August 12, 2019

NESTED BIT INDEX EXPLICIT REPLICATION HEADER BASED SCALABLE SEAMLESS BIDIRECTIONAL FORWARDING DETECTION MULTIPOINT PATH VALIDATION VIA CONTROLLED RESPONDERS

Nagendra Kumar Nainar

Carlos M. Pignataro

Reshad Rahman

Ijsbrand Wijnands

NESTED BIT INDEX EXPLICIT REPLICATION HEADER BASED SCALABLE SEAMLESS BIDIRECTIONAL FORWARDING DETECTION MULTIPOINT PATH VALIDATION VIA CONTROLLED RESPONDERS

AUTHORS:
Nagendra Kumar Nainar
Carlos M. Pignataro
Reshad Rahman
Ijsbrand Wijnands

ABSTRACT

Techniques are described herein for a Bit Index Explicit Replication (BIER) - Operations Administration and Management (OAM) - Bitstring (BOB) prototype. An associated lookup semantic may also be provided. These techniques may help scale responses from receivers without changing the behavior or path on the transit nodes.

DETAILED DESCRIPTION

Internet Engineering Task Force (IETF) Request for Comments (RFC) 8279 proposes a stateless replication architecture referred to as Bit Index Explicit Replication (BIER). BEIR is an optimal and scalable solution for multipoint replication using bitstring based forwarding and following a unicast path.

Multipoint Bidirectional Forwarding Detection (BFD) may be used in any multipoint scenario to perform a continuity check from the headend to the tailends. Draft-ietf-bfd-multipoint proposes using an all-zero Your Discriminator and allowing the tailends to maintain a state entry with the source address and My Discriminator to respond for the control packet. draft-hu-bier-bfd extends this further for BIER by bootstrapping BIER Ping (ietf-bier-ping) to carry a BFD discriminator and create the state entries on the tailends.

One of the main challenges associated with multipoint technology is the scalability of the headend to process the BFD response from a large number of endpoints. Traditional BFD introduces silent tail and active tail procedures, but this does not help as the active tail procedure still requires the headend to process much of the response(s). Applying BFD / Seamless BFD (SBFD) for BIER may incur the same scale issues.

While the aforementioned scale issues in BIER may be handled by tweaking the bitstring in the BIER header, this is not true in-band data path validation. For example, a

BFR Ingress Router (BFIR) sending data traffic for a first stream with bitstring 10111 may set a different bitstring in the BIER header for BFD/SBFD packets to control the number of responses that may result from taking different data paths due to a bug or the way forwarding is implemented on BFRs.

As described herein, a BIER-in-BIER based Operations Administration and Management (OAM) approach may be implemented. The outer BIER header may be set to the stream/string that needs to be validated and thereby retain true in-band path validation. The inner header (with a BIER-OAM-Bitstring (BOB) prototype) may be used to control the responding receivers at scale and leverage a new Anycast BIER discriminator for SBFD. In particular, all BIER edge nodes (BFIR / BFR Egress Router (BFER)) may be assigned with the same discriminator value and a reflector session may be created for the same.

Both the BOB prototype and lookup semantic may be utilized. Each BFER node may create a “For Us” bitstring with its bit identifier set to 1 and the remaining bits set to 0. For example, a BFER with a BFR identifier of 2 and a BitString Length (BSL) of 5 may have a bitstring of 00010; a BFER with a BFR identifier of 3 and a BSL of 5 may have a bitstring of 00100; a BFER with a BFR identifier of 4 and a BSL of 5 may have a bitstring of 01000; etc.

A prototype may be used in the BIER header “Proto” field that instructs that the payload under the BIER header is another BIER header with the BOB prototype and has a new lookup semantic.

Any node that receives a BIER packet with the “Proto” field set to the BOB prototype may extract the bitstring and perform an AND operation with the “For Us” bitstring. If the resulting bitstring is 0, it may drop the packet. If the resulting bitstring is its own BFR identifier, it may process it further. The use of nested BIER headers ensures that the outer BIER header with the bitstring of the relevant streams to be validated may safeguard against bad implementation or bugs and help the OAM probe stay in a true in-band data path. Furthermore, use of the inner BIER header may control the responding endpoints and does not necessarily need the other nodes to process the OAM probes. This combination may be used for scalable path validation.

Figure 1 below illustrates BIER discriminator assignment and advertisement.

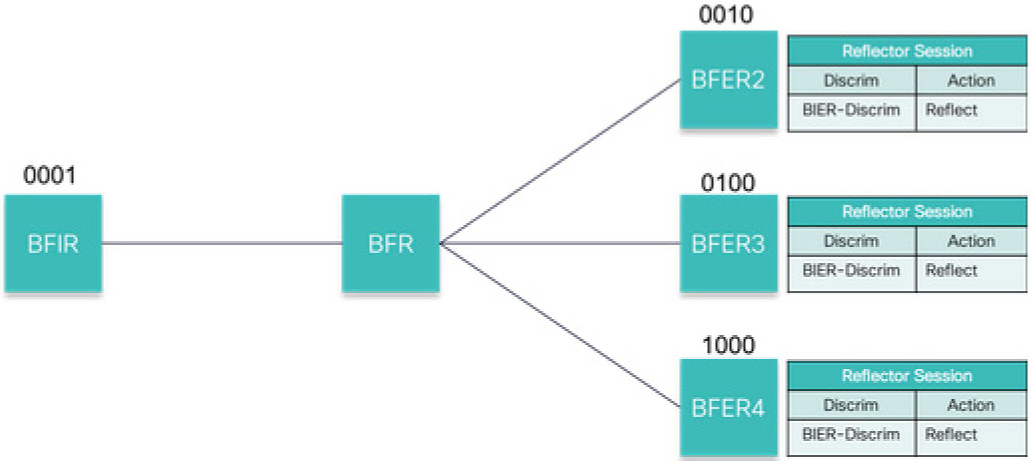


Figure 1

Each BIER domain may be assigned a domain-wide unique discriminator ("BIER-Discrim"). It may be manually configured on all BFIR/BFER or advertised via Interior Gateway Protocol (IGP).

Using any of the aforementioned advertisement approaches, each BFIR/BFER node may create a local reflector session. Any BFR node that is not an edge (e.g., any BFR that is assigned a BFR identifier) may not necessarily create this reflector session.

When any BFIR is required to validate a set of BFERs, it generates the SBFD control packet that includes My Discriminator set to a local value, Your Discriminator set to BIER-Discrim, and other values set to default for an SBFD control packet.

It will be appreciated that the multipoint flag may not be required in this case. The reflector session may not require an indication that this is a multipoint session, using a combination of the source address and My Discriminator value for session identification.

When the BFIR is expecting the response from all the BFERs, the BFD control packet may be encapsulated with a BIER header and the BFIR identifier set as a local value with the set of BFER identifiers as the bitstring. This is suitable for scenarios where the number of BFERs is small.

When the BFIR is expecting a response from a selective set of BFERs, it may generate the BFD control packet and encapsulate the packet. As illustrated in Figure 2 below, the topmost BIER header may include the bitstring and entropy that is similar to the stream for which the path needs to be validated from the BFIR. The use of the same bitstring and entropy may have a similar forwarding lookup, forwarding path, and action on any transit BFR nodes without requiring differentiation of OAM packets from data packets.

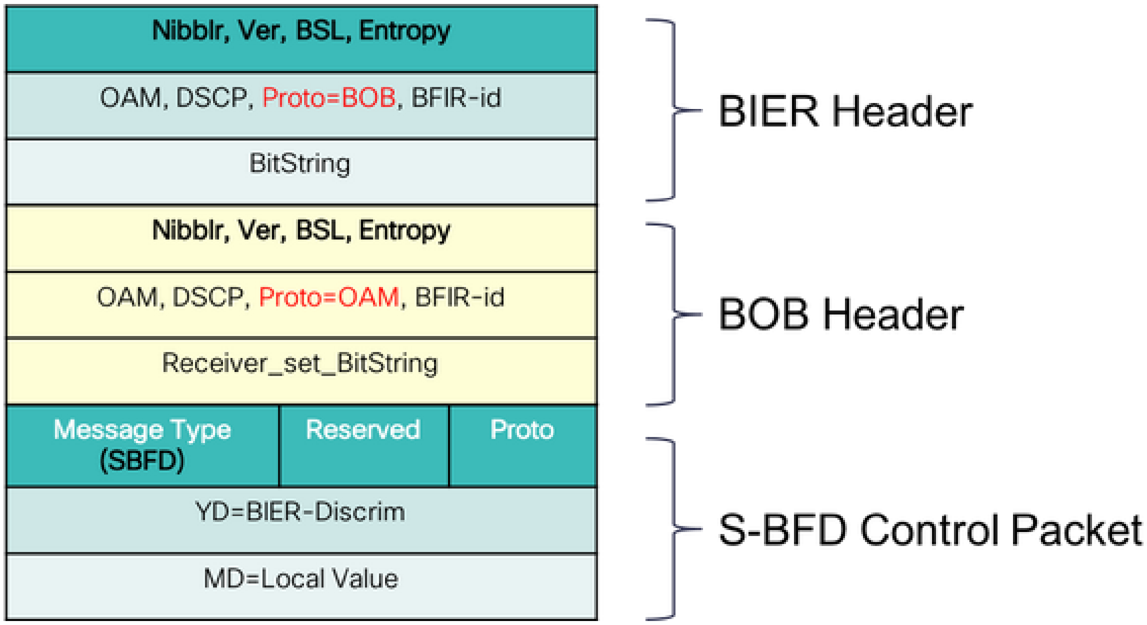
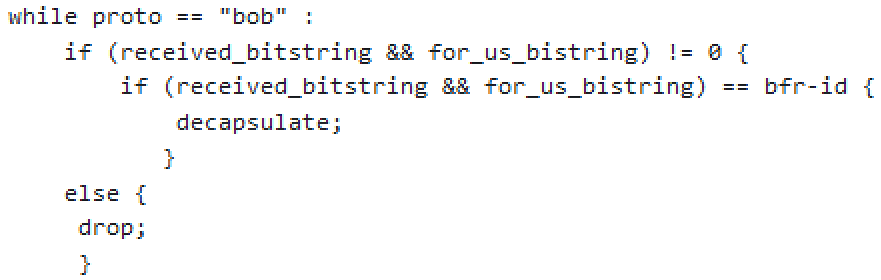


Figure 2

As illustrated in Figure 3 below, upon receiving the packet, a BFER may decapsulate the outer header, use the bitstring in the BOB header and perform an AND operation with the “For Us” bitstring. If the result is 0, it drops the packet. If the result is its own BFR identifier, it may decapsulate and handover the packet to the SBFD process to reflect the response.



```
while proto == "bob" :
    if (received_bitstring && for_us_bistring) != 0 {
        if (received_bitstring && for_us_bistring) == bfr-id {
            decapsulate;
           }
    else {
     drop;
     }
```

Figure 3

As illustrated in Figure 4 below, the BFIR may act as an ingress node for stream S1,G1 while BFER2, BFER3 and BFER4 serve as the egress nodes. The BFIR may be intended to validate the path while required to scale the response. In order to do so, it may generate the SBFD control packet with Your Discriminator set to BIER-Discrim and My Discriminator set to a local value. It may further encapsulate the control packet with the

BOB header and set the bitstring to 1010 with an intention that it is expecting the response from BFER2 and BFER4. The “Proto” field in the BOB header may be set to OAM. This may be further encapsulated with the BIER header with the bitstring set to 1110 (all receivers of S1, G1). The “Proto” field in the outer header may be set to BOB.

Upon receiving the packet, BFER2 and BFER4 may perform an AND operation with the bitstring in the BOB header and the local “For Us” bitstring. For example, BFER2 may receive the bitstring in the BOB header as 1010 and perform an AND operation with 0010. The resulting bitstring is 0010, which matches the local bit identifier. It may process this further and reflect the SBFD response to the BFIR. Similarly, BFER3 may perform an AND operation between 1010 and 0100, resulting in “0,” causing it to drop the packet.

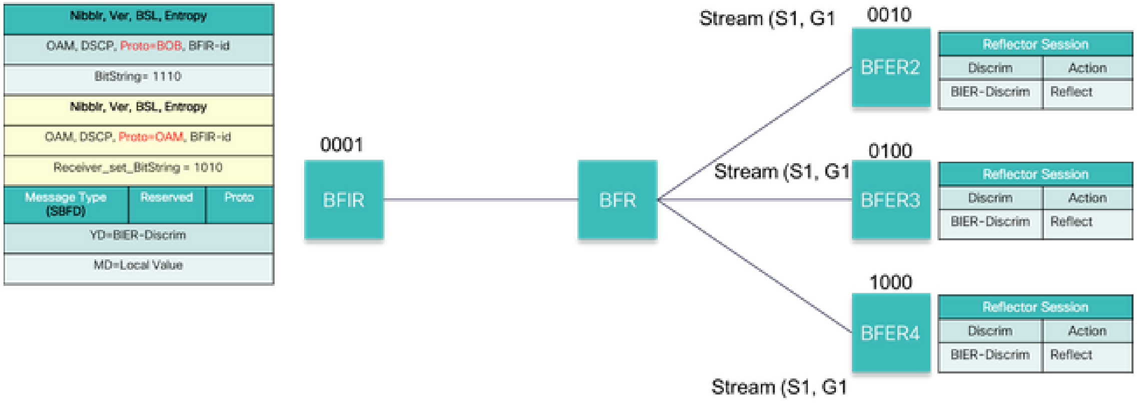


Figure 4

As defined in IETF RFC 8279, the forwarding may be influenced by a set of BFERs (“BitString” field) and entropy fields in the BIER header. The next protocol field does not play a role in the forwarding. As such, encoding the BOB does not influence how the packet is forwarded. This allows the bitstring and entropy field to be treated like data traffic while the inner header fields control the response. The response processing may be different from the path of the data packet(s), but this is not necessarily a concern because when the forward path to the bit positions is tested the response is largely irrelevant.

The BOB prototype may be similar to the BIER-in-BIER concept. The lookup semantic may perform an AND operation between the bitstring in the BOB header and the local “For Us” bitstring. Anycast BIER-Discrim may assign the same value on all edge BIER nodes. The combination of BOB, lookup, and BIER-Discrim may be used for scalable path validation.

In summary, techniques are described herein for a BOB prototype. An associated lookup semantic may also be provided. These techniques may help scale responses from receivers without changing the behavior or path on the transit nodes. It will be appreciated that operations relating to BIER as described herein may be generalized for any suitable bitmask technology or encapsulation.