of the Seed increases the difficulty of breaking the key, and makes off-line dictionary attacks infeasible.

10. Acknowledgements

The authors would like to thank Jeff Haas and Reshad Rahman for their reviews of and suggestions for the document.

11. References

11.1. 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.

11.2. Informative References

- [FNV1A] Noll, L. C., "FNV-1a", <http://www.isthe.com/chongo/tech/comp/fnv/index.html#FNV-1a>, 2013.
- [I-D.ietf-bfd-optimizing-authentication] Jethanandani, M., Mishra, A., Saxena, A., and M. Bhatia, "Optimizing BFD Authentication", Work in Progress, Internet-Draft, draft-ietf-bfd-optimizing-authentication-13, 1 August 2021, <<https://www.ietf.org/archive/id/draft-ietf-bfd-optimizing-authentication-13.txt>>.
- [ISAAC] Jenkins, R. J., "ISAAC", <http://www.burtleburtle.net/bob/rand/isaac.html>, 1996.

Authors' Addresses

Mahesh Jethanandani
Kloud Services
Email: mjethanandani@gmail.com

Sonal Agarwal
Cisco Systems, Inc
170 W. Tasman Drive
San Jose, CA 95070
United States of America
Email: agarwaso@cisco.com
URI: www.cisco.com

Ashesh Mishra
O3b Networks
Email: mishra.ashesh@gmail.com

Ankur Saxena
Ciena Corporation
3939 North First Street
San Jose, CA 95134
United States of America
Email: ankurpsaxena@gmail.com

Alan DeKok
Network RADIUS SARL
100 CentrepoinTE Drive #200
Ottawa ON K2G 6B1
Canada
Email: aland@freeradius.org

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 14, 2021

A. Mishra
SES
M. Jethanandani
Kloud Services
A. Saxena
Ciena Corporation
S. Pallagatti
VMware
M. Chen
Huawei
P. Fan
China Mobile
April 12, 2021

BFD Stability
draft-ietf-bfd-stability-10

Abstract

This document describes extensions to the Bidirectional Forwarding Detection (BFD) protocol to measure BFD stability. Specifically, it describes a mechanism for detection of BFD packet loss.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on October 14, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Use Cases	3
4. BFD NULL-Authentication Type	3
5. Theory of Operation	3
5.1. Loss Measurement	4
6. ietf-bfd-stability YANG Module	4
6.1. Data Model Overview	4
6.2. YANG Module	5
7. IANA Considerations	9
7.1. The "IETF XML" Registry	9
7.2. The "YANG Module Names" Registry	9
8. Security Consideration	9
9. Contributors	10
10. Acknowledgements	10
11. References	10
11.1. Normative References	10
11.2. Informative References	12
Authors' Addresses	12

1. Introduction

The Bidirectional Forwarding Detection (BFD) [RFC5880] protocol operates by transmitting and receiving BFD control packets, generally at high frequency, over the datapath being monitored. In order to prevent significant data loss due to a datapath failure, BFD session detection time as defined in BFD [RFC5880] is set to the smallest feasible value.

This document proposes a mechanism to detect lost packets in a BFD session in addition to the datapath fault detection mechanisms of BFD. Such a mechanism presents significant value to measure the stability of BFD sessions and provides data to the operators for the cause of a BFD failure.

This document does not propose any BFD extension to measure data traffic loss or delay on a link or tunnel and the scope is limited to BFD packets.

2. Terminology

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 RFC 2119 [RFC2119] and RFC 8174 [RFC8174].

The reader is expected to be familiar with the BFD [RFC5880], Optimizing BFD Authentication [I-D.ietf-bfd-optimizing-authentication] and BFD Secure Sequence Numbers [I-D.ietf-bfd-secure-sequence-numbers].

3. Use Cases

Bidirectional Forwarding Detection as defined in BFD [RFC5880] cannot detect any BFD packet loss if the loss does not last for detection time. This document proposes a method to detect a dropped packet on the receiver. For example, if the receiver receives BFD control packet k at time t but receives packet k+3 at time t+10ms, and never receives packet k+1 and/or k+2, then it has experienced a drop.

This proposal enables BFD implementations to generate diagnostic information on the health of each BFD session that could be used to preempt a failure on a datapath that BFD was monitoring by allowing time for a corrective action to be taken.

In a faulty datapath scenario, an operator can use BFD health information to trigger delay and loss measurement OAM protocol, Connectivity Fault Management (CFM) [IEEE802.1ag] or Loss Measurement (LM)-Delay Measurement (DM)) as defined by A One-way Active Measurement Protocol (OWAMP) [RFC4656] to further isolate the issue.

4. BFD NULL-Authentication Type

The functionality proposed for BFD stability measurement is achieved by appending an authentication section with the NULL Authentication type (as defined in Optimizing BFD Authentication [I-D.ietf-bfd-optimizing-authentication]) to the BFD control packets that do not have authentication enabled.

5. Theory of Operation

This mechanism allows operators to measure the loss of BFD control packets.

When using MD5 or SHA authentication, BFD uses an authentication section that carries the Sequence Number. However, if non-meticulous authentication is being used, or no authentication is in use, then

the non-authenticated BFD control packets MUST include an authentication section with the NULL Authentication type.

5.1. Loss Measurement

Loss measurement counts the number of BFD control packets missed at the receiver during any Detection Time period. The loss is detected by comparing the Sequence Number field in the Auth TLV (NULL or otherwise) in successive BFD control packets. The Sequence Number in each successive control packet generated on a BFD session by the transmitter is incremented by one. This loss count can then be exposed using the YANG module defined in the subsequent section.

The first BFD authentication section with a non-zero sequence number, in a valid BFD control packet, processed by the receiver is used for bootstrapping the logic. When using secure sequence numbers, if the expected values are pre-calculated, the value must be matched to detect lost packets as defined in BFD secure sequence numbers [I-D.ietf-bfd-secure-sequence-numbers].

6. ietf-bfd-stability YANG Module

6.1. Data Model Overview

This YANG module augments the "ietf-bfd" module to add to the per-session set of counters a 'loss-packet-count' for BFD packets that are lost but do not necessarily result in the BFD session going down.

```

module: ietf-bfd-stability
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh
    /bfd-ip-sh:sessions/bfd-ip-sh:session
    /bfd-ip-sh:session-statistics:
    +--ro lost-packet-count?   yang:counter32
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/bfd:bfd/bfd-ip-mh:ip-mh
    /bfd-ip-mh:session-groups/bfd-ip-mh:session-group
    /bfd-ip-mh:sessions/bfd-ip-mh:session-statistics:
    +--ro lost-packet-count?   yang:counter32
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/bfd:bfd/bfd-lag:lag
    /bfd-lag:sessions/bfd-lag:session/bfd-lag:member-links
    /bfd-lag:micro-bfd-ipv4/bfd-lag:session-statistics:
    +--ro lost-packet-count?   yang:counter32
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/bfd:bfd/bfd-lag:lag
    /bfd-lag:sessions/bfd-lag:session/bfd-lag:member-links
    /bfd-lag:micro-bfd-ipv6/bfd-lag:session-statistics:
    +--ro lost-packet-count?   yang:counter32
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/bfd:bfd/bfd-mpls:mpls
    /bfd-mpls:session-groups/bfd-mpls:session-group
    /bfd-mpls:sessions/bfd-mpls:session-statistics:
    +--ro lost-packet-count?   yang:counter32

```

6.2. YANG Module

This YANG module imports Common YANG Types [RFC6991], A YANG Data Model for Routing [RFC8349], and YANG Data Model for Bidirectional Forwarding Detection (BFD) [I-D.ietf-bfd-yang].

```

<CODE BEGINS> file "ietf-bfd-stability@2021-04-11.yang"
module ietf-bfd-stability {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-bfd-stability";
  prefix "bfds";

  import ietf-yang-types {
    prefix "yang";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing {
    prefix "rt";
    reference

```

```
    "RFC 8349: A YANG Data Model for Routing Management
      (NMDA version)";
  }

  import ietf-bfd {
    prefix bfd;
    reference
      "I-D.ietf-bfd-yang: YANG Data Model for Bidirectional
        Forwarding Detection.";
  }

  import ietf-bfd-ip-sh {
    prefix bfd-ip-sh;
    reference
      "I-D.ietf-bfd-yang: YANG Data Model for Bidirectional
        Forwarding Detection.";
  }

  import ietf-bfd-ip-mh {
    prefix bfd-ip-mh;
    reference
      "I-D.ietf-bfd-yang: YANG Data Model for Bidirectional
        Forwarding Detection.";
  }

  import ietf-bfd-lag {
    prefix bfd-lag;
    reference
      "I-D.ietf-bfd-yang: YANG Data Model for Bidirectional
        Forwarding Detection.";
  }

  import ietf-bfd-mpls {
    prefix bfd-mpls;
    reference
      "I-D.ietf-bfd-yang: YANG Data Model for Bidirectional
        Forwarding Detection.";
  }

  organization
    "IETF BFD Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/bfd>
      WG List: <bfd@ietf.org>

      Authors: Mahesh Jethanandani (mjethanandani@gmail.com)
               Ashesh Mishra (mishra.ashesh@gmail.com)"
```

Ankur Saxena (ankurpsaxena@gmail.com)
Santosh Pallagatti (santosh.pallagatti@gmail.com)
Mach Chen (mach.chen@huawei.com)
Peng Fan (fanp08@gmail.com).";

description

"This YANG module augments the base BFD YANG model to add attributes related to BFD Stability. In particular it adds a per session count for BFD packets that are lost.

Copyright (c) 2021 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC XXXX (<https://www.rfc-editor.org/info/rfcXXXX>); see the RFC itself for full legal notices.

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 BCP 14 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here.";

```
revision "2021-04-11" {  
  description  
    "Initial Version.";  
  reference  
    "RFC XXXX, BFD Stability.";  
}
```

```
augment "/rt:routing/rt:control-plane-protocols/" +  
  "rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh/" +  
  "bfd-ip-sh:sessions/bfd-ip-sh:session/" +  
  "bfd-ip-sh:session-statistics" {  
  leaf lost-packet-count {  
    type yang:counter32;  
    description  
      "Number of BFD packets that were lost without bringing the  
      session down.";  
  }  
}
```

```
    description
      "Augment the 'bfd' container to add attributes related to BFD
      stability.";
  }

  augment "/rt:routing/rt:control-plane-protocols/" +
    "rt:control-plane-protocol/bfd:bfd/bfd-ip-mh:ip-mh/" +
    "bfd-ip-mh:session-groups/bfd-ip-mh:session-group/" +
    "bfd-ip-mh:sessions/bfd-ip-mh:session-statistics" {
    leaf lost-packet-count {
      type yang:counter32;
      description
        "Number of BFD packets that were lost without bringing the
        session down.";
    }
    description
      "Augment the 'bfd' container to add attributes related to BFD
      stability.";
  }

  augment "/rt:routing/rt:control-plane-protocols/" +
    "rt:control-plane-protocol/bfd:bfd/bfd-lag:lag/" +
    "bfd-lag:sessions/bfd-lag:session/bfd-lag:member-links/" +
    "bfd-lag:micro-bfd-ipv4/bfd-lag:session-statistics" {
    leaf lost-packet-count {
      type yang:counter32;
      description
        "Number of BFD packets that were lost without bringing the
        session down.";
    }
    description
      "Augment the 'bfd' container to add attributes related to BFD
      stability.";
  }

  augment "/rt:routing/rt:control-plane-protocols/" +
    "rt:control-plane-protocol/bfd:bfd/bfd-lag:lag/" +
    "bfd-lag:sessions/bfd-lag:session/bfd-lag:member-links/" +
    "bfd-lag:micro-bfd-ipv6/bfd-lag:session-statistics" {
    leaf lost-packet-count {
      type yang:counter32;
      description
        "Number of BFD packets that were lost without bringing the
        session down.";
    }
    description
      "Augment the 'bfd' container to add attributes related to BFD
      stability.";
```

```
    }  
  
    augment "/rt:routing/rt:control-plane-protocols/" +  
        "rt:control-plane-protocol/bfd:bfd/bfd-mpls:mpls/" +  
        "bfd-mpls:session-groups/bfd-mpls:session-group/" +  
        "bfd-mpls:sessions/bfd-mpls:session-statistics" {  
        leaf lost-packet-count {  
            type yang:counter32;  
            description  
                "Number of BFD packets that were lost without bringing the  
                session down.";  
        }  
        description  
            "Augment the 'bfd' container to add attributes related to BFD  
            stability.";  
    }  
}  
<CODE ENDS>
```

7. IANA Considerations

7.1. The "IETF XML" Registry

This document registers one URIs in the "ns" subregistry of the "IETF XML" registry [RFC3688]. Following the format in [RFC3688], the following registration is requested:

URI: urn:ietf:params:xml:ns:yang:ietf-bfd-stability
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace.

7.2. The "YANG Module Names" Registry

This document registers one YANG module in the "YANG Module Names" registry YANG [RFC6020]. Following the format in YANG [RFC6020], the following registrations are requested:

name: ietf-bfd-stability
namespace: urn:ietf:params:xml:ns:yang:ietf-bfd-stability
prefix: bfds
reference: RFC XXXX

8. Security Consideration

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure

transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446]. The NETCONF Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

The YANG module does not define any writeable/creatable/deletable data nodes.

The only readable data nodes in YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. The model does not define any readable subtrees and data nodes.

The YANG module does not define any RPC operations.

9. Contributors

Manav Bhatia

10. Acknowledgements

Authors would like to thank Nobo Akiya, Jeffery Haas, Dileep Singh, Basil Saji, Sagar Soni, Albert Fu and Mallik Mudigonda who also contributed to this document.

11. References

11.1. Normative References

[I-D.ietf-bfd-optimizing-authentication]

Jethanandani, M., Mishra, A., Saxena, A., and M. Bhatia, "Optimizing BFD Authentication", draft-ietf-bfd-optimizing-authentication-11 (work in progress), July 2020.

[I-D.ietf-bfd-secure-sequence-numbers]

Jethanandani, M., Agarwal, S., Mishra, A., Saxena, A., and A. DeKok, "Secure BFD Sequence Numbers", draft-ietf-bfd-secure-sequence-numbers-07 (work in progress), December 2020.

- [I-D.ietf-bfd-yang]
Rahman, R., Zheng, L., Jethanandani, M., Pallagatti, S.,
and G. Mirsky, "YANG Data Model for Bidirectional
Forwarding Detection (BFD)", draft-ietf-bfd-yang-17 (work
in progress), August 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
DOI 10.17487/RFC3688, January 2004,
<<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection
(BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010,
<<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for
the Network Configuration Protocol (NETCONF)", RFC 6020,
DOI 10.17487/RFC6020, October 2010,
<<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
and A. Bierman, Ed., "Network Configuration Protocol
(NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,
<<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure
Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011,
<<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types",
RFC 6991, DOI 10.17487/RFC6991, July 2013,
<<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF
Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017,
<<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

11.2. Informative References

- [IEEE802.1ag] Institute of Electrical and Electronics Engineers, Inc., "802.1ag - Connectivity Fault Management", September 2007, <<https://www.ieee802.org/1/pages/802.1ag.html>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.

Authors' Addresses

Ashesh Mishra
SES

Email: mishra.ashesh@gmail.com

Mahesh Jethanandani
Kloud Services
CA
USA

Email: mjethanandani@gmail.com

Ankur Saxena
Ciena Corporation
3939 North 1st Street
San Jose, CA 95134
USA

Email: ankurpsaxena@gmail.com
URI: www.ciena.com

Santosh Pallagatti
VMware
Bangalore, Karnataka 560103
India

Email: santosh.pallagatti@gmail.com

Mach Chen
Huawei

Email: mach.chen@huawei.com

Peng Fan
China Mobile
32 Xuanwumen West Street
Beijing, Beijing
China

Email: fanp08@gmail.com

BFD Working Group
Internet-Draft
Updates: 5880 (if approved)
Intended status: Standards Track
Expires: 12 August 2022

W. Cheng
R. Wang
China Mobile
X. Min, Ed.
ZTE Corp.
R. Rahman
Individual
R. Boddireddy
Juniper Networks
8 February 2022

Unaffiliated BFD Echo
draft-ietf-bfd-unaffiliated-echo-04

Abstract

Bidirectional Forwarding Detection (BFD) is a fault detection protocol that can quickly determine a communication failure between two forwarding engines. This document proposes a use of the BFD Echo where the local system supports BFD but the neighboring system does not support BFD.

This document updates RFC 5880.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 12 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
1.1. Conventions Used in This Document	3
2. Updates to RFC 5880	3
3. Unaffiliated BFD Echo Procedures	6
4. Unaffiliated BFD Echo Applicability	8
5. Security Considerations	8
6. IANA Considerations	8
7. Acknowledgements	8
8. Contributors	9
9. References	9
9.1. Normative References	9
9.2. Informative References	9
Authors' Addresses	10

1. Introduction

To minimize the impact of device/link faults on services and improve network availability, a network device must be able to quickly detect faults in communication with adjacent devices. Measures can then be taken to promptly rectify the faults to ensure service continuity.

BFD [RFC5880] is a low-overhead, short-duration method to detect faults on the communication path between adjacent forwarding engines. The faults can be on interfaces, data link(s), and even the forwarding engines. It is a single, unified mechanism to monitor any media and protocol layers in real time.

BFD defines Asynchronous and Demand modes to satisfy various deployment scenarios. It also supports an Echo function to reduce the device requirement for BFD. When the Echo function is activated, the local system sends BFD Echo packets and the remote system loops back the received Echo packets through the forwarding path. If several consecutive BFD Echo packets are not received by the local system, then the BFD session is declared to be Down.

When using BFD Echo function, there are two typical scenarios as below:

- * Full BFD protocol capability with affiliated Echo function. This scenario requires both the local device and the neighboring device to support the full BFD protocol.
- * BFD Echo-Only method without full BFD protocol capability. This scenario requires only the local device to support sending and demultiplexing BFD Control packets.

The latter scenario is referred to as Unaffiliated BFD Echo in this document.

Section 6.2.2 of [BBF-TR-146] describes one use case of the Unaffiliated BFD Echo. Section 2 of [I-D.wang-bfd-one-arm-use-case] describes another use case of the Unaffiliated BFD Echo.

This document describes the use of the Unaffiliated BFD Echo over IPv4 and IPv6 for single IP hop.

1.1. Conventions Used in This Document

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Updates to RFC 5880

The Unaffiliated BFD Echo described in this document reuses the BFD Echo function as described in [RFC5880] and [RFC5881], but does not require BFD Asynchronous or Demand mode. When using the Unaffiliated BFD Echo, only the local system has the BFD protocol enabled; the remote system just loops back the received BFD Echo packets as regular data packets.

This document updates [RFC5880] with respect to its descriptions on the BFD Echo function as follows.

The 4th paragraph of Section 3.2 of [RFC5880] is updated as below:

OLD TEXT

An adjunct to both modes is the Echo function.

NEW TEXT

An adjunct to both modes is the Echo function, which can also be running independently.

OLD TEXT

Since the Echo function is handling the task of detection, the rate of periodic transmission of Control packets may be reduced (in the case of Asynchronous mode) or eliminated completely (in the case of Demand mode).

NEW TEXT

Since the Echo function is handling the task of detection, the rate of periodic transmission of Control packets may be reduced (in the case of Asynchronous mode) or eliminated completely (in the case of Demand mode). The Echo function may also be used independently, with neither Asynchronous nor Demand mode.

The 3rd and 9th paragraphs of Section 6.1 of [RFC5880] are updated as below:

OLD TEXT

Once the BFD session is Up, a system can choose to start the Echo function if it desires and the other system signals that it will allow it. The rate of transmission of Control packets is typically kept low when the Echo function is active.

NEW TEXT

When a system is running with Asynchronous or Demand mode, once the BFD session is Up, it can choose to start the Echo function if it desires and the other system signals that it will allow it. The rate of transmission of Control packets is typically kept low for Asynchronous mode or eliminated completely for Demand mode when the Echo function is active.

OLD TEXT

If the session goes Down, the transmission of Echo packets (if any) ceases, and the transmission of Control packets goes back to the slow rate.

NEW TEXT

In Asynchronous mode, if the session goes Down, the transmission of Echo packets (if any) ceases, and the transmission of Control packets goes back to the slow rate. Demand mode MUST NOT be active if the session goes Down.

The 2nd paragraph of Section 6.4 of [RFC5880] is updated as below:

OLD TEXT

When a system is using the Echo function, it is advantageous to choose a sedate reception rate for Control packets, since liveness detection is being handled by the Echo packets. This can be controlled by manipulating the Required Min RX Interval field (see section 6.8.3).

NEW TEXT

When a system is using the Echo function with Asynchronous mode, it is advantageous to choose a sedate reception rate for Control packets, since liveness detection is being handled by the Echo packets. This can be controlled by manipulating the Required Min RX Interval field (see section 6.8.3). Note that a system operating in Demand mode would direct the remote system to cease the periodic transmission of BFD Control packets, by setting the Demand (D) bit in its BFD Control packets.

The 2nd paragraph of Section 6.8 of [RFC5880] is updated as below:

OLD TEXT

When a system is said to have "the Echo function active" it means that the system is sending BFD Echo packets, implying that the session is Up and the other system has signaled its willingness to loop back Echo packets.

NEW TEXT

When a system in Asynchronous or Demand mode is said to have "the Echo function active" it means that the system is sending BFD Echo packets, implying that the session is Up and the other system has signaled its willingness to loop back Echo packets.

The 7th paragraph of Section 6.8.3 of [RFC5880] is updated as below:

OLD TEXT

When the Echo function is active, a system SHOULD set `bfd.RequiredMinRxInterval` to a value of not less than one second (1,000,000 microseconds). This is intended to keep received BFD Control traffic at a negligible level, since the actual detection function is being performed using BFD Echo packets.

NEW TEXT

When the Echo function is active with Asynchronous mode, a system SHOULD set `bfd.RequiredMinRxInterval` to a value of not less than one second (1,000,000 microseconds). This is intended to keep received BFD Control traffic at a negligible level, since the actual detection function is being performed using BFD Echo packets. While a system operating in Demand mode would not receive BFD Control traffic.

The 1st and 2nd paragraphs of Section 6.8.9 of [RFC5880] are updated as below:

OLD TEXT

BFD Echo packets MUST NOT be transmitted when bfd.SessionState is not Up. BFD Echo packets MUST NOT be transmitted unless the last BFD Control packet received from the remote system contains a nonzero value in Required Min Echo RX Interval.

NEW TEXT

When a system is using the Echo function with either Asynchronous or Demand mode, BFD Echo packets MUST NOT be transmitted when bfd.SessionState is not Up, and BFD Echo packets MUST NOT be transmitted unless the last BFD Control packet received from the remote system contains a nonzero value in Required Min Echo RX Interval.

OLD TEXT

BFD Echo packets MAY be transmitted when bfd.SessionState is Up. The interval between transmitted BFD Echo packets MUST NOT be less than the value advertised by the remote system in Required Min Echo RX Interval...

NEW TEXT

When a system is using the Echo function with either Asynchronous or Demand mode, BFD Echo packets MAY be transmitted when bfd.SessionState is Up, and the interval between transmitted BFD Echo packets MUST NOT be less than the value advertised by the remote system in Required Min Echo RX Interval...

3. Unaffiliated BFD Echo Procedures

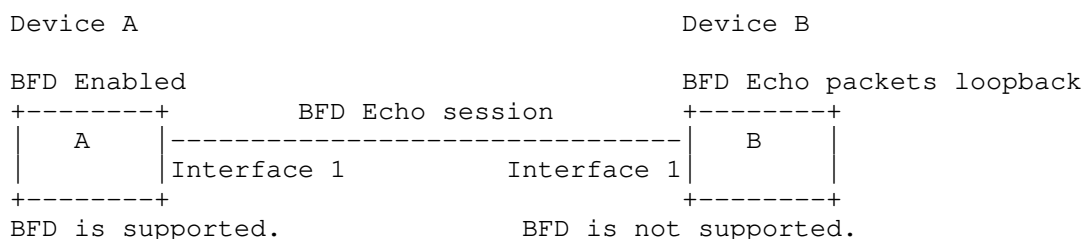


Figure 1: Unaffiliated BFD Echo diagram

As shown in Figure 1, device A supports BFD, whereas device B does not support BFD. Device A would send BFD Echo packets, and after receiving the BFD Echo packets sent from device A, the one-hop-away BFD peer device B immediately loops them back by normal IP

forwarding, this allows device A to rapidly detect a connectivity loss to device B. Note that device B would not intercept any received BFD Echo packet or parse any BFD protocol field within the BFD Echo packet.

To rapidly detect any IP forwarding faults between device A and device B, a BFD Echo session MUST be created at device A, and the BFD Echo session MUST follow the BFD state machine defined in Section 6.2 of [RFC5880], except that the received state is not sent but echoed from the remote system, and AdminDown state is ruled out because AdminDown effectively means removal of BFD Echo session. In this case, although BFD Echo packets are transmitted with destination UDP port 3785 as defined in [RFC5881], the BFD Echo packets sent by device A are BFD Control packets too, the looped BFD Echo packets back from device B would drive BFD state change at device A, substituting the BFD Control packets sent from the BFD peer. Also note that when device A receives looped BFD Control packets, the validation procedures of [RFC5880] are used.

Once a BFD Echo session is created at device A, it starts sending BFD Echo packets, which MUST include BFD Echo session demultiplexing fields, such as BFD "Your Discriminator" defined in [RFC5880] (BFD "My Discriminator" can be set to 0 to avoid confusion), except for BFD "Your Discriminator", device A can also use IP source address or UDP source port to demultiplex BFD Echo session, or there is only one BFD Echo session running at device A. Device A would send BFD Echo packets with IP destination address destined for itself, such as the IP address of interface 1 of device A. All BFD Echo packets for the session MUST be sent with a Time to Live (TTL) or Hop Limit value of 255.

Within the BFD Echo packet, the "Desired Min TX Interval" and "Required Min RX Interval" defined in [RFC5880] may be populated with one second, which however has no real application and would be ignored by the receiver.

Considering that the BFD peer device B wouldn't advertise "Required Min Echo RX Interval" as defined in [RFC5880], the transmission interval for sending BFD Echo packets MUST be provisioned at device A, how to make sure the BFD peer device B is willing and able to loop back BFD Echo packets sent with the provisioned transmission interval is outside the scope of this document. Similar to what's specified in [RFC5880], the BFD Echo session begins with the periodic, slow transmission of BFD Echo packets, the slow transmission rate SHOULD be no less than one second per packet, until the session is Up, after that the provisioned transmission interval is applied, and reverting back to the slow rate once the session goes Down. Considering that the BFD peer wouldn't advertise "Detect Mult" as defined in

[RFC5880], the "Detect Mult" for calculating the Detection Time MUST be provisioned at device A, the Detection Time at device A is equal to the provisioned "Detect Mult" multiplied by the provisioned transmission interval.

4. Unaffiliated BFD Echo Applicability

Some devices that would benefit from the use of BFD may be unable to support the full BFD protocol. Examples of such devices include servers running virtual machines, or Internet of Things (IoT) devices. The Unaffiliated BFD Echo can be used when two devices are connected and only one of them supports the BFD protocol, and the other is capable of looping BFD Echo packets.

5. Security Considerations

All Security Considerations from [RFC5880] and [RFC5881] apply.

Note that the Unaffiliated BFD Echo prevents the use of Unicast Reverse Path Forwarding (URPF) [RFC3704] [RFC8704] in strict mode.

As specified in Section 5 of [RFC5880], since BFD Echo packets may be spoofed, some form of authentication SHOULD be included. Considering the BFD Echo packets in this document are also BFD Control packets, the "Authentication Section" as defined in [RFC5880] for BFD Control packet is RECOMMENDED to be included within the BFD Echo packet.

In order to mitigate the potential reflector attack by the remote attackers, or infinite loop of the BFD Echo packets, it's RECOMMENDED to put two requirements on the device looping BFD Echo packets, the first one is that a packet SHOULD NOT be looped unless it has a TTL or Hop Limit value of 255, and the second one is that a packet being looped MUST NOT reset the TTL or Hop Limit value to 255, and MUST use a TTL or Hop Limit value of 254.

6. IANA Considerations

This document has no IANA action requested.

7. Acknowledgements

The authors would like to acknowledge Ketan Talaulikar, Greg Mirsky and Santosh Pallagatti for their careful review and very helpful comments.

The authors would like to acknowledge Jeff Haas for his insightful review and very helpful comments.

8. Contributors

Liu Aihua
ZTE
Email: liu.aihua@zte.com.cn

Qian Xin
ZTE
Email: qian.xin2@zte.com.cn

Zhao Yanhua
ZTE
Email: zhao.yanhua3@zte.com.cn

9. References

9.1. 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://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, <<https://www.rfc-editor.org/info/rfc5881>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

9.2. Informative References

- [BBF-TR-146] Broadband Forum, "BBF Technical Report - Subscriber Sessions Issue 1", 2013, <<https://www.broadband-forum.org/technical/download/TR-146.pdf>>.

[I-D.wang-bfd-one-arm-use-case]

Wang, R., Cheng, W., Zhao, Y., and A. Liu, "Using One-Arm BFD in Cloud Network", Work in Progress, Internet-Draft, draft-wang-bfd-one-arm-use-case-00, 18 November 2019, <<https://www.ietf.org/archive/id/draft-wang-bfd-one-arm-use-case-00.txt>>.

[RFC3704] Baker, F. and P. Savola, "Ingress Filtering for Multihomed Networks", BCP 84, RFC 3704, DOI 10.17487/RFC3704, March 2004, <<https://www.rfc-editor.org/info/rfc3704>>.

[RFC8704] Sriram, K., Montgomery, D., and J. Haas, "Enhanced Feasible-Path Unicast Reverse Path Forwarding", BCP 84, RFC 8704, DOI 10.17487/RFC8704, February 2020, <<https://www.rfc-editor.org/info/rfc8704>>.

Authors' Addresses

Weiqiang Cheng
China Mobile
Beijing
China

Email: chengweiqiang@chinamobile.com

Ruixue Wang
China Mobile
Beijing
China

Email: wangruixue@chinamobile.com

Xiao Min (editor)
ZTE Corp.
Nanjing
China

Email: xiao.min2@zte.com.cn

Reshad Rahman
Individual
Kanata
Canada

Email: reshad@yahoo.com

Raj Chetan Boddireddy
Juniper Networks

Email: rchetan@juniper.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 6 June 2022

E. Chen
Palo Alto Networks
N. Shen
Zededa
R. Raszuk
NTT Network Innovations
R. Rahman
3 December 2021

Unsolicited BFD for Sessionless Applications
draft-ietf-bfd-unsolicited-09

Abstract

For operational simplification of "sessionless" applications using BFD, in this document we present procedures for "unsolicited BFD" that allow a BFD session to be initiated by only one side, and be established without explicit per-session configuration or registration by the other side (subject to certain per-interface or per-router policies).

We also introduce a new YANG module to configure and manage "unsolicited BFD". The YANG module in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

Requirements Language

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 6 June 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Procedures for Unsolicited BFD	3
3. State Variables	4
4. YANG Data Model	5
4.1. Unsolicited BFD Hierarchy	5
4.2. Unsolicited BFD Module	6
5. IANA Considerations	10
6. Acknowledgments	10
7. Security Considerations	10
7.1. BFD Protocol Security Considerations	11
7.2. YANG Module Security Considerations	11
8. References	12
8.1. Normative References	12
8.2. Informative References	14
Authors' Addresses	14

1. Introduction

The current implementation and deployment practice for BFD ([RFC5880] and [RFC5881]) usually requires BFD sessions be explicitly configured or registered on both sides. This requirement is not an issue when an application like BGP [RFC4271] has the concept of a "session" that involves both sides for its establishment. However, this requirement can be operationally challenging when the prerequisite "session" does not naturally exist between two endpoints in an application. Simultaneous configuration and coordination may be required on both sides for BFD to take effect. For example:

- * When BFD is used to keep track of the "liveness" of the nexthop of static routes. Although only one side may need the BFD functionality, currently both sides need to be involved in specific configuration and coordination and in some cases static routes are created unnecessarily just for BFD.
- * When BFD is used to keep track of the "liveness" of the third-party nexthop of BGP routes received from the Route Server [RFC7947] at an Internet Exchange Point (IXP). As the third-party nexthop is different from the peering address of the Route Server, for BFD to work, currently two routers peering with the Route Server need to have routes and nexthops from each other (although indirectly via the Router Server), and the nexthop of each router must be present at the same time. These issues are also discussed in [I-D.ietf-idr-rs-bfd].

Clearly it is beneficial and desirable to reduce or eliminate unnecessary configurations and coordination in these "sessionless" applications using BFD.

In this document we present procedures for "unsolicited BFD" that allow a BFD session to be initiated by only one side, and be established without explicit per-session configuration or registration by the other side (subject to certain per-interface or per-router policies).

With "unsolicited BFD" there is potential risk for excessive resource usage by BFD from "unexpected" remote systems. To mitigate such risks, several mechanisms are recommended in the Security Considerations section.

Compared to the "Seamless BFD" [RFC7880], this proposal involves only minor procedural enhancements to the widely deployed BFD itself. Thus we believe that this proposal is inherently simpler in the protocol itself and deployment. As an example, it does not require the exchange of BFD discriminators over an out-of-band channel before the BFD session bring-up.

When BGP Add-Path [RFC7911] is deployed at an IXP using the Route Server, multiple BGP paths (when exist) can be made available to the clients of the Router Server as described in [RFC7947]. The "unsolicited BFD" can be used in BGP route selection by these clients to eliminate paths with "inaccessible nexthops".

2. Procedures for Unsolicited BFD

With "unsolicited BFD", one side takes the "Active role" and the other side takes only the "Passive role" as described in [RFC5880].

On the passive side, the "unsolicited BFD" SHOULD be explicitly configured on an interface or globally (apply to all interfaces). The BFD parameters can be either per-interface or per-router based. It MAY also choose to use the parameters that the active side uses in its BFD Control packets. The "My Discriminator", however, MUST be chosen to allow multiple unsolicited BFD sessions.

The active side starts sending the BFD Control packets as specified in [RFC5880]. The passive side does not send BFD Control packets.

When the passive side receives a BFD Control packet from the active side with 0 as "Your Discriminator" and does not find an existing BFD session, the passive side MAY create a matching BFD session toward the active side, if permitted by local configuration.

It would then start sending the BFD Control packets and perform necessary procedure for bringing up, maintaining and tearing down the BFD session. If the BFD session fails to get established within certain specified time, or if an established BFD session goes down, the passive side would stop sending BFD Control packets and MAY delete the BFD session created until the BFD Control packets is initiated by the active side again.

When an Unsolicited BFD session goes down, an implementation MAY retain the session state for a period of time, which may be configurable. Retaining this state can be useful for operational purposes.

The "Passive role" may change to the "Active role" when a local client registers for the same BFD session, and from the "Active role" to the "Passive role" when there is no longer any locally registered client for the BFD session.

3. State Variables

This document defines a new state variable called Unsolicited Role.

bfd.UnsolicitedRole

The operational mode of BFD interface when configured for unsolicited behaviour. Options can be either PASSIVE, ACTIVE or NULL (NULL - not initialized) for unsolicited BFD sessions. Default (not configured for unsolicited behaviour) MUST be set to NULL if present on the interface.

4. YANG Data Model

This section extends the YANG data model for BFD [RFC9127] to cover unsolicited BFD. We import [RFC8349] since the "bfd" container in [RFC9127] is under "control-plane-protocol".

4.1. Unsolicited BFD Hierarchy

Configuration for unsolicited BFD parameters for IP single-hop sessions can be done at 2 levels:

- * Globally, i.e. for all interfaces. This requires support for the "unsolicited-params-global" feature.
- * For specific interfaces. This requires support for the "unsolicited-params-per-interface" feature.

For operational data, a new "unsolicited" container has been added for BFD IP single-hop sessions.

The tree diagram below uses the graphical representation of data models, as defined in [RFC8340].

```

module: ietf-bfd-unsolicited

augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh:
  +--rw unsolicited {bfd-unsol:unsolicited-params-global}?
    +--rw enabled? boolean
    +--rw local-multiplier? multiplier
    +--rw (interval-config-type)?
      +--:(tx-rx-intervals)
        | +--rw desired-min-tx-interval? uint32
        | +--rw required-min-rx-interval? uint32
      +--:(single-interval) {single-minimum-interval}?
        +--rw min-interval? uint32
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh
  /bfd-ip-sh:interfaces:
  +--rw unsolicited {bfd-unsol:unsolicited-params-per-interface}?
    +--rw enabled? boolean
    +--rw local-multiplier? multiplier
    +--rw (interval-config-type)?
      +--:(tx-rx-intervals)
        | +--rw desired-min-tx-interval? uint32
        | +--rw required-min-rx-interval? uint32
      +--:(single-interval) {single-minimum-interval}?
        +--rw min-interval? uint32
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh
  /bfd-ip-sh:sessions/bfd-ip-sh:session:
  +--ro unsolicited
    +--ro role? bfd-unsol:unsolicited-role

```

4.2. Unsolicited BFD Module

```

<CODE BEGINS> file "ietf-bfd-unsolicited@2021-11-23.yang"
module ietf-bfd-unsolicited {

  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-bfd-unsolicited";

  prefix "bfd-unsol";

  // RFC Ed.: replace occurrences of YYYY with actual RFC numbers
  // and remove this note

  import ietf-bfd-types {

```

```
    prefix "bfd-types";
    reference
      "RFC 9127: YANG Data Model for Bidirectional Forwarding Detection
      (BFD)";
  }

  import ietf-bfd {
    prefix "bfd";
    reference
      "RFC 9127: YANG Data Model for Bidirectional Forwarding Detection
      (BFD)";
  }

  import ietf-bfd-ip-sh {
    prefix "bfd-ip-sh";
    reference
      "RFC 9127: YANG Data Model for Bidirectional Forwarding Detection
      (BFD)";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management
      (NMDA version)";
  }

  organization "IETF BFD Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/bfd/>
    WG List: <rtg-bfd@ietf.org>

    Editors: Enke Chen (enchen@paloaltonetworks.com),
             Naiming Shen (naiming@zededa.com),
             Robert Raszuk (robert@raszuk.net),
             Reshad Rahman (reshad@yahoo.com)";

  description
    "This module contains the YANG definition for BFD unsolicited
    as per RFC YYYY.

    Copyright (c) 2021 IETF Trust and the persons
    identified as authors of the code. All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
```

set forth in Section 4.c of the IETF Trust's Legal Provisions
Relating to IETF Documents
(<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC YYYY; see
the RFC itself for full legal notices.";

```
reference "RFC YYYY";

revision 2021-11-23 {
  description
    "Initial revision.";
  reference
    "RFC YYYY: Unsolicited BFD for Sessionless Applications.";
}

/*
 * Feature definitions
 */
feature unsolicited-params-global {
  description
    "This feature indicates that the server supports global
    parameters for unsolicited sessions.";
  reference
    "RFC YYYY: Unsolicited BFD for Sessionless Applications.";
}

feature unsolicited-params-per-interface {
  description
    "This feature indicates that the server supports per-interface
    parameters for unsolicited sessions.";
  reference
    "RFC YYYY: Unsolicited BFD for Sessionless Applications.";
}

/*
 * Type Definitions
 */
typedef unsolicited-role {
  type enumeration {
    enum unsolicited-active {
      description "Active role";
    }
    enum unsolicited-passive {
      description "Passive role";
    }
  }
  description "Unsolicited role";
}
```

```

    }

    /*
    * Augments
    */
    augment "/rt:routing/rt:control-plane-protocols/"
        + "rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh" {
        if-feature bfd-unsol:unsolicited-params-global;
        description
            "Augmentation for BFD unsolicited parameters";
        container unsolicited {
            description
                "BFD unsolicited top level container";
            leaf enabled {
                type boolean;
                default false;
                description
                    "BFD unsolicited enabled globally for IP single-hop.";
            }
            uses bfd-types:base-cfg-parms;
        }
    }

    augment "/rt:routing/rt:control-plane-protocols/"
        + "rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh/"
        + "bfd-ip-sh:interfaces" {
        if-feature bfd-unsol:unsolicited-params-per-interface;
        description
            "Augmentation for BFD unsolicited on IP single-hop interface";
        container unsolicited {
            description
                "BFD IP single-hop interface unsolicited top level
                container";
            leaf enabled {
                type boolean;
                default false;
                description
                    "BFD unsolicited enabled on this interface.";
            }
            uses bfd-types:base-cfg-parms;
        }
    }

    augment "/rt:routing/rt:control-plane-protocols/"
        + "rt:control-plane-protocol/bfd:bfd/bfd-ip-sh:ip-sh/"
        + "bfd-ip-sh:sessions/bfd-ip-sh:session" {
        description
            "Augmentation for BFD unsolicited on IP single-hop session";
    }

```

```
    container unsolicited {
      config false;
      description
        "BFD IP single-hop session unsolicited top level container";
      leaf role {
        type bfd-unsol:unsolicited-role;
        description "Role.";
      }
    }
  }
}
<CODE ENDS>
```

5. IANA Considerations

This document registers the following namespace URI in the "IETF XML Registry" [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-bfd-unsolicited

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

This document registers the following YANG module in the "YANG Module Names" registry [RFC6020]:

Name: ietf-bfd-unsolicited

Namespace: urn:ietf:params:xml:ns:yang:ietf-bfd-unsolicited

Prefix: bfd-unsol

Reference: RFC YYYY

6. Acknowledgments

Authors would like to thank Acee Lindem, Greg Mirsky, Jeffrey Haas, Raj Chetan and Tom Petch for their review and valuable input.

7. Security Considerations

7.1. BFD Protocol Security Considerations

The same security considerations and protection measures as those described in [RFC5880] and [RFC5881] normatively apply to this document. With "unsolicited BFD" there is potential risk for excessive resource usage by BFD from "unexpected" remote systems. To mitigate such risks, the following measures are mandatory:

- * Limit the feature to specific interfaces, and to a single-hop BFD with "TTL=255" [RFC5082]. For numbered interfaces, the source address of an incoming BFD packet should belong to the subnet of the interface on which the BFD packet is received. For unnumbered interfaces the above check should be aligned with routing protocol addresses running on such pair of interfaces.
- * Apply "policy" to allow BFD packets only from certain subnets or hosts.
- * Deploy the feature only in certain "trustworthy" environment, e.g., at an IXP, or between a provider and its customers.
- * Adjust BFD parameters as needed for the particular deployment and scale.
- * Use BFD authentication.

7.2. YANG Module Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

```
/routing/control-plane-protocols/control-plane-protocol/bfd/ip-sh  
/unsolicited:
```

- * data node "enabled" enables creation of unsolicited BFD IP single-hop sessions globally, i.e. on all interfaces. See Section 7.1.
- * data nodes local-multiplier, desired-min-tx-interval, required-min-rx-interval and min-interval all impact the parameters of the unsolicited BFD IP single-hop sessions.

/routing/control-plane-protocols/control-plane-protocol/bfd/ip-sh
/interfaces/interface/unsolicited:

- * data node "enabled" enables creation of unsolicited BFD IP single-hop sessions on a specific interface. See Section 7.1.
- * data nodes local-multiplier, desired-min-tx-interval, required-min-rx-interval and min-interval all impact the parameters of the unsolicited BFD IP single-hop sessions on the interface.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/routing/control-plane-protocols/control-plane-protocol/bfd/ip-sh
/sessions/session/unsolicited: access to this information discloses the role of the local system in the creation of the unsolicited BFD session.

8. References

8.1. 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <<https://www.rfc-editor.org/info/rfc5082>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://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, <<https://www.rfc-editor.org/info/rfc5881>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

- [RFC9127] Rahman, R., Ed., Zheng, L., Ed., Jethanandani, M., Ed., Pallagatti, S., and G. Mirsky, "YANG Data Model for Bidirectional Forwarding Detection (BFD)", RFC 9127, DOI 10.17487/RFC9127, October 2021, <<https://www.rfc-editor.org/info/rfc9127>>.

8.2. Informative References

- [I-D.ietf-idr-rs-bfd] Bush, R., Haas, J., Scudder, J. G., Nipper, A., and C. Dietzel, "Making Route Servers Aware of Data Link Failures at IXPs", Work in Progress, Internet-Draft, draft-ietf-idr-rs-bfd-09, 21 September 2020, <<https://www.ietf.org/archive/id/draft-ietf-idr-rs-bfd-09.txt>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [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, <<https://www.rfc-editor.org/info/rfc7880>>.
- [RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <<https://www.rfc-editor.org/info/rfc7911>>.
- [RFC7947] Jasinska, E., Hilliard, N., Raszuk, R., and N. Bakker, "Internet Exchange BGP Route Server", RFC 7947, DOI 10.17487/RFC7947, September 2016, <<https://www.rfc-editor.org/info/rfc7947>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.

Authors' Addresses

Enke Chen
Palo Alto Networks

Email: enchen@paloaltonetworks.com

Naiming Shen
Zededa

Email: naiming@zededa.com

Robert Raszuk
NTT Network Innovations
940 Stewart Dr
Sunnyvale, CA 94085
United States of America

Email: robert@raszuk.net

Reshad Rahman
Canada

Email: reshad@yahoo.com