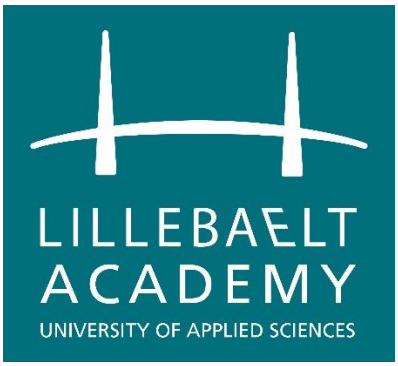
IT Technology
Project Network report



LILLEBAELT ACADEMY
UNIVERSITY OF APPLIED SCIENCE

Author
Michal Skorczewski – mich74p0@edu.eal.dk
Martin Grønholdt - mart80c7@edu.eal.dk

5 June 2017

Table of Contents

1 Introduction

1.1 Introduction

During the second semester, our group was working on SRX Juniper Devices. The final goal was to have configurations about important networking subjects listed below. This document describes time frames, project plan and topics from project plan.

1.2 Project Plan

Project plan was given by EAL teacher Peter Liljehof Thomsen.

Task Name	Duration	Start	Finish
Project Network 2. Semester	**31.14 days?**	**Mon 1/9/17**	**Mon 5/8/17**
Stage-1: Ethernet L2 & L3 Switching	**9 days**	**Mon 1/9/17**	**Tue 2/14/17**
VLAN Implementation	1 day	Mon 1/9/17	Tue 1/10/17
VLAN Trunking (802.1q)	1 day	Tue 1/10/17	Mon 1/16/17
VLAN L3-Interface	1 day	Mon 1/16/17	Tue 1/17/17
Virtual Routers (SRX & EX)	2 days	Tue 1/17/17	Tue 1/24/17
Ethernet OAM	1 day	Tue 1/24/17	Mon 1/30/17
Troubleshooting & Monitoring	1 day	Mon 1/30/17	Tue 1/31/17
Stage-2: Intermediate Routing	**15 days**	**Tue 1/31/17**	**Mon 4/3/17**
IPv6	2 days	Tue 1/31/17	Tue 2/14/17
OSPF	2 days	Tue 2/14/17	Tue 2/21/17
IS-IS	2 days	Tue 2/21/17	Tue 2/28/17
Route Re-Distribution (OSPF/IS-IS)	3 days	Tue 2/28/17	Tue 3/7/17
BGP (iBGP & eBGP w. OSPF)	3 days	Tue 3/7/17	Mon 3/20/17
Route Redistribution (BGP/OSPF)	3 days	Tue 3/21/17	Mon 4/3/17
Stage-3: Security	**12 days**	**Mon 4/3/17**	**Tue 5/16/17**
Routing Policies	2 days	Mon 4/3/17	Mon 4/10/17
Route Redistribution	2 days	Mon 4/10/17	Mon 4/17/17
RE/PFE	2 days	Mon 4/17/17	Tue 4/25/17
Firewall Filters	3 days	Tue 4/25/17	Mon 5/8/17
CoS	3 days	Mon 5/8/17	Tue 5/16/17
Finalize Report	4 days	Tue 5/16/17	Mon 6/5/17
Project End/Hand-in	0 days	Mon 6/5/17	Mon 6/5/17

1.3 Responsibilities

Vlan Implementation	Michal Skorczewski
Vlan Trunking (802.1q)	Michal Skorczewski
Vlan L3- Interface	Michal Skorczewski
Vurtual Routers	Michal Skorczewski
Ethernet OAM	Michal Skorczewski
Troubleshooting & Monitoring	Michal Skorczewski
Ipv6	Martin Gronholdt
OSPF	Martin Gronholdt
IS-IS	Martin Gronholdt
Route Re-distribution OSPF/IS-IS	Martin Gronholdt
BGP	Michal Skorczewski
Router Redistribution BGP/OSPF	Michal Skorczewski
Routing Policies	Michal Skorczewski
Route Redistribution	Martin Gronholdt
RE/PFE	Michal Skorczewski
Firewall Filters	Martin Gronholdt
CoS	Michal Skorczewski

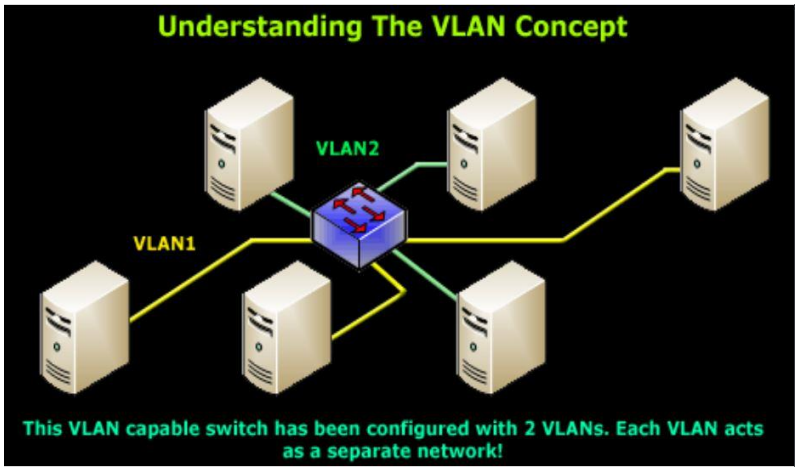
2 Ethernet L2 & L3 Switching

(Full configuration files are on GitHub: https://github.com/miskor/Project_network/tree/master/APPENDIX/Stage-1%20Ethernet%20L2%20and%20L3%20Switching)

Ethernet LANs were created for small network that mostly carried text, over years the type of data carried by LANs grew to include larger types of data like voice, graphics and video. This complex data, combined with the speed of transmission became too much of a load for the original Ethernet LAN design, packet collisions were slowing down larger LANs. The IEEE 802.1D-2004 standard helped evolve LANs to cope with the higher data and transmission requirements by defining the concept of bridging.

- Bridging divides a single physical LAN (now broadcast domain) into more than one virtual LANs
- By default, system on one VLAN don't see the traffic associated with systems on other VLANs on the same network

IEEE 802.1Q is the standard defining VLANs. Each VLAN is identified by a unique 802.1Q ID, only IDs 1 through 4094 can be assigned to VLANs during configuration, IDs 0 and 4095 are reserved by Junos OS and cannot be assigned.



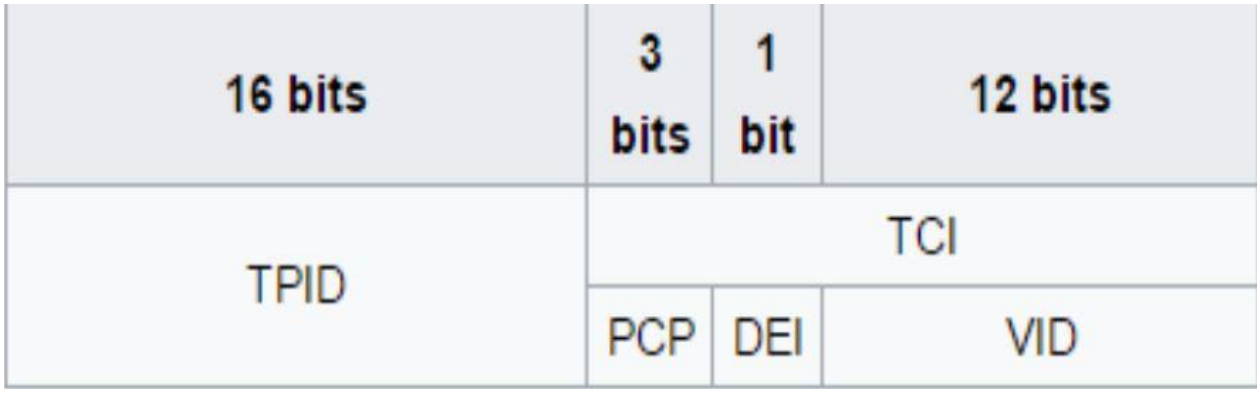
Ethernet packets includes:

- Tag protocol identifier
- EtherType field, which identifies which protocol is transported. When a device with configured VLAN generates a packet, this field includes a value of 0x8100, which means that the packet is a VLAN-tagged packet.
- The packet also has a VLAN ID field that includes the unique 802.1Q ID, which identifies the VLAN to which the packet belongs

The Ethernet Frame + 802.1q Tag

Layer	Preamble	Start of frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)	Payload	Frame check sequence (32-bit CRC)	Interpacket gap
	7 octets	1 octet	6 octets	6 octets	(4 octets)	2 octets	46(42)[b]–1500 octets	4 octets	12 octets
Layer 2 Ethernet frame			**Wireshark Display**			← 64–1518(1522) octets →			
Layer 1 Ethernet packet	**Complete Packet**					← 72–1526(1530) octets →			

The “layer 1 Ethernet Packet” is everything what is transmitted.
The “layer 2 Ethernet Frame” is the display on monitor interface.



The 802.1q tag:
Tag protocol ID (TPID): a 16-bit field set to a value of 0x8100

Tag control information (TCI)

- Priority code point (PCP): 3-bit field which refers to the IEEE 802.1p class of service and maps to the frame priority
- Drop eligible indicator (DEI): 1 bit field, may be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion
- VLAN identifier (VID) a 12 bit field specifying the VLAN to which the frame belongs

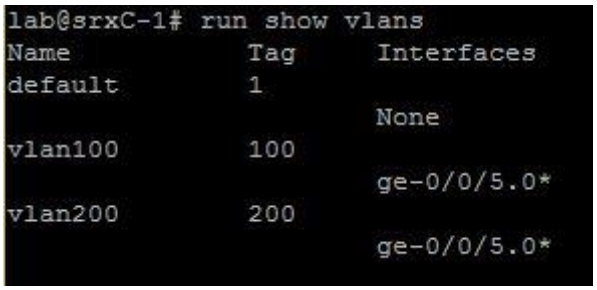
2.1 VLAN Implementation

Step 1: Create a layer 2 vlan
set vlans <vlan-name> vlan-id <vlan-id>

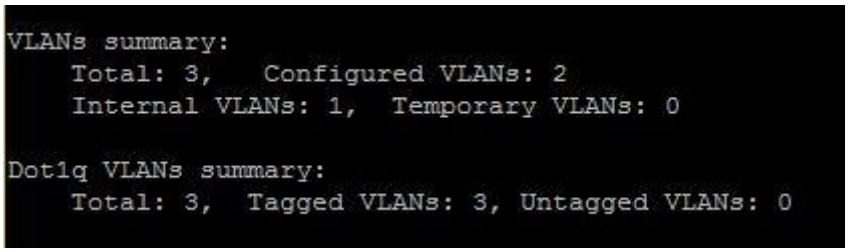
Step 2: Create a logical layer 3 VLAN interface:
set interfaces vlan unit <unit> family inet address <ip address/mask>

Step 3: Link the layer 2 VLAN to the layer 3 VLAN interface:
set vlans <vlan-name> l3-interface vlan.<unit mentioned above>

The result in project after implementing vlans can be displayed using command: *show vlans.*

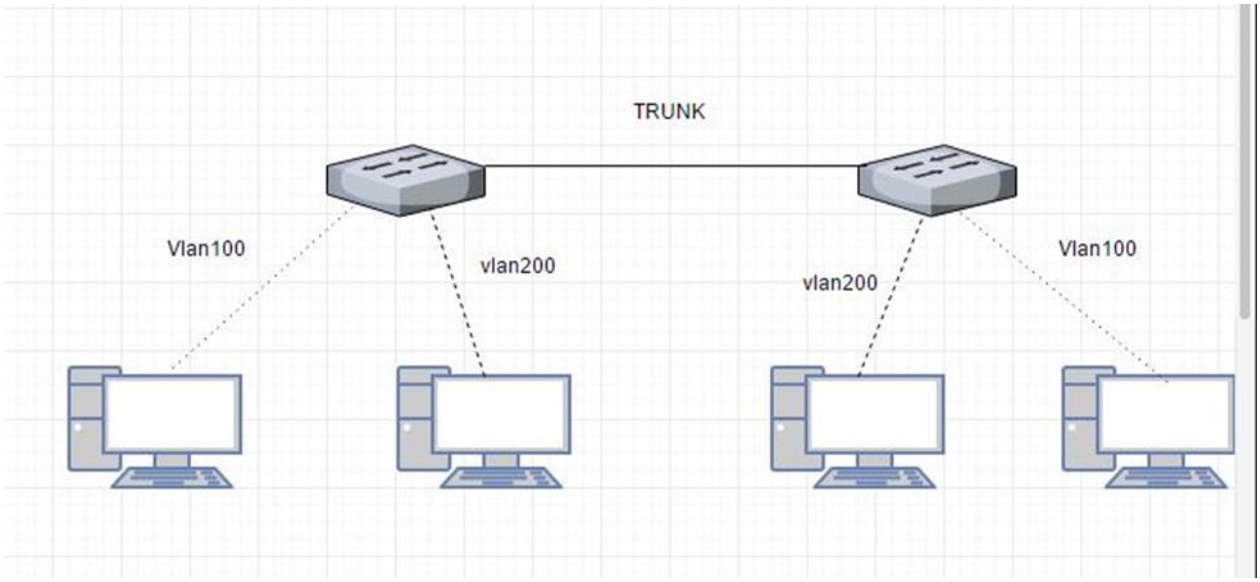


Other command that can be used to display configured vlans is: *show vlans summary.*
In the display, untagged and tagged vlans are shown.



2.2 VLAN Trunking

Trunk mode interfaces are used to connect switches to one trunk interface. Traffic sent between switches can then consist of packets from multiple VLANs, with those packets multiplexed so they can be sent over the same physical connection.



The trunk interface is a switched interface, it must have a corresponding interface on a second switch.

QUICK CONFIGURATION:
set interfaces <interface> unit <unit number> family ethernet-switching port-mode access vlan members <vlan-name>

Only VLANs named in members <vlan-name> have access over the Trunk.

2.3 VLAN L3-Interface

In order to configure the switch to perform L3 switching it is necessary to assign VLAN interface to VLAN using command:
Set vlans <vlan name> l3-interface vlan.<vlan number>

This is how it looks when the VLAN is implemented with L3 interface.

```
lab@srxC-1# run show vlans brief
                                                        Ports
Name            Tag       Primary Address           Active/Total
default         1
vlan100         100       192.168.100.1/24            1/1
vlan200         200       192.168.200.1/24            1/1
```

2.4 Virtual Routers

In Junos Software, a virtual router is a routing instance type. It is a collection of routing tables, interfaces and routing option settings. Routing instance virtual router can act like a normal router, with policies and routing options. It makes it possible to isolate traffic without using multiple routing devices to segment the network.
To establish a virtual router, it is necessary to follow a few steps.

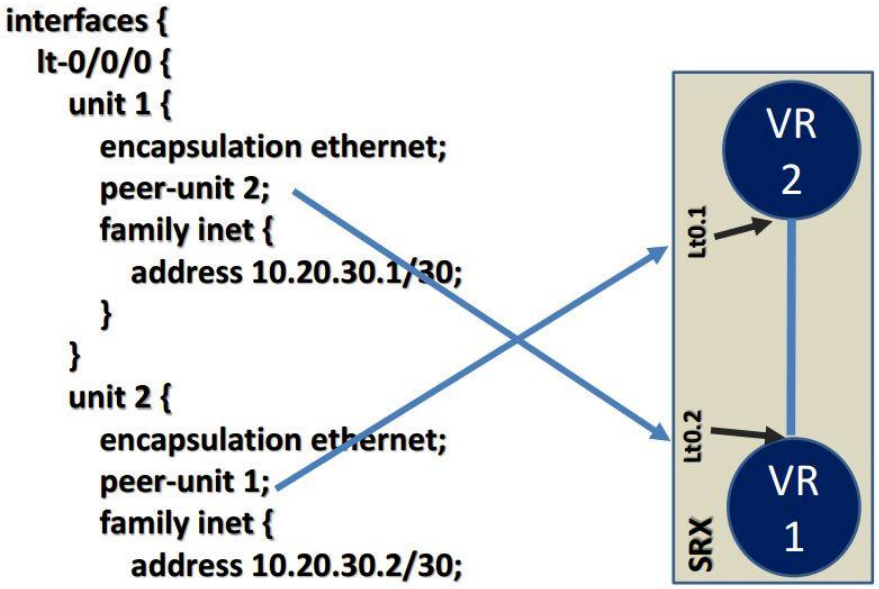
- Create a virtual router
- Assign an interface to a virtual router

- Assign an interface to a zone

It is possible to assign other routing option to virtual router.
To share routes in more than one routing instance it is optional to select physical or logical connection. Physical connection is a normal interface (for example ge-0/0/0 or so-0/0/0) it can be established using cables or VMNets in VMware.

2.5 Logical Tunnels

Logical tunnels (lt0 interfaces) can be used only on SRX Juniper devices.
To connect two routing instances with a logical connection logical tunnel interface should be configured for each instance. Then, it is mandatory to configure a peer relationship between the logical tunnel interfaces, thus creating a point-to-point connection. To create a point-to-point connection logical tunnel must configured using the lt-fpc/pic/port format.
Each logical tunnel interface should be configured with a proper encapsulation type.
It is important to configure only one peer unit for each logical interface. (Unit 0 cannot peer with both unit 1 and unit 2.)



2.6 Ethernet OAM

Ethernet Operations, Administration, and Maintenance
Ethernet OAM is a set of tools that network manager uses to know the way how Ethernet links are working. Ethernet OAM should:

- Rely only on the media access control (MAC) address or virtual local area network (VLAN) identifier for troubleshooting
- Work independently of the actual Ethernet transport and function over physical Ethernet ports, or a virtual service such as pseudo wire, and so on.
- Isolate faults over a flat (or single operator) network architecture or a nested or hierarchical (or multi-provider) networks.

2.7 Troubleshooting & Monitoring

2.7.1 Troubleshooting

Juniper SRX devices provides a set of commands that can be used for troubleshooting. Mostly it gives an opportunity to view log files, environment of router and alarms.
Few troubleshooting commands:

- show chassis environment

Command above allows to display temperatures, fans and power supply on SRX device.

```
lab@srxC-1# run show chassis environment
Class Item                           Status     Measurement
Temp  Routing Engine                 OK         44 degrees C / 111 degrees F
      Routing Engine CPU             OK         42 degrees C / 107 degrees F
Fans  SRX240 PowerSupply fan 1       OK         Spinning at normal speed
      SRX240 PowerSupply fan 2       OK         Spinning at normal speed
      SRX240 CPU fan 1               OK         Spinning at normal speed
      SRX240 CPU fan 2               OK         Spinning at normal speed
      SRX240 IO  fan 1               OK         Spinning at normal speed
      SRX240 IO  fan 2               OK         Spinning at normal speed
Power Power Supply 0                 OK
```

- show chassis hardware

Using command above causes display hardware information like serial numbers, part numbers and version.

```
lab@srxC-1# run show chassis hardware
Hardware inventory:
Item             Version  Part number  Serial number     Description
Chassis                                AH4113AK0093      SRX240H-POE
Routing Engine   REV 57   750-021794   ACKD8069          RE-SRX240H-POE
FPC 0                                                   FPC
  PIC 0                                                 16x GE Base PIC
Power Supply 0
```

- show chassis alarms

This command is used for displaying information about alarms in real time.

```
lab@srxC-1# run show chassis alarms
No alarms currently active
```

Troubleshooting commands above were used on Juniper SRXC-1 in school's lab.

2.7.2 Monitoring

Besides troubleshooting Juniper gives command to monitor interfaces, traffic and routes on device.
Most common command for monitoring routes in router is: *show route.*
Where next-hops, routes, preferences and protocols.

```
r1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.100.0/24   *[Direct/0] 2d 23:02:37
                    > via vlan.100
192.168.100.1/32   *[Local/0] 2d 23:02:39
                      Local via vlan.100
224.0.0.5/32       *[OSPF/10] 2d 22:36:54, metric 1
                      MultiRecv

r2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.200.0/24   *[Direct/0] 2d 23:02:37
                    > via vlan.200
192.168.200.1/32   *[Local/0] 2d 23:02:39
                      Local via vlan.200
224.0.0.5/32       *[OSPF/10] 2d 22:36:54, metric 1
                      MultiRecv
```

show interfaces terse displays admin and link state of interfaces as well as protocol its using and addresses.

```
lab@srxC-1# run show interfaces terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    up
ge-0/0/0.0              up    up   inet     10.210.14.143/26
gr-0/0/0                up    up
ip-0/0/0                up    up
lsq-0/0/0               up    up
lt-0/0/0                up    up
mt-0/0/0                up    up
sp-0/0/0                up    up
sp-0/0/0.0              up    up   inet
                                   inet6
sp-0/0/0.16383          up    up   inet     10.0.0.1              --> 10.0.0.16
                                            10.0.0.6              --> 0/0
                                            128.0.0.1             --> 128.0.1.16
                                            128.0.0.6             --> 0/0
ge-0/0/1                up    up
ge-0/0/2                up    up
ge-0/0/3                up    up
```

In Juniper Devices, it is possible to monitor traffic inside the router, using *monitor traffic.*
It can be used instead of using Wireshark.

```
lab@srxC-1# run monitor traffic
verbose output suppressed, use <detail> or <extensive> for full protocol decode
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.
Address resolution timeout is 4s.
Listening on ge-0/0/0, capture size 96 bytes

10:56:12.643796  In STP 802.1w, Rapid STP, Flags [Learn, Forward], bridge-id 8000.3c:61:04:66:94:01.820d, length 43
 10:56:14.554787  In STP 802.1w, Rapid STP, Flags [Learn, Forward], bridge-id 8000.3c:61:04:66:94:01.820d, length 43
 Reverse lookup for 10.210.14.189 failed (check DNS reachability).
Other reverse lookup failures will not be reported.
Use <no-resolve> to avoid reverse lookups on IP addresses.
```

2.8 Sources

VLAN Presentation – EAL

http://searchnetworking.techtarget.com/definition/virtual-LAN

https://kb.juniper.net/InfoCenter/index?page=content&id=KB11000

https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/l3-interface-bridging.html

https://kb.juniper.net/InfoCenter/index?page=content&id=KB21260

https://www.juniper.net/documentation/en_US/junos12.3/topics/concept/layer-2-802-1ag-ethernet-oam-overview-mx-solutions.html

3 Intermediate Routing

(Full configuration files are on GitHub: https://github.com/miskor/Project_network/tree/master/APPENDIX/Stage-2%20Intermediate%20Routing)

3.1 IPv6

IPv6 is the most recent version of the IP protocol an important change from IPv4 is that IPv6 uses 128-bits for address space instead of the 32-bit address space of IPv4. This is important since the IPv4 address pool is all but exhausted. IPv4 provides about 4.29 billion addresses which means that far from every person on earth can have a device with an IPv4 address.

3.1.1 IPv6 headers

The IPv6 header is simpler than that of IPv4 which allows for faster processing. The IPv6 header is fixed in size at 40 bytes (the IPv4 header has a maximum size of 60 bytes) and includes options as extension headers that are only there when actually used. This makes the design easier to extend for future additions.

1 byte	**1 byte**	**1 byte**	**1 byte**
Version	Traffic Class	Flow Label	
Payload length		Next Header	Hop Limit
Source Address			
Destination Address			

Illustration 1: The IPv6 fixed header format.

Referring to Illustration 1 above this is a description of the header fields:

- **Version:** Protocol version number = 6.
- **Traffic Class:** The 6 most-significant bits are used to classify packets. The remaining 2 bits are used in order to signal impending congestion.
- **Flow Label:** Used to label sequences of packages that are to be treated the same way by allowing for more efficient processing by routers.
- **Payload Length:** Length of the payload limited to 64K y the size of this field. IPv6 can use an extension called Jumbo frames to send larger payloads. The length counts everything after the fixed header, included the extension headers.
- **Next Header:**
    - If the next header is UDP or TCP, this field will contain the same protocol numbers as in IPv4, for example, protocol number 6 for TCP or 17 for UDP. The protocol numbers are available through IANA at https://www.iana.org/assignments/protocol-numbers/
    - If extension headers are used, this is the type of the next extension header.
- **Hop Limit:** Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
- **Source Address:** 128-bit address of the source of the packet.
- **Destination Address:** 128-bit address of the intended recipient of the packet, this might be the ultimate recipient if a Routing header is present.

Below is an illustration of the extension headers and payloads organisation, coming after the fixed IPv6 header.

Fixed header	Extension header 1	Extension header ...	Extension header *N*	Payload

Illustration 2: IPv6 fixed, and extension -header followed by the payload.

3.1.2 IPv6 addressing

IPv6 addresses are either unicast, anycast, or multicast addresses

- Unicast addresses identify a single network interface and the IP packages will be sent to that interface.
- Anycast addresses are assigned to a group of interfaces. Anycast addresses has the same format as unicast addresses and differ only by the fact that they are present on more than one interface in the network. An anycast packet is delivered to one member, typically the nearest host according to the routing protocols definition.
- Multicast addresses are assigned to a group of interfaces, each member processes a message send to a multicast address.

An IPv6 address has 128 bits and address is divided into eight 16-bit hexadecimal blocks

separated by colons. For example:

2001:0DB8:0000:0000:00C0:FEFE:BADA:5500

There are some rules for shortening addresses like this:

- Leading zeroes can be skipped.

2001:DB8:0:0:C0:FEFE:BADA:5500

- A double colon can replace a string of consecutive zeroes in more than one 16-bit address block, but a double colon can only be used once in the address
    - 2001.DB8::C0:FEFE:BADA:5500

Because of the transition from IPv4 to IPv6, still in progress, a special syntax has been introduced where the least 32 bits of an address can be written in the familiar dot notation of IPv4:

::ffff:c000:0280 and ::ffff:192.0.2.128 is the same, but the second form will be more recognisable if you have ever worked with IPv4.

3.1.2.1 Special addresses

- ::/127 – Is for point to point connections like the IPv4 /30 prefix. In IPv6 there is no need for the broadcast and network address.
- ::/128 – The unspecified address that corresponds to the IPv4 0.0.0..0/32 address. This address must never be assigned to an interface, but is for instance used to get software to listen for incoming connections on all interfaces
- ::/0 - The default route address that corresponds to the IPv4 0.0.0.0/0 in IPv4.
- ::1/128 – The loopback address.
- fe80::/10 – This is a link-local address and compares to the auto-configuration type address of IPv4. The last 64 bits are usually chosen as the interface hardware address in modified EUI-64 format, basically adding ff:ee in the middle of the 48-bit MAC address. Addresses in the link-local prefix are only valid and unique on a single link.
- fc00::/7 - Unique local addresses (ULAs), these addresses are comparable to IPv4 private addresses (10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16). Like their IPv4 counterparts they are routable only within a set of cooperating sites. There are provisions in this range for doing addresses that will most likely not clash when merging more networks.
- ::ffff:0:0/96 - This prefix is designated as an *IPv4-mapped IPv6 address*.
- 64:ff9b::/96 - Addresses with this prefix are used for automatic IPv4/IPv6 translation.
- 2002::/16 - This prefix is used for 6to4 addressing.
- 2001::/32 — Used for Teredo tunnelling, a transition technology that gives full IPv6 connectivity for IPv6-capable hosts that are on the IPv4 Internet, it functions from behind NAT devices.
- 2001:db8::/32 — This prefix is used in documentation.

3.1.3 Juniper configuration.

This is a basic configuration setting IPv6 addresses on a logical tunnel.

```
interfaces {
    lt-0/0/0 {
        #vSRX-1
        unit 11 {
            encapsulation ethernet;
            peer-unit 21;
```

The inet6 family tells the SRX to use the inet6 routing table.

```
            family inet6 {
                address fdaa:dead:beef:1::1/127;
            }
        }
        #vSRX-2
```

```
        unit 21 {
            encapsulation ethernet;
            peer-unit 11;
            family inet6 {
                address fdaa:dead:beef:1::1/127;
            }
        }
    }
```

3.2 Sources:

Wikipedia - Open Shortest Path First - https://en.wikipedia.org/wiki/Open_Shortest_Path_First

The Internet Engineering Task Force – RFC2460 - https://tools.ietf.org/html/rfc2460

IANA – Protocol numbers - https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

EAL - IPv6 Essentials Addressing

3.3 OSPF

OSPF is short for Open Shortest Path First and is the name of a routing protocol. OSPF is an interior gateway protocol which means that it is used to exchange routing information within an autonomous system. OSPF Version 2 was defined in RFC2328 in 1998 and Version 3 in RFC5340 in 2008. Version 3 is an update to support IPv6.
OSPF forms IP datagrams directly and packages them using protocol number 89 and implements its own transport layer error detection and correction functions. OSPF uses multicast addressing for distributing routing information within a broadcast domain.
Routers running OSPF communicate with neighbouring routers on connected interfaces to establish the state of connections:

- **Down:** Initial state of the connection indicating that no resent communication has been received.
- **Init:** A HELLO packet has been received from a neighbour but the routers have not established two-way communication.
- **Exchange:** The router is sending its link state database to the neighbour in database description packets. Each packet has a sequence number that is explicitly acknowledged.
- **Loading:** The router requests the most recent link-state advertisements from its neighbour discovered in the Exchange state.
- **Full:** The end state when all adjacent routers have reached the Full state and the link state database of o the neighbours are fully synchronized.

3.3.1 OSPF areas

Areas in OSPF are used to administratively group networks and host in an AS together, areas are identified by 32-bit numbers. The topology of an area is unknown outside that area

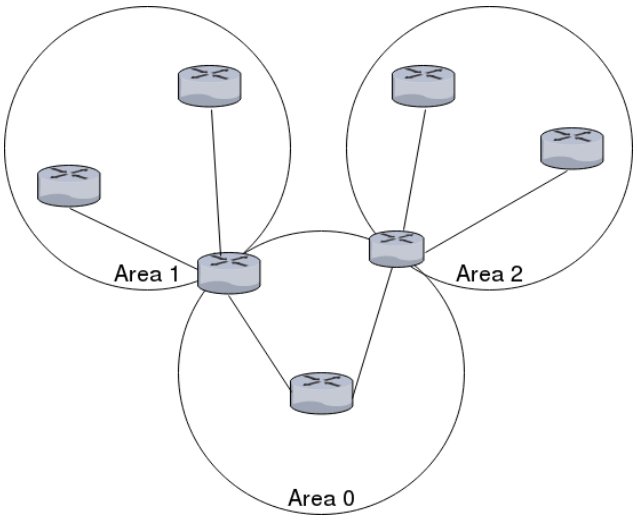


Illustration 1: OSPF areas

An example of a network split into areas are shown in Illustration 1. The routers fully inside the areas (circles) are called internal routers, these are all connected to devices inside the same area. The routers on the borders between to areas are called area border routers or ABRs. Area 0 has a special role as the backbone area that distributes routes between areas. All ABRs are connected to the backbone, and the backbone area most be contiguous, if not physically, by using virtual links. The backbone has no ABRs and the routers in area 1 must go through area 0 to talk to routers in area 2. It is the backbones job to redistribute routing information between the other areas.

3.3.2 Designated Router

To not load down the network with routing traffic in large networks, OSPF uses designated routers. Routers in the same network sends their link state information to the designated router. The designated router sends the link advertisements on behalf of the network and participates in synchronising the link state database by establishing adjacencies. The designated router is found through election.

- The router priorities are evaluated and the router with the highest priority is selected as the designated router.
- If there is more than one router with the same priority, the one with the highest router identifier is chosen.
- If the no router ids are configured the election will go by the IP address of the first interface that comes online. This is usually the loopback interface.
- If nothing of the above, the first hardware interface with an IP address will be used for the election.

By default, routing devices has a priority of 128. The priorities work like this:

- 0: the router will not be considered in the election.
- 1: the router has the least chance of being elected.

3.3.3 Dijkstra's algorithm

OSPF (and IS-IS described later) uses Dijkstra’s algorithm to calculate the shortest path between to nodes. The algorithm works as follows:

1. Set the first node as the current node. Assign a distance to each node, the value is zero for the current node and infinity for all others. Create a set of all unvisited nodes.
2. Calculate the distance to all neighbours of the current node and compare to this distance to the current value. Assign the new distance if it is smaller.

3. When distances to all neighbours have been calculated, mark the current node as visited and remove it from the set of unvisited nodes.
4. When the destination node is marked as visited, if looking for a route between specific nodes, the path has been found. When doing a complete traversal of the graph, there is no nodes in the unvisited set that has a lower distance than infinity, there is no connection between the initial node, and the unvisited nodes and the algorithm has finished.
5. If none of the above is the case, select the node from the unvisited set that has the lowest distance value as the current node, and repeat from step 2.

As suggested in point 4 above this algorithm can gradually move its way through the network calculating shortest paths.

3.3.4 Quick configuration example

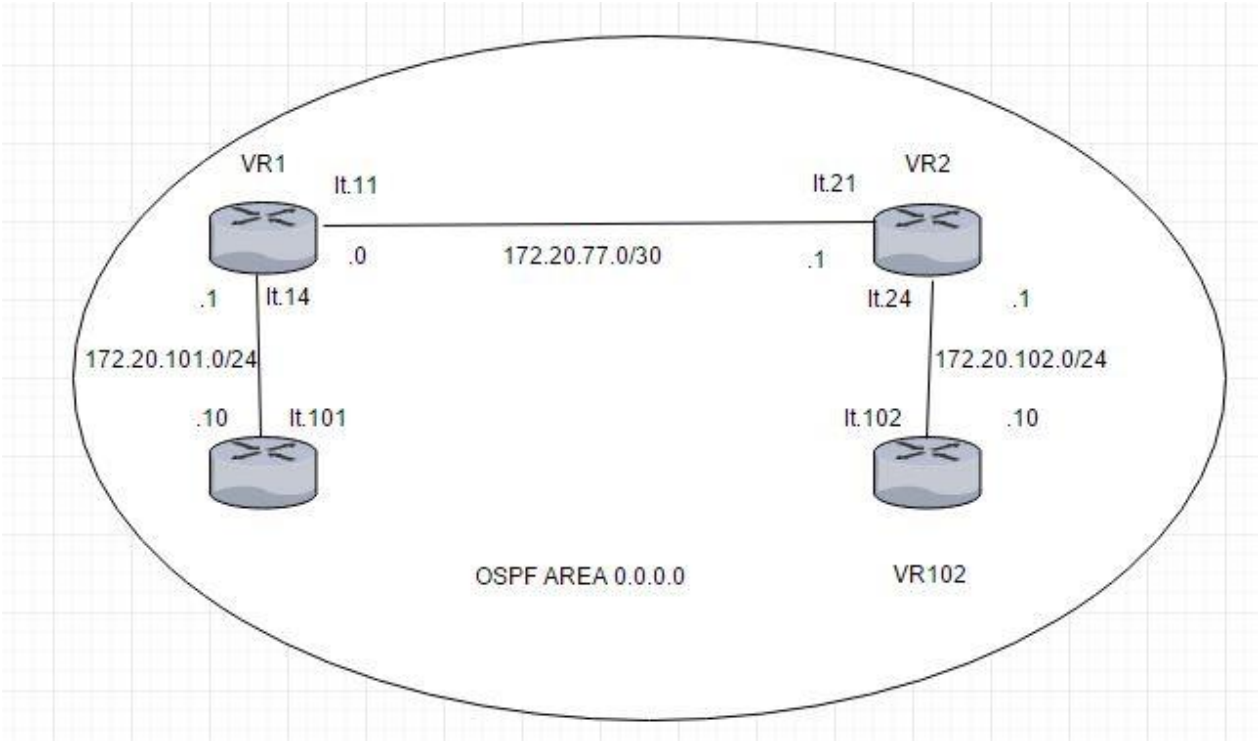


Illustration 2: Network diagram of the OSPF example configuration.

The above diagram illustrates an example configuration of virtual router to use OSPF. Connection between the virtual routers are connected using logical tunnels.

This is the routing instance designated “VR1” in the illustration above:

```
routing-instances {
    gangstin {
```

The logical tunnel interfaces and loopback interface are set up according to the illustration above.

```
        instance-type virtual-router;
        interface lt-0/0/0.11;
```

```
        interface lt-0/0/0.14;
        interface lo0.1;
```

Next comes the actual OSPF configuration. This configuration includes all interfaces in area 0, the backbone area.

```
        protocols {
            ospf {
                area 0.0.0.0 {
                    interface lt-0/0/0.11;
                    interface lt-0/0/0.14;
                    interface lo0.1;
                }
            }
        }
    }
}
```

This configuration is mirrored on each virtual router, except of course the interfaces change according to the diagram above.

3.3.5 Sources:

Wikipedia - Open Shortest Path First - https://en.wikipedia.org/wiki/Open_Shortest_Path_First
Wikipedia - Dijkstra's algorithm - https://en.wikipedia.org/wiki/Dijkstra's_algorithm
Juniper website - Understanding OSPF Areas - https://www.juniper.net/documentation/en_US/junos/topics/concept/ospf-routing-understanding-ospf-areas-overview.html
EAL - OSPF/Open Shortest Path First presentation

3.4 IS-IS

IS-IS is an interior gateway protocol designed for routing traffic in an administrative domain it is published as ISO/IEC 10589:2002. Each router floods link state information through the network and independently builds its own database of the networks topology by collecting this information. IS-IS has similarities with OSPF but is a layer 2 protocol and does not use the IP protocol to communicat<e routing information. As with OSPF, IS-IS uses Dijkstra's algorithm to compute the best path to the network. This algorithm is further described in the OSPF section.

Data Link Header	IS-IS Header	IS-IS Data	
Data Link Header	IP Header	OSPF Header	OSPF Data

Illustration 1: IS-IS (top) and OSPF (bottom) encapsulation.

The difference between the OSPF Layer 3 encapsulation and that of the Layer 2 IS-IS is shown above. IS-IS is easy to extend by using TLV (Type-Length-Value) field. For instance, IS-IS did not originally support IPv4 and later IPv6 but they were added using TLVs.

1 Byte	1 Byte	*"Length"* bytes

Type	Length	Value

Illustration 2: Format of the TLVs used by IS-IS

The format of a TLV is shown above. Each TLV contain information concerning routing of a certain type as by the first byte. The length of this information is variable and specified in the second byte. The actual information is type dependant but spans the number of bytes given in the previous field.

3.4.1 Terminology and OSPF differences

An IS-IS network is a single autonomous system (AS) or routing domain consisting of:

- End systems are network entities that send and receive packets.
- Intermediate systems send and receive packets and relay packets, also known as a router.
- ISO packages are called network PDUs (See Protocol Data Unit below).
- A single AS can be divided into smaller groups called areas.
- Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas.
    - Level 1 Intermediate System: Route within the same area or towards Level 2 systems
    - Level 2 Intermediate System: Route between areas and towards other ASs.
- There is no Area 0 like OSPF.
- IS-IS is neutral with regards to the type of network addresses it can route.
- OSPF areas are tied to interfaces making it possible for the Area Border Routers (ABRs) to be in more than one area at once. An IS-IS router is only in one area and the border is between the routers.

3.4.2 Protocol Data Units

IS-IS exchanges information in Protocol Data Units (PDUs) here are the different types:

- IS-IS hello (IIH) PDUs
    - Broadcast to discover identity of neighbouring IS-IS routers.
    - Determine whether neighbours are Level 1 or Level 2 intermediate systems.
- Link-state PDUs (LSPs)
    - Describes the state of adjacencies in neighbouring IS-IS systems.
    - Flooded periodically throughout an area to keep information up to date between IS-IS systems.
- Sequence Number Packets (SNP)
    - Complete sequence number PDUs (CSNPs)
    - Partial sequence number PDUs (PSNPs)

3.4.3 PDU Format

1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte

Protocol Identifier	Header Length	Version	ID Length	PDU type	Version	Reserved	Maximum Area Address
PDU Length		Remaining lifetime		LSP ID			
LSP ID cont.				Sequence number			
Checksum		P, ATT, and IS Type bits		TLVs (variable length)			

Illustration 3: The Protocol Data Unit.

Field details

- Protocol Identifier
  - Always 0x83
- Version
  - Always 1.
- ID Length
  - 0 means 6.
- PDU Type
  - 15: LAN Level-1 Hello
  - 16: LAN Level-2 Hello
  - 17: Point-to-point Hello
  - 18: Level-1 LSP
  - 20: Level-2 LSP
  - 24: Level-1 Complete SNP
  - 25: Level-2 Complete SNP
  - 26: Level-1 Partial SNP
  - 27: Level-2 Partial SNP
- Version
  - Always 1
- Maximum Area Addresses
  - 0: indicates the IS only supports three area addresses (by default).
  - Others: up to 254 indicates the number of areas allowed.
- PDU Length, Remaining lifetime, LSP ID, Sequence number, and Checksum are PDU specific.
- P, ATT, and IS type bits.
  - ATT bit is set if IS is connected to another area
  - OL bit is set is the link-state database is overloaded
  - IS Type bits determine a L1 or L2 router

        - Level 1 router: 1
        - Level 2 router: 3
- TLVs
    - Level 1 PDU: 1, 2, 10, 22, 128, 129, 132, 134, 135, 137, 222, 229, 232, 235, 236
    - Level 2 PDU: 1, 2, 10, 22, 128, 129, 130, 132, 134, 135, 137, 222, 229, 232, 235, 236

3.4.4 PDU TLV types

- 1: Area Address
- 2: IS reachability
- 10: Authentication
- 22: Extended IS reachability
- 128: IP internal reachability
- 129: Protocols supported
- 130: IP external reachability
- 132: IP interface address
- 134: TE IP router ID
- 135: Extended IP reachability
- 137: Dynamic host name resolution
- Multiple topologies (routing instances) supported
    - TLVs 222, 229, and 235
- 232 and 236: IPv6 is support

3.4.5 IS-IS configuration on the Juniper SRX

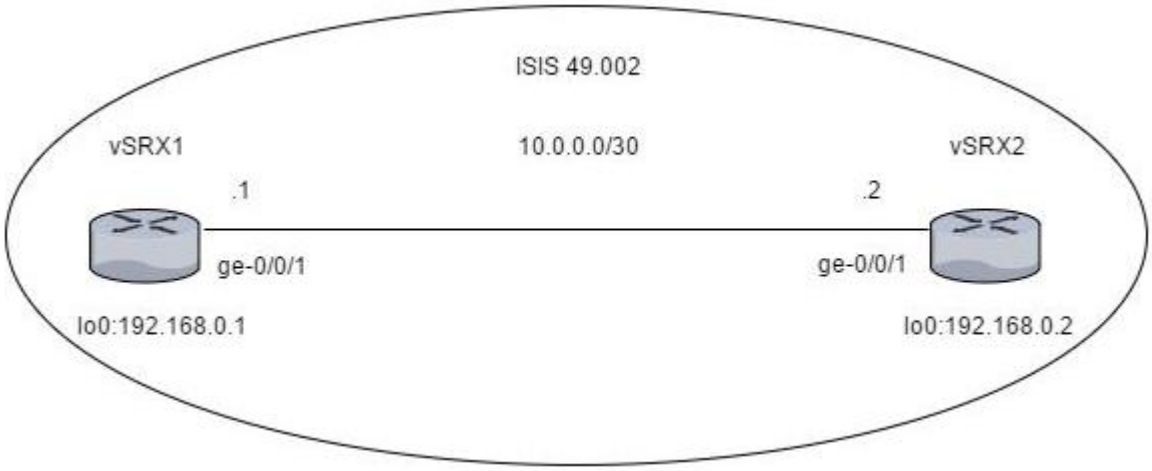


Illustration 4: The example IS-IS network on the Juniper SRXs.

The IS-IS configuration needs an ISO network address and it is configured on the lo0 interface:

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32;
            }
            family iso {
                address 49.0002.0192.0168.0001.00;
            }
        }
    }
}
```

49	00002	0192	0168	0001	00
AFI (1 byte)					
Area ID (variable 1-13 bytes)		System ID (6 bytes)			NSEL (1 byte)

- Area ID:
  - Variable part of the NET address.
    - The first byte is the AFI (Address Family Identifier), 49 means private addressing.
    - The following bytes of the AREA ID can be arbitrary filled to represent the Area number and is useful for Level 1 adjacency.
- System ID:
  - This is a unique id for the router like the OSPF router-id. In this case it is derived from the loopback IP address.

- NSEL:
  - The NSEL is like the Protocol field of the IP header. It must be set to 0x00 for adjacencies to come up.

Interfaces that are to handle IS-IS PDUs needs to have the “family iso” statement added.

```
interfaces {
    ge-0/0/1 {
        unit 0 {
            description to-vSRX2;
            family inet {
                address 10.0.0.1/30;
            }
            family iso;
        }
    }
}
```

The interfaces must be added in the “protocols” hierarchy under “isis” and any optional IS-IS parameters configured.

```
protocols {
    isis {
        interface ge-0/0/1.0;
        interface lo0.0;
    }
}
```

In the security hierarchy enable packet-based processing for the “mpls” and “iso” family.

```
security {
    forwarding-options {
        family {
            mpls {
                mode packet-based;
            }
            iso {
                mode packet-based;
            }
        }
    }
}
```

```
root> ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: icmp_seq=0 ttl=64 time=0.024 ms
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.070 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.477 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.339 ms
^C
--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.024/0.227/0.477/0.188 ms
```

Illustration 5: Pinigng vSRX1 to test the connection.

As shown in the network diagram above two machines are connected using the above configuration, with individual addresses.

```
root> show route

inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/30        *[Direct/0] 00:05:07
                    > via ge-0/0/1.0
10.0.0.1/32        *[Local/0] 00:05:11
                      Local via ge-0/0/1.0
192.168.0.1/32     *[Direct/0] 00:05:30
                    > via lo0.0
192.168.0.2/32     *[IS-IS/15] 00:03:02, metric 10
                    > to 10.0.0.2 via ge-0/0/1.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

49.0002.0192.0168.0001/72
                   *[Direct/0] 00:05:30
                    > via lo0.0
```

Illustration 6: Showing that IS-IS routes are in the routing table.

The ping works and the IS-IS route is in the routing table as shown above and below.

```
root# run show isis database
IS-IS level 1 link-state database:
LSP ID                 Sequence Checksum Lifetime Attributes
.00-00                      0x3   0x37ed     1063 L1 L2
0192.0168.0002.00-00        0x4   0x796c     1086 L1 L2
0192.0168.0002.02-00        0x1   0x3c7c     1061 L1 L2
0192.0168.1002.00-00        0x3   0xc32a     1084 L1 L2
0192.0168.1002.02-00        0x1   0xd4b1     1084 L1 L2
  5 LSPs

IS-IS level 2 link-state database:
LSP ID                 Sequence Checksum Lifetime Attributes
.00-00                      0x4   0xfd94     1063 L1 L2
0192.0168.0002.00-00        0x5   0x6fd5     1086 L1 L2
0192.0168.0002.02-00        0x1   0x3c7c     1061 L1 L2
0192.0168.1002.00-00        0x5   0x3177     1095 L1 L2
0192.0168.1002.02-00        0x1   0xd4b1     1084 L1 L2
  5 LSPs
```

Illustration 7: The IS-IS link state database content

The isis database has the correct entries as well.

3.4.6 Sources:

EAL – IS-IS Intermediate System to Intermediate System.

3.5 Route Re-Distribution (OSPF/IS-IS)

This chapter is about re-distributing OSPF to IS-IS and visa verse. Since each routing protocol has its own chapter this is mainly summarising the configuration on the Juniper SRX series router.

3.5.1 Configuring the Juniper SRX

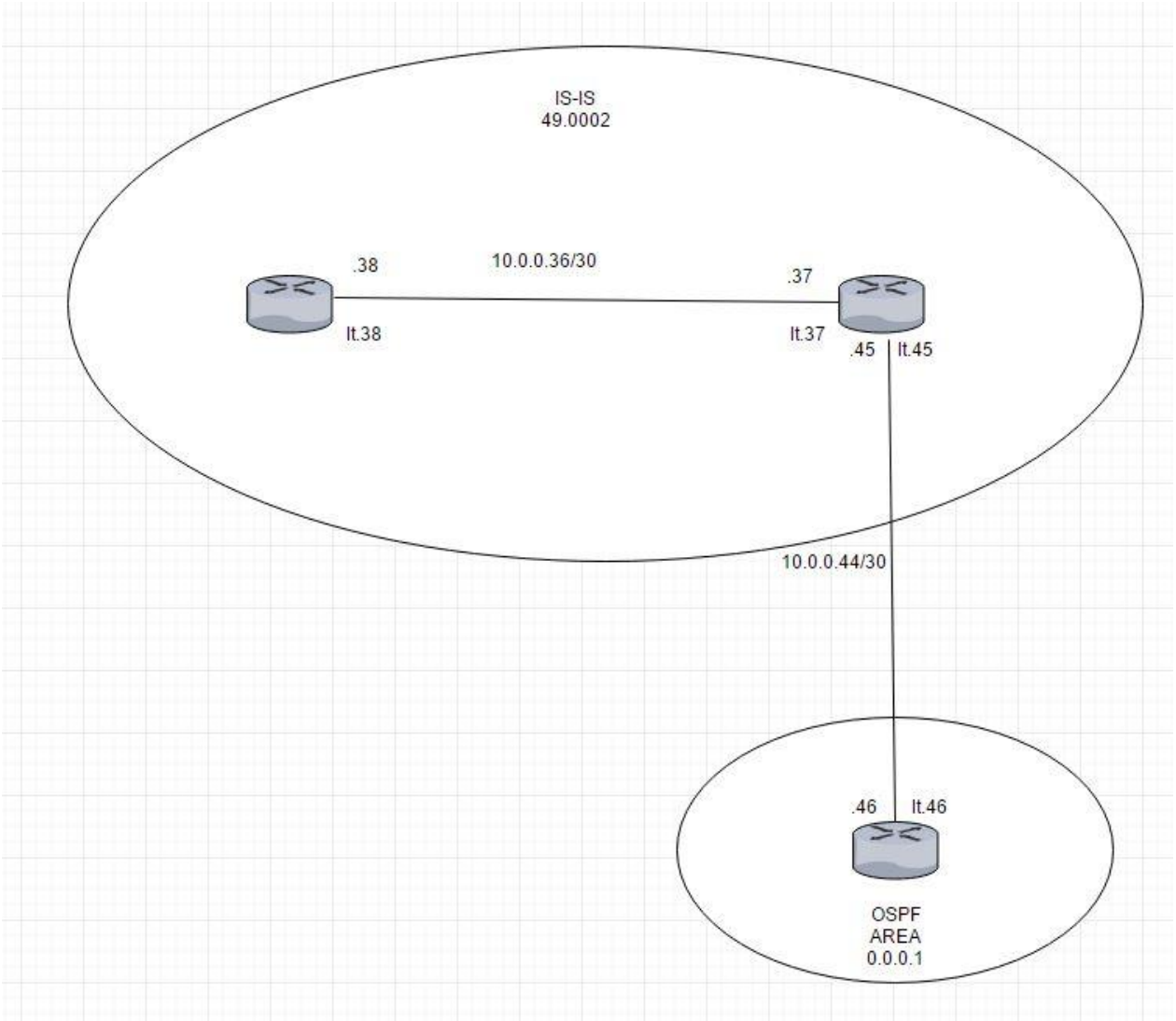


Illustration 1: The network diagram of the network where OSPF - IS-IS redistrubution is configured

(The full configuration is available in the appendix)
The OSPF router has a standard OSPF configuration but the loopback interface has multiple addresses used to simulate routing destinations.

```
Interfaces {
```

The ge-0/0/9 interface is connected to the router doing redistribution.

```
    ge-0/0/9 {
       unit 0 {
            family inet {
                address 10.0.0.46/30;
            }
        }
    }
```

The lo0 interface is configured to have several addresses to simulate route destinations.

```
    lo0 {
        unit 0 {
            family inet {
                address 192.168.1.1/32;
                address 192.168.2.1/32;
                address 192.168.3.1/32;
                address 192.168.0.1/32;
            }
        }
    }
}
```

Packages to the simulated destination are discarded.

```
routing-options {
    static {
        route 192.168.0.0/24 discard;
        route 192.168.1.0/24 discard;
        route 192.168.2.0/24 discard;
        route 192.168.3.0/24 discard;
    }
    autonomous-system 22;
}
```

The loopback interface and the interface connected to the IS-IS router is added to OSPF area 1 and the static routes are exported into OSPF using the “ospf” policy.

```
protocols {
    ospf {
        export ospf;
        area 0.0.0.1 {
            interface ge-0/0/9.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}


policy-options {
    policy-statement ospf {
        term 1 {
            from protocol static;
            then accept;
        }
    }
}
security {
    forwarding-options {
        family {
            mpls {
```

```
                mode packet-based;
            }
        }
    }
}
```

The router that does redistribution has both OSPF and IS-IS configured on it.

```
interfaces {
```

The first interface connected to the router running OSPF configured above.

```
    ge-0/0/5 {
        unit 0 {
            family inet {
                address 10.0.0.45/30;
            }
        }
    }
```

These interfaces connect the router doing OSPF/IS-IS redistribution (ge-0/0/6) to the routing instance that does IS-IS shown at the far-left side of the network diagram. Notice that “family iso” is included as described in the IS-IS chapter.

```
    ge-0/0/6 {
        unit 0 {
            family inet {
                address 10.0.0.38/30;
            }
            family iso;
        }
    }
    ge-0/0/7 {
        unit 0 {
            family inet {
                address 10.0.0.37/30;
            }
            family iso;
        }
    }
```

Set the routers ISO addresses on the loopback interface. The router is in a private area (49) with an ID of 2. The redistributing router has a system id of 0172.0016.0907 and the pure IS-IS router has an ID of 0172.0016.0305.

```
    lo0 {
        unit 0 {
            family inet {
                address 176.16.1.2/32;
            }
            family iso {
                address 49.0002.0172.0016.0907.00;
            }
        }
        unit 1 {
            family inet {
```

```
                address 172.16.3.5/32;
            }
            family iso {
                address 49.0002.0172.0016.0305.00;
            }
        }
    }
}
```

All routers are in AS17.

```
routing-options {
    autonomous-system 17;
}
```

Allow export of IS-IS into OSPF and OSPF into the IS-IS and allow the IS-IS routers to talk to each other.

```
protocols {
    isis {
        export [ ospf-isis send-direct-to-isis-neighbors ];
        interface ge-0/0/7.0;
        interface lo0.0;
    }
    ospf {
        export send-direct-to-ospf-neighbors;
        area 0.0.0.1 {
            interface ge-0/0/5.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}
policy-options {
```

Allow traffic from the dummy routing destinations set up in the OSPF router.

```
    policy-statement ospf-isis {
        term 1 {
            from {
                protocol ospf;
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
```

Allow traffic between IS-IS and OSPF and vice versa.

```
    policy-statement send-direct-to-isis-neighbors {
        from {
            protocol direct;
            route-filter 10.0.0.44/30 exact;
        }
```

```
        then accept;
    }
    policy-statement send-direct-to-ospf-neighbors {
        from {
            protocol direct;
            route-filter 10.0.0.36/30 exact;
        }
        then accept;
    }
}
```

This is the routing instance that does only IS-IS.

```
routing-instances {
    buddy {
        instance-type virtual-router;
        interface ge-0/0/6.0;
        interface lo0.1;
        protocols {
            isis {
                interface ge-0/0/6.0;
                interface lo0.1;
            }
        }
    }
}
```

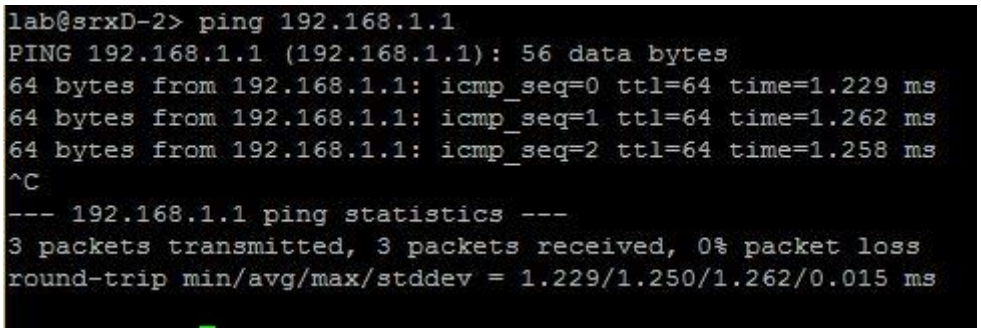


Illustration 2: Pinging the router running OSPF and getting an answer back.

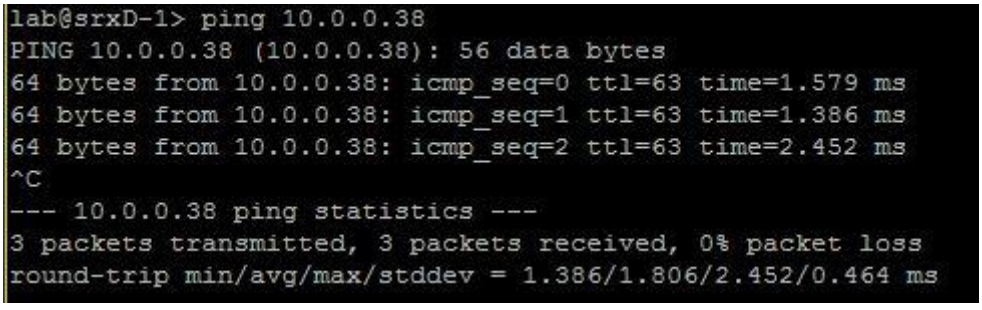


Illustration 3: Pinging the virtual router buddy and getting an answer back.

The OSPF routing table contains routes to all the dummy destinations.

```
lab@srxD-2> show route protocol ospf

inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.0/24     *[OSPF/150] 02:21:39, metric 0, tag 0
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.0.1/32     *[OSPF/10] 02:21:39, metric 1
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.1.0/24     *[OSPF/150] 02:21:39, metric 0, tag 0
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.1.1/32     *[OSPF/10] 02:21:39, metric 1
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.2.0/24     *[OSPF/150] 02:21:39, metric 0, tag 0
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.2.1/32     *[OSPF/10] 02:21:39, metric 1
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.3.0/24     *[OSPF/150] 02:21:39, metric 0, tag 0
                    > to 10.0.0.46 via ge-0/0/5.0
192.168.3.1/32     *[OSPF/10] 02:21:39, metric 1
                    > to 10.0.0.46 via ge-0/0/5.0
224.0.0.5/32       *[OSPF/10] 02:22:37, metric 1
                       MultiRecv

buddy.inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

buddy.iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Illustration 4: The OSPF routing table in the redistributing router.

Connections between the routers are clearly working, looking at the routing table the OSPF system there is in fact a route that says 10.0.0.36/30 that is the network where buddy the virtual router is at.

```
lab@srxD-1> show route

inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.36/30       *[OSPF/150] 02:22:27, metric 0, tag 0
                    > to 10.0.0.45 via ge-0/0/9.0
10.0.0.44/30       *[Direct/0] 02:23:32
                    > via ge-0/0/9.0
10.0.0.46/32       *[Local/0] 02:24:55
                      Local via ge-0/0/9.0
10.210.14.0/24     *[Direct/0] 02:24:44
                    > via ge-0/0/0.0
10.210.14.149/32   *[Local/0] 02:24:55
                      Local via ge-0/0/0.0
176.16.1.2/32      *[OSPF/10] 02:22:27, metric 1
                    > to 10.0.0.45 via ge-0/0/9.0
192.168.0.0/24     *[Static/5] 02:25:48
                      Discard
192.168.0.1/32     *[Direct/0] 02:25:48
                    > via lo0.0
192.168.1.0/24     *[Static/5] 02:25:48
                      Discard
192.168.1.1/32     *[Direct/0] 02:25:48
                    > via lo0.0
192.168.2.0/24     *[Static/5] 02:25:48
                      Discard
192.168.2.1/32     *[Direct/0] 02:25:48
                    > via lo0.0
192.168.3.0/24     *[Static/5] 02:25:48
                      Discard
192.168.3.1/32     *[Direct/0] 02:25:48
                    > via lo0.0
224.0.0.5/32       *[OSPF/10] 02:25:51, metric 1
                      MultiRecv
```

Illustration 5: The routing table of the router running purely OSPF.

The virtual router buddy has an IS-IS routing table that includes the network of the OSPF router 10.0.0.44/30 and the dummy routes on the loopback interface.

```
lab@srxD-2> show route protocol isis

inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.3.5/32      *[IS-IS/15] 02:20:01, metric 10
                    > to 10.0.0.38 via ge-0/0/7.0

buddy.inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.44/30       *[IS-IS/160] 02:20:01, metric 20
                    > to 10.0.0.37 via ge-0/0/6.0
176.16.1.2/32      *[IS-IS/15] 02:20:01, metric 10
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.0.0/24     *[IS-IS/160] 02:19:58, metric 10
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.0.1/32     *[IS-IS/160] 02:19:58, metric 11, tag2 1
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.1.0/24     *[IS-IS/160] 02:19:58, metric 10
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.1.1/32     *[IS-IS/160] 02:19:58, metric 11, tag2 1
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.2.0/24     *[IS-IS/160] 02:19:58, metric 10
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.2.1/32     *[IS-IS/160] 02:19:58, metric 11, tag2 1
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.3.0/24     *[IS-IS/160] 02:19:58, metric 10
                    > to 10.0.0.37 via ge-0/0/6.0
192.168.3.1/32     *[IS-IS/160] 02:19:58, metric 11, tag2 1
                    > to 10.0.0.37 via ge-0/0/6.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

buddy.iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Illustration 6: IS-IS routing table of the redistributing router

3.6 BGP

3.6.1 BGP

BGP is an exterior gateway protocol that uses TCP on port 179 to establish connection between the routers. It is used to exchange information between routers in different autonomous systems. Routing information of BGP includes the complete route to each destination.
The Border Gateway Protocol exchanges information about network reachability with other BGP systems. It uses network reachability information to create a graph of AS connectivity, which enforce policy decisions at the Autonomous System level. There are two options in connections between networks, it can be private point-to-point link or an exchange.

3.6.2 BGP Message Types

BGP has four types of messages, each with own role in setting up, maintaining, or tearing down a BGP session. List of messages types:

- OPEN message
- UPDATE message
- KEEPALIVE message
- NOTIFICATION message

Important information is that these messages cannot be exchanged until two BGP routers have set up TCP session on port 179. Errors will display BGP NOTIFICATION messages that will close connection.

3.6.3 Autonomous Systems (AS)

Autonomous system is a group of router that are under single technical administration. Normally use a common set of metrics to share routing information with the set of routers.
Until 2007 AS were defined as a 16-bit integers, because of the size of Internet and number of devices we are running out of AS numbers. That is why in 2007 32 bit AS numbers were introduced. It allows us to use numbers from 0 to 4 294 967 295.

Number	Bits	Description	Reference
0	16	Reserved	[RFC1930]
1 - 23455	16	Public ASN's	
23456	16	Reserved for AS Pool Transition	[RFC6793]
23457 - 64534	16	Public ASN's	
64000 - 64495	16	Reserved by IANA	
64496 - 64511	16	Reserved for use in documentation/sample code	[RFC5398]
64512 - 65534	16	Reserved for Private Use	
65535	16	Reserved	
65536 - 65551	32	Reserved for use in documentation and sample code	[RFC4893][RFC5398]
65552 - 131071	32	Reserved	
131072 - 4199999999	32	Public 32-bit ASN's	
4200000000 - 4294967294	32	Reserved for Private Use	[RFC6996]
4294967295	32	Reserved	

Illustration above shows number of ASs and description

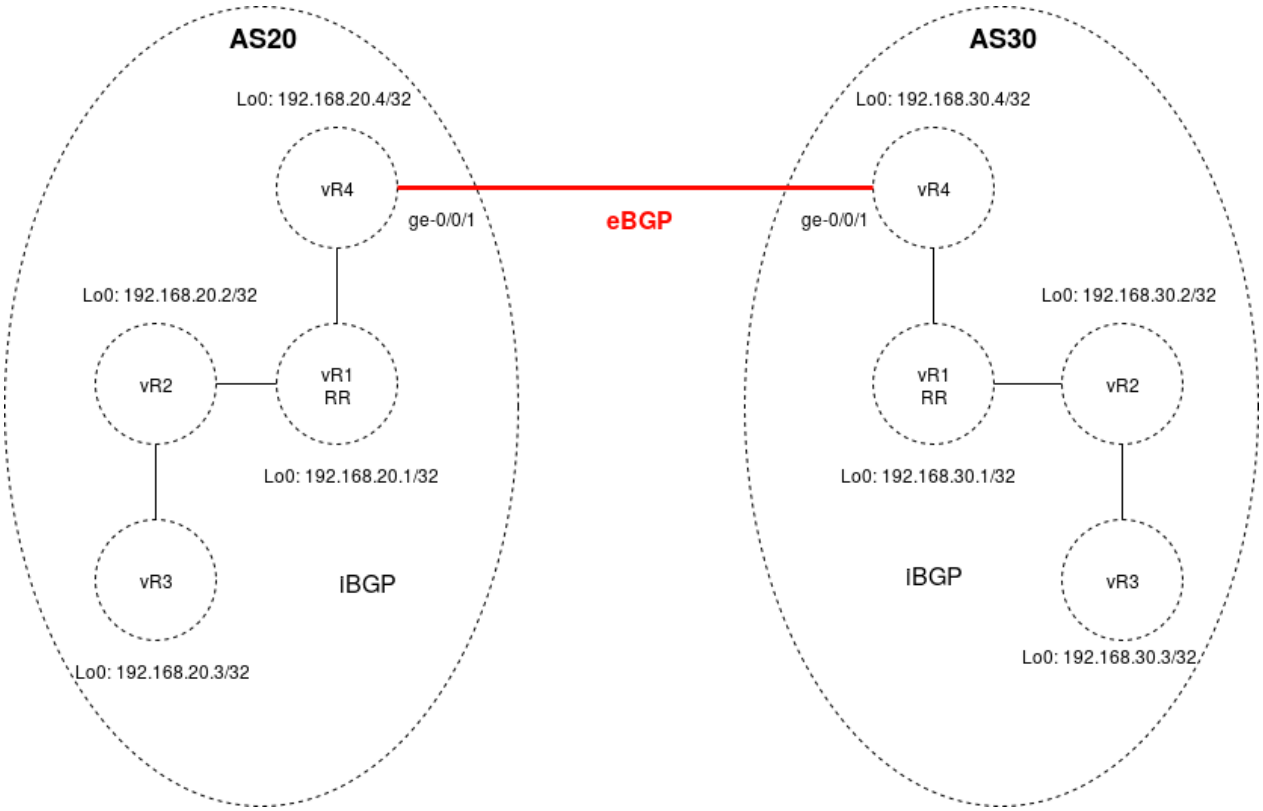
3.6.4 AS Path and Attributes

- BGP systems exchange routing information which include complete route to each destination and additional information about the route.
- AS path is the name of the route to each destination, additional route information is included in path attributes.
- BGP uses AS paths and path attributes to determine topology of the network.
- If BGP knows the topology, it can detect and eliminate routing loops, as well as selecting groups of routes to enforce preferences and policy decisions.

BGP uses modes to communicate with internal and external peers, two primary modes are:

- Internal BGP (iBGP)
- External BGP (eBGP)

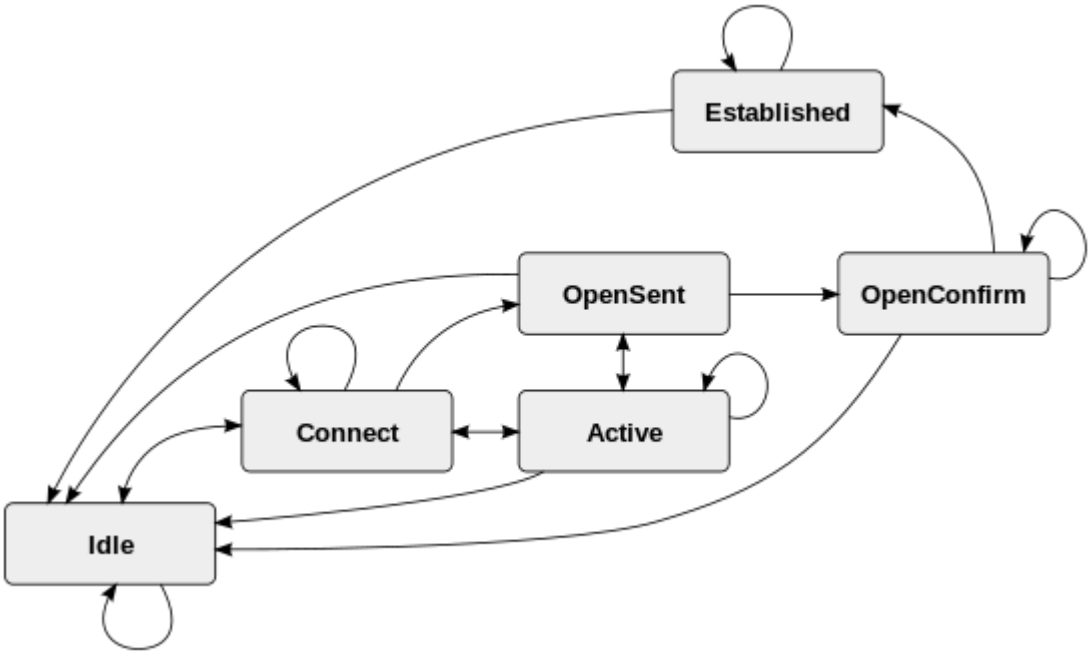
Internal BGP runs inside one Autonomous System, whereas External works just between multiple ASs.
Peer ASs establish links through and external peer BGP session. Then all route advertisement between external peers takes place by means of information exchange.
BGP network used in project:



3.6.5 Finite State Machine

BGP uses a Finite State Machine to make decisions. It consists of six states:

- Idle;
- Connect;
- Active;
- OpenSent;
- OpenConfirm;
- Established.



Idle State:
Router is searching for a routing table to check if a route to neighbour already exist.
Connect:
Router found a route to BGP neighbour. Three-way TCP handshake is completed.
OpenSent:
Open message with BGP session parameters is sent.
OpenConfirm:
Router received agreement for establishing session.
Active:
Router didn't receive agreement for establishing session.
Established:
Peering connection is up, routing process begins.

3.6.6 Internal BGP: (Full configuration in Appendix)

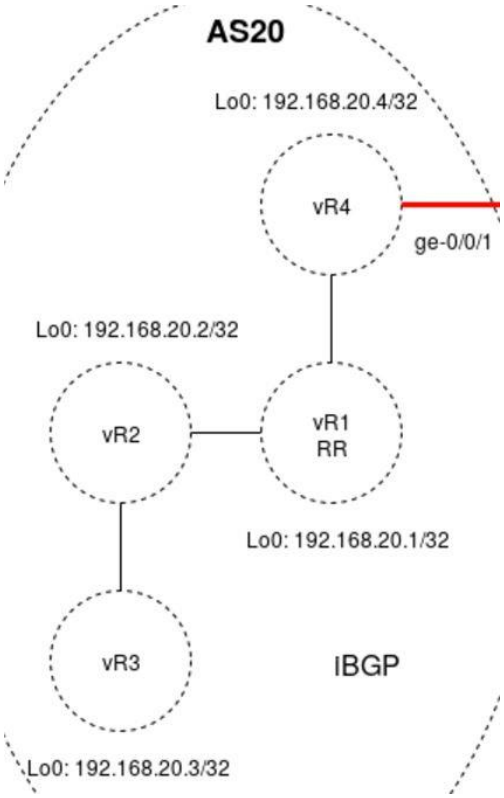


Figure 1 shows how iBGP topology looks like

In the network used in project all devices in Autonomous System 20 are meshed in the group internal-peers. Each device has configured loopback address, logical tunnel according to connection between each device and router-id which is requirement to use BGP and OSPF in a routing instance.

Example from device vR4:

```
vR4 {
    instance-type virtual-router;
    interface lt-0/0/0.6;
    interface ge-0/0/1.0;
    interface lo0.4;
    routing-options {
       router-id 192.168.20.4;
       autonomous-system 20;
    }
    protocols {
       bgp {
          group internal-peers {
             type internal;
             local-address 192.168.20.4;
             export [ send-direct send-ospf ];
             neighbor 192.168.20.1;
             neighbor 192.168.20.2;
             neighbor 192.168.20.3;
          }
```

It is possible to display the router-id values of a routing instances by using *show route instance detail.*

```
root@vSRX1> show route instance detail
  vR1:
            Router ID: 192.168.20.1
            Type: virtual-router    State: Active
            Interfaces:
              lt-0/0/0.2
              lt-0/0/0.1
              lo0.1
            Tables:
              vR1.inet.0             : 34 routes (13 active, 0 holddown, 8 hidden)
          vR2:
            Router ID: 192.168.20.2
            Type: virtual-router    State: Active
            Interfaces:
              lt-0/0/0.4
              lt-0/0/0.3
              lo0.2
            Tables:
              vR2.inet.0             : 20 routes (13 active, 0 holddown, 3 hidden)


          vR3:
            Router ID: 192.168.20.3
            Type: virtual-router    State: Active
            Interfaces:
              lt-0/0/0.5
              lo0.3
            Tables:
              vR3.inet.0             : 19 routes (12 active, 0 holddown, 0 hidden)


          vR4:
            Router ID: 192.168.20.4
            Type: virtual-router    State: Active
            Interfaces:
              lt-0/0/0.6
              ge-0/0/1.0
              lo0.
          Tables:
              vR4.inet.0             : 21 routes (13 active, 0 holddown, 1 hidden)
```

3.6.7 External BGP: (Full configuration in Appendix)

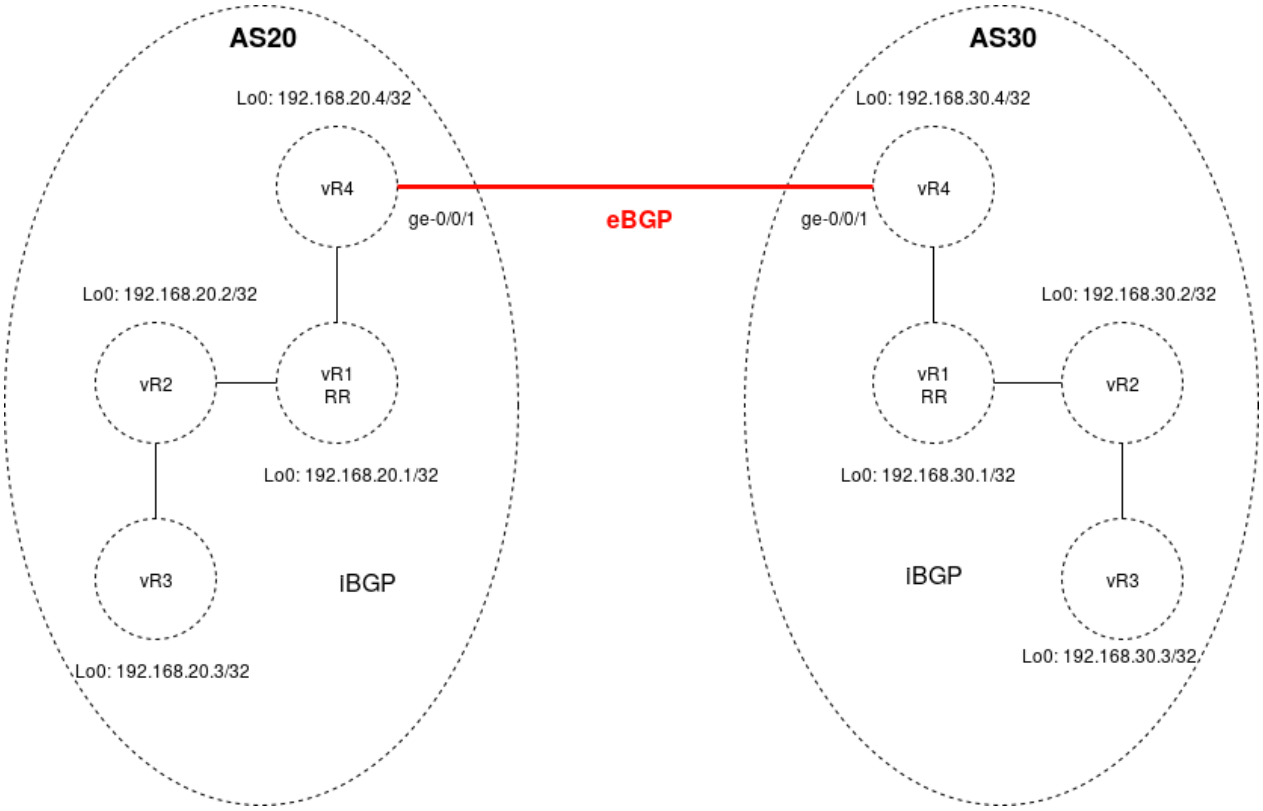


Figure 2 shows topology with 2 ASs and eBGP

In network used in project device vR4 in AS20 and vR4 in AS30 has BGP peer sessions to a group of peers called external-peers:
Example from device vR4 in AS20:

```
group external-peers {
    type external;
    advertise-peer-as;
    neighbor 172.20.5.2 {
        peer-as 30;
    }
}
```

It has defined neighbour with router-id in AS30.

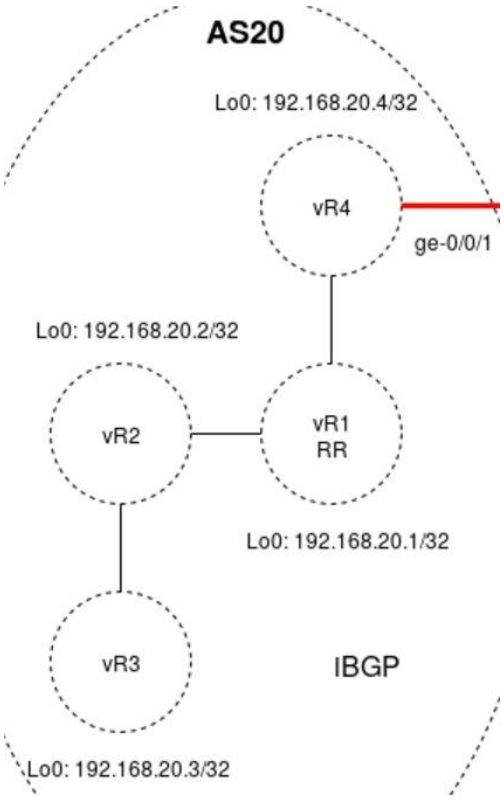
3.6.8 Problems with BGP

- iBGP scalability – each device must be configured with peers to each other in a full mash (everyone speaks directly to everyone). In a huge network, it can degrade performance of router due to memory or CPU requirements
- Instability – when the device is misconfigured it may get into cycle between up states and down states. It is known as a route flapping, is causes that other devices are informed about broken route and then broken device is removed from routing table
- Routing table growth – Internet infrastructure is growing all the time, and if the older routing devices are not able to handle huge number of routes, it can create a problem that the device will not be a good gateway between the parts of the internet.

- Load-balancing problem – balance in multi-homed networks due to limitations of the BGP peers. This may cause that one link is congested when other links are optimized well.
- Security issues – by default BGP routers accept BGP routes from other devices, it means that Internet is potentially vulnerable for disruptions. BGP is vulnerable to IP hijacking.

3.6.9 Route Reflector

Route Reflector is a device in internal BGP which has peer relationships with all iBGP devices, but also has configured the cluster statement and a cluster identifier. This cause that device vR4 is peered with just vR1 not with all devices in iBGP, the same situation happens in vR2 and vR3, where peering is configured just with route reflector.



Route Reflector is configured both in AS20, as well as in AS30.

3.6.10 Proofs, monitoring and troubleshooting

Command *show bgp group* shows information about groups including, ASs, policy options, peers, addresses, routing tables.

```
root@vSRX1>
show bgp
group
                Group Type: Internal    AS: 20                    Local AS: 20
                  Name: internal-peers  Index: 0                  Flags: <Export Eval>
                  Export: [ send-direct send-ospf ]
                  Options: <Cluster>
                  Holdtime: 0
                  Total peers: 3        Established: 3
                  192.168.20.4+54693
```

```
  192.168.20.2+179
  192.168.20.3+179
  vR1.inet.0: 3/24/16/0


Group Type: Internal    AS: 20                    Local AS: 20
  Name: internal-peers  Index: 1                  Flags: <Export Eval>
  Export: [ send-direct send-ospf ]
  Holdtime: 0
  Total peers: 1        Established: 1
  192.168.20.1+54090
  vR2.inet.0: 3/10/7/0


Group Type: Internal    AS: 20                    Local AS: 20
  Name: internal-peers  Index: 2                  Flags: <Export Eval>
  Export: [ send-direct send-ospf ]
  Holdtime: 0
  Total peers: 1        Established: 1
  192.168.20.1+57784
  vR3.inet.0: 3/10/10/0


Group Type: Internal    AS: 20                    Local AS: 20
  Name: internal-peers  Index: 3                  Flags: <Export Eval>
  Export: [ send-direct send-ospf ]
  Holdtime: 0
  Total peers: 3        Established: 1
  192.168.20.1+179
  192.168.20.2
  192.168.20.3
  vR4.inet.0: 0/7/6/0


Group Type: External                            Local AS: 20
  Name: external-peers  Index: 4                Flags: <>
  Options: <AdvertisePeerAs>
  Holdtime: 0
  Total peers: 1        Established: 1
  172.20.5.2+179
  vR4.inet.0: 2/3/3/0


Groups: 5  Peers: 9    External: 1    Internal: 8    Down peers: 2   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
vR1.inet.0            24          3          0          0          0          0
vR2.inet.0            10          3          0          0          0          0
vR3.inet.0            10          3          0          0          0          0
```

```
vR4.inet.0            10          2          0          0          0          0
vR1.mdt.0              0          0          0          0          0          0
vR2.mdt.0              0          0          0          0          0          0
vR3.mdt.0              0          0          0          0          0          0
vR4.mdt.0              0          0          0          0          0          0
```

Command *show bgp summary* displays number of groups, peers up and down, ASs, and state

```
root@vSRX1>
show bgp
summary
                Groups: 5 Peers: 9 Down peers: 2
                Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
                State|#Active/Received/Accepted/Damped...
                172.20.5.2               30         10         11       0       0        3:11
                Establ
                  vR4.inet.0: 2/3/3/0
                192.168.20.1             20         14         12       0       0        2:25
                Establ
                  vR2.inet.0: 3/10/7/0
                192.168.20.1             20         12         11       0       0        2:17
                Establ
                  vR3.inet.0: 3/10/10/0
                192.168.20.1             20          9         12       0       0        2:05
                Establ
                  vR4.inet.0: 0/7/6/0
                192.168.20.2             20         11         14       0       0        2:25
                Establ
                  vR1.inet.0: 0/7/4/0
                192.168.20.2             20          0          4       0       0        3:43
                Active
                192.168.20.3             20         10         12       0       0        2:17
                Establ
                  vR1.inet.0: 0/7/7/0
                192.168.20.3             20          0          4       0       0        3:43
                Active
                192.168.20.4             20         12         10       0       0        2:05
                Establ
```

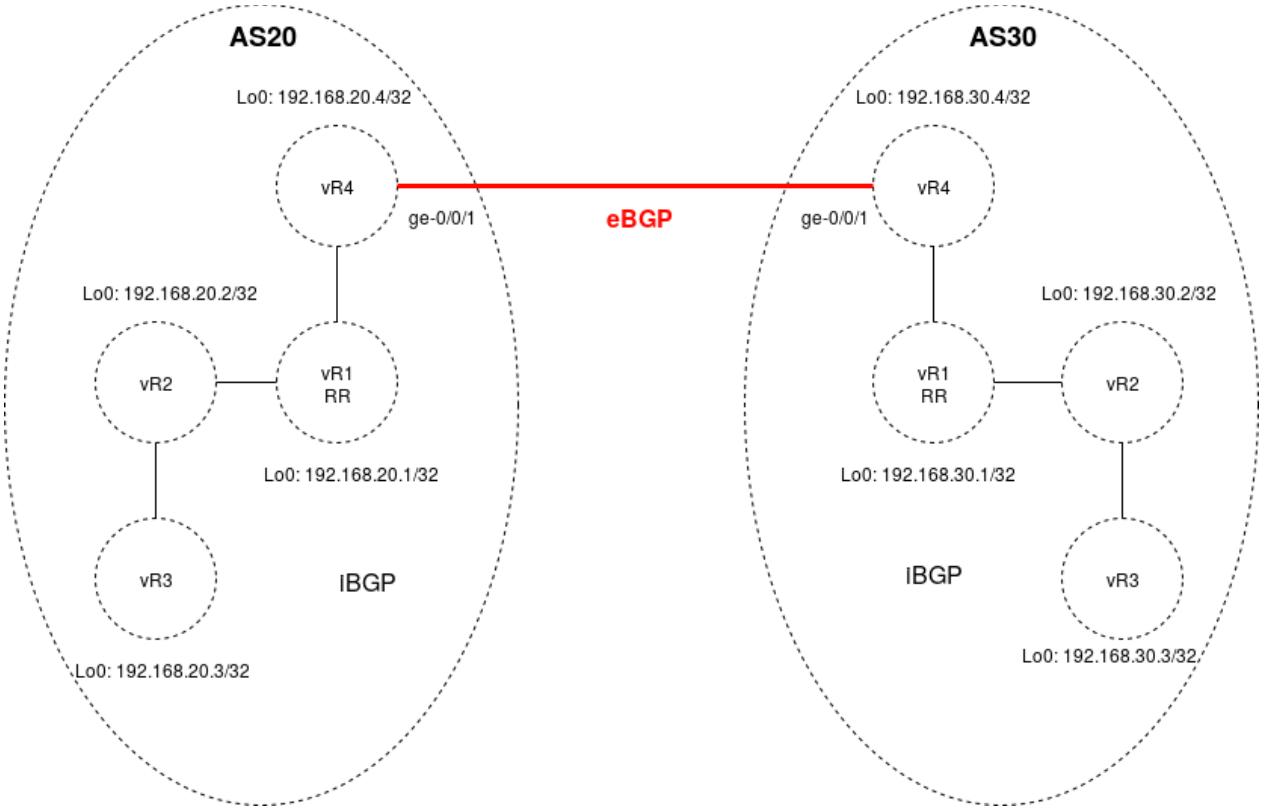
3.6.11 Sources:

https://en.wikipedia.org/wiki/Border_Gateway_Protocol

BGP presentation – EAL

3.7 Route Re-Distribution (BGP/OSPF)

In route re-distribution exercise the goal was to export OSPF routes from AS20 (OSPF was used as an IGP) to eBGP between AS20 and AS30.



It was possible by configuring export ospf policy under [edit policy options] including statements:

```
from protocol ospf
then accept
```

And then the policy send-ospf was inserted to vR4, under [protocols bgp] hierarchy in external peers.
Because of this device vR3 in AS20 can reach device vR3 in AS30.

To prove this, we used *show route* command to display full routing table. Below was attached just routing table from device vR4 in AS20.

```
vR4.inet.0: 18 destinations, 22 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.20.2.0/30      *[OSPF/10] 00:35:13, metric 2
                    > to 172.20.4.2 via lt-0/0/0.6
                    [BGP/170] 00:34:53, localpref 100, from 192.168.20.1
                      AS path: I
                    > to 172.20.4.2 via lt-0/0/0.6
172.20.3.0/30      *[OSPF/10] 00:35:08, metric 3
                    > to 172.20.4.2 via lt-0/0/0.6
```

```
172.20.4.0/30      *[Direct/0] 00:36:03
                    > via lt-0/0/0.6
                    [BGP/170] 00:34:53, localpref 100, from 192.168.20.1
                      AS path: I
                    > to 172.20.4.2 via lt-0/0/0.6
172.20.4.1/32      *[Local/0] 00:36:03
                      Local via lt-0/0/0.6
172.20.5.0/24      *[Direct/0] 00:36:01
                    > via ge-0/0/1.0
                    [BGP/170] 00:35:57, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
172.20.5.1/32      *[Local/0] 00:36:03
                      Local via ge-0/0/1.0
172.30.2.0/30      *[BGP/170] 00:08:23, MED 2, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
172.30.3.0/30      *[BGP/170] 00:08:23, MED 3, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
172.30.4.0/30      *[BGP/170] 00:35:57, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
192.168.20.1/32    *[OSPF/10] 00:35:13, metric 1
                    > to 172.20.4.2 via lt-0/0/0.6
                    [BGP/170] 00:34:53, localpref 100, from 192.168.20.1
                      AS path: I
                    > to 172.20.4.2 via lt-0/0/0.6
192.168.20.2/32    *[OSPF/10] 00:35:13, metric 2
                    > to 172.20.4.2 via lt-0/0/0.6
192.168.20.3/32    *[OSPF/10] 00:35:08, metric 3
                    > to 172.20.4.2 via lt-0/0/0.6
192.168.20.4/32    *[Direct/0] 00:36:29
                    > via lo0.4
192.168.30.1/32    *[BGP/170] 00:08:23, MED 1, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
192.168.30.2/32    *[BGP/170] 00:08:23, MED 2, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
192.168.30.3/32    *[BGP/170] 00:08:23, MED 3, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
192.168.30.4/32    *[BGP/170] 00:35:57, localpref 100
                      AS path: 30 I
                    > to 172.20.5.2 via ge-0/0/1.0
224.0.0.5/32       *[OSPF/10] 00:36:31, metric 1
                      MultiRecv
```

4 Security

(Full configuration files are on GitHub:
https://github.com/miskor/Project_network/tree/master/APPENDIX/Stage-3%20Security)

4.1 Routing Policies

Routing Policies allow users to control the flow of routing information between the routing protocols and the routing tables and between the routing tables and the forwarding table. Routing policy allows you to control which routes the routing protocols store in and retrieve from the routing table.

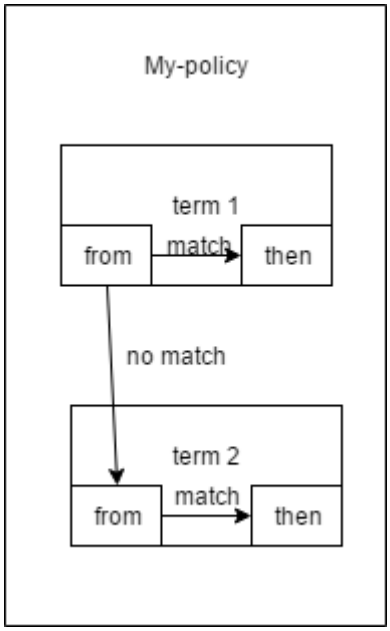
Routing policy was used because:

- It was not an intention to import all routes into routing table. Routes were specified by proper match criteria in terms.
- According to route redistribution it was intended to transfer active routes learned from another routing protocol (BGP/OSPF, IS-IS/OSPF)

4.1.1 Default Routing Policies

Protocol	Import Policy	Export Policy
BGP	Accept all BGP routes and import into `inet.0`	Accept all active BGP routes
OSPF	Accept all OSPF routes and import into `inet.0`	Reject everything (protocol floods by default)
IS-IS	Accept all IS-IS routes and import into `inet.0`	Reject everything (protocol floods by default)
RIP	Accept all RIP routes from explicitly configured neighbors and import into `inet.0`	Reject everything

4.1.2 Routing Policy Flow:



Number of terms in routing policy is equal or bigger than 0, the software evaluates terms until it reaches a terminating action or end policy. Names of policies and terms are defined by user. Each term contains “from” and “then” statement. The first describe match condition(s) and the second one describes action that is taken if a “from” statement is matched.

Match criteria that was used in project:

- from protocol direct
- from protocol ospf
- route-filters (192.168.0.0/22 longer; 10.0.0.44/30 exact; 10.0.0.36/30 exact)

"from" statement describes match conditions

Match types used in project: longer, exact.

- Exact match the specified prefix and mask exactly (10.0.0.36/30 exact)
- Longer match routes that have longer masks (192.168.0.0/22 longer)

Actions
"then" statement describes the actions to take if a "from" statement is matched.
Just one terminating action was used in project, it was "accept"

Implementing Routing Policy.
Defined routing policy is always located under the [edit policy-options] hierarchy on Juniper Device. (configuration in Appendix)
Quick set-up:
[edit policy-options]
set policy-options policy statement ospf-isis term 1 from protocol ospf
set policy-options policy statement ospf-isis term 1 from route-filter 192.168.0.0/22 longer
set policy-options policy statement ospf-isis term 1 then accept

```
policy-options {
                        policy-statement ospf-isis {
                            term 1 {
                                from {
                                    protocol ospf;
                                    route-filter 192.168.0.0/22 longer;
                                }
                                then accept;
                            }
                        }
```

Applied routing policies as import or export policies can be found at different hierarchy levels (for example under routing-instances or protocols)

[edit protocols]
set isis export ospf-isis
set isis interface ge-0/0/7.0
set isis interface lo0.0

```
protocols {
                    isis {
                        export [ ospf-isis send-direct-to-isis-neighbors ];
                        interface ge-0/0/7.0;
                        interface lo0.0;
```

```
        }
```

4.1.3 Sources:

Routing Policy presentation – EAL

https://www.juniper.net/documentation/en_US/junos/information-products/pathway-pages/config-guide-policy/config-guide-policy.html

4.2 Route Redistribution

In Juniper devices routes are redistributed by importing and exporting routing policies in specific protocols or in the general "protocols" hierarchy. The principles are discussed in the chapter about Routing Policies.

Route Redistribution is a term from Cisco systems, and does not really apply to the same way in Juniper. In Juniper within a specific protocol routes are always shared but there are default import/export policies for each protocol, the ones used in this project are shown in the table below.

Importing or Exporting Protocol	Default Import Policy	Default Export Policy
BGP	Accept all received BGP IPv4 routes learned from configured neighbors and import into the inet.0 routing table. Accept all received BGP IPv6 routes learned from configured neighbors and import into the inet6.0 routing table.	Readvertise all learned BGP routes to all BGP speakers, while following protocol-specific rules that prohibit one IBGP speaker from readvertising routes learned from another IBGP speaker, unless it is functioning as a route reflector.
IS-IS	Accept all IS-IS routes and import into the inet.0 and inet6.0 routing tables. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)
OSPF	Accept all OSPF routes and import into the inet.0 routing table. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)

Table 1: Default import and export policies for protocols. Full tale is available at the Juniper home page at: https://www.juniper.net/documentation/en_US/junos12.3/topics/reference/general/policy-protocol-import-export-defaults.html

In the example below a policy is used to redistribute OSPF routes into eBGP:

```
policy-options {
    policy-statement send-ospf {
        term 1 {
            from protocol ospf;
```

```
            then accept;
        }
    }
}
```

The above policy is used to accept routes from the OSPF protocol.

```
protocols {
    bgp {
        group external-peers {
            type external;
            export send-ospf;
            advertise-peer-as;
            neighbor 172.20.5.2 {
                peer-as 30;
            }
        }
    }
}
```

The policy is then exported in to the BGP group with the external type, to have eBGP redistribute the OSPF routes.
From operational mode, the test command is used to see the positive police matches for a specific prefix like show in this example for the policy “send-ospf” and the prefix 172.20.5.0/24.

```
root@vSRX1> test policy send-ospf 172.20.5.0/24

vR1.inet.0: 18 destinations, 25 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.20.5.0/24      *[BGP/170] 00:05:52, localpref 100, from 192.168.20.4
                     AS path: I
                   > to 172.20.4.1 via lt-0/0/0.1

vR2.inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.20.5.0/24      *[BGP/170] 00:05:51, localpref 100, from 192.168.20.1
                     AS path: I
                   > to 172.20.2.2 via lt-0/0/0.3

vR3.inet.0: 17 destinations, 20 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.20.5.0/24      *[BGP/170] 00:05:51, localpref 100, from 192.168.20.1
                     AS path: I
                   > to 172.20.3.2 via lt-0/0/0.5

vR4.inet.0: 18 destinations, 22 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.20.5.0/24      *[Direct/0] 00:07:02
                   > via ge-0/0/1.0
```

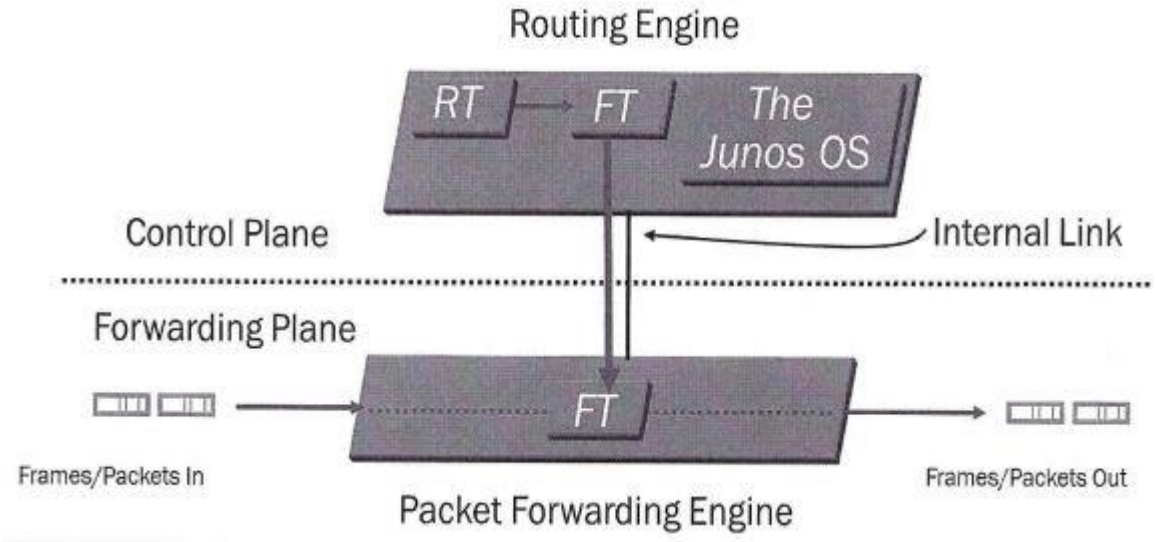
```
                [BGP/170] 00:06:59, localpref 100
                  AS path: 30 I
                > to 172.20.5.2 via ge-0/0/1.0
172.20.5.1/32      *[Local/0] 00:07:03
                    Local via ge-0/0/1.0

Policy send-ospf: 5 prefix accepted, 0 prefix rejected
```

This output is from running the above command on the first router in the BGP example. We see that the prefix was accepted 5 times, ones for each virtual instance except the vR4 that has the local route as well, and never rejected.

4.3 RE/PFE

All platforms running the Junos OS share a common design goal: clean separation of control and forwarding functions.



The Routing Engine is in the control plane, it is the brain of the Juniper Device, responsible for performing protocol updates and system management, it runs various daemons that reside inside a protected memory environment. The Routing Engine maintains the routing tables, bridging table and forwarding table and connects to the Packet Forwarding Engine through an internal link. The RE provides the CLI in addition to the J-Web GUI.
The Packet Forwarding Engine is responsible for forwarding transit traffic through the device. In many Juniper platforms, the PFE uses application-specific integrated circuits (ASICs) for increased performance.

The PFE receives the layer 2 and layer 3 forwarding table (FT) from Routing Engine. FT updates are a high priority for the Junos OS kernel and are performed incrementally. It implements various services such as policing, stateless firewall filtering, and class of service

Transit Traffic consists of all traffic that enters an ingress network port, is compared against the forwarding table entries, and is forwarded out an egress network port toward its destination. Exception Traffic does not pass through the local device but rather requires some form of special handling. Examples of exception traffic:

- Packets addressed to the chassis (telnet, pings traceroutes)
- Traffic that requires the generation of ICMP messages

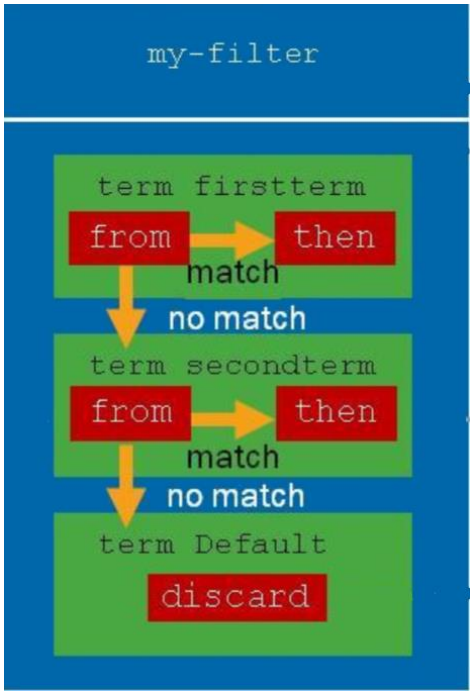
Exception traffic is rare-limited on the internal link to protect the RE from potential DoS attacks

Source: Introduction to the Junos Operating System – Student Guide Revision V-15.a

4.4 Firewall Filters

Firewall filters control the traffic entering and leaving a networking device in a stateless fashion. Stateless means that the firewall filters uses static values like protocol, source, and destination address it cannot interpret traffic patterns or data flows. The firewall filters process each packet individually.
Firewall filters can perform the following actions:

- Restrict traffic destined for the Routing Engine based on its source, protocol, and application.
- Limit the traffic rate of packets destined for the Routing Engine.
- Process fragmented packages by configuring the filter to support this.



In Juniper devices firewall filters are built of a list of terms. Each "term" has a from statement that is used for matching the package. If this term is true the "then" statement of that term takes effect. The terms are matched sequentially starting from the first, which means that the order they are appear in will determine the order that the terms are matched.

The from statements describes the match condition. Some common match criteria are:

- Match most header field in the package.
- Match conditions include
  - Numeric range
  - Address
  - Bit field

Actions can terminate (while accepting, discarding or rejecting) the sequence of terms. They may also modify the flow, perform monitoring task, modify the forwarding class or perform rate limiting.

Common firewall filter actions are as follows:

- Terminating actions:
  - accept
  - discard
  - reject
- Flow control:
  - next term
- Action modifiers:
  - count, log, and syslog
  - forwarding-class and loss-priority policer

- Everything that is not explicitly allowed will be discarded.

4.4.1 Configuration

Firewall filters are configured in the "firewall" hierarchy.
Firewall {

The next line limits the filters to IPv4 packages.

```
family inet {
          filter filter1 {
```

This is an example of filtering by protocol, traffic except TCP and UDP is accepted.

```
                    term term1 {
                              from {
                                        protocol-except [tcp udp];
                              }
                              then {
                                        accept;
```

```
                    }
                }
```

Taking into account the above term, the next term is only reached for TCP and UDP traffic since all other types are accepted above. The next term will reject packages from the 192.168.0.0/16 network.

```
                term term2 {
                    from {
                        address 192.168/16;
                    }
                    then {
                        reject;
                    }
                }
```

Again, taking into account the above rules all packages that are TCP or UDP and not coming from the 192.168.0.0/16 network are accepted by the next term, after increasing the counter "pkts_remote_man". This count feature is used for monitoring.

```
                term term3 {
                    from {
                        destination-port [ssh telnet];
                    }
                    then {
                        count pkts_remote_man;
                        accept;
                    }
                }
```

The last term is the default term, that rejects everything that did not match any of the terms above.

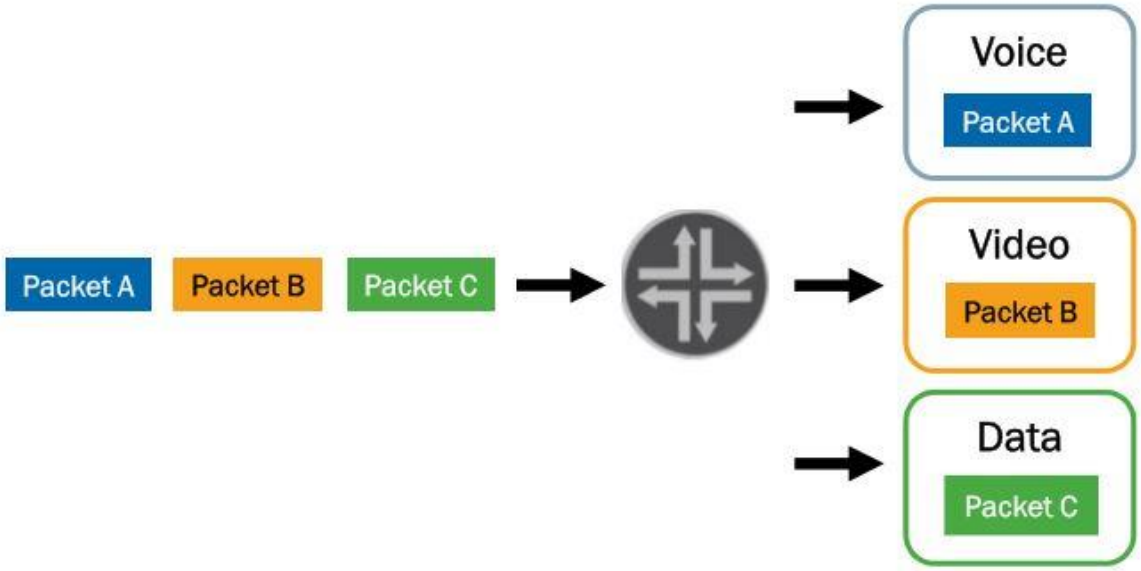
```
                term term4 {
                    then {
                        reject;
                    }
                }
            }
}
```

Firewall filters play a role in Class of Service, to classify the traffic as will be discussed in the "Class of Service" chapter and as mentioned earlier can also limit the amount of traffic according to the condition of the terms.

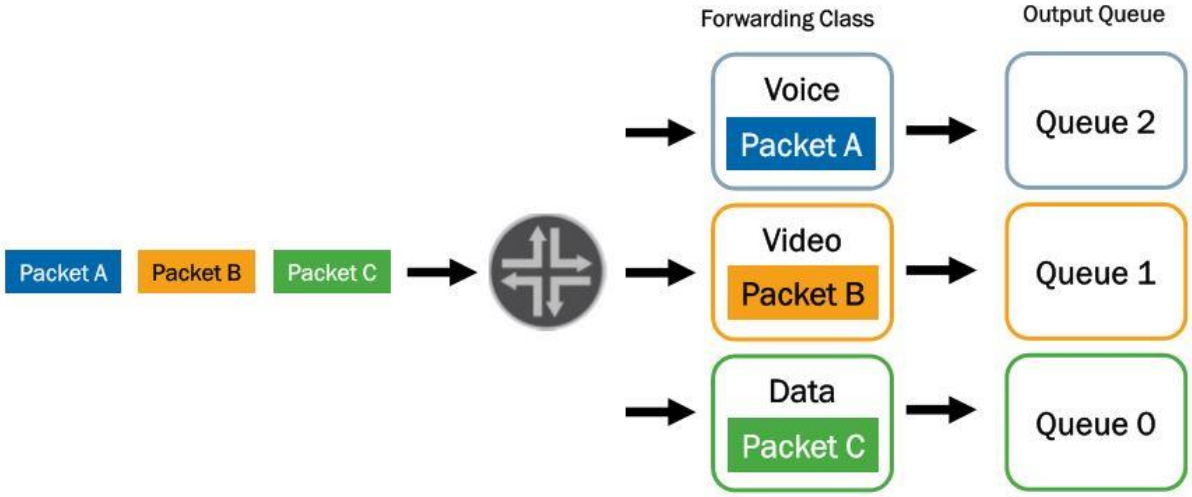
4.5 Class of Service

Class of service provides to the device mechanisms for categorizing traffic with different types of payload. The classification is based mostly on the type of application, which can be divided in three

main categories voice, video, data.

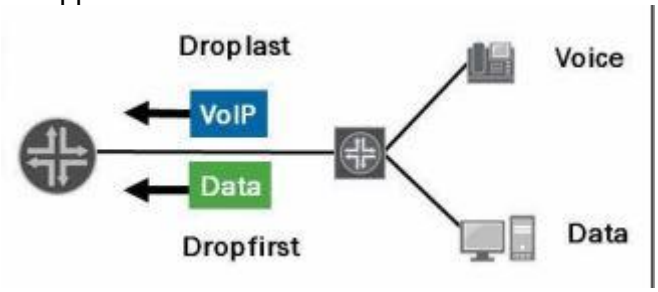


CoS prioritize latency-sensitive traffic (for example Voice over Internet Protocol), it can allocate bandwidth of connection for different classes. Class of Service to assign an output to a packet, it must associate packet with the forwarding class. Forwarding classes also can identify traffic that should receive proper treatment.



4.5.1 Loss Priority

Loss priority can be configured if user want to tell the system which priority it should give to packet to choose what should be dropped at first.

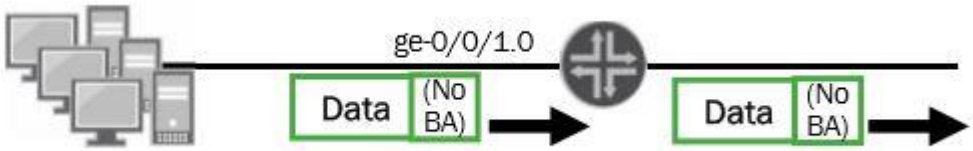


4.5.2 CoS Deployment Models

There are two prime deployment models:

1. Multifield classifier is used in in-the-box model

   It is defined under [edit firewall family inet] hierarchy with "from" and "then" statement and then applied under [edit interfaces <interface> unit <unit number> family <family name> filter] with *input/output <name of classifier>*



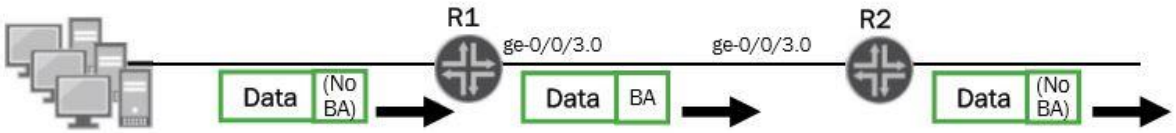
2. In across-the-network model Behaviour Aggregate rewrite, and BA classifier is in the core and multifield classifier at edge.

   Behaviour Aggregate rewrite is defined under [edit class-of-service]

```
interfaces {
     ge-0/0/2 {
          scheduler-map my-sched-map;
          unit 0 {
               rewrite-rules {
                    inet-precedence default;
               }
          }
     }
```

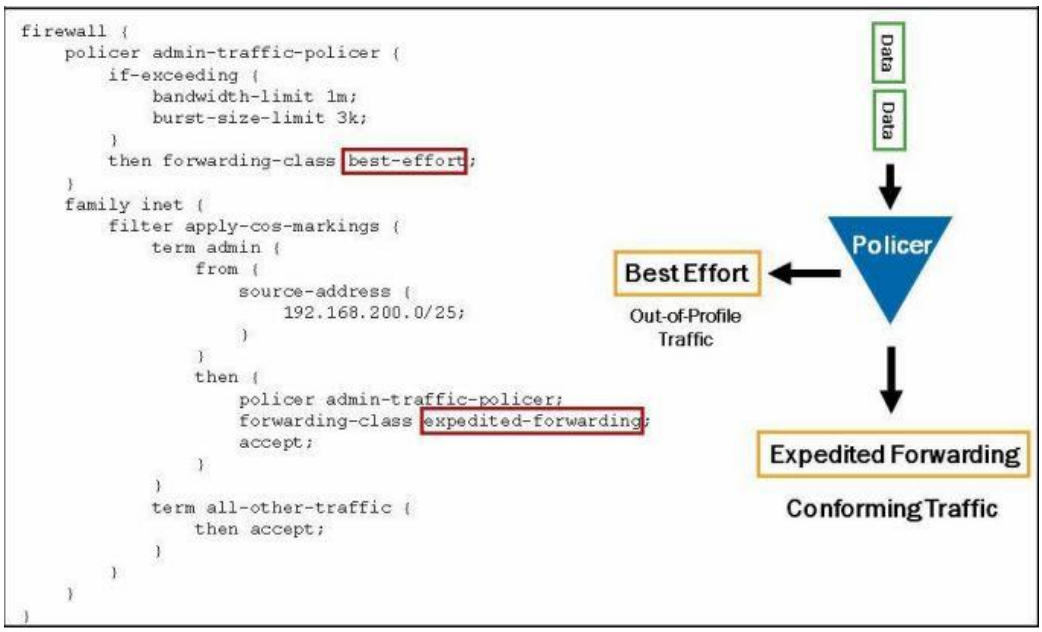
While BA Classifier is defined also under [edit class-of-service]

```
interfaces {
     ge-0/0/3 {
          scheduler-map my-sched-map;
          unit 0 {
               classifiers {
                    inet-precedence default;
               }
```



4.5.3 Policers

Policers allows to limit traffic to a specified bandwidth and burst size. It is possible to configure Junos device to assign forwarding class or loss priority to traffic exceeding the configured limits. These policers can be configured using *forwarding-class* and/or *loss-priority* statement after "then" statement of policer.



4.5.4 Queuing

Juniper devices are associating each forwarding class with queue number. On most devices that runs Junos OS default queues mapping looks like:

- 0: best-effort
- 1: expedited-forwarding
- 2: assured-forwarding
- 3: network-control

It is possible to display current queue and forwarding class mapping using *show class-of-service forwarding-class*

Display:

```
lab@srxC-1> show class-of-service forwarding-class
Forwarding class                  ID                      Queue  Policing priority    SPU priority
  best-effort                     0                           0          normal             low
  expedited-forwarding            1                           1          normal             low
  assured-forwarding              2                           2          normal             low
  network-control                 3                           3          normal             low
```

4.5.5 Defining Forwarding Classes

Forwarding classes can be configured under [edit class-of-service forwarding-class], using command
Set forwarding-classes queue <number> <name>

4.5.6 Scheduling overview

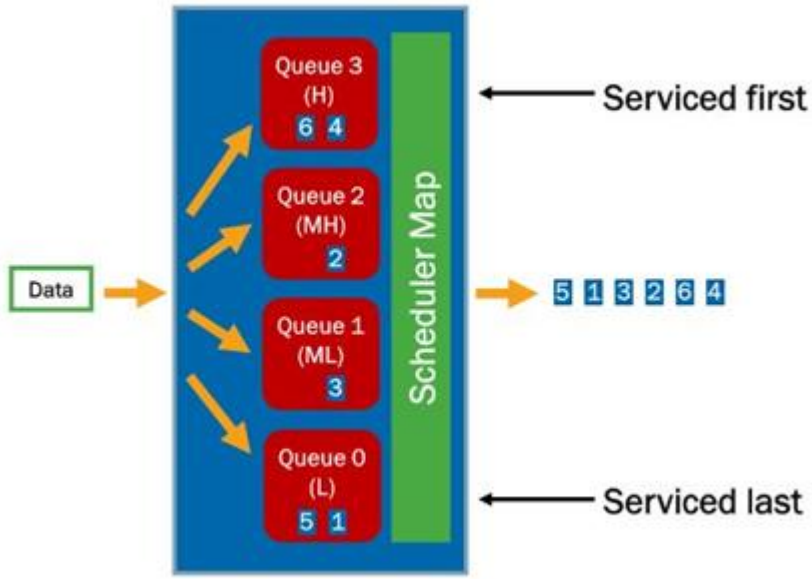
There are 4 main components of scheduling:

1. Priority
2. Transmission rate
3. Buffer size
4. RED configuration

Point 1 and 2 define the order of transmitting packets, while 3 and 4 define the storage and packets to drop.

4.5.7 Queue Priority

According to assigned priority queues can receive service, most common priorities include:



- High
- Medium high
- Medium low
- Low

4.5.8 Defining CoS on Juniper Devices

- Schedulers are configured under the [edit class-of-service schedulers] hierarchy:
  set <name of scheduler> transmit-rate percent 40
  set <name of scheduler> buffer-size percent 40
  set <name of scheduler> priority low
  example from project:

```
schedulers {
    best-effort-sched {
        transmit-rate percent 40;
        buffer-size percent 40;
        priority low;
    }
```

- Forwarding classes and queues are associated with schedulers by scheduler maps:
  [edit class-of-service scheduler-maps]
  Set <name of sched map> forwarding-class best-effort scheduler <name of sched.>
  Example from project:

```
scheduler-maps {
        my-sched-map {
            forwarding-class best-effort scheduler best-effort-sched;
```

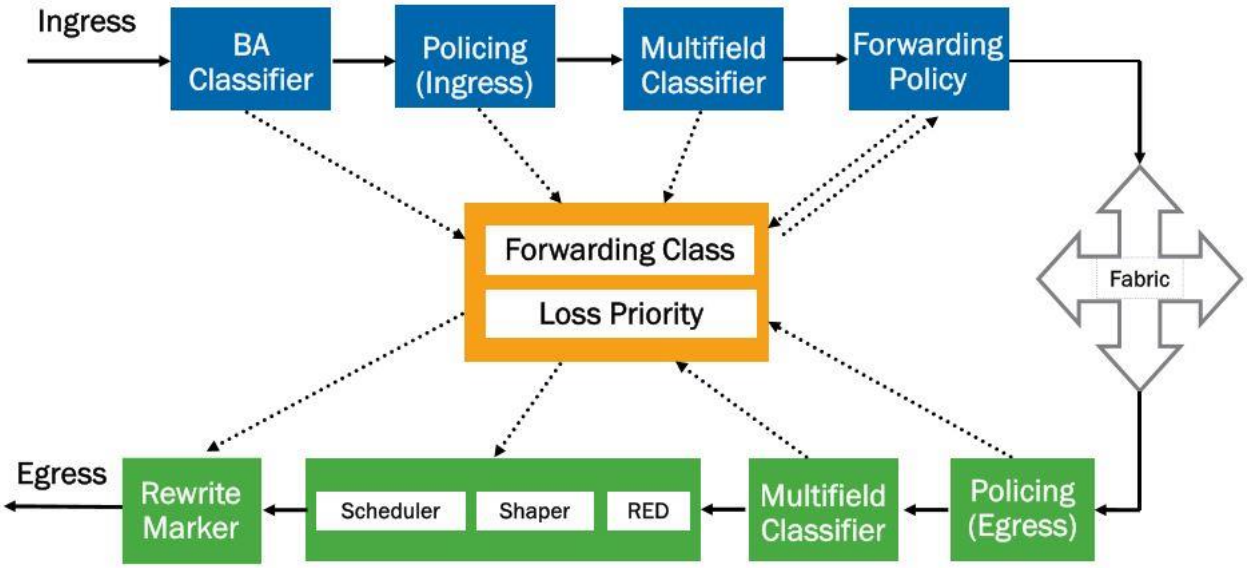
- Scheduler maps can be applied to interfaces under [edit class-of-service interfaces] hierarchy.
  [edit class-of-service interfaces]
  set <interface> scheduler-map <name of sched. Map>

  Example from project:

```
interfaces {
    ge-0/0/2 {
        scheduler-map my-sched-map;
```

4.5.9 CoS Processing

Illustration below shows how Class of Service is processing.



5 Conclusion

5.1 Conclusion about Project

Conclusion after this document is that Juniper SRX devices and systems are well structured with security and processing, user can easily monitor traffic, can check processes in the routing/switching device.

5.2 Project Management

During the second semester, we have been working on configurations. For the project management was used project plan, as well as online software like Slack, Facebook, and draw.io. Sometime

topologies were made on whiteboard in school building and on normal sheet of paper. Time frames were defined by teacher – Peter Liljehof Thomsen - in the project plan.