June 2022

# Mechanism to Distinguish Between SR-TE and Non-TE S-BFD Return Paths at SR Head Node

Anonymous

# Mechanism to Distinguish Between SR-TE and Non-TE S-BFD Return Paths at SR Head Node

## ABSTRACT

Segment Routing is an emerging technology in Multiprotocol Label Switching (MPLS) network, Seamless Bidirectional Forwarding Detection (BFD) over these networks is used as Operations, Administration, and Maintenance (OAM) mechanisms to provide fast switchovers and thus meet customer Service Level Agreement (SLA). BFD running over Label-switch Paths (LSP's) can have MPLS/IP encapsulation in forward and backward directions. In case of Seamless BFD (S-BFD) running over Segment Routing-Traffic Engineered (SR-TE) networks, it may happen that BFD switches to a different MPLS/IP path instead of user configured MPLS path thus compromising the guaranteed SLA.

## DETAILED DESCRIPTION

Consider Figure 1 below representing a sample topology. S-BFD initiator session is configured on node A. The responder node B either has a S-BFD reflector session or general Segment Routing (SR) SID-list forwarding-plane.

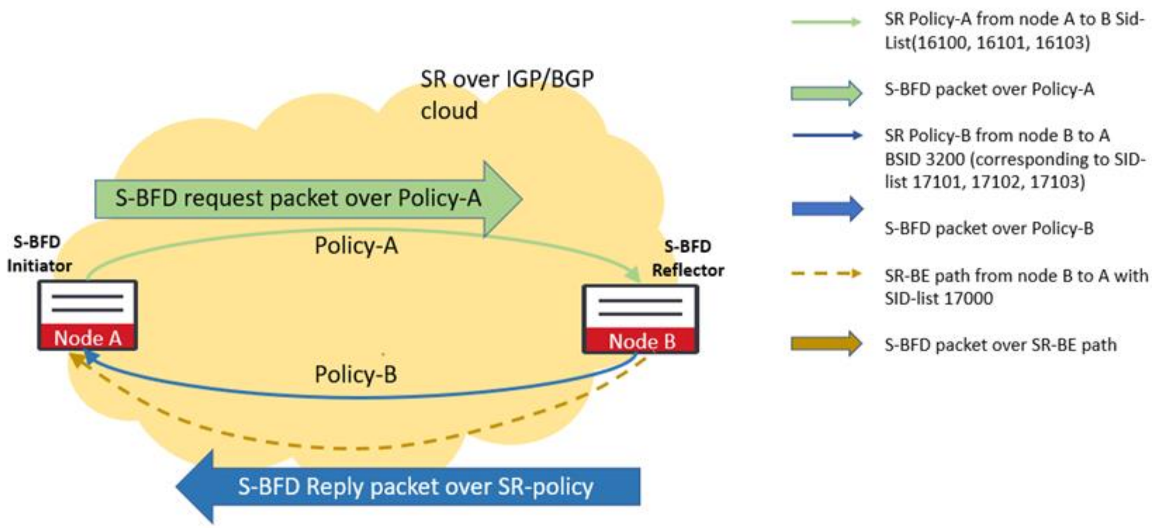


**Figure 1**

## Problem Description:

S-BFD is used for monitoring faults over SR networks. Thus, user configures S-BFD session on a SR-Policy or Candidate path on Initiator node A providing a BSID for some reverse SR-policy path. The S-BFD session is operationally up with BFD frames being MPLS encapsulated for forward and reverse path. During network fault on reverse path, policy is operationally down. In such a scenario, SR-policy path will move to SR-BE path, which shall have different label stack. This change in reverse path of S-BFD reply frames may go unnoticed by operator as S-BFD will still be operationally up without detecting any fault during change of encapsulation on reverse path.

In the same scenario, if protection is enabled on SR-policy using a backup candidate-path, but reverse path from Reflector has same BSID, for both the S-BFD session. In this case if S-BFD detects the fault on initiator due to fault on reply-path, S-BFD will make the backup Candidate-path operationally up, with S-BFD reply frames being sent over the same SR-BE or IP path (i.e., not with configured BSID label-stack). This will go unnoticed by user and thus shall compromise the SLA because of unreliability of new S-BFD reply path.

**Solution Description:**

It is proposed to leverage existing “diagnostic” fields of BFD reply packets when “state” field is Up, to carry information about the reverse path chosen by the responder node B. As per RFC standards and common implementation, “diagnostic” field is always zero (No Diagnostic) when “state” field is Up, as there is no further information to be communicated to peer when network path is fine.

```
                   +--+
ADMIN DOWN,        |  |
TIMER              |  V
                +------+   UP                +------+
                |      |-------------------->|      |----+
                | DOWN |                     |  UP  |    | UP
                |      |<--------------------|      |<---+
                +------+   ADMIN DOWN,       +------+
                           TIMER
```

**Figure 2**

Secondly, with reference to S-BFD initiator FSM above (from RFC 7880), when reflector session is admin-enabled, S-BFD reply packet’s “state” is always Up. So, all use cases of network fault and recovery are concerned with only the Up state in both BFD request and reply packets.

The processing flows at initiator node A and responder node B are depicted in the flowcharts in Figure 3, Figure 4, and Figure 5.

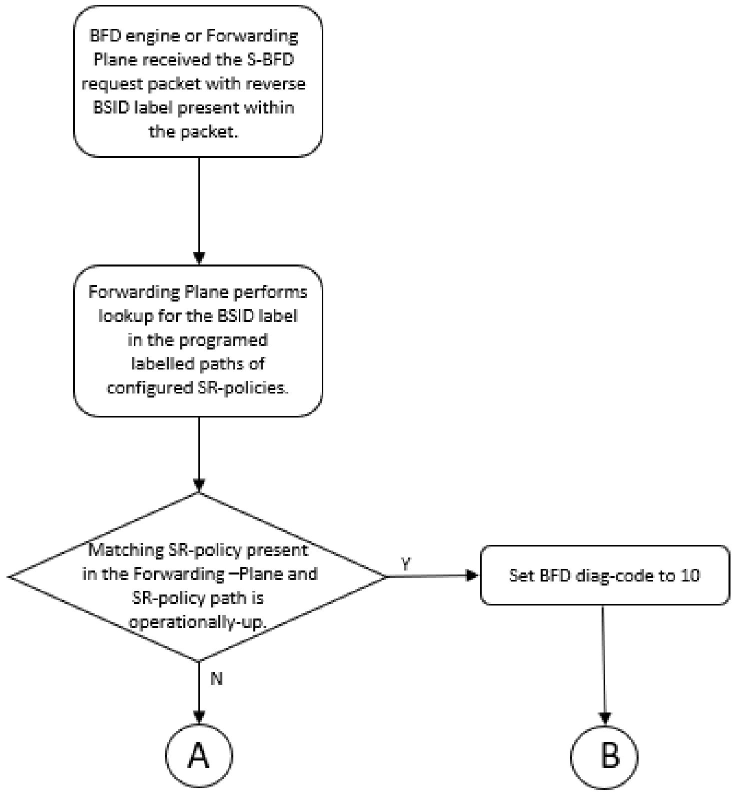


**Figure 3**

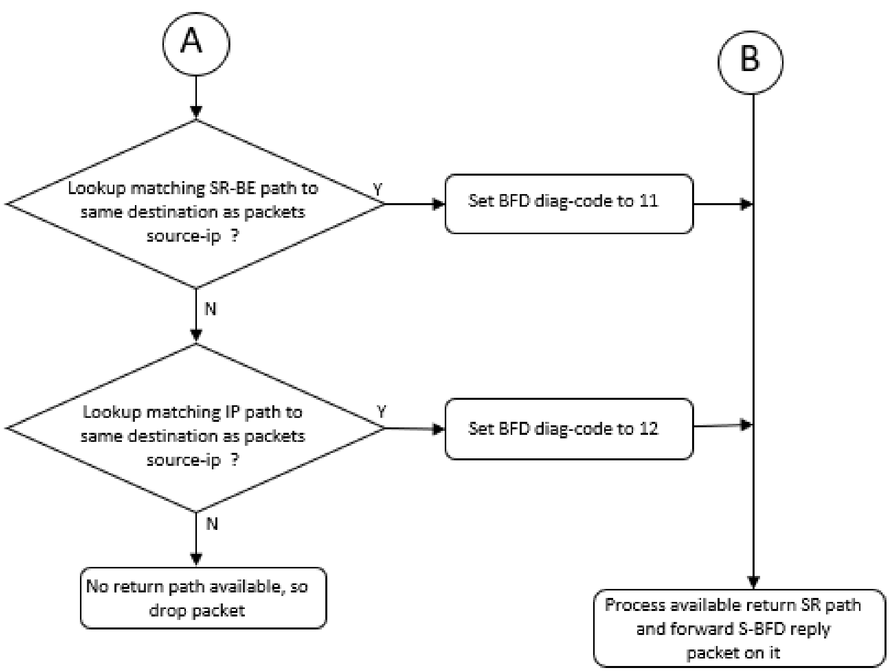


**Figure 4**

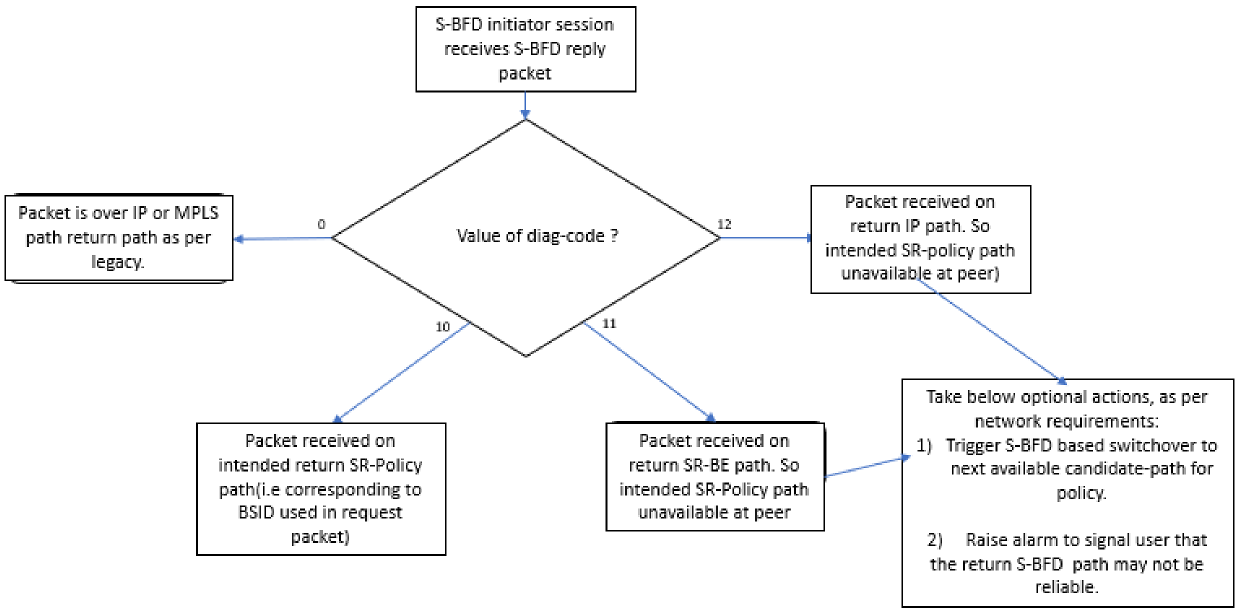


**Figure 5**

The solution implies that initiator node A can take more informed switchover/ reversion decisions. In particular, below two key scenarios can only be handled using this proposed solution:

1) **Switchover scenario:**
   Assume a fault occurs in reverse SR-policy path and responder node B does a fast switchover to SR-BE path, such that S-BFD initiator session on node A remains hitless. With the proposed solution, S-BFD initiator session at A will know (via changed "diagnostic" value) that although BFD reply packets are received, but they are not via intended SR-policy path. So, SR-policy on node A may decide to switchover to an alternate candidate-path, since the BFD return path is sub-optimal.

2) **Reversion scenario:**
   Assume S-BFD initiators are configured for each candidate-path for forward SR-policy, and a fault in reverse SR-policy caused a switchover from primary to backup candidate-path. Now, the S-BFD session on primary candidate-path comes Up again on reverse SR-BE path. So, SR-policy on node A may decide to avoid reversion, since it knows that the although the primary candidate-path's S-BFD session is Up on SR-BE path, but the network fault still exists in reverse SR-policy path.

The present solution leverages "diagnostic" field of BFD packet for transmitting path information to initiator S-BFD session. This field is practically unused when initiator session state is Up. With the present solution it is now possible for S-BFD initiator session to be aware about the actual return path (SR-policy, SR Best-Effort, or IP) taken by BFD reply packet.

Figure 6 and Figure 7 show the problem scenario where Figure 6 shows S-BFD is operationally up over Policy Sid-List from initiator and reflector, and Figure 7 shows S-BFD is operationally up over Policy Sid-List from Initiator and reply over IP from Responder.

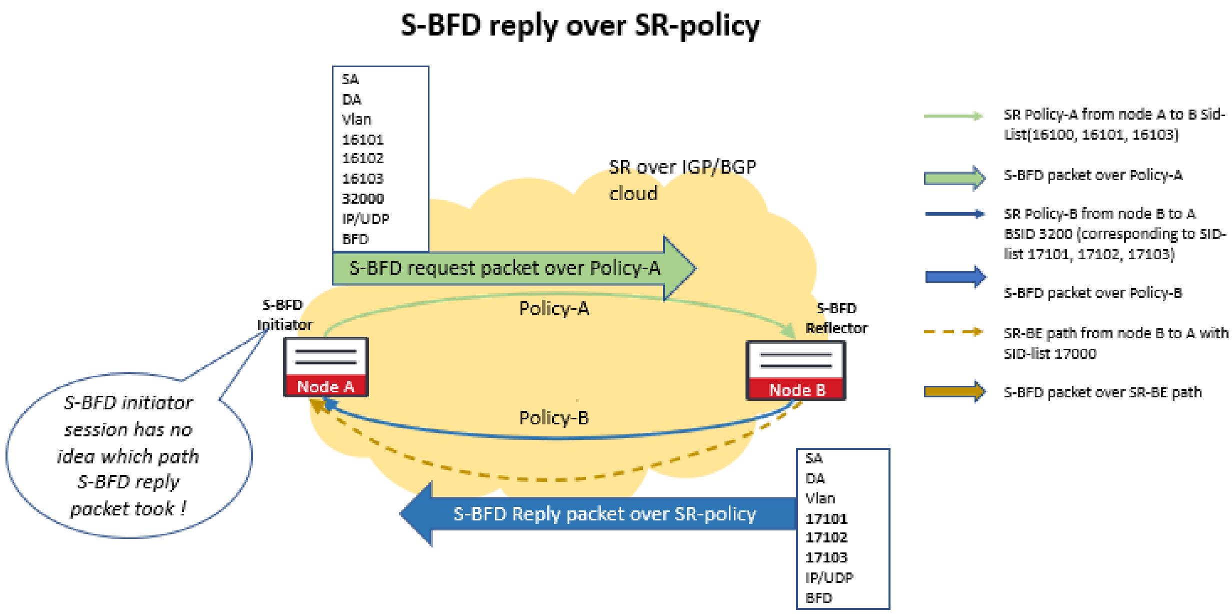


**Figure 6**

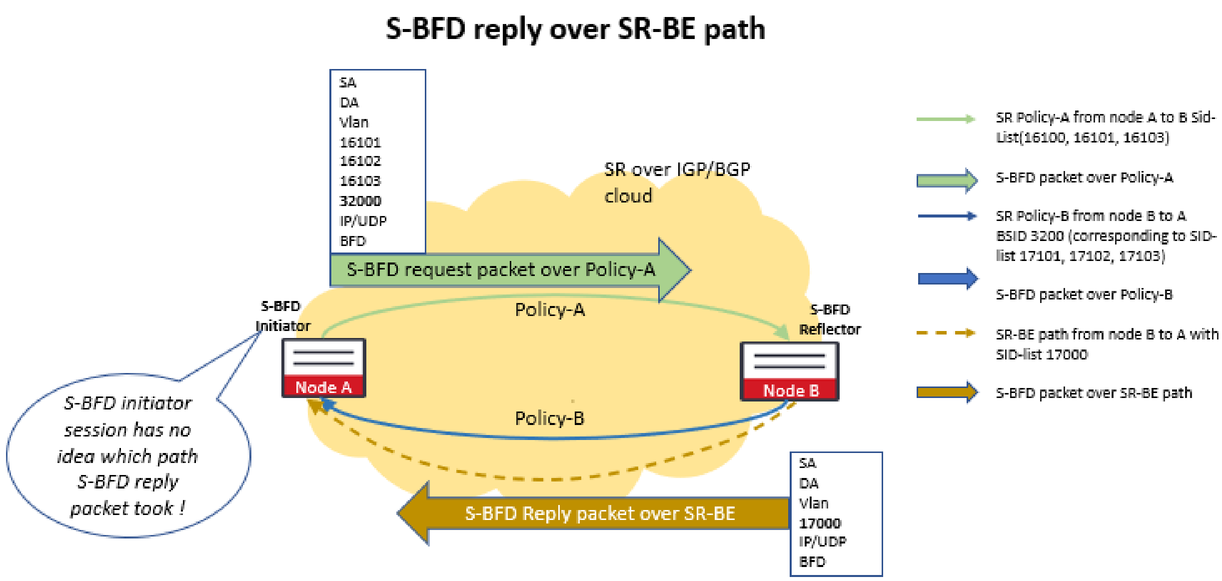


**Figure 7**

Figure 8 and Figure 9 show the solution scenario where Figure 8 shows S-BFD is operationally up over Policy Sid-List from initiator and reflector, and Figure 9 shows S-

BFD is operationally up over Policy Sid-List from Initiator and reply over IP from Responder.

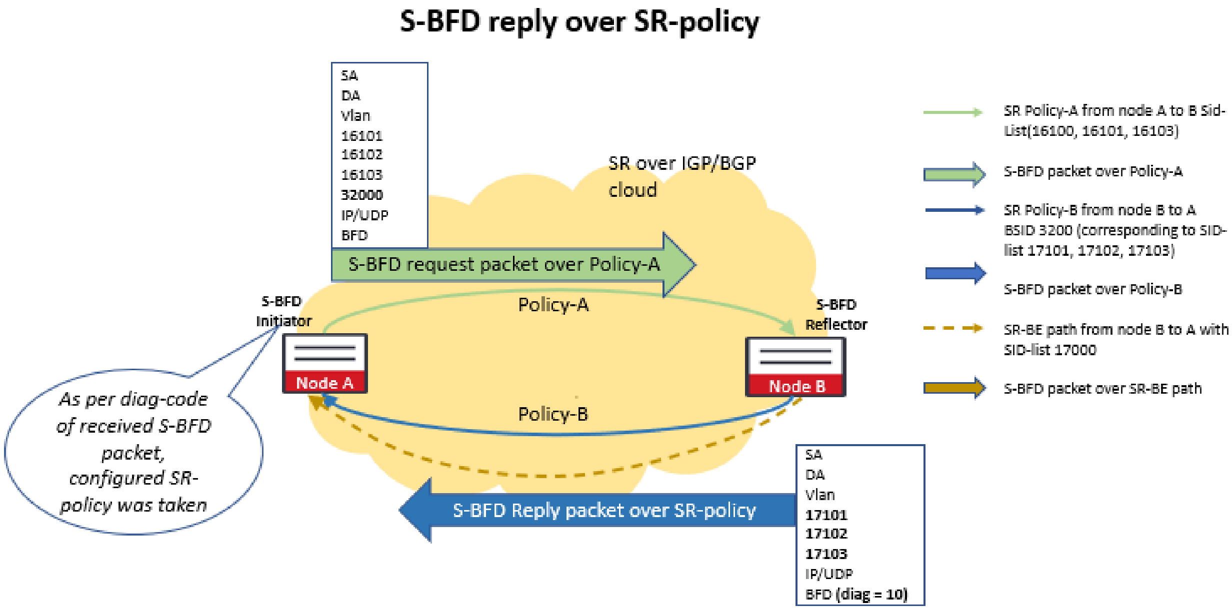


**Figure 8**

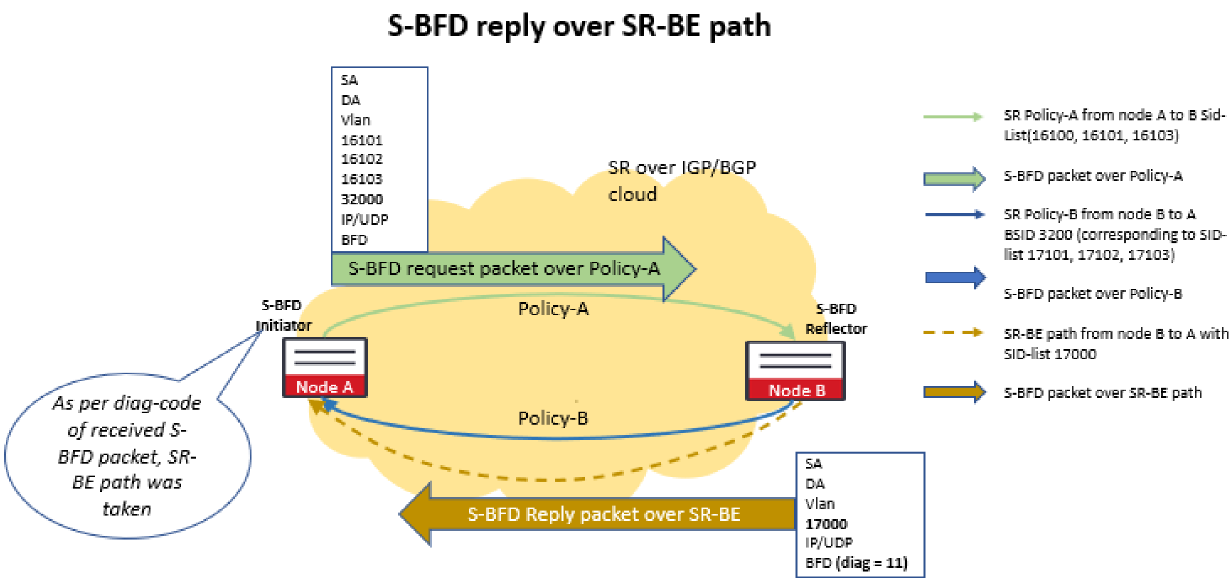


**Figure 9**

It will be appreciated that some embodiments described herein may include one or more generic or specialized processors ("one or more processors") such as microprocessors, digital signal processors, customized processors, and Field-Programmable Gate Arrays (FPGAs) and unique stored program instructions (including both software and firmware) that control the one or more processors to implement, in conjunction with certain nonprocessor circuits, some, most, or all of the functions of the methods and/or systems described herein. Alternatively, some or all functions may be implemented by a state machine that has no stored program instructions, or in one or more Application-Specific Integrated Circuits (ASICs), in which each function or some combinations of certain of the functions are implemented as custom logic. Of course, a combination of the aforementioned approaches may be used. Moreover, some embodiments may be implemented as a non-transitory computer-readable storage medium having computer-readable code stored thereon for programming a computer, server, appliance, device, etc. each of which may include a processor to perform methods as described and claimed herein. Examples of such computer-readable storage mediums include, but are not limited to, a hard disk, an optical storage device, a magnetic storage device, a ROM (Read Only Memory), a PROM (Programmable Read-Only Memory), an EPROM (Erasable Programmable Read-Only Memory), an EEPROM (Electrically Erasable Programmable Read-Only Memory), Flash memory, and the like. When stored in the non-transitory computer-readable medium, the software can include instructions executable by a processor that, in response to such execution, cause a processor or any other circuitry to perform a set of operations, steps, methods, processes, algorithms, etc.

Although the present disclosure has been illustrated and described herein with reference to preferred embodiments and specific examples thereof, it will be readily apparent to those of ordinary skill in the art that other embodiments and examples may perform similar functions and/or achieve like results. All such equivalent embodiments and examples are within the spirit and scope of the present disclosure.