SPRING
Internet-Draft
Intended status: Informational
Expires: September 5, 2019

C. Filsfils
F. Clad, Ed.
P. Camarillo, Ed.
Cisco Systems, Inc.
A. AbdelSalam
Gran Sasso Science Institute
S. Salsano
Universita di Roma "Tor Vergata"
O. Bonaventure
Universite catholique de Louvain
J. Horn
J. Liste
Cisco

March 4, 2019

SRv6 interoperability report
draft-filsfils-spring-srv6-interop-02

Abstract

Segment Routing (SR) can be applied to the IPv6 data plane by leveraging a new type of routing extension header, called Segment Routing Header (SRH). An SRH contains an ordered list, or sequence, of segments representing topological or service-based instructions, or any combination of those.

This draft provides an overview of IPv6 Segment Routing (SRv6) implementations and interoperability. It makes an inventory of the various pieces of hardware and software that have been demonstrated to support the processing of an SRH, and details some interoperability scenarios that have been showcased at a public event.

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 September 5, 2019.

Table of Contents

1. Introduction

Segment Routing (SR), defined in [RFC8402], allows a node to steer packets through a controlled sequence of instructions, called segments, by prepending an SR header to the packets. The IPv6 instantiation of Segment Routing (SRv6) leverages the Segment Routing Header (SRH), a new type of IPv6 routing extension header defined in [I-D.ietf-6man-segment-routing-header].

As described in [I-D.filsfils-spring-srv6-network-programming], an SRv6 segment is a network instruction composed of a locator and a function that is encoded as an IPv6 address. The segment locator is an IPv6 prefix used by routing devices to steer the packet along the

shortest IGP path up to the node that advertises this prefix. Since the active segment is placed in the Destination Address field of the IPv6 header, steering by intermediate devices only involves plain IPv6 forwarding and does not require any SR-specific capability. Once the packet reaches the node indicated by the segment locator, the segment function is identified and the packet is processed accordingly. Although the SR functions are locally defined on each node, a set of general purpose functions have been proposed for standardization in [I-D.filsfils-spring-srv6-network-programming] in order to ease interoperability. This set is further extended with specific purposes functions in [I-D.ietf-dmm-srv6-mobile-uplane] and [I-D.xuclad-spring-sr-service-programming].

2. Routing platforms supporting SRH processing

The hardware and software platforms listed below have been demonstrated to support the processing of an SRH as described in [I-D.ietf-6man-segment-routing-header]. This section also indicates the supported SRv6 functions and transit behaviors on open-source software.

Open-source platforms:

- Linux kernel[ref-1][ref-2]: End, End.X, End.T, End.DX2, End.DX6, End.DX4, End.DT6, End.B6, End.B6.Encaps, T.Insert, T.Encaps, T.Encaps.L2

- Linux srext module: End, End.X, End.DX2, End.DX6, End.DX4, End.AD, End.AM

- FD.io VPP: End, End.X, End.DX2, End.DX6, End.DX4, End.DT6, End.DT4, End.B6, End.B6.Encaps, End.AS, End.AD, End.AM, T.Insert, T.Encaps, T.Encaps.L2

Cisco:

- Cisco ASR 1000 with IOS XE engineering code

- Cisco ASR 9000 with IOS XR engineering code

- Cisco NCS 5500 with IOS XR engineering code

Barefoot Networks:

- Barefoot Networks Tofino

3. Applications supporting SRH

In addition to the aforementioned routing platforms, the following open-source applications have been extended to support the processing of IPv6 packets containing an SRH. For Wireshark, tcpdump, iptables and nftables, these extensions have been included in the mainstream version.

- Wireshark[ref-3]
- tcpdump[ref-4]
- iptables[ref-5][ref-6]
- nftables[ref-7]
- Snort[ref-8]

4. Interoperability testing

The following interoperability testing scenarios were publicly showcased on August 21-24, 2017 at the SIGCOMM conference.

Five different implementations of SRv6 behaviors were used for this testing:

- Software implementation in Linux using the srext kernel module created by University of Rome, Tor Vergata, Italy.
- Software implementation in the FD.io Vector Packet Processor (VPP) virtual router.
- Hardware implementation in Barefoot Networks Tofino NPU using the P4 programming language.
- Hardware implementation in Cisco NCS 5500 router using commercially available NPU.
- Hardware implementation in Cisco ASR 1000 router using custom ASIC.

4.1. Testbed configuration

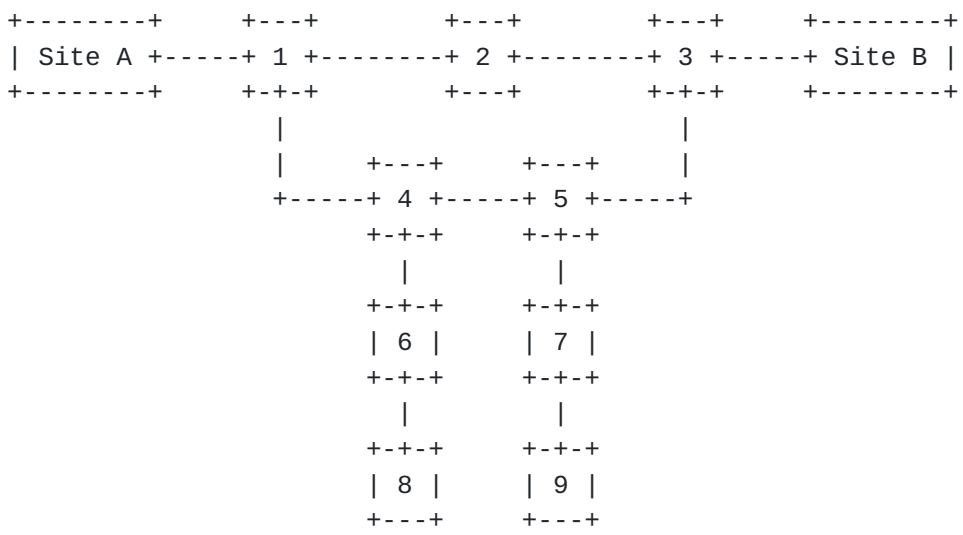

Figure 1: Testbed topology

As shown in the figure above, the testbed consisted of 9 different nodes:

- Nodes 1 and 3 are Cisco ASR 1000 routers running IOS XE engineering code
- Node 2 is a Cisco ASR 9000 router (realizing IPv6 forwarding only)
- Node 4 is a Barefoot Wedge 100BF router
- Node 5 is a Cisco NCS 5500 router running IOS XR engineering code
- Node 6 is a Linux server running the srext kernel module
- Node 7 is a Linux server running the FD.io VPP
- Node 8 is a Linux container running Snort
- Node 9 is a Linux container running iptables

On sites A and B the TRex software is used for traffic generation.

In addition, after every layer 3 operation in the network, Wireshark traffic analyzer is used to verify proper functionality.

4.2. Scenarios

4.2.1. L3 VPN

This scenario covers simple L3 VPN functionality for IPv6 traffic using the SRv6 behaviors T.Encaps and End.DX6 in conjunction with regular IPv6 routing.

- IPv6 traffic is generated in site A and sent towards a destination in site B.
- Node 1 performs the T.Encaps operation on the received traffic. Each packet is encapsulated with an IPv6 header and an SRH containing 1 segment, owned by node 3, then forwarded along the shortest path to 3.
- Node 2 performs standard IPv6 forwarding.
- Node 3 performs the End.DX6 function on the received traffic. Each packet is decapsulated then forwarded on the appropriate interface towards site B.
- Traffic generator in site B captures and verifies the received traffic.

4.2.2. L3 VPN with traffic engineering

This scenario covers L3 VPN overlay with underlay optimization in a single SRv6 encapsulation (IPv6 header + SRH). It leverages the same T.Encaps and End.DX6 behaviors as the first scenario, combined with the End and End.X functions.

- IPv6 traffic is generated in site A and sent towards a destination in site B.
- Node 1 performs the T.Encaps operation on the received traffic. Each packet is encapsulated with an IPv6 header and an SRH containing 3 segments, respectively owned by nodes 4, 5 and 3. The outer IPv6 Destination Address is set to the first segment and the packets are forwarded accordingly.
- Node 4 performs the End function, forwarding the packets on the shortest paths towards node 5.
- Node 5 performs the End.X function, forwarding the packets on a specific interface towards node 3.
- Node 3 performs the End.DX6, decapsulating and then forwarding each packet on the appropriate interface towards site B.

- Traffic generator in site B captures and verifies the received traffic.

4.2.3. L3 VPN with traffic engineering and service chaining

This scenario covers L3 VPN overlay with underlay optimization and service chaining. Snort and iptables are SR-unaware services in this situation and accessed via SRv6 dynamic proxy endpoint functions implemented on nodes 6 and 7.

- IPv6 traffic is generated in site A and sent towards a destination in site B.
- Node 1 performs the T.Encaps operation on the received traffic. Each packet is encapsulated with an IPv6 header and an SRH containing 5 segments, respectively owned by nodes 4, 6, 5, 7 and 3. The outer IPv6 Destination Address is set to the first segment and the packets are forwarded accordingly.
- Node 4 performs the End.X function, forwarding the packets on a specific interface towards node 6.
- Node 6 performs the End.AD function, sending the inner IPv6 packet via 8 (Snort) and restoring the encapsulation on the way back.
- Node 5 performs the End function, forwarding the packets on the shortest paths towards node 7.
- Node 7 performs the End.AD function, sending the inner IPv6 packet via 9 (iptables) and restoring the encapsulation on the way back.
- Node 3 performs the End.DX6, decapsulating and then forwarding each packet on the appropriate interface towards site B.
- Traffic generator in site B captures and verifies the received traffic.

5. Contributors

David Lebrun, Prem Jonnalagadda and Milad Sharif substantially contributed to the content of this document.

6. Acknowledgements

TBD

7. IANA Considerations

This document does not require any action from IANA.

8. Security Considerations

This document does not introduce any security consideration.

9. Informative References

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-07 (work in progress), February 2019.

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-16 (work in progress), February 2019.

[I-D.ietf-dmm-srv6-mobile-uplane]
Matsushima, S., Filsfils, C., Kohno, M., Camarillo, P., daniel.voyer@bell.ca, d., and C. Perkins, "Segment Routing IPv6 for Mobile User Plane", draft-ietf-dmm-srv6-mobile-uplane-03 (work in progress), October 2018.

[I-D.xuclad-spring-sr-service-programming]
Clad, F., Xu, X., Filsfils, C., daniel.bernier@bell.ca, d., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", draft-xuclad-spring-sr-service-programming-01 (work in progress), October 2018.

[ref-1] "Implementing IPv6 Segment Routing in the Linux Kernel", July 2017, <https://doi.org/10.1145/3106328.3106329>.

[ref-2] "Reaping the Benefits of IPv6 Segment Routing", October 2017, <https://inl.info.ucl.ac.be/publications/reaping-benefits-ipv6-segment-routing>.

[ref-3] "Add support for Segment Routing (Type 4) Extension Header", June 2016, <https://code.wireshark.org/review/gitweb?p=wireshark.git;a=commit;h=d6e9665872989c5f343fce47484abe415d77486c>.

[ref-4] "Add support for IPv6 routing header type 4", December 2017, <https://github.com/the-tcpdump-group/tcpdump/commit/9c33608cb2fb6a64e1b76745efa530a63de08100>.

[ref-5] "[net-next,v2] netfilter: add segment routing header 'srh' match", January 2018, <https://patchwork.ozlabs.org/patch/856578/>.

[ref-6] "[iptables,v2] extensions: add support for 'srh' match", January 2018, <https://patchwork.ozlabs.org/patch/859206/>.

[ref-7] "[nft] nftables: Adding support for segment routing header 'srh'", March 2018, <http://patchwork.ozlabs.org/patch/879061/>.

[ref-8] "IPv6 Segment Routing (SRv6) aware snort", March 2018, <https://github.com/SRouting/sr-snort>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

Authors' Addresses

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Francois Clad (editor)
Cisco Systems, Inc.
France

Email: fclad@cisco.com

Pablo Camarillo Garvia (editor)
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

Ahmed AbdelSalam
Gran Sasso Science Institute
Italy

Email: ahmed.abdelsalam@gssi.it

Stefano Salsano
Universita di Roma "Tor Vergata"
Italy

Email: stefano.salsano@uniroma2.it

Olivier Bonaventure
Universite catholique de Louvain
Belgium

Email: olivier.bonaventure@uclouvain.be

Jakub Horn
Cisco
USA

Email: jakuhorn@cisco.com

Jose Liste
Cisco
USA

Email: jliste@cisco.com