7705 Service Aggregation Router Gen 2

Release 26.3.R1

Router Configuration Guide

3HE 29568 AAAA TQZZA 01
Edition: 01
March 2026

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Table of contents

List of tables

List of figures

1 Getting started

1.1 About this guide

This guide describes logical IP routing interfaces, virtual routers, and IP and MAC-based filtering, and presents configuration and implementation examples.

This guide is organized into functional chapters and provides concepts and descriptions of the implementation flow, as well as Command Line Interface (CLI) syntax and command usage.

Unless otherwise indicated, the topics and commands described in this guide apply only to the 7705 SAR Gen 2 platforms listed in Platforms and terminology.

Command outputs shown in this guide are examples only; actual displays may differ depending on supported functionality and user configuration.

Note: Unless otherwise indicated, CLI commands, contexts, and configuration examples in this guide apply for both the classic CLI and the MD-CLI.

The SR OS CLI trees and command descriptions can be found in the following guides:

- *7705 SAR Gen 2 Classic CLI Command Reference Guide*
- *7705 SAR Gen 2 Clear, Monitor, Show, Tools CLI Command Reference Guide* (for both the MD-CLI and classic CLI)
- *7705 SAR Gen 2 MD-CLI Command Reference Guide*

Note: This guide generically covers Release 26.*x*.R*x* content and may contain some content that will be released in later maintenance loads. See the *SR OS R26.x.Rx Software Release Notes*, part number 3HE 29176 000*x* TQZZA, for information about features supported in each load of the Release 26.*x*.R*x* software. For a list of features and CLI commands that are present in SR OS but not supported on the 7705 SAR Gen 2 platforms, see "SR OS Features not Supported on SAR Gen 2" in the *SR OS R26.x.Rx Software Release Notes*.

1.2 Platforms and terminology

Note:
Unless explicitly noted otherwise, this guide uses the terminology defined in the following table to collectively designate the specified platforms.

Table 1: Platforms and terminology

Platform	Collective platform designation
7705 SAR-Hx	7705 SAR Gen 2
7705 SAR-Mx	

Platform	Collective platform designation
7705 SAR-1	

1.3 Conventions

This section describes the general conventions used in this guide.

1.3.1 Precautionary and information messages

The following information symbols are used in the documentation.

DANGER: Danger warns that the described activity or situation may result in serious personal injury or death. An electric shock hazard could exist. Before you begin work on this equipment, be aware of hazards involving electrical circuitry, be familiar with networking environments, and implement accident prevention procedures.

WARNING: Warning indicates that the described activity or situation may, or will, cause equipment damage, serious performance problems, or loss of data.

Caution: Caution indicates that the described activity or situation may reduce your component or system performance.

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.3.2 Options or substeps in procedures and sequential workflows

Options in a procedure or a sequential workflow are indicated by a bulleted list. In the following example, at step 1, the user must perform the described action. At step 2, the user must perform one of the listed options to complete the step.

Example: Options in a procedure

1. User must perform this step.
2. This step offers three options. User must perform one option to complete this step.
   - This is one option.
   - This is another option.
   - This is yet another option.

Substeps in a procedure or a sequential workflow are indicated by letters. In the following example, at step 1, the user must perform the described action. At step 2, the user must perform two substeps (a. and b.) to complete the step.

Example: Substeps in a procedure

1. User must perform this step.
2. User must perform all substeps to complete this action.
   a. This is one substep.
   b. This is another substep.

Nested substeps within a procedure or a sequential workflow are indicated by roman numerals. In the following example, at step 1, the user must perform the described action. At step 2, the user must perform two substeps (a. and b.) to complete the step. At substep b, the user must perform two additional substeps (i. and ii.) to complete the step.

Example: Nested substeps in a procedure

1. User must perform this step.
2. User must perform all substeps to complete this action.
   a. This is one substep.
   b. User must perform all nested substeps to complete this action.
      i. This is a nested substep.
      ii. This is another nested substep.

2 IP router configuration

This chapter provides information about commands required to configure basic router command options.

2.1 Configuring IP router command options

To provision services on a Nokia router, logical IP routing interfaces must be configured to associate attributes such as an IP address, port, or the system with the IP interface.

A special type of IP interface is the system interface. A system interface must have an IP address with a 32-bit subnet mask. The system interface is used as the router identifier by higher-level protocols such as OSPF and BGP, unless overwritten by an explicit router ID.

The following router features can be configured:

- Interfaces
- Autonomous systems
- Confederations
- Proxy ARP

2.1.1 Interfaces

Nokia routers use different types of interfaces for various functions. Interfaces must be configured with information such as the interface type (network and system) and address. A port is not associated with a system interface. An interface can be associated with the system (loopback address).

2.1.1.1 Network interface

A network interface (a logical IP routing interface) can be configured on one of the following entities:

- physical or logical port

2.1.1.2 Network domains

To determine which network ports (and, therefore, which network complexes) are eligible to transport traffic of individual SDPs, network-domain is provided. Network-domain information is then used for the sap-ingress queue allocation algorithm applied to VPLS SAPs. This algorithm is optimized in so that no sap-ingress queues are allocated if the specified port does not belong to the network-domain used in the specified VPLS. Also, sap-ingress queues are not allocated toward network ports (regardless of the network-domain membership) if the specified VPLS does not contain any SDPs.

SAP-ingress queue allocation considers the following:

- SHG membership of individual SDPs
- network-domain definition under SDP to restrict the topology in which the specified SDP can be set-up

The implementation supports four network-domains within any VPLS.

Network-domain configuration at the SDP level is ignored when the SDP is used for Epipe or Ipipe bindings.

Network-domain configurations are irrelevant for Layer 3 services (Layer 3 VPN and IES services). Network-domain configurations can be defined in the base routing context and associated only with network interfaces in this context. Network domains are not applicable to loopback and system interfaces.

The network-domain information is only used for ingress VPLS sap queue-allocation. It is not considered by routing during SDP setup. Therefore, if the specified SDP is routed through network interfaces that are not part of the configured network domain, the packets are still forwarded, but their QoS and queuing behavior is based on default settings. Also, the packet does not appear in SAP statistics.

There is always one network-domain with the reserved name default. The interfaces always belong to a default network-domain. It is possible to assign a specific interface to different user-defined network-domains. The loopback and system interfaces are also associated with the default network-domain at the creation. However, any attempt to associate those interfaces with any explicitly defined network-domain is blocked at the CLI level because there is no benefit for that association.

Any SDP can be assigned only to one network domain. If none is specified, the system assigns the default network-domain. This means that all SAPs in VPLS have queues reaching all fwd-complexes serving interfaces that belong to the same network-domains as the SDPs.

It is possible to assign or remove network-domain association of the interface/SDP without requiring deletion of the respective object.

2.1.1.3 System interface

The system interface is associated with the network entity (such as a specific router or switch), not a specific interface. The system interface is also referred to as the loopback address. The system interface is associated during the configuration of the following entities:

- the termination point of service tunnels
- the hops when configuring MPLS paths and LSPs
- the addresses on a target router for BGP and LDP peering

The system interface is used to preserve connectivity (when routing reconvergence is possible) when an interface fails or is removed. The system interface is used as the router identifier, and a system interface must have an IP address with a 32-bit subnet mask.

2.1.1.4 Unicast reverse path forwarding check

Unicast reverse path forwarding check (uRPF) helps to mitigate problems that are caused by the introduction of malformed or forged (spoofed) IP source addresses into a network by discarding IP packets that lack a verifiable IP source address. For example, a number of common types of denial-of-service (DoS) attacks, including smurf and tribe flood network (TFN), can take advantage of forged or rapidly changing source addresses to allow attackers to thwart efforts to locate or filter the attacks. For Internet service providers (ISPs) that provide public access, uRPF deflects such attacks by forwarding only packets with source addresses that are valid and consistent with the IP routing table. This action protects the network of the ISP, its customer, and the rest of the Internet.

uRPF is supported for both IPv4 and IPv6 on network and access. It is supported on any IP interface, including base router, IES, VPRN, and subscriber group interfaces.

In strict mode, uRPF checks whether the incoming packet has a source address that matches a prefix in the routing table, and whether the interface expects to receive a packet with this source address prefix.

In loose mode, uRPF checks whether the incoming packet has a source address that matches a prefix in the routing table; loose mode does not check whether the interface expects to receive a packet with a specific source address prefix.

Loose mode uRPF check is supported for ECMP, IGP shortcuts, and VPRN MP-BGP routes. Packets coming from a source that matches any ECMP, IGP shortcut, or VPRN MP-BGP route passes the uRPF check even when uRPF is set to strict mode on the incoming interface.

In the case of ECMP, this allows a packet received on an IP interface configured in strict uRPF mode to be forwarded if the source address of the packet matches an ECMP route, even if the IP interface is not a next-hop of the ECMP route or not a member of any ECMP routes. The strict-no-ecmp uRPF mode may be configured on any interface that is known to not be a next-hop of any ECMP route. When a packet is received on this interface, and the source address matches an ECMP route, the packet is dropped by uRPF.

If there is a default route, the following is included in the uRPF check:

- A loose mode uRPF check always succeeds.
- A strict mode uRPF check only succeeds if the source address matches any route (including the default route) where the next-hop is on the incoming interface for the packet.

Otherwise, the uRPF check fails.

If the source IP address matches a discard/blackhole route, the packet is treated as if it failed the uRPF check.

2.1.1.5 Configuring link delay

The delay represents the unidirectional link delay from the local router to the remote router (that is, the forward-path latency). The interface delay is a link property and is typically calculated as the combination of speed of light versus fiber length versus fiber composition. Typically, these delay components are not subject to sudden change in a network. If change occurs, it tends to be because of fiber cuts (such as light out) or Layer 1 reroute events.

If **delay** is configured for all links in the network, the attribute can be used as a feasible metric for SR flex-algo applications.

The static delay represents a forward-path metric, in microseconds, between two routers. It is not possible to configure a delay on a loopback or system interface; the delay IGP extension TLVs (specified in RFC 8570) are not defined for stub links. The delay is encoded in IGP application-specific attributes (for example, for IS-IS, see *draft-ietf-isis-te-app-14.txt*). The delay can be configured upon other interface links.

The default setting is no delay, which means that IGP (for example, IS-IS) does not add a link delay metric TLV. The lack of this TLV in flex-algo causes the link with the no delay TLV setting to be pruned from the topology.

The following example shows the configuration of link delay.

Output example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface interface-name {
        if-attribute {
            delay {
```

```
                static microseconds
            }
        }
    }
```

Output example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------------
echo "IP Configuration"
#------------------------------------------------
        interface "interface-name"
            if-attribute
                delay
                    static microseconds
                exit
            exit
            no shutdown
        exit
```

The static delay can be configured within the range 1 to 16777214 microseconds.

2.1.2 Router ID

The router ID, a 32-bit number, uniquely identifies the router within an autonomous system (AS) (see Autonomous systems). In protocols such as OSPF, routing information is exchanged between areas—groups of networks that share routing information. It can be set to be the same as the loopback address. The router ID is used by both OSPF and BGP routing protocols in the routing table manager instance.

There are several ways to obtain the router ID. On each router, the router ID can be obtained in the following ways.

- Define the router ID. Use the following command to define the value that becomes the router ID.

  ```
  configure router
  ```

- Configure the system interface with an IP address. If the router ID is not manually configured in the **configure router** context, the system interface acts as the router ID. Use the following command to configure the system interface with an IP address.

  ```
  configure router interface
  ```

- If neither the system interface nor router ID are implicitly specified, the router ID is inherited from the last four bytes of the MAC address.
- The router can be obtained from the protocol level; for example, BGP.

2.1.3 Autonomous systems

Networks can be grouped into areas. An area is a collection of network segments within an AS that have been administratively assigned to the same group. The topology of an area is concealed from the rest of the AS, which results in a significant reduction in routing traffic.

Routing in the AS takes place on two levels, depending on whether the source and destination of a packet reside in the same area (intra-area routing) or different areas (inter-area routing). In intra-area routing,

the packet is routed solely on information obtained within the area; no routing information obtained from outside the area can be used. This protects intra-area routing from the injection of bad routing information.

Routers that belong to more than one area are called area border routers. All routers in an AS do not have an identical topological database. An area border router has a separate topological database for each area it is connected to. Two routers, which are not area border routers, belonging to the same area, have identical area topological databases.

ASs share routing information, such as routes to each destination and information about the route or AS path, with other ASs using BGP. Routing tables contain lists of next-hops, reachable addresses, and associated path cost metrics to each router. BGP uses the information and path attributes to compile a network topology.

2.1.4 Confederations

Configuring confederations is optional and should be implemented only to reduce the iBGP mesh inside an AS. An AS can be logically divided into smaller groupings called sub-confederations and then assigned a confederation ID (similar to an autonomous system number (ASN)). Each sub-confederation has fully meshed iBGP and connections to other ASs outside of the confederation.

The sub-confederations have eBGP-type peers to other sub-confederations within the confederation. They exchange routing information as if they were using iBGP. Command options such as next hop, metric, and local preference are preserved. The confederation appears and behaves like a single AS.

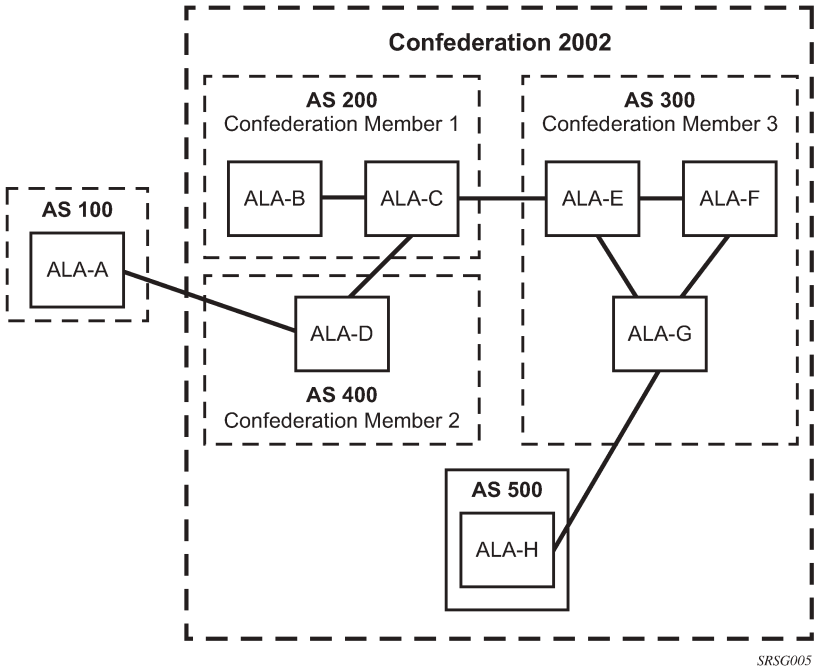
Confederations have the following characteristics:

- A large AS can be sub-divided into sub-confederations.
- Routing within each sub-confederation is accomplished via iBGP.
- eBGP is used to communicate between sub-confederations.
- BGP speakers within a sub-confederation must be fully meshed.
- Each sub-confederation (member) of the confederation has a different ASN. The ASNs used are typically in the private AS range of 64512 to 65535.

To migrate from a non-confederation configuration to a confederation configuration requires a major topology change and configuration modifications on each participating router. Setting BGP policies to select an optimal path through a confederation requires other BGP modifications.

There are no default confederations. Router confederations must be explicitly created. The following figure shows an example of a confederation configuration.

Figure 1: Confederation configuration



2.1.5 Proxy ARP

Proxy ARP is the technique in which a router answers ARP requests intended for another node. The router appears to be present on the same network as the "real" node that is the target of the ARP and takes responsibility for routing packets to the "real" destination. Proxy ARP can help nodes on a subnet reach remote subnets without configuring routing or a default gateway. Typical routers only support proxy ARP for directly attached networks; the router is targeted to support proxy ARP for all known networks in the routing instance where the virtual interface proxy ARP is configured.

To support DSLAM and other edge-like environments, proxy ARP supports policies that allow the provider to configure prefix lists that determine the target networks and source hosts for which proxy ARP is attempted.

In addition, the proxy ARP implementation supports the ability to respond for other hosts within the local subnet domain. This is needed in environments such as DSL where multiple hosts are in the same subnet but cannot reach each other directly.

Static ARP is used when a Nokia router needs to know about a device on an interface that cannot or does not respond to ARP requests. The configuration can state that if it has a packet with a specific IP address, to send it to the corresponding ARP address. Use proxy ARP so the router responds to ARP requests on behalf of another device.

2.1.6 Exporting an inactive BGP route from a VPRN

Use the following command to provide an IP VPN command option that allows the best BGP route learned by a VPRN to be exported as a VPN-IP route even when that BGP route is inactive because of the presence of a more preferred BGP-VPN route from another PE.

```
configure service vprn export-inactive-bgp
```

This “best-external” type of route advertisement is useful in active or standby multihoming scenarios because it can ensure that all PEs have knowledge of the backup path provided by the standby PE.

2.1.7 DHCP relay

Because DHCP requests are broadcast packets that normally do not propagate outside of their IP subnet, a DHCP relay agent intercepts such requests and forwards them as unicast messages to a configured DHCP server.

When forwarding a DHCP message, the relay agent sets the GIADDR in the packet to the IP address of its ingress interface. This allows DHCP clients to use a DHCP server on a remote network. From both a scalability and a security point of view, it is recommended that the DHCP relay agent is positioned as close as possible to the client terminals.

DHCP relay is used in a Layer 3 environment, and therefore is only supported in IES services and VPRN services.

When DHCP clients and servers are in different VPRN routing instances of which one is the Base routing instance, route leaking (GRT-leaking) should be used to relay DHCPv4 and DHCPv6 messages between a VPRN and the Global Routing Table (GRT).

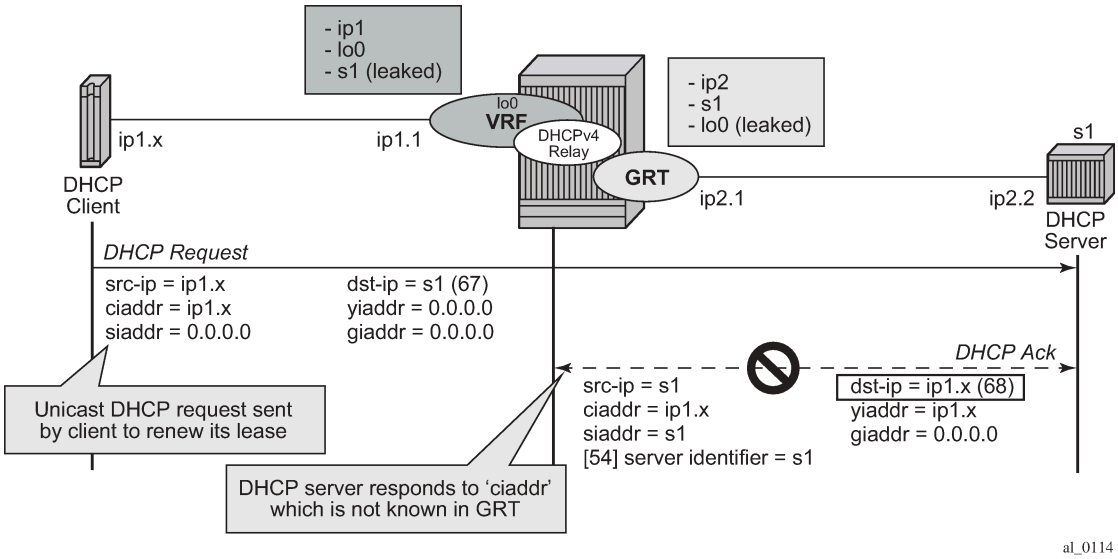
While DHCP relay is not implemented in a VPLS, it is still possible to insert or modify Option 82 information.

In a routed CO environment, the subscriber interface’s group interface DHCP relay is stateful.

2.1.8 DHCPv4 relay proxy

In network deployments where DHCPv4 client subnets cannot be leaked in the DHCPv4 server routing instance, unicast renewal messages (DHCP ACKs) cannot be routed in the DHCPv4 server routing instance, as shown in Figure 2: Unicast renewal routing problem. The DHCP server sets the destination IP address of the DHCP ACK to the client IP address (ciaddr) as received in the DHCP REQUEST message. Because there is no route available for the client subnet in the DHCP server routing instance, the DHCP ACK cannot be delivered.

Figure 2: Unicast renewal routing problem



The unicast renewal routing problem shown in Figure 2: Unicast renewal routing problem can be solved with a relay proxy function that enhances the DHCPv4 relay. Use the following command to resolve this problem.

- **MD-CLI**

```
configure service ies interface ipv4 dhcp relay-proxy
configure service ies subscriber-interface group-interface ipv4 dhcp relay-proxy
configure service ies subscriber-interface ipv4 dhcp relay-proxy
configure service ies interface ipv4 dhcp relay-proxy
configure service ies subscriber-interface group-interface ipv4 dhcp relay-proxy
configure service ies subscriber-interface ipv4 dhcp relay-proxy
```

- **classic CLI**

```
configure service ies interface dhcp relay-proxy
configure service ies subscriber-interface dhcp relay-proxy
configure service ies subscriber-interface group-interface dhcp relay-proxy
configure service vprn interface dhcp relay-proxy
configure service vprn subscriber-interface dhcp relay-proxy
configure service vprn subscriber-interface group-interface dhcp relay-proxy
```

With the **relay-proxy** command in the DHCPv4 relay on a regular interface or group interface, the unicast renewals are now also relayed to the DHCPv4 server, as described below and shown in Figure 3: Relay unicast messages:

- In the client to server direction, the source IP address is updated and the gateway IP address (gi-address) field is added before sending the message to the intended DHCP server (the message is not broadcasted to all configured DHCP servers.
- In the server to client direction, the GI address field is removed and the destination IP address is updated with the value of the IP address (yiaddr) field.

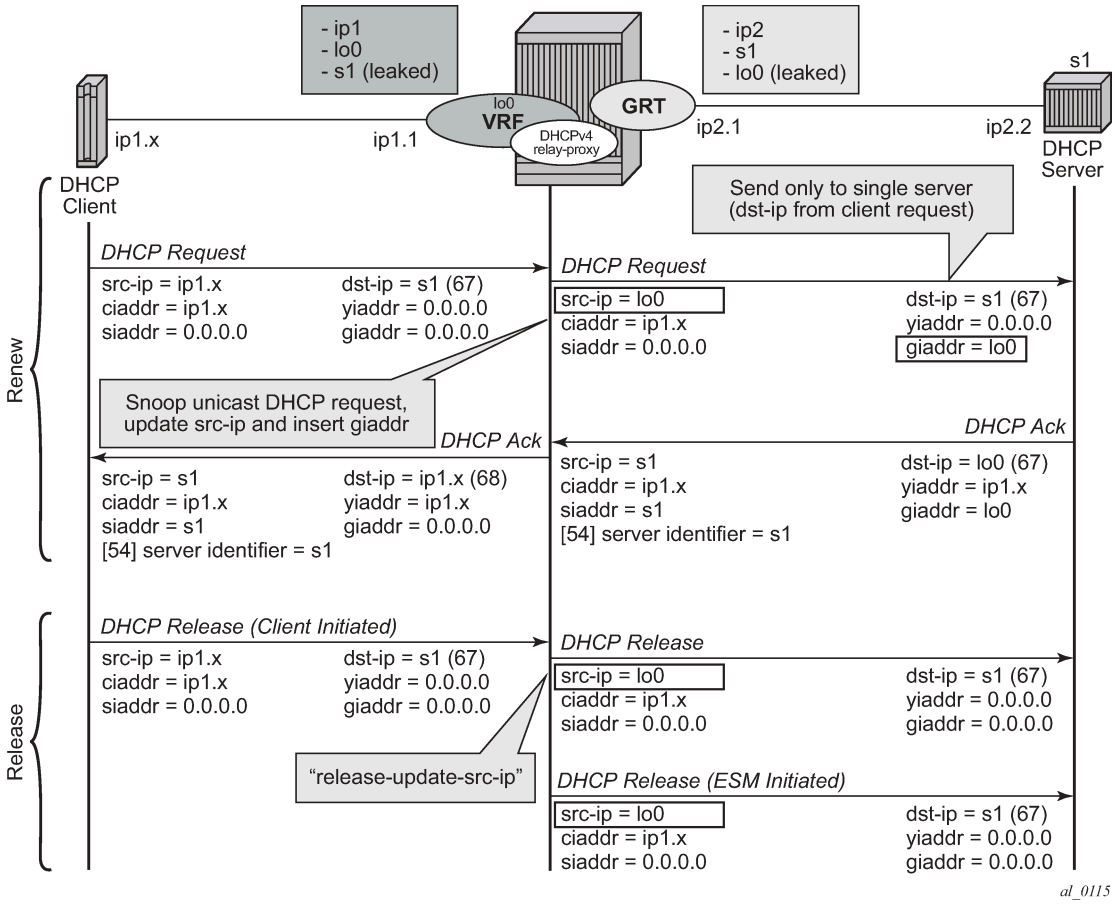
When **relay-proxy** is enabled, the GI address can be configured to any local address that is configured in the same routing instance. The GI address is the only address that must be leaked in the DHCPv4 server

routing instance because a DHCPv4 server always sends the response on a relayed packet to the relay agent using the gi-address as the destination IP address.

By default, unicast DHCPv4 RELEASE messages are forwarded transparently by a relay proxy function. The optional **release-update-src-ip** command option updates the source IP address with the value that is used for all relayed DHCPv4 messages, as shown in Figure 3: Relay unicast messages.

DHCPv4 FORCERENEW messages that are sent from a trusted external DHCPv4 server to a DHCPv4 relay agent configured as a relay proxy are forwarded to the DHCP client, if a corresponding DHCPv4 lease exists; otherwise, the DHCPv4 FORCERENEW messages are dropped.

Figure 3: Relay unicast messages



The **relay-proxy** command can also be used to hide the DHCPv4 server address for DHCP clients. This prevents the client from learning the DHCPv4 server infrastructure details such as the IP address and number of servers. Hiding infrastructure details helps in Denial of Service (DoS) prevention.

The optional **siaddr-override** command option in relay-proxy enables DHCPv4 server IP address hiding toward the client. The client interacts with the relay proxy as if it is the DHCP server. In addition to the relay proxy functions as described earlier, the following actions are performed when DHCPv4 server IP address hiding is configured:

- In all DHCP messages to the client, the value of the following header fields and DHCP options containing the DHCP server IP address is replaced with the configured IP address:
  - the source IP address

  - the siaddr field in the DHCPv4 header if it is not equal to zero in the message received from the server
  - the Server Identification option (DHCPv4 option 54) if present in the original server message
- The DHCP OFFER selection occurs during initial binding. Only the first DHCP OFFER message is forwarded to the client. Subsequent DHCP OFFER messages from different servers are silently dropped.

The **siaddr-override** command option can be any local address in the same routing instance. If DHCP relay lease split is enabled, **siaddr-override** command option has priority over the **emulated-server** configured in the proxy server and is used as the source IP address.

The active DHCPv4 server IP address obtained from the DHCP OFFER selection is required for the IP address hiding function and is stored in the lease state record. Therefore, the following command must be enabled on the interface when **siaddr-override** is configured.

- **MD-CLI**

```
configure service ies interface ipv4 dhcp lease-populate
configure service ies subscriber-interface group-interface ipv4 dhcp lease-populate
configure service vprn interface ipv4 dhcp lease-populate
configure service vprn subscriber-interface ipv4 dhcp lease-populate
configure service vprn subscriber-interface group-interface ipv4 dhcp lease-populate
```

- **classic CLI**

```
configure service ies interface dhcp lease-populate
configure service ies subscriber-interface group-interface dhcp lease-populate
configure service vprn interface dhcp lease-populate
configure service vprn subscriber-interface dhcp lease-populate
configure service vprn subscriber-interface group-interface dhcp lease-populate
```

Figure 4: DHCP server IP address hiding/initial binding shows the initial lease binding phase of a relay proxy with DHCP server address hiding enabled. In the absence of a DHCP lease state in the initial lease binding phase, the DHCP server IP address resulting from the OFFER selection is stored in a DHCP transaction cache. After successful lease binding, the DHCP server IP address is added to the lease state record.

In a host creation failure scenario, if no transaction cache or lease state is available when a DHCP REQUEST message is received, then the DHCP REQUEST is silently dropped. The drop reason can be found by enabling DHCP debug.

Figure 4: DHCP server IP address hiding/initial binding

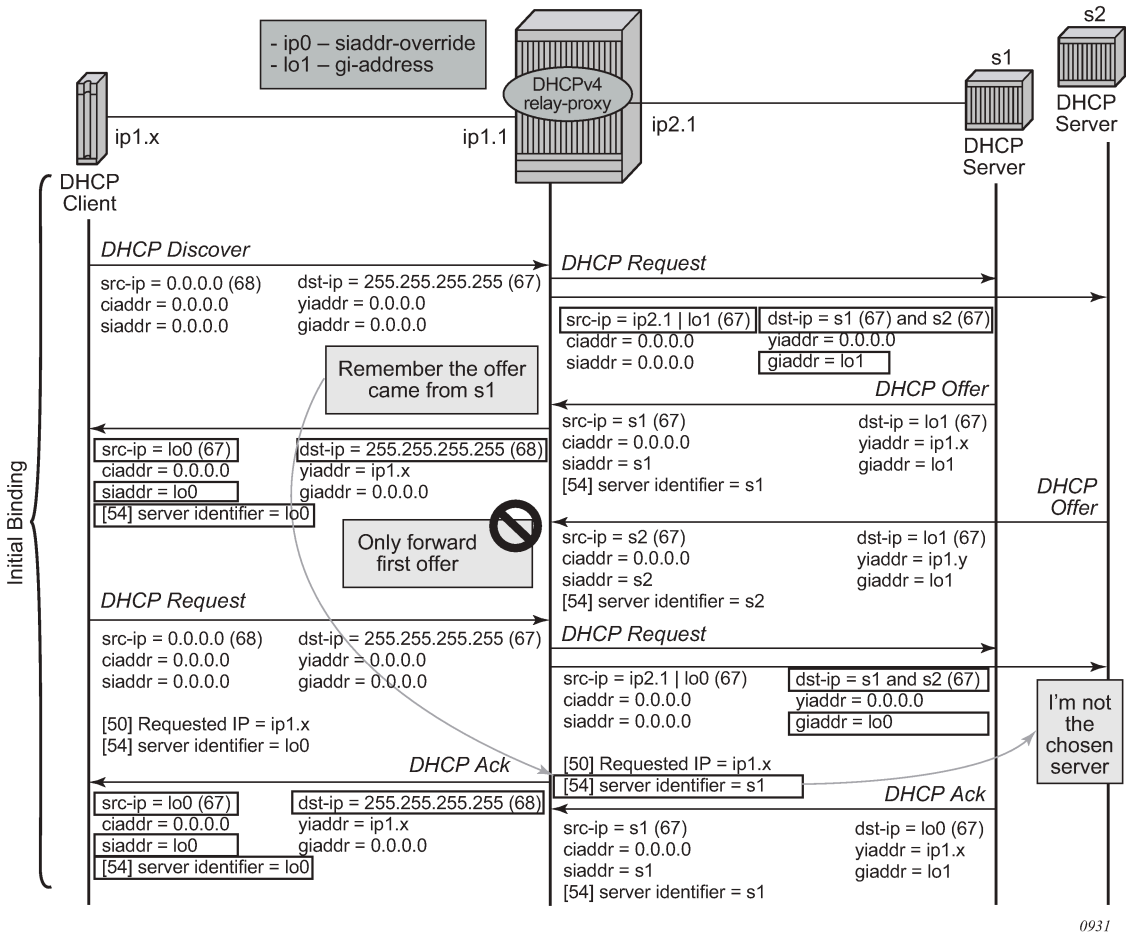


Figure 5: DHCP server IP address hiding/lease renewal shows the lease renewal phase of a relay proxy with DHCP server address hiding enabled. A unicast REQUEST (renew) is relayed only to the DHCP server owning the lease. A broadcast REQUEST (rebind) is relayed to all configured DHCP servers.

During lease renewal, the DHCP server IP address can be updated in the lease state if the DHCP ACK is received from a different server. This optimizes the DHCP proxy relay operation in a DHCP server failover scenario. This is shown in Figure 6: DHCP server IP address hiding, lease renewal with active server failure.

Figure 5: DHCP server IP address hiding/lease renewal

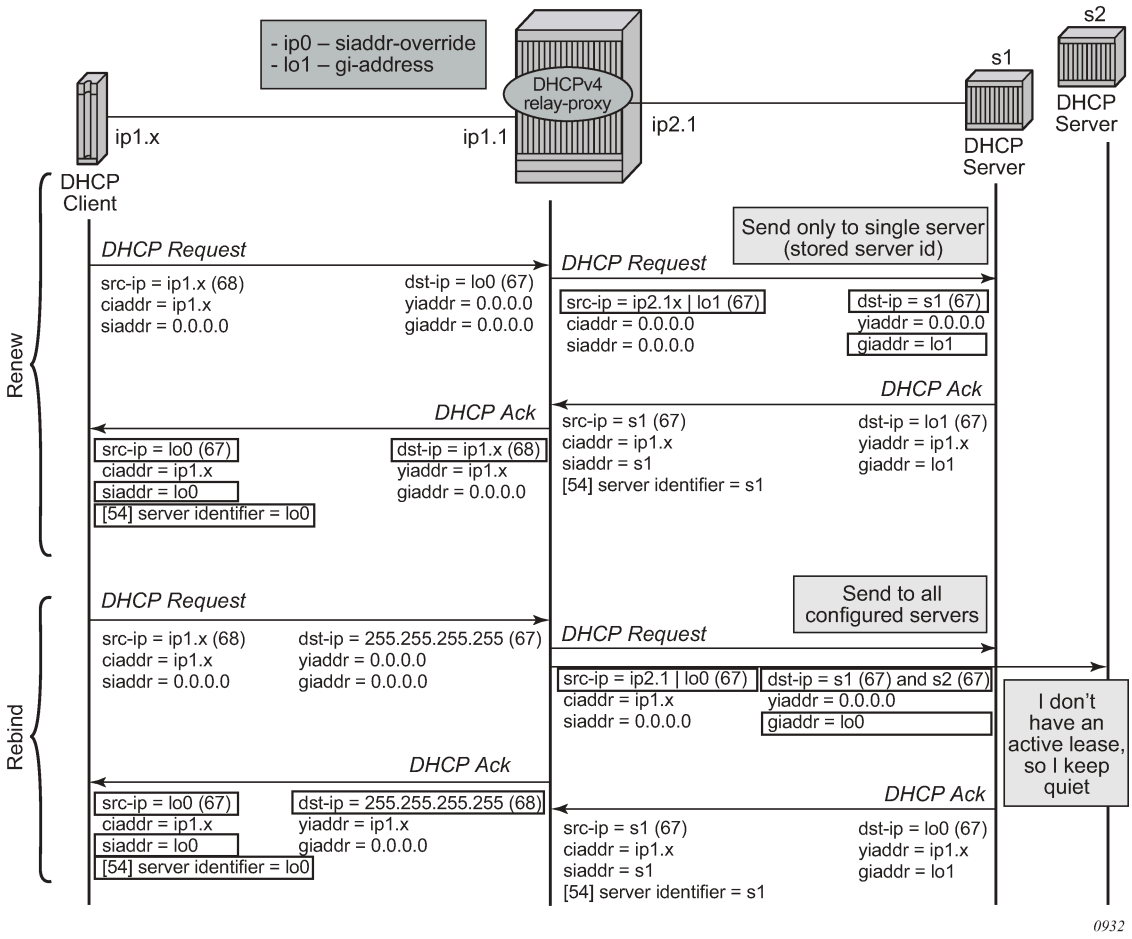


Figure 6: DHCP server IP address hiding, lease renewal with active server failure

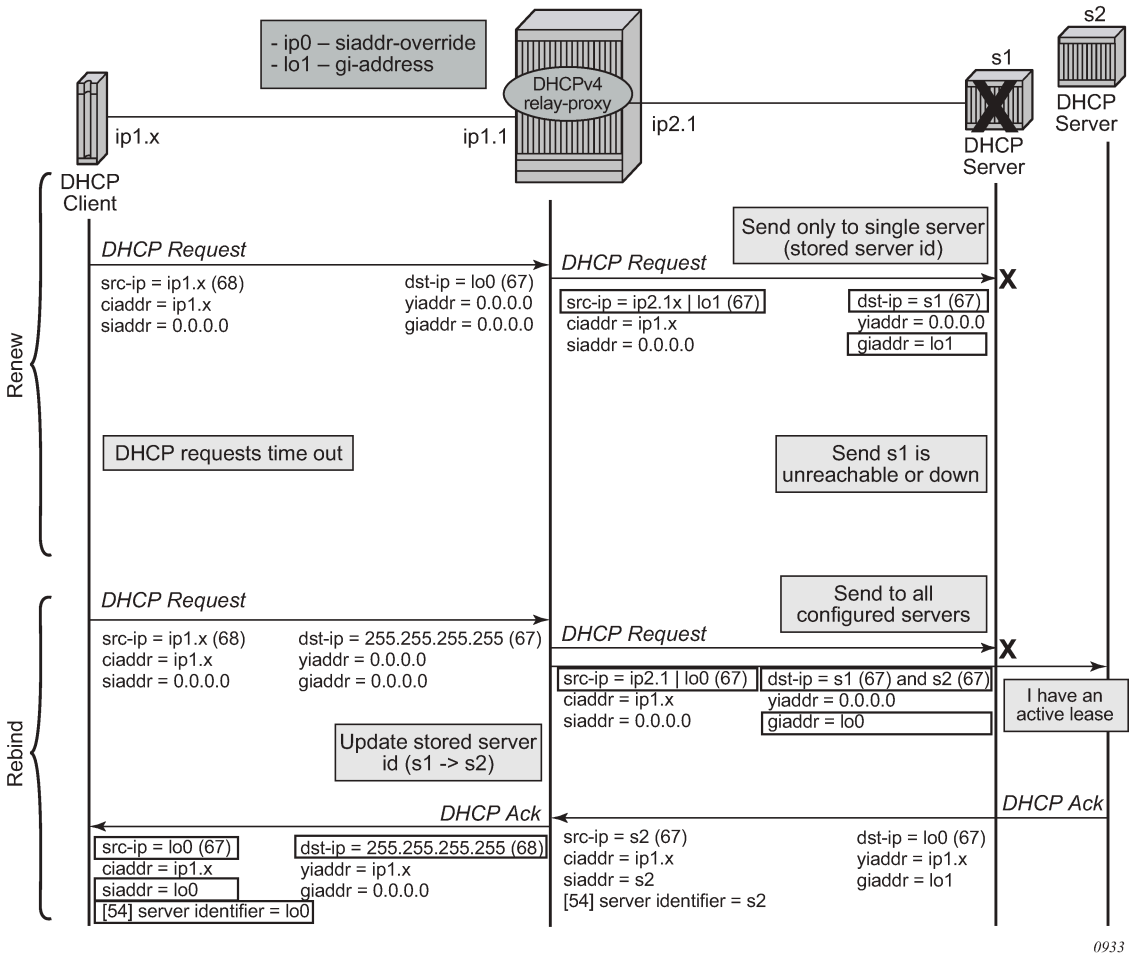
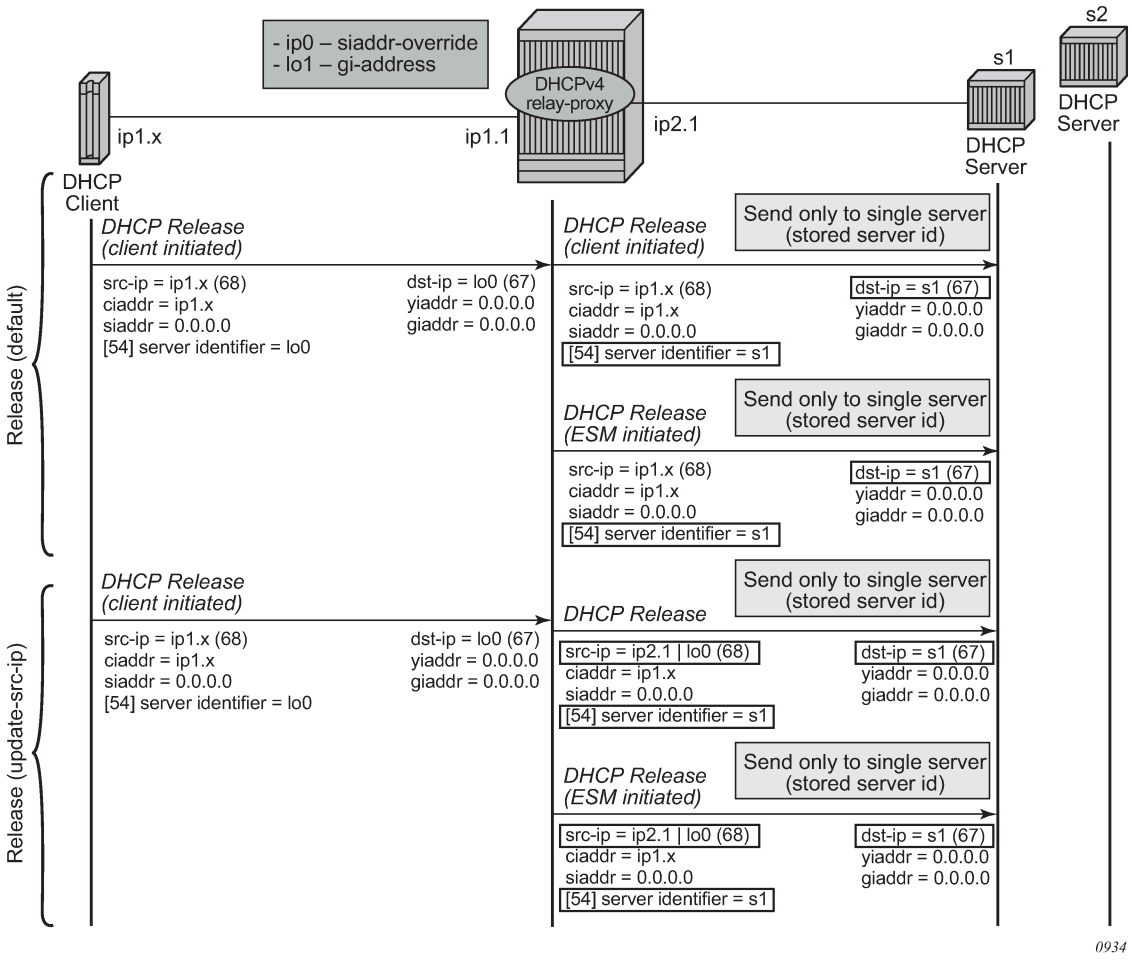


Figure 7: DHCP server IP address hiding, release shows the release in a relay proxy scenario with DHCP server address hiding enabled. The RELEASE message is sent only to the DHCP server owning the lease. Optionally, the source IP address can be updated.

Figure 7: DHCP server IP address hiding, release



Relay proxy can be enabled on subscriber group-interfaces and regular interfaces in an IES or VPRN service.

A relay proxy function is not supported with a double DHCPv4 relay (Layer 3 DHCPv4 relay in front of a DHCPv4 relay with **relay-proxy** enabled).

For retail subscriber interfaces, **relay-proxy** is configured as shown in the following example.

Example: MD-CLI

```
[ex:/configure service vprn "1"]
A:admin@node-2# info
    customer "1"
    interface "lo0" {
        loopback true
        ipv4 {
            primary {
                address 192.0.2.10
                prefix-length 32
            }
        }
    }
```

```
    interface "lo1" {
        loopback true
        ipv4 {
            primary {
                address 192.0.2.11
                prefix-length 32
            }
        }
    }
    subscriber-interface "sub-int-1" {
        ipv4 {
            address 10.1.0.254 {
                prefix-length 24
            }
        }
        group-interface "group-int-1-1" {
            ipv4 {
                dhcp {
                    admin-state enable
                    server [172.16.1.1]
                    gi-address 192.0.2.11
                    src-ip-addr gi-address
                    lease-populate {
                        max-leases 32767
                    }
                    relay-proxy {
                        release-update-src-ip true
                        siaddr-override 192.0.2.10
                    }
                }
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>service>vprn$ info
----------------------------------------------
            shutdown
            interface "lo0" create
                address 192.0.2.10/32
                loopback
            exit
            interface "lo1" create
                address 192.0.2.11/32
                loopback
            exit
            subscriber-interface "sub-int-1" create
                address 10.1.0.254/24
                group-interface "group-int-1-1" create
                    dhcp
                        server 172.16.1.1
                        lease-populate 32767
                        relay-proxy release-update-src-ip siaddr-override 192.0.2.10
                        gi-address 192.0.2.11 src-ip-addr
                        no shutdown
                    exit
                exit
            exit
----------------------------------------------
```

2.1.9 DHCP client

In the base router context, Ethernet ports can be configured with a router interface that supports a DHCP client. When the node operates as a DHCP client, it learns the IP address of the interface via dynamic IP address assignment. The DHCP client functionality is enabled by issuing the **no shutdown** command on the DHCP client in the **configure router interface autoconfigure dhcp-client** context. The following output shows an example of a router interface enabled as a DHCP client.

```
*A:DUT# config# router interface "station-wlan-ifc"
    port 1/4/4
    autoconfigure dhcp-client
        no shutdown
        exit
    exit
```

The 7705 SAR Gen 2 supports up to three DHCP clients per node .

When the DHCP client is enabled, changes to the DHCP client configuration take effect when the **shutdown** command is issued followed by the **no shutdown** command.

If DHCP relay configurations exist on the node, the DHCP client cannot be enabled until the DHCP relay configurations are removed. Similarly, if DHCP client configurations exist on the node, DHCP relay cannot be enabled until the DHCP client configurations are removed.

The DHCP client only supports IPv4.

When the DHCP client first becomes operational, it learns an IP address from a remote DHCP server using a DHCPDISCOVER message.

The node only sends a DHCPDISCOVER message if:

- the DHCP client is enabled and the router interface is operationally up. Shutting down the DHCP client forces the release of the IP address.
- a DHCP NAK message is received from the DHCP server that invalidates the previous DHCP DISCOVER message or any existing lease

When a DHCP client is shut down, all cached values (such as IP addresses and DHCP options) are cleared. They are rediscovered by issuing the **no shutdown** command.

If the port comes operationally up while the DHCP client is enabled and a DHCP discovery was not previously completed or a DHCP release was previously issued, then DHCP discovery is performed. If the port comes operationally up while the DHCP client is enabled and there was a previously completed DHCP discovery, then the DHCP client performs a DHCPREQUEST using the previously cached DHCP information from the discovery.

The operator can force a rediscovery procedure by executing the restart command in the **tools perform router autoconfigure dhcp-client interface** context.

The requested DHCP lease time can be configured using the CLI; however, the DHCP server can override this value. The node tracks the DHCP lease time and sends a DHCPREQUEST when half the lease time has elapsed. An IP address lease can be renewed manually using the **tools perform router autoconfigure dhcp-client interface lease-renew** command.

If the router interface goes down, the DHCP client parameters are cached for the interface. When the interface comes back up, if an IP address has been allocated and the lease time has not expired, the DHCP router interface sends a DHCPREQUEST to confirm that it can continue to use the IP address associated with the lease.

DHCP options must be configured in the CLI to make use of options received by the DHCP server. Any options received from the DHCP server are ignored if the corresponding options are not specified in the CLI. The DHCP client options are **router**, **static-route**, and **dns-server**. They are configured in the **config router interface autoconfigure dhcp-client request-options** context.

The **show router route-table protocol dhcp-client** command can be used to view the active routes in the routing table that have been learned by the DHCP client. If the same route is received from more than one DHCP client, the route received from the DHCP server with the lowest ID (option 54) is installed in the route table.

The **show router dns** command can be used to view whether the DNS server has been configured to send request messages to the DHCP server. The node supports up to six DNS server entries learned by the DHCP clients. Only the first six DNS servers are stored by the node; any subsequent DNS servers that are learned are ignored.

The CLI provides the option to use the router from the DHCPOFFER as the default gateway. In some scenarios, the router that is reachable from the WLAN port or an Ethernet port is the default gateway. In other scenarios, the cellular interface has reachability to the default gateway. The DHCP client **router** option (under **request-options**) enables the **router** request option in the DHCPOFFER message. If the **router** option is enabled, the default gateway is assigned by the DHCP server.

The DHCPDISCOVER message sent from the node to the DHCP server contains the following options:

- chaddr—the MAC address of the client, either the WLAN or Ethernet port
- Option 51—the configured IP address lease time
- Option 53—the DHCP message type (DISCOVER)
- Option 60—a user-configurable vendor class identifier, either a hexadecimal string or an ASCII string
- Option 61—a user-defined client identifier: a hexadecimal string, an ASCII string, an interface name, or the client MAC address
- Option 55—the parameter request list:
  - Option 1—the subnet mask value
  - Option 3—the router option, a list of IP addresses for routers on the client subnet (unused if not enabled in the CLI)
  - Option 54—the DHCP server address

The DHCPOFFER message from the DHCP server must contain the following options at a minimum:

- yiaddr—the DHCP router interface IP address
- Option 1—the subnet mask value
- Option 3—the router option, a list of IP addresses for routers on the client subnet
- Option 51—the configured IP address lease time
- Option 53—the DHCP message type (OFFER)
- Option 54—the DHCP server address

When responding to the server DHCPOFFER or when extending the time of an existing lease, the DHCPREQUEST message sent from the node to the DHCP server contains the following options:

- chaddr—the client MAC address
- Option 50—the requested IP address; this address is the same as the address contained in the yiaddr field that was received in the DHCPOFFER message

- Option 53—the DHCP message type (REQUEST)
- Option 54—the DHCP server address; this address is the same as the address received in the OFFER message
- Option 51—the IP address lease time; this value is the same as the lease time received in the OFFER message
- Option 60—the vendor class identifier; this value is the same as the vendor class identifier in the DISCOVER message
- Option 61—the client identifier; this value is the same as the client identifier in the DISCOVER message
- Option 55—the parameter request list:
  - Option 1—the subnet mask value
  - Option 3—the router option (unused if not enabled in the CLI)
  - Option 6—the DNS server option (unused if not enabled in the CLI)
  - Option 54—the DHCP server address
  - Option 121—the static-route option (unused if not enabled in the CLI)

When the DHCP client is shut down, a DHCPRELEASE message is sent to the DHCP server.

For BGP peers to other nodes behind the WLAN AP, the BGP local address can be set using the router interface name where the DHCP client is configured so that changes in the interface address because of DHCP messages are reflected in the local address of BGP sessions using this interface as the local address.

2.1.9.1 Restrictions on configuring a router interface with DHCP client enabled

When a DHCP client is enabled on a router interface, the following protocols and services are supported on this interface:

- BGP with local-address set to this interface name
- Layer 3 VPRN services using mp-BGP
- Layer 2 VPLS/VPWS services using BGP-VPLS and BGP-VPWS
- Static routing using this interface as the next-hop
- IPsec secured interface

Note: Other routing protocols, unicast and multicast-based services, and OAM functionality not specified in the preceding list may be configurable, but are not supported on a DHCP client-enabled interface.

When a DHCP client is enabled on a router interface, the following commands cannot be configured in the **configure router interface** context:

- **address**
- **secondary**
- **dhcp**
- **unnumbered**
- **loopback**

If any of the commands in the preceding list are enabled, the **no shutdown** command is not available for the DHCP client until the commands are removed.

2.1.9.2 Route policy option for DHCP client

Routes can be imported from the DHCP client to other routing protocols with the **configure router policy-options policy-statement entry from protocol dhcp-client** command.

2.1.9.3 GRE termination for services over a DHCP client

A router interface configured as a DHCP client supports the following service types: VLL, VPLS, and VPRN. These services use a GRE SDP as a transport tunnel.

When a DHCP client is enabled on a router interface and an address is learned by the client, there is no configuration required in order to terminate GRE SDPs on that interface IP address. GRE termination is enabled on a DHCP client address when the client learns the address.

2.1.10 IP versions

The SR OS implements IP routing functionality, providing support for IP version 4 (IPv4) and IP version 6 (IPv6). IP version 6 (RFC 1883, *Internet Protocol, Version 6 (IPv6)*) is a version of the Internet Protocol designed as a successor to IP version 4 (IPv4) (RFC-791, *Internet Protocol*). The changes from IPv4 to IPv6 affect the following categories:

- **expanded addressing capabilities**

  IPv6 increases the IP address size from 32 bits (IPv4) to 128 bits, to support more levels of addressing hierarchy, a much greater number of addressable nodes, and simpler autoconfiguration of addresses. The scalability of multicast routing is improved by adding a scope field to multicast addresses. Also, a new type of address called an anycast address is defined and is used to send a packet to any one of a group of nodes.

- **header format simplification**

  Some IPv4 header fields have been dropped or made optional to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.

- **improved support for extensions and options**

  Changes in the way IP header options are encoded allows for more efficient forwarding, less stringent limits on the length of options, and greater flexibility for introducing new options in the future.

- **flow labeling capability**

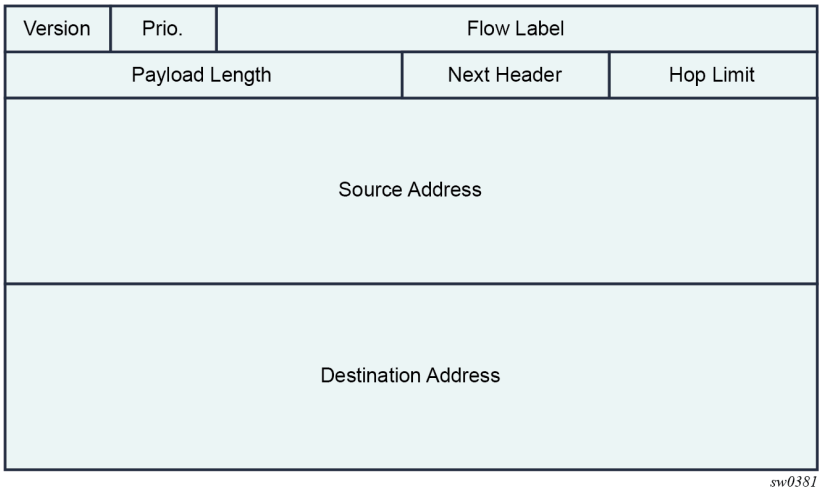
  The capability to enable the labeling of packets belonging to particular traffic flows for which the sender requests special handling, such as non-default quality of service or "real-time" service, was added in IPv6.

- **authentication and privacy capabilities**

  Extensions to support authentication, data integrity, and (optional) data confidentiality are specified for IPv6.

The following figure shows the IPv6 header format.

Figure 8: IPv6 header format



The following table lists IPv6 header fields and their descriptions.

Table 2: IPv6 header field descriptions

Field	Description
Version	4-bit Internet Protocol version number = 6.
Prio.	4-bit priority value.
Flow Label	24-bit flow label.
Payload Length	16-bit unsigned integer; the length of payload, for example, the rest of the packet following the IPv6 header, in octets; if the value is zero, the payload length is carried in a jumbo payload hop-by-hop option
Next Header	8-bit selector. Identifies the type of header immediately following the IPv6 header. This field uses the same values as the IPv4 protocol field.
Hop Limit	8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if the hop limit is decremented to zero.
Source Address	128-bit address of the originator of the packet.
Destination Address	128-bit address of the intended recipient of the packet (possibly not the ultimate recipient if a routing header is present).

2.1.10.1 IPv6 address format

IPv6 uses a 128-bit address, as opposed to the IPv4 32-bit address. Unlike IPv4 addresses, which use the dotted-decimal format, with each octet assigned a decimal value from 0 to 255, IPv6 addresses use the colon-hexadecimal format X:X:X:X:X:X:X:X, where each X is a 16-bit section of the 128-bit address. For example:

2001:0db8:0000:0000:0000:0000:0000:0000

Leading zeros must be omitted from each block in the address. A series of zeros can be replaced with a double colon. For example:

2001:db8::

The double colon can only be used one time in an address.

The IPv6 prefix is the part of the IPv6 address that represents the network identifier, which appears at the beginning of the address. The IPv6 prefix length, which begins with a forward slash (/), shows how many bits of the address make up the network identifier. For example, the address 2001:db8:8086:6502::1/64 means that the first 64 bits of the address represent the network identifier; the remaining 64 bits represent the node identifier.

Note: IPv6 addresses and prefixes are displayed according to RFC 5952, *A Recommendation for IPv6 Address Text Representation*.

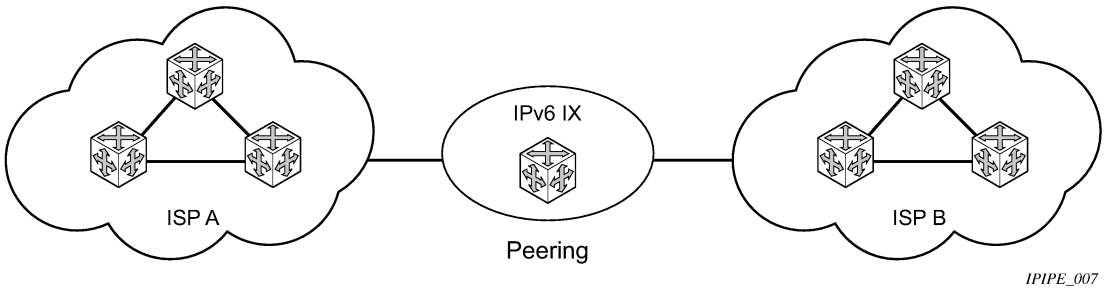
2.1.10.2 IPv6 applications

Examples of the IPv6 applications supported by the SR OS include:

- **IPv6 Internet exchange peering**

  Figure 9: IPv6 Internet exchange shows an IPv6 Internet exchange where multiple ISPs peer over native IPv6

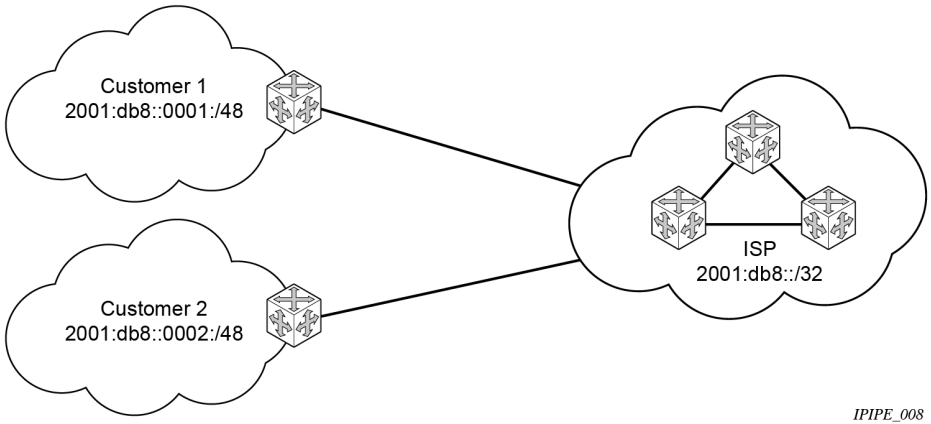
  Figure 9: IPv6 Internet exchange



- **IPv6 transit services**

  Figure 10: IPv6 transit services shows IPv6 transit services provided by an ISP.

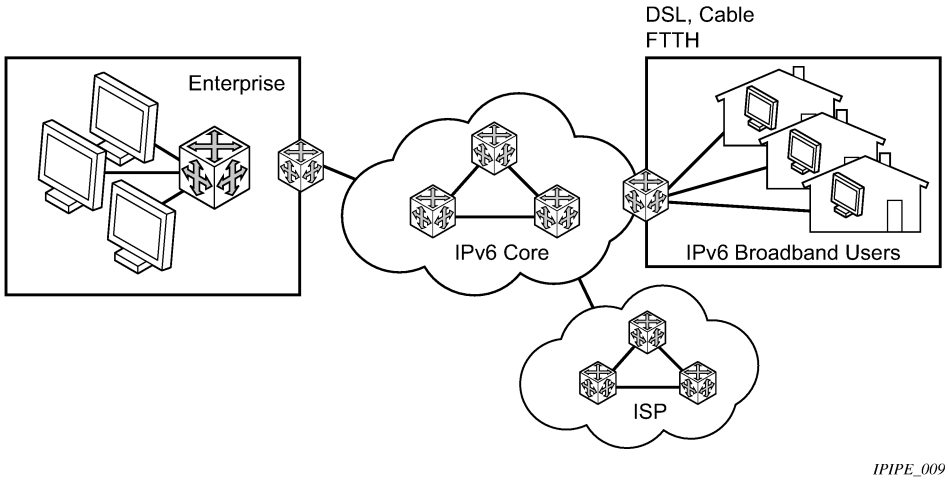
Figure 10: IPv6 transit services



- IPv6 services to enterprise customers and home users

  Figure 11: IPv6 services to enterprise customers and home users shows IPv6 services to enterprise and home broadband users.

  Figure 11: IPv6 services to enterprise customers and home users



- **IPv6 over IPv4 relay services**

  IPv6 over IPv4 tunnels are one of many IPv6 transition methods to support IPv6 in an environment where not only IPv4 exists but native IPv6 networks depend on IPv4 for greater IPv6 connectivity. Nokia routers support dynamic IPv6 over IPv4 tunneling. The IPv4 source and destination address are taken from configuration, the source address is the IPv4 system address and the IPv4 destination is the next hop from the configured IPv6 over IPv4 tunnel.

  IPv6 over IPv4 is an automatic tunnel method that gives a prefix to the attached IPv6 network. Figure 12: IPv6 over IPv4 tunnels shows IPv6 over IPv4 tunneling to transition from IPv4 to IPv6.

Figure 12: IPv6 over IPv4 tunnels

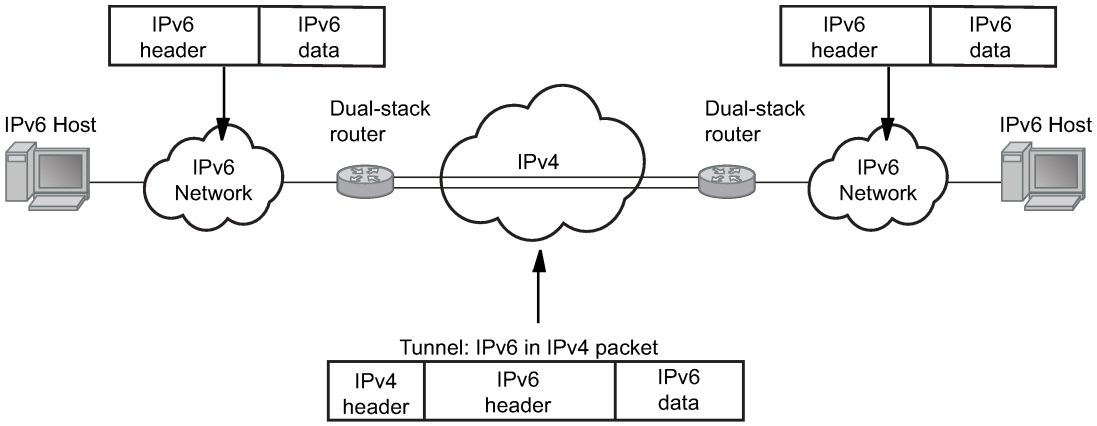


Fig_29a

2.1.10.3 DNS

The DNS client is extended to use IPv6 as transport and to handle the IPv6 address in the DNS AAAA resource record from an IPv4 or IPv6 DNS server. An assigned name can be used instead of an IPv6 address because IPv6 addresses are more difficult to remember than IPv4 addresses.

2.1.10.3.1 DNS resolution using a VPRN

When using a management VPRN, to allow DNS resolution via VPRN, as an example, DNS for all packets—routed through the Global Routing Table or the VPRN—the user must enable a redirect VPRN configuration under the base DNS server.

Use the following command to enable the redirect VPRN configuration.

```
configure router dns redirect-vprn service
```

When the redirect-vprn configuration is enabled, all packets have their URLs resolved through the configured redirect-vprn service. Only a single redirect-vprn configuration is supported.

As a prerequisite for the DNS resolution through the VPRN, the VPRN DNS server must be configured with at least a primary-dns IP address (IPv4 or IPv6). If the VPRN DNS server is not configured, all packet resolution fails, even if the BOF DNS server is configured, because the redirect-vprn configuration forces all packets through the redirect-vprn service for resolution.

The **redirect-vprn** command is not available at bootup, because the configuration is not loaded yet. Until the **redirect-vprn** command is executed, all DNS resolution is possible only through the BOF DNS configuration. The redirect-vprn configuration becomes active at runtime, after the configuration file is loaded and the **redirect-vprn** command is executed.

If the **redirect-vprn** command is not configured, DNS resolution occurs as follows:

- The global routing packets use the BOF DNS server.

- The VPRN packets use the configured VPRN DNS server. If the VPRN DNS server is not configured, the resolution occurs through the BOF DNS server.

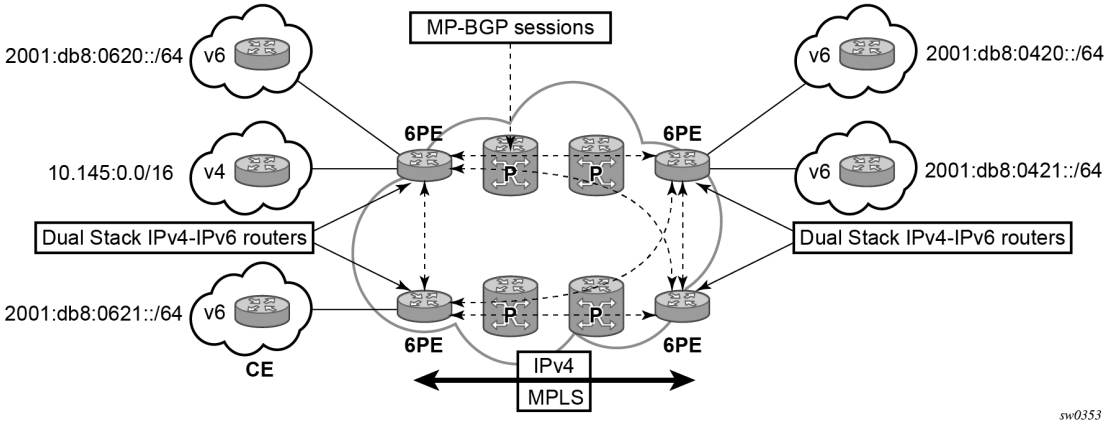
For information about management VPRNs, see *Node Management Using VPRN* in the *7705 SAR Gen 2 Layer 3 Services Guide: IES and VPRN*.

2.1.10.4 IPv6 Provider Edge over MPLS

The IPv6 Provider Edge (6PE) solution allows IPv6 domains to communicate with each other over an IPv4 MPLS core network. Because forwarding is based on MPLS labels, backbone infrastructure upgrades and core router reconfiguration is not required in this architecture. 6PE is a cost-effective solution for IPv6 deployment.

The following figure shows an example 6PE topology in an autonomous system (AS).

Figure 13: Example of a 6PE topology within an AS



2.1.10.4.1 6PE control plane support

The 6PE MP-BGP routers support:

- IPv4 and IPv6 dual-stack
- MP-BGP to exchange IPv6 reachability information:
  - The 6PE routers exchange IPv6 reachability information using MP-BGP (AFI 2, SAFI 4).
  - An IPv4 address of the 6PE router is encoded as an IPv4-mapped IPv6 address in the BGP next-hop field. This is usually the IPv4 system address.
  - The 6PE router binds MPLS labels to the IPv6 prefixes it advertises. SR OS routers advertise the IPv6 explicit null (value 2) in advertised 6PE routes but accept any arbitrary label from peers.
- The most preferred tunnel to the BGP next-hop allowed by the 6PE resolution filter is used to tunnel the traffic to the remote 6PE router. Use the following command to configure the 6PE resolution filter.

```
configure router bgp next-hop-resolution labeled-routes transport-tunnel family resolution-filter
```

2.1.10.4.2 6PE data plane support

The ingress 6PE router can push two or more MPLS labels to send the packets to the egress 6PE router. The top labels are associated with transport tunnel resolution. The remote 6PE router advertises the bottom label in MP-BGP. Typically, the IPv6 explicit null (value 2) label is used, but arbitrary values can be received when the remote 6PE router is not an SR OS router.

The egress 6PE router pops the top transport labels. When the IPv6 explicit null label is exposed, indicating that an IPv6 packet is encapsulated, the egress 6PE router pops the IPv6 explicit null label and performs an IPv6 route lookup to find the next hop for the IPv6 packet.

2.1.11 Static route resolution using tunnels

Use the commands in the following context to forward packets of a static route to an indirect next-hop over a tunnel programmed in TTM:

- **MD-CLI**

  ```
  configure router static-routes route indirect tunnel-next-hop
  ```

  In the MD-CLI, if the **tunnel-next-hop** context is configured and **resolution** is set to **none**, the binding to the tunnel is removed and resolution resumes in RTM to IP next-hops.

- **classic CLI**

  ```
  configure router static-route-entry indirect tunnel-next-hop
  ```

  In the classic CLI, if the **tunnel-next-hop** context is configured and **resolution** is set to **disabled**, the binding to the tunnel is removed and resolution resumes in RTM to IP next-hops.

If the **resolution** is set to **any**, any supported tunnel type in the static route context is selected following TTM preference.

The following tunnel types are supported in a static route context: LDP, RSVP-TE, Segment Routing (SR) Shortest Path, and Segment Routing Traffic Engineering (SR-TE):

- **LDP**

  The **ldp** command option instructs the code to search for an LDP LSP with a FEC prefix corresponding to the address of the indirect next-hop. Both LDP IPv4 FEC and LDP IPv6 FEC can be used as the tunnel next-hop. However, only an indirect next-hop of the same family (IPv4 or IPv6) as the prefix of the route can use an LDP FEC as the tunnel next-hop. In other words, an IPv4 (IPv6) prefix can only be resolved to an LDP IPv4 (IPv6) FEC.

- **RSVP-TE**

  The **rsvp-te** command option instructs the code to search for the set of lowest metric RSVP-TE LSPs to the address of the indirect next-hop. The LSP metric is provided by MPLS in the tunnel table. The static route treats a set of RSVP-TE LSPs with the same lowest metric as an ECMP set.

  The user has the option of configuring a list of RSVP-TE LSP names to be used exclusively instead of searching in the tunnel table. In that case, all LSPs must have the same LSP metric in order for the static route to use them as an ECMP set. Otherwise, only the LSPs with the lowest common metric value are selected.

A P2P auto-lsp that is instantiated via an LSP template can be selected in TTM when **resolution** is set to **any**. However, it is not recommended to configure an **auto-lsp** *name* explicitly under the **rsvp-te** node as the auto-generated name can change if the node reboots, which blackholes the traffic of the static route.

- **SR shortest path**

  When the **sr-isis** or **sr-ospf** command options are enabled, an SR tunnel to the indirect next-hop is selected in the TTM from the lowest preference IS-IS or OSPF instance, and if many instances have the same lowest preference, it is selected from the lowest numbered IS-IS or OSPF instance. Both SR-ISIS IPv4 and SR-ISIS IPv6 tunnels can be used as tunnel next-hops. However, only an indirect next-hop of the same family (IPv4 or IPv6) as the prefix of the route can use an SR-ISIS tunnel as a tunnel next-hop. In other words, an IPv4 (IPv6) prefix can only be resolved to a SR-ISIS IPv4 (IPv6).

- **SR-TE**

  The **sr-te** command option instructs the code to search for the set of lowest metric SR-TE LSPs to the address of the indirect next-hop. The LSP metric is provided by MPLS in the tunnel table. The static route treats a set of SR-TE LSPs with the same lowest metric as an ECMP set.

  The user has the option of configuring a list of SR-TE LSP names to be used exclusively instead of searching in the tunnel table. In that case, all LSPs must have the same LSP metric in order for the static route to use them as an ECMP set. Otherwise, only the LSPs with the lowest common metric value are selected.

Realize that the resolution filter, under static-route entry, does not validate the provided lsp-name type of the LSP against the requested filter context protocol type.

If one or more explicit tunnel types are specified using the **resolution-filter** command option, only these tunnel types are selected again following the TTM preference.

The user must set **resolution** to **filter** to activate the list of tunnel-types configured under resolution-filter.

If **disallow-igp** is enabled, the static route is not activated using IP next-hops in RTM if no tunnel next-hops are found in TTM.

2.1.11.1 Static route ECMP support

The following is the ECMP behavior of a static route:

- ECMP is supported when resolving in RTM multiple static routes of the same prefix with multiple user-entered indirect IP next-hops. The system picks as many direct next-hops as available in RTM beginning from the first indirect next-hop and up to the value of the **ecmp** command option in the system.
- ECMP is also supported when resolving in TTM a static route to a single indirect next-hop using a LDP tunnel when LDP has multiple direct next-hops.
- ECMP is supported when resolving in TTM a static route to a single indirect next-hop using a RSVP-TE tunnel type when there is more than one RSVP LSP with the same lowest metric to the indirect next-hop.
- ECMP is supported when resolving in TTM a static route to a single indirect next-hop using a list of user-configured RSVP-TE LSP names when these LSPs have the same metric to the indirect next-hop.
- ECMP is supported when resolving in TTM multiple static routes of the same prefix with multiple user-entered indirect next-hops, each binding to a tunnel type. The system picks as many tunnel next-hops as available in TTM beginning from the first indirect next-hop and up to the value of the **ecmp** command

option in the system. The spraying of flow packets is performed over the entire set of resolved next-hops that correspond to the selected indirect next-hops.

- ECMP is supported when resolving concurrently in RTM and TTM multiple static routes of the same prefix with multiple user-entered indirect tunnel next-hops. There is no support for mixing IP and tunnel next-hops for the same prefix using different indirect next-hops. Tunnel next-hops are preferred over IP next-hops.

2.1.11.2 Static route using flexible algorithms tunnels

When configuring a static route toward an indirect next hop, the path selection based upon the constraints of a particular Flex-Algorithm should be considered. In such a use case, it is necessary to steer traffic into a corresponding flexible algorithm segment routing tunnel. This can be achieved with the **tunnel-next-hop flex-algo** command. This uses the specified flexible algorithm to construct a tunnel toward the indirect static-route next-hop.

The use of this command assumes that the router is participating in the flexible algorithm. This command instructs the router to lookup the indirect next-hop using flexible algorithm tunnels. The static route is not activated if a flexible algorithm-aware tunnel does not exist in the indirect next-hop.

When a router receives an IP packet, the static-route entry may steer toward the indirect next-hop using a flexible algorithm-aware SR tunnel, provided that such a tunnel exists. If the tunnel does not exist, the route is not active and the received IP packet is dropped, as long as no longest prefix match (LPM) route exists.

When the **flex-algo** command is configured, the resolution filter can only use matching flexible algorithm-aware SR tunnels created by flex-algo aware routing protocols (for example, SR IS-IS). If such an entry does not exist in the tunnel-table, the static-route entry does not become active.

Use the commands in the following context to configure static routes using flexible algorithms:

- **MD-CLI**

```
configure router static-routes route indirect tunnel-next-hop flex-algo
```

- **classic CLI**

```
configure router static-route-entry indirect tunnel-next-hop
```

2.2 Aggregate next hop

This feature adds the ability to configure an indirect next-hop for aggregate routes. The indirect next-hop specifies where packets are forwarded if they match the aggregate route, but is not a more-specific route in the IP forwarding table.

2.3 Invalidate next-hop based on ARP/neighbor cache state

This feature invalidates next-hop entries for static routes when the next-hop is no longer reachable on directly connected interfaces. This invalidation is based on ARP and Neighbor Cache state information.

When a next-hop is detected as no longer reachable because of ARP/neighbor cache expiry, the route's next-hop is set as unreachable to prevent the SR from sending continuous ARPs/neighbor solicitations

triggered by traffic destined for the static route prefix. When the next-hop is detected as reachable via ARP or neighbor advertisements, the state of the next-hop is set back to valid.

2.3.1 Invalidate next-hop based on IPv4 ARP

This feature invalidates a static route based on the reachability of the next-hop in the ARP cache when the **validate-next-hop** command is enabled for an IPv4 static route. Use the commands in the following contexts to enable the validate-next-hop feature for an IPv4 static route:

- **MD-CLI**

```
configure router static-routes route next-hop
configure service vprn static-routes route next-hop
```

- **classic CLI**

```
configure router static-route-entry next-hop
configure service vprn static-route-entry next-hop
```

In this case, when the ARP entry for the next-hop is INVALID or not populated, the static route must remain invalid/inactive. When an ARP entry for the next-hop is populated based on a gratuitous ARP received or periodic traffic destined for it and the usual ARP who-has procedure, the static route becomes valid/active and is installed.

2.3.1.1 Invalidate next-hop based on neighbor cache state

This feature invalidates a static route based on the reachability of the next-hop in the neighbor cache when the **validate-next-hop** command is enabled for an IPv6 static route.

Use the commands in the following contexts to enable validate-next-hop for an IPv4 static route:

- **MD-CLI**

```
configure router static-routes route next-hop
configure service vprn static-routes route next-hop
```

- **classic CLI**

```
configure router static-route-entry next-hop
configure service vprn static-route-entry next-hop
```

In this case, when the Neighbor Cache entry for next-hop is INVALID or not populated, the static route must remain invalid/inactive. When an NC entry for next-hop is populated based on a neighbor advertisement received, or periodic traffic destined for it and the usual NS/NA procedure, the static route becomes valid/active and is installed.

2.4 IP interface strip-label behavior

The strip-label feature causes arriving MPLS encapsulated traffic to be stripped of all MPLS labels (up to five) before processing the packet through Policy Based Routing (PBR) filters. Use the following command to configure strip-label.

```
configure router interface strip-label
```

If the packets do not have an IP header immediately following the MPLS label stack after the strip, they are discarded. Only MPLS encapsulated IP, IGP shortcuts, and VPRN over MPLS packets are processed. However, IPv4 and IPv6 packets that arrive without any labels are supported on an interface with strip label enabled.

The **strip-label** command operates in promiscuous mode. The router does not filter on the destination MAC address of the Ethernet frames. In some network designs, multiple ports may be tapped and combined into an interface toward the router. Promiscuous mode allows all of these flows to be processed without requiring the destination MAC address to be updated to match the router address.

To associate an interface that is configured with the **strip-label** command with a port, the port must be configured as single-fiber.

Packets subject to the strip-label action and mirrored (using mirrors or Lawful Intercept) contain the original MPLS labels (and other Layer 2 encapsulation) in the mirrored copy of the packet, as they appeared on the wire when the **mirror-dest** type is the default type “ether”. If the **mirror-dest** type is “ip-only”, the mirrored copy of the packet does not contain the original Layer 2 encapsulation or the stripped MPLS labels.

This command is supported on:

- optical ports for the 7705 SAR Gen 2
- null/dot1q encaps
- network ports
- IPv4
- IPv6

2.5 LDP shortcut for IGP route resolution

This feature enables you to forward user IP packets and specified control IP packets using LDP shortcuts over all network interfaces in the system that participate in the IS-IS and OSPF routing protocols. The default is to disable the LDP shortcut across all interfaces in the system.

Use the following commands to use LDP shortcuts for IGP route resolution:

- **MD-CLI**

```
configure router ldp ldp-shortcut ipv4
configure router ldp ldp-shortcut ipv6
```

- **classic CLI**

```
configure router ldp-shortcut [ipv4][ipv6]
```

2.5.1 IGP route resolution

When LDP shortcut is enabled, LDP populates the RTM with next-hop entries corresponding to all prefixes for which it activated an LDP FEC. For an activated prefix, two route entries are populated in RTM. One corresponds to the LDP shortcut next-hop and has an owner of LDP. The other one is the regular IP next-hop. The LDP shortcut next-hop always has preference over the regular IP next-hop for forwarding user packets and specified control packets over a specific outgoing interface to the route next-hop.

The prior activation of the FEC by LDP is done by performing an exact match with an IGP route prefix in RTM. It can also be done by performing a longest prefix match with an IGP route in RTM if the aggregate-prefix-match command option is enabled globally in LDP.

The LDP next-hop entry is not exported to the LDP control plane or to any other control plane protocols except OSPF, IS-IS, and an OAM control plane specified in Handling of control packets.

This feature is not restricted to /32 IPv4 prefixes or /128 IPv6 FEC prefixes. However, only /32 IPv4 and /128 IPv6 FEC prefixes are populated in the tunnel table for use as a tunnel by services.

All user and specified control packets for which the longest prefix match in RTM yields the FEC prefix are forwarded over the LDP LSP. The following is an example of the resolution process.

Assume that the egress LER advertised a FEC for some /24 prefix using the fec-originate command. At the ingress LER, LDP resolves the FEC by checking in RTM that an exact match exists for this prefix. After the LDP activates the FEC, it programs the NHLFE in the egress datapath and the LDP tunnel information in the ingress datapath tunnel table.

Next, LDP provides the shortcut route to RTM, which associates it with the same /24 prefix. There are two entries for this /24 prefix: the LDP shortcut next-hop and the regular IP next-hop. The latter was used by LDP to validate and activate the FEC. RTM then resolves all user prefixes that succeed a longest prefix match against the /24 route entry to use the LDP LSP.

Now assume that the aggregate-prefix-match was enabled and that LDP found a /16 prefix in RTM to activate the FEC for the /24 FEC prefix. In this case, RTM adds a new, more-specific route entry of /24 and has the next-hop as the LDP LSP. However, RTM does not have a specific /24 IP route entry. RTM then resolves all user prefixes that succeed a longest prefix match against the /24 route entry to use the LDP LSP. All other prefixes that succeed a longest prefix match against the /16 route entry uses the IP next-hop. LDP shortcut also works when using RIP for routing.

2.5.2 LDP-IGP synchronization

See the *7705 SAR Gen 2 MPLS Guide* for information about LDP-IGP synchronization.

2.5.3 LDP shortcut forwarding plane

After the LDP activates an FEC for a prefix and programs RTM, it also programs the ingress tunnel table in IOM or online cards with the LDP tunnel information.

When an IPv4 packet is received on an ingress network interface, a subscriber IES interface, or a regular IES interface, the lookup of the packet by the ingress IOM or line card results in the packet being sent labeled with the label stack corresponding to the NHLFE of the LDP LSP when the preferred RTM entry corresponds to an LDP shortcut.

If the preferred RTM entry corresponds to an IP next-hop, the IPv4 packet is forwarded unlabeled.

The switching from the LDP shortcut next-hop to the regular IP next-hop when the LDP FEC becomes unavailable depends on whether the next-hop is still available. If it is (for example, the LDP FEC was withdrawn because of LDP control plane issues) the switchover should be faster. If the next-hop determination requires IGP to re-converge, this takes longer. However, no target is set.

The switching from a regular IP next-hop to an LDP shortcut next-hop usually occurs only when both are available. However, the programming of the NHLFE by LDP and the programming of the LDP tunnel information in the ingress IOM or line cards tunnel table are asynchronous. If the tunnel table is configured first, it is possible that traffic is black-holed for some time.

2.5.4 ECMP considerations

When ECMP is enabled and multiple equal-cost next-hops exist for the IGP route, the ingress IOM or line card sprays the packets for this route based on the hashing routine currently supported for IPv4 packets.

When the preferred RTM entry corresponds to an LDP shortcut route, spraying is performed across the multiple next-hops for the LDP FEC. The FEC next-hops can either be direct link LDP neighbors or T-LDP neighbors reachable over RSVP LSPs, in the case of LDP-over-RSVP, but not both. This is as per ECMP for LDP.

When the preferred RTM entry corresponds to a regular IP route, spraying is performed across regular IP next-hops for the prefix.

Spraying across regular IP next-hops and LDP-shortcut next-hops concurrently is not supported.

2.5.5 Handling of control packets

All control plane packets do not see the LDP shortcut route entry in RTM with the exception of the following control packets, which are forwarded over an LDP shortcut when enabled:

- A locally generated or in transit ICMP ping and trace route of an IGP route. The transit message appears as a user packet to the ingress LER node.
- A locally generated response to a received ICMP ping or trace route message.

All other control plane packets that require an RTM lookup and knowledge of which destination is reachable over the LDP shortcut continues to be forwarded over the IP next-hop route in RTM.

2.5.6 Handling of multicast packets

Multicast packets cannot be forwarded or received from an LDP LSP. This is because there is no support for the configuration of such an LSP as a tunnel interface in PIM. Only an RSVP P2MP LSP is currently allowed.

If a multicast packet is received over the physical interface, the uRPF check does not resolve to the LDP shortcut because the LDP shortcut route in RTM is not made available to multicast application.

2.5.7 Interaction with BGP route resolution to an LDP FEC

There is no interaction between an LDP shortcut for BGP next-hop resolution and the LDP shortcut for IGP route resolution. BGP continues to resolve a BGP next-hop to an LDP shortcut if the user enabled the

following command option in BGP. The following example shows the configuration to enable the resolution of a BGP next-hop to an LDP shortcut.

Example: MD-CLI

```
[ex:/configure router "Base" bgp next-hop-resolution shortcut-tunnel]
A:admin@node-2# info
    family ipv4 {
        resolution-filter {
            ldp true
        }
    }
```

Example: classic CLI

```
A:node-2>config>router>bgp>next-hop-res>shortcut-tunn# info
----------------------------------------------
                family ipv4
                    resolution-filter
                        ldp
                    exit
                exit
----------------------------------------------
```

2.5.8 Interaction with static route resolution to an LDP FEC

A static route continues to be resolved by searching an LDP LSP whose FEC prefix matches the specified indirect next-hop for the route. In contrast, the LDP shortcut for IGP route resolution uses the LDP LSP as a route. The most specific route for a prefix is selected and, if both a static and IGP routes exist, the RTM route type preference is used to select one.

2.5.9 LDP control plane

For the LDP shortcut to be usable, SR OS must originate a <FEC, label> binding for each IGP route it learns of even if it did not receive a binding from the next-hop for that route. The router must assume that it is an egress LER for the FEC until the route disappears from the routing table or the next-hop advertises a binding for the FEC prefix. In the latter case, SR OS becomes a transit LSR for the FEC.

SR OS originates a <FEC, label> binding for its system interface address only by default. The only way to originate a binding for local interfaces and routes that are not local to the system is by using the fec-originate capability.

Use the **fec-originate** command to generate bindings for all non-local routes for which this node acts as an egress LER for the corresponding LDP FEC. Specifically, this feature must support the FEC origination of IGP learned routes and subscriber/host routes statically configured or dynamically learned over subscriber IES interfaces.

An LDP LSP used as a shortcut by IPv4 packets may also be tunneled using the LDP-over-RSVP feature.

2.6 Weighted load-balancing over interface next-hops

When the **weighted-ecmp** command is configured in the base router context or a VPRN, any IPv4 or IPv6 static, IS-IS, or OSPF route associated with the routing instance can be programmed into the datapath to use weighted load-balancing across the interface next-hops of the route.

Use the following commands to configure weighted ECMP in the base router context or in a VPRN.

```
configure router weighted-ecmp
configure service vprn weighted-ecmp
```

In order for weighted ECMP to be supported across the interface next-hops of an IS-IS or OSPF route the following conditions must be met.

- All of the calculated ECMP next-hops must be interface next-hops.
- All of the calculated ECMP next-hop interfaces must have a non-zero load-balancing-weight value configured in the following context. Use the commands in the following context to configure a non-zero load-balancing-weight value.

  ```
  configure router isis interface
  ```

  By default, IS-IS or OSPF interfaces have a zero weight (no load-balancing-weight); non-zero values must be configured explicitly. Values cannot be auto-derived.

In order for weighted ECMP to be supported across the interface next-hops of a static route the following conditions must be met.

- All of the configured ECMP next-hops must be direct next-hops (resolved to an interface). The ECMP next-hops are the next-hops with the lowest preference that also have the lowest metric.
- All of the configured ECMP next-hop interfaces must have a non-zero load-balancing-weight value configured in the following context. Use the commands in the following context to configure a non-zero load-balancing-weight value:
  - **MD-CLI**

    ```
    configure router static-routes route next-hop
    ```

  - **classic CLI**

    ```
    configure router static-route-entry next-hop
    ```

  By default, static route next-hops have a zero weight (no load-balancing-weight); non-zero values must be configured explicitly. Values cannot be auto-derived. The ECMP next-hops are the next-hops with the lowest preference that also have the lowest metric.

The **load-balancing-weight** commands in the IS-IS or OSPF and static route configuration trees accept a value between 0 and 4294967295.

If an IPv4 or IPv6 BGP route has a BGP next-hop resolved by a static, IS-IS, or OSPF ECMP route and **ibgp-multipath** is configured under BGP, traffic forwarded to the BGP next-hop is sprayed according to the load-balancing-weights of the interface next-hops.

2.7 IP FRR for static route entry

IP Fast ReRoute (FRR) is supported when the **backup-next-hop** command is configured for a static route entry. IP FRR support uses 1+1 protection by using a single backup next-hop address when the single primary next-hop fails. Only 1+1 protection is supported during backup without ECMP capability. Next-hop forwarding information for the backup next-hop address from the IP Routing Table Manager (RTM) is used to install a pre-resolved IP or tunneled fast reroute backup path to the backup next-hop. The configured backup next-hop IP address can be directly or indirectly connected through an IGP, a BGP, or a tunnel. The backup next-hop must be of the same IP address family as the primary next-hop (for example, an IPv4 primary next-hop can be protected using an IPv4 backup next-hop).

Note: FRR for static route entries is only supported for IP traffic on FP-based platforms.

IP FRR for static route is supported in the base router and service VPRN contexts.

If the primary next-hop of the static route entry fails and the IP FRR backup next-hop is activated, then the backup tag is applied to the static route and the configured preference and metric for the primary hop is inherited. If the primary next-hop is activated again, then make-before-break functionality is used to avoid any packet loss.

The following example shows the IP FRR configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    static-routes {
        route 10.10.0.0/16 route-type unicast {
            tag 20
            backup-tag 100
            next-hop "101.1.1.1" {
                preference 100
                backup-next-hop {
                    address 50.1.1.2
                }
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#--------------------------------------------------
"Static Route Configuration"
#--------------------------------------------------
        static-route-entry 10.10.0.0/16
            tag 20
            backup-tag 100
            next-hop 101.1.1.1
                preference 100
                backup-next-hop
                    address 50.1.1.2
                exit
            exit
        exit
```

The logic behavior applied to the associated tag of the static route entry is summarized in the following table.

Table 3: Static route tag for IP FRR configuration

Primary NH	Backup NH	StaticRoute State	StaticRoute Tag
UP	UP	UP	20[1]
UP	DOWN	UP	20[1]
DOWN	UP	UP	100[1]
DOWN	DOWN	DOWN	—

IGP export policies can use the tag and the backup-tag as match criteria when exporting a static route entry using route policies. The export policies may introduce unique export properties for each tag (for example, resulting in different IGP metrics) and may make an exported route more or less desirable when the primary next-hop fails and the backup next-hop is activated.

The following limitations apply in the IP FRR for static route entries.

- Only the primary next-hop has IP FRR support. The backup next-hop has no IP FRR support if it suddenly becomes unreachable.
- If multiple next-hops are configured with a backup for a static route entry, then IP FRR is activated if there is only one remaining primary next-hop active. If multiple primary next-hops can be activated, then the static route entry uses ECMP and the backup next-hop IP FRR functionality is not used.
- If the primary next-hop fails and the backup next-hop is used as the primary hop, then the backup next-hop uses the configured backup tag (or 0, if not configured) and inherits the configured preference and metric of the primary next-hop (or the default values, if not configured).
- The backup inherits the preference and the metric of the primary next-hop, however, it does not support any of the features configured on the primary next-hop (for example, BFD, CPE check, LDP sync, and so on) even when the backup becomes the active next-hop.
- If the primary next-hop of a static route entry, configured with a backup next-hop, is held down because hold-down is configured on static routes, the backup next-hop is also held down and is not used for traffic, even in cases where the backup-next-hop can be activated.
- The following tunnel types are supported:
  - OSPF or ISIS shortcuts using RSVP-TE and SR-TE
  - BGP VPN-v4/v6 or BGP shortcut routes over LDP, RSVP, SR-ISIS, SR-OSPF, LDPoRSVP, SR-TE, GREv4, SR policy, MPLS forward policy, and RIB API
  - backup-next-hop recursion through indirect next-hop static-route-entry with resolution filter for LDP, RSVP, LDPoRSVP, SR-TE, SR-ISIS, SR-OSPF, SR policy, MPLS forward policy, or RIB API
- LDP-FRR using a static-route is not mutually supported in combination with static-route backup-next-hop for the same static route.
- Any other backup-next-hop types are considered as non-supported. For example:
  - Locally aggregated BGP routes

[1] The tag value is based on the preceding IP FRR example configuration provided.

- BGP routes when the BGP next-hop is recursively resolved through another BGP route
- 6over4 tunnel
- GREv6 tunnel
- OSPF or IS-IS shortcuts using LDP, SR-ISIS, SR-OSPF, and LDPoRSVP (generic IGP shortcut limitation not only for backup-next-hop)
- OSPF or IS-IS shortcuts over SR policy, MPLS forward policy, and RIB API (generic IGP shortcut limitation not only for backup-next-hop)
- BGP-LU over LDP, RSVP, LDPoRSVP, SR-TE, SR-OSPF, SR-ISIS, SR policy, MPLS forward policy and RIB API
- 4PE
- 6PE

2.8 Router interface encryption with NGE

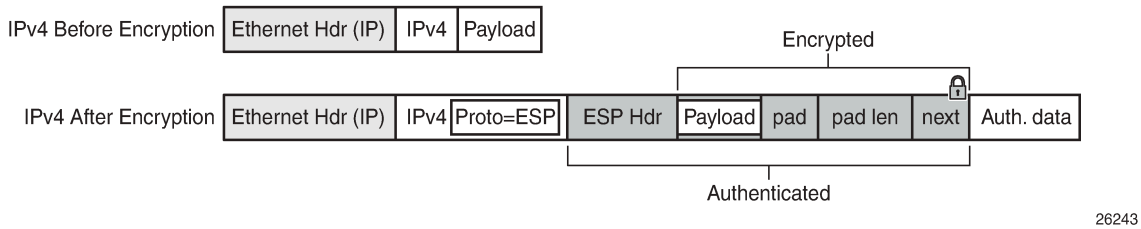
NGE nodes support Layer 3 encryption on router interfaces for IPv4 traffic. NGE is not supported on dual-stack IPv4/IPv6 or IPv6-only interfaces. See the *7705 SAR Gen 2 Services Overview Guide* for more information about platforms that support NGE.

NGE is enabled on a router interface by configuring the **group-encryption** command on the router interface. The interface is considered part of the NGE domain, and any received packets that are NGE-encrypted are decrypted if the key group is configured on the node. To encrypt packets egressing the interface, the outbound key group must be configured on the interface. All IP packets, such as self-generated traffic or packets forwarded from router interfaces that are not inside the NGE domain, are encrypted when egressing the interface. There are some exceptions to this general behavior, as described in the sections that follow; for example, GRE-MPLS and MPLSoUDP packets are not encrypted when router interface encryption is enabled.

The outbound and inbound key groups configured on the router interface determine which keys are used to encrypt and decrypt traffic. See the *7705 SAR Gen 2 Services Overview Guide* for more information about configuring key groups.

To perform encryption, router interface encryption reuses the IPsec transport mode packet format as shown in Figure 14: Router Interface Encryption Packet Format (IPsec Transport Mode).

Figure 14: Router Interface Encryption Packet Format (IPsec Transport Mode)



The protocol field in the IP header of an NGE packet is always set to "ESP". Within an NGE domain, the SPI that is included in the ESP header is always an SPI for the key group configured on the router interface. Other fields in the IP header, such as the source and destination addresses, are not altered by

NGE router interface encryption. Packets are routed through the NGE domain and decrypted when the packet leaves the NGE domain.

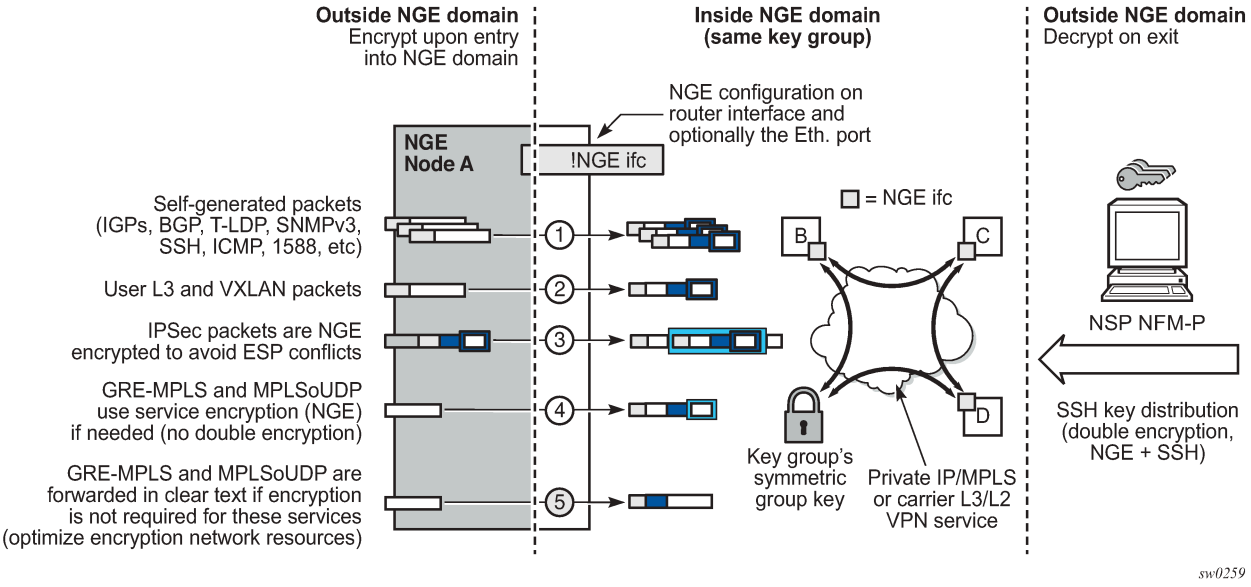
The group keys used on an NGE-enabled router interface provide encryption of broadcast and multicast packets within the GRT. For example, OSPF uses a broadcast address to establish adjacencies, which can be encrypted by NGE without the need to establish point-to-point encryption tunnels. Similarly, multicast packets are also encrypted without point-to-point encryption tunnels.

2.8.1 NGE domains

An NGE domain is a group of nodes and router interfaces forming a network that uses a single key group to create a security domain. NGE domains are created when router interface encryption is enabled on router interfaces that need to participate in the NGE domain. The NSP NFM-P assists users in managing the nodes and interfaces that participate in the NGE domain. See the *NSP NFM-P User Guide* for more information.

Figure 15: NGE Domain Transit shows various traffic types crossing an NGE domain.

Figure 15: NGE Domain Transit



In Figure 15: NGE Domain Transit, nodes A, B, C, and D have router interfaces configured with router interface encryption enabled. Traffic is encrypted when entering the NGE domain using the key group configured on the router interface and is decrypted when exiting the NGE domain. Traffic may traverse multiple hops before exiting the NGE domain, yet decryption only occurs on the final node when the traffic exits the NGE domain.

Various traffic types are supported and encrypted when entering the NGE domain, as illustrated by the following items on node A in Figure 15: NGE Domain Transit:

- Item 1: Self-generated packets

  These packets, which include all types of control plane and management packets such as OSPF, BGP, LDP, SNMPv3, SSH, ICMP, RSVP-TE, and 1588, are encrypted.

- Item 2: User Layer 3 and VXLAN packets

Any Layer 3 user packets that are routed into the NGE domain from an interface outside the NGE domain are encrypted. Any VXLAN packets that are routed into the NGE domain from this NGE node are encrypted.

- Item 3: IPsec packets

  IPsec packets are NGE-encrypted when entering the NGE domain to ensure that the IPsec packets' security association information does not conflict with the NGE domain.

GRE-MPLS- or MPLSoUDP-based service traffic consists of Layer 3 packets, and router interface NGE is not applied to these types of packets. Instead, service-level NGE is used for encryption to avoid double-encrypting these packets and impacting throughput and latencies. The two types of GRE-MPLS or MPLSoUDP packets that can enter the NGE domain are illustrated by items 4 and 5 in Figure 15: NGE Domain Transit.

- Item 4: GRE-MPLS and MPLSoUDP packets (SDP or VPRN) with service-level NGE enabled

  These encrypted packets use the key group that is configured on the service. The services key group may be different from the key group configured on the router interface where the GRE-MPLS or MPLSoUDP packet enters the NGE domain.

- Item 5: GRE-MPLS and MPLSoUDP packets (SDP or VPRN) with NGE disabled

  These packets are not encrypted and can traverse the NGE domain in clear text. If these packets require encryption, SDP or VPRN encryption must be enabled.

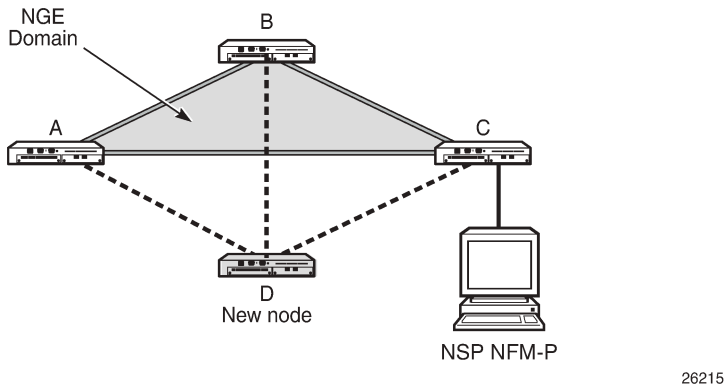
Creating an NGE domain from the NSP NFM-P requires the user to determine the type of NGE domain being managed. This indicates whether NGE gateway nodes are required to manage the NGE domain, and other operational considerations. The two types of NGE domains are:

- Private IP/MPLS network NGE domain
- Private over intermediary network NGE domain

2.8.1.1 Private IP/MPLS network NGE domain

One type of NGE domain is a private IP/MPLS network, as shown in Figure 16: Private IP/MPLS network NGE domain.

Figure 16: Private IP/MPLS network NGE domain



In a private IP/MPLS network NGE domain, all interfaces are owned by the user and there is no intermediary service provider needed to interconnect nodes. Each interface is a point-to-point private link

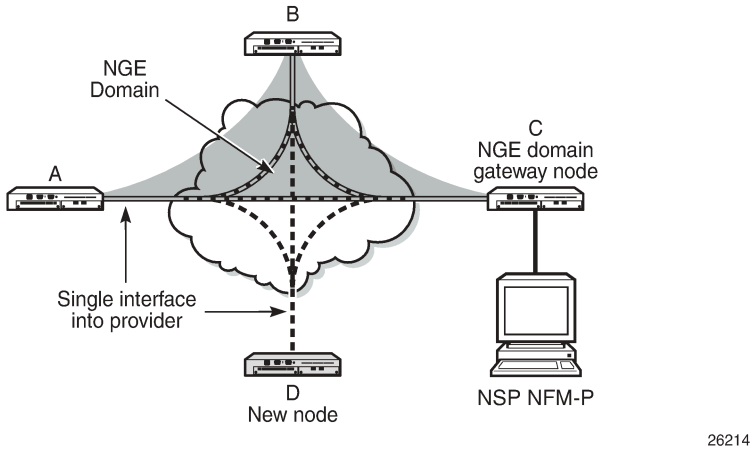
between private nodes. When a new node is added to this type of NGE domain (node D in Figure 16: Private IP/MPLS network NGE domain), the links that connect node D to the existing nodes in the NGE domain (nodes A, B, and C) must be enabled with NGE router interface encryption. Links from the new node to the existing nodes are enabled one at a time. The NSP NFM-P provides tools that simplify adding nodes to the NGE domain and enabling NGE on their associated interfaces. In this type of NGE domain, each interface is a direct link between two nodes and is not used to communicate with multiple nodes over a broadcast medium offered by an intermediary network. Also, there are no NGE gateway nodes required between the NSP NFM-P and new nodes entering the NGE domain.

2.8.1.2 Private over intermediary network NGE domain

The other type of NGE domain is a private IP/MPLS network that traverses an intermediary network NGE domain; the intermediary network is used to interconnect nodes in the NGE domain using a multipoint-to-multipoint service. The intermediary network is typically a service provider network that provides a private IP VPN service or a private VPLS service used to interconnect a private network that does not mimic point-to-point links as described in the Private IP/MPLS network NGE domain section.

This type of NGE domain is shown in Figure 17: Private over intermediary network NGE domain.

Figure 17: Private over intermediary network NGE domain



Private over intermediary network NGE domains have nodes with links that connect to a service provider network where a single link can communicate with multiple nodes over a Layer 3 service such as a VPRN. In Figure 17: Private over intermediary network NGE domain, node A has NGE enabled on its interface with the service provider and uses that single interface to communicate with nodes B and C, and eventually with node D when node D has been added to the NGE domain. This type of NGE domain requires the recognition of NGE gateway nodes that allow the NSP NFM-P to reach new nodes that enter the domain. Node C is designated as a gateway node.

When node D is added to the NGE domain, it must first have the NGE domain key group downloaded to it from the NSP NFM-P. The NSP NFM-P creates an NGE exception ACL on the gateway node, C, to allow communication with node D using SNMPv3 and SSH through the NGE domain. After the key group is downloaded, the NSP NFM-P enables router interface encryption on node D's interface with the service provider and node D is now able to participate in the NGE domain. The NSP NFM-P automatically removes the IP exception ACL from node C when node D enters the NGE domain.

See Router interface NGE domain concepts for more information.

2.8.2 Router interface NGE domain concepts

An NGE domain is a group of nodes whose router interfaces in the base routing context (GRT) are enabled for router interface NGE. An interface without router interface NGE enabled is considered to be outside the NGE domain. NGE domains use only one key group when the domain is created; however, two key groups may be active at when if some links within the NGE domain are in transition from one key group to the other.

Figure 18: Inside and outside NGE domains illustrates the NGE domain concept. Table 4: Inside and outside NGE domains configuration scenarios describes the three configuration scenarios inside the NGE domain.

Figure 18: Inside and outside NGE domains

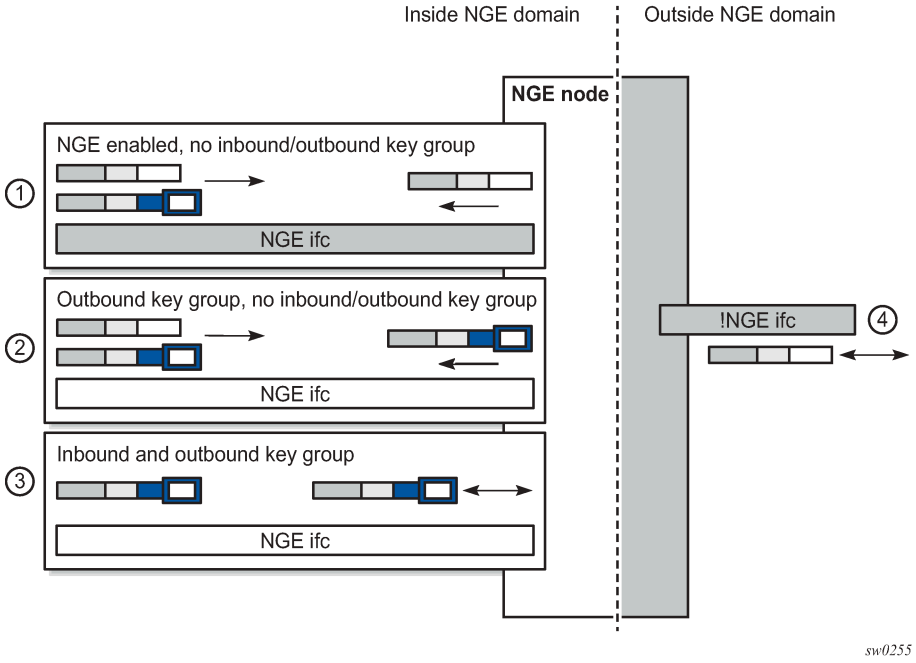


Table 4: Inside and outside NGE domains configuration scenarios

Key	Description
1	NGE enabled, no inbound/outbound key group Outbound packets are sent without encrypting; inbound packets can be NGE-encrypted or clear text
2	Outbound key group, no inbound key group Outbound packets are encrypted using the interface key group if not already encrypted; inbound packets can be NGE-encrypted or clear text
3	Inbound and outbound key group

Key	Description
	Outbound packets are encrypted using the interface key group if not already encrypted; inbound packets must be encrypted using the interface key group keys
4	Outside the NGE domain, the interface is not configured for NGE; any ESP packets are IPsec packets

A router interface is considered to be inside the NGE domain when it has been configured with **group-encryption** on the interface. When **group-encryption** is configured on the interface, the router can receive unencrypted packets or NGE-encrypted packets from any configured key group on the router, but any other type of IPsec-formatted packet is not allowed. If an IPsec-formatted packet is received on an interface that has **group-encryption** enabled, it does not pass NGE authentication and is dropped. Therefore, IPsec packets cannot exist within the NGE domain without first being converted to NGE packets. This conversion requirement delineates the boundary of the NGE domain and other IPsec services.

When NGE router interface encryption is enabled and only an outbound key group is configured, the interface can receive unencrypted packets or NGE-encrypted packets from any configured key group on the router. All outbound packets are encrypted using the outbound key group if the packet was not already encrypted further upstream in the network.

When NGE router interface encryption has been configured with both an inbound and outbound key group, only NGE packets encrypted with the key group security association can be sent and received over the interface.

When there is no NGE router interface encryption, the interface is considered outside the NGE domain where NGE is not applied.

See the "NGE Packet Overhead and MTU Considerations" section in the *7705 SAR Gen 2 Services Overview Guide* for MTU information related to enabling NGE on a router interface.

2.8.3 GRE-MPLS and MPLSoUDP packets inside the NGE domain

NGE router interface encryption is never applied to GRE-MPLS or MPLSoUDP packets, for example:

- GRE with the GRE protocol ID set to MPLS Unicast (0x8847) or Multicast (0x8848)
- UDP packets with destination port = 6635)

GRE-MPLS and MPLSoUDP packets that enter the NGE domain or transit the NGE domain are forwarded as is.

Because these GRE-MPLS and MPLSoUDP packets provide transport for MPLS-based services, they already use the NGE services-based encryption techniques for MPLS, such as SDP or VPRN-based encryption. To avoid double encryption, the packets are left in cleartext when entering an NGE domain or crossing intermediate nodes in the NGE domain, and are forwarded as needed when exiting an NGE domain.

2.8.4 EVPN-VXLAN tunnels and services

NGE router interface encryption does not differentiate between EVPN-VXLAN tunnels and other L3 traffic, and therefore encrypts all EVPN-VXLAN traffic that egresses the node.

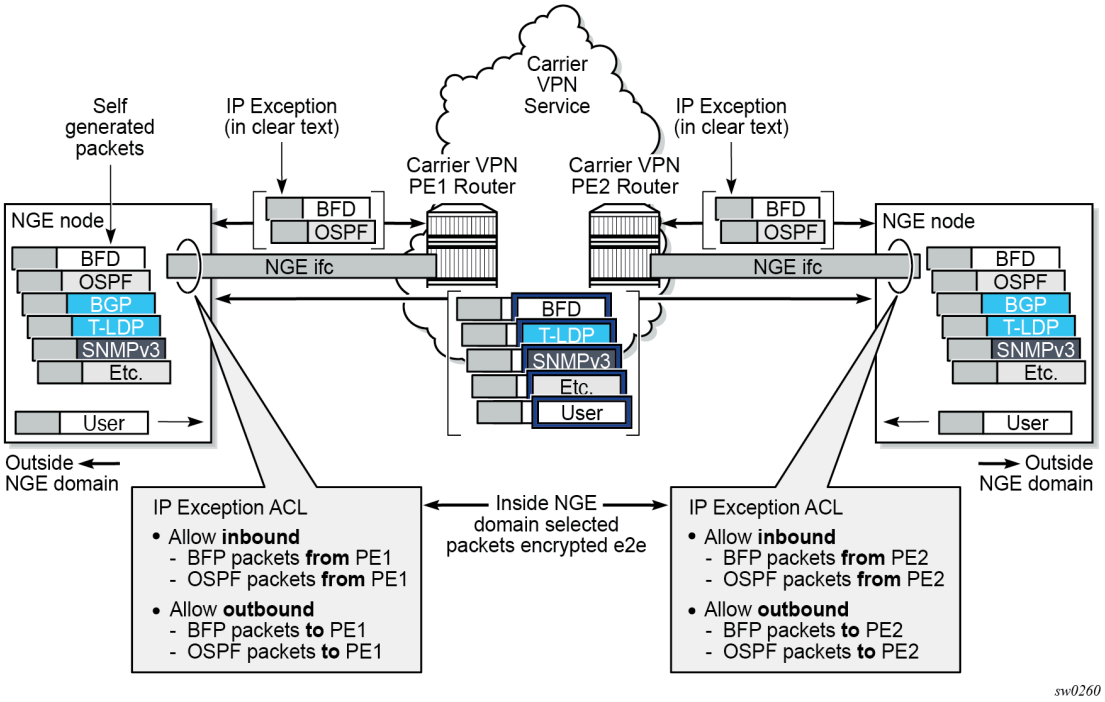
For received encrypted EVPN-VXLAN packets, if the VXLAN tunnel terminates on the node (that is, the destination IP is for a VTEP on this node), then the NGE packet is decrypted and the EVPN-VXLAN traffic is processed as if NGE encryption never took place.

2.8.5 Router encryption exceptions using ACLs

In some cases, Layer 3 packets may need to cross the NGE domain in clear text, such as when an NGE-enabled router needs to peer with a non-NGE-capable router to exchange routing information. This can be accomplished by using a router interface NGE exception filter applied on the router interface for the required direction, inbound or outbound.

Figure 19: Router interface NGE exception filter example shows the use of a router interface NGE exception filter.

Figure 19: Router interface NGE exception filter example



The inbound or outbound exception filter is used to allow specific packet flows through the NGE domain in clear text, where there is an explicit inbound and outbound key group configured on the interface. The behavior of the exception filter for each router interface configuration is as follows:

- NGE enabled, no inbound or outbound key group

  In this scenario, the router does not encrypt outbound traffic, and so the outbound exception filter is not applied. The router can still receive inbound NGE packets, so the exception filter is applied to inbound packets. If the filter detects a match, clear text packets can be received and forwarded by the router.

- Outbound key group, no inbound key group

  The outbound exception filter is applied to outbound traffic, and packets that match the filter are not encrypted on egress. The router can receive inbound NGE packets without an inbound key group set

and applies the exception filter to inbound packets. If the filter detects a match, clear text packets can be received and forwarded by the router.

- Inbound and outbound key group

  The inbound and outbound exception filters are applied, and any packets that match are passed in clear text.

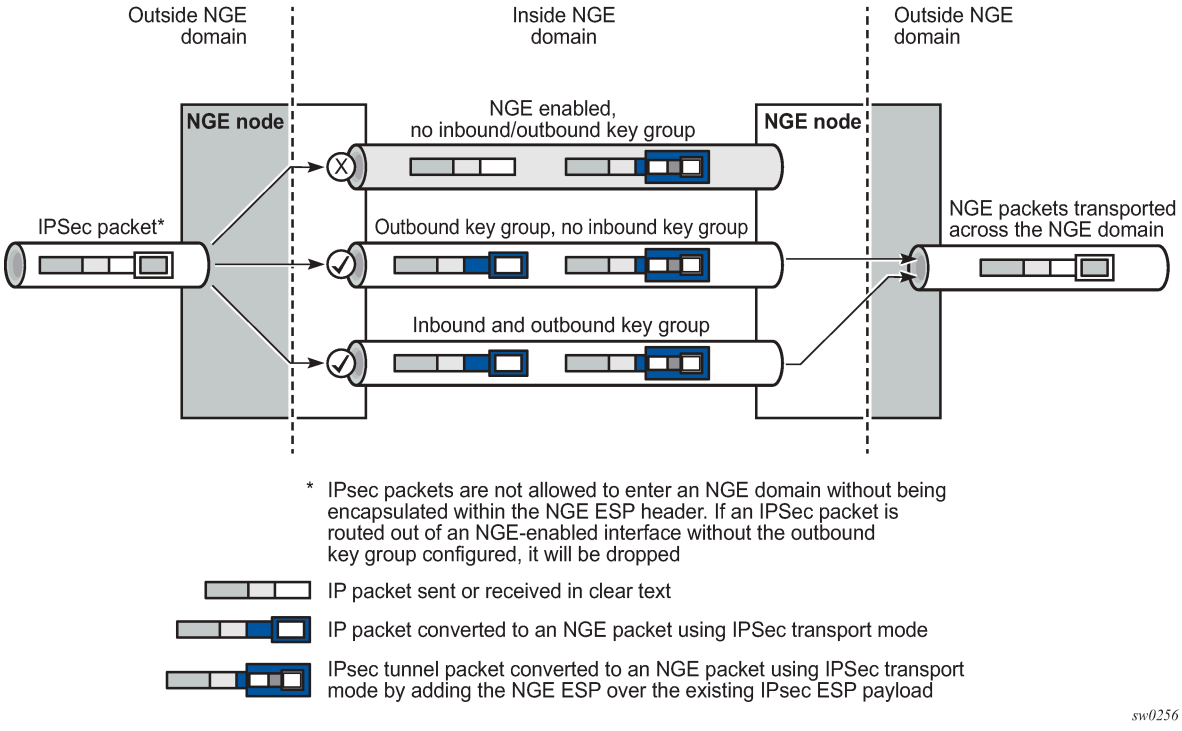
2.8.6 IPsec packets crossing an NGE domain

IPsec packets can cross the NGE domain because they are still considered Layer 3 packets. To avoid confusion between the security association used in an IPsec packet and the one used in a router interface NGE packet, the router always applies NGE to any IPsec packet that traverses the NGE domain.

IPsec packets that originate from a router within the NGE domain are not allowed to enter the NGE domain. The only exception to this restriction is OSPFv3 packets.

Figure 20: IPsec packets transiting an NGE domain shows how IPsec packets can transit an NGE domain.

Figure 20: IPsec packets transiting an NGE domain



An IPsec packet enters the router from outside the NGE domain. When the router determines that the egress interface to route the packet is inside an NGE domain, it selects an NGE router interface with one of the following configurations.

- NGE enabled with no inbound or outbound key group configured

  This link cannot forward the IPsec packet without adding the NGE ESP, but because nothing is configured for the outbound key group, the packet must be dropped.

- NGE enabled with outbound key group configured and no inbound key group configured — the packet originates outside the NGE domain, so the router adds an ESP header over the existing ESP and encrypts the payload using the NGE domain keys for the configured outbound key group.
- NGE enabled with both inbound and outbound key groups configured — the packet originates outside the NGE domain, so the router adds an ESP header over the existing ESP and encrypts the payload using the NGE domain keys for the configured outbound key group.

OSPFv3 IPsec support also uses IPsec transport mode packets. These packets originate from the CPM, which is considered outside the NGE domain; however, the above rules for encapsulating the packets with an NGE ESP apply and allow these packets to successfully transit the NGE domain.

2.8.7 Multicast packets traversing the NGE domain

Multicast packets that traverse an NGE domain can be categorized into two main scenarios:

- Scenario 1

  Multicast packets that ingress the router on an interface that is outside the NGE domain. These packets can egress a variety of interfaces that are either inside or outside the NGE domain.

- Scenario 2

  Multicast packets that ingress the router on an interface that is inside the NGE domain. These packets can egress a variety of interfaces that are either inside or outside the NGE domain. This scenario has two cases:

  – Scenario 2a

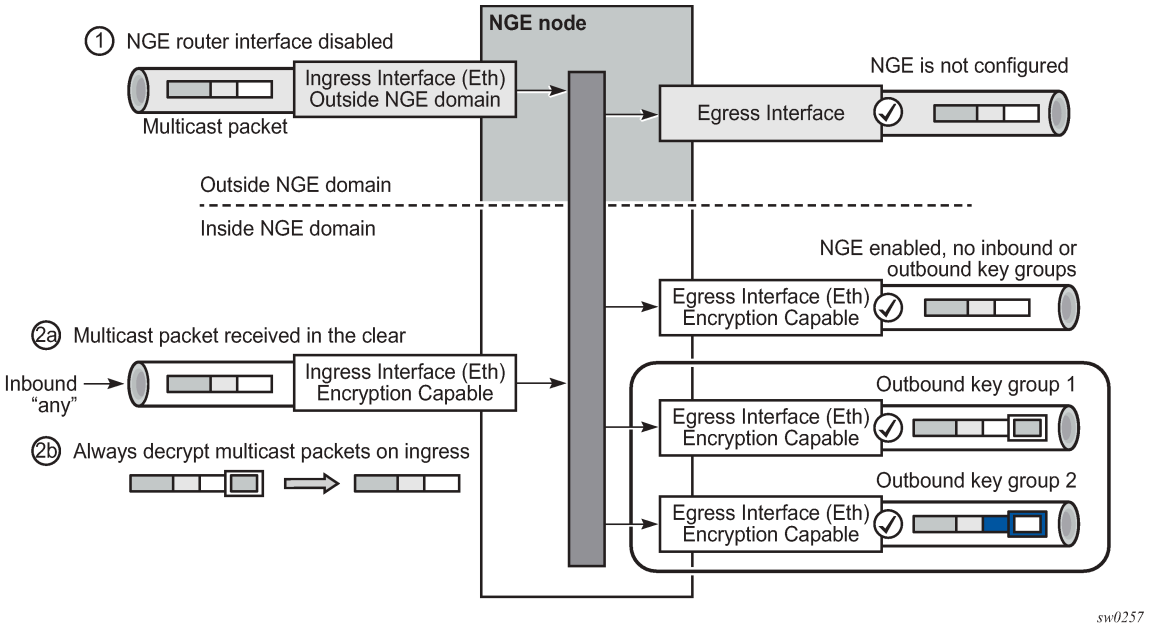
    The ingress multicast packet is not yet NGE-encrypted.

  – Scenario 2b

    The ingress multicast packet is NGE-encrypted.

Figure 21: Processing multicast packets shows these scenarios.

Figure 21: Processing multicast packets



Multicast packets received from outside the NGE domain (Scenario 1) are processed similarly to multicast packets received from inside the NGE domain (Scenarios 2a and 2b).

The processing rule is that multicast packets are always forwarded as clear text over the fabric. This means that for Scenario 2b, when a multicast packet is received on an encryption-capable interface and is NGE-encrypted, the packet is always decrypted first so that it can be processed in the same way as packets in Scenarios 1 and 2a.

On egress, the following scenarios apply:

- Egressing an interface outside the NGE domain

  Packets are processed in the same way as any multicast packets forwarded out a non-NGE interface.

- Egressing an NGE router interface and no inbound or outbound key group is configured

  The router forwards these packets out from the egress interface without encrypting them because there is no outbound key group configured. This behavior also applies to unicast packets in the same scenario.

- egressing an NGE router interface with the outbound key group configured — the router encrypts the multicast packet using the SPI keys of the outgoing SA configured in the key group. This behavior also applies to unicast packets in the same scenario.

2.8.8 Assigning key groups to router interfaces

Prerequisites

Assigning key groups to router interfaces involves the following three steps:

Step1 is required so that the router can initialize and differentiate the interface for NGE traffic before accepting or sending NGE packets. This assigns the interface to an NGE domain.

Assigning key groups to a router interface in steps 2 and 3 is similar to assigning key groups to SDPs or VPRN-based services. An outbound key group cannot be configured for a router interface without first enabling the **group-encryption** command.

When group-encryption is enabled and no inbound key group is configured, the router accepts NGE Layer 3 packets that were encrypted using keys from any security association configured in any key group on the system. If the packet specifies a security association that is not configured in any key group on the node, the packet is dropped.

The outbound key group references the key group to use when traffic egresses the router on the router interface. The inbound key group is used to make sure ingress traffic is using the correct key group on the router interface. If ingress traffic is not using the correct key group, the router counts these packets as errors.

Procedure

Step 1. Enable NGE with the **group-encryption** command.

Step 2. Configure the outbound key group.

Step 3. Configure the inbound key group.

2.8.9 NGE and BFD support

When NGE is enabled on a router interface, BFD packets that originate from the network processor on the adapter card or from the system are encrypted in the same way as BFD packets that are generated by the CPM.

2.8.10 NGE and ACL interactions

When NGE is enabled on a router interface, the ACL function is applied as follows:

- **on ingress**

  Normal ACLs are applied to traffic received on the interface that could be either NGE-encrypted or clear text. For NGE-encrypted packets, this implies that only the source, destination, and IP options are available to filter on ingress, as the protocol is ESP, and the packet is encrypted. If an IP exception ACL is also configured on the interface, the IP exception ACL is applied first to allow any clear text packets to ingress as needed. After the IP exception ACL is applied and if another filter or ACL is configured on the interface, the other filter processes the remaining packet stream (NGE-encrypted and IP exception ACL packets), and other ACL functions such as PBR or Layer 4 information filtering could be applied to any clear text packets that passed the exception ACL.

- **on egress**

  ACLs are applied to packets before they are NGE-encrypted as per normal operation without NGE enabled.

2.8.11 Router interface NGE and ICMP interactions over the NGE domain

Typically, ICMP works as expected over an NGE domain when all routers participating in the NGE domain are NGE-capable; this includes running an NGE domain over a private IP/MPLS network. When an ICMP

message is required, the NGE packet is decrypted first, and the original packet is restored to create a detailed ICMP message using the original packet's header information.

When the NGE domain crosses a Layer 3 service provider, or crosses over routers that are not NGE-aware, it is not possible to create a detailed ICMP message using the original packet's information, as the NGE packet protocol is always set to ESP. Furthermore, the NGE router that receives these ICMP messages drops them because the messages are not NGE-encrypted.

The combination of dropping ICMP messages at the NGE border node and the missing unencrypted packet details in the ICMP information can cause problems with diagnosing network issues.

To help with diagnosing network issues, additional statistics are available on the interface to show whether ICMP messages are being returned from a foreign node. The following statistics are included in the group encryption NGE statistics for an interface:

- Group Enc Rx ICMP DestUnRch Pkts
- Group Enc Rx ICMP TimeExc Pkts
- Group Enc Rx ICMP Other Pkts

These statistics are used when clear text ICMP messages are received on an NGE router interface. The Invalid ESP statistics are not used in this situation even though the packet does not have a correct NGE ESP header. If there is no ingress exception ACL configured on the interface to allow the ICMP messages to be forwarded, the messages are counted and dropped.

If more information is required for these ICMP messages, such as source or destination address information, a second ICMP filter can be configured on the interface to allow logging of the ICMP messages. If the original packet information is also required, an egress exception ACL can be configured with the respective source or destination address information, or other criteria, to allow the original packet to enter the NGE domain in clear text and determine which flows are causing the ICMP failures.

2.8.12 1588v2 encryption with NGE

If a router interface is enabled for encryption and Layer 3 1588v2 packets are sent, they are encrypted using NGE. This means that if port timestamping is enabled on a router interface with NGE, the port timestamp is applied to the Layer 3 1588v2 packet using software-based timestamping instead of hardware-based timestamping, and consequently, timing accuracy may degrade. The exact level of timing or synchronization degradation is dependent on many factors, and testing is recommended to measure any impact.

If there is a need to support Layer 3 1588v2 with better accuracy for frequency or better time using port timestamping, an NGE exception ACL is required to keep the Layer 3 1588v2 packets in clear text. The exception ACL must enable UDP packets with destination port 319 to be sent in clear text.

2.9 Process overview

The following items are components to configure basic router command options:

- **interface**

  A logical IP routing interface. When created, attributes like an IP address, port, link aggregation group, or the system can be associated with the IP interface.

- **address**

The address associates the device system name with the IP system address. An IP address must be assigned to each IP interface.

- **system interface**

  This creates an association between the logical IP interface and the system (loopback) address. The system interface address is the circuitless address (loopback) and is used by default as the router ID for protocols such as OSPF and BGP.

- **router ID**

  (Optional) The router ID specifies the router IP address.

- **autonomous system**

  (Optional) AS is a collection of networks that are subdivided into smaller, more manageable areas.

- **confederation**

  (Optional) This option creates confederation-autonomous systems within an AS to reduce the number of iBGP sessions required within an AS.

2.10 Configuration notes

The following information describes router configuration requirements:

- A system interface and associated IP address must be specified.
- Boot options file (BOF) options must be configured before configuring router command options.
- Confederations can be configured before protocol connections (such as BGP) and peering command options are configured.

2.11 Configuring an IP router with CLI

This following sections provide information to configure an IP router.

2.11.1 IP router configuration overview

About this task

In a Nokia router, an interface is a logical named entity. An interface is created by specifying an interface name under the **configure router** context. This is the global router configuration context where objects like static routes are defined. An IP interface name can be up to 32 alphanumeric characters, must start with a letter, and is case-sensitive; for example, the interface name "1.1.1.1" is not allowed, but "int-1.1.1.1" is allowed.

If the interface name already exists, the router changes the context to maintain that IP interface. If the interface name already exists within another service ID or is an IP interface defined within the **configure router** commands, an error occurs and the context does not change to that IP interface.

To create an interface, the following basic configuration tasks must be performed:

Procedure

Step 1. Assign a name to the interface.

Step 2. Associate an IP address with the interface.

Step 3. Associate the interface with a network interface or the system interface.

Step 4. Configure the appropriate routing protocols.

Expected outcome

A system interface and network interface are configured.

2.11.1.1 System interface

The system interface is associated with a network entity (such as a specific Nokia router), not a specific interface. The system interface is also referred to as the loopback address. The system interface is associated during the configuration of the following entities:

- the termination point of service tunnels
- the hops when configuring MPLS paths and LSPs
- the addresses on a target router for BGP and LDP peering

The system interface is used to preserve connectivity (when routing reconvergence is possible) when an interface fails or is removed. The system interface is used as the router identifier. A system interface must have an IP address with a 32-bit subnet mask.

2.11.1.2 Network interface

A network interface can be configured on one of the following entities:

- physical or logical port
- SONET/SDH channel

2.11.2 Basic configuration

See each specific chapter for specific routing protocol information and command syntax to configure protocols such as IS-IS and BGP.

The most basic router configuration must have the following:

- system name
- system address

The following example shows the 7705 SAR Gen 2 router configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    autonomous-system 100
    router-id 10.10.10.103
    confederation {
        confed-as-num 1000
```

```
        members 100 { }
        members 200 { }
        members 300 { }
    }
...
    interface "system" {
        ipv4 {
            primary {
                address 10.10.10.103
                prefix-length 32
            }
        }
    }
    interface "to-104" {
        port 1/1/1
        ipv4 {
            primary {
                address 10.0.0.103
                prefix-length 24
            }
        }
    }
...
    isis 0 {
        loopfree-alternate {
        }
    }
```

Example: classic CLI

```
A:node-2>config# info
. . .
#------------------------------------------
# Router Configuration
#------------------------------------------
    router
        interface "system"
            address 10.10.10.103/32
        exit
        interface "to-104"
            address 10.0.0.103/24
            port 1/1/1
            exit
        exit
        autonomous-system 100
        confederation 1000 members 100 200 300
   router-id 10.10.10.103
...
    exit
    isis
    exit
...
#------------------------------------------
```

2.11.3 Common configuration tasks

This section describes the basic system configuration tasks.

2.11.3.1 Configuring a system name

Use the **system** command to configure a name for the device. The name is used in the prompt string. Only one system name can be configured. If multiple system names are configured, the last one configured overwrites the previous entry.

If special characters are included in the system name string (for example, spaces, "#", or "?"), the entire string must be enclosed in double quotes. Use the following syntax to configure the system name.

Example: MD-CLI

```
[ex:/configure system]
A:admin@node-2# info
    name "node-2"
    location "Mt.View, CA, NE corner of FERG 1 Building"
    coordinates "37.390, -122.05500 degrees lat."
```

Example: classic CLI

```
A:node-2>config>system# info
#------------------------------------------
# System Configuration
#------------------------------------------
        name "node-2"
        location "Mt.View, CA, NE corner of FERG 1 Building"
        coordinates "37.390, -122.05500 degrees lat."
```

2.11.3.2 Configuring interfaces

The following command sequences create a system and a logical IP interface. The system interface assigns an IP address to the interface, then associates the IP interface with a physical port. The logical interface can associate attributes like an IP address or port.

The system interface cannot be deleted.

2.11.3.2.1 Configuring a system interface

Use the following command to configure a system interface.

```
configure router interface
```

2.11.3.2.2 Configuring a network interface

Use the commands in the following context to configure a network interface.

```
configure router interface
```

The following example shows network interface configuration information.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
interface "system" {
        ipv4 {
            primary {
                address 10.10.0.4/32
                prefix-length 32
            }
        }
     }
interface "to-ALA-2" {
        port 1/1/1
        egress {
            filter {
                ip "10"
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
        interface "system"
            address 10.10.0.4/32
        exit
        interface "to-ALA-2"
            address 10.10.24.4/24
            port 1/1/1
            egress
                filter ip 10
            exit
        exit
...
#------------------------------------------
```

Use the following command to enable CPU protection.

```
configure router interface cpu-protection
```

Use the commands in the following context to configure CPU protection policies.

```
configure system security cpu-protection
```

For more information, see the *7705 SAR Gen 2 System Management Guide*.

2.11.3.2.3 Assigning a key group to a router interface

Note: This implementation applies to the classic CLI.

The following example shows key group configuration for a router interface.

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        interface demo
            group-encryption
                encryption-keygroup 6 direction inbound
                encryption-keygroup 6 direction outbound
                exit
            no shutdown
            exit
        exit
...
----------------------------------------------
```

2.11.3.2.4 Configuring IPv6

Example: Default configuration when IPv6 is enabled on an interface (MD-CLI)

```
[ex:/configure router "Base" interface "demo"]
A:admin@node-2# info
    admin-state enable
    port 1/2/37
    ipv6 {
        icmp6 {
            packet-too-big {
                number 100
                seconds 10
            }
            param-problem {
                number 100
                seconds 10
            }
            redirects {
                number 100
                seconds 10
            }
            time-exceeded {
                number 100
                seconds 10
            }
            unreachables {
                number 100
                seconds 10
            }
        }
    }
```

Example: Default configuration when IPv6 is enabled on an interface (classic CLI)

```
A:node-2>config>router>if# info
----------------------------------------------
            port 1/2/37
            ipv6
            exit
            no shutdown

A:node-2>config>router>if>ipv6# info detail
```

```
-----------------------------------------------
                icmp6
                    packet-too-big 100 10
                    param-problem 100 10
                    redirects 100 10
                    time-exceeded 100 10
                    unreachables 100 10
                exit
```

Use the commands in the following context to configure IPv6 on a router interface that you want to configure differently from the default configuration.

```
configure router interface ipv6 icmp6
```

Example: Configuration of IPv6 on a router interface (MD-CLI)

```
[ex:/configure router "Base" interface "demo"]
A:admin@node-2# info
    port 1/2/3
    ipv4 {
        primary {
            address 10.11.10.1
            prefix-length 24
        }
    }
    ipv6 {
        address 2001:db8::1 {
            prefix-length 24
        }
    }
```

Example: Configuration of IPv6 on a router interface (classic CLI)

```
A:node-2>config>router>if# info
-----------------------------------------------
            address 10.11.10.1/24
            port 1/2/37
            ipv6
                address 2001:db8::1/24
            exit
-----------------------------------------------
```

2.11.3.2.5 Configuring IPv6 over IPv4

The following sections provide several examples of the features that must be configured (tunnel ingress and egress node) to implement IPv6 over IPv4 relay services.

2.11.3.2.6 Tunnel ingress node

The following example shows the configuration of the interface through which the IPv6 over IPv4 traffic leaves the node. This must be configured on a network interface.

Example: MD-CLI

```
[ex:/configure router "Base"]
```

```
A:admin@node-2# info
...
    static-routes {
        route 3ffe::c8c8:c802/128 route-type unicast {
            indirect 10.200.200.2 {
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
...
#------------------------------------------------
echo "Static Route Configuration"
#------------------------------------------------
        static-route-entry 3ffe::c8c8:c802/128
            indirect 10.200.200.2
                shutdown
                tunnel-next-hop
                    resolution disabled
                exit
            exit
        exit
--------------------------------------------
```

The following example shows the configuration of the network interface.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface "ip-1.1.1.1" {
        port 1/1/1
        ipv4 {
            primary {
                address 10.1.1.1
                prefix-length 30
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        interface "ip-1.1.1.1"
            address 10.1.1.1/30
            port 1/1/1
          exit
...
----------------------------------------------
```

Both the IPv4 and IPv6 system addresses must be configured. The following example shows the configuration of interface information.

Example: MD-CLI

```
[ex:/configure router "Base"]
```

```
A:admin@node-2# info
...
    interface "system" {
        ipv4 {
            primary {
                address 10.0.113.1
                prefix-length 32
            }
        }
        ipv6 {
            address 3ffe::c8c8:c801 {
                prefix-length 128
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        interface "system"
            address 10.0.113.1/32
            ipv6
                address 3ffe::c8c8:c801/128
            exit
        exit
...
----------------------------------------------
```

2.11.3.2.6.1 Learning the tunnel endpoint IPv4 system address

The following example shows the OSPF configuration to learn the IPv4 system address of the tunnel endpoint.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    ospf 0 {
        area 0.0.0.0 {
            interface "ip-1.1.1.1" {
            }
            interface "system" {
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        ospf
            area 0.0.0.0
                interface "system"
                exit
                interface "ip-1.1.1.1"
                exit
```

```
            exit
        exit
-----------------------------------------------
```

2.11.3.2.6.2 Configuring an IPv4 BGP peer

The following example shows the configuration of an IPv4 BGP peer with (IPv4 and) IPv6 protocol families.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
     bgp {
        router-id 203.0.113.1
        }
        export {
            policy ["ospf3"]
        }
        group "main" {
            type internal
            family {
                ipv4 true
                ipv6 true
            }
        }
        neighbor "203.0.113.2" {
            group "main"
            peer-as 1
            local-as {
                as-number 1
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
-----------------------------------------------
...
        bgp
            export "ospf3"
            router-id 203.0.113.1
            group "main"
                family ipv4 ipv6
                type internal
                neighbor 203.0.113.2
                    local-as 1
                    peer-as 1
                exit
            exit
        exit
...
-----------------------------------------------
```

2.11.3.2.6.3 IPv6 over IPv4 tunnel configuration example

The IPv6 address is the next-hop as it is received through BGP. The IPv4 address is the system address of the tunnel's endpoint.

See Configuring an IPv4 BGP peer for an example that shows the configuration of a policy to export IPv6 routes into BGP.

The following example shows an IPv6 over IPv4 tunnel configuration.

Example: MD-CLI

```
[ex:/configure policy-options]
A:admin@node-2# info
    policy-statement "ospf3" {
        description "Plcy Stmnt For 'From ospf3 To bgp'"
        entry 10 {
            description "Entry From Protocol ospf3 To bgp"
            from {
                protocol {
                    name [ospf3]
                }
            }
            to {
                protocol {
                    name [bgp]
                }
            }
            action {
                action-type accept
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        policy-options
            policy-statement "ospf3"
                description "Plcy Stmnt For 'From ospf3 To bgp'"
                entry 10
                    description "Entry From Protocol ospf3 To bgp"
                    from
                        protocol ospf3
                    exit
                    to
                        protocol bgp
                    exit
                    action accept
                    exit
                exit
            exit
        exit
...
----------------------------------------------
```

2.11.3.2.7 Tunnel egress node

The following example shows the configuration of the interface through which the IPv6 over IPv4 traffic leaves the node. It must be configured on a network interface. Both the IPv4 and IPv6 system addresses must be configured.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
  static-routes {
        route 3ffe::c8c8:c801/128 route-type unicast {
            indirect 10.0.113.1 {
                }
        }
```

Example: classic CLI

```
A:node-2>config>router# info
#--------------------------------------------------
"Static Route Configuration"
#--------------------------------------------------
        static-route-entry 3ffe::c8c8:c801/128
            indirect 10.0.113.1
...
                exit
            exit
        exit
```

The following example shows the network interface configuration with both IPv4 and IPv6 addresses configured.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface "ip-1.1.1.2" {
        port 1/1/1
        ipv4 {
            primary {
                address 10.1.1.2
                prefix-length 30
            }
        }
    }
    interface "system" {
        ipv4 {
            primary {
                address 10.0.113.2
                prefix-length 32
            }
        }
        ipv6 {
            address 3ffe::c8c8:c802 {
                prefix-length 128
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        interface "ip-1.1.1.2"
            address 10.1.1.2/30
            port 1/1/1
        exit
        interface "system"
            address 10.0.113.2/32
            ipv6
                address 3ffe::c8c8:c802/128
            exit
        exit
----------------------------------------------
```

2.11.3.2.7.1 Learning the tunnel endpoint IPv4 system address

The following example shows the configuration of the OSPF configuration to learn the IPv4 system address of the tunnel endpoint.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
ospf 0 {
        area 0.0.0.0 {
            interface "ip-1.1.1.2" {
            }
            interface "system" {
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        ospf
            area 0.0.0.0
                interface "system"
                exit
                interface "ip-1.1.1.2"
                exit
            exit
        exit
----------------------------------------------
```

2.11.3.2.7.2 Configuring an IPv4 BGP peer

The following example shows the configuration an IPv4 BGP peer with (IPv4 and) IPv6 protocol families.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
    bgp {
        router-id 203.0.113.2
        export {
            policy ["ospf3"]
        }
        group "main" {
            type internal
            family {
                ipv4 true
                ipv6 true
            }
        }
        neighbor "203.0.113.1" {
            group "main"
            peer-as 1
            local-as {
                as-number 1
            }
        }
    }
```

Example: classic CLI

```
A:node-2config>router# info
----------------------------------------------
...
        bgp
            export "ospf3"
            router-id 203.0.113.2
            group "main"
                family ipv4 ipv6
                type internal
                neighbor 203.0.113.1
                    local-as 1
                    peer-as 1
                exit
            exit
        exit
...
----------------------------------------------
```

2.11.3.2.7.3 IPv6 over IPv4 tunnel configuration example

The IPv6 address is the next-hop as it is received through BGP. The IPv4 address is the system address of the tunnel's endpoint.

See Configuring an IPv4 BGP peer for an example of the configuration of a policy to export IPv6 routes into BGP.

The following example shows an IPv6 over IPv4 tunnel configuration.

Example: MD-CLI

```
[ex:/configure policy-options]
A:admin@node-2# info
```

```
    policy-statement "ospf3" {
        description "Plcy Stmnt For 'From ospf3 To bgp'"
        entry 10 {
            description "Entry From Protocol ospf3 To bgp"
            from {
                protocol {
                    name [ospf3]
                }
            }
            to {
                protocol {
                    name [bgp]
                }
            }
            action {
                action-type accept
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        policy-options
            policy-statement "ospf3"
                description "Plcy Stmnt For 'From ospf3 To bgp'"
                entry 10
                    description "Entry From Protocol ospf3 To bgp"
                    from
                        protocol ospf3
                    exit
                    to
                        protocol bgp
                    exit
                    action accept
                    exit
                exit
            exit
        exit
----------------------------------------------
```

2.11.3.2.8 Router advertisement

To configure the router to originate router advertisement messages on an interface, the interface must be configured under the router-advertisement context and be enabled. All other router advertisement configuration command options are optional.

Use the commands in the following contexts to configure router advertisement:

- **MD-CLI**

```
configure router ipv6 router-advertisement
configure service vprn ipv6 router-advertisement
```

- **classic CLI**

```
configure router router-advertisement
```

```
configure service vprn router-advertisement
```

The following example shows a router advertisement configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
    ipv6 {
        router-advertisement {
            interface "n1" {
                admin-state enable
                use-virtual-mac true
                prefix 2001:db8:2::/64 {
                }
                prefix 2001:db8:3::/64 {
                    autonomous true
                    on-link true
                    preferred-lifetime 604800
                    valid-lifetime 2592000
                }
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router>router-advert# info
----------------------------------------------
            interface "n1"
                prefix 2001:db8:3::/64
                exit
                use-virtual-mac
                no shutdown
            exit
----------------------------------------------
*A:node-2>config>router>router-advert# interface n1
*A:node-2>config>router>router-advert>if# prefix 2001:db8:3::/64
A:node-2>config>router>router-advert>if>prefix# info
----------------------------------------------
                    autonomous
                    on-link
                    preferred-lifetime 604800
                    valid-lifetime 2592000
----------------------------------------------
```

2.11.3.2.9 Configuring IPv6

The following example shows the default configuration when IPv6 is enabled on the interface.

Example: Default IPv6 configuration (MD-CLI)

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface "test" {
        port 1/3/37
        ipv6 {
        }
    }
```

```
[ex:/configure router "Base" interface "test" ipv6]
A:admin@node-2# info detail
...
    icmp6 {
        packet-too-big {
            admin-state enable
            number 100
            seconds 10
        }
        param-problem {
            admin-state enable
            number 100
            seconds 10
        }
        redirects {
            admin-state enable
            number 100
            seconds 10
        }
        time-exceeded {
            admin-state enable
            number 100
            seconds 10
        }
        unreachables {
            admin-state enable
            number 100
            seconds 10
        }
    }
```

Example: Default IPv6 configuration (classic CLI)

```
A:node-2>config>router# info
----------------------------------------------
#--------------------------------------------------
   "IP Configuration"
#--------------------------------------------------
        interface "test"
            port 1/3/37
            ipv6
                exit
            no shutdown
        exit
A:node-2>config>router>if>ipv6$ info detail
----------------------------------------------
                icmp6
                    packet-too-big 100 10
                    param-problem 100 10
                    redirects 100 10
                    time-exceeded 100 10
                    unreachables 100 10
                exit
```

The following example shows an IPv6 configuration.

Example: IPv6 configuration (MD-CLI)

```
[ex:/configure router "Base" interface "test"]
A:admin@node-2# info
    port 1/3/37
    ipv4 {
```

```
        primary {
            address 10.11.10.1
            prefix-length 24
        }
    }
    ipv6 {
        address 2001:db8::1 {
            prefix-length 24
        }
    }
```

Example: IPv6 configuration (classic CLI)

```
A:node-2>config>router>if# info
----------------------------------------------
            address 10.11.10.1/24
            port 1/3/37
            ipv6
                address 2001:db8::1/24
            exit
----------------------------------------------
```

2.11.3.2.9.1 IPv6 over IPv4 tunnel configuration example

The IPv6 address is the next-hop as it is received through BGP. The IPv4 address is the system address of the tunnel's endpoint.

Use the commands in the following context to export IPv6 routes into BGP.

```
configure router bgp export
```

The following example shows an IPv6 over IPv4 tunnel configuration.

Example: MD-CLI

```
[ex:/configure policy-options]
A:admin@node-2# info
    policy-statement "ospf3" {
        description "Plcy Stmnt For 'From ospf3 To bgp'"
        entry 10 {
            description "Entry From Protocol ospf3 To bgp"
            from {
                protocol {
                    name [ospf3]
                }
            }
            to {
                protocol {
                    name [bgp]
                }
            }
            action {
                action-type accept
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        policy-options
            policy-statement "ospf3"
                description "Plcy Stmnt For 'From ospf3 To bgp'"
                entry 10
                    description "Entry From Protocol ospf3 To bgp"
                    from
                        protocol ospf3
                    exit
                    to
                        protocol bgp
                    exit
                    action accept
                    exit
                exit
            exit
        exit
----------------------------------------------
```

2.11.3.2.10 Configuring proxy ARP

To configure proxy ARP, you can configure:

- A prefix list. Use the commands in the following context to configure a prefix list:
  - **MD-CLI**

    ```
    configure policy-options prefix-list
    ```

  - **classic CLI**

    ```
    configure router policy-options prefix-list
    ```

- A route policy statement and apply the specified prefix list. Use the commands in the following context to configure a route policy statement and apply the specified prefix list:
  - **MD-CLI**

    ```
    configure policy-options policy-statement
    ```

  - **classic CLI**

    ```
    configure router policy-options policy-statement
    ```

    - In the policy statement **entry to** context, specify the host source address(es) for which ARP requests can or cannot be forwarded to non-local networks, depending on the specified action.
    - In the policy statement **entry from** context, specify network prefixes that ARP requests to be forwarded or not forwarded depending on the action if a match is found. For more information about route policies, see the *7705 SAR Gen 2 Unicast Routing Protocols Guide*.

- Apply the policy statement to the proxy ARP configuration. Use the commands in the following context to apply the policy statement to the proxy ARP configuration.

```
configure router interface
```

Example: Prefix list and policy statement configuration (MD-CLI)

```
[ex:/configure]
A:admin@node-2# info
 policy-options {
        prefix-list "prefixlist1" {
            prefix 10.20.30.0/24 type through {
                through-length 32
            }
        }
        policy-statement "ProxyARPpolicy" {
            entry 10 {
                from {
                    prefix-list ["prefixlist1"]
                }
                to {
                    prefix-list ["prefixlist2"]
                }
                action {
                    action-type reject
                }
            }
            default-action {
                action-type accept
            }
        }
    }
```

Example: Prefix list and policy statement configuration (classic CLI)

```
A:node-2>config>router>policy-options# info
----------------------------------------------
            prefix-list "prefixlist1"
                    prefix 10.20.30.0/24 through 32
            exit
            prefix-list "prefixlist2"
                    prefix 10.10.10.0/24 through 32
            exit
...
            policy-statement "ProxyARPpolicy"
                entry 10
                    from
                        prefix-list "prefixlist1"
                    exit
                    to
                        prefix-list "prefixlist2"
                    exit
                    action reject
                exit
                default-action accept
                exit
            exit
...
----------------------------------------------
```

The following example shows a proxy ARP configuration.

Example: Proxy ARP configuration (MD-CLI)

```
[ex:/configure router "Base" interface "iparptest"]
A:admin@node-2# info
    ipv4 {
        primary {
            address 192.0.2.59
            prefix-length 24
        }
        neighbor-discovery {
            local-proxy-arp true
            proxy-arp-policy ["ProxyARPpolicy"]
        }
    }
```

Example: Proxy ARP configuration (classic CLI)

```
A:node-2>config>router>if# info
----------------------------------------------
            address 192.0.2.59/24
            local-proxy-arp
            proxy-arp-policy "ProxyARPpolicy"
            exit
----------------------------------------------
```

2.11.3.3 Deriving the router ID

The router ID defaults to the address specified in the system interface command. If the system interface is not configured with an IP address, the router ID inherits the last four bytes of the MAC address.

Use the commands in the following context to configure the router ID manually.

```
configure router
```

Use the commands in the following context, on the BGP protocol level, to define a BGP router ID.

```
configure router bgp router-id
```

Note: A router ID configured under the **bgp router-id** context is only used within BGP.

If a new router ID is configured, protocols are not automatically restarted with the new router ID. The next time a protocol is initialized, the new router ID is used. An interim period of time can occur when different protocols use different router IDs. To force the new router ID, issue the shutdown and no shutdown commands for each protocol that uses the router ID, or restart the entire router.

It is possible to configure SR OS to operate with an IPv6 only BOF and no IPv4 system interface address. When configured in this manner, the user must explicitly define IPv4 router IDs for protocols such as OSPF and BGP because there is no mechanism to derive the router ID from an IPv6 system interface address.

The following example shows a router ID configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    router-id 10.10.0.4
    interface "system" {
        ipv4 {
            primary {
                address 10.10.0.4
                prefix-length 32
            }
        }
    }
```

Example: classic CLI

```
A:node-2config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
        interface "system"
            address 10.10.0.4/32
        exit
    . . .
        router-id 10.10.0.4
#------------------------------------------
```

2.11.3.4 Configuring a confederation

Configuring a confederation is optional. The AS and confederation topology design should be carefully planned. Autonomous system (AS), confederation, and BGP connection and peering must be explicitly created on each participating router. Identify AS numbers, confederation numbers, and members participating in the confederation.

See the BGP section for CLI syntax and command descriptions.

The following example shows the configuration of the confederation topology in Figure 1: Confederation configuration.

Note:

- Confederations can be preconfigured before configuring BGP connections and peering.
- Each confederation can have up to 15 members.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    autonomous-system 100
    router-id 10.10.10.103
    confederation {
        confed-as-num 2002
        members 200 { }
        members 300 { }
        members 400 { }
    }
    interface "system" {
        ipv4 {
```

```
            primary {
                address 10.10.10.103
                prefix-length 32
            }
        }
    }
    interface "to-104" {
        port 1/1/1
        ipv4 {
            primary {
                address 10.0.0.103
                prefix-length 24
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
        interface "system"
            address 10.10.10.103/32
        exit
        interface "to-104"
            address 10.0.0.103/24
            port 1/1/1
        exit
        autonomous-system 100
        confederation 2002 members 200 300 400
        router-id 10.10.10.103

#------------------------------------------
```

2.11.3.5 Configuring an autonomous system

Configuring an autonomous system is optional. The following example shows the configuration of an autonomous system.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    autonomous-system 100
    router-id 10.10.10.103
    interface "system" {
        ipv4 {
            primary {
                address 10.10.10.103
                prefix-length 32
            }
        }
    }
    interface "to-104" {
        port 1/1/1
        ipv4 {
            primary {
                address 10.0.0.103
```

```
                prefix-length 24
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
        interface "system"
            address 10.10.10.103/32
        exit
   interface "to-104"
            address 10.0.0.103/24
            port 1/1/1
            exit
        exit
        autonomous-system 100
        router-id 10.10.10.103
#------------------------------------------
```

2.12 Service management tasks

This section describes the service management tasks.

2.12.1 Changing the system name

The **system** command sets the name of the device and is used in the prompt string. Only one system name can be configured. If multiple system names are configured, the last one configured overwrites the previous entry.

Use the following syntax to change the system name.

```
configure system name
```

The following example shows the system name change.

Example: MD-CLI

```
[ex:/configure system]
A:admin@node-2# info
    name "node-2"
    location "Mt.View, CA, NE corner of FERB 1 Building"
    coordinates "37.390, -122.05500 degrees lat."
    management-interface {
        configuration-mode mixed
    }
```

Example: classic CLI

```
A:node-2>config>system# info
#------------------------------------------------
echo "System Configuration"
```

```
#--------------------------------------------------
        name "node-2"
        location "Mt.View, CA, NE corner of FERB 1 Building"
        coordinates "37.390, -122.05500 degrees lat."
        management-interface
            configuration-mode mixed
        exit
...
#--------------------------------------------------
```

2.12.2 Modifying an interface configuration

This section provides examples of commands to use to modify the router interface configuration.

Example: Modifying IP address information (MD-CLI)

```
*[ex:/configure router "Base"]
A:admin@node-2# interface "to-sr1"

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# admin-state disable

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# ipv4

*[ex:/configure router "Base" interface "to-sr1" ipv4]
A:admin@node-2# primary

*[ex:/configure router "Base" interface "to-sr1" ipv4 primary]
A:admin@node-2# delete address

*[ex:/configure router "Base" interface "to-sr1" ipv4 primary]
A:admin@node-2# address 10.0.0.25

*[ex:/configure router "Base" interface "to-sr1" ipv4 primary]
A:admin@node-2# prefix-length 24

*[ex:/configure router "Base" interface "to-sr1" ipv4 primary]
A:admin@node-2# exit

*[ex:/configure router "Base" interface "to-sr1" ipv4]
A:admin@node-2# exit

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# admin-state enable
```

Example: Modifying the port information (MD-CLI)

```
*[ex:/configure router "Base"]
A:admin@node-2# interface "to-sr1"

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# admin-state disable

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# delete port

*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# port 1/1/2
```

```
*[ex:/configure router "Base" interface "to-sr1"]
A:admin@node-2# admin-state enable
```

Example: Modified output (MD-CLI)

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface "system" {
        ipv4 {
            primary {
                address 10.10.10.103
                prefix-length 32
            }
        }
    }
    interface "to-sr1" {
        admin-state enable
        port 1/1/2
        ipv4 {
            primary {
                address 10.0.0.25
                prefix-length 24
            }
        }
    }
```

Example: Modifying IP address information (classic CLI)

```
*A:node-2>config>router# interface “to-sr1”
*A:node-2>config>router>if# shutdown
*A:node-2>config>router>if# no address
*A:node-2>config>router>if# address 10.0.0.25/24
*A:node-2>config>router>if# no shutdown
```

Example: Modifying the port information (classic CLI)

```
*A:node-2>config>router# interface “to-sr1”
*A:node-2>config>router>if# shutdown
*A:node-2>config>router>if# no port
*A:node-2>config>router>if# port 1/1/2
*A:node-2>config>router>if# no shutdown
```

Example: Modified output (classic CLI)

```
A:node-2>config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
        interface "system"
            address 10.0.0.103/32
        exit
        interface "to-sr1"
            address 10.0.0.25/24
            port 1/1/2
        exit
        router-id 10.10.0.3
#------------------------------------------
```

2.12.3 Removing a key group from a router interface

Note: This implementation applies to the classic CLI.

The following example shows the commands to remove a key group from a router interface.

Example: classic CLI

```
*A:node-2>config>router# interface demo
*A:node-2>config>router>if# group-encryption
*A:node-2>config>router>if>group-encryp# no encryption-keygroup 6 direction inbound
*A:node-2>config>router>if>group-encryp# no encryption-keygroup 6 direction outbound
```

The following example shows that the key group configuration has been removed from a router interface.

Example: classic CLI

```
A:node-2>config>router# info
----------------------------------------------
...
        interface demo
            group-encryption
                exit
            no shutdown
            exit
        exit
...
----------------------------------------------
```

2.12.4 Changing the key group for a router interface

Note: This information applies to the classic CLI.

Use the following commands to change the key group on a router interface. The following example shows the inbound and outbound key groups being changed from key group 6 to key group 8.

Example: classic CLI

```
*A:node-2>config>router# interface demo
*A:node-2>config>router>if# group-encryption
*A:node-2>config>router>if>group-encryp# no encryption-keygroup 6 direction inbound
*A:node-2>config>router>if>group-encryp# encryption-keygroup 8 direction outbound
*A:node-2>config>router>if>group-encryp# encryption-keygroup 8 direction inbound
```

The following example shows that the key group configuration has been changed for the router interface.

Example: classic CLI

```
A:node-2config>router# info
----------------------------------------------
...
        interface demo
            group-encryption
                encryption-keygroup 8 direction inbound
```

```
                encryption-keygroup 8 direction outbound
                exit
            no shutdown
            exit
        exit
...
-----------------------------------------------
```

2.12.5 Deleting a logical IP interface

The following example shows how to delete a logical IP interface. Consider the following before you attempt to delete a logical IP interface:

1. Before an IP interface can be deleted, it must first be administratively disabled.
2. After the interface is administratively disabled, it can be deleted.

Example: MD-CLI

In MD-CLI, the **delete** command used with the **interface** command removes the entry.

```
*[ex:/configure router "Base"]
A:admin@node-2# interface "test-interface"

*[ex:/configure router "Base"]
A:admin@node-2# admin-state disable

*[ex:/configure router "Base"]
A:admin@node-2# exit

*[ex:/configure router "Base"]
A:admin@node-2# delete interface "test-interface"
```

Example: classic CLI

In classic CLI, the **no** form of the **interface** command typically removes the entry, but all entity associations must be shut down or deleted before an interface can be deleted.

```
*A:node-2>config>router# interface "test-interface"
*A:node-2>config>router>if$ shutdown
*A:node-2>config>router>if$ exit
*A:node-2>config>router# no interface "test-interface"
```

3 VRRP

3.1 VRRP overview

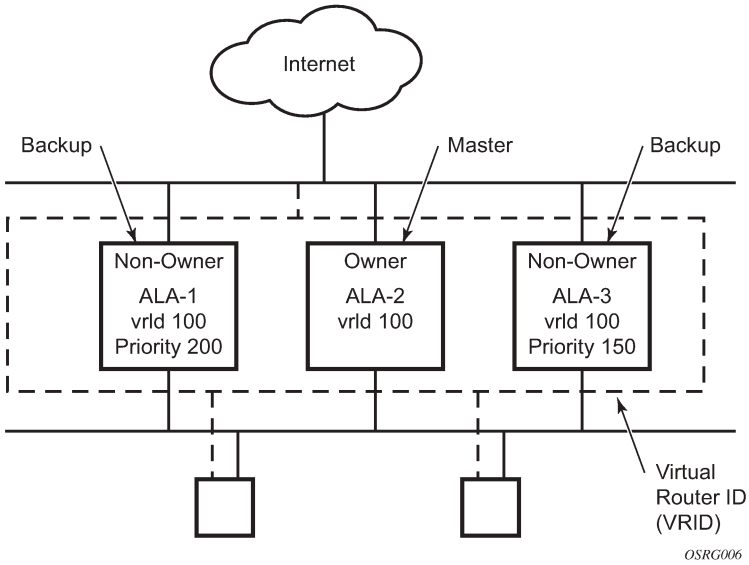
The Virtual Router Redundancy Protocol (VRRP) for IPv4 is defined in the IETF RFC 3768, *Virtual Router Redundancy Protocol*. VRRP for IPv6 is specified in *draft-ietf-vrrp-unified-spec-02.txt*. VRRP describes a method of implementing a redundant IP interface shared between two or more routers on a common LAN segment, allowing a group of routers to function as one virtual router. When this IP interface is specified as a default gateway on hosts directly attached to this LAN, the routers sharing the IP interface prevent a single point of failure by limiting access to this gateway address. VRRP can be implemented on IES service interfaces and on core network IP interfaces.

The VRRP standard RFC 3768 uses the term "master" state to denote the virtual router that is currently acting as the active forwarding router for the VRRP instance. This guide uses the term "active" as much as possible.

If the virtual router in master state fails, the backup router configured with the highest acceptable priority becomes the active virtual router. The new active router assumes normal packet forwarding for the local hosts.

The following figure shows an example of a VRRP configuration.

Figure 22: VRRP configuration



3.2 VRRP components

This section describes the VRRP components.

3.2.1 Virtual router

A virtual router is a logical entity managed by VRRP that acts as a default router for hosts on a shared LAN. It consists of a Virtual Router Identifier (VRID) and a set of associated IP addresses (or an address) across a common LAN. A VRRP router can be the backup for one or more virtual routers.

The purpose of supporting multiple IP addresses within a single virtual router is for multi-netting. This is a common mechanism that allows multiple local subnet attachments on a single routing interface. Up to four virtual routers are possible on a single Nokia IP interface. The virtual routers must be in the same subnet. Each virtual router has its own VRID, state machine, and messaging instance.

3.2.2 IP address owner

VRRP can be configured in either an owner or non-owner mode. The owner is the VRRP router whose virtual router IP address is the same as the real interface IP address. This is the router that responds to packets addressed to one of the IP addresses for ICMP pings, TCP connections, and others. All other virtual router instances participating in this message domain must have the same VRID configured and cannot be configured as owner.

Nokia routers allow the virtual routers to be configured as non-owners of the IP address. VRRP on a router can be configured to allow non-owners to respond to ICMP echo requests when they become the virtual router in master state for the VRRP instance. Telnet and other connection-oriented protocols can also be configured for master. However, the individual application conversations (connections) do not survive a VRRP failover. A non-owner VRRP router operating as a backup does not respond to any packets addressed to any of the virtual router IP addresses.

3.2.3 Primary and secondary IP addresses

A primary address is an IP address selected from the set of real interface addresses. VRRP advertisements are always sent using the primary IP address as the source of the IP packet.

An IP interface must always have a primary IP address assigned for VRRP to be active on the interface. Nokia routers support both primary and secondary IP addresses (multi-netting) on the IP interface. The virtual router VRID primary IP address is always the primary address on the IP interface. VRRP uses the primary IP address as the IP address placed in the source IP address field of the IP header for all VRRP messages sent on that interface.

3.2.4 Virtual router

The VRRP router that controls the IP addresses associated with a virtual router is considered to be in the master state, is the active router for the VRRP instance, and is responsible for forwarding packets sent to the VRRP IP address. An election process provides dynamic failover of the forwarding responsibility if the master becomes unavailable. This allows any of the virtual router IP addresses on the LAN to be used as the default first hop router by end hosts. This enables a higher availability default path without requiring configuration of dynamic routing or router discovery protocols on every end host.

If the master is unavailable, each backup virtual router for the VRID compares the configured priority values to determine the master role. In case of a tie, the virtual router with the highest primary IP address becomes master.

The preempt command option can be set to false to prevent a backup virtual router with a better priority value from becoming master when an existing non-owner virtual router is the current master. This is determined on a first-come, first-served basis.

While master, a virtual router routes and originates all IP packets into the LAN using the physical MAC address for the IP interface as the Layer 2 source MAC address, not the VRID MAC address. ARP packets also use the parent IP interface MAC address as the Layer 2 source MAC address while inserting the virtual router MAC address in the appropriate hardware address field. VRRP messages are the only packets transmitted using the virtual router MAC address as the Layer 2 source MAC address.

3.2.5 Virtual router backup

A new virtual router master is selected from the set of VRRP routers available to assume forwarding responsibility for a virtual router, in case the current master fails.

3.2.6 Owner and non-owner VRRP

The owner controls the IP address of the virtual router and is responsible for forwarding packets sent to this IP address. The owner assumes the role of the master virtual router. Only one virtual router in the domain can be configured as owner. All other virtual router instances participating in this message domain must have the same VRID configured.

The most important command option to be defined on a non-owner virtual router instance is the priority. The priority defines a virtual router's selection order in the master election process. The priority value and the preempt mode determine the virtual router with the highest priority to become the master virtual router.

The base priority is used to determine the in-use priority of the virtual router instance as modified by any optional VRRP priority control policy. VRRP priority control policies can be used to either override or adjust the base priority value depending on events or conditions within the chassis.

For information about non-owner access command options, see VRRP non-owner accessibility.

For owner virtual router instances, use the following commands to define the IP addresses that are advertised within VRRP advertisement messages:

- **MD-CLI**

```
configure router interface ipv4 vrrp backup
configure router interface ipv6 vrrp backup
```

- **classic CLI**

```
configure router interface vrrp backup
configure router interface ipv6 vrrp backup
```

For owner virtual router instances, after you define the IP addresses that are advertised within VRRP advertisement messages, this communicates the IP addresses that the master is advertising to backup virtual routers receiving the messages. The specified unicast IPv4 address must be equal to one of the existing IP addresses in the parental IP interface (primary or secondary) or the **backup** command fails.

For non-owner virtual router instances, the **backup** command for IPv4 or IPv6 creates an IP interface IP address used for routing IP packets and communicating with the system, based on which access command options are enabled (ntp-reply, ping-reply, telnet-reply, and ssh-reply). The specified unicast IPv4 address must exist on one of the local subnets of the parental IP interface. If the specified address does

not exist on one of the local subnets of the parental IP interface or if the specified address uses the same IP address as the parental IP interface, the **backup** command fails.

The **backup** command must be executed successfully at least once before the virtual router instance can enter the operational state.

The new interface IP address created with the **backup** command assumes the mask and command options of the corresponding parent IP interface IP address. The unicast IPv4 address is only active when the virtual router instance is operating in the master state. When not operating as master, the virtual router instance acts as if it is operationally down. It does not respond to ARP requests made to the unicast IPv4 address, nor does it route packets received with its VRID-derived source MAC address. A non-master virtual router instance always silently discards packets destined for the unicast IPv4 address. One virtual router instance may only have one virtual router IP address from a parental local subnet. Multiple virtual router instances can define a virtual router IP address from the same local subnet, as long as each IP address is different.

3.2.7 Configurable command options

As well as to backup IP addresses, to facilitate configuration of a virtual router on Nokia routers, the following command options can be defined in owner configurations:

- VRID
- Message interval and master inheritance
- VRRP message authentication
- Authentication data
- Virtual MAC address

The following command options can be defined in non-owner configurations:

- VRID
- Priority
- Message interval and master inheritance
- Master down interval
- Preempt mode
- VRRP message authentication
- Authentication data
- Virtual MAC address
- Inherit master VRRP router’s advertisement interval timer
- Policies

3.2.7.1 VRID

The Virtual Router ID (VRID) must be configured with the same value on each virtual router associated with the redundant IP addresses. It is placed in all VRRP advertisement messages sent by each virtual router.

3.2.7.2 Priority

The priority value affects the interaction between this VRID and the same VRID of other virtual routers participating on the same LAN. A higher-priority value defines a greater priority in becoming the virtual router master for the VRID. The priority value can only be configured when the defined IP address on the IP interface is different from the virtual router IP address (non-owner mode).

When the IP address on the IP interface matches the virtual router IP address (owner mode), the priority value is fixed at 255, the highest value possible. This virtual router member is considered the owner of the virtual router IP address. There can only be one owner of the virtual router IP address for all virtual router members.

The priority value 0 is reserved for VRRP advertisement message purposes. It is used to tell other virtual routers in the same VRID that this virtual router is no longer acting as master, triggering a new election process. When this happens, each backup virtual router sets its master down timer equal to the skew time value. This shortens the time until one of the backup virtual routers becomes master.

The current master virtual router must transmit a VRRP advertisement message immediately upon receipt of a VRRP message with priority set to 0. This prevents another backup from becoming master for a short period of time.

Non-owner virtual routers may be configured with a priority of 254 through 1. The default value is 100. Multiple non-owners can share the same priority value. When multiple non-owner backup virtual routers are tied (transmit VRRP advertisement messages simultaneously) in the election process, all attempts to become master simultaneously. The one with the best priority wins the election. If the priority value in the message is equal to the master’s local priority value, the primary IP address of the local master and of the message is evaluated as the tie breaker. The higher IP address becomes master. (The primary IP address is the source IP address of the VRRP advertisement message.)

The priority is also used to determine when to preempt the existing master. If the preempt mode value is true, VRRP advertisement messages from inferior (lower- priority) masters are discarded, causing the master down timer to expire and the transition to master state.

The priority value also dictates the skew time added to the master timeout period.

3.2.7.3 IP addresses

Each virtual router with the same VRID should be defined with the same set of IP addresses. These are the IP addresses being used by hosts on the LAN as gateway addresses. Multinetting supports 16 IP addresses on the IP interface; up to 16 addresses can be assigned to a specific virtual router instance.

3.2.7.4 Message interval and master inheritance

Each virtual router is configured with a message interval per VRID within which it participates. This command option must be the same for every virtual router on the VRID.

For IPv4, the default advertisement interval is 1 s and can be configured between 100 ms and 255 s 900 ms. For IPv6, the default advertisement interval is 1 s and can be configured between 100 ms and 40 s 950 ms.

As specified in the RFC, the advertisement interval field in every received VRRP advertisement message must match the locally configured advertisement interval. If a mismatch occurs, depending on the inherit configuration, the current master's advertisement interval setting can be used to operationally override the

locally configured advertisement interval setting. If the current master changes, the new master setting is used. If the local virtual router becomes master, the locally configured advertisement interval is enforced.

If a VRRP advertisement message is received with an advertisement interval set to a value different from the local value and the inherit command option is disabled, the message is discarded without processing.

The master virtual router on a VRID uses the advertisement interval to load the advertisement timer, specifying when to send the next VRRP advertisement message. Each backup virtual router on a VRID uses the advertisement interval (with the configured local priority) to determine the master down timer value.

VRRP advertisement messages that are fragmented, or contain IP options (IPv4), or contain extension headers (IPv6) require a longer message interval to be configured.

3.2.7.5 Skew time

The skew time is used to add a time period to the master down interval. This is not a configurable command option. It is determined from the current local priority of the virtual router's VRID. To calculate the skew time, the virtual router evaluates the following formula:

For IPv4: Skew Time equals ((256 - priority) / 256) seconds

For IPv6: Skew Time equals (((256 - priority) * Master_Adver_Interval) / 256) centiseconds

The higher the priority value, the shorter the skew time is. This means that virtual routers with a lower priority transition to master slower than virtual routers with a higher priority.

3.2.7.6 Master down interval

The master down interval is a calculated value used to load the master down timer. When the master down timer expires, the virtual router enters the master state. To calculate the master down interval, the virtual router evaluates the following formula:

Master Down Interval = (3 × Operational Advertisement Interval) + Skew Time

The operational advertisement interval is dependent upon the state of the inherit command option. When the inherit command option is enabled, the operational advertisement interval is determined from the current master's advertisement interval field in the VRRP advertisement message. When inherit is disabled, the operational advertisement interval must be equal to the locally configured advertisement interval.

The master down timer is only operational when the local virtual router is operating in backup mode.

3.2.7.7 Preempt mode

Preempt mode is a true or false configured value that controls whether a specific backup virtual router preempts a lower-priority master. The IP address owner always becomes master when available. Preempt mode cannot be set to false on the owner virtual router. The default value for preempt mode is true.

When the preempt mode is true, a master non-owner virtual router only allows itself to be preempted when the incoming VRRP advertisement message priority field value is one of the following:

- Greater than the virtual router in-use priority value
- Equal to the in-use priority value, and the source IP address (primary IP address) is greater than the virtual router instance primary IP address

A backup router only attempts to become the master router if the preempt mode is true and the received VRRP advertisement priority field is less than the virtual router in-use priority value.

3.2.7.8 VRRP message authentication

The authentication type command option defines the type of authentication used by the virtual router in VRRP advertisement message authentication. VRRP message authentication is applicable to IPv4 only. The current master uses the configured authentication type to indicate any egress message manipulation that must be performed in conjunction with any supporting authentication command options before transmitting a VRRP advertisement message.

The configured authentication type value is transmitted in the message authentication type field with the appropriate authentication data field filled in. Backup routers use the authentication type message field value in interpreting the contained authentication data field within received VRRP advertisement messages.

VRRP supports three message authentication methods that provide varying degrees of security. The supported authentication types are:

- 0 – No Authentication
- 1 – Simple Text Password
- 2 – IP Authentication Header

3.2.7.8.1 Authentication type 0 – no authentication

The use of type 0 indicates that VRRP advertisement messages are not authenticated (provides no authentication). The master transmitting VRRP advertisement messages transmit the value 0 in the egress messages authentication type field and the authentication data field. Backup virtual routers receiving VRRP advertisement messages with the authentication type field equal to 0 ignore the authentication data field in the message.

All compliant VRRP advertisement messages are accepted. The following fields within the received VRRP advertisement message are checked for compliance (the VRRP specification may require additional checks):

- IP header checks specific to VRRP
  - IP header destination IP address – must be 224.0.0.18
  - IP header TTL field – must be equal to 255; the packet must not have traversed any IP routed hops
  - IP header protocol field – must be 112 (decimal)
- VRRP message checks
  - Version field – must be set to the value of 2
  - Type field – must be set to the value of 1 (advertisement)
  - Virtual router ID field – must match one of the configured VRIDs on the ingress IP interface (all other fields are dependent on matching the virtual router ID field to one of the interfaces configured VRID command options)
  - Priority field – must be equal to or greater than the VRID in-use priority or be equal to 0 (if equal to the VRID in-use priority and 0, requires further processing about master/backup and senders IP address to determine validity of the message)

- Authentication type field – must be equal to 0
- Advertisement interval field – must be equal to the VRID configured advertisement interval
- Checksum field – must be valid
- Authentication data fields – must be ignored

VRRP messages not meeting the criteria are silently dropped.

3.2.7.8.2 Authentication type 1 – simple text password

The use of type 1 indicates that VRRP advertisement messages are authenticated with a clear (simple) text password. All virtual routers participating in the virtual router instance must be configured with the same 8-octet password. Transmitting virtual routers place a value of 1 in the VRRP advertisement message authentication type field and put the configured simple text password into the message authentication data field. Receiving virtual routers compare the message authentication data field with the local configured simple text password, based on the message authentication type field value of 1.

The same checks are performed as for type 0, with the following exceptions for VRRP message checks. (The VRRP specification may require additional checks.)

- **Authentication type field**

  The Authentication type field must be equal to 1.

- **Authentication data fields**

  The Authentication data fields must be equal to the VRID configured simple text password.

Any VRRP message not meeting the type 0 verification checks with the preceding exceptions are silently discarded.

3.2.7.8.3 Authentication failure

Any received VRRP advertisement message that fails authentication must be silently discarded with an invalid authentication counter incremented for the ingress virtual router instance.

3.2.7.9 Authentication data

This feature is different from the VRRP advertisement message field with the same name. This uses any required authentication information that is pertinent to the configured authentication type. The type of authentication data used for each authentication type is described in the following table.

Table 5: Authentication data type

Authentication type	Authentication data
0	None, authentication is not performed
1	Simple text password consisting of 8 octets

3.2.7.10 Virtual MAC address

The MAC address can be used instead of an IP address in ARP responses when the virtual router instance is master. The MAC address configuration must be the same for all virtual routers participating as a virtual router, or indeterminate connectivity by the attached IP hosts results. All VRRP advertisement messages are transmitted with *ieee-mac-address* as the source MAC.

3.2.7.11 VRRP advertisement message IP address list verification

VRRP advertisement messages contain an IP address count field that indicates the number of IP addresses listed in the sequential IP address fields at the end of the message.

The implementation always logs mismatching events. The decision on where and whether to forward the generated messages depends on the configuration of the event manager.

To facilitate the sending of mismatch log messages, each virtual router instance keeps the mismatch state associated with each source IP address in the VRRP master table. Whenever the state changes, a mismatch log message is generated indicating the source IP address within the message, the mismatch or match event, and the time of the event.

With secondary IP address support, multiple IP addresses may be found in the list and should match the IP address on the virtual router instance. Owner and non-owner virtual router instances have the supported IP addresses explicitly defined, making mismatched supported IP address within the interconnected virtual router instances a provisioning issue.

3.2.7.12 Inherit master VRRP router’s advertisement interval timer

The virtual router instance can inherit the master VRRP router’s advertisement interval timer, which is used by backup routers to calculate the master down timer.

The inheritance is only configurable in the non-owner nodal context. The inheritance is used to allow the current virtual router instance master to dictate the master down timer for all backup virtual routers.

3.2.7.13 IPv6 virtual router instance operationally up

After the IPv6 virtual router is configured with a minimum of one link-local backup address, the parent interface’s router advertisement must be configured to use the virtual MAC address for the virtual router to be considered operationally up.

3.2.7.14 Policies

Policies can be configured to control VRRP priority with the virtual router instance. VRRP priority control policies can be used to override or adjust the base priority value, depending on events or conditions within the chassis.

The policy can be associated with more than one virtual router instance. The priority events within the policy override or diminish the base priority dynamically affecting the in-use priority. As priority events clear in the policy, the in-use priority can eventually be restored to the base priority value.

Policies can only be configured in the non-owner VRRP context. For non-owner virtual router instances, if policies are not configured, then the base priority is used as the in-use priority.

3.3 VRRP priority control policies

The Nokia implementation of VRRP supports control policies to manipulate virtual router participation in the VRRP master election process and master self-deprecation. The local priority value for the virtual router instance is used to control the election process and master state.

3.3.1 VRRP virtual router policy constraints

Priority control policies can only be applied to non-owner VRRP virtual router instances. Owner VRRP virtual routers cannot be controlled by a priority control policy because they are required to have a priority value of 255 that cannot be diminished. Only one VRRP priority control policy can be applied to a non-owner virtual router instance.

Multiple VRRP virtual router instances may be associated with the same IP interface, allowing multiple priority control policies to be associated with the IP interface.

An applied VRRP priority control policy only affects the in-use priority on the virtual router instance when the preempt mode has been enabled. A virtual router instance with preempt mode disabled always uses the base priority as the in-use priority, ignoring any configured priority control policy.

3.3.2 VRRP virtual router instance base priority

Non-owner virtual router instances must have a base priority value between 1 and 254. The value 0 is reserved for master termination. The value 255 is reserved for owners. The default base priority for non-owner virtual router instances is the value 100.

The base priority is the starting priority for the VRRP instance. The actual in-use priority for the VRRP instance is derived from the base priority and an optional VRRP priority control policy.

3.3.3 VRRP priority control policy delta in-use priority limit

A VRRP priority control policy enforces an overall minimum value that the policy can subtract from the VRRP virtual router instance base priority. This value provides a lower limit to the delta priority events manipulation of the base priority.

A delta priority event is a conditional event defined in the priority control policy that subtracts an amount from the current, in-use priority for all VRRP virtual router instances to which the policy is applied. Multiple delta priority events can apply simultaneously, creating a dynamic priority value. The base priority for the instance, less the sum of the delta values, derives the actual priority value in-use.

An explicit priority event is a conditional event defined in the priority control policy that explicitly defines the in-use priority for the virtual router instance. The explicitly defined values are not affected by the delta in-use priority limit. When multiple explicit priority events happen simultaneously, the lowest value is used for the in-use priority. The configured base priority is not a factor in explicit priority overrides of the in-use priority.

The allowed range of the Delta In-Use Priority Limit is 1 to 254. The default is 1, which prevents the delta priority events from operationally disabling the virtual router instance.

3.3.4 VRRP priority control policy priority events

The main function of a VRRP priority control policy is to define conditions or events that impact the ability of the system to communicate with outside hosts or portions of the network. When one or multiple of these events are true, the base priority on the virtual router instance is either overwritten with an explicit value, or a sum of delta priorities is subtracted from the base priority. The result is the in-use priority for the virtual router instance. Any priority event may be configured as an explicit event or a delta event.

Explicit events override all delta events. When multiple explicit events occur, the event with the lowest priority value is assigned to the in-use priority. As events clear, the in-use priority is reevaluated accordingly and adjusted dynamically.

Delta priority events also have priority values. When no explicit events have occurred within the policy, the sum of the occurring delta events priorities is subtracted from the base priority of each virtual router instance. If the result is lower than the delta in-use priority limit, the delta in-use priority limit is used as the in-use priority for the virtual router instance. Otherwise, the in-use priority is set to the base priority less the sum of the delta events.

Each event generates a VRRP priority event message indicating the policy ID, the event type, the priority type (delta or explicit), and the event priority value. Another log message is generated when the event is no longer true, indicating that it has been cleared.

3.3.4.1 Priority event hold-set timers

Hold-set timers are used to dampen the effect of a flapping event. A flapping event is where the event continually transitions between clear and set. The hold-set value is loaded into a hold-set timer that prevents a set event from transitioning to the cleared state until it expires.

Each time an event transitions from cleared to set, the timer is loaded and begins to count down to zero. If the timer reaches zero, the event is allowed to enter the cleared state again. Entering the cleared state is always dependent on the object controlling the event conforming to the requirements defined in the event. It is possible, on some event types, to have a further set action reload the hold-set timer. This extends the amount of time that must expire before entering the cleared state.

See LAG degrade priority event for an example of a hold-set timer setting.

3.3.4.2 Port down priority event

The port down priority event is assigned to either a physical port or a SONET/SDH channel for the 7705 SAR Gen 2. The port or channel operational state is evaluated to determine a port down priority event or event clear.

When the port or channel operational state is up, the port down priority event is considered false or cleared. When the port or channel operational state is down, the port down priority event is considered true or set.

3.3.4.3 LAG degrade priority event

The LAG degrade priority event is tied to an existing LAG. The LAG degrade priority event is conditional on the percentage of available port bandwidth on the LAG. Multiple bandwidth percentage thresholds may be defined, each with its own priority value.

If the LAG transitions from one threshold to the next, the previous threshold priority value is subtracted from the total delta sum while the new threshold priority value is added to the sum. The new sum is then subtracted from the base priority and compared to the delta in-use priority limit to derive the new in-use priority on the virtual router instance.

The following example illustrates a LAG priority event and interaction with the hold- set timer in changing the in-use priority.

The following state and timer settings are used for the LAG events shown in the following table:

- User-defined thresholds: 2 ports down, 4 ports down, 6 ports down
- LAG configured ports: 8 ports
- hold-set timer (hold-set): 5 seconds

Table 6: LAG events

Time (seconds)	LAG port state	Command option	State	Comments
0	All ports down	Event State	Set - 8 ports down	—
		Event Threshold	6 ports down	—
		Hold-set Timer	5 seconds	Set to **hold-set** command option
1	One port up	Event State	Set - 8 ports down	Cannot change until hold-set timer expires
		Event Threshold	6 ports down	—
		Hold-set Timer	5 seconds	Event does not affect timer
2	All ports up	Event State	Set - 8 ports down	Still waiting for hold-set timer expiry
		Event Threshold	6 ports down	—
		Hold-set Timer	3 seconds	—
5	All ports up	Event State	Cleared - All ports up	—
		Event Threshold	None	Event cleared
		Hold-set Timer	Expired	—
100	Five ports down	Event State	Set - 5 ports down	—
		Event Threshold	4 ports down	—

Time (seconds)	LAG port state	Command option	State	Comments
		Hold-set Timer	Expired	Set to **hold-set** command option
102	Three ports down	Event State	Set - 5 ports down	—
		Event Threshold	4 ports down	—
		Hold-set Timer	3 seconds	—
103	All ports up	Event State	Set - 5 ports down	—
		Event Threshold	4 ports down	—
		Hold-set Timer	2 second	—
104	Two ports down	Event State	Set - 5 ports down	—
		Event Threshold	4 ports down	—
		Hold-set timer	1 second	Current threshold is 5, so 2 down has no effect
105	Two ports down	Event State	Set - 2 ports down	—
		Event Threshold	2 ports down	—
		Hold-set timer	Expired	—
200	Four ports down	Event State	Set - 2 ports down	—
		Event Threshold	4 ports down	—
		Hold-set timer	5 seconds	Set to hold-set command option
202	Seven ports down	Event State	Set - 7 ports down	Changed because of increase
		Event Threshold	6 ports down	—
		Hold-set timer	5 seconds	Set to **hold-set** because of threshold increase
206	All ports up	Event State	Set - 7 ports down	—
		Event Threshold	6 ports down	—
		Hold-set timer	1 second	—
207	All ports up	Event State	Cleared - All ports up	—
		Event Threshold	None	Event cleared
		Hold-set timer	Expired	—

3.3.4.4 Host unreachable priority event

The host unreachable priority event creates a continuous ping task that is used to test connectivity to a remote host. The path to the remote host and the remote host must be capable and configured to accept ICMP echo request and replies for the ping to be successful.

The ping task is controlled by interval and size parameters that define how often the ICMP request messages are transmitted and the size of each message. A historical missing reply parameter defines when the ping destination is considered unreachable.

When the host is unreachable, the host unreachable priority event is considered true or set. When the host is reachable, the host unreachable priority event is considered false or cleared.

3.3.4.5 Route unknown priority event

The route unknown priority event defines a task that monitors the existence of a route prefix in the system routing table.

The route monitoring task can be constrained by a condition that allows a prefix that is less specific than the defined prefix to be considered as a match. The source protocol can be defined to indicate which protocol the installed route must be populated with. To further define match criteria when multiple instances of the route prefix exist, an optional next-hop command option can be defined.

When a route prefix exists within the active route table that matches the defined match criteria, the route unknown priority event is considered false or cleared. When a route prefix does not exist within the active route table that matches the defined criteria, the route unknown priority event is considered true or set.

3.4 VRRP non-owner accessibility

Although the RFC states that only VRRP owners can respond to ping and other management-oriented protocols directed to the VRID IP addresses, the routers allow an override of this restraint on a per VRRP virtual router instance basis.

3.4.1 Non-owner access ping reply

When non-owner access ping reply is enabled on a virtual router instance, ICMP echo request messages destined for the non-owner virtual router instance IP addresses are not discarded at the IP interface when operating in master mode. ICMP echo request messages are always discarded in backup mode.

When non-owner access ping reply is disabled on a virtual router instance, ICMP echo request messages destined for the non-owner virtual router instance IP addresses are silently discarded in both the master and backup modes.

3.4.2 Non-owner access Telnet

When non-owner access Telnet is enabled on a virtual router instance, authorized Telnet sessions may be established that are destined for the virtual router instance IP addresses when operating in master mode. Telnet sessions are always discarded at the IP interface when destined for a virtual router IP address operating in backup mode. Enabling non-owner access Telnet does not guarantee Telnet access; correct

management and security features must be enabled to allow Telnet on this interface and possibly from the source IP address.

When non-owner access Telnet is disabled on a virtual router instance, Telnet sessions destined for the non-owner virtual router instance IP addresses are silently discarded in both master and backup modes.

3.4.3 Non-owner access SSH

When non-owner access SSH is enabled on a virtual router instance, authorized SSH sessions may be established that are destined for the virtual router instance IP addresses when operating in master mode. SSH sessions are always discarded at the IP interface when destined for a virtual router IP address operating in backup mode. Enabling non-owner access SSH does not guarantee SSH access; correct management and security features must be enabled to allow SSH on this interface and possibly from the specified source IP address. SSH is applicable to IPv4 VRRP only.

When non-owner access SSH is disabled on a virtual router instance, SSH sessions destined for the non-owner virtual router instance IP addresses are silently discarded in both master and backup modes.

3.5 VRRP instance inheritance

VRRP Instance Inheritance allows multiple VRRP instances to follow the state of a lead VRRP instance. This allows the VRRP control plane to handle more VRRP instances without requiring increased control message volumes.

The lead VRRP instance is configured by including the oper-group configuration statement within the lead VRRP instance configuration. The lead instance must be configured with the necessary message timers to detect VRRP failures at the configured rate. The following VRRP instances, referred to as following instances, are then associated with the appropriate lead VRRP instance by including the monitor-oper-group statement (for example, **monitor-oper-group “vrrp-LI-1”**).

The following are VRRP instance inheritance behaviors:

- One VRRP instance acts as the leading instance and behaves normally. This instance is configured with timers to attain the required detection times.
- The user can associate additional VRRP instances with the leading VRRP instance by configuring the following instances to monitor the lead oper-group instance.
- Command options associated with the instance state or priority are ignored within a following VRRP instance.
- If the lead instance becomes primary, all following instances assume a primary role for their respective VRRP instances.
- If the lead instance transitions from primary to standby, all the following instances transition to standby.
- If the lead instance transitions to a down state, all following instances transition to standby.

3.5.1 Configuration guidelines

The following guidelines apply to VRRP instance inheritance when configuring multiple VRRP instances that are bound together and share a common state with a lead VRRP instance.

- Nokia recommends all following VRRP instances exist on a common set of ports or LAG interface as the lead VRRP instance.
- Only include a single VRRP instance on the oper-group used for the lead VRRP instance.
- A VRRP instance cannot include both an oper-group and a monitor-oper-group simultaneously.
- A VRRP instance cannot be configured to monitor an oper-group and also be configured as passive.

3.5.2 VRRP instance inheritance configuration tasks

3.5.2.1 Lead VRRP instance configuration

Configure the lead VRRP instance with timer intervals for the wanted detection time. The key addition is the inclusion of the **oper-group** command to the configuration. The following example shows the lead VRRP instance configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    interface "base-1-1" {
        port 1/1/3:1
        ipv6 {
            link-local-address {
                address fe80::1
                duplicate-address-detection false
            }
            address 2500::1 {
                prefix-length 64
                duplicate-address-detection false
            }
            vrrp 1 {
                backup [2500::10 fe80::1:1]
                message-interval 5
                mac 00:00:5e:00:02:01
                priority 130
                ping-reply true
                oper-group "op-v6LI-1"
                bfd-liveness {
                    dest-ip 2000::2
                    service-name "100"
                    interface-name "bfd-1-1"
                }
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#--------------------------------------------------
echo "IP Configuration"
#--------------------------------------------------
        interface "base-1-1"
            port 1/1/3:1
            ipv6
                address 2500::1/64 dad-disable
```

```
                link-local-address fe80::1 dad-disable
                vrrp 1
                    backup 2500::10
                    backup fe80::1:1
                    priority 130
                    ping-reply
                    message-interval 5
                    mac 00:00:5e:00:02:01
                    oper-group "op-v6LI-1"
                    bfd-enable name "100" interface "bfd-1-1" dst-ip 2000::2
                exit
            exit
            no shutdown
        exit
-----------------------------------------
```

3.5.2.2 Following VRRP instances

To configure VRRP instances with slower timer intervals include the **monitor-oper-group** command for MD-CLI and the **oper-group** command for classic CLI in the configuration. The following example shows VRRP instance configuration with slower timer intervals.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# interface base-1-2
     interface "base-1-2" {
        port 1/1/3:2
        ipv6 {
            link-local-address {
                address fe80::2
                duplicate-address-detection false
            }
            address 2500:0:1::1 {
                prefix-length 64
                duplicate-address-detection false
            }
            vrrp 1 {
                backup [2500:0:1::10 fe80::1:2]
                message-interval 40
                mac 00:00:5e:00:02:01
                priority 130
                ping-reply true
                monitor-oper-group "op-v6LI-1"
                bfd-liveness {
                    dest-ip 2000::2
                    service-name "100"
                    interface-name "bfd-1-1"
                }
            }
        }
    }
```

Example: classic CLI

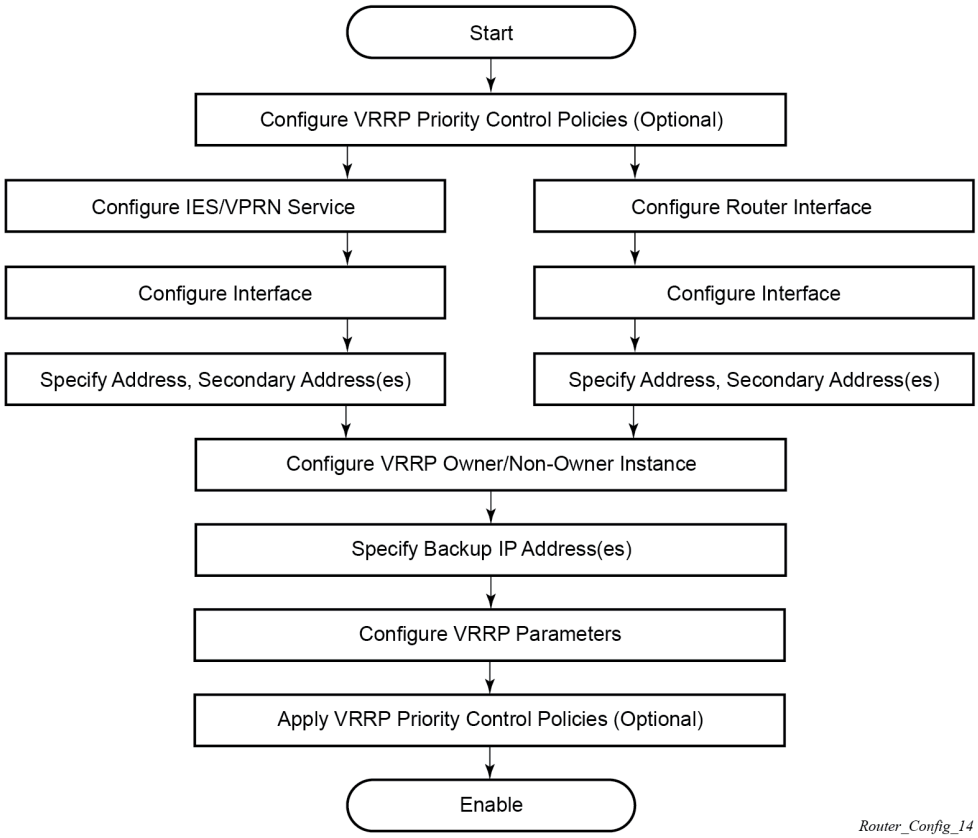
```
A:node-2>config>router# info
#--------------------------------------------------
echo "IP Configuration"
#--------------------------------------------------
        interface "base-1-2"
```

```
            port 1/1/3:2
            ipv6
                address 2500:0:1::1 dad-disable
                link-local-address fe80::1 dad-disable
                vrrp 1
                    backup 2500:0:1::10
                    backup fe80::1:2
                    message-interval 40
                    priority 130
                    ping-reply
                    mac 00:00:5e:00:02:01
                    oper-group "op-v6LI-1"
                    bfd-enable name "100" interface "bfd-1-1" dst-ip 2000::2
                exit
            exit
            no shutdown
        exit
```

3.6 VRRP configuration process overview

Figure 23: VRRP configuration and implementation flow shows the process to configure and implement VRRP command options.

Figure 23: VRRP configuration and implementation flow



3.7 Configuration notes

This section describes VRRP configuration restrictions.

3.7.1 General

The following apply when configuring VRRP:

- Creating and applying VRRP policies are optional.
- **backup** command:
  - The backup IP addresses must be on the same subnet. The backup addresses explicitly define which IP addresses are in the VRRP advertisement message IP address list.
  - In the owner mode, the backup IP address must be identical to one of the interface IP addresses. The backup address explicitly defines which IP addresses are in the VRRP advertisement message IP address list.
  - For IPv6, one of the configured backup addresses must be the link-local address of the owner VRRP instance.

3.8 Configuring VRRP with CLI

This section provides information about configuring VRRP using the CLI.

3.8.1 VRRP configuration overview

Configuring VRRP policies and VRRP instances on interfaces and router interfaces is optional. The basic owner and non-owner VRRP configurations on an IES or router interface must specify the **backup** command option.

VRRP helps eliminate the single point of failure in a routed environment by using virtual router IP addresses shared between two or more routers connecting the common domain. VRRP provides dynamic failover of the forwarding responsibility if the master becomes unavailable.

The VRRP implementation allows one master per IP subnet. All other VRRP instances in the same domain must be in backup mode.

3.8.1.1 Preconfiguration requirements

Configuring VRRP policies:

VRRP policies must be configured before they can be applied to an interface or IES/VPRN VRRP instance. VRRP policies are configured in the **configure vrrp** context.

Configuring VRRP on an IES or VPRN service interface:

- The service customer account must be created before configuring an IES or VPRN VRRP instance.

- The interface address must be specified in the both the owner and non-owner IES, VPRN, or router interface instances.

3.8.2 Basic VRRP configurations

Configure VRRP command options in the following contexts.

3.8.2.1 VRRP policy

Configuring and applying VRRP policies are optional. There are no default VRRP policies. Each policy must be explicitly defined. A VRRP configuration must include the following:

- policy ID
- at least one of the following priority events:
  - port down
  - LAG port down
  - host unreachable
  - route unknown

Example: VRRP policy configuration for the 7705 SAR Gen 2 (MD-CLI)

```
[ex:/configure vrrp policy 100]
A:admin@node-2# info
    delta-in-use-limit 50
    priority-event {
        host-unreachable "10.10.24.4" {
            drop-count 25
        }
        port-down 4/1/2 {
            hold-set 43200
            priority {
                priority-level 100
                event-type delta
            }
        }
        port-down 4/1/3 {
            priority {
                priority-level 200
                event-type explicit
            }
        }
        route-unknown 10.10.0.0/32 {
            protocol [bgp]
            priority {
                priority-level 50
                event-type delta
            }
        }
    }
```

Example: VRRP policy configuration for the 7705 SAR Gen 2 (classic CLI)

```
A:node-2>config>vrrp>policy# info
----------------------------------------------
```

```
            delta-in-use-limit 50
            priority-event
                port-down 4/1/2
                    hold-set 43200
                    priority 100 delta
                exit
                port-down 4/1/3
                    priority 200 explicit
                exit
                lag-port-down 1
                    number-down 3
                        priority 50 explicit
                    exit
                exit
                host-unreachable 10.10.24.4
                    drop-count 25
                exit
                route-unknown 10.10.0.0/32
                    priority 50 delta
                    protocol bgp
                exit
            exit
-----------------------------------------------
```

3.8.2.2 VRRP IES service configuration

VRRP command options are configured within an IES service with two contexts: owner or nonowner. The user specifies the status when creating the VRRP configuration. When configured as owner, the virtual router instance owns the backup IP addresses. All other virtual router instances participating in this message domain must have the same VRID configured, and cannot be configured as owner.

SR OS supports passive VRRP, which does not require multiple VRRP instances to achieve default gateway load-balancing. Passive VRRP is a VRRP setting in which the transmission and reception of keepalive messages is completely suppressed; and therefore, the VPRN interface always behaves as the active router. Passive VRRP is enabled by adding the **passive** keyword to the VRRP instance at creation. For passive VRRP, the convergence time for link or node failures is not affected by the VRRP convergence, as all nodes in the VRRP instance are acting as active routers.

For IPv4, the user can configure up to four VRIDs on an IES service interface, with each of the four VRIDs able to manage up to 16 backup IP addresses. For IPv6, the user can configure four VRIDs on an IES service interface for FP4 cards and later, with each VRID able to manage up to 10 backup IP addresses.

VRRP command options configured within an IES service must include the following:

- VRID
- backup IP addresses

Example: IES service owner and nonowner VRRP configuration (MD-CLI)

```
[ex:/configure service ies "1"]
A:ex@node-2# info
    customer "1"
    interface "testing" {
        sap 1/1/55:0 {
        }
        ipv4 {
            primary {
                address 10.10.10.16
                prefix-length 24
```

```
            }
            vrrp 12 {
                authentication-key "z1rddawlDRZWzirCADyRv4MfzJVlDQsv hash2"
                backup [10.10.10.15]
                policy 1
            }
        }
    }
    interface "tuesday" {
        sap 7/1/1.2.2 {
        }
        ipv4 {
            primary {
                    address 10.10.36.2
                    prefix-length 24
                }
                vrrp 19 {
                    authentication-key "ungWv48Bz+pBQUDeXa4iI5Jsnw== hash2"
                    backup [10.10.36.2]
                    owner true
                }
            }
        }
    }
```

Example: IES service owner and nonowner VRRP configuration (classic CLI)

```
A:node-2>config>service# info
----------------------------------------------
...
        ies 1 name "1" customer 1 create
            interface "tuesday" create
                address 10.10.36.2/24
                sap 7/1/1.2.2 create
                vrrp 19 owner
                    backup 10.10.36.2
                    authentication-key "z1rddawlDRZWzirCADyRv4MfzJVlDQsv" hash2
                exit
            exit
            interface "testing" create
                address 10.10.10.16/24
                sap 1/1/55:0 create
                vrrp 12
                    backup 10.10.10.15
                    policy 1
                    authentication-key "ungWv48Bz+pBQUDeXa4iI5Jsnw==" hash2
                exit
            exit
            no shutdown
----------------------------------------------
```

3.8.2.2.1 Configure VRRP for IPv6

The following example shows a VRRP for IPV6 configuration and applies to the 7705 SAR Gen 2. The interface must be configured first.

Example: MD-CLI

```
[ex:/configure router "Base" ipv6 router-advertisement]
A:admin@node-2# info
```

```
    interface "Application-interface-101" {
        use-virtual-mac true
    }

[ex:/configure service ies "100"]
A:admin@node-2# info
    description "Application VLAN 921"
    customer "1"
    interface "Application-interface-101" {
        sap ccag-1.a:921 {
            description "cross connect to VPLS 921"
        }
        ipv4 {
            primary {
                address 10.152.2.220
                prefix-length 28
            }
            vrrp 217 {
                backup [10.152.2.222]
                priority 254
                ping-reply true
            }
        }
        ipv6 {
            link-local-address {
                address fe80::d68f:1:221:fffd
                duplicate-address-detection false
            }
            address 2001:db8:d68f:1:221::fffd {
                prefix-length 64
            }
            vrrp 219 {
                backup [fe80::d68f:1:221:ffff]
                priority 254
                ping-reply true
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router>router-advert# info
----------------------------------------------
            interface "Application-interface-101"
                use-virtual-mac
                no shutdown
            exit
...
----------------------------------------------

A:node-2>config>service>ies# info
----------------------------------------------
            description "Application VLAN 921"
            interface "Application-interface-101" create
                address 10.152.2.220/28
                vrrp 217
                    backup 10.152.2.222
                    priority 254
                    ping-reply
                exit
                ipv6
                    address 2001:db8:D68F:1:221::FFFD/64
                    link-local-address fe80::d68f:1:221:fffd dad-disable
```

```
                    vrrp 219
                        backup fe80::d68f:1:221:ffff
                        priority 254
                        ping-reply
                    exit
                exit
                sap ccag-1.a:921 create
                    description "cross connect to VPLS 921"
                exit
            exit
            no shutdown
----------------------------------------------
```

3.8.2.3 VRRP router interface command options

VRRP command options are configured on a router interface with two contexts: owner or nonowner. The user specifies the status is specified when creating the VRRP configuration. When configured as owner, the virtual router instance (VRID) owns the backed up IP addresses. All other virtual router instances participating in this message domain must have the same VRID configured and cannot be configured as owner.

SR OS supports passive VRRP, which does not require multiple VRRP instances to achieve default gateway load-balancing. Passive VRRP is a VRRP setting in which the transmission and reception of keepalive messages is completely suppressed; and therefore, the VPRN interface always behaves as the active router. Passive VRRP is enabled by adding the **passive** keyword to the VRRP instance at creation. For passive VRRP, the convergence time for link or node failures is not affected by the VRRP convergence, as all nodes in the VRRP instance are acting as active routers.

For IPv4, the user can configure up to four VRIDs on a router interface, with each VRID able to manage up to 16 backup IP addresses. For IPv6, the user can configure four VRIDs on a router interface for FP4 cards and later, with each VRID able to manage up to 10 backup IP addresses.

VRRP command options configured on a router interface must include the following:

- VRID
- backup IP addresses

Example: Router interface owner and nonowner VRRP configuration (MD-CLI)

```
[ex:/configure router "Base"]
A:admin@node-2#
...
    interface "system" {
    ipv4 {
            primary {
                address 10.10.0.4 {
                prefix-length 32
            }
        }
    }
    interface "test1" {
        ipv4 {
            primary {
                address 10.10.14.1
                prefix-length 24
            }
            secondary 10.10.16.1 {
                prefix-length 24
            }
```

```
            secondary 10.10.17.1 {
                prefix-length 24
            }
            secondary 10.10.18.1 {
                prefix-length 24
            }
        }
    }
    interface "test2" {
        ipv4 {
            primary {
                address 10.10.10.23
                prefix-length 24
            }
            vrrp 1 {
                authentication-key "V+mMCSI1pnX+5dHDE729xj4E3YCngRQ= hash2"
                backup [10.10.10.23]
                owner true
            }
        }
    }
```

Example: Router interface owner and nonowner VRRP configuration (classic CLI)

```
A:node-2>config>router# info
#------------------------------------------
echo "IP Configuration "
#------------------------------------------
        interface "system"
            address 10.10.0.4/32
        exit
        interface "test1"
            address 10.10.14.1/24
            secondary 10.10.16.1/24
            secondary 10.10.17.1/24
            secondary 10.10.18.1/24
        exit
        interface "test2"
            address 10.10.10.23/24
            vrrp 1 owner
                backup 10.10.10.23
                authentication-key "V+mMCSI1pnX+5dHDE729xj4E3YCngRQ=" hash2
            exit
        exit
#------------------------------------------
```

3.8.3 Common configuration tasks

This section provides a brief overview of the tasks that must be performed to configure VRRP and provides the CLI commands.

VRRP command options are defined under a service interface or a router interface context. An IP address must be assigned to each IP interface. Only one IP address can be associated with an IP interface but several secondary IP addresses also be associated.

Owner and non-owner configurations must include the following command options:

- All participating routers in a VRRP instance must be configured with the same VRID.

- All participating non-owner routers can specify up to 16 backup IP addresses (IP addresses that the master is representing). The owner configuration must include at least one backup IP address.
- For IPv6, all participating routers must be configured with the same link-local backup address (the one configured for the owner instance).

Other owner and non-owner configurations include the following optional commands:

- authentication-key
- MAC
- message-interval

In addition to the common command options, the following non-owner commands can be configured:

- master-int-inherit
- ntp-reply
- priority
- policy
- ping-reply
- preempt
- telnet-reply
- ssh-reply (IPv4 only)
- [no] shutdown

3.8.3.1 Creating interface command options

If multiple subnets are configured on an Ethernet interface, VRRP can be configured on each subnet.

The following example shows an IP interface configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
    router-id 10.10.0.1
    interface "system" {
    ipv4 {
            primary {
                address 10.10.0.1 {
                prefix-length 32
            }
        }
    }
    interface "testA" {
        ipv4 {
            primary {
                address 10.123.123.123
                prefix-length 24
            }
        }
    }
     interface "testB" {
        ipv4 {
            primary {
                address 10.10.14.1
```

```
                prefix-length 24
            }
            secondary 10.10.16.1 {
                prefix-length 24
            }
            secondary 10.10.17.1 {
                prefix-length 24
            }
            secondary 10.10.18.1 {
                prefix-length 24
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
echo "IP Configuration "
#------------------------------------------
        interface "system"
            address 10.10.0.1/32
        exit
        interface "testA"
            address 10.123.123.123/24
        exit
        interface "testB"
            address 10.10.14.1/24
            secondary 10.10.16.1/24
            secondary 10.10.17.1/24
            secondary 10.10.18.1/24
        exit
        router-id 10.10.0.1
#------------------------------------------
```

3.8.4 Configuring VRRP policy components

The following example shows VRRP policy component configurations.

Example: MD-CLI

```
[ex:/configure vrrp policy 1]
A:admin@node-2# info
    delta-in-use-limit 50
    priority-event {
        port-down 1/1/2 {
            hold-set 43200
            priority {
                priority-level 100
                event-type delta
            }
        }
        route-unknown 0.0.0.0/0 {
            protocol [isis]
        }
    }
```

Example: classic CLI

```
A:node-2>config>vrrp# info
----------------------------------------------
        policy 1
            delta-in-use-limit 50
            priority-event
                port-down 1/1/2
                    hold-set 43200
                    priority 100 delta
                exit
                route-unknown 0.0.0.0/0
                    protocol isis
                exit
            exit
        exit
----------------------------------------------
```

3.8.4.1 Configuring service VRRP

VRRP command options can be configured on an interface in a service to provide virtual default router support, which allows traffic to be routed without relying on a single router in case of failure. VRRP can be configured in the following two ways.

3.8.4.1.1 Non-owner VRRP

The following example shows a basic non-owner VRRP configuration.

Example: MD-CLI

```
[ex:/configure service ies "100"]
A:admin@node-2# info
    interface "testing" {
        sap 1/1/55:0 {
         ipv4 {
            primary {
                address 10.10.10.16
                prefix-length 24
            }
        ipv4 {
            vrrp 12 {
                authentication-key "testabc"
                backup [10.10.10.15]
                policy 1
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>service>ies# info
----------------------------------------------
            interface "testing" create
                address 10.10.10.16/24
                sap 1/1/55:0 create
                vrrp 12
                    backup 10.10.10.15
```

```
                    policy 1
                    authentication-key "testabc"
                exit
            exit
-----------------------------------------
```

3.8.4.1.2 Owner service VRRP

The following example shows an owner service VRRP configuration.

Example: MD-CLI

```
[ex:/configure router "Base" interface "test2"]
A:admin@node-2# info
    ipv4 {
        primary {
            address 10.10.10.23
            prefix-length 24
        }
        vrrp 1 {
            authentication-key "testabc"
            backup [10.10.10.23]
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
echo "IP Configuration "
#------------------------------------------
...
        interface "test2"
            address 10.10.10.23/24
            vrrp 1 owner
                backup 10.10.10.23
                authentication-key "testabc"
            exit
        exit
#------------------------------------------
```

3.8.4.2 Configuring router interface VRRP command options

VRRP command options can be configured on an interface in an interface to provide virtual default router support, which allows traffic to be routed without relying on a single router in case of failure. VRRP can be configured in following two ways.

3.8.4.2.1 Router interface VRRP non-owner

The following example shows a router interface VRRP non-owner configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2# info
```

```
        interface "if-test" {
            ipv4 {
                primary {
                    address 10.20.30.40
                    prefix-length 24
                }
                secondary 10.10.50.1 {
                    prefix-length 24
                }
                secondary 10.10.60.1 {
                    prefix-length 24
                }
                secondary 10.10.70.1 {
                    prefix-length 24
                }
                vrrp 1 {
                    authentication-key "testabc hash2"
                    backup [10.10.50.2 10.10.60.2 10.10.70.2 10.20.30.41]
                    ping-reply true
                    telnet-reply true
                }
            }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
     interface "if-test"
            address 10.20.30.40/24
            secondary 10.10.50.1/24
            secondary 10.10.60.1/24
            secondary 10.10.70.1/24
            vrrp 1
                backup 10.10.50.2
                backup 10.10.60.2
                backup 10.10.70.2
                backup 10.20.30.41
                ping-reply
                telnet-reply
                authentication-key "testabc" hash2
            exit
        exit
#------------------------------------------
```

3.8.4.2.2 Router interface VRRP owner

The following example shows a router interface VRRP owner configuration.

Example: MD-CLI

```
[ex:/configure router "Base"]
A:admin@node-2#
    }
    interface "vrrpowner" {
        ipv4 {
            primary {
                address 10.10.10.23
                prefix-length 24
            }
            vrrp 1 {
```

```
                authentication-key "testabc hash2"
                backup [10.10.10.23]
                owner true
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>router# info
#------------------------------------------
     interface "vrrpowner"
            address 10.10.10.23/24
            vrrp 1 owner
                backup 10.10.10.23
                authentication-key "testabc" hash2
            exit
            no shutdown
        exit
#------------------------------------------
```

3.9 VRRP configuration management tasks

This section describes the VRRP configuration management tasks.

3.9.1 Modifying a VRRP policy

To access a specific VRRP policy, specify the policy ID. Use the following command to display a list of VRRP policies.

```
show vrrp policy
```

The following example shows the modified VRRP policy configuration.

Example: MD-CLI

```
[ex:/configure vrrp policy 100]
A:admin@node-2# info
    delta-in-use-limit 50
    priority-event {
        host-unreachable "10.10.24.4" {
            drop-count 25
        }
        port-down 1/1/2 {
            hold-set 43200
        }
        port-down 1/1/3 {
            priority {
                priority-level 200
                event-type explicit
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>vrrp>policy# info
----------------------------------------------
            delta-in-use-limit 50
            priority-event
                port-down 1/1/2
                    hold-set 43200
                    priority 100 delta
                exit
                port-down 1/1/3
                    priority 200 explicit
                exit
                host-unreachable 10.10.24.4
                    drop-count 25
                exit
            exit
----------------------------------------------
```

3.9.1.1 Deleting a VRRP policy

VRRP policies are only applied to non-owner VRRP instances. A VRRP policy cannot be deleted if it is applied to an interface or to an IES service. Each instance in which the policy is applied must be deleted.

Use the following command to show where VRRP policies are applies to an entity.

```
show vrrp policy
```

The following example shows where VRRP policies are applied to an entity.

Output example: MD-CLI

```
===============================================================================
VRRP Policies
===============================================================================
Policy    Current             Current      Current      Delta Applied Svc
Id        Priority & Effect   Explicit     Delta Sum    Limit         Context
-------------------------------------------------------------------------------
1         200 Explicit        200          100          50          Yes
15        254                 None         None         1           No
32        100                 None         None         1           No
===============================================================================
```

Output example: classic CLI

```
===============================================================================
VRRP Policies
===============================================================================
Policy    Current             Current      Current      Delta       Applied
Id        Priority & Effect   Explicit     Delta Sum    Limit
-------------------------------------------------------------------------------
1         200 Explicit        200          100          50          Yes
15        254                 None         None         1           No
32        100                 None         None         1           No
===============================================================================
```

The following example shows the deletion of a VRRP policy.

Example: MD-CLI

```
*[ex:/configure vrrp]
A:admin@node-2# delete policy 1

*[ex:/configure vrrp]
A:admin@node-2# commit
```

Example: classic CLI

```
*A:node-2>config>vrrp# no policy 1
```

3.9.2 Modifying service and interface VRRP command options

This section describes the process for modifying service and interface VRRP command options.

3.9.2.1 Modifying non-owner command options

After a VRRP instance is created as non-owner, it cannot be modified to the owner state. The VRID must be deleted, then recreated with the **owner** keyword, to invoke IP address ownership.

3.9.2.2 Modifying owner command options

After a VRRP instance is created as **owner**, it cannot be modified to the non-owner state. The VRID must be deleted, then recreated without the owner keyword, to remove IP address ownership.

Entering the **owner** command option is optional when entering the VRID for modification purposes.

3.9.2.3 Deleting VRRP from an interface or service

The VRID does not need to be shut down to remove the virtual router instance from an interface or service.

Use the following command to remove the virtual router instance:

- **MD-CLI**

```
configure service ies interface ipv4 delete vrrp
```

- **classic CLI**

```
configure service ies interface vrrp shutdown
configure service ies interface no vrrp
```

4 Filter policies

This chapter provides information about filter policies and management.

4.1 ACL filter policy overview

ACL filter policies, also referred to as Access Control Lists (ACLs) or just "filters", are sets of ordered rule entries specifying packet match criteria and actions to be performed to a packet upon a match. Filter policies are created with a unique filter ID and filter name. The filter name needs to be assigned during the creation of the filter policy. If a name is not specified at creation time, then SR OS assigns a string version of the filter ID as the name.

There are three main filter policies: **ip-filter** for IPv4, **ipv6-filter** for IPv6, and **mac-filter** for MAC level filtering. Additionally, the filter policy **scope** defines if the policy can be reused between different interfaces, embedded in another filter policy or applied at the system level:

- **exclusive filter**

  An exclusive filter defines policy rules explicitly for a single interface. An exclusive filter allows the highest level of customization but uses the most resources, because each exclusive filter consumes hardware resources on line cards on which the interface exists.

- **template filter**

  A template filter uses an identical set of policy rules across multiple interfaces. Template filters use a single set of resources per line card, regardless of how many interfaces use a specific template filter policy on that line card. Template filter policies used on access interfaces consume resources on line cards only if at least one access interface for a specific template filter policy is configured on a specific line card.

- **embedded filter**

  An embedded filter defines a common set of policy rules that can then be used (embedded) by other exclusive or template filters in the system. This allows optimized management of filter policies.

- **system filter**

  A system filter policy defines a common set of policy rules that can then be activated within other exclusive/template filters. It can be used, for example, as a system-level set of deny rules. This allows optimized management of common rules (similarly to embedded filters). However, active system filter policy entries are not duplicated inside each policy that activates the system policy (as is the case when embedding is used). The active system policy is downloaded after to line cards, and activated filter policies are chained to it.

After the filter policy is created, the policy must then be associated with interfaces or services, or with other filter policies (if the created policy cannot be directly deployed on an interface or service), so the incoming or outgoing traffic can be subjected to filter rules. Filter policies are associated with interfaces or services separately in the ingress and egress directions. A policy deployed on ingress and egress direction can be the same or different. In general, Nokia recommends using different filter policies for the ingress and egress directions and to use different filter policies per service type, because filter policies support different match criteria and different actions for different directions/service contexts.

A filter policy is applied to a packet in the ascending rule entry order. When a packet matches all the command options specified in a filter entry's match criteria, the system takes the action defined for that entry. If a packet does not match the entry command options, the packet is compared to the next higher numerical filter entry rule, and so on.

In classic CLI, if the packet does not match any of the entries, the system executes the default action specified in the filter policy: drop or forward.

In MD-CLI, if the packet does not match any of the entries, the system executes the default action specified in the filter policy: drop or accept.

For Layer 2, either an IPv4/IPv6 or MAC filter policy can be applied. For Layer 3 and network interfaces, an IPv4/IPv6 policy can be applied. For R-VPLS service, a Layer 2 filter policy can be applied to Layer 2 forwarded traffic and a Layer 3 filter policy can be applied to Layer 3 routed traffic. For dual-stack interfaces, if both IPv4 and IPv6 filter policies are configured, the policy applied are based on the outer IP header of the packet. Non-IP packets do not affect an IP filter policy, so the default action in the IP filter policy do not apply to these packets. Egress IPv4 QoS-based classification criteria are ignored when egress MAC filter policy is configured on the same interface.

Additionally, platforms that support Network Group Encryption (NGE) can use IP exception filters. IP exception filters scan all outbound traffic entering an NGE domain and allow packets that match the exception filter criteria to transit the NGE domain unencrypted. See Router encryption exceptions using ACLs for information about IP exception filters supported by NGE nodes.

4.1.1 Filter policy basics

The following subsections define main functionality supported by filter policies.

4.1.1.1 Filter policy packet match criteria

This section defines packet match criteria supported on SR OS for IPv4, IPv6, and MAC filters. Supported criteria types depend on the hardware platform and filter direction.

General notes:

- If multiple unique match criteria are specified in a single filter policy entry, all criteria must be met in order for the packet to be considered a match against that filter policy entry (logical AND).
- Any match criteria not explicitly defined is ignored during match.
- An ACL filter policy entry with match criteria defined, but no action configured, is considered incomplete and inactive (an entry is not downloaded to the line card). A filter policy must have at least one entry active for the policy to be considered active.
- An ACL filter entry with no match conditions defined matches all packets.
- Because an ACL filter policy is an ordered list, entries should be configured (numbered) from the most explicit to the least explicit.

4.1.1.2 IPv4/IPv6 filter policy entry match criteria

This section describes the IPv4 and IPv6 match criteria supported by SR OS. The criteria are evaluated against the outer IPv4 or IPv6 header and a Layer 4 header that follows (if applicable). Support for match

criteria may depend on hardware or filter direction. Nokia recommends not configuring a filter in a direction or on hardware where a match criterion is not supported because this may lead to unwanted behavior.

IPv4 and IPv6 filter policies support three four filter types, including normal, source MAC, packet length, and destination class, with each supporting a different set of match criteria.

The match criteria available using the normal filter type are defined in this section. Layer 3 match criteria include:

- **DSCP**

  Match the specified DSCP command option against the Differentiated Services Code Point/Traffic Class field in the IPv4 or IPv6 packet header.

- **source IP, destination IP, or IP**

  Match the specified source or destination IPv4 or IPv6 address prefix against the IP address field in the IPv4 or IPv6 packet header. The user can optionally configure a mask to be used in a match. The **ip** command can be used to configure a single filter-policy entry that provides non-directional matching of either the source or destination (logical OR).

- **flow label**

  Match the specified flow label against the Flow label field in IPv6 packets. The user can optionally configure a mask to be used in a match. This operation is supported on ingress filters.

- **protocol**

  Match the specified protocol against the Protocol field in the IPv4 packet header (for example, TCP, UDP, IGMP) of the outer IPv4. "*" can be used to specify TCP or UDP upper-layer protocol match (Logical OR).

- **Next Header**

  Match the specified upper-layer protocol (such as, TCP, UDP, IGMPv6) against the Next Header field of the IPv6 packet header. "*" can be used to specify TCP or UDP upper-layer protocol match (Logical OR).

  Use the following command to match against up to six extension headers.

  ```
  configure system ip ipv6-eh max
  ```

  Use the following command to match against the Next Header value of the IPv6 header.

  ```
  configure system ip ipv6-eh limited
  ```

Fragment match criteria

Match for the presence of fragmented packet. For IPv4, match against the MF bit or Fragment Offset field to determine whether the packet is a fragment. For IPv6, match against the Next Header field for the Fragment Extension Header value to determine whether the packet is a fragment. Up to six extension headers are matched against to find the Fragmentation Extension Header.

IPv4 and IPv6 filters support matching against initial fragment using **first-only** or non-initial fragment **non-first-only**.

IPv4 match fragment **true** or **false** criteria are supported on both ingress and egress.

IPv4 match fragment **first-only** or **non-first-only** are supported on ingress only.

Operational note for fragmented traffic

IP and IPv6 filters defined to match TCP, UDP, ICMP, or SCTP criteria (such as source port, destination port, port, TCP ACK, TCP SYN, ICMP type, ICMP code) with command options of zero or false also match non-first fragment packets if other match criteria within the same filer entry are also met. Non-initial fragment packets do not contain a UDP, TCP, ICMP or SCTP header.

IPv4 options match criteria

You can configure the following IPv4 options match criteria exist:

- **IP option**

  Matches the specified command option value in the first option of the IPv4 packet. A user can optionally configure a mask to be used in a match.

- **option present**

  Matches the presence of IP options in the IPv4 packet. Padding and EOOL are also considered as IP options. Up to six IP options are matched against.

- **multiple option**

  Matches the presence of multiple IP options in the IPv4 packet.

- **source route option**

  Matches the presence of IP Option 3 or 9 (Loose or Strict Source Route) in the first three IP options of the IPv4 packet. A packet also matches this rule if the packet has more than three IP options.

IPv6 extension header match criteria

You can configure the following IPv6 Extension Header match criteria:

- **Authentication Header extension header**

  Matches for the presence of the Authentication Header extension header in the IPv6 packet. This match criterion is supported on ingress only.

- **Encapsulating Security Payload extension header**

  Matches for the presence of the Encapsulating Security Payload extension header in the IPv6 packet. This match criterion is supported on ingress only.

- **hop-by-hop options**

  Matches for the presence of hop-by-hop options extension header in the IPv6 packet. This match criterion is supported on ingress only.

- **Routing extension header type 0**

  Matches for the presence of Routing extension header type 0 in the IPv6 packet. This match criterion is supported on ingress only.

Upper-layer protocol match criteria

You can configure the following upper-layer protocol match criteria:

- **ICMP/ICMPv6 code field header**

  Matches the specified value against the code field of the ICMP or ICMPv6 header of the packet. This match is supported only for entries that also define protocol or next-header match for the ICMP or ICMPv6 protocol.

- **ICMP/ICMPv6 type field header**

Matches the specified value against the type field of the ICMP or ICMPv6 header of the packet. This match is supported only for entries that also define the protocol or next-header match for the ICMP or ICMPv6 protocol.

- **source port number, destination port number, or port**

  Matches the specified port, port list, or port range against the source port number or destination port number of the UDP, TCP, or SCTP packet header. An option to match either source or destination (Logical OR) using a single filter policy entry is supported by using a directionless port. Source or destination match is supported only for entries that also define protocol/next-header match for TCP, UDP, SCTP, or TCP or UDP protocols. Match on SCTP source port, destination port, or port is supported on ingress filter policy.

- **TCP ACK, TCP CWR, TCP ECE, TCP FIN, TCP NS, TCP PSH, TCP RST, TCP SYN, TCP URG**

  Matches the presence or absence of the TCP flags defined in RFC 793, RFC 3168, and RFC 3540 in the TCP header of the packet. This match criteria also requires defining the protocol/next-header match as TCP in the filter entry. TCP CWR, TCP ECE, TCP FIN, TCP NS, TCP PSH, TCP URG are supported on FP4 and FP5-based line cards only. TCP ACK, TCP RST, and TCP SYN are supported on all FP-based cards. When configured on other line cards, the bit for the unsupported TCP flags is ignored.

- **tcp-established**

  Matches the presence of the TCP flags ACK or RST in the TCP header of the packet. This match criteria requires defining the protocol/next-header match as TCP in the filter entry.

Note: Non initial fragmented packets do not match filter entries configured with layer 4 header match criteria. Only the first fragment of a packet includes the layer 4 header information.

For filter type match criteria

Additional match criteria for source MAC, packet length, and destination class are available using different filter types. See Filter policy type for more information.

IP prefixes, protocol numbers, TCP-UDP ports, and packet length values or ranges can also be grouped in the command **configure filter match-list**; see Filter policy advanced topics for more information.

4.1.1.3 IP exception filters

An NGE node supports IPv4 exception filters. See Router encryption exceptions using ACLs for information about IP exception filters supported by NGE nodes.

IP exception filters allow specific flows to pass through an IPsec-secured interface in the clear, should exceptions be required on the IPsec secured interface.

4.1.1.4 Filter policy actions

The following actions are supported by ACL filter policies:

- **drop**

  Allows users to deny traffic to ingress or egress the system.

  – **IPv4 packet-length and IPv6 payload-length conditional drop**

Traffic can be dropped based on IPv4 packet length or IPv6 payload length by specifying a packet length or payload length value or range within the drop filter action (the IPv6 payload length field does not account for the size of the fixed IP header, which is 40 bytes).

This filter action is supported on ingress IPv4 and IPv6 filter policies only, if the filter is configured on an egress interface the **packet-length** or **payload-length** match condition is always true.

This **drop** condition is a filter entry action evaluation, and not a filter entry match evaluation. Within this evaluation, the condition is checked after the packet matches the entry based on the specified filter entry match criteria.

Packets that match a filter policy entry match criteria and the **drop** *packet-length-value* or *payload-length-value* are dropped. Packets that match only the filter policy entry match criteria and do not match the **drop** *packet-length-value* or **drop** *payload-length value* are forwarded with no further matching in following filter entries.

Packets matching this filter entry and not matching the conditional criteria are not logged, counted, or mirrored.

– **IPv4 TTL and IPv6 hop limit conditional drop**

Traffic can be dropped based on a IPv4 TTL or IPv6 hop limit by specifying a TTL or hop limit value or range within the **drop** filter action.

This filter action is supported on ingress IPv4 and IPv6 filter policies only. If the filter is configured on an egress interface the packet-length or payload-length match condition is always true.

This **drop** condition is a filter entry action evaluation, and not a filter entry match evaluation. Within this evaluation, the condition is checked after the packet matches the entry based on the specified filter entry match criteria.

Packets that match filter policy entry match criteria and the drop TTL or drop hop limit value are dropped. Packets that match only the filter policy entry match criteria and do not match the drop TTL value or drop hop limit value are forwarded with no further match in following filter entries.

Packets matching this filter entry and not matching the conditional criteria are not logged, counted, or mirrored.

– **pattern conditional drop**

Traffic can be dropped when it is based on a pattern found in the packet header or data payload. The pattern is defined by an expression, mask, offset type, and offset value match in the first 256 bytes of a packet.

The pattern expression is up to 8 bytes long. The **offset-type** command identifies the starting point for the offset value and the supported offset-type command options are:

- **layer-3**: layer 3 IP header
- **layer-4**: layer 4 protocol header
- **data**: data payload for TCP or UDP protocols
- **dns-qtype**: DNS request or response query type

The content of the packet is compared with the expression/mask value found at the offset type and offset value as defined in the filter entry. For example, if the pattern is expression 0xAA11, mask 0xFFFF, offset-type data, offset-value 20, the filter entry compares the content of the first 2 bytes in the packet data payload found 20 bytes after the TCP/UDP header with 0xAA11.

This drop condition is a filter entry action evaluation, and not a filter entry match evaluation. Within this evaluation, the condition is checked after the packet matches the entry based on the specified filter entry match criteria.

Packets that match a filter policy's entry match criteria and the pattern, are dropped. Packets that match only the filter policy's entry match criteria and do not match the pattern, are forwarded without a further match in subsequent filter entries.

This filtering capability is supported on ingress IPv4 and IPv6 policies using FP4-based line cards, and cannot be configured on egress. A filter entry using a pattern, is not supported on FP2 or FP3-based line cards. If programmed, the pattern is ignored and the action is forward.

Packets matching this filter entry and not matching the conditional criteria are not logged, counted, or mirrored.

- **drop extracted traffic**

  Traffic extracted to the CPM can be dropped using ingress IPv4 and IPv6 filter policies based on filter match criteria. Any IP traffic extracted to the CPM is subject to this filter action, including routing protocols, snooped traffic, and TTL expired traffic.

  Packets that match the filter entry match criteria and extracted to the CPM are dropped. Packets that match only the filter entry match criteria and are not extracted to the CPM are forwarded with no further match in the subsequent filter entries.

  Cflowd, log, mirror, and statistics apply to all traffic matching the filter entry, regardless of the drop or forward action.

- **forward**

  Allows users to accept traffic to ingress or egress the system and be subject to regular processing.

- **accept a conditional filter action**

  Allows users to accept a conditional filter action. Use the following commands to configure a conditional filter action:

  – **MD-CLI**

  ```
  configure filter ip-filter entry action accept-when
  configure filter ipv6-filter entry action accept-when
  ```

  – **classic CLI**

  ```
  configure filter ip-filter entry action forward-when
  configure filter ipv6-filter entry action forward-when
  ```

  – **pattern conditional accept**

  Traffic can be accepted based on a pattern found in the packet header or data payload. The pattern is defined by an expression, mask, offset type, and offset value match in the first 256 bytes of a packet. The pattern expression is up to 8 bytes long. The offset type identifies the starting point for the offset value and the supported offset types are:

  - **layer-3**: Layer 3 IP header
  - **layer-4**: Layer 4 protocol header
  - **data**: data payload for TCP or UDP protocols
  - **dns-qtype**: DNS request or response query type

– The content of the packet is compared with the expression/mask value found at the offset type and offset value defined in the filter entry. For example, if the pattern is expression 0xAA11, mask 0xFFFF, offset-type data, and offset-value 20, then the filter entry compares the content of the first 2 bytes in the packet data payload found 20 bytes after the TCP/UDP header with 0xAA11.

  This accept condition is a filter entry action evaluation, and not a filter entry match evaluation. Within this evaluation, the condition is checked after the packet matches the entry based on the specified filter entry match criteria. Packets that match a filter policy's entry match criteria and the pattern, are accepted. Packets that match only the filter policy's entry match criteria and do not match the pattern, are dropped without a further match in subsequent filter entries.

  This filtering capability is supported on ingress IPv4 and IPv6 policies using FP4-based line cards and cannot be configured on egress. A filter entry using a pattern is not supported on FP2 or FP3-based line cards. If programmed, the pattern is ignored and the action is drop.

  Packets matching this filter entry and not matching the conditional criteria are not logged, counted, or mirrored.

- **rate limit**

  This action allows users to rate limit traffic matching a filter entry match criteria using IPv4, IPv6, or MAC filter policies.

  If multiple interfaces (including LAG interfaces) use the same **rate-limit** filter policy on different FPs, then the system allocates a rate limiter resource for each FP; an independent rate limit applies to each FP.

  If multiple interfaces (including LAG interfaces) use the same **rate-limit** filter policy on the same FP, then the system allocates a single rate limiter resource to the FP; a common aggregate rate limit is applied to those interfaces.

  Note that traffic extracted to the CPM is not rate limited by an ingress **rate-limit** filter policy while any traffic generated by the router can be rate limited by an egress **rate-limit** filter policy.

  rate-limit filter policy entries can coexist with cflowd, log, and mirror regardless of the outcome of the rate limit.

  Rate limit policers are configured with the maximum burst size (MBS) equals the committed burst size (CBS) equals 10 ms of the rate and high-prio-only equals 0.

  Interaction with QoS: Packets matching an ingress **rate-limit** filter policy entry bypass ingress QoS queuing or policing, and only the filter rate limit policer is applied. Packets matching an egress **rate-limit** filter policy bypass egress QoS policing, normal egress QoS queuing still applies.

  – **Kilobits-per-second and packets-per-second rate limit**
    The rate-limit action can be defined using kilobits per second or packets per second and is supported on both ingress and egress filter policies. The MBS value can also be configured using the kilobits-per-second policer.

    The packets-per-second rate limit and kilobits-per-second MBS are not supported when using a MAC filter policy.

  – **IPv4 packet-length and IPv6 payload-length conditional rate limit**
    Traffic can be rate limited based on the IPv4 packet length and IPv6 payload length by specifying a packet-length value or payload-length value or range within the rate-limit filter action. The IPv6 payload-length field does not account for the size of the fixed IP header, which is 40 bytes.

    This filter action is supported on ingress IPv4 and IPv6 filter policies only and cannot be configured on egress access or network interfaces.

This rate-limit condition is part of a filter entry action evaluation, and not a filter entry match evaluation. It is checked after the packet is determined to match the entry based on the configured filter entry match criteria.

Packets that match a filter policy's entry match criteria and the rate-limit packet-length value or rate-limit payload-length value are rate limited. Packets that match only the filter policy's entry match criteria and do not match the rate-limit packet-length value or rate-limit payload-length value are forwarded with no further match in subsequent filter entries.

Cflowd, logging, and mirroring apply to all traffic matching the ACL entry regardless of the outcome of the rate limiter and regardless of the packet-length value or payload-length value.

– **IPv4 TTL and IPv6 hop-limit conditional rate limit**

Traffic can be rate limited based on the IPv4 TTL or IPv6 hop-limit by specifying a TTL or hop-limit value or range within the rate-limit filter action using ingress IPv4 or IPv6 filter policies.

The match condition is part of action evaluation (for example, after the packet is determined to match the entry based on other match criteria configured). Packets that match a filter policy entry match criteria and the **rate-limit ttl** or *hop-limit* value are rate limited. Packets that match only the filter policy entry match criteria and do not match the **rate-limit** *ttl* or *hop-limit* value are forwarded with no further matching in the subsequent filter entries.

Cflowd, logging, and mirroring apply to all traffic matching the ACL entry regardless of the outcome of the **rate-limit** value and the *ttl-value* or *hop-limit-value*.

- – **next-hop address**

Changes the IP destination address used in routing from the address in the packet to the address configured in this PBR action. The user can configure whether the next-hop IP address must be direct (local subnet only) or indirect (any IP). In the indirect case, 0.0.0.0 (for IPv4) or :: (for IPv6) is allowed. Default routes may be different per VRF. This functionality is supported for ingress IPv4/IPv6 filter policies only, and is deployed on Layer 3 interfaces.

If the configured next-hop is not reachable, traffic is dropped and a "ICMP destination unreachable" message is sent. If **indirect** is not specified but the IP address is a remote IP address, traffic is dropped.

– **redirect policy**

Implements PBR next-hop or PBR next-hop router action with the ability to select and prioritize multiple redirect targets and monitor the specified redirect targets so PBR action can be changed if the selected destination goes down. Supported for ingress IPv4 and IPv6 filter policies deployed on Layer 3 interfaces only. See sectionRedirect policies for further details.

– **remark DSCP**

Allows a user to remark the DiffServ Code Points (DSCP) of packets matching filter policy entry criteria. Packets are remarked regardless of QoS-based in- or out-of-profile classification and QoS-based DSCP remarking is overridden. DSCP remarking is supported both as a main action and as an extended action. As a main action, this functionality applies to IPv4 and IPv6 filter policies of any scope and can only be applied at ingress on either access or network interfaces of Layer 3 services only. Although the filter is applied on ingress the DSCP remarking effectively performed on egress. As an extended action, this functionality applies to IPv4 and IPv6 filter policies of any scope and can be applied at ingress on either access or network interfaces of Layer 3 services, or at egress on Layer 3 subscriber interfaces.

– **router**

Changes the routing instance a packet is routed in from the upcoming interface's instance to the routing instance specified in the PBR action (supports both GRT and VPRN redirect). It is supported for ingress IPv4/IPv6 filter policies deployed on Layer 3 interfaces. The action can be combined with the next-hop action specifying direct/indirect IPv4/IPv6 next hop. Packets are dropped if they cannot be routed in the configured routing instance. See section "Traffic Leaking to GRT" in the *7705 SAR Gen 2 Layer 3 Services Guide: IES and VPRN* for more information.

- **ISA forward processing actions**

  ISA processing actions allow users to allow ingress traffic and send it for ISA processing as per specified ISA action. See Configuring filter policies with CLI for command details. The following ISA actions are supported:

  – **GTP local breakout**

    Forwards matching traffic to NAT instead of being GTP tunneled to the mobile user's PGW or GGSN. The action applies to GTP-subscriber-hosts. If filter is deployed on other entities, **action forward** is applied. Supported for IPv4 ingress filter policies only. If ISAs performing NAT are down, traffic is dropped.

  – **NAT**

    Forwards matching traffic for NAT. Supported for IPv4/IPv6 filter policies for Layer 3 services in GRT or VPRN. If ISAs performing NAT are down, traffic is dropped.

    For classic CLI options, see the *7705 SAR Gen 2 Classic CLI Command Reference Guide*.

    For MD-CLI options, see *7705 SAR Gen 2 MD-CLI Command Reference Guide*.

  – **reassemble**

    Forwards matching packets to the reassembly function. Supported for IPv4 ingress filter policies only. If ISAs performing reassemble are down, traffic is dropped.

  – **TCP for MSS adjustment**

    Forwards matching packets (TCP SYN) to an ISA BB group for MSS adjustment. In addition to the IP filter, the user also needs to configure the MSS adjust group under the Layer 3 service to specify the group ID and the new segment-size.

- **HTTP redirect**

  Implements the HTTP redirect captive portal. HTTP GET is forwarded to CPM card for captive portal processing by router. See the HTTP redirect (captive portal) section for more information.

- **ignore match**

  This action allow the user to disable a filter entry, as a result the entry is not programmed in hardware.

In addition to the preceding actions:

- A user can select a default action for a filter policy. The default action is executed on packets subjected to an active filter when none of the filter's active entries matches the packet. By default, filter policies have default action set to drop but the user can select a default action to be forward instead.
- Table 7: Default behavior when a PBR/PBF target is down defines default behavior for packets matching a PBR/PBF filter entry when a target is down.

Table 7: Default behavior when a PBR/PBF target is down

PBR/PBF action	Default behavior when down
Forward next-hop (any type)	Drop
Forward redirect-policy	Forward when redirect policy is shutdown
Forward redirect-policy	Forward when destination tests are enabled and the best destination is not reachable
Forward redirect-policy	Drop when destination tests are not enabled and the best destination is not reachable
Forward router	Drop

4.1.1.5 Viewing filter policy actions

A number of parameters determine the behavior of a packet after it has been matched to a defined criterion or set of criteria:

- the action configured by the user
- the context in which a filter policy is applied. For example, applying a filter policy in an unsupported context can result in simply forwarding the packet instead of applying the configured action.
- external factors, such as the reachability (according to specific test criteria) of a target

Use the following commands to display how a packet is handled by the system.

```
show filter ip
show filter ipv6
show filter mac
```

This section describes the key information displayed as part of the output for the preceding **show** commands, and how to interpret the information.

From a configuration point of view, the **show** command output displays the main action (primary and secondary), as well as the extended action.

The "PBR Target Status" field shows the basic information that the system has of the target based on simple verification methods. This information is only shown for the filter entries which are configured in redundancy mode (that is, with both primary and secondary main actions configured), and for ESI redirections. Specifically, the target status in the case of redundancy depends on several factors; for example, on a match in the routing table for next-hop redirects, or on VXLAN tunnel resolution for ESI redirects.

The "Downloaded Action" field specifically describes the action that the system performs on the packets that match the criterion (or criteria). This typically depends on the context in which the filter has been applied (whether it is supported or not), but in the case of redundancy, it also depends on the target status. For example, the downloaded action is the secondary main action when the target associated with the primary action is down. In the nominal (for example, non-failure condition) case the "Downloaded Action" reflects the behavior a packet is subject to. However, in transient cases (for example, in the case of a failure) it may not be able to capture what effectively happens to the packet.

The output also displays relevant information such as the default action when the target is down (see Table 7: Default behavior when a PBR/PBF target is down) as well as the overridden default action when **pbr-down-action-override** has been configured.

There are situations where, collectively, this information does not capture what effectively happens to the packet throughout the system. Use the following commands to perform advanced checks and display accurate packet fates.

```
show filter ip effective-action
show filter ipv6 effective-action
show filter mac effective-action
```

The criteria for determining when a target is down. While there is little ambiguity on that aspect when the target is local to the system performing the steering action, ambiguity is much more prominent when the target is distant. Therefore, because the use of **effective-action** triggers advanced tests, a discrepancy is introduced compared to the action when **effective-action** command option is not used. This is, for example, be the case for redundant actions.

4.1.1.6 Filter policy statistics

Filter policies support per-entry, packet/byte match statistics. The cumulative matched packet/Byte counters are available per ingress and per egress direction. Every packet arriving on an interface/service/subscriber using a filter policy increments ingress or egress (as applicable) matched packet/Byte count for a filter entry the packet matches (if any) on the line card the packet ingresses/egresses. For each policy, the counters for all entries are collected from all line cards, summarized and made available to an operator.

Filter policies applied on access interfaces are downloaded only to line cards that have interfaces associated with those filter policies. If a filter policy is not downloaded to any line card, the statistics show 0. If a filter policy is being removed from any of the line cards the policy is currently downloaded to (as result of association change or when a filter becomes inactive), the associated statistics are reset to 0.

Downloading a filter policy to a new line card continues incrementing existing statistics.

Operational notes:

Conditional action match criteria filter entries for **ttl**, **hop-limit**, **packet-length**, and **payload-length** support logging and statistics when the condition is met, allowing visibility of filter matched and action executed. If the condition is not met, packets are not logged and statistics against the entry are not incremented.

4.1.1.7 Filter policy logging

The SR OS supports logging of information from the packets that match a specific filter policy. Logging is configurable per filter policy entry by specifying a preconfigured filter log using the following command.

```
configure filter log
```

A filter log can be applied to ACL filters. Operators can configure multiple filter logs and specify the following:

- memory allocated to a filter log destination
- Syslog ID for a filter log destination
- filter logging summarization

- wrap-around behavior

The following are notes related to filter log summarization.

- The implementation of the feature applies to filter logs with destination Syslog.
- Summarization logging is the collection and summarization of log messages for one specific log ID within a period of time.
- The summarization interval is 100 seconds.
- Upon activation of a summary, a mini-table with a source and destination address and a count is created for each filter type (IPv4, IPv6, and MAC).
- Every received log packet (according to filter match) is examined for source or destination address.
- If the log packet (source or destination address) matches a source or destination address entry in the mini-table as a result of a packet previously received, the summary counter of the matching address is incremented.
- If the source or destination address of the log messages does not match an entry already present in the table, the source or destination address is stored in a free entry in the mini-table.
- When the mini-table has no more free entries, only the total counter is incremented.
- Upon the expiry of the summarization interval, the mini-table for each type is flushed to the Syslog destination.

Operational note

Conditional action match criteria filter entries for TTL, hop limit, packet length, and payload length support logging and statistics when the condition is met, allowing visibility of filter matched and action executed. If the condition is not met, packets are not logged and statistics against the entry are not incremented.

4.1.1.8 Filter policy management

4.1.1.8.1 Modifying Existing Filter Policy

There are several ways to modify an existing filter policy. A filter policy can be modified through configuration change or can have entries populated through dynamic, policy-controlled dynamic interfaces; for example, RADIUS, OpenFlow, FlowSpec, or Gx. Although in general, SR OS ensures filter resources exist before a filter can be modified, because of the dynamic nature of the policy-controlled interfaces, a configuration that was accepted may not be applied in H/W because of lack of resources. When that happens, an error is raised.

A filter policy can be modified directly—by changing/adding/deleting the existing entry in that filter policy—or indirectly. Examples of indirect change to filter policy include changing embedded filter entry this policy embeds (see the Filter policy scope and embedded filters section) or changing redirect policy this filter policy uses.

Finally, a filter policy deployed on a specific interface can be changed by changing the policy the interface is associated with.

All of the preceding changes can be done in service. A filter policy that is associated with service/interface cannot be deleted unless all associations are removed first.

For a large (complex) filter policy change, it may take a few seconds to load and initiate the filter policy configuration. Filter policy changes are downloaded to line cards immediately; therefore, users should use filter policy copy or transactional CLI to ensure partial policy change is not activated.

4.1.1.8.2 Filter policy copy

Perform bulk operations on filter policies by copying one filter's entries to another filter. Either all entries or a specified entry of the source filter can be selected to copy. When entries are copied, entry order is preserved unless the destination filter's entry ID is selected (applicable to single-entry copy).

Filter policy copy and renumbering in classic CLI

Note: The information applies to classic CLI.

SR OS supports entry copy and entry renumbering operations to assist in filter policy management.

Use the following commands to copy and overwrite filter entries.

```
configure filter copy ip-filter
configure filter copy ipv6-filter
configure filter copy mac-filter
```

The **copy** command allows overwriting of the existing entries in the destination filter by specifying the **overwrite** command option when using the **copy** command. Copy can be used, for example, when creating new policies from existing policies or when modifying an existing filter policy (an existing source policy is copied to a new destination policy, the new destination policy is modified, then the new destination policy is copied back to the source policy with overwrite specified).

Entry renumbering allows you to change the relative order of a filter policy entry by changing the entry ID. Entry renumbering can also be used to move two entries closer together or further apart, thereby creating additional entry space for new entries.

4.1.2 Filter policy advanced topics

4.1.2.1 Match list for filter policies

The filter match lists **ip-prefix-list**, **ipv6-prefix-list**, **protocol-list**, **port-list**, **ip-packet-length-list**, and **ipv6-packet-length-list** define a list of IP prefixes, IP protocols, TCP-UDP ports, and packet-length values or ranges that can be used as match criteria for line card IP and IPv6 filters. Additionally, **ip-prefix-list**, **ipv6-prefix-list**, and **port-list** can also be used in CPM filters.

A match list simplifies the filter policy configuration with multiple prefixes, protocols, or ports that can be matched in a single filter entry instead of creating an entry for each.

The same match list can be used in one or many filter policies. A change in match list content is automatically propagated across all policies that use that list.

4.1.2.1.1 Apply-path

The router supports the autogeneration of IPv4 and IPv6 prefix list entries for BGP peers that are configured in the Base router, in VPRN services, or for interfaces configured locally.

In the case of interfaces, the system creates a prefix list using the Base router, VPRN, or IES interface configuration.

Use the following commands to configure the autogeneration of IPv6 or IPv4 prefix list entries.

```
configure filter match-list ip-prefix-list apply-path
configure filter match-list ipv6-prefix-list apply-path
```

This capability simplifies the management of CPM filters to allow control traffic from trusted configured IP addresses only. The user can perform the following actions using the **apply-path** filter:

- specify one or more regex expression matches per match list, including wildcard matches (".*")
- mix autogenerated entries with statically configured entries within a match list

Additional rules are applied when using **apply-path** as follows:

- operational and administrative states of a specific router configuration are ignored when auto-generating address prefixes
- duplicates are not removed when populated by different autogeneration matches and static configuration
- configuration fails if auto-generation of an address prefix results in the filter policy resource exhaustion on a filter entry, system, or line-card level

4.1.2.1.2 Prefix-exclude

A prefix can be excluded from an IPv4 or IPv6 prefix list by using the **prefix-exclude** command.

For example, when the user needs to drop or forward traffic to 10.0.0.0/16 with the exception of 10.0.2.0/24, the following options are available.

By applying **prefix-exclude**, a single IP prefix list with two prefixes is configured:

Example: MD-CLI

```
[ex:/configure filter match-list]
A:admin@node-2# info
    ip-prefix-list "list-1" {
        prefix 10.0.0.0/16 { }
        prefix-exclude 10.0.2.0/24 { }
    }
```

Example: classic CLI

```
A:node-2>config>filter>match-list# info
----------------------------------------------
        ip-prefix-list "list-1" create
            prefix 10.0.0.0/16
            prefix-exclude 10.0.2.0/24
        exit
----------------------------------------------
```

Without applying **prefix-exclude**, all eight included subnets should be manually configured in the IP prefix list. The following example shows the manual configuration of an IP prefix list.

Example: MD-CLI

```
[ex:/configure filter match-list]
A:admin@node-2# info
    ip-prefix-list "list-1" {
        prefix 10.0.0.0/16 { }
        prefix 10.0.0.0/23 { }
        prefix 10.0.3.0/24 { }
        prefix 10.0.4.0/22 { }
        prefix 10.0.8.0/21 { }
        prefix 10.0.16.0/20 { }
        prefix 10.0.32.0/19 { }
        prefix 10.0.64.0/18 { }
        prefix 10.0.128.0/17 { }
    }
```

Example: classic CLI

```
A:node-2>config>filter>match-list# info
----------------------------------------------
            ip-prefix-list "list-1" create
                prefix 10.0.0.0/16
                prefix 10.0.0.0/23
                prefix 10.0.3.0/24
                prefix 10.0.4.0/22
                prefix 10.0.8.0/21
                prefix 10.0.16.0/20
                prefix 10.0.32.0/19
                prefix 10.0.64.0/18
                prefix 10.0.128.0/17
            exit
----------------------------------------------
```

This is a time consuming and error-prone task compared to using the **prefix-exclude** command.

The filter resources, consumed in hardware, are identical between the two configurations.

A filter match-list using **prefix-exclude** is mutually exclusive with **apply-path**, and is not supported as a match criterion in CPM filter.

Configured **prefix-exclude** prefixes are ignored when no overlapping larger subnet is configured in the prefix-list. For example: prefix-exclude 1.1.1.1/24 is ignored if the only included subnet is 10.0.0.0/16.

4.1.2.2 Filter policy scope and embedded filters

The system supports four different filter policies:

- scope template
- scope exclusive
- scope embedded
- scope system

Each scope provides different characteristics and capabilities to deploy a filter policy on a single interface, multiple interfaces or optimize the use of system resources or the management of the filter policies when sharing a common set of filter entries.

Template and exclusive

A scope template filter policy can be reused across multiple interfaces. This filter policy uses a single set of resources per line card regardless of how many interfaces use it. Template filter policies used on access interfaces consume resources on line cards where the access interfaces are configured only. A scope template filter policy is the most common type of filter policies configured in a router.

A scope exclusive filter policy defines a filter dedicated to a single interface. An exclusive filter allows the highest level of customization but uses the most resources on the system line cards as it cannot be shared with other interfaces.

Embedded

To simplify the management of filters sharing a common set of filter entries, the user can create a scope embedded filter policy. This filter can then be included in (embedded into) a scope template, scope exclusive, or scope system filter.

Using a scope embedded filter, a common set of filter entries can be updated in a single place and deployed across multiple filter policies. The scope embedded is supported for IPv4 and IPv6 filter policies.

A scope embedded filter policy is not directly downloaded to a line card and cannot be directly referenced in an interface. However, this policy helps the network user provision a common set of rules across different filter policies.

The following rules apply when using a scope embedded filter policy:

- The user explicitly defines the offset at which to insert a filter of scope embedded in a template, exclusive, or system filter. The embedded filter entry-id X becomes entry-id (X + offset) in the main filter.
- Multiple filter scope embedded policies can be included (embedded into) in a single filter policy of scope template, exclusive, or system.
- The same scope embedded filter policy can be included in multiple filter policies of scope template, exclusive, or system.
- Configuration modifications to embedded filter policy entries are automatically applied to all filter policies that embed this filter.
- The system performs a resource management check when a filter policy of scope embedded is updated or embedded in a new filter. If resources are not available, the configuration is rejected. In rare cases, a filter policy resource check may pass but the filter policy can still fail to load because of a resource exhaustion on a line card (for example, when other filter policy entries are dynamically configured by applications like RADIUS in parallel). If that is the case, the embedded filter policy configured is deactivated (configuration is changed from activate to inactivate).
- An embedded filter is never embedded partially in a single filter and resources must exist to embed all the entries in a specific exclusive, template or system filter. However, an embedded filter may be embedded only in a subset of all the filters it is referenced into, only those where there are sufficient resources available.
- Overlapping of filter entries between an embedded filter and a filter of scope template, exclusive or system filter can happen but should be avoided. It is recommended instead that network users use a large enough offset value and an appropriate filter entry-id in the main filter policy to avoid overlapping. In case of overlapping entries, the main filter policy entry overwrites the embedded filter entry.
- Configuring a default action in a filter of scope embedded is not required as this information is not used to embed filter entries.

Figure 24: Embedded Filter Policy shows a configuration with two filter policies of scope template, filter 100 and 200 each embed filter policy 10 at a different offset:

- Filter policy 100 and 200 are of scope template.
- Filter policy 10 of scope embedded is configured with 4 filter entries: entry-id 10, 20, 30, 40.
- Filter policy 100 embed filter 10 at offset 0 and includes two additional static entries with entry-id 20010 and 20020.
- Filter policy 200 embed filter 10 at offset 10000 and includes two additional static entries with entry-id 100 and 110.
- As a result, filter 100 automatically creates entry 10, 20, 30, 40 while filter 200 automatically creates entry 10010, 10020, 10030, 10040. Filter policy 100 and 200 consumed in total 12 entries when both policies are installed in the same line card.

Example: Scope embedded filter configuration (MD-CLI)

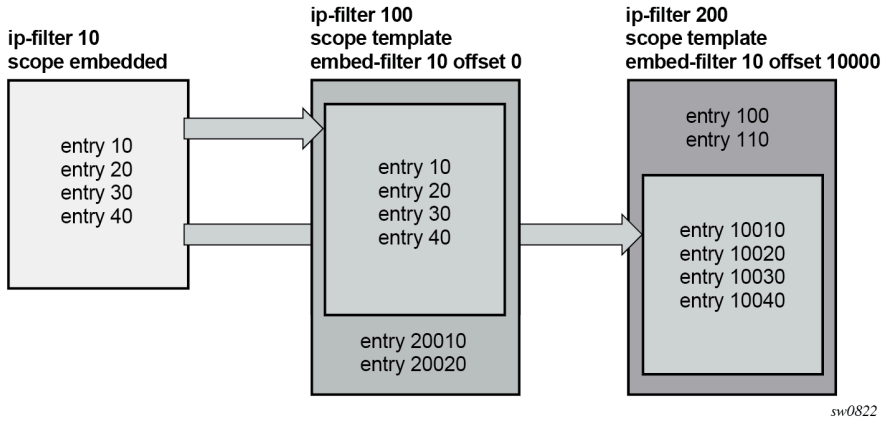
```
[ex:/configure filter]
A:admin@node-2# info
...
    ip-filter "10" {
        scope embedded
        entry 10 {
        }
        entry 20 {
        }
        entry 30 {
        }
        entry 40 {
        }
    }
    ip-filter "100" {
        scope template
        entry 20010 {
        }
        entry 20020 {
        }
        embed {
            filter "10" offset 0 {
            }
        }
    }
    ip-filter "200" {
        scope template
        entry 100 {
        }
        entry 110 {
        }
        embed {
            filter "10" offset 10000 {
            }
        }
    }
```

Example: Scope embedded filter configuration (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
        ip-filter 10 name "10" create
            scope embedded
            entry 10 create
            exit
            entry 20 create
            exit
```

```
            entry 30 create
            exit
            entry 40 create
            exit
        exit
        ip-filter 100 name "100" create
            scope template
            embed-filter 10
            entry 20010 create
            exit
            entry 20020 create
            exit
        exit
        ip-filter 200 name "200" create
            scope template
            embed-filter 10 offset 10000
            entry 100 create
            exit
            entry 110 create
            exit
        exit
-----------------------------------------------
```

Figure 24: Embedded Filter Policy



System

The scope system filter policy provides the most optimized use of hardware resources by programming filter entries after the line cards regardless of how many IPv4 or IPv6 filter policies of scope template or exclusive use this filter. The system filter policy entries are not duplicated inside each policy that uses it, instead, template or exclusive filter policies can be chained to the system filter using the **chain-to-system-filter** command.

When a template of exclusive filter policy is chained to the system filter, system filter rules are evaluated first before any rules of the chaining filter are evaluated (that is chaining filter's rules are only matched against if no system filter match took place).

The system filter policy is intended primarily to deploy a common set of system-level deny rules and infrastructure-level filtering rules to allow, block, or rate limit traffic. Other actions like, for example, PBR actions, or redirect to ISAs should not be used unless the system filter policy is activated only in filters used by services that support such action. The NAT action is not supported and should not be configured. Failure to observe these restrictions can lead to unwanted behavior as system filter actions are not verified

against the services the chaining filters are deployed for. System filter policy entries also cannot be the sources of mirroring.

System filter policies can be populated using CLI, SNMP, NETCONF, OpenFlow and FlowSpec. System filter policy entries cannot be populated using RADIUS or Gx.

The following example shows the configuration of an IPv4 system filter:

- System filter policy 10 includes a single entry to rate limit NTP traffic to the Infrastructure subnets.
- Filter policy 100 of scope template is configured to use the system filter using the **chain-to-system-filter** command.

Example: IPv4 system filter configuration (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
    ip-filter "10" {
        scope system
        entry 10 {
            description "Rate Limit NTP to the Infrastructure"
            match {
                protocol udp
                dst-ip {
                    ip-prefix-list "Infrastructure IPs"
                }
                dst-port {
                    eq 123
                }
            }
            action {
                accept
                rate-limit {
                    pir 2000
                }
            }
        }
    }
    ip-filter "100" {
        description "Filter scope template for network interfaces"
        chain-to-system-filter true
    }
    system-filter {
        ip "10" { }
    }
```

Example: IPv4 system filter configuration (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
        ip-filter 10 name "10" create
            scope system
            entry 10 create
                description "Rate Limit NTP to the Infrastructure"
                match protocol udp
                    dst-ip ip-prefix-list "Infrastructure IPs"
                    dst-port eq 123
                exit
                action
                    rate-limit 2000
                exit
            exit
        exit
```

```
        ip-filter 100 name "100" create
            chain-to-system-filter
            description "Filter scope template for network interfaces"
        exit
        system-filter
            ip 10
        exit
-----------------------------------------------
```

4.1.2.3 Filter policy type

The filter policy type defines the list of match criteria available in a filter policy. It provides filtering flexibility by reallocating the CAM in the line card at the filter policy level to filter traffic using additional match criteria not available using filter type normal. The filter type is specific to the filter policy, it is not a system wide or line card command option. You can configure different filter policy types on different interfaces of the same system and line card.

MAC filter supports three different filter types: normal, ISID, or VID.

IPv4 and IPv6 filters support four different filter types: normal, source MAC, packet length, or destination class.

4.1.2.3.1 IPv4, IPv6 filter type packet-length

The following match criteria are available using packet-length filter type, in addition to the match criteria that are available using the normal filter type:

- **packet length**

  Total packet length including both the IP header and payload for IPv4 and IPv6 ingress and egress filter policies.

- **TTL or hop limit**

  Match criteria available using FP4-based cards and ingress filter policies; if configured on FP2- or FP3-based cards, the TTL or \hop-limit match criteria part of the filter entries are not programmed in the line card.

The following match criteria are not available for filter entries in a packet-length type filter policy:

- **IPv4**

  DSCP, IP option, option present, multiple option, source-route option

- **IPv6**

  flow label

For a QoS policy assigned to the same service or interface endpoint on egress as a packet-length type filter policy, QoS IP criteria cannot use DSCP match criteria with no restriction to ingress.

This filter type is available for both ingress and egress on all service and router interfaces endpoints with the exception of video ISA, service templates, and PW templates.

Dynamic filter entry embedding using OpenFlow and VSD is not supported using this filter type.

4.1.2.3.2 IPv4, IPv6 filter type destination-class

This filter policy provides BGP destination-class value match criterion capability using egress IPv4 and IPv6 filters, and is supported on network, IES, VPRN, and R-VPLS.

The following match criteria from the normal filter type are not available using the destination-class filter type:

- **IPv4**

  DSCP, IP option, option present, multiple option, source-route option

- **IPv6**

  flow label

Filtering egress on destination class requires the **destination-class-lookup** command to be enabled on the interface that the packet ingresses on. For a QoS policy or filter policy assigned to the same interface, the DSCP remarking action is performed only if a destination-class was not identified for this packet.

System filters, as well as dynamic filter embedding using OpenFlow, FlowSpec, and VSD, are not supported using this filter type.

4.1.2.3.3 IPv4 and IPv6 filter type and embedding

IPv4 and IPv6 filter policy of scope embedded must have the same **type** as the main filter policy of scope template, exclusive or system embedding it:

- If this condition is not met the filter cannot be embedded.
- When embedded, the main filter policy cannot change the filter type if one of the embedded filters is of a different type.
- When embedded, the embedded filter cannot change the filter type if it does not match the main filter policy.

Similarly, the system filter **type** must be identical to the template or exclusive filter to allow chaining when using the **chain-to-system-filter** command.

4.1.2.4 Filter policies and dynamic policy-driven interfaces

Filter policy entries can be statically configured using CLI, SNMP, or NETCONF or dynamically created using BGP FlowSpec, OpenFlow, VSD (XMPP).

Dynamic filter entries for FlowSpec, OpenFlow, and VSD can be inserted into an IPv4 or IPv6 filter policy. The filter policy must be either exclusive or a template. Additionally, FlowSpec embedding is supported when using a filter policy that defines system-wide filter rules.

BGP FlowSpec

BGP FlowSpec routes are associated with a specific routing instance (based on the AFI/SAFI and possibly VRF import policies) and can be used to create filter entries in a filter policy dynamically.

Configure FlowSpec embedding using the following contexts:

- **MD-CLI**

```
configure filter ip-filter embed flowspec
configure filter ipv6-filter embed flowspec
```

- **classic CLI**

```
configure filter ip-filter embed-filter flowspec
configure filter ipv6-filter embed-filter flowspec
```

The following rules apply to FlowSpec embedding:

- The user explicitly defines both the offset at which to insert FlowSpec filter entries and the router instance the FlowSpec routes belong to. The embedded FlowSpec filter entry ID is chosen by the system, in accordance with RFC 5575 *Dissemination of Flow Specification Rules*.

  Note: These entry IDs are not necessarily sequential and do not necessarily follow the order at which a rule is received.

- The user can configure the maximum number of FlowSpec filter entries in a specific filter policy at the router or VPRN level using the **ip-filter-max-size** and **ipv6-filter-max-size** commands. This limit defines the boundary for FlowSpec embedding in a filter policy (the offset and maximum number of IPv4 or IPv6 FlowSpec routes).
- When the user configures a template or exclusive filter policy, the router instance defined in the dynamic filter entry for FlowSpec must match the router interface that the filter policy is applied to.
- When using a filter policy that defines system-wide rules, embedding FlowSpec entries from different router instances is allowed and can be applied to any router interfaces.
- See section IPv4/IPv6 filter policy entry match criteria on embedded filter scope for recommendations on filter entry ID spacing and overlapping of entries.

The following information describes the FlowSpec configuration that follows:

- The maximum number of FlowSpec routes in the base router instance is configured for 50,000 entries using the **ip-filter-max-size** command.
- The filter policy 100 (template) is configured to embed FlowSpec routes from the base router instance at offset 100,000. The offset chosen in this example avoids overlapping with statically defined entries in the same policy. In this case, the statically defined entries can use the entry ID range 1-99999 and 149999-2M for defining static entries before or after the FlowSpec filter entries.

The following example shows the FlowSpec configuration.

Example: FlowSpec configuration (MD-CLI)

```
[ex:/configure router "Base"]
A:admin@node-2# info
    flowspec {
        ip-filter-max-size 50000
    }

[ex:/configure filter ip-filter "100"]
A:admin@node-2# info
...
    ip-filter "100" {
        embed {
            flowspec offset 100000 {
                router-instance "Base"
```

```
            }
        }
    }
```

Example: FlowSpec configuration (classic CLI)

```
A:node-2>config>router# info
----------------------------------------------
        flowspec
            ip-filter-max-size 50000
        exit
----------------------------------------------
A:node-2>config>filter# info
----------------------------------------------
        ip-filter 100 name "100" create
            embed-filter flowspec router "Base" offset 100000
        exit
----------------------------------------------
```

OpenFlow

The embedded filter infrastructure is used to insert OpenFlow rules into an existing filter policy. Policy-controlled auto-created filters are re-created on system reboot. Policy controlled filter entries are lost on system reboot and need to be reprogrammed.

VSD

VSD filters are created dynamically using XMPP and managed using a Python script so rules can be inserted into or removed from the correct VSD template or embedded filters. XMPP messages received by the 7705 SAR Gen 2 are passed transparently to the Python module to generate the appropriate CLI. See the *7705 SAR Gen 2 Layer 2 Services and EVPN Guide* for more information about VSD filter provisioning, automation, and Python scripting details.

RADIUS or Diameter for subscriber management:

The user can assign filter policies or filter entries used by a subscriber within a preconfigured filter entry range defined for RADIUS or Diameter. See the filter RADIUS-related commands for more information.

4.1.2.5 Extended action for performing two actions at a time

In some deployment scenarios, for example, to realize service function chaining, users may want to perform a second action in addition to a traffic steering action. SR OS supports this behavior by configuring an extended action for a main action. This functionality is supported for Layer 3 traffic steering (that is, PBR) and specifically for the following main actions:

- forward ESI (Layer 3 version)
- forward LSP
- forward next-hop indirect router
- forward redirect-policy
- forward router
- forward VPRN target

The capability to specify an extended action is also supported in the case of PBR redundancy, for the following actions:

- forward next-hop indirect router
- forward VPRN target BGP next hop

The supported extended action is: **remark dscp**

Use the commands in the following contexts to configure the extended action:

- **MD-CLI**

```
configure filter ip-filter entry action ignore-match
configure filter ip-filter entry action ignore-match
```

- **classic CLI**

```
configure filter ip-filter entry action extended-action
configure filter ipv6-filter entry action extended-action
```

Extended Action Restrictions

For forward LSP and for actions supporting redundancy, the extended action is not performed when the PBR target is down.

4.1.2.6 Destination MAC rewrite when deploying policy-based forwarding

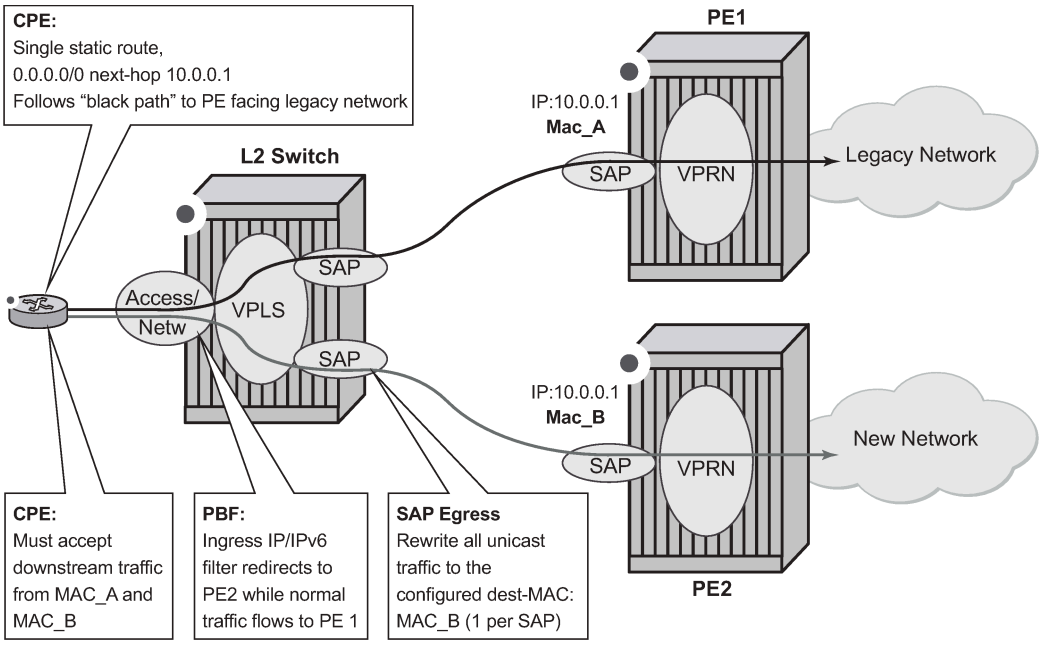
For Layer 2 Policy-Based Forwarding (PBF) redirect actions, a far-end router may discard redirected packets when the PBF changes the destination IP interface the packet arrives on. This happens when a far-end IP interface uses a different MAC address than the IP interface reachable via normal forwarding (for example, one of the routers does not support a configurable MAC address per IP interface).

Use the following command to avoid the discards and deploy egress destination MAC rewrite functionality for VPLS SAPs.

```
configure service vpls sap egress dest-mac-rewrite
```

Figure 25: Layer 2 policy-based forwarding (PBF) redirect action shows a deployment.

Figure 25: Layer 2 policy-based forwarding (PBF) redirect action



When enabled, all unicast packets have their destination MAC rewritten to the user-configured value on a Layer 2 switch VPLS SAP. Multicast and broadcast packets are unaffected. The feature:

- Is supported for regular and split-horizon group Ethernet SAPs in a regular VPLS Service
- Is expected to be deployed on a SAP that faces far-end IP interface (either a SAP that is the target of PBF action, as shown in Figure 25: Layer 2 policy-based forwarding (PBF) redirect action, or a VPLS SAP of a downstream Layer 2 switch that is connected to a far-end router—not shown).
- Applies to any unicast egress traffic including LI and mirror.

Restrictions:

The following command and the SAP MAC ingress and egress loopback feature are mutually exclusive:

- **MD-CLI**

```
tools perform service id loopback eth sap mac-swap
```

- **classic CLI**

```
tools perform service id loopback eth sap start mac-swap
```

4.1.2.7 Network port VPRN filter policy

The network port Layer 3 service-aware filter feature allows users to deploy VPRN service aware ingress filtering on network ports. A single ingress filter of scope template can each be defined for IPv4 and for IPv6 against a VPRN service. The filter applies to all unicast traffic arriving on auto-bind and explicit-spoke network interfaces for that service. The network interface can be either Inter-AS, or Intra-AS. The filter

does not apply to traffic arriving on access interfaces (SAP, spoke SDP, network-ingress (CsC, rVPLS, eVPN).

The same filter can be used on access interfaces of the specific VPRN, can embed other filters (including OpenFlow), can be chained to a system filter, and can be used by other Layer 2 or Layer 3 services.

The filter is deployed on all line cards (chassis network mode D is required). There are no limitations related to filter match/action criteria or embedding. The filter is programmed on line cards against ILM entries for this service. All label-types are supported. If an ILM entry has a filter index programmed, that filter is used when the ILM is used in packet forwarding; otherwise, no filter is used on the service traffic.

Restrictions

Network port Layer 3 service-aware filters do not support FlowSpec and LI (cannot use filter inside LI infrastructure nor have LI sources within the VPRN filter).

4.1.2.8 IP exception filters

IP exception filters scan all outbound traffic entering an NGE domain or an IPsec secured interface and allow packets that match the exception filter criteria to transit the NGE domain or secured interface unencrypted. For information about IP exception filters supported by NGE nodes, see Router encryption exceptions using ACLs.

The most basic IP exception filter policy must have the following:

- an exception filter policy ID
- scope, either exclusive or template
- at least one filter entry with a specified matching criteria

4.1.2.9 Redirect policies

SR OS-based routers support configuring of IPv4 and IPv6 redirect policies. Redirect policies allow specifying multiple redirect target destinations and defining status check test methods used to validate the ability for a destination to receive redirected traffic. This destination monitoring allows routers to react to target destination failures. To specify an IPv4 redirect policy, define all destinations to be IPv4. To specify an IPv6 redirect policy, define all destinations to be IPv6. IPv4 redirect policies can only be deployed in IPv4 filter policies. IPv6 redirect policy can only be deployed in IPv6 filter policies.

Redirect policies support the following destination tests:

- **ping test** with configurable interval, drop-count, and time-out
- **unicast-rt-test** for unicast routing reachability, supported only when the router instance is configured for a specific redirect policy. The test yields true if the route to the specified destination exists in the RTM for the configured router instance.

Each destination is assigned an initial or base priority describing relative importance of this destination within the policy. The destination with the highest priority value is selected as the most-preferred destination and programmed on line cards in filter policies using this redirect policy as an action. Only destinations that are not disabled by the programmed test (if configured) are considered when selecting the most-preferred destination.

In some deployments, it may not be necessary to switch from a currently active, most-preferred redirect-policy destination when a new more-preferred destination becomes available. Use the following command to enable sticky destination functionality to support such deployments.

```
configure filter redirect-policy sticky-dest
```

When enabled, the currently active destination remains active unless it goes down or a user forces the switch. Use the following command to force the switch.

```
tools perform filter redirect-policy activate-best-dest
```

An optional **sticky-dest** hold-time-up value or **no-hold-time-up** command option is available to delay programming the sticky destination in the redirect policy (transition from **action forward** to PBR action to the most-preferred destination). When the timer is enabled, the first destination that comes up is not programmed and instead the timer is started. After the timer expires, the most-preferred destination at that time is programmed (which may be a different destination from the one that started the timer). The following restrictions apply:

- When the manual switchover to most-preferred destination is executed, the *hold-time-up* delay is stopped.
- When the timer value is modified, the new value takes immediate effect and the timer is restarted with the new value (or expired if **no-hold-time-up** is configured).

Note: The **unicast-rt-test** command fails when performed in the context of a VPRN routing instance when the destination is routable only through **grt-leak** functionality. Nokia recommends using the **ping-test** functionality in these cases.

The following restrictions apply to the redirect policy feature:

- Redirect policies are supported for ingress IPv4 and IPv6 filter policies only.
- Different platforms support different scales for redirect policies. Contact Nokia technical support to ensure the planned deployment does not exceed the recommended scale.

4.1.2.9.1 Binding redirect policies

Redirect policies can switch from a specific destination to a new destination in a coordinated manner as opposed to independently as a function of the reachability test results of their configured destinations. Use the commands in following context to bind together destinations of redirect policies.

```
configure filter redirect-policy-binding
```

SR OS combines the reachability test results (either TRUE or FALSE) from each of the bound destinations and forms a master test result which prevails over each independent result. The combined result can be obtained by applying either an AND function or an OR function. For the AND function, all destinations must be UP (reachability test result equals TRUE) for each destination to be considered UP. Conversely, a single destination must be DOWN for each to be considered DOWN; for the OR case, a single destination needs to be UP for each destination to be considered UP. Apart from the master test, which overrides the test result of each destination forming a binding, redirect policies are unaltered. For stickiness capability, switching toward a more-preferred destination in a specified redirect policy does not occur until the timers (if any) of each of the associated destinations have expired.

There is no specific constraint about destinations that can be bound together. For example, it is possible to bind destinations of different address families (IPv4 or IPv6), destinations with no test, destinations with multiple tests, or destinations of redirect policies which are administratively down. However, some specific scenarios exist when binding redirect policies:

- A destination that is in the Administratively down state is considered DOWN (that is, as if its test result was negative, even if no test had been performed).
- An Administratively down redirect policy is equivalent to a policy with all destinations in an Administratively down state. The system performs a simple forward.
- A destination with no test is considered always UP.
- If a destination has multiple tests, all tests must be positive for the destination to be considered UP (logical AND between its own tests results).
- Destination tests are performed even if a redirect policy has not been applied (that is, not declared as an action of a filter which itself has been applied).

4.1.2.10 HTTP redirect (captive portal)

SR OS routers support redirecting HTTP traffic by using the line card ingress IP and IPv6 filter policy action HTTP redirect. This capability is mainly used in the **configure subscriber-mgmt** context to redirect a subscriber web session to a captive portal landing page:

Examples of use cases include redirecting a subscriber after initial connection to a new network to accept the terms of service, or a subscriber out-of-quota redirection.

Traffic matching the HTTP redirect filter entry is sent to the SF/CPM for HTTP redirection:

- The SF/CPM completes the TCP three-way handshake for new TCP sessions on behalf of the intended server, and responds to the HTTP GET request with a 302 redirect. Therefore, the subscriber web session is redirected to the portal landing page configured in the HTTP redirect filter action.
- Non TCP flows are ignored.
- TCP flows other than HTTP, matching an **http-redirect** filter action, are TCP reset after the three-way handshake. Therefore, it is recommended to configure the **http-redirect** filter entry to match only TCP port 80. HTTPs uses TLS as underlying protocol, and cannot be redirected to a landing page.

Additional subscriber information may be required by the captive portal. This information can be appended as variables in the **http-redirect** URL and automatically substituted with the relevant subscriber session data, as follows:

- $IP: subscriber host IP address
- $MAC: subscriber host MAC address
- $URL: original requested URL
- $SAP: subscriber SAP
- $SUB: subscriber identification string
- $CID: circuit-ID, or interface-ID of the subscriber host (hexadecimal format)
- $RID: remote-ID of the subscriber host (hexadecimal format)
- $SAPDESC: configured SAP description

The recommended filter configuration to redirect HTTP traffic page is described in the following information using ingress ip-filter policy "10":

- entry 10: Allows DNS UDP port 53 to a list of allowed DNS servers. Allowing DNS is mandatory for a web client to resolve a URL in the first place. The UDP port directionality indicates DNS request. The destination IP match criteria is optional, creating a list that includes the user DNS, and the most common open DNS servers provide the most security, allowing, alternatively, UDP -port 53 alone is another option.
- entry 20: Allows HTTP TCP port 80 traffic to the portal landing page defined as a prefix-list. The TCP port directionality indicates an HTTP request. Optionally, the user can create an additional entry to allow TCP port 443 in case the landing page uses both HTTP and HTTPS.
- entry 30: Redirects all TCP port 80 traffic, other than entry 20, to the landing page URL http://www.mydomain/com/redirect.html?subscriber=$SUB&ipaddress=$IP&mac=$MAC&location=$SAP .
- entry 40: Drops explicitly any other IP flows, as in the following configuration example.

Example: Redirect HTTP filter configuration (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
   ip-filter "10" {
        entry 10 {
            description "Allow DNS Traffic to DNS servers"
            match {
                protocol udp
                ip {
                ip-prefix-list "dns-servers"
                }
                dst-port {
                    eq 53
                }
            }
            action {
                accept
            }
        }
        entry 20 {
            description "Allow HTTP traffic to redirect portal"
            match {
                protocol tcp
                ip {
                ip-prefix-list "portal-servers"
                }
                dst-port {
                    eq 80
                }
            }
            action {
                accept
            }
        }
        entry 30 {
            description "HTTP Redirect all other TCP 80 flows"
            match {
                protocol tcp
                dst-port {
                    eq 80
                }
            }
            action {
                http-redirect {
                    url "http://www.mydomain/com/redirect.html?
subscriber=$SUB&ipaddress=$IP&mac=$MAC&location=$SAP."
                }
```

```
            }
        }
        entry 40 {
            description "Drop anything else"
            action {
                drop
            }
        }
    }
```

Example: Redirect HTTP filter configuration (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
        ip-filter 10 name "10" create
            entry 10 create
                description "Allow DNS Traffic to DNS servers"
                match protocol udp
                    dst-ip ip-prefix-list "dns-servers"
                    dst-port eq 53
                exit
                action
                    forward
                exit
            exit
            entry 20 create
                description "Allow HTTP traffic to redirect portal"
                match protocol tcp
                    dst-ip ip-prefix-list "portal-servers"
                    dst-port eq 80
                exit
                action
                    forward
                exit
            exit
            entry 30 create
                description "HTTP Redirect all other TCP 80 flows"
                match protocol tcp
                    dst-port eq 80
                exit
                action
                    http-redirect "http://www.mydomain/com/redirect.html?
subscriber=$SUB&ipaddress=$IP&mac=$MAC&location=$SAP."
                exit
            exit
            entry 40 create
                description "Drop anything else"
                action
                    drop
                exit
            exit
        exit
----------------------------------------------
```

Also, the router supports two redirect scale modes that are configurable at the system level. The **optimized-mode** improves the number of HTTP redirect sessions supported by system as compared to if optimized mode is disabled.

Example: Optimized-mode configuration (MD-CLI)

```
[ex:/configure system cpm-http-redirect]
A:admin@node-2# info detail
```

```
 ...
    optimized-mode true
```

Example: Optimized-mode configuration (classic CLI)

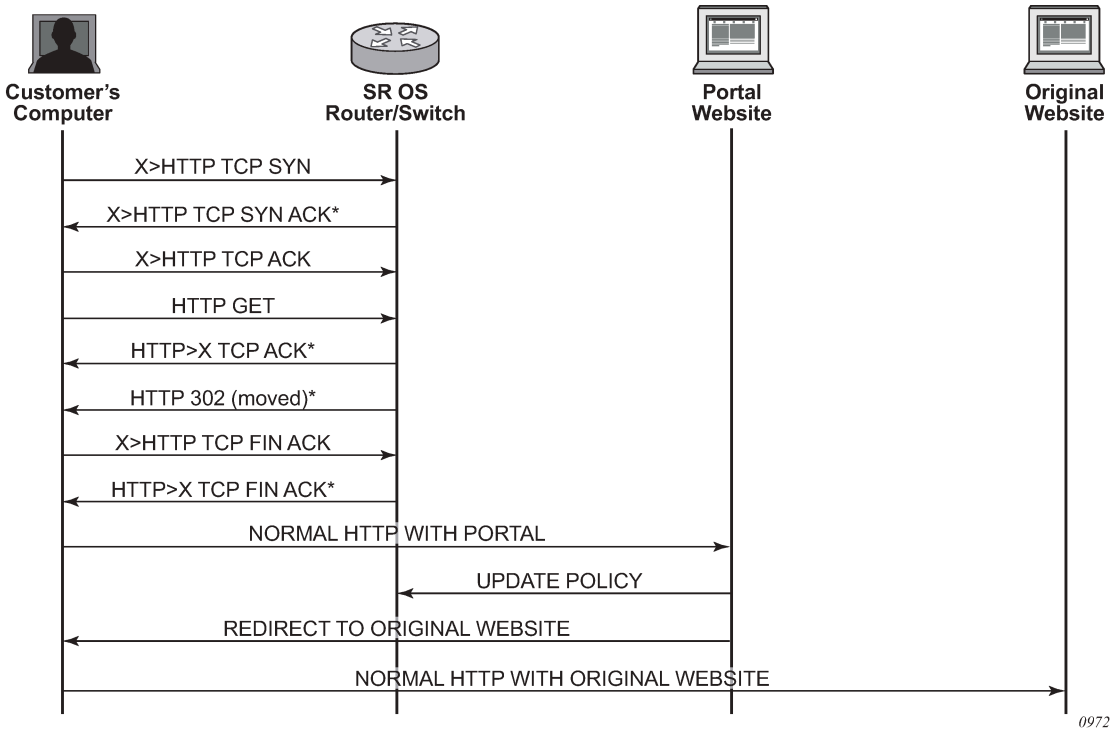
```
A:node-2>config>system>cpm-http-redirect# info detail
----------------------------------------------
            optimized-mode
----------------------------------------------
```

4.1.2.10.1 Traffic flow

The following example provides a brief scenario of a subscriber connecting to a new network, where it is required to authenticate or accept the network terms of use, before getting access to Internet:

1. The subscriber typically receives an IP address upon connecting to the network using DHCP, and is assigned a filter policy to redirect HTTP traffic to a web portal.
2. The subscriber HTTP session TCP traffic is intercepted by the router. The CPM completes the TCP three-way handshake on behalf of the destination HTTP server, and responds to the HTTP request with an HTTP 302 "Moved Temporarily" response. This response contains the URL of the web portal configured in the filter policy.
3. Upon receiving this redirect message, the subscriber web browser closes the original TCP session, and opens a new TCP session to the redirection portal.
4. The subscriber can now authenticate or accept the terms of use. After, the subscriber filter policy is dynamically modified.
5. The subscriber can now connect to the original Internet site.

Figure 26: Web redirect traffic flow



4.2 Configuring filter policies with CLI

This section provides information to configure filter policies using the CLI.

4.2.1 Common configuration tasks

This section provides a brief overview of the tasks that must be performed for all IPv4, IPv6, and MAC filter configurations and provides the CLI commands.

4.2.1.1 Creating an IPv4 filter policy

A filter policy has the following attributes:

- policy ID and policy name
- scope: template, exclusive, embedded, system
- type: normal
- one or more filter entries defining match criteria and action
- default action to define how packets that do not match any of the filter entries are handled

Use the commands in the following context to create a template IPv4 filter policy.

```
configure filter ip-filter
```

4.2.1.1.1 IPv4 filter entry

Within a filter policy, configure filter entries which contain criteria against which ingress, egress traffic is matched. The action specified in the entry determines how the packets are handled, such as drop or forward.

Configure the following to create an IPv4 filter entry:

1. Enter a filter entry ID.
2. Configure the filter matching criteria.
3. Configure the filter action.

The following example shows an IPv4 filter entry configuration.

Example: MD-CLI

```
[ex:/configure filter ip-filter "1"]
A:admin@node-2# info
    description "filter-main"
    scope exclusive
    entry 10 {
        description "no-91"
        match {
            src-ip {
                address 10.10.0.100/24
            }
            dst-ip {
                address 10.10.10.91/24
            }
        }
        action {
            drop
        }
    }
```

Example: classic CLI

```
A:node-2>config>filter>ip-filter# info
----------------------------------------------
            description "filter-main"
            scope exclusive
            entry 10 create
                description "no-91"
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.100/24
                exit
                action drop
            exit
----------------------------------------------
```

4.2.1.2 Creating an IPv6 filter policy

IPv6 filter policy configuration mimics IP filter policy configuration. See Creating an IPv4 filter policy.

4.2.1.3 Creating an IPv4 exception filter policy

Configuring and applying IPv4 exception filter policies is optional. Each exception filter policy must have the following:

- an exception filter policy ID
- scope specified, either exclusive or template
- at least one filter entry with matching criteria specified

4.2.1.3.1 IP exception filter policy

Use the commands in the following context to create an IP exception filter policy.

```
configure filter ip-exception
```

The following example displays a template IP exception filter policy configuration.

Example: MD-CLI

```
[ex:/configure filter]
A:admin@node-2# info
    ip-exception "1" {
        description "IP-exception"
    }
```

Example: classic CLI

```
A:node-2>config>filter# info
----------------------------------------------
...
        ip-exception 1 create
            description "IP-exception"
            scope template
        exit
...
----------------------------------------------
```

4.2.1.3.2 IP exception entry matching criteria

Within an exception filter policy, configure exception entries that contain criteria against which ingress, egress, and network traffic is matched. Packets that match the entry criteria are allowed to transit the NGE domain in cleartext.

Configure the following to create an IP exception entry:

1. Enter an exception filter entry ID. The system does not dynamically assign a value.
2. Specify matching criteria.

Use the commands in the following context to configure the IP exception filter matching criteria.

```
configure filter ip-exception entry match
```

The following example shows an IP exception entry matching criteria configuration.

Example: MD-CLI

```
[ex:/configure filter ip-exception "2"]
A:admin@node-2# info
    description "exception-main"
    entry 1 {
        match {
            src-ip {
                address 10.10.10.10/32
            }
            dst-ip {
                address 10.10.10.91/24
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>filter>ip-except# info
----------------------------------------------
            description "exception-main"
            scope exclusive
            entry 1 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.10/32
                exit
            exit
----------------------------------------------
```

4.2.1.4 Creating an IPv6 exception filter policy

Configuring and applying IPv6 exception filter policies is optional. Each exception filter policy must have the following:

- an exception filter policy ID
- at least one filter entry with matching criteria specified

4.2.1.4.1 IPv6 exception filter policy

Use the commands in the following context to create an IPv6 exception filter policy.

```
configure filter ipv6-exception
```

The following example shows an IPv6 exception filter policy configuration.

Example: MD-CLI

```
*[ex:/configure filter]
```

```
A:admin@node-2# info
    ipv6-exception "1" {
        description "IPv6-exception"
    }
```

Example: classic CLI

```
*A:node-2>config>filter# info
----------------------------------------------
...
        ipv6-exception 1 create
            description "IPv6-exception"
        exit
...
----------------------------------------------
```

4.2.1.4.2 IPv6 exception entry matching criteria

Within an exception filter policy, configure exception entries that contain criteria against which ingress and network traffic is matched. Packets that match the entry criteria are allowed to transit the IPsec domain in cleartext.

Configure the following to create an IPv6 exception entry:

1. Enter an exception filter entry ID. The system does not dynamically assign a value.
2. Specify matching criteria.

Use the commands in the following context to configure IPv6 exception filter matching criteria.

```
configure filter ipv6-exception entry match
```

The following example shows an IPv6 exception entry matching criteria configuration.

Example: MD-CLI

```
[ex:/configure filter ipv6-exception "2"]
A:admin@node-2# info
    description "exception main"
    entry 1 {
        match {
            src-ip {
                address 2001:db8::2/128
            }
            dst-ip {
                address 2001:db8::1/128
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>filter>ipv6-except# info
----------------------------------------------
            description "exception-main"
            entry 1 create
                match
                    dst-ip 2001:db8::1/128
                    src-ip 2001:db8::2/128
```

```
            exit
          exit
-----------------------------------------
```

4.2.1.5 Creating a match list for filter policies

To create a match list, the user must do the following:

1. Specify a type of match list (for example, an IPv4 address prefix list).
2. Define a unique match list name (for example, an IPv4-Deny-List).
3. Specify at least one entry in the list (for example, a valid IPv4 prefix).

The following example shows the IPv4 prefix list configuration and its usage in an IPv4 filter policy.

Example: MD-CLI

```
[ex:/configure filter]
A:admin@node-2# info
...
    match-list {
        ip-prefix-list "IPv4-Deny-List" {
            description "IPv4-Deny-list"
            prefix 10.0.0.0/21 { }
            prefix 10.254.0.0/24 { }
        }
    }
    ip-filter "ip-edge-filter" {
        scope template
        filter-id 10
        entry 10 {
            match {
                src-ip {
                    ip-prefix-list "IPv4-Deny-List"
                }
            }
            action {
                drop
            }
        }
    }
```

Example: classic CLI

```
A:node-2>config>filter# info
----------------------------------------------
      match-list
        ip-prefix-list "IPv4-Deny-List" create
           description "IPv4-Deny-list"
           prefix 10.0.0.0/21
           prefix 10.254.0.0/24
        exit
     exit
     ip-filter 10 name "ip-edge-filter" create
        scope template
        entry 10 create
           match
              src-ip ip-prefix-list "IPv4-Deny-List"
           exit
           action
```

```
            drop
          exit
      exit
exit

-------------------------------------------
```

4.2.1.6 Applying filter policies

Filter policies can be associated with the entities listed in Table 8: Applying filter policies.

Table 8: Applying filter policies

IPv4 and IPv6 Filter Policies	MAC Filter Policies
Epipe SAP, spoke SDP	Epipe SAP, spoke SDP
spoke SDP	—
IES interface SAP, spoke SDP, R-VPLS	—
spoke SDP	—
VPLS mesh SDP, spoke SDP, SAP	VPLS mesh SDP, spoke SDP, SAP
VPRN interface SAP, spoke SDP, R-VPLS, network ingress	—
Network interface	—

4.2.1.6.1 Applying IPv4/IPv6 and MAC filter policies to a service

IP and MAC filter policies are applied by associating them with a SAP or spoke SDP in ingress or egress direction as needed. Filter ID is used to associate an existing filter policy, or if defined, a filter name for that filter policy can be used in the CLI.

Example: IP and MAC filters assigned to an ingress and egress SAP and spoke SDP (MD-CLI)

```
[ex:/configure service epipe "5"]
A:admin@node-2# info
    admin-state enable
...
    spoke-sdp 8:8 {
        ingress {
            filter {
                ip "epipe sap default filter"
            }
        }
        egress {
            filter {
                mac "91"
            }
        }
    }
    sap 1/1/1 {
```

```
        ingress {
            filter {
                ip "10"
            }
        }
        egress {
            filter {
                mac "92"
            }
        }
    }
```

Example: IP and MAC filters assigned to an ingress and egress SAP and spoke SDP (classic CLI)

```
A:node-2>config>service>epipe# info
----------------------------------------------
            sap 1/1/1 create
                ingress
                    filter ip 10
                exit
                egress
                    filter mac 92
                exit
            exit
            spoke-sdp 8:8 create
                ingress
                    filter ip “epipe sap default filter”
                exit
                egress
                    filter mac 91
                exit
            exit
            no shutdown
----------------------------------------------
```

Example: IPv6 filters assigned to an IES service interface (MD-CLI)

```
[ex:/configure service ies "1001"]
A:admin@node-2# info
    admin-state enable
    customer "1"
    interface "testA" {
        sap 2/1/3:0 {
            ingress {
                filter {
                    ipv6 "100"
                }
            }
            egress {
                filter {
                    ipv6 "100"
                }
            }
        }
        ipv4 {
            primary {
                address 192.22.1.1
                prefix-length 24
            }
        }
        ipv6 {
```

```
        }
    }
```

Example: IPv6 filters assigned to an IES service interface (classic CLI)

```
A:node-2>config>service# info
----------------------------------------------
        ies 1001 name "1001" customer 1 create
            interface "testA" create
                address 192.22.1.1/24
                ipv6
                exit
                sap 2/1/3:0 create
                    ingress
                        filter ipv6 100
                    exit
                    egress
                        filter ipv6 100
                    exit
                exit
                no shutdown
            exit
----------------------------------------------
```

4.2.1.6.2 Applying IPv4/IPv6 filter policies to a network port

IP filter policies can be applied to network IPv4 and IPv6 interfaces. MAC filters cannot be applied to network IP interfaces or to routable IES services. Similar to applying filter policies to service, IPv4/IPv6 filter policies are applied to network interfaces by associating a policy with ingress and egress direction as required. Filter ID is used to associate an existing filter policy, or if defined, a filter name for that filter ID policy can be used in the CLI.

Example: IP filter applied to an interface at ingress (MD-CLI)

```
[ex:/configure router "Base"]
A:admin@node-2# info
...
    interface "to-104" {
        port 1/1/1
        egress {
            filter {
                ip "default network egress policy"
            }
        }
        ingress {
            filter {
                ip "10"
            }
        }
        ipv4 {
            primary {
                address 10.0.0.103
                prefix-length 24
            }
        }
    }
...
```

Example: IP filter applied to an interface at ingress (classic CLI)

```
A:node-2>config>router# info
#------------------------------------------
# IP Configuration
#------------------------------------------
...
        interface "to-104"
            address 10.0.0.103/24
            port 1/1/1
            ingress
                filter ip 10
            exit
            egress
                filter ip “default network egress policy”
            exit
        exit
...
#------------------------------------------
```

Example: IPv4 and IPv6 filters applied to an interface at ingress and egress (MD-CLI)

```
[ex:/configure router "Base" interface "test1"]
A:admin@node-2# info
    port 1/1/1
    egress {
        filter {
            ip "2"
            ipv6 "1"
        }
    }
    ingress {
        filter {
            ip "2"
            ipv6 "1"
        }
    }
    ipv6 {
        address 3ffe::101:101 {
            prefix-length 120
        }
    }
```

Example: IPv4 and IPv6 filters applied to an interface at ingress and egress (classic CLI)

```
A:node-2>config>router>if# info
----------------------------------------------
            port 1/1/1
            ipv6
                address 3FFE::101:101/120
            exit
            ingress
                filter ip 2
                filter ipv6 1
            exit
            egress
                filter ip 2
                filter ipv6 1
            exit
----------------------------------------------
```

4.2.1.7 Creating a redirect policy

Configuring and applying redirect policies is optional. Each redirect policy must include the following:

- a destination IP address
- a priority (default is 100)

Configuring a ping test is recommended.

The following example shows the configuration for a redirect policy.

Example: MD-CLI

```
[ex:/configure filter]
A:admin@node-2# info
    redirect-policy "redirect1" {
        admin-state enable
        destination 10.10.10.104 {
            priority 105
        }
        destination 10.10.10.105 {
            admin-state enable
            priority 95
            ping-test {
                timeout 30
                drop-count 5
            }
        }
        destination 10.10.10.106 {
            admin-state enable
            priority 90
        }
    }
```

Example: classic CLI

```
A:node-2>config>filter# info
----------------------------------------------
        redirect-policy "redirect1" create
            destination 10.10.10.104 create
                priority 105
                exit
                no shutdown
            exit
            destination 10.10.10.105 create
                priority 95
                ping-test
                    timeout 30
                    drop-count 5
                exit
                no shutdown
            exit
            destination 10.10.10.106 create
                priority 90
                exit
                no shutdown
            exit
...
----------------------------------------------
```

4.3 Filter management tasks

This section describes filter policy management tasks.

4.3.1 Renumbering filter policy entries

The system exits the matching process when the first match is found and then executes the actions in accordance with the specified action. Because the ordering of entries is important, the numbering sequence may need to be rearranged. Entries should be numbered from the most explicit to the least explicit.

Example: Renumbering filter policy entries (MD-CLI)

```
*[ex:/configure filter ip-filter "11"]
A:admin@node-2# rename entry 10 to 15

*[ex:/configure filter ip-filter "11"]
A:admin@node-2# rename entry 20 to 10

*[ex:/configure filter ip-filter "11"]
A:admin@node-2# rename entry 40 to 1
```

Example: Original filter numbers and updated filter numbers configuration (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
...
    ip-filter "11" {
        description "filter-main"
        scope exclusive
        entry 10 {
            description "no-91"
            interface-sample false
            match {
                src-ip {
                    address 10.10.10.103/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                forward {
                    redirect-policy "redirect1"
                }
            }
        }
        entry 20 {
            match {
                src-ip {
                    address 10.10.0.100/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
```

```
                drop
            }
        }
        entry 30 {
            match {
                src-ip {
                    address 10.10.0.200/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                accept
            }
        }
        entry 40 {
            match {
                src-ip {
                    address 10.10.10.106/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
    }
...
-----------------------------------------------
[ex:/configure filter]
A:admin@node-2# info
...
ip-filter "11" {
        description "filter-main"
        scope exclusive
        entry 1 {
            match {
                src-ip {
                    address 10.10.10.106/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
        entry 10 {
            match {
                src-ip {
                    address 10.10.0.100/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
```

```
        entry 15 {
            description "no-91"
            filter-sample true
            interface-sample false
            match {
                src-ip {
                    address 10.10.10.103/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                forward {
                    redirect-policy "redirect1"
                }
            }
        }
        entry 30 {
            match {
                src-ip {
                    address 10.10.0.200/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                accept
            }
        }
    }
...
```

Example: Renumbering filter policy entries (classic CLI)

```
*A:node-2>config>filter>ip-filter# renum 10 15
*A:node-2>config>filter>ip-filter# renum 20 10
*A:node-2>config>filter>ip-filter# renum 40 1
```

Example: Original filter numbers and updated filter numbers configuration (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
...
        ip-filter 11 create
            description "filter-main"
            scope exclusive
            entry 10 create
                description "no-91"
                filter-sample
                interface-disable-sample
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.103/24
                exit
                action forward redirect-policy redirect1
            exit
            entry 20 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.100/24
```

```
                exit
                action drop
            exit
            entry 30 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.200/24
                exit
                action forward
            exit
            entry 40 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action drop
            exit
        exit
...
-----------------------------------------------
A:node-2>config>filter# info
-----------------------------------------------
...
        ip-filter 11 create
            description "filter-main"
            scope exclusive
            entry 1 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action drop
            exit
            entry 10 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.100/24
                exit
                action drop
            exit
            entry 15 create
                description "no-91"
                filter-sample
                interface-disable-sample
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.103/24
                exit
                action forward redirect-policy
                     redirect1
            exit
            entry 30 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.200/24
                exit
                action forward
            exit
        exit
...
-----------------------------------------------
```

4.3.2 Modifying a filter policy

There are several ways to modify an existing filter policy. A filter policy can be modified dynamically as part of subscriber management dynamic insertion or removal of filter policy entries. A filter policy can be modified indirectly by configuration change to a match list the filter policy uses (as described earlier in this guide). In addition, a filter policy can be directly edited as described in the following information.

To access a specific IPv4, IPv6, or MAC filter, you must specify the filter ID, or if defined, filter name.

Example: Modifying a filter (MD-CLI)

In MD-CLI, you can use **delete** to remove a command option from the configuration.

```
*[ex:/configure filter ip-filter "11"]
A:admin@node-2# description "New IP filter info"

*[ex:/configure filter ip-filter "11"]
A:admin@node-2# entry 2

*[ex:/configure filter ip-filter "11" entry 2]
A:admin@node-2# description "new entry"

*[ex:/configure filter ip-filter "11" entry 2]
A:admin@node-2# action drop

*[ex:/configure filter ip-filter "11" entry 2]
A:admin@node-2# match dst-ip address 10.10.10.104/32

*[ex:/configure filter ip-filter "11" entry 2]
A:admin@node-2# exit
```

Example: Modified IP filter output (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
    ip-filter "11" {
        description "New IP filter info"
        scope exclusive
        entry 1 {
            match {
                src-ip {
                    address 10.10.10.106/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
        entry 2 {
            description "new entry"
            match {
                dst-ip {
                    address 10.10.10.104/32
                }
            }
            action {
                drop
            }
```

```
        }
        entry 10 {
            match {
                src-ip {
                    address 10.10.0.100/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
        entry 15 {
            description "no-91"
            match {
                src-ip {
                    address 10.10.10.103/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                accept
            }
        }
        entry 30 {
            match {
                src-ip {
                    address 10.10.0.200/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                accept
            }
        }
    }
```

Example: Modifying a filter (classic CLI)

In classic CLI you can use the **no** form of the command to remove the command options or return the command option to the default.

```
*A:node-2>config>filter>ip-filter# description "New IP filter info"
*A:node-2>config>filter>ip-filter# entry 2 create
*A:node-2>config>filter>ip-filter>entry$ description "new entry"
*A:node-2>config>filter>ip-filter>entry# action drop
*A:node-2>config>filter>ip-filter>entry# match dst-ip 10.10.10.104/32
*A:node-2>config>filter>ip-filter>entry# exit
```

Example: Modified IP filter output (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
...
        ip-filter 11 create
            description "New IP filter info"
```

```
            scope exclusive
            entry 1 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action drop
            exit
            entry 2 create
                description "new entry"
                match
                    dst-ip 10.10.10.104/32
                exit
                action drop
            exit
            entry 10 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.100/24
                exit
                action drop
            exit
            entry 15 create
                description "no-91"
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.103/24
                exit
                action forward
            exit
            entry 30 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.0.200/24
                exit
                action forward
            exit
        exit
...
-----------------------------------------------
```

4.3.3 Deleting a filter policy

Before deleting a filter, the filter associations must be removed from all the applied ingress and egress SAPs and network interfaces.

Example: Removing the filter from an SAP network interface (MD-CLI)

In MD-CLI, use the **delete** command in all contexts where the filter is used to remove the filter.

```
*[ex:/configure service]
A:admin@node-2# epipe 5

*[ex:/configure service epipe "5"]
A:admin@node-2# sap 1/1/2:3

*[ex:/configure service epipe "5" sap 1/1/2:3]
A:admin@node-2# ingress

*[ex:/configure service epipe "5" sap 1/1/2:3 ingress]
```

```
A:admin@node-2# delete filter
```

After you have removed the filter from the SAPs network interfaces, you can delete the filter. The following example shows the deletion of a filter.

Example: Deleting a filter (MD-CLI)

```
*[ex:/configure filter]
A:admin@node-2# delete ip-filter 11
```

Example: Removing the filter from an SAP network interface (classic CLI)

In classic CLI, use the **no filter** command in all contexts where the filter is used to remove the filter.

```
*A:node-2>config>service# epipe 5
*A:node-2>config>service>epipe# sap 1/1/2:3
*A:node-2>config>service>epipe>sap# ingress
*A:node-2>config>service>epipe>sap>ingress# no filter
```

After you have removed the filter from the SAPs network interfaces, you can delete the filter. The following example shows the deletion of a filter.

Example: Deleting a filter (classic CLI)

```
*A:node-2>config>filter# no ip-filter 11
```

4.3.4 Modifying a redirect policy

To access a specific redirect policy, the policy name must be specified.

Example: Modifying a redirect policy (MD-CLI)

Use the **delete** form of the command to remove the command options or return the command option to the default.

```
*[ex:/configure filter]
A:admin@node-2# redirect-policy redirect1

*[ex:/configure filter redirect-policy "redirect1"]
A:admin@node-2# description "New redirect info"

*[ex:/configure filter redirect-policy "redirect1"]
A:admin@node-2# destination 10.10.10.104

*[ex:/configure filter redirect-policy "redirect1" destination 10.10.10.104]
A:admin@node-2# priority 105

*[ex:/configure filter redirect-policy "redirect1" destination 10.10.10.104]
A:admin@node-2# ping-test timeout 20

*[ex:/configure filter redirect-policy "redirect1" destination 10.10.10.104]
A:admin@node-2# ping-test drop-count 7
```

Example: Modified redirect policy output (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
```

```
    redirect-policy "redirect1" {
        admin-state enable
        description "New redirect info"
        destination 10.10.10.104 {
            admin-state enable
            description "New redirect info"
            priority 105
            ping-test {
                timeout 20
                drop-count 7
            }
        }
        destination 10.10.10.105 {
            admin-state enable
            priority 95
            ping-test {
                timeout 30
                drop-count 5
            }
        }
    }
```

Example: Modifying a redirect policy (classic CLI)

Use the **no** form of the command to remove the command options or return the command option to the default.

```
*A:node-2>config>filter# redirect-policy redirect1
*A:node-2>config>filter>redirect-policy# description "New redirect info"
*A:node-2>config>filter>redirect-policy# destination 10.10.10.104
*A:node-2>config>filter>redirect-policy>dest# priority 105
*A:node-2>config>filter>redirect-policy>dest# ping-test timeout 20
*A:node-2>config>filter>redirect-policy>dest# ping-test drop-count 7
```

Example: Modified redirect policy output (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
...
        redirect-policy "redirect1" create
            description "New redirect info"
            destination 10.10.10.104 create
                priority 105
                ping-test
                    timeout 20
                    drop-count 7
                exit
                no shutdown
            exit
            destination 10.10.10.105 create
                priority 95
                ping-test
                    timeout 30
                    drop-count 5
                exit
                no shutdown
            exit
            no shutdown
        exit
...
----------------------------------------------
```

4.3.5 Deleting a redirect policy

Before a redirect policy can be deleted from the filter configuration, the policy association must be removed from the IP filter.

The following example shows the command usage to replace the configured redirect policy (**redirect1**) with a different redirect policy (**redirect2**) and then removing the **redirect1** policy from the filter configuration.

Example: Replacing and deleting a redirect policy (MD-CLI)

```
*[ex:/configure filter]
A:admin@node-2# ip-filter 11

*[ex:/configure filter ip-filter "11"]
A:admin@node-2# entry 1

*[ex:/configure filter ip-filter "11" entry 1]
A:admin@node-2# action forward redirect-policy redirect2

*[ex:/configure filter ip-filter "11" entry 1]
A:admin@node-2# exit

*[ex:/configure filter ip-filter "11"]
A:admin@node-2# exit

*[ex:/configure filter]
A:admin@node-2# delete redirect-policy redirect1
```

Example: Output after deleting a redirect policy (MD-CLI)

```
[ex:/configure filter ip-filter "11"]
A:admin@node-2# info
    description "This is new"
    scope exclusive
    entry 1 {
        interface-sample false
        match {
            src-ip {
                address 10.10.10.106/24
            }
            dst-ip {
                address 10.10.10.91/24
            }
        }
        action {
            forward {
                redirect-policy "redirect2"
            }
        }
    }
    entry 2 {
        description "new entry"
...
```

Example: Replacing and deleting a redirect policy (classic CLI)

```
*A:node-2>config>filter# ip-filter 11
*A:node-2>config>filter>ip-filter# entry 1
*A:node-2>config>filter>ip-filter>entry# action forward redirect-policy "redirect2"
*A:node-2>config>filter>ip-filter>entry# exit
```

```
*A:node-2>config>filter>ip-filter# exit
*A:node-2>config>filter# no redirect-policy "redirect1"
```

Example: Output after deleting a redirect policy (classic CLI)

```
A:node-2>config>filter>ip-filter# info
----------------------------------------------
            description "This is new"
            scope exclusive
            entry 1 create
               filter-sample
               interface-disable-sample
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action forward redirect-policy redirect2
            exit
            entry 2 create
                description "new entry"
...
----------------------------------------------
```

4.3.6 Copying filter policies

When changes are to be made to an existing filter policy applied to a one or more SAPs or network interfaces, Nokia recommends to first copy the applied filter policy, then modify the copy and then overwrite the applied policy with the modified copy. This ensures that a policy being modified is not applied when partial changes are done as any filter policy edits are applied immediately to all services where the policy is applied.

New filter policies can also be created by copying an existing policy and renaming the new filter.

The following example displays the copying of the configuration information from an existing IP filter policy "11" to create a new filter policy "12" that can then be edited. After edits are completed, they can be used to overwrite existing policy "11".

Example: Copying a filter policy (MD-CLI)

```
*[ex:/configure filter]
A:admin@node-2# copy ip-filter 11 to ip-filter 12
```

Example: Copied filter policy output (MD-CLI)

```
[ex:/configure filter]
A:admin@node-2# info
   ip-filter "11" {
        description "This is new"
        scope exclusive
        entry 1 {
            match {
                src-ip {
                    address 10.10.10.106/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
```

```
            action {
                drop
                }
            }
        }
        entry 2 {
...
    ip-filter "12" {
        description "This is new"
        scope exclusive
        entry 1 {
            match {
                src-ip {
                    address 10.10.10.106/24
                }
                dst-ip {
                    address 10.10.10.91/24
                }
            }
            action {
                drop
            }
        }
        entry 2 {
...
```

Example: Copying a filter policy (classic CLI)

```
*A:node-2>config>filter# copy ip-filter 11 to 12
```

Example: Copied filter policy output (classic CLI)

```
A:node-2>config>filter# info
----------------------------------------------
...
        ip-filter 11 create
            description "This is new"
            scope exclusive
            entry 1 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action drop
            exit
            entry 2 create
...
        ip-filter 12 create
            description "This is new"
            scope exclusive
            entry 1 create
                match
                    dst-ip 10.10.10.91/24
                    src-ip 10.10.10.106/24
                exit
                action drop
            exit
            entry 2 create
...
```

5 Standards and protocol support

Note:
The information provided in this chapter is subject to change without notice and may not apply to all platforms.

Nokia assumes no responsibility for inaccuracies.

5.1 Bidirectional Forwarding Detection (BFD)

RFC 5880, *Bidirectional Forwarding Detection (BFD)*

RFC 5881, *Bidirectional Forwarding Detection (BFD) IPv4 and IPv6 (Single Hop)*

RFC 5882, *Generic Application of Bidirectional Forwarding Detection (BFD)*

5.2 Border Gateway Protocol (BGP)

draft-hares-idr-update-attrib-low-bits-fix-01, *Update Attribute Flag Low Bits Clarification*

draft-ietf-idr-add-paths-guidelines-08, *Best Practices for Advertisement of Multiple Paths in IBGP*

draft-ietf-idr-best-external-03, *Advertisement of the best external route in BGP*

draft-ietf-idr-bgp-gr-notification-01, *Notification Message support for BGP Graceful Restart*

draft-ietf-idr-bgp-optimal-route-reflection-10, *BGP Optimal Route Reflection (BGP-ORR)*

draft-ietf-idr-error-handling-03, *Revised Error Handling for BGP UPDATE Messages*

draft-ietf-idr-link-bandwidth-03, *BGP Link Bandwidth Extended Community*

RFC 1772, *Application of the Border Gateway Protocol in the Internet*

RFC 1997, *BGP Communities Attribute*

RFC 2385, *Protection of BGP Sessions via the TCP MD5 Signature Option*

RFC 2439, *BGP Route Flap Damping*

RFC 2545, *Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*

RFC 2858, *Multiprotocol Extensions for BGP-4*

RFC 2918, *Route Refresh Capability for BGP-4*

RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*

RFC 4360, *BGP Extended Communities Attribute*

RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 4456, *BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*

RFC 4486, *Subcodes for BGP Cease Notification Message*

RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*

RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

RFC 4724, *Graceful Restart Mechanism for BGP* – helper mode

RFC 4760, *Multiprotocol Extensions for BGP-4*

RFC 4798, *Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*

RFC 5004, *Avoid BGP Best Path Transitions from One External to Another*

RFC 5065, *Autonomous System Confederations for BGP*

RFC 5291, *Outbound Route Filtering Capability for BGP-4*

RFC 5396, *Textual Representation of Autonomous System (AS) Numbers* – asplain

RFC 5492, *Capabilities Advertisement with BGP-4*

RFC 5668, *4-Octet AS Specific BGP Extended Community*

RFC 6286, *Autonomous-System-Wide Unique BGP Identifier for BGP-4*

RFC 6368, *Internal BGP as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 6793, *BGP Support for Four-Octet Autonomous System (AS) Number Space*

RFC 6810, *The Resource Public Key Infrastructure (RPKI) to Router Protocol*

RFC 6811, *Prefix Origin Validation*

RFC 6996, *Autonomous System (AS) Reservation for Private Use*

RFC 7311, *The Accumulated IGP Metric Attribute for BGP*

RFC 7606, *Revised Error Handling for BGP UPDATE Messages*

RFC 7607, *Codification of AS 0 Processing*

RFC 7752, *North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*

RFC 7911, *Advertisement of Multiple Paths in BGP*

RFC 7999, *BLACKHOLE Community*

RFC 8092, *BGP Large Communities Attribute*

RFC 8097, *BGP Prefix Origin Validation State Extended Community*

RFC 8212, *Default External BGP (EBGP) Route Propagation Behavior without Policies*

RFC 8277, *Using BGP to Bind MPLS Labels to Address Prefixes*

RFC 9294, *Application-Specific Link Attributes Advertisement Using the Border Gateway Protocol - Link State (BGP LS)*

RFC 9494, *Long-Lived Graceful Restart for BGP*

5.3 Bridging and management

IEEE 802.1AB, *Station and Media Access Control Connectivity Discovery*

IEEE 802.1ad, *Provider Bridges*

IEEE 802.1AX, *Link Aggregation*

IEEE 802.1D, *MAC Bridges*

IEEE 802.1p, *Traffic Class Expediting*

IEEE 802.1Q, *Virtual LANs*

IEEE 802.1s, *Multiple Spanning Trees*

IEEE 802.1w, *Rapid Reconfiguration of Spanning Tree*

5.4 Certificate management

RFC 4210, *Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)*

RFC 4211, *Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)*

RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*

RFC 6712, *Internet X.509 Public Key Infrastructure -- HTTP Transfer for the Certificate Management Protocol (CMP)*

RFC 7030, *Enrollment over Secure Transport*

RFC 7468, *Textual Encodings of PKIX, PKCS, and CMS Structures*

5.5 Ethernet

IEEE 802.3x, *Ethernet Flow Control*

5.6 Ethernet VPN (EVPN)

draft-ietf-bess-evpn-ipvpn-interworking-15, *EVPN Interworking with IPVPN*

draft-ietf-bess-evpn-l3mh-proto-00, *EVPN Multi-Homing support for L3 services*

draft-rbickhart-evpn-ip-mac-proxy-adv-04, *Proxy MAC-IP Advertisement in EVPN*

RFC 7432, *BGP MPLS-Based Ethernet VPN*

RFC 8214, *Virtual Private Wire Service Support in Ethernet VPN*

RFC 8365, *A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)*

RFC 8560, *Seamless Integration of Ethernet VPN (EVPN) with Virtual Private LAN Service (VPLS) and Their Provider Backbone Bridge (PBB) Equivalents*

RFC 9047, *Propagation of ARP/ND Flags in an Ethernet Virtual Private Network (EVPN)*

RFC 9135, *Integrated Routing and Bridging in Ethernet VPN (EVPN)*

RFC 9136, *IP Prefix Advertisement in Ethernet VPN (EVPN)*

RFC 9161, *Operational Aspects of Proxy ARP/ND in Ethernet Virtual Private Networks*

RFC 9251, *Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Proxies for Ethernet VPN (EVPN)*

5.7 gRPC Remote Procedure Calls (gRPC)

cert.proto version 0.1.0, *gNOI Certificate Management Service*

file.proto version 0.1.0, *gNOI File Service*

gnmi.proto version 0.8.0, *gNMI Service Specification*

gnmi_ext.proto, *gNMI Commit Confirmed Extension*

gnmi_ext.proto, *gNMI Config Subscription Extension*

gnmi_ext.proto, *gNMI Depth Extension*

system.proto version 1.0.0, *gNOI System Service*

tunnel.proto version 0.2, *gRPC Tunnel Service*

PROTOCOL-HTTP2, *gRPC over HTTP2*

5.8 Intermediate System to Intermediate System (IS-IS)

draft-ietf-isis-mi-02, *IS-IS Multi-Instance*

draft-ietf-lsr-igp-ureach-prefix-announce-01, *IGP Unreachable Prefix Announcement* – without U-Flag and UP-Flag

draft-kaplan-isis-ext-eth-02, *Extended Ethernet Frame Size Support*

ISO/IEC 10589:2002 Second Edition, *Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473)*

RFC 1195, *Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*

RFC 2973, *IS-IS Mesh Groups*

RFC 3359, *Reserved Type, Length and Value (TLV) Codepoints in Intermediate System to Intermediate System*

RFC 3719, *Recommendations for Interoperable Networks using Intermediate System to Intermediate System (IS-IS)*

RFC 3787, *Recommendations for Interoperable IP Networks using Intermediate System to Intermediate System (IS-IS)*

RFC 5120, *M-ISIS: Multi Topology (MT) Routing in IS-IS*

RFC 5130, *A Policy Control Mechanism in IS-IS Using Administrative Tags*

RFC 5301, *Dynamic Hostname Exchange Mechanism for IS-IS*

RFC 5302, *Domain-wide Prefix Distribution with Two-Level IS-IS*

RFC 5303, *Three-Way Handshake for IS-IS Point-to-Point Adjacencies*

RFC 5304, *IS-IS Cryptographic Authentication*

RFC 5305, *IS-IS Extensions for Traffic Engineering TE*

RFC 5306, *Restart Signaling for IS-IS* – helper mode

RFC 5308, *Routing IPv6 with IS-IS*

RFC 5309, *Point-to-Point Operation over LAN in Link State Routing Protocols*

RFC 5310, *IS-IS Generic Cryptographic Authentication*

RFC 6213, *IS-IS BFD-Enabled TLV*

RFC 6232, *Purge Originator Identification TLV for IS-IS*

RFC 6233, *IS-IS Registry Extension for Purges*

RFC 7775, *IS-IS Route Preference for Extended IP and IPv6 Reachability*

RFC 7794, *IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability* – sections 2.1 and 2.3

RFC 7981, *IS-IS Extensions for Advertising Router Information*

RFC 7987, *IS-IS Minimum Remaining Lifetime*

RFC 8202, *IS-IS Multi-Instance* – single topology

RFC 8570, *IS-IS Traffic Engineering (TE) Metric Extensions* – Min/Max Unidirectional Link Delay metric for flex-algo, RSVP, SR-TE

RFC 8919, *IS-IS Application-Specific Link Attributes*

RFC 9885, *Multi-Part TLVs in IS-IS*

5.9 Internet Protocol (IP) general

RFC 768, *User Datagram Protocol*

RFC 793, *Transmission Control Protocol*

RFC 854, *Telnet Protocol Specifications*

RFC 1350, *The TFTP Protocol (revision 2)*

RFC 2784, *Generic Routing Encapsulation (GRE)*

RFC 3164, *The BSD syslog Protocol*

RFC 4250, *The Secure Shell (SSH) Protocol Assigned Numbers*

RFC 4251, *The Secure Shell (SSH) Protocol Architecture*

RFC 4252, *The Secure Shell (SSH) Authentication Protocol* – publickey, password

RFC 4253, *The Secure Shell (SSH) Transport Layer Protocol*

RFC 4254, *The Secure Shell (SSH) Connection Protocol*

RFC 4511, *Lightweight Directory Access Protocol (LDAP): The Protocol*

RFC 4513, *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms* – TLS

RFC 4632, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*

RFC 5082, *The Generalized TTL Security Mechanism (GTSM)*

RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2* – TLS client, RSA public key

RFC 5289, *TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)*

RFC 5425, *Transport Layer Security (TLS) Transport Mapping for Syslog* – RFC 3164 with TLS

RFC 5656, *Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer* – ECDSA

RFC 5925, *The TCP Authentication Option*

RFC 5926, *Cryptographic Algorithms for the TCP Authentication Option (TCP-AO)*

RFC 6398, *IP Router Alert Considerations and Usage* – MLD

RFC 6528, *Defending against Sequence Number Attacks*

RFC 8446, *The Transport Layer Security (TLS) Protocol Version 1.3*

RFC 8907, *The Terminal Access Controller Access-Control System Plus (TACACS+) Protocol*

5.10 Internet Protocol (IP) multicast

RFC 1112, *Host Extensions for IP Multicasting*

RFC 2236, *Internet Group Management Protocol, Version 2*

RFC 2365, *Administratively Scoped IP Multicast*

RFC 2375, *IPv6 Multicast Address Assignments*

RFC 2710, *Multicast Listener Discovery (MLD) for IPv6*

RFC 3376, *Internet Group Management Protocol, Version 3*

RFC 3446, *Anycast Rendevous Point (RP) mechanism using Protocol Independent Multicast (PIM) and Multicast Source Discovery Protocol (MSDP)*

RFC 3590, *Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*

RFC 3618, *Multicast Source Discovery Protocol (MSDP)*

RFC 3810, *Multicast Listener Discovery Version 2 (MLDv2) for IPv6*

RFC 4541, *Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches*

RFC 4604, *Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast*

RFC 4610, *Anycast-RP Using Protocol Independent Multicast (PIM)*

RFC 4611, *Multicast Source Discovery Protocol (MSDP) Deployment Scenarios*

RFC 5059, *Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)*

RFC 5186, *Internet Group Management Protocol Version 3 (IGMPv3) / Multicast Listener Discovery Version 2 (MLDv2) and Multicast Routing Protocol Interaction*

RFC 5384, *The Protocol Independent Multicast (PIM) Join Attribute Format*

RFC 7761, *Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)*

RFC 8487, *Mtrace Version 2: Traceroute Facility for IP Multicast*

5.11 Internet Protocol (IP) version 4

RFC 791, *Internet Protocol*

RFC 792, *Internet Control Message Protocol*

RFC 826, *An Ethernet Address Resolution Protocol*

RFC 1034, *Domain Names - Concepts and Facilities*

RFC 1035, *Domain Names - Implementation and Specification*

RFC 1191, *Path MTU Discovery* – router specification

RFC 1519, *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*

RFC 1812, *Requirements for IPv4 Routers*

RFC 1918, *Address Allocation for Private Internets*

RFC 2131, *Dynamic Host Configuration Protocol*; Relay only

RFC 2132, *DHCP Options and BOOTP Vendor Extensions* – DHCP

RFC 2401, *Security Architecture for Internet Protocol*

RFC 3021, *Using 31-Bit Prefixes on IPv4 Point-to-Point Links*

RFC 3046, *DHCP Relay Agent Information Option (Option 82)*

RFC 3768, *Virtual Router Redundancy Protocol (VRRP)*

RFC 4884, *Extended ICMP to Support Multi-Part Messages* – ICMPv4 and ICMPv6 Time Exceeded

5.12 Internet Protocol (IP) version 6

RFC 2464, *Transmission of IPv6 Packets over Ethernet Networks*

RFC 2529, *Transmission of IPv6 over IPv4 Domains without Explicit Tunnels*

RFC 3122, *Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification*

RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

RFC 3587, *IPv6 Global Unicast Address Format*

RFC 3596, *DNS Extensions to Support IP version 6*

RFC 3633, *IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6*

RFC 3736, *Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6*

RFC 3971, *SEcure Neighbor Discovery (SEND)*

RFC 4007, *IPv6 Scoped Address Architecture*

RFC 4191, *Default Router Preferences and More-Specific Routes* – Default Router Preference

RFC 4193, *Unique Local IPv6 Unicast Addresses*

RFC 4291, *Internet Protocol Version 6 (IPv6) Addressing Architecture*

RFC 4443, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*

RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)*

RFC 5095, *Deprecation of Type 0 Routing Headers in IPv6*

RFC 5722, *Handling of Overlapping IPv6 Fragments*

RFC 5798, *Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6* – IPv6

RFC 5952, *A Recommendation for IPv6 Address Text Representation*

RFC 6164, *Using 127-Bit IPv6 Prefixes on Inter-Router Links*

RFC 8021, *Generation of IPv6 Atomic Fragments Considered Harmful*

RFC 8200, *Internet Protocol, Version 6 (IPv6) Specification*

5.13 Internet Protocol Security (IPsec)

draft-ietf-ipsec-isakmp-mode-cfg-05, *The ISAKMP Configuration Method*

draft-ietf-ipsec-isakmp-xauth-06, *Extended Authentication within ISAKMP/Oakley (XAUTH)*

RFC 2401, *Security Architecture for the Internet Protocol*

RFC 2403, *The Use of HMAC-MD5-96 within ESP and AH*

RFC 2404, *The Use of HMAC-SHA-1-96 within ESP and AH*

RFC 2405, *The ESP DES-CBC Cipher Algorithm With Explicit IV*

RFC 2406, *IP Encapsulating Security Payload (ESP)*

RFC 2407, *IPsec Domain of Interpretation for ISAKMP (IPsec DoI)*

RFC 2408, *Internet Security Association and Key Management Protocol (ISAKMP)*

RFC 2409, *The Internet Key Exchange (IKE)*

RFC 2410, *The NULL Encryption Algorithm and Its Use With IPsec*

RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*

RFC 3526, *More Modular Exponential (MODP) Diffie-Hellman group for Internet Key Exchange (IKE)*

RFC 3566, *The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec*

RFC 3602, *The AES-CBC Cipher Algorithm and Its Use with IPsec*

RFC 3706, *A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers*

RFC 3947, *Negotiation of NAT-Traversal in the IKE*

RFC 3948, *UDP Encapsulation of IPsec ESP Packets*

RFC 4106, *The Use of Galois/Counter Mode (GCM) in IPsec ESP*

RFC 4109, *Algorithms for Internet Key Exchange version 1 (IKEv1)*

RFC 4301, *Security Architecture for the Internet Protocol*

RFC 4303, *IP Encapsulating Security Payload*

RFC 4307, *Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)*

RFC 4308, *Cryptographic Suites for IPsec*

RFC 4434, *The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)*

RFC 4543, *The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH*

RFC 4754, *IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)*

RFC 4835, *Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)*

RFC 4868, *Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec*

RFC 4945, *The Internet IP Security PKI Profile of IKEv1/ISAKMP, IKEv2 and PKIX*

RFC 5019, *The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments*

RFC 5282, *Using Authenticated Encryption Algorithms with the Encrypted Payload of the IKEv2 Protocol*

RFC 5903, *ECP Groups for IKE and IKEv2*

RFC 5996, *Internet Key Exchange Protocol Version 2 (IKEv2)*

RFC 5998, *An Extension for EAP-Only Authentication in IKEv2*

RFC 6379, *Suite B Cryptographic Suites for IPsec*

RFC 6380, *Suite B Profile for Internet Protocol Security (IPsec)*

RFC 6960, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*

RFC 7296, *Internet Key Exchange Protocol Version 2 (IKEv2)*

RFC 7321, *Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)*

RFC 7383, *Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation*

RFC 7427, *Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)*

RFC 8784, *Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security*

5.14 Label Distribution Protocol (LDP)

draft-pdutta-mpls-ldp-adj-capability-00, *LDP Adjacency Capabilities*

draft-pdutta-mpls-ldp-v2-00, *LDP Version 2*

RFC 3037, *LDP Applicability*

RFC 3478, *Graceful Restart Mechanism for Label Distribution Protocol* – helper mode

RFC 5036, *LDP Specification*

RFC 5283, *LDP Extension for Inter-Area Label Switched Paths (LSPs)*

RFC 5443, *LDP IGP Synchronization*

RFC 5561, *LDP Capabilities*

RFC 5919, *Signaling LDP Label Advertisement Completion*

5.15 Multiprotocol Label Switching (MPLS)

RFC 3031, *Multiprotocol Label Switching Architecture*

RFC 3032, *MPLS Label Stack Encoding*

RFC 3270, *Multi-Protocol Label Switching (MPLS) Support of Differentiated Services* – E-LSP

RFC 3443, *Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks*

RFC 5332, *MPLS Multicast Encapsulations*

RFC 5884, *Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)*

RFC 6424, *Mechanism for Performing Label Switched Path Ping (LSP Ping) over MPLS Tunnels*

RFC 7308, *Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)*

RFC 7746, *Label Switched Path (LSP) Self-Ping*

5.16 Network Address Translation (NAT)

RFC 4787, *Network Address Translation (NAT) Behavioral Requirements for Unicast UDP*

RFC 5382, *NAT Behavioral Requirements for TCP*

RFC 5508, *NAT Behavioral Requirements for ICMP*

5.17 Network Configuration Protocol (NETCONF)

RFC 5277, *NETCONF Event Notifications*

RFC 6020, *YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)*

RFC 6022, *YANG Module for NETCONF Monitoring*

RFC 6241, *Network Configuration Protocol (NETCONF)*

RFC 6242, *Using the NETCONF Protocol over Secure Shell (SSH)*

RFC 6243, *With-defaults Capability for NETCONF*

RFC 8071, *NETCONF Call Home and RESTCONF Call Home* – NETCONF

RFC 8342, *Network Management Datastore Architecture (NMDA)* – Startup, Candidate, Running and Intended datastores

RFC 8525, *YANG Library*

RFC 8526, *NETCONF Extensions to Support the Network Management Datastore Architecture* – <get-data> operation

5.18 Media sanitization

NIST Special Publication 800-88 Revision 1, *Guidelines for Media Sanitization* – CF, MMC, SSD, SD, USB

5.19 Open Shortest Path First (OSPF)

RFC 1765, *OSPF Database Overflow*

RFC 2328, *OSPF Version 2*

RFC 3101, *The OSPF Not-So-Stubby Area (NSSA) Option*

RFC 3509, *Alternative Implementations of OSPF Area Border Routers*

RFC 3623, *Graceful OSPF Restart Graceful OSPF Restart* – helper mode

RFC 3630, *Traffic Engineering (TE) Extensions to OSPF Version 2*

RFC 4811, *OSPF Out-of-Band Link State Database (LSDB) Resynchronization* – OSPFv2

RFC 4812, *OSPF Restart Signaling* – OSPFv2

RFC 4576, *Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 4577, *OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*

RFC 5185, *OSPF Multi-Area Adjacency*

RFC 5243, *OSPF Database Exchange Summary List Optimization*

RFC 5250, *The OSPF Opaque LSA Option*

RFC 5309, *Point-to-Point Operation over LAN in Link State Routing Protocols*

RFC 5642, *Dynamic Hostname Exchange Mechanism for OSPF*

RFC 6549, *OSPFv2 Multi-Instance Extensions*

RFC 6987, *OSPF Stub Router Advertisement*

RFC 7471, *OSPF Traffic Engineering (TE) Metric Extensions* – Min/Max Unidirectional Link Delay metric for flex-algo, RSVP, SR-TE

RFC 7684, *OSPFv2 Prefix/Link Attribute Advertisement*

RFC 7770, *Extensions to OSPF for Advertising Optional Router Capabilities*

RFC 8920, *OSPF Application-Specific Link Attributes*

5.20 Path Computation Element Protocol (PCEP)

draft-alvarez-pce-path-profiles-04, *PCE Path Profiles*

draft-ietf-pce-binding-label-sid-15, *Carrying Binding Label/Segment Identifier (SID) in PCE-based Networks.* – MPLS binding SIDs

RFC 5440, *Path Computation Element (PCE) Communication Protocol (PCEP)*

RFC 8231, *Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE*

RFC 8253, *PCEPS: Usage of TLS to Provide a Secure Transport for the Path Computation Element Communication Protocol (PCEP)*

RFC 8281, *PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model*

RFC 8408, *Conveying Path Setup Type in PCE Communication Protocol (PCEP) Messages*

RFC 8664, *Path Computation Element Communication Protocol (PCEP) Extensions for Segment Routing*

5.21 Pseudowire (PW)

draft-ietf-l2vpn-vpws-iw-oam-04, *OAM Procedures for VPWS Interworking*

MFA Forum 12.0.0, *Multiservice Interworking - Ethernet over MPLS*

MFA Forum 13.0.0, *Fault Management for Multiservice Interworking v1.0*

MFA Forum 16.0.0, *Multiservice Interworking - IP over MPLS*

RFC 3916, *Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)*

RFC 3985, *Pseudo Wire Emulation Edge-to-Edge (PWE3)*

RFC 4385, *Pseudo Wire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN*

RFC 4446, *IANA Allocations for Pseudowire Edge to Edge Emulation (PWE3)*

RFC 4447, *Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)*

RFC 4448, *Encapsulation Methods for Transport of Ethernet over MPLS Networks*

RFC 5085, *Pseudowire Virtual Circuit Connectivity Verification (VCCV): A Control Channel for Pseudowires*

RFC 5659, *An Architecture for Multi-Segment Pseudowire Emulation Edge-to-Edge*

RFC 5885, *Bidirectional Forwarding Detection (BFD) for the Pseudowire Virtual Circuit Connectivity Verification (VCCV)*

RFC 6073, *Segmented Pseudowire*

RFC 6310, *Pseudowire (PW) Operations, Administration, and Maintenance (OAM) Message Mapping*

RFC 6391, *Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network*

RFC 6575, *Address Resolution Protocol (ARP) Mediation for IP Interworking of Layer 2 VPNs*

RFC 6718, *Pseudowire Redundancy*

RFC 6829, *Label Switched Path (LSP) Ping for Pseudowire Forwarding Equivalence Classes (FECs) Advertised over IPv6*

RFC 6870, *Pseudowire Preferential Forwarding Status bit*

RFC 7023, *MPLS and Ethernet Operations, Administration, and Maintenance (OAM) Interworking*

RFC 7267, *Dynamic Placement of Multi-Segment Pseudowires*

RFC 7392, *Explicit Path Routing for Dynamic Multi-Segment Pseudowires* – ER-TLV and ER-HOP IPv4 Prefix

RFC 8395, *Extensions to BGP-Signaled Pseudowires to Support Flow-Aware Transport Labels*

5.22 Quality of Service (QoS)

RFC 2430, *A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)*

RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*

RFC 2597, *Assured Forwarding PHB Group*

RFC 3140, *Per Hop Behavior Identification Codes*

RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*

5.23 Remote Authentication Dial In User Service (RADIUS)

draft-oscca-cfrg-sm3-02, *The SM3 Cryptographic Hash Function*

RFC 2865, *Remote Authentication Dial In User Service (RADIUS)*

RFC 2866, *RADIUS Accounting*

RFC 3162, *RADIUS and IPv6*

RFC 6613, *RADIUS over TCP* – with TLS

RFC 6614, *Transport Layer Security (TLS) Encryption for RADIUS*

RFC 6929, *Remote Authentication Dial-In User Service (RADIUS) Protocol Extensions*

5.24 Resource Reservation Protocol - Traffic Engineering (RSVP-TE)

RFC 2702, *Requirements for Traffic Engineering over MPLS*

RFC 2747, *RSVP Cryptographic Authentication*

RFC 2961, *RSVP Refresh Overhead Reduction Extensions*

RFC 3097, *RSVP Cryptographic Authentication -- Updated Message Type Value*

RFC 3209, *RSVP-TE: Extensions to RSVP for LSP Tunnels*

RFC 4124, *Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering*

RFC 4125, *Maximum Allocation Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*

RFC 4127, *Russian Dolls Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering*

RFC 4561, *Definition of a Record Route Object (RRO) Node-Id Sub-Object*

RFC 5817, *Graceful Shutdown in MPLS and Generalized MPLS Traffic Engineering Networks*

5.25 Routing Information Protocol (RIP)

RFC 1058, *Routing Information Protocol*

RFC 2080, *RIPng for IPv6*

RFC 2082, *RIP-2 MD5 Authentication*

RFC 2453, *RIP Version 2*

5.26 Segment Routing (SR)

RFC 8287, *Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes*

RFC 8426, *Recommendations for RSVP-TE and Segment Routing (SR) Label Switched Path (LSP) Coexistence*

RFC 8476, *Signaling Maximum SID Depth (MSD) Using OSPF* – node MSD

RFC 8491, *Signaling Maximum SID Depth (MSD) Using IS-IS* – node MSD

RFC 8660, *Segment Routing with the MPLS Data Plane*

RFC 8661, *Segment Routing MPLS Interworking with LDP*

RFC 8665, *OSPF Extensions for Segment Routing*

RFC 8667, *IS-IS Extensions for Segment Routing*

RFC 8669, *Segment Routing Prefix Segment Identifier Extensions for BGP*

RFC 9256, *Segment Routing Policy Architecture*

RFC 9350, *IGP Flexible Algorithm*

5.27 Simple Network Management Protocol (SNMP)

draft-blumenthal-aes-usm-04, *The AES Cipher Algorithm in the SNMP's User-based Security Model* – CFB128-AES-192 and CFB128-AES-256

draft-ietf-isis-wg-mib-06, *Management Information Base for Intermediate System to Intermediate System (IS-IS)*

draft-ietf-mpls-ldp-mib-07, *Definitions of Managed Objects for the Multiprotocol Label Switching, Label Distribution Protocol (LDP)*

draft-ietf-mpls-lsr-mib-06, *Multiprotocol Label Switching (MPLS) Label Switching Router (LSR) Management Information Base Using SMIv2*

draft-ietf-mpls-te-mib-04, *Multiprotocol Label Switching (MPLS) Traffic Engineering Management Information Base*

draft-ietf-ospf-mib-update-08, *OSPF Version 2 Management Information Base*

draft-ietf-vrrp-unified-mib-06, *Definitions of Managed Objects for the VRRP over IPv4 and IPv6* – IPv6

ESO-CONSORTIUM-MIB revision 200406230000Z, *esoConsortiumMIB*

IANA-ADDRESS-FAMILY-NUMBERS-MIB revision 200203140000Z, *ianaAddressFamilyNumbers*

IANAifType-MIB revision 200505270000Z, *ianaifType*

IANA-RTPROTO-MIB revision 200009260000Z, *ianaRtProtoMIB*

IEEE8021-CFM-MIB revision 200706100000Z, *ieee8021CfmMib*

IEEE8021-PAE-MIB revision 200101160000Z, *ieee8021paeMIB*

IEEE8023-LAG-MIB revision 200006270000Z, *lagMIB*

LLDP-MIB revision 200505060000Z, *lldpMIB*

RFC 1157, *A Simple Network Management Protocol (SNMP)*

RFC 1212, *Concise MIB Definitions*

RFC 1215, *A Convention for Defining Traps for use with the SNMP*

RFC 1724, *RIP Version 2 MIB Extension*

RFC 1901, *Introduction to Community-based SNMPv2*

RFC 2021, *Remote Network Monitoring Management Information Base Version 2 using SMIv2*

RFC 2206, *RSVP Management Information Base using SMIv2*

RFC 2578, *Structure of Management Information Version 2 (SMIv2)*

RFC 2579, *Textual Conventions for SMIv2*

RFC 2580, *Conformance Statements for SMIv2*

RFC 2787, *Definitions of Managed Objects for the Virtual Router Redundancy Protocol*

RFC 2819, *Remote Network Monitoring Management Information Base*

RFC 2856, *Textual Conventions for Additional High Capacity Data Types*

RFC 2863, *The Interfaces Group MIB*

RFC 2864, *The Inverted Stack Table Extension to the Interfaces Group MIB*

RFC 2933, *Internet Group Management Protocol MIB*

RFC 3014, *Notification Log MIB*

RFC 3273, *Remote Network Monitoring Management Information Base for High Capacity Networks*

RFC 3410, *Introduction and Applicability Statements for Internet Standard Management Framework*

RFC 3430, *Simple Network Management Protocol (SNMP) over Transmission Control Protocol (TCP) Transport Mapping*

RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*

RFC 3412, *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)*

RFC 3413, *Simple Network Management Protocol (SNMP) Applications*

RFC 3414, *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*

RFC 3415, *View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*

RFC 3416, *Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)*

RFC 3417, *Transport Mappings for the Simple Network Management Protocol (SNMP)* – SNMP over UDP over IPv4

RFC 3418, *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*

RFC 3419, *Textual Conventions for Transport Addresses*

RFC 3434, *Remote Monitoring MIB Extensions for High Capacity Alarms*

RFC 3584, *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*

RFC 3593, *Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals*

RFC 3635, *Definitions of Managed Objects for the Ethernet-like Interface Types*

RFC 3826, *The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model*

RFC 3877, *Alarm Management Information Base (MIB)*

RFC 4001, *Textual Conventions for Internet Network Addresses*

RFC 4022, *Management Information Base for the Transmission Control Protocol (TCP)*

RFC 4113, *Management Information Base for the User Datagram Protocol (UDP)*

RFC 4273, *Definitions of Managed Objects for BGP-4*

RFC 4292, *IP Forwarding Table MIB*

RFC 4293, *Management Information Base for the Internet Protocol (IP)*

RFC 4878, *Definitions and Managed Objects for Operations, Administration, and Maintenance (OAM) Functions on Ethernet-Like Interfaces*

RFC 7420, *Path Computation Element Communication Protocol (PCEP) Management Information Base (MIB) Module*

RFC 7630, *HMAC-SHA-2 Authentication Protocols in the User-based Security Model (USM) for SNMPv3*

5.28 Timing

RFC 3339, *Date and Time on the Internet: Timestamps*

RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*

RFC 8573, *Message Authentication Code for the Network Time Protocol*

5.29 Two-Way Active Measurement Protocol (TWAMP)

RFC 5357, *A Two-Way Active Measurement Protocol (TWAMP)* – server, unauthenticated mode

RFC 5938, *Individual Session Control Feature for the Two-Way Active Measurement Protocol (TWAMP)*

RFC 6038, *Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features*

RFC 8545, *Well-Known Port Assignments for the One-Way Active Measurement Protocol (OWAMP) and the Two-Way Active Measurement Protocol (TWAMP)* – TWAMP

RFC 8762, *Simple Two-Way Active Measurement Protocol* – unauthenticated

RFC 8972, *Simple Two-Way Active Measurement Protocol Optional Extensions* – unauthenticated

RFC 9503, *Simple Two-Way Active Measurement Protocol (STAMP) Extensions for Segment Routing Networks* – excluding Sections 3, 4.1.2 and 4.1.3

5.30 Virtual Private LAN Service (VPLS)

RFC 4761, *Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling*

RFC 4762, *Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*

RFC 5501, *Requirements for Multicast Support in Virtual Private LAN Services*

RFC 6074, *Provisioning, Auto-Discovery, and Signaling in Layer 2 Virtual Private Networks (L2VPNs)*

RFC 7117, *Multicast in Virtual Private LAN Service (VPLS)*

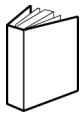
5.31 Yet Another Next Generation (YANG)

RFC 6991, *Common YANG Data Types*

RFC 7950, *The YANG 1.1 Data Modeling Language*

RFC 7951, *JSON Encoding of Data Modeled with YANG*

Customer document and product support



Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback