

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  
03b 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.

Internet-Draft

Securing next sequence number

March 2022

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

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

## [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 has good hashing properties.

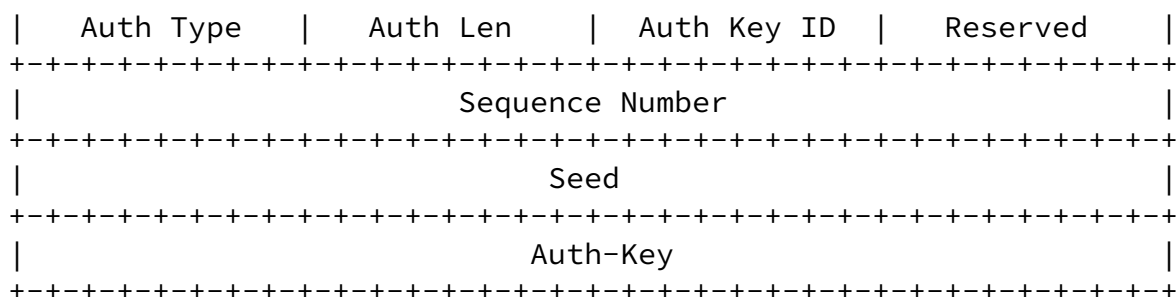
## 2. Requirements Language

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:

[illegible]



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.

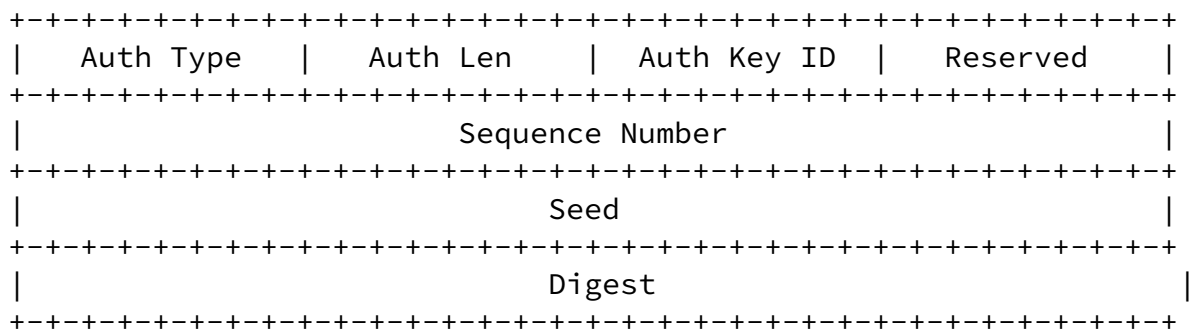
When the receiving party receives a BFD packet with an expected sequence number, and the correct corresponding ISAAC output, it knows that only the authentic sending party could have sent that message. The sending party is therefore alive/up, and intended to send the message.

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:

0	1	2	3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1			



## 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

Internet-Draft

Securing next sequence number

March 2022

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 immediately 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](http://www.cisco.com)

Ashesh Mishra  
03b 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](mailto:aland@freeradius.org)

