ETSI GR IPE 009 V1.1.1 (2022-07)

IPv6 Enhanced innovation (IPE); SRv6 based SFC for Value-Added Service in an Operator Network

Disclaimer

The present document has been produced and approved by the IPv6 Enhanced innovation (IPE) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference

DGR/IPE-009

Keywords

IPv6, network, service, SFC

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

If you find a security vulnerability in the present document, please report it through our Coordinated Vulnerability Disclosure Program:
https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) IPv6 Enhanced innovation (IPE).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

From the trend evolution of ubiquitously distributed Cloud, Edge, and terminal, the new generation of information network is transforming from the network infrastructure with information transmission as the core to the intelligent Cloud network infrastructure with integrated computing, storage, and network resources.

With the development of networks and services, more and more value-added services (like FW, DPI) need to be deployed on-demand. Originally, a variety of services could be deployed in the central Cloud resource pool. With the development of MEC technology, and the requirements of network performance such as network quality and request delay, as well as the increasing demand for flexible addition and reduction of service functions, operators will deploy service functions on Edge nodes. Through the network, Edge nodes realize collaboration and service functions are connected in series to meet users' demand for flexible scheduling of service functions and solve problems such as response time, resource optimization, and network efficiency.

Service Function Chaining (SFC) provides a flexible solution. SFC encapsulation is added to data packets to implement on-demand deployment and dynamic adjustment of value-added functions. In the Segment Routing IPv6 (SRv6) network, the SFC function is realized through the programmable forwarding path. Compared with Network Service Header (NSH) and other solutions, SFC has obvious advantages in deployment complexity and other aspects. At present, some Value-Added Service (VAS) manufacturers and routing equipment manufacturers have also developed products based on SRv6 and conducted relevant SFC tests. There was research on SFC standards based on SRv6 in IETF. Research on SFC technologies based on SRv6 is necessary to promote technology development and maturity of the industrial chain.

The present document shows how operators can deploy SRv6-based SFC and focuses on scenario requirements, SRv6-based SFC techniques, SRv6-aware firewall, development architecture, and experiments tests.

Clause 4 describes two usage scenarios from the operator's viewpoint. The first is about Smart services for government/enterprise clients, and the second is about smart services for home terminals. These two scenarios are very important for operators, who are actively evolving from traditional communication service providers to integrated information service enterprises.

The key technical points of realizing SFC based on SRv6 are introduced in clause 5, including basic concepts, two different modes for realizing SFC depending on.

Clause 6 discusses the SRv6-aware network firewall use case.

Clause 7 focuses on practical deployment solutions for operators, including experiment network design, the Orchestrator System, reference points, and the flow Sequence diagram of SFC development.

Clause 8 carries out the experimental tests and analyses based on the deployment solution in clause 7. The tests verify the feasibility of SRv6-based SFC.

Clause 9 gives the conclusion and future work.

1 Scope

The present document guides an operator to flexibly and on-demand develop SFC (Service Function Chain) based on SRv6.

First, the present document analyses the requirements and scenarios of SFC in an operator network; then describes scheme design for operator providing value-added services using SRv6 based SFC.

The main content includes five major blocks:

- Usage Scenarios Description - clause 4.
- SRv6-based SFC - clause 5.
- SRv6-aware Network Firewall - clause 6.
- Development Practice of SRv6 based SFC in Operator Network - clause 7.
- Experimental tests - clause 8.
- Conclusion - clause 9.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user concerning a particular subject area.

[i.1] IETF RFC 7665: "Service Function Chaining (SFC) Architecture".

[i.2] IETF RFC 7498: "Problem Statement for Service Function Chaining".

[i.3] IETF RFC 8598: "An MPLS-Based Forwarding Plane for Service Function Chaining".

[i.4] IETF RFC 8249: "Service Function Chaining (SFC) Operations, Administration, and Maintenance (OAM) Framework".

[i.5] IETF RFC 8300: "Network Service Header (NSH)".

[i.6] IETF RFC 8986: "Segment Routing over IPv6 (SRv6) Network Programming".

[i.7] Understanding Segment Routing IPv6.

NOTE: Available at https://support.huawei.com/enterprise/en/doc/EDOC1000173015/d169625f/understanding-segment-routing-ipv6.

[i.8] IETF RFC 8402: "Segment Routing Architecture".

[i.9] IETF RFC 8200: "Internet Protocol, Version 6 (IPv6) Specification".

[i.10] IETF draft-ietf-spring-sr-service: "Service Programming with Segment Routing".

NOTE: Available at draft-ietf-spring-sr-service-programming-06 - Service Programming with Segment Routing.

[i.11] Ahmed Abdelsalam, PHD Thesis: "Service Function Chainning with Segment Routing".

NOTE: Available at https://iris.gssi.it/retrieve/handle/20.500.12571/9921/2721/service-function-chaining-with-segment-routing.pdf.

[i.12] IETF RFC 8924 "Service Function Chaining (SFC) Operations, Administration, and Maintenance (OAM) Framework".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

abstract topology: used for orchestrating service functions that are different from network topology, just including some key network nodes like the nodes connecting to computing power resource pools, the access nodes, and the nodes which are used for connecting to other network domains

service path: result of orchestrating service functions according to abstract topology, consisting of a set of ordered nodes that the service passes through, including the access nodes, the necessary intermediate network nodes, and the nodes connected to the computing resource pools

SFC (Service Function Chaining): used to describe the definition and instantiation of an ordered list of instances of such service functions, and the subsequent "steering" of traffic flows through those service functions

NOTE: IETF has issued a series of standards for SFC, such as IETF RFC 7498 [i.2], IETF RFC 7665 [i.1], IETF RFC 8924 [i.12], and so on.

SR (Segment Routing): proposed in 2013 and borrows some ideas from source routing, SR aims to combine different segments into a path and insert segment information into packets at the ingress of the path to guide packet forwarding

NOTE: A transit node only needs to forward the packet according to the segment information carried in the packet. Each path segment - referred to simply as a segment - is identified by a Segment Identifier (SID).

SRv6 (Segment Routing IPv6): next-generation IP bearer protocol that combines Segment Routing (SR) and IPv6. Utilizing existing IPv6 forwarding technology, SRv6 implements network programming through flexible IPv6 extension headers

NOTE 1: SRv6 reduces the number of required protocol types, offers great extensibility and programmability, and meets the diversified requirements of more new services. It also provides high reliability and offers exciting Cloud service application potential.

NOTE 2: SRv6 was mentioned in the SR Architecture document as early as 2013 when the SR was first proposed: *"The Segment Routing architecture can be directly applied to the MPLS dataplane with no change on the forwarding plane. It requires minor extension to the existing link-state routing protocols. Segment Routing can also be applied to IPv6 with a new type of routing extension header.* "See IETF RFC 8402 [i.8].

NOTE 3: In February 2021, IETF released IETF RFC 8986 [i.6] "Segment Routing over IPv6 (SRv6) Network Programming". The present document defines the SRv6 Network Programming concept and specifies the base set of SRv6 behaviors that enables the creation of interoperable overlays with underlay optimization. SRv6 Network Programming divides a 128-bit SRv6 SID into fields including Locator and Function. Locator provides the routing capability, and Function can represent processing behaviors and identify services.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AS	Autonomous System
BNG	Broadband Network Gateway
CPE	Customer Premises Equipment
DC	Data Center
DDoS	Distributed Denial of Service
DPI	Deep Packet Inspection
DT	Decapsulation and Table Lookup
FW	FireWall
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
MEC	Multi-access Edge Computing
MPLS	Multi-Protocol Label Switching
NF	Network Function
NSH	Network Service Header
OA	Office Automation
PDU	Protocol Data Unit
PE	Provider Edge
SC	Service Classifier
SERA	SEgment Routing Aware Firewall
SF	Service Function
SFC	Service Function Chaining
SFF	Service Function Forwarder
SFP	Service Function Path
SID	Segment ID
SLA	Service Level Agreement
SQL	Structured Query Language
SR	Segment Routing
SRH	Segment Routing Header
SRv6	Segment Routing IPv6
TLV	Type-Length-Value
VAS	Value-Added Service
VLAN ID	Virtual Local Area Network Identifier
VNF	Virtual Network Function
VPN	Virtual Private Network

4 Usage scenarios description

4.1 Smart services for government/enterprise clients

Today, operators are actively evolving from traditional communication service providers to integrated information service enterprises. Operators need to organize many value-added services and network application products of information applications to meet customer needs and better improve product sales.

Government and enterprise business operators target government departments, enterprises, and public institutions. Government and enterprise businesses related to big data, Internet of Things (IoT), Cloud computing, mobile OA (Office Automation), mobile law enforcement, and individual law enforcement form an important part of an operator's activity. Operators provide clients with a variety of Value-Added Service (VAS) products through virtualization and Cloud computing technology. Therefore, how to effectively and quickly design a variety of combined product service chains to meet the needs of different customers is an interesting and important question to investigate.

The following scenarios describe the smart service for government or enterprise clients.

Through the shared VAS resource pools, value-added services such as FW, DDoS, and parental control can be provided for existing services such as internet services or government/Enterprise private services to increase the income of operators.

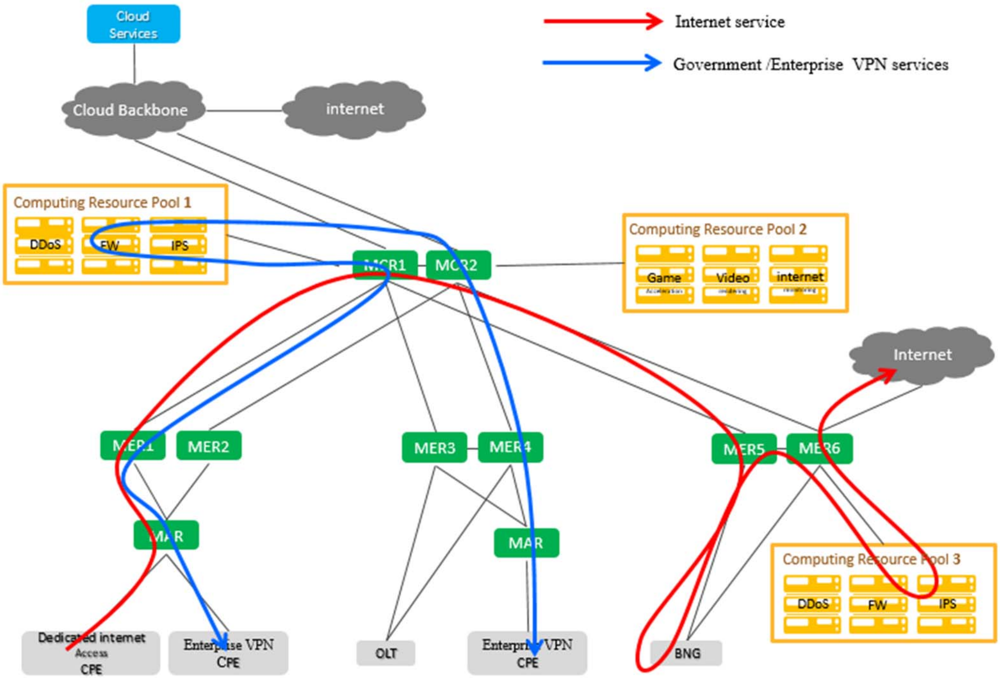


Figure 1: Smart Services for Government/Enterprise Clients

In Figure 1, the three yellow boxes are Computing Resource Pools, which provide different types of value-added services, such as Distributed Denial of Service (DDoS), FireWall, game acceleration, etc.

Through the shared VAS resource pools, value-added services such as FW, DDoS, and parental control can be provided for existing services such as internet services or government/enterprise private services to increase the revenue of an operator.

The blue line represents enterprise service data that can be used to subscriber's personal services.

4.2 Smart services for home terminals

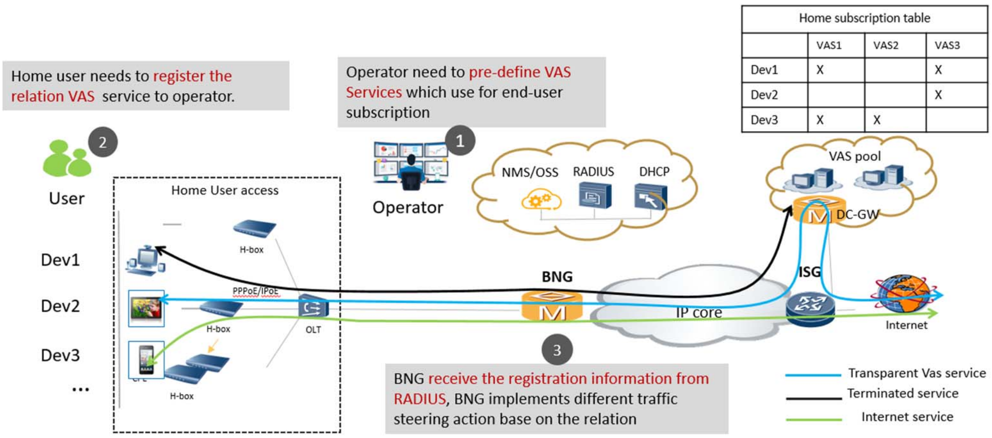
The smart home has developed from the initial era of a single intelligent terminal product to the era of interconnection and interaction of multiple intelligent terminals. This means a new form of smart home products and a new business model. Moreover, operators play a key role in the implementation of smart home interconnection, they will continue to provide VASs for smart home terminals and promote the gradual improvement of smart home ecology.

The following scenario is for home end-users with mobile phones, tablets, home computers, etc. The process is as follows:

- Step 1:The operator deploys value-added services and publishes subscribable service messages to all the home terminal users.
- Step 2: The home terminal users register the needed VAS services from the operator.

- Step 3:The operator's Broadband Network Gateway (BNG) receives the registration information and takes different traffic steering actions based on regulations.

In this way, each end-user can enjoy their customized value-added services.



Home user needs to register the relation VAS service to operator.

Operator need to pre-define VAS Services which use for end-user subscription

BNG receive the registration information from RADIUS, BNG implements different traffic steering action base on the relation

Home subscription table			
	VAS1	VAS2	VAS3
Dev1	X		X
Dev2			X
Dev3	X	X	

Figure 2: Smart Services for Home Terminals

The above business model is a challenge for operators. It requires operators to be able to quickly schedule the computing power resources according to home terminal users' needs but also establishes the service function chain with as little configuration as possible to reduce network errors and facilitate the management of the service function chain.

5 Why SRv6-based SFC

5.1 Related concepts

5.1.1 Concept of SRv6

SRv6's natural programmable capabilities make it better to meet new network service requirements. Thanks to its compatibility with IPv6, it simplifies the deployment and implementation of network services. SRv6 not only breaks down the boundary between Cloud and network but enables operators to avoid becoming the providers of simple pipes and extend networks to user terminals and service resource pools.

Segment Routing v6 (SRv6) protocol is constructed around IPv6 and integrates smoothly into existing IPv6 deployments. With the depletion of global IP addresses, the network gradually transitions from IPv4 to IPv6. SRv6 technology is developing rapidly and has been widely used in kinds of applications. The development of SRv6 is overwhelming.

SRv6 is inseparable from IPv6. As defined in IETF RFC 8200 [i.9], an IPv6 packet consists of three parts:

1) IPv6 basic header: it has eight fields and a fixed length of 40 octets. This header is required in every IPv6 packet to provide basic packet forwarding information which is parsed by all devices on the corresponding forwarding path.

2) IPv6 extension header: an IPv6 packet can carry one or more extension headers or none at all. The source node of a packet adds one or more extension headers to the packet only when other nodes may perform special handling.

3) Upper-layer Protocol Data Unit (PDU) is usually composed of an upper-layer protocol header and its payload.

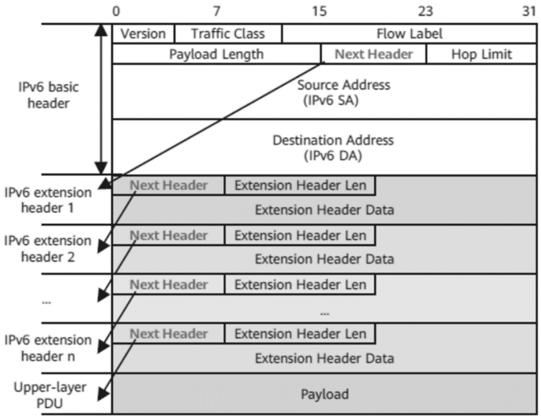


Figure 3: IPv6 packet format

To implement SR based on the IPv6 forwarding plane, a new type of IPv6 RH called Segment Routing Header (SRH) is defined. The SRH, which the ingress adds to each IPv6 packet, stores IPv6 path constraint information (segment lists) to specify an IPv6 explicit path. Transit nodes forward the packets according to the path information contained in the SRH.

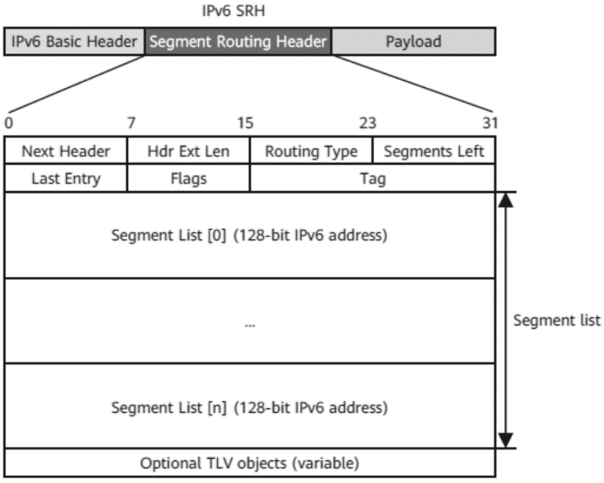


Figure 4: SRH structure

Although SRv6 SIDs use the IPv6 address format, they are not typical IPv6 addresses. Each SRv6 SID has 128 bits, meaning that it can represent almost anything. To avoid wasting such a large address space only for route forwarding, SRv6 designers took a clever approach when designing SIDs.

SRv6 SID usually consists of the Locator, Function, and Arguments parts expressed in the Locator:Function:Arguments format. The Locator part occupies the most significant bits in the IPv6 address, the Function part follows, and the Argument part occupies the remaining bits.

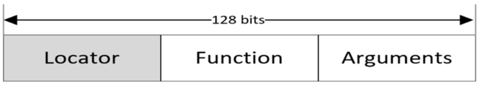


Figure 5: SRv6 SID structure

5.1.2 SRv6 characteristics

The main characteristics of SRv6 [i.7] are:

1) Good flexibility and Programmability: The extension capability of SRv6 SID provides SRv6 with network programming capabilities. The service orchestrator can specify networks and applications (service chains) based on SLAs and service requirements to provide flexible programmability.

2) Promotion of integration of Cloud and network: As SRv6 has the native IPv6 attribute, and both SRv6 and common IPv6 packets have the same packet header, SRv6 can implement communication between network nodes by leveraging only IPv6 reachability. Data Center Networks can easily support IPv6. Thereby, with SRv6 technology, an operator's network can be deployed in DCs and even extended to user terminals.

3) Easy for cross-as applications deployment or large-scale applications deployment: With IPv6 reachability, SRv6 can be easily deployed across Autonomous Systems (ASs). Host routes do not need to be flooded across ASs, and only aggregated routes need to be imported. This greatly reduces the number of routes and simplifies routing policies. SIDs that use IPv6 address space are suitable for large-scale network planning. Correspondingly, MPLS label space is limited, and unified planning and maintenance of device SIDs are complex for large-scale application development.

4) Compatibility with Existing Networks: SRv6 is compatible with existing IPv6 networks, allowing services to be quickly provisioned on demand. As a network-wide upgrade is not required during service deployment, existing investments on the live network are fully protected.

5) Fast deployment for new businesses or services: SRv6 policy only needs to be configured on the ingress node and egress node, shortening the deployment time and improving deployment efficiency.

5.1.3 SR-MPLS vs SRv6

SR currently involves two data planes: MPLS and IPv6 [i.10].

When SR is applied to the MPLS data plane, it is called SR-MPLS and uses MPLS labels as SIDs.

When SR is applied to the IPv6 data plane, it is called SRv6 and uses IPv6 addresses as SIDs.

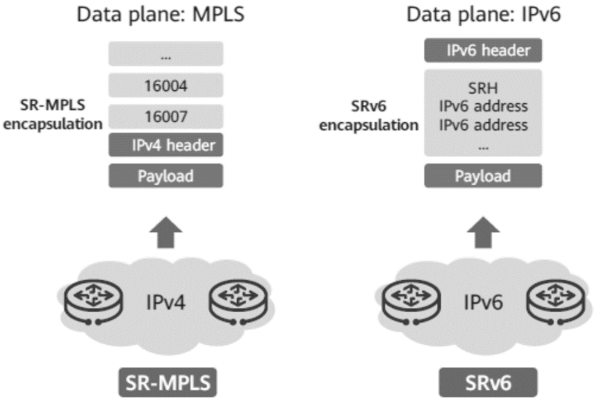


Figure 6: SR-MPLS and SRv6 comparison

Although SR-MPLS has good path programmability, it is unsuitable for services that need to carry metadata, such as Service Function Chaining (SFC) as the extensity of MPLS encapsulation is limited. In addition, the mode in which MPLS needs to add labels to the IP packet header eliminates the universality of IP technology in packets [i.3]. In this case, network devices need to support MPLS label forwarding hop by hop, raising the requirements on network devices to some extent. As such, MPLS is regarded as a dedicated technology for operators' backbone networks. It is generally not deployed in data centers; instead, it is limited to only operators' backbone networks or new metro networks.

5.2 Introduce to SRv6 based SFC

5.2.1 SRv6 aware mode

The Service Function Chaining (SFC) technology logically connects services on network devices to provide an ordered service set for the application layer. By adding Service Function Path (SFP) information to original packets, SFC enables packets to pass through Service Functions (SFs) along a specified path.

Figure 7 below shows the SRv6 aware mode development:

- SC node: Completes the definition of service function chaining and classification policy. The traffic sent from the SC node carries the SRH header and the packet encapsulation format is the same as that of normal SRv6 policy.
- SF nodes: A network device that supports some service function.
- SFF nodes: If SF supports SRv6, SFF performs SRv6 forwarding.
- End node: Identify the End. DT tag, strip the SRH header, and forward the packets according to the IP forwarding table.

In the SRv6 aware mode, the SF nodes can support SRv6. SF can connect directly to the SFF node and publish its own SIDs. The corresponding SIDs for service functions are called service SIDs.

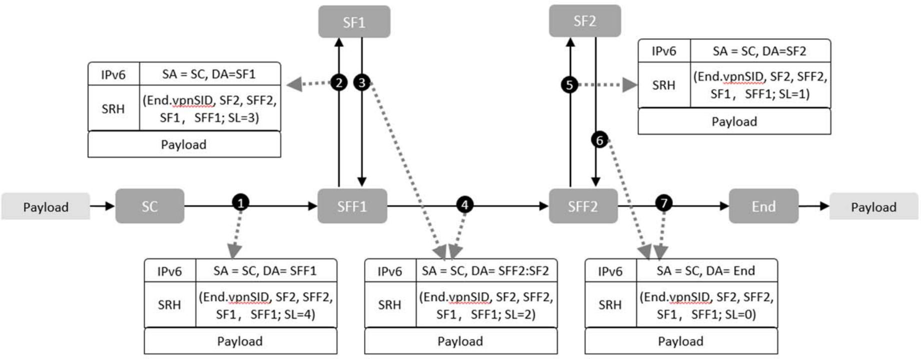

SF1
SF2
IPv6
SA = SC, DA=SF1
SRH
(End.vpnSID, SF2, SFF2, SF1， SFF1; SL=3)
Payload
IPv6
SA = SC, DA=SF2
SRH
(End.vpnSID, SF2, SFF2, SF1， SFF1; SL=1)
Payload
2
3
5
6
Payload
SC
1
SFF1
4
SFF2
7
End
Payload
IPv6
SA = SC, DA= SFF1
SRH
(End.vpnSID, SF2, SFF2, SF1， SFF1; SL=4)
Payload
IPv6
SA = SC, DA= SFF2:SF2
SRH
(End.vpnSID, SF2, SFF2, SF1， SFF1; SL=2)
Payload
IPv6
SA = SC, DA= End
SRH
(End.vpnSID, SF2, SFF2, SF1， SFF1; SL=0)
Payload

Figure 7: SRv6-based SFC development mode: SRv6 aware mode (SF support SRv6)

The data forwarding process under SRv6-aware mode is as follows [i.4]:

1) The SC node classifies the received data packets based on classification policy and redirects the classified traffic to the SRv6 Policy. Based on the SRv6 Policy, the SC node encapsulates the packets into the SRv6 mode. The destination address is the SFF1 address. In the SRH, the IPv6 addresses are: SFF1 address, SF1 address, SFF2 address, SF2 address, End.vpnSID.

2) After the SFF1 receives the packet, it changes the destination address of the packet to SF1 and forwards the packet to SF1.

3) After the SF1 receives and processes the packet, SF1 sends it to SFF1 and changes the distention address to SFF2.

4) After the SFF2 and SF2 receive the packet, it performs operations similar to SFF1 and SF1.

5) When the packet arrives at the End node, it finds that its End SID and the destination address of the packet are the same. So, the End node performs the instruction related to the End SID. For example, decapsulating the packet into the original IPv4 packet, then forwarding it to the corresponding IPv4 VPN or public network based on IPv4 rules specified by the End node.

5.2.2 SRv6 unaware mode

Currently, many SF products do not support SRv6 forwarding, so the SRv6-unaware mode is needed. Under this mode, a SRv6 proxy needs to be deployed between the SF node and SFF node to process and forward SRv6 packets. The SRv6 proxy forwards packets from the SRv6 network to SRv6 Unaware SF, and then encapsulates the packets processed by SRv6 Unaware SF into SRv6 packets.

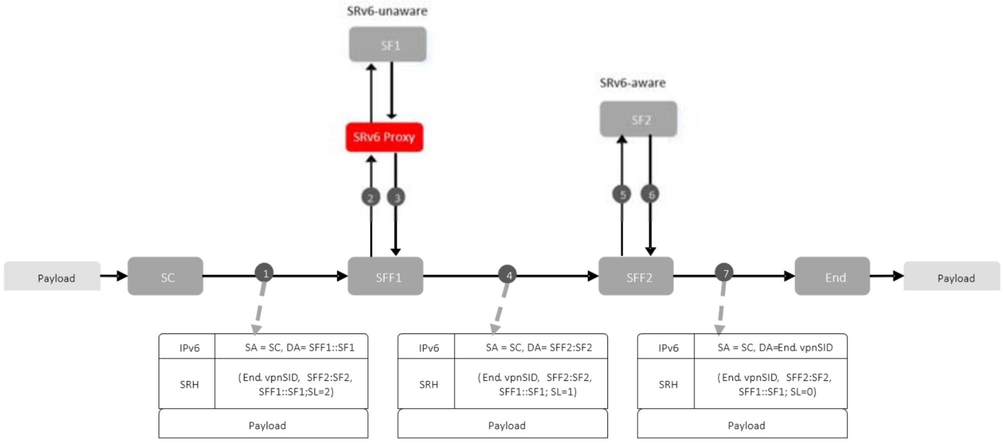


Figure 8: SRv6-based SFC development mode: SRv6 unaware mode

- **SC node:** Defines the service function chaining and traffic classification policies. The packet sent from the SC node carries the Segment Routing Header (SRH) and is encapsulated in the same format as a common SRv6 Policy packet.
- **SFF node:** SFF performs SRv6 forwarding.
- **SRv6 Proxy:** For an SF that does not support SRv6, A SRv6 Proxy is needed, which strips the SRH header and forwards the original packet to the SF node. After receiving the processed message from SF, the SRv6 Proxy re-encapsulates the SRH.
- **End node:** Identifies the End.DT tag strips the SRH and forwards the packets according to the IP forwarding table.

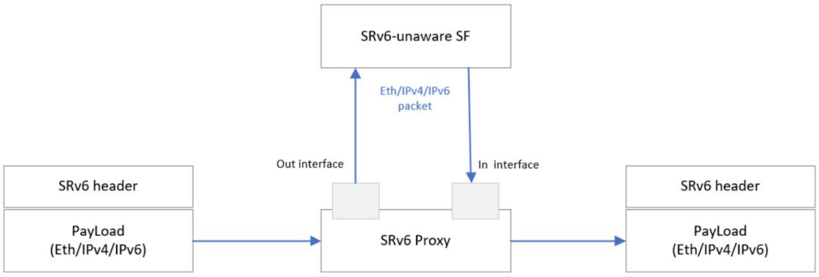


Figure 9: the SRv6 Proxy

The SRv6 proxy has multiple service SIDs according to different proxy types. The Service SIDs defined in the IETF SPRING Working Group are as follows [i.4]:

- End.AS: static Proxy SID. Published by the SRv6 Proxy. The function of End. AS is to strip SRv6 packet headers and send original packets to SF through the corresponding interface or the interface corresponding to the VLAN ID of the virtual interface. When the packets carrying the VLAN ID are returned from SF to the proxy, the SRv6 proxy encapsulates the cached SRv6 packet headers according to the VLAN ID. Keep forwarding.
- End.AD: dynamic Proxy SID. Based on the static proxy, the ability of dynamic learning is added.

5.3 Benefits of SRv6-based SFC

IETF RFC 7665 [i.1] proposes the Service Function Chaining (SFC) architecture, and IETF RFC 8300 [i.5] proposes the Network Service Header (NSH) as the encapsulation to implement the SFC architecture. NSH contains service chaining paths and Metadata, which can be shared among different services. SRv6 SFC has some advantages:

1) SRv6 is one of the primary protocols for most types of infrastructure (Metro Network, Backbone Network, Data Center). It does not need a gateway for conversion to the SFC farm. Hence, it is the only way to organize smooth end-to-end services that includes SFC on the traffic path.

2) Network devices, Virtual Network Function (VNF), and host operating systems have limited support for NSH. But most hardware switches do not support network services. Implementing the functionality designed by NSH depends on the ability of the VNF in the service chain to manipulate the paths and the meta-information, but many VNFs do not support this.

3) NSH needs to maintain the state of all service devices in each service chain, which greatly limits scalability.

SR supports explicit programming of the data packet forwarding path in the head node, which naturally supports SFC. Moreover, SR does not need to maintain the flow by flow forwarding state in nodes of the network, which makes SR's service deployment much simpler.

The SFC based on SR only needs to issue the SFC policy to the head node, and it does not need to configure all the network nodes. This also reduces the difficulty of SFC deployment, especially in the control plane.

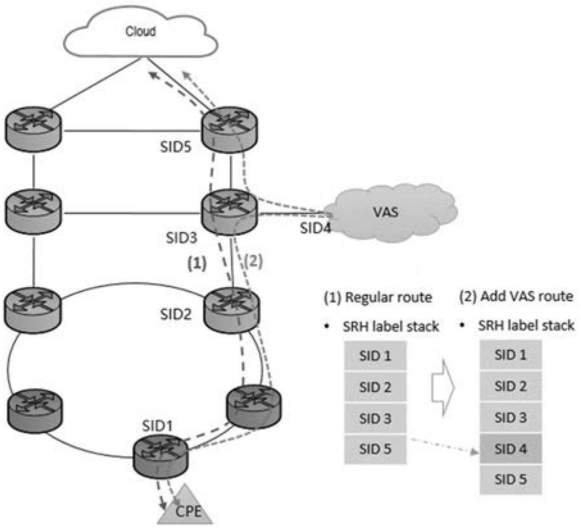


Figure 10: Example of SFC path management using SRv6

As shown in Figure 10, before subscribing to the VAS service, the label stack of SRH is 1-2-3-5. After subscribing to the VAS service, only the related SID of VAS needs to be inserted into the label stack. That is the SID 4 in the dark box.

SRv6-based SFC enables operators to have the ability to schedule services in large networks. NSH is more suitable for a single domain. SRv6 SFC has obvious advantages in cross-domain networks. At present, the direction of product R & D is to rely on the operator's special line to complete the scheduling of security services in the large network. The adoption of NSH is more complex, and SRv6 SFC is naturally adapted to the product above.

6 SRv6-aware network firewall

6.1 Introduction

SRv6-aware network firewall can process SRv6 information in data packets. They can be integrated into the SRv6 service chain without any agent [i.10]. This simplifies the configuration and management of the SFC infrastructure [i.11]. At the same time, this has a positive impact on the performance of SFC-enabled nodes, as complex classification procedures are no longer required.

In addition, it allows the implementation of advanced SFC functions because the SRH information is retained when the network firewall processes the data packets. This clause introduces the design and implementation of SRv6-aware network functions.

The design considerations for developing SRv6-aware NFs are defined in clause 6.2. The design, implementation, and performance of the SRv6 aware firewall (named SERA) are described in clause 6.3.

6.2 Designing SRv6-aware NFs

SRv6-aware NFs can handle SRv6 encapsulated traffic, which means they can handle raw data packets, even though they have been encapsulated using SRv6. They can identify the order of the NFs that have already processed the packet, and the order of the subsequent NFs that still need to process the packet. This opens up the possibility of advanced service chain operations, such as branching and looping.

When a node running SRv6 aware NF receives a SRv6 encapsulated data packet whose destination address matches the segment assigned to SRv6 aware NF, the complete SRv6 data packet will be processed to the VNF. VNF can act on the outer IPv6 header, SRH, and inner data packets of data packets. It can perform some operation or even discard it.

Finally, it uses the next SID in the SID list to update the packet destination address and forward the packet accordingly. In this clause, a firewall is used as an example to define the design considerations for SRv6-aware NF. However, these considerations have a universal value because they can be applied to many types of network functions that need to be deployed on SRv6-based SFC environments (such as DPI, IDS). The firewall essentially works according to a set of rules to accept or discard received packets. Each rule consists of a condition and an action. The condition is based on the attributes of the received data packet. Once the data packet satisfies the conditions expressed by the rule conditions, relevant operations are performed on the data packet.

It is assumed that the SRv6-aware firewall should support two working modes: basic mode and advanced mode. In the basic mode, the SRv6 aware firewall should work as a traditional firewall but does not require a SRv6 proxy. In particular, SRv6-aware firewalls should be able to use the same set of rules defined for traditional firewalls and apply them directly to SRv6 encapsulated packets carrying SRH information and additional IPv6 header.

To give a specific example, if the existing rules include the condition of the source IPv6 address, and the original IPv6 packet has been encapsulated with IPv6-in-IPv6, it does not make sense to treat the external IPv6 source address of the received packet as the condition and the source address of the original data packet should be checked. The use case scenario is to virtualize old firewalls and execute them in a server on the SFC infrastructure, without changing the old rules, and without the need for SRv6 proxy functions.

In advanced mode, SRv6-aware firewalls should support rules with extended conditions. These rules can not only explicitly include attributes from the original packet, but also include attributes from SRH and external packets. In particular, SRv6-aware firewalls can utilize SRv6 SID parameters, TLVs, or tags. It can also apply differentiation based on the active SRv6 SID (that is, apply different rule sets for different SIDs). As for actions, in advanced mode, SRv6-aware firewalls should be able to support SRv6-specific actions. For example, a SRv6-specific operation may be able to skip the next SID in the segment list, so that when certain conditions of the packet are met, the "branch" can be operated instead of the usual linear exploration of the VNF chain. A use case for this function scenario is a service chain that includes a firewall, then an Intrusion Detection System (IDS), and allows IDS to be skipped to obtain a subset of traffic that matches certain conditions.

Another requirement is that SRv6-aware firewall applications should be able to choose what to do based on the information contained in the SID. This is consistent with the SRv6 network programming method, which minimizes the state information maintained in the node and stores explicit state information in the data packet. The use case scenario, in this case, is that instead of reconfiguring some firewall rules in a specific firewall running at the core of the SFC infrastructure, the same result can be obtained by changing the SID in the injected SID list. The edge node sends the data to the data. Bag. The biggest advantage is that only the edge nodes need to be reconfigured.

6.3 SERA (SEgment Routing Aware firewall)

6.3.1 SERA basic mode

SEgment Routing Aware firewall (SERA) is an advanced SRv6-aware firewall, capable of taking stateless actions programmed in the SRH. It supports both basic and advanced modes of SRv6-aware firewalls. The basic mode, the SERA firewall solution avoids the need to (re) classify packets in the intermediate SFC nodes that host the SRv6-aware firewall. The advanced mode supports new firewall actions that can operate on the SRH segment list, allowing to make branches in the VNF chain. To the best of our knowledge, the SERA firewall is one of excellent SRv6- aware applications.

In the basic mode, SERA applies the firewall processing to the original packets of SRv6 traffic. The proposed packet processing architecture is shown below in Figure 11. Each received packet goes through a SRv6 pre-processor that splits traffic into SRv6 and non-SRv6 traffic.

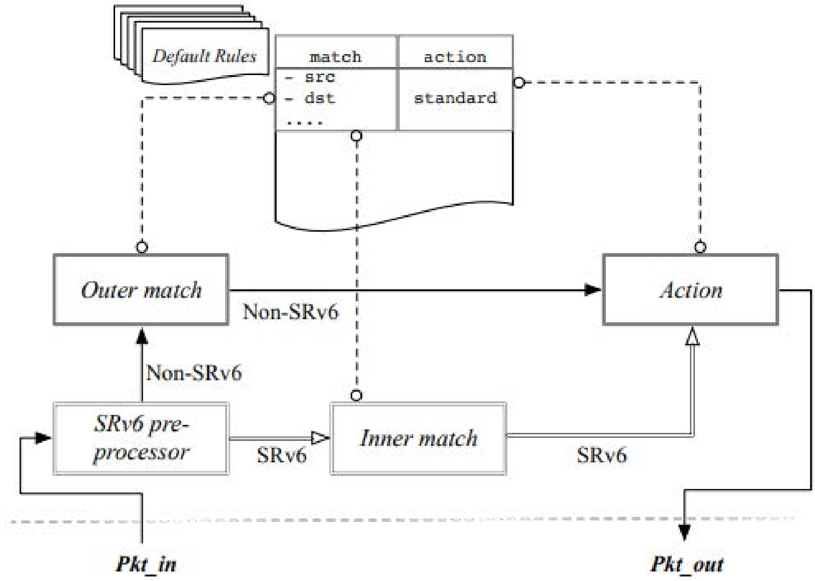


Figure 11: SERA Architecture: basic mode

Non SRv6 traffic does not require any special processing and is processed as in a SRv6-unaware firewall, as represented with the solid-line path in Figure 11. SRv6 traffic follows a different path through the firewall, represented by a double-line path in the figure. In this path, the firewall, using the Inner match module, evaluates the rules on the inner packet and considers the impact of the SRv6 encapsulation. The Inner match module is responsible for getting the original source and destination information from SRv6 packets and comparing them to the defined rules. Once a packet matches one of the rules, the Action module applies the associated action (e.g. ACCEPT, DROP) on that packet.

6.3.2 SERA advanced mode

In the advanced mode, SERA extends the iptables capabilities with new matching capabilities and new SRv6-specific actions. It introduces a new type of iptables rules (SERA rules) that have extended conditions on the attributes of the outer packet, inner packet, and the SRH header.

The architecture of the advanced mode (in Figure 12) is defined incrementally concerning the basic mode in Figure 11 by adding the SRH match module and replacing the Action block with the Extended Action block. Since the matching could be performed on both the original and the outer packet headers, the SRv6 traffic follows a more complex path, as shown in Figure 12. Unlike in the basic mode SERA, all received packets are first processed by the Outer match block, which applies parts of the extended rules on the outer packet. The SRv6 pre-processor does the same job as in the basic mode SERA by splitting traffic into non-SRv6 and SRv6 traffic. Non-SRv6 traffic goes directly to the Action module, while SRv6 traffic is directed to the Inner match module. The Inner match module works as in the basic mode, but the rules that drive its behaviour are written differently.

For example, with an extended rule, it is possible to match on the outer source and destination IPv6 addresses (denoted as src, dst) and on the original ones (denoted as inner-src, inner-dst). The Inner match block takes care of the matching of the inner source and destination (the ones of the original packet). The SRH match block is concerned with the matching between the SRH extension part of the rules and the SRH of received SRv6 packets. Finally, each packet (SRv6 or non-SR) that satisfies the matching condition of a rule goes to the Extended Action module. It extends the Action module present in the architecture of the Basic mode by allowing the introduction of SRv6-specific actions and the standard ones. A SRv6-specific action is an advanced action that can be applied to SRv6-encapsulated packets. It may modify or process SRv6-encapsulated packets based on SRH information.

Listed below are some examples of SRv6-specific actions, but the set of these actions can be extended to cover more complex SFC use-cases.

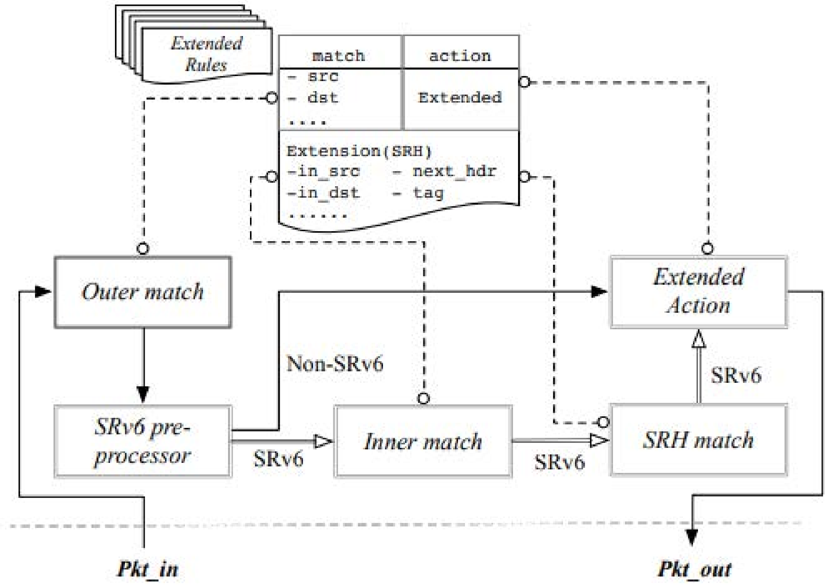


Figure 12: SERA Architecture: advanced mode

- seg6-go-next: Similar to the SRv6 End behavior from the SRv6 network programming model. It sends packets towards the next SID from SRH. The seg6-go-next serves as an ACCEPT action for SRv6 encapsulated packets.
- seg6-skip-next: Instructs the SERA firewall to skip the next SID in the SRH.
- seg6-go-last: Instructs the SERA firewall to skip the remaining part of the segment list and process the last segment.
- seg6-eval-args: Generic action to support programming actions into the SRH content.

7 Development practice of SRv6 based SFC in the operator network

7.1 Experimental development practice design

This clause describes how to implement an SFC capability based on SRv6 in an operator's network. The practical deployment scheme is designed considering the following factors, as shown in Figure 13:

- The experimental network topology design: the experimental network is designed to contain two different domains, a SRv6 domain in which all key network devices support SRv6 forwarding and a non-SRv6 domain in which some devices do not support SRv6 forwarding, but all the key network devices in those two domains support IPv6 forwarding. In this way, the ability of the SFC to schedule computing power resources in multiple different domains can be verified, and the configuration and management of the SFC in different domains can be researched.

- The network topology shown in Figure 13 is only an abstract schematic diagram. Only key network devices are listed. In the SRv6 domain, devices R11, R12, R21, and R22 are key devices that support SRv6 forwarding, and device R21 connects to the computing resource pool. R12 is also the connection point between the experimental test network and the existing operator's network. In a non-SRv6 domain, device R31 is connected to an SFF device that supports SRv6 forwarding, and the SFF device is connected to another computing resource pool.

- As for the design of the computing power resource pool, two computing power resource pools capable of providing service functions were deployed, which were respectively connected to the two different network domains mentioned above, to facilitate the experimental test of scheduling multiple service functions.

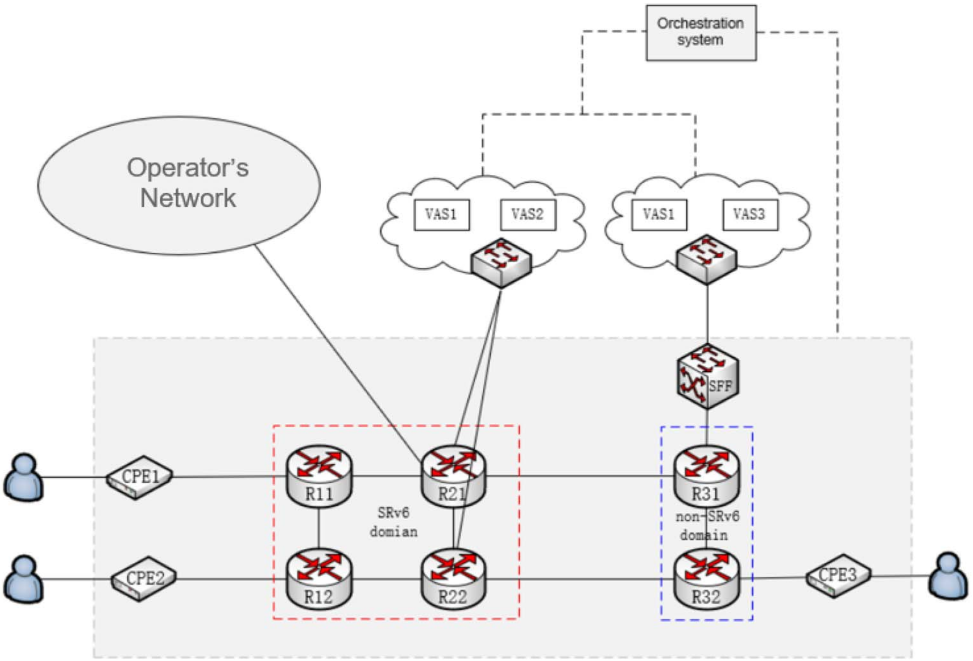


Figure 13: Schematic of practical deployment solution in an operator's network

7.2 Orchestration System

The orchestration system is connected with the network controller and Cloud control. The orchestration system obtains network topology information through the network controller and computing resource pool information through the Cloud controller, including value-added services that the computing resource pool can provide.

The orchestration system is mainly focused on service orchestrating based on network resources and computing resources as mentioned above. According to users' service requirements, the orchestration system selects the suitable computing resources to provide services and decides the service path. The orchestration system stores abstract network information, service information, and computing resources. It may also include shareable service information, historical service request information, and so on. All information used for service orchestrating is stored in a database.

In addition, this system can select computing resources and decide the service path. In this case, the service path is the result of the orchestrating service functions according to abstract topology, consisting of a set of ordered nodes that the service passes through, including the access nodes, the necessary intermediate network nodes, and the nodes connected to the computing resource pools.

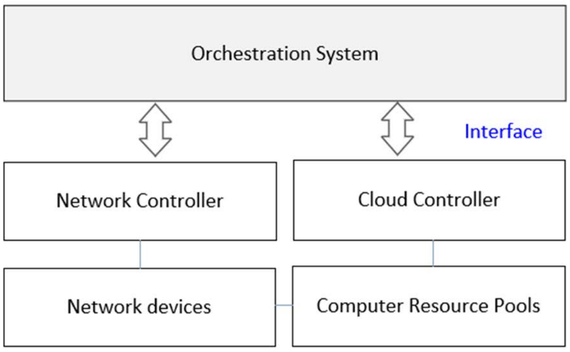


Figure 14: The Orchestration System

The network controller collects the network topology information and reports the topology to the orchestration system. It can also calculate the detailed forwarding path based on network topology according to the service path and then generates the configurations corresponding to the forwarding path and sends them to related SRv6 nodes.

The Cloud controller manages the computing resource pool and collects service resource information that needs to report to the orchestrator system. The service resource information may include service type, service state, and so on. The Cloud controller configures the services resource according to the orchestration's instructions.

7.3 Reference Points

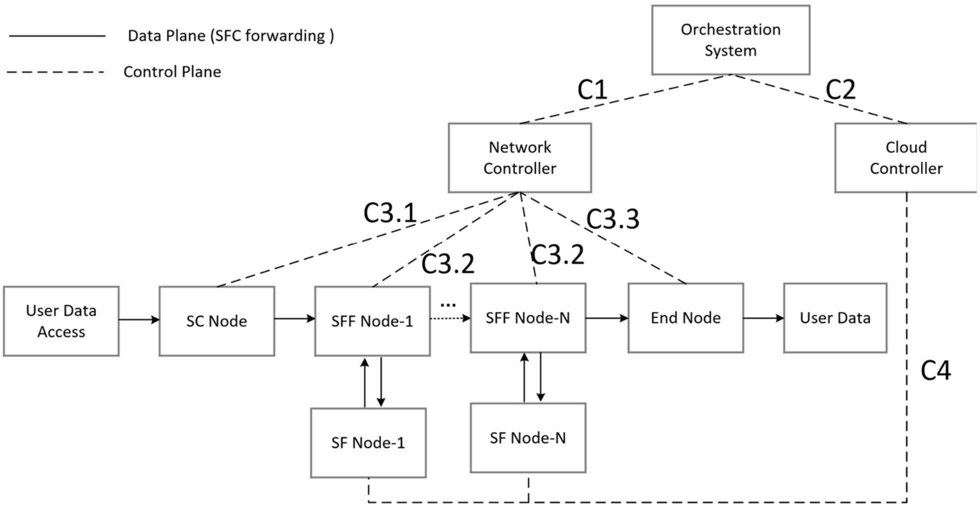


Figure 15: Reference points of service function chaining based on SRv6

- Reference point C1: The Orchestration System interacts with the network controller via reference point C1. The Orchestration System gets topology information from the network controller and sends service path information to the network controller after finishing the service function chaining orchestration.

- Reference Point C2: The Orchestration System interacts with the Cloud controller via reference point C2. The Orchestration System gets service function information from the Cloud controller and sends the service configuration requirement to the Cloud controller after finishing the service function chaining orchestration.
- Reference Point C3: The network controller interacts with network nodes:
    - Reference Point C3.1: The network controller sends the SC configuration to the SC node via reference point C3.1. The configuration is about the segment list of SRv6 and traffic classifier policy.
    - Reference Point C3.2: The network controller sends the SFF forwarding policy to the SFF node via reference point C3.2. If the related SF node does not support SRv6, the network controller also needs to send the SRv6 proxy configuration to the SFF node.
    - Reference Point C3.3: The network controller sends the configuration of the end of service function chaining to the end node via reference point C3.3.
- Reference Point C4: The Cloud controller configures the service function via C4 and also gets the service state via C4.

7.4 Service development

The following describes the information flow of deploying service:

- Step 1: The user sends a service requirement information message to the Orchestrator System through a portal, website, or other methods.
- Step 2: After receiving the requirement message, the orchestrator selects computing resources according to strategy and abstract topology, which are stored in the database, and then sends a service path information message to the network controller. In the meantime, the Orchestrator System sends selected computing resource information to the Cloud controller.
- Step 3: The network controller calculates the responding forward path based on global network topology, and then by querying the routing table, the network controller obtains the forwarding path information. Then network controller generates the SRv6 configuration files and sends them to SRv6 devices.
- Step 4: The Cloud controller generates Cloud-related configurations and sends them to the responding Cloud resource pool.
- Step 5: The SRv6 devices respond to the network controller that the configuration is ready.
- Step 6: The Cloud resource pool configures the required service resources and sends the responses to the Cloud controller to acknowledge that the computing resource configuration is ready.
- Step 7: The network controller responses are sent to the orchestrator to acknowledge that the forwarding path is ready.
- Step 8: The Cloud controller responses are sent to the orchestrator to acknowledge that the computing resource is ready.
- Step 9: The Orchestrator responses are sent to the user to acknowledge that the service has been deployed.

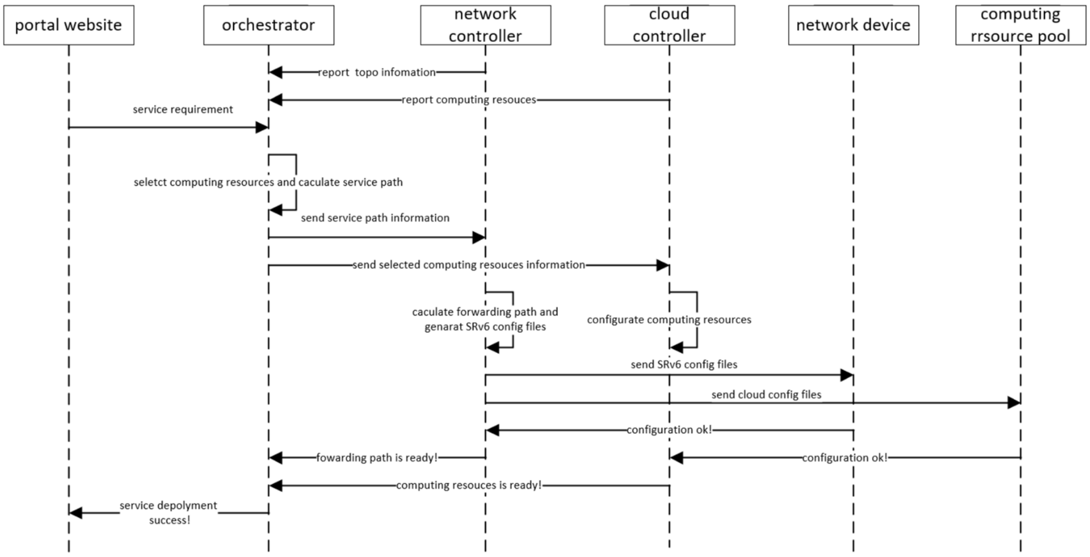

portal website
orchestrator
network controller
cloud controller
network device
computing rrsource pool
report topo infomation
report computing resouces
service requirement
seletct computing resources and caculate service path
send service path information
send selected computing resouces information
caculate forwarding path and genarat SRv6 config files
configurate computing resources
send SRv6 config files
send cloud config files
configuration ok!
fowarding path is ready!
configuration ok!
computing resouces is ready!
service depolyment success!

Figure 16: The information flow of the deploying service

8 Experimental tests

8.1 Test targets

In this test case, two scenarios of SFC are described. The first one is about end-to-end users who go through a single VAS service, and the second is about end-to-end users who go through two different service functions in the different Cloud.

Since most enterprises are still using IPv4, the IPv4 data flow is simulated in the test case to make it more in line with current network. When the data flow enters the IPv6 domain, it will be encapsulated with IPv6 packet header, that is, it will become an IPv6 data flow. When the data flow is forwarded from IPv6 domain to the client, V6 header will be peeled off to expose the original IPv4 message, after that the flow will be forwarded following the rules of V4.

8.2 Test cases

8.2.1 SFC scheduling of single-service in single-Cloud (SRv6-aware mode)

Test Scheme:

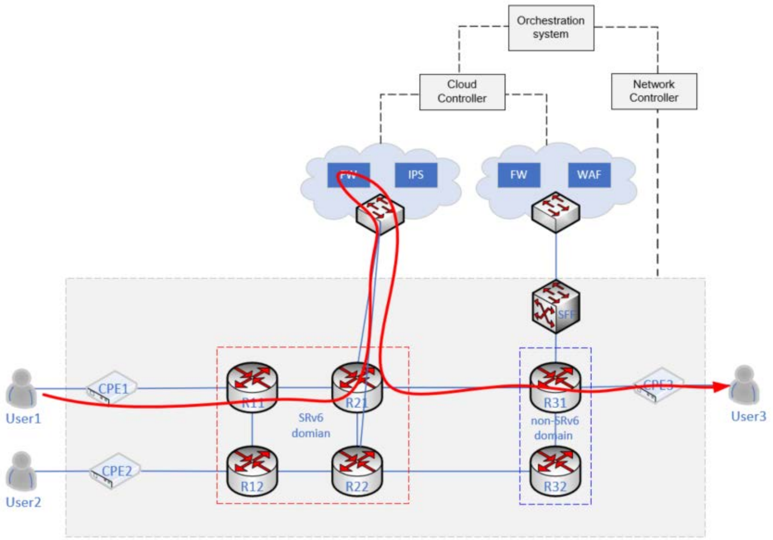


Figure 17: Test scheme for SRv6-aware mode

Test procedure:

1) Use an orchestration system to create a VPN from CPE1 to CPE3.

2) Create two IP flows from User1 to User3, whose source IPs are 192.168.152.5 and 192.168.152.6, the Test Result 1 is expected.

3) Check if the path is the same as the expected Test Result 2.

4) In the FW service function of the left Cloud, a policy that can deny the IP flow whose source IP is 192.168.152.6 is created.

5) Use the Orchestration System to add the FW service of the left Cloud to the VPN path, then create two IP flows the same as those created in Step 2, and the Test Result 3 is expected.

6) Check if the path is the same as the expected Test Result 4.

Expected Test Results

1) The two flows are 100 % received at User3.

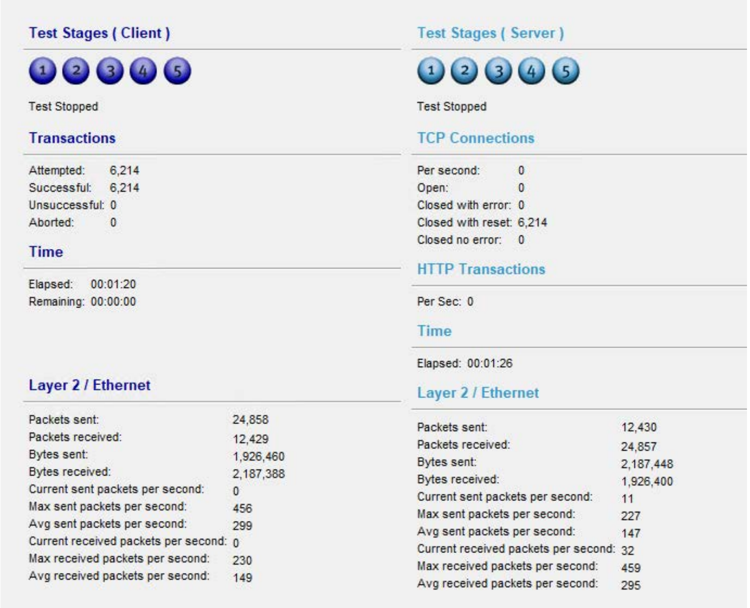
2) The path is as follow: CPE1->R11->R21->R31->CPE3.

3) The flows are 50 % received at User3.

4) The path is as follows: CPE1->R11->R21->FW->R21->R31->CPE3, as the red line showing in the Test Scheme.

Test Report:

1) Following Steps 1-3 in the test procedure, the test result is as shown below in Figure 18 (1) shows the 100 % flows, and picture (2) shows the path.



Test Stages (Client)

1 2 3 4 5

Test Stopped

Transactions

Attempted:	6,214
Successful:	6,214
Unsuccessful:	0
Aborted:	0

Time

Elapsed:	00:01:20
Remaining:	00:00:00

Layer 2 / Ethernet

Packets sent:	24,858
Packets received:	12,429
Bytes sent:	1,926,460
Bytes received:	2,187,388
Current sent packets per second:	0
Max sent packets per second:	456
Avg sent packets per second:	299
Current received packets per second:	0
Max received packets per second:	230
Avg received packets per second:	149

Test Stages (Server)

1 2 3 4 5

Test Stopped

TCP Connections

Per second:	0
Open:	0
Closed with error:	0
Closed with reset:	6,214
Closed no error:	0

HTTP Transactions

Per Sec: 0

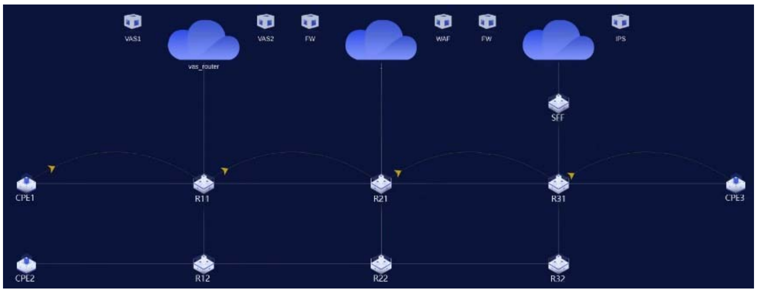
Time

Elapsed: 00:01:26

Layer 2 / Ethernet

Packets sent:	12,430
Packets received:	24,857
Bytes sent:	2,187,448
Bytes received:	1,926,400
Current sent packets per second:	11
Max sent packets per second:	227
Avg sent packets per second:	147
Current received packets per second:	32
Max received packets per second:	459
Avg received packets per second:	295

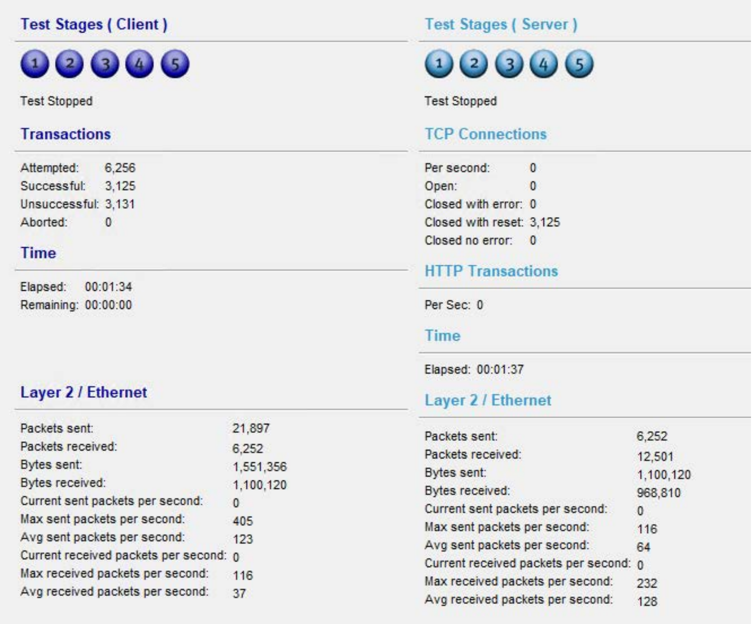
(1) shows flow statistics



(2) shows the related path

Figure 18: The test result and the test path under SRv6-aware mode

2) Following Steps 4-6 in the test procedure, the test results are shown in Figure 19 below, (1) shows the 50 % flows and (2) shows the path.



Test Stages (Client)

1 2 3 4 5

Test Stopped

Transactions

Attempted:	6,256
Successful:	3,125
Unsuccessful:	3,131
Aborted:	0

Time

Elapsed:	00:01:34
Remaining:	00:00:00

Layer 2 / Ethernet

Packets sent:	21,897
Packets received:	6,252
Bytes sent:	1,551,356
Bytes received:	1,100,120
Current sent packets per second:	0
Max sent packets per second:	405
Avg sent packets per second:	123
Current received packets per second:	0
Max received packets per second:	116
Avg received packets per second:	37

Test Stages (Server)

1 2 3 4 5

Test Stopped

TCP Connections

Per second:	0
Open:	0
Closed with error:	0
Closed with reset:	3,125
Closed no error:	0

HTTP Transactions

Per Sec: 0

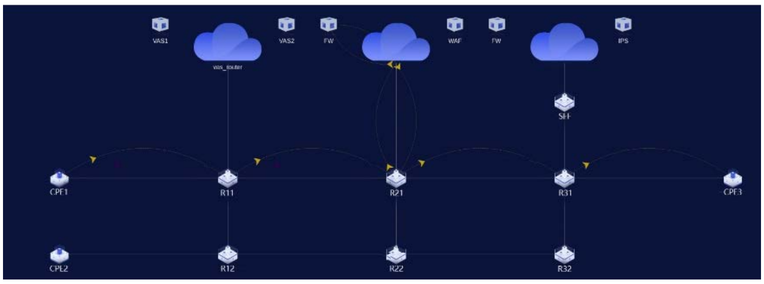
Time

Elapsed: 00:01:37

Layer 2 / Ethernet

Packets sent:	6,252
Packets received:	12,501
Bytes sent:	1,100,120
Bytes received:	968,810
Current sent packets per second:	0
Max sent packets per second:	116
Avg sent packets per second:	64
Current received packets per second:	0
Max received packets per second:	232
Avg received packets per second:	128

(1) shows flow statistics



(2) shows the related path

Figure 19: Test result and test path under SRv6 aware mode when the VAS was added

8.2.2 SFC scheduling of multi-service in multi-Cloud (Hybrid mode)

Test Scheme:

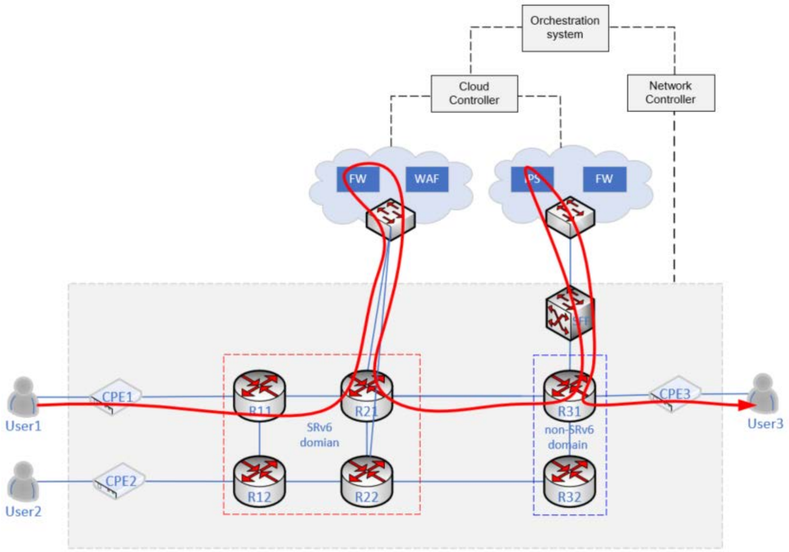


Figure 20: The Test scheme for hybrid mode

Test procedure:

1) To support the SFC, an extra SRv6 device was added so that the SFF could guide the flow into the Cloud, such as the scheme above. Use an orchestration system to create one VPN from CPE1 to CPE3 across the SRv6 domain and non-SRv6 domain.

2) When two IP flows from User1 to User3, source IPs 192.168.152.5 and 192.168.152.6, and the second flow contains all the SQL instructions with source IPs of 192.168.152.6 are created, the Test Result 1 is expected.

3) Check if the path is the same as Test Results 2.

4) In the IPS (Intrusion Prevention System) service of the right Cloud, a policy that can deny the SQL intrusion is created.

5) Use the Orchestration System to add the FW service of the left Cloud and the IPS service of the right Cloud in the VPN path, two IP flows the same as those that can be created in Step 2 are created, and the Test Result 3 is expected.

6) Check if the path is the same as the Test Results 4.

Expected Test Results:

1) The two flows are 100 % received at User3.

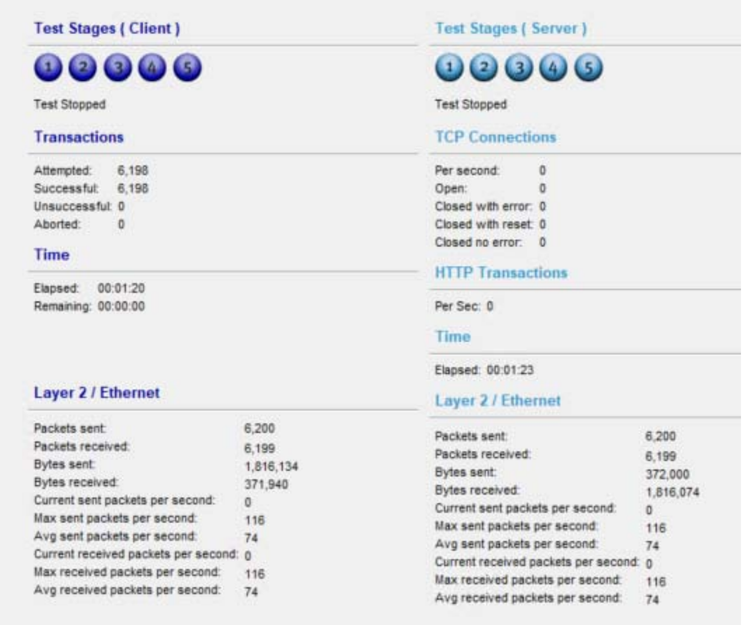
2) The path is as follow: CPE1-> R11->R21->R31->CPE3.

3) The flows are 50 % received at User3.

4) The path is as follow: CPE1-> R11->R21-> FW->R21->R31->IPS->R31 ->CPE3, as the red line showing in the Test Scheme.

Test Report:

1) Following Steps 1-3 in the test procedure, the test results will be as shown below in Figure 21, (1) shows the 100 % flows and the (2) shows the path.



Test Stages (Client)

Test Stopped

Transactions

Attempted: 6,198
Successful: 6,198
Unsuccessful: 0
Aborted: 0

Time

Elapsed: 00:01:20
Remaining: 00:00:00

Layer 2 / Ethernet

Packets sent:	6,200
Packets received:	6,199
Bytes sent:	1,816,134
Bytes received:	371,940
Current sent packets per second:	0
Max sent packets per second:	116
Avg sent packets per second:	74
Current received packets per second:	0
Max received packets per second:	116
Avg received packets per second:	74

Test Stages (Server)

Test Stopped

TCP Connections

Per second: 0
Open: 0
Closed with error: 0
Closed with reset: 0
Closed no error: 0

HTTP Transactions

Per Sec: 0

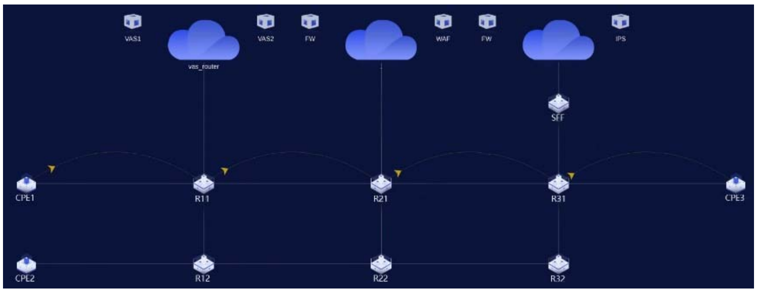
Time

Elapsed: 00:01:23

Layer 2 / Ethernet

Packets sent:	6,200
Packets received:	6,199
Bytes sent:	372,000
Bytes received:	1,816,074
Current sent packets per second:	0
Max sent packets per second:	116
Avg sent packets per second:	74
Current received packets per second:	0
Max received packets per second:	116
Avg received packets per second:	74

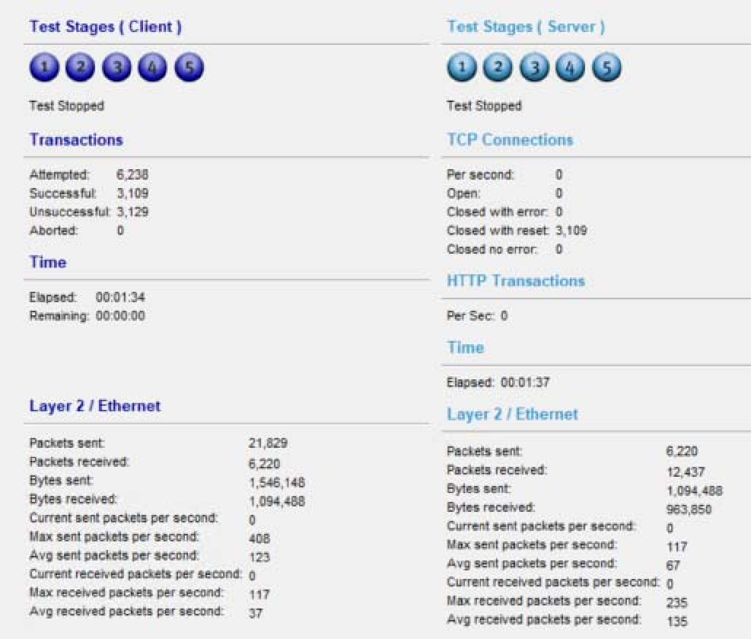
(1) shows flow statistics



(2) shows the related path

Figure 21: The test result and test path under the hybrid mode

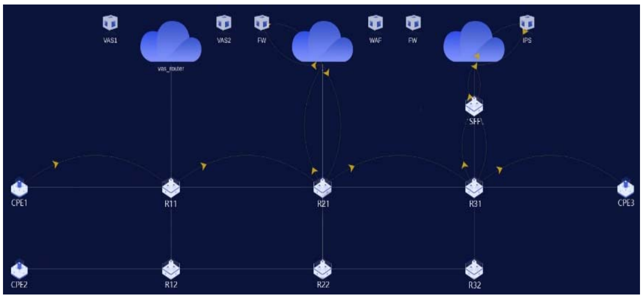
2) Following Steps 4-6 in the test procedure, test results 3 and 4 are shown in Figure 22 below, (1) shows the 50 % flows, and (2) shows the path.



```
Test Stages ( Client )
1 2 3 4 5
Test Stopped

Transactions
Attempted:     6,238
Successful:    3,109
Unsuccessful:  3,129
Aborted:       0

Time
Elapsed:    00:01:34
Remaining:  00:00:00

Layer 2 / Ethernet
Packets sent:                          21,829
Packets received:                      6,220
Bytes sent:                            1,546,148
Bytes received:                        1,094,488
Current sent packets per second:       0
Max sent packets per second:           408
Avg sent packets per second:           123
Current received packets per second:   0
Max received packets per second:       117
Avg received packets per second:       37
```

```
Test Stages ( Server )
1 2 3 4 5
Test Stopped

TCP Connections
Per second:          0
Open:                0
Closed with error:   0
Closed with reset:   3,109
Closed no error:     0

HTTP Transactions
Per Sec:  0

Time
Elapsed:  00:01:37

Layer 2 / Ethernet
Packets sent:                          6,220
Packets received:                      12,437
Bytes sent:                            1,094,488
Bytes received:                        963,850
Current sent packets per second:       0
Max sent packets per second:           117
Avg sent packets per second:           67
Current received packets per second:   0
Max received packets per second:       235
Avg received packets per second:       135
```

(1) the flow statistics



(2) the related path

Figure 22: The test result and test path under the hybrid mode when the VASes were added

9 Conclusion and future work

SFC can flexibly schedule service resources on-demand, and it is a key technology for the integration of computing resources and network resources. Using SRv6 to realize the SFC function is a very good technical method as SR supports explicit programming of data packet forwarding path in the head node, which naturally supports SFC.

The present document is a deployment practice for implementing SFC based on SRv6. Through experimental tests, the feasibility of realizing SFC by SRv6 is verified:

- Using SRv6 technology, end-to-end SFC path management can be realized.
- Using the configuration SRv6 technology to achieve SFC is simple, regardless of the SRv6-aware mode or the SRv6-unaware mode:
  a) For the SRv6-aware mode, making the SFC path according to the demand is only necessary, and then set the policy for the corresponding SC node.
  b) For the SRv6-unaware mode, the SRv6 proxy needs to be configured to forward packets to SF, but as more and more service function devices support SRv6, this problem will not exist in the future.
- Using SRv6 technology to achieve SFC can flexibly schedule computing resources and realize efficient utilization of resources.

The present document is only the beginning of implementing SFC based on SRv6, and further research work is required to improve the service. In the future, SRv6-based SFC will continue to be deployed and tested, including the impact of network performance problems on the orchestration system, service chain function management, service chain function fault location, and recovery. In addition, the use of SRv6 to provide programmable services, like SID as a Service (SIDaaS), will also be researched and deployed.

History

Document history		
V1.1.1	July 2022	Publication