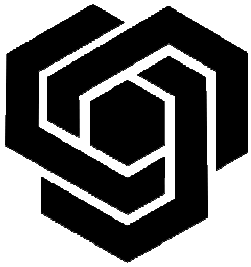

Master Thesis

Network design with guaranteed End-to-End QoS

Supervisor:
Sri Hanuma Chitty

Supervisor:
prof. Alexander Tsenov

Student:
Radostina Gercheva

Aalborg, Denmark
June 2013

Table of Contents

Introduction....4
1. State of art....5
1.1 The MPLS in Mobile Backhaul Initiative....5
1.2 Migration from 2G/3G to LTE....5
1.3 Current Status of the 16 bit Autonomous System Number Pool....7
1.4 Traffic Classification....9
1.5 Self-Organized networks....10
2. Problem definition....11
3. Goals....12
4. Methods....13
4.1 Multi-Protocol Label Switching....13
4.1.1. MPLS label....15
4.2. Resource reSerVation Protocol....16
4.2.1. RSVP basis....17
4.2.2. RSVP message types....18
4.3. Virtual Private Network (VPN) - L2VPN and L3VPN....23
4.3.1. MPLS Layer 3 VPNs....23
4.3.2. MPLS Layer 2 VPNs....26
4.3.3. Multi-Point Connectivity....27
4.4. Dynamic routing protocol Border Gateway Protocol (BGP)....29
4.4.1. BGP Peers....29
4.4.2. External BGP sessions....29
4.4.3. Internal BGP sessions....29
4.4.4. Relations establishment in BGP....30
4.4.5. BGP message types....30
4.5. Dynamic routing protocol - Open Shortest Path First....31
4.5.1. Shortest path Algorithm....32
4.5.2. OSPF Path Cost....32
4.5.3. Areas and Area Border Routers....32
4.5.4. Link-state packet types and OSPF header....33
4.5.6. OSPF header fields....34
4.6. Make-Before-Break rule....34
4.7. Fast Reroute....34
4.7.1. Link Protection....35
4.7.2. Node Protection....35
4.7.3. Bandwidth Protection....35
4.8. Quality of service....35
4.9. Class of Service....36
4.9.1. Type of Service....37
4.9.2. Differentiated Services Codepoint....38
4.10. Used equipment for building the network architectures....39
4.10.1. MX960....39

4.10.2. SRX210....40
4.10.3. NTOOLS traffic generator....41
4.10.4 JDSU testing module....42
5. Verification of the proposed network architecture approach 43
5.1. Logical System Configuration....47
5.3. Interfaces in the designed evaluation network architecture....52
5.4. Quality of Service in the designed NGN network architectures....55
5.4.1. Configuring CoS-Based Forwarding....55
5.4.2. Scheduler-maps....57
5.4.3. Schedulers....58
5.5. Designing Plain IP architecture....59
5.6. Implementing IP/MPLS network model....60
5.6.1. Extending the function of OSPF....61
5.6.2. Configuring MPLS protocol....61
5.6.3. Configuring RSVP-TE protocol....62
5.6.4. Configuring BGP protocol....63
5.6.5. Label Switched Paths implementation 64
5.6.6. Building Layer3VPN service....66
6. Results from the evaluation of the designed network architectures....69
6.1 Performance of the network architecture 69
6.2. IGP protocol OSPF....69
6.3. Signaling protocol RSVP....72
6.4. Performance of MPLS....74
6.5. Protocol BGP....76
6.6. Performance of established Layer 3 VPN Service 80
6.7. Quality of Service and Latency verification of the tested network architectures....81
6.7.1 QoS evaluation 81
6.7.2. Latency measurements in the built network models....85
Conclusion....86
Future development....87
References:....88
Appendix A....89

Introduction

There are many benefits that have led network operators to deploy MPLS in parts of the network beyond the core, deploying it in aggregation as well as access networks. However, these deployments are typically not in a single MPLS domain. In parallel, Ethernet has become the preferred cost-effective choice for building infrastructure, with increased investments being put into building Ethernet-based packet networks and services migrating away from time-division multiplexing (TDM) transport. These two trends have influenced many developments in MPLS, including ways to deliver MPLS services over Ethernet for both unicast and multicast traffic.

From a business standpoint, network operators continue to be challenged as they look for ways to make service delivery cost-effective and efficient. The need for one converged packet network to deliver all fixed and mobile services, regardless of the last mile or access technology, keeps getting stronger. These factors combined have led the industry to innovate and invest in features and functions that take MPLS to the access network and build single domain MPLS networks. Seamless MPLS is the umbrella portfolio that addresses this need, and provides the framework for taking MPLS to the access in a scalable fashion, extending the benefits of traffic engineering and guaranteed servicelevel agreements (SLAs) with deterministic network resiliency.

In a typical network, the traffic through the network is heterogeneous and consists of flows from multiple applications and utilities. Many of these applications are unique and have their own requirements with respect to network parameters such as delay, jitter, etc. Unless these requirements are met, the quality and usability of these applications will be severely compromised. While meeting these requirements in a Local Area Network (LAN) with its huge bandwidth might be easy, it usually is a challenge to meet them on the WANs, which have bandwidth constraints.

1. State of art

1.1 The MPLS in Mobile Backhaul Initiative

The Multi-Protocol Label Switching (MPLS) in Mobile Backhaul Initiative (MMBI) sugests a framework for the transport of radio traffic over packet-based access, aggregation, and core networks as shown in Figure 1.1. The framework pays attention to the possible deployment models and provides recommendations on how to use MPLS in each of these models. Creating such a manual allows vendors and service providers to select the appropriate feature sets for their specific scenario.

Areas within the scope of the MMBI include Quality of Service (QoS) considerations (for example, to support specific service types), resiliency capabilities, clocking and synchronization, Operations Administration and Maintenance (OAM), and support for various transport network layers and Long Term Evolution (LTE). [2]

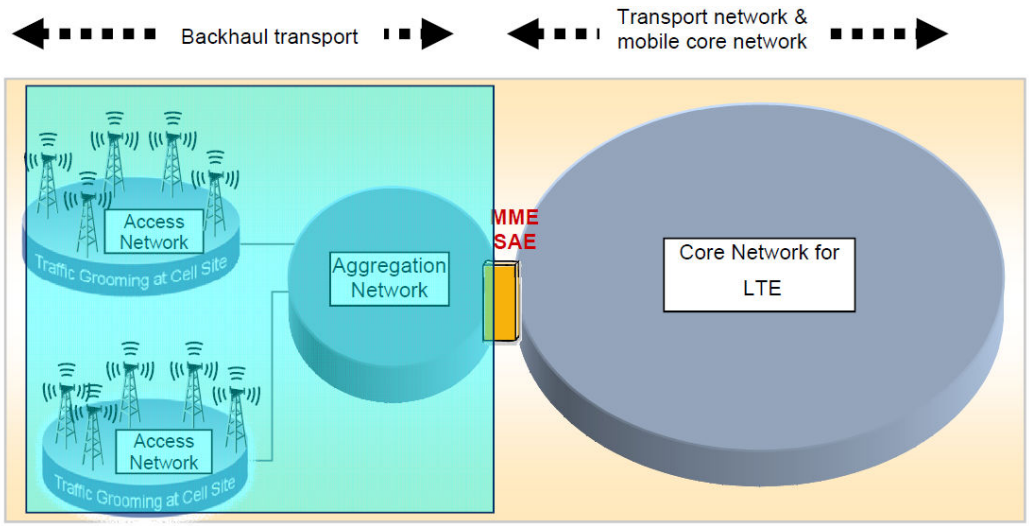

Figure 1.1 - MPLS in Mobile Backhaul Initiative [2]

1.2 Migration from 2G/3G to LTE

In a 2G/3G Radio Access Network (RAN), the base station manages the radio interface with the mobile station and the controller controls one or more base stations to provide handling of functions such as radio-channel setup, handovers, and so on. A hub-and-spoke topology allows communication from base station to controller and controller to base station as shown in Figure 1.2. In an LTE RAN, the base station itself has some controller functionality and can communicate with other base stations directly via Full-Mesh topology. An LTE base station communicates with a Mobility Management Entity (MME) and a serving gateway (S-GW) most commonly via a star topology as shown in Figure 1.3. [2]

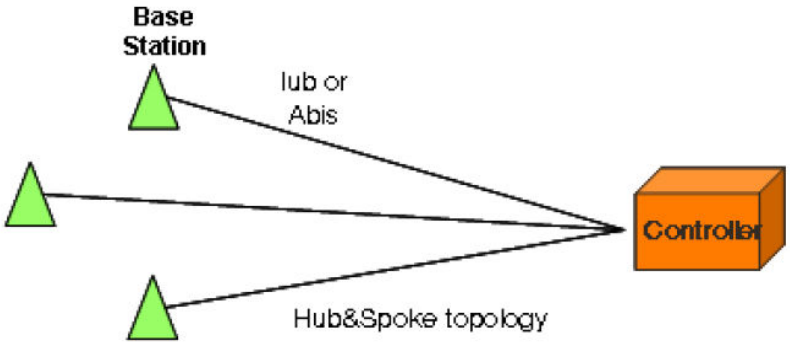

Figure 1.2 - 2G/3G RAN Topology [2]

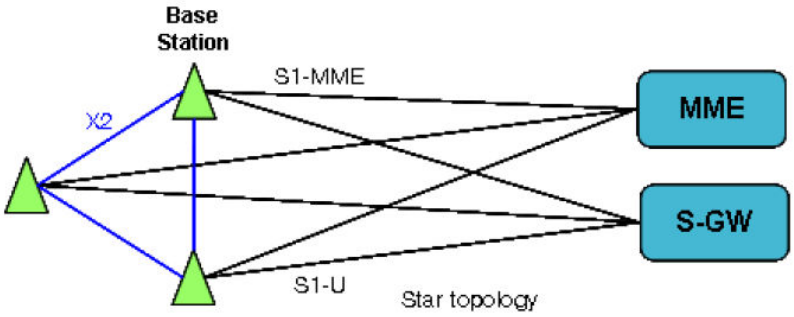

Figure 1.3 - LTE RAN Topology [2]

While Wideband Code Division Multiple Access (WCDMA) and High-Speed Packet Access (HSPA) have made significant drawbacks toward efficient mobile data and multimedia information exchange, LTE will provide extended network performance and reduced cost per byte that will allow it to deliver on the promise of mobile broadband.

Coexistence, interoperability, roaming, and handover between LTE and existing 2G/3G networks and services are inherent design goals, so that full mobility support can be confidently assumed. When introducing new network technologies, service providers expect that their existing investment will be protected and that deployed infrastructure can be reused to the greatest possible extent.

The main focus of the migration is thereby directed to topics representing a major part of a service provider's total cost of ownership:

• Deployment of LTE on existing sites and sharing of common infrastructure (such as antenna masts, site infrastructure such as power supply, air conditioning, feeder cables, and antennas)

• Sharing of backhaul equipment between LTE and other access network technologies provided at the same site. This means the same Cell Site Gateway (CSG) deployed to connect a 2G Base Transceiver Station (BTS) and 3G Node B to the packet-switched network could also be used to connect an LTE eNode B. CSG sharing to aggregate 2G/3G/LTE mobile traffic enables the use of the same fiber if only one fiber is available at the Node B site. The expected goal of

service providers is to backhaul 2G/3G/LTE mobile traffic, together with fixed traffic issued from any type of Digital Subscriber Line (xDSL) service, microwave, and Fiber To The Node (FTTx) access nodes, through a converged IP/MPLS core network for cost efficiency.

• Compatibility of transport solutions used for 2G, 3G, and LTE. For service providers it is important that the solutions used in the backhaul Internet Protocol (IP) Transport Network Layer (TNL) for 2G, 3G, and LTE be similar to use and unify operational tasks such as provisioning, monitoring, and OAM procedures. 2G and 3G networks are already migrating to IP transport, but with LTE it will be mandatory as the 3GPP has specified (3GPP spec TS 36.300) IP transport as the sole TNL for LTE. Therefore, other than a hybrid approach with TDM and packet transport in parallel, service providers will need to renovate the backhaul network and add packet transport capabilities, typically using pseudowires to emulate the legacy interfaces (TDM pseudowire for 2G, ATM pseudowire for 3G).

• Common network management platforms [2]

1.3 Current Status of the 16 bit Autonomous System Number Pool

The (now historical) 16-bit Autonomous System (AS) number space is a 16 bit field, with 65,536 unique values. From this pool 1023 numbers are reserved for local or private use, and 3 are reserved for special use. The remaining pool of 64510 numbers are available for use to support the Internet's public inter-domain routing system shown on Figure 1.4. IANA holds a pool of unallocated AS numbers, while the remainder have already been allocated by Internet Assigned Numbers Authority (IANA) for assignment.

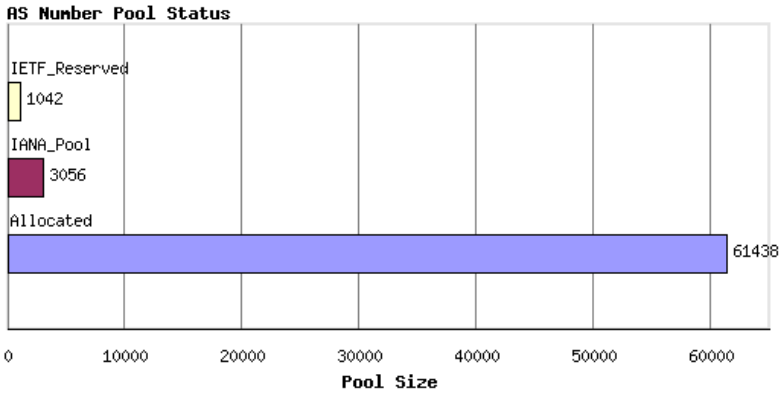

Figure 1.4 - AS Number Pool Status [1]

Any individual AS number can be in any one of the following states: part of the IANA unallocated number pool, part of the unassigned pool held by an Regional Internet Registry (RIR), assigned to an end user entity but not advertised in the routing system, and assigned and advertised in BGP. The current total of AS numbers according to this set of states is shown in Figure 1.5.

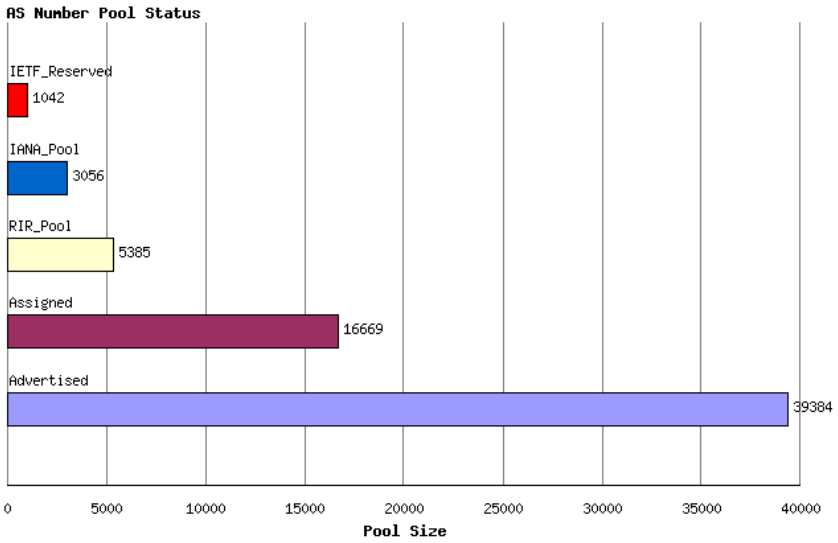

Figure 1.5 - AS Pools by State [1]

With that cautionary note about making assumptions about the future in mind, it is now possible to use this data to generate some predictive models of AS Number consumption. The used technique is to use the recent AS number consumption data in a time series, apply a best fit to the data using a least squares best fit, and then using the derived slope and intercept to generate a predictive trend line. This technique provides a linear best fit projection model. It may be the case that the underlying environment is driven by compound growth factors that create accelerating growth. The technique used here is to perform a least squares best fit on the logarithm of the data points, derive the best fit slope and intercept to generate a trend series, and then exponentiation of the trend data to create a best fit trend model that exhibits exponential growth.

The first data series analyzed in this fashion is that of the IANA allocation series. The past 900 days of IANA allocation activity have been analyzed using both linear and exponent rend analysis. [1]

Using this exponential trend model of advertised AS number growth and a linear trend model of the ratio of unadvertised to advertised AS numbers, it is now possible to model each RIR's projected AS allocation rate and then look at the predicted behavior of the RIR pool size for each RIR.

Using this model of each RIR's relative rate of allocation, it is possible to generate a model of AS number consumption. Here the end point is the date where the first RIR has exhausted its available pool of 2-byte AS numbers, and no further numbers are available in the IANA unallocated pool to replenish the RIR's pool. The data available suggests a best fit predictive model where this will occur in September 2014 shown in Figure 1.6. [1]

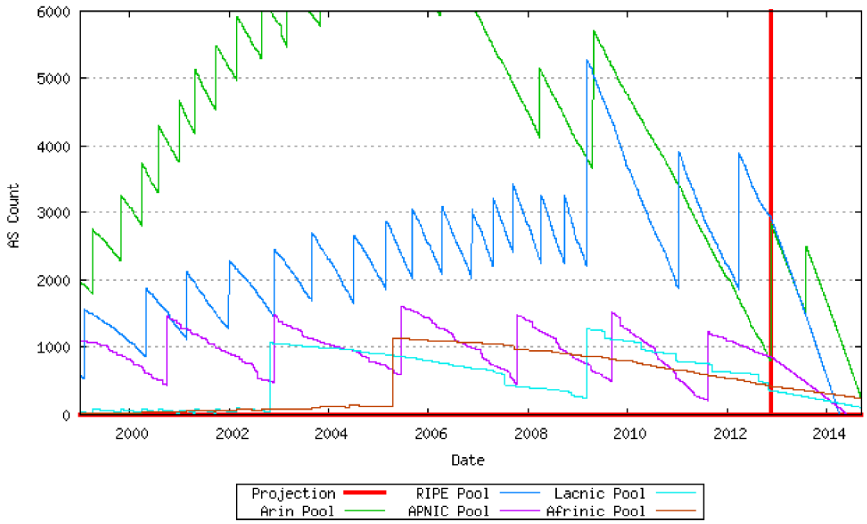

Figure 1.6 - RIR Pool size - projection [1]

1.4 Traffic Classification

Traffic management on the core networks must exist in order to properly prioritize different applications across the limited network bandwidth and ensure that these requirements are met. In addition, a proper understanding of the applications and protocols in the network traffic is essential for any network manager to implement appropriate security policies. In a real network, user perception also matters. Although a user application might allow large delays or jitter, the user might be very sensitive to long wait times. Managing network traffic thus requires a judicious balance of all these priorities.

Classification of traffic is only the first step that helps identify different applications and protocols that exist in a network. Various actions, such as monitoring, discovery, control, and optimization can then be performed on the identified traffic with the end goal of improving the network performance. Typically, once the packets are classified (identified) as belonging to a particular application or protocol, they are marked or flagged. These markings or flags help the router determine appropriate service policies to be applied for those flows.

In other words:

- Classification is a technique that identifies the application or protocol, and
- Marking is the process that colors the packets (or just lets them through untouched) based on certain classification policies, which are used by the routers internally, or further downstream (depending on the kind of coloring) to provide appropriate treatment to those packets.

1.5 Self-Organized networks

Implementation of Self-Organized Networks is wide spread the last couple of years. The self-organizing network is a network in which the tasks of configuration, operation, and optimization are automated. This type of architecture is used to reduce operational expenses and in mean time satisfy the user in congested traffic. Network nodes bill for a great part of networks' deployment, and maintenance costs. This is why so much efforts are put in the development of SON architectures.

In a great help for the design of SON are algorithms used in the creation of neural networks. This is caused from the nonlinear behavior of such architectures. The neural networks operation can be used as a basis for developing SON. Algorithms which are helpful for SON design include adaptive operation, feed-forward networks, multi-layer operation.

Feed-forward mechanisms using backpropagation algorithms can be used for "training" the network itself. In case of network failure the network can reorganize with minimum impact on the services, and at the same time start self-healing process if the problem is in the network software.

Learning in SON can be based on nonlinear optimization methods. A SON network can be regarded as a multi-layer network which is linear, and based on collected data can fix problems in the network and put it back to its previous condition. This is made by repairing the problem elements, based on parameters taken from the former state. fixing all RBF centers and nonlinearities in the hidden layer. One method which can be helpful is the Radial Basis Function, because of its multidimensional interpolation techniques. It can work with chosen typical nonlinearities and defined subsets of network parameters. The training of SON network can be realized applying the RBF algorithms [8].

2. Problem definition

- The network ha to be reliable. To have fast re-convergence. To support great number of devices. To allow addition of new devices with least number of new equipment and minimal configuration changes. So scalable and redundant network architecture has to be designed.
- The count of AS rapidly seizes (expected to be exhausted in September 2014). The use of AS number has to be limited, to delay the exhaustion as much as possible. Avoidance of Autonomous systems pool overuse has to be achieved.
- The network has to be able to operate with many kinds on technologies, to be independent from the access part of the architecture. The network has to be able to cooperate with most of the upper layer protocols. The designed network architecture has to be multipurpose, media and technology independent.
- The network has to serve multiple services and types of traffic. The real-time traffic needs to be delivered with minimal delay and packet loss. The service traffic flow has to be uncorrupted during all maintained sessions. Best-effort traffic must be delivered without data loss. This leads to the need of carefully planned and optimized Quality of Service in the Next Generation Network (NGN).

3. Goals

- Optimized network architecture – The network architecture must have less devices and simplified configuration, so the traffic running in it can be processed faster.
- Single Autonomous systems to be used to avoid AS pool exhaustion – since the AS pool has almost came to its end, using as least count of ASs as possible is a must.
- Implementation of multi-area IGP – separate area for each network region – using this approach the inter-protocol operability and inter-domain operability are avoided. Cease the time to process the traffic running through the network
- The network has to be flexible and independent from the media and the running technologies – The designed architecture has to be multipurpose. One converged network for all types of traffic (data, real-time, multimedia, gaming).
- Find the optimal parameters to classify traffic and implement the needed QoS in the designed next generation network architecture – Optimized classification to serve all kinds of traffic in the network. Depending on the specific case classification may be based on ports, packet size, VLAN number or a combination of them.
- Comparison between Plain IP and IP/MPLS network architectures – Two evaluation network models to be tested. Comparison of QoS and latency to be made for both network architectures. This way evaluating the benefits of using IP/MPLS NGN architectures in the LTE Backhaul.

4. Methods

The chosen technology for implementation of the NGN is Multi-Protocol Labeled Switching, because it provides media and technology independence in the network architecture (Figure 4.1). It works with wide variety of network models. In the following paragraphs all technologies, protocols and algorithms used for the design of the evaluation network architectures will be discussed.

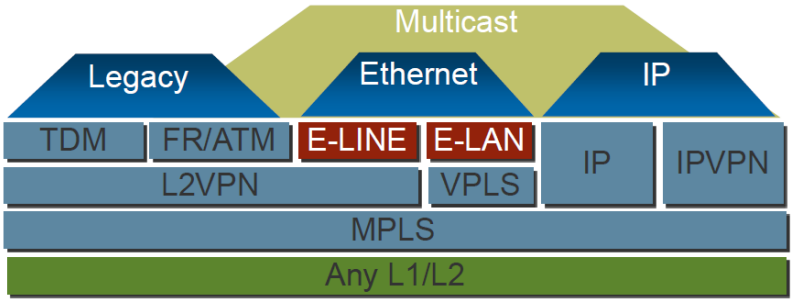

Figure 4.1 – MPLS operability [3]

4.1 Multi-Protocol Label Switching

MPLS works with IP routing protocols such as OSPF, BGP, IS-IS, which have been upgraded to work with it. MPLS enables traffic management from source to destination. Furthermore, packets can move independently, from the network algorithms of IGP protocols, using manually configured routes.

Paths established in a MPLS network are called Label Switched Path (LSP). Each MPLS enabled router, is called Label Switching Router (LSR). Redirection is usually based on the packet header containing the numerical value of the label.

Label Switched Path - It is created by the MPLS protocol. The path is unidirectional and is within a single autonomous system (Autonomous System, AS) or domain. The use of such paths allows maximum control of traffic. MPLS LSPs can also be fully realized with traffic engineering, such paths should be built bi-directional. Ways to establish MPLS LSP:

- **dinamically** - Dynamic LSP are established by the signaling protocol without intervention of the administrator. Only the edge router (ingress router) is configured with the information needed to establish paths. The rest of the network devices receivesignaling only messages during the establishment of the paths.
- **statically** - this way requires explicit LSP configuration. In this case, the administrator decides how to redirect the traffic, and what labels are used to identify resource allocation. Static LSPs consume fewer resources, do not require signaling protocol and don't need to store information about their condition. Problem for the use of such paths

occurs with lack of information about the network topology, it is also easy to make mistakes when configuring them. Moreover, a single device failure causes complete traffic loss.

Label Switching Router (LSR) - each IP router using MPLS can process the MPLS header and the values contained therein. LSR is responsible for forwarding the data packets through LSPs.

Types of LSR - input (ingress), intermediate (transit), penultimate (penultimate) and output (egress), shown in Figure 4.2

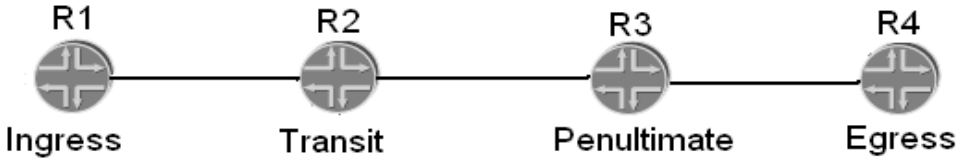

Figure 4.2 MPLS router types

Ingress LSR - is the entry point of the user data in MPLS-based topology.
In the LSR the client IPv4 packet is encapsulated in MPLS header via label push operation.Upon completion of this operation, packets are forwarded to the egress router through the corresponding LSPs. Each LSP requires ingress router and only one router can be ingress for one LSP.

Transit LSR (intermediary router) - each router in an LSP, which is located between the ingress LSR and the egress LSR. One path may contain from 0 to 253 such devices.
Operations performed by the transit LSR:

- When a packet is received the value of its label is read;
- The MPLS forwarding table is checked, for a corresponding output label to be found for the given input label;
- Label switching is used to replace the input label with the output label;
- The value of the Time-To-Live (TTL) field is decreased by "1";
- The packet is forwarded to the next router along the way.

During these operation the information in the IP header is never used [15].

Penultimate LSR - the last router before the LSP's egress LSR . Usually its task is to perform label pop operation - to remove the MPLS label from the data packet. This operation is called penultimate hop popping. It supports network scalability and reduces the load on the egress router [15].

Penultimate hop popping operation:

- When reading the label given node understands that it is the penultimate and which is the next router;
- Removes the stack of labels;
- Decreases the value of the TTL field by "1";
- Forwards the packet based on the label, which has been at the top of the stack [10].

Egress LSR (output router) - is the ultimate router for an LSP. It receives packets from the penultimate node and compares the IPv4 address, in the packet's header, with its routing table. Then redirects the packet to the next hop. Each LSP must have egress router and only one router can be egress for a given LSP [15].

4.1.1. MPLS label

Redirection of packets in MPLS backbone network is based on the labels that are set by the LSRs. The designation of the MPLS labels is done manually or automatically. Nodes exchange information about the compliance between the labels for the established LSPs.

When a packet is received the LSR replaces the initial label that the packet has with its corresponding one for this node. These labels are locally relevant for each link, which means that a given value may be used in several different connections.

Labels are coded in 32 bit shim header, which is set by the ingress router. MPLS adds its header between the IP header and the header of the data link layer (Layer2). The MPLS header format is shown in Figure 4.3.

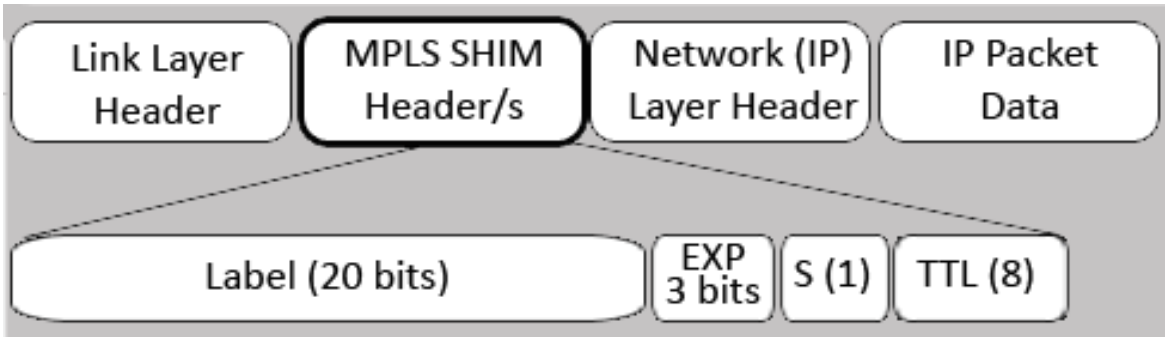

Figure 4.3 MPLS header

MPLS header fields

- **Label** - 20 bits. Contains the value of the label which determines to which LSP the packet belongs to. It perceives a value from 0 to 1,048,576 and has local significance;
- **Experimental bits** - 3 bits. Used for standardization experiments. It contains a Class of Service (CoS) field, but the CoSs are still not defined;
- **Stacking bit** - indicates whether the MPLS label is followed by an IP header or another MPLS label. A value of "1" indicates following IP packet, the value "0" - another MPLS label.
- **Time To Live** - copies the value of the IP header TTL field. Used to prevent traffic loops. Each router along the path decreases the TTL field value by "1". If the TTL reaches value of "0", the packet is dropped. When the last MPLS label is removed, its TTL value is copied to the IP header.

Reserved MPLS label values - from 0 to 15 are reserved by the Internet Engineering Task Force, and are common for all routers. [15]

Packet processing

The packet processing algorithm is shown on figure 4.4.

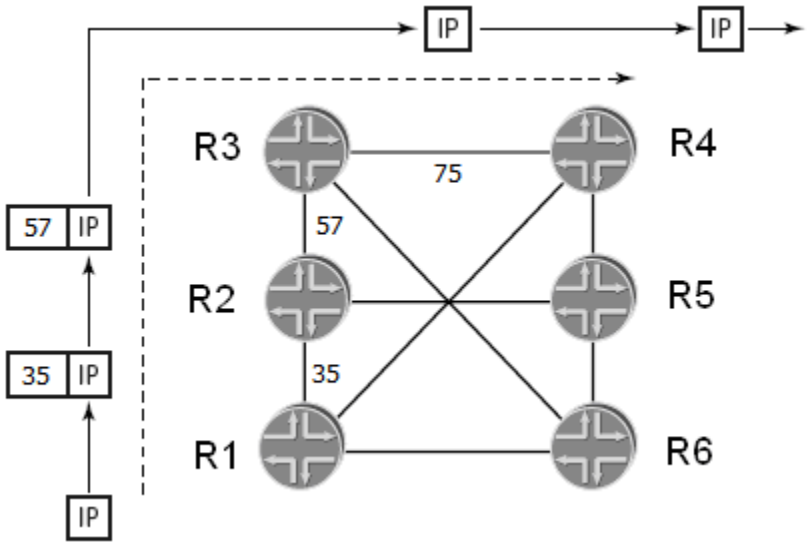

Figure 4.4 Packet processing

An LSP has been established from R1 to R4. All downstream routers have label values set for their upstream neighbors; R4 has assigned 75, R3 - 57, and R2 - 35. When an IP packet arrives at the ingress router - R1, the following process is started:

- R1 performs an IPv4 routing lookup for the destination IP address. It finds that the next hop for the route is an LSP to R4. MPLS header is added to the packet with label value of 35 and the packet is forwarded to R2.
- R2 receives the MPLS packet with a label of 35. It performs an MPLS forwarding table lookup and starts a swap operation. The label of 35 is removed and replaced with label of 57. The packet is forwarded towards next hop along the LSP.
- R3 receives the MPLS packet with label of 57. It performs an MPLS forwarding table lookup and starts pop operation, because its downstream peer advertised a value of 75. R3 removes the MPLS header from the packet and forwards the IPv4 packet to the next hop of the LSP.
- When R4 receives the IPv4 packet, it performs a routing table lookup and forwards the packet towards next hop of the route.

4.2. Resource reSerVation Protocol

Resource reSerVation Protocol (RSVP) is defined by recommendations RFC2205 [11] and RFC3209 [12] and is designed to serve the integrated Internet services. The protocol works with both IPv4, and with IPv6. Hosts use RSVP to reserve the required bandwidth for their

network services to assure the quality of service along the network. Routers use RSVP to ensure End-to End QoS in the network architecture. The protocol reserves nodes along the path to the receiver. These nodes establish and maintain the quality requirements.

4.2.1. RSVP basis

When a device requires a quality of service in the network, RSVP carries the resources request through the network.It passes through each device on the way to the receiver which the information can pass through. RSVP attempts to reserve the required resources on each node. The closer the resource request is to the receiving host, in different branches of the network, it starts to merge. Reservation request merges as it travels along the way to the receiver. (Figure 4.5)

Key advantage of RSVP is its scalability and the ability to be used in large multicast groups, but it is also able to do unicast resource reservation. RSVP does not have routing algorithms, it relies on other routing protocols when determining the paths to transfer the request to reserve. When change in the protocol routing database occurres, based on received routing updates, RSVP establishes new paths with reserved resources, based on these changes. [12]

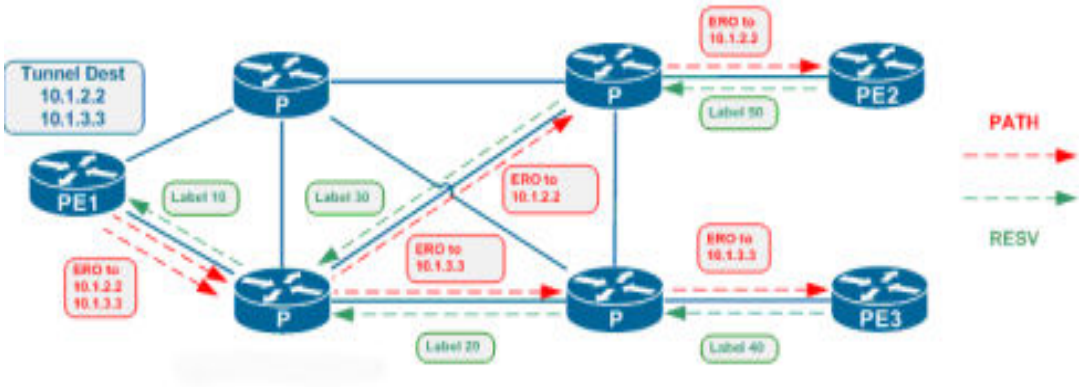

Figure 4.5 RSVP operation

Devices that support both RSVP and MPLS, can associate the MPLS labels with the RSVP flows. When LSP is established, traffic that flows through it is determined by the label set by the ingress router. Mapping between MPLS label and traffic flow can be done based on different criteria. Packets having the same label at a given node belong to the same Forwarding Equivalence Class (FEC) and define single "RSVP flow". When a traffic flow is associated with a LSP, this path becomes "LSP Tunnel". When a mapping between label and traffic flow exists, the router can determine via which resource reserved path to redirect the packet, based on the label. When a network architecture has signaling protocols enabled, it uses the downstream-on-demand label distribution. The ingress router initializes a request for mapping a particular MPLS label to given LSP tunnel. This is performed with RSVP Path message, containing LABEL_REQUEST object.Labels are determined by the downstream flow.

They are distributed as content in the Resv message with the upstream flow. For this purpose in the Resv message a special object LABEL is added.

A network with enabled signaling protocol also supports explicit routing capabilities. This is easily done by adding EXPLICIT_ROUTE object to the Path message. The object contains a series of hops, which define the explicitly routed path. By using this object, the packet switching path that RSVP-MPSL flow goes through can be predetermined, regardless of the IP routing protocols. Explicitly routed path can be created administratively or automatically, based on certain criteria for quality of service or compliance with particular policies (policies), regarding the state of the network.

Advantage of the use of RSVP is that the required resources along the network are pre-allocated for LSP tunnels on the way to the receiver. Resources allocation is not mandatory and LSPs can be established without special reservation requirements. Such paths can be used for best effort traffic or for fall-back and recovery (restoration) policies, under certain circumstances. [11]

4.2.2. RSVP message types

RSVP Common Header Format

RSVP messages have a common header followed by "body" of objects with variable length. RSVP messages fulfill a set rules, which define eligible the object types for a given message.

The order of the objects is determined by the Backus-Naur Form (BNF), although it is allowed the objects to be arranged in any order. The common format of the header is shown in Figure 4.6.

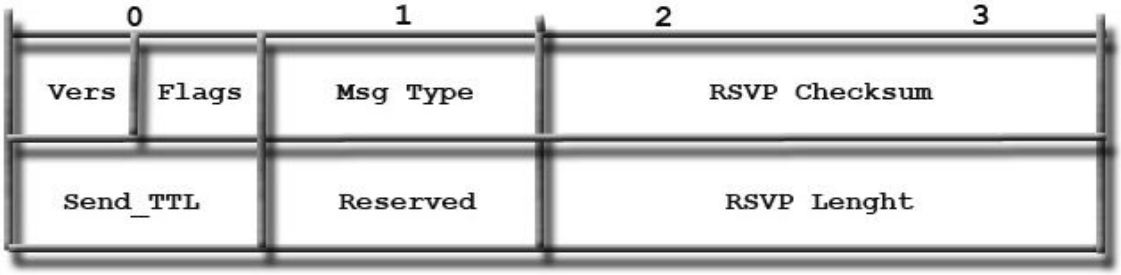

Figure 4.6 Common format of RSVP packet header

Common header fields

- **Vers** - 4 bits - contains the protocol version.
- **Flags** - 4 bits from 0x01 to 0x08 are assigned, and others have not yet been defined.
- **Msg Type** - 8 bits, assigned as follows: 1 = Path; 2 = Resv; 3 = PathErr; 4 = ResvErr; 5 = PathTear; 6 = ResvTear; 7 = ResvConf.
- **RSVP Checksum** - 16 bits, contains the checksum. If all bits are zero shows that the

checksum has not been used.

- **Send_TTL** - 8 bits. It copies the contents of the TTL field of the original IP packet.
- **Reserved** - 8 reserved bits.
- **RSVP Length** - 16 bits, presents the entire length of the RSVP message in bytes. Including the common header and a variable-length objects that follow [13].

Common Object Format

Each object consists of 32-bit header, followed by one or more 32-bit words. The format is shown in Figure 4.7.

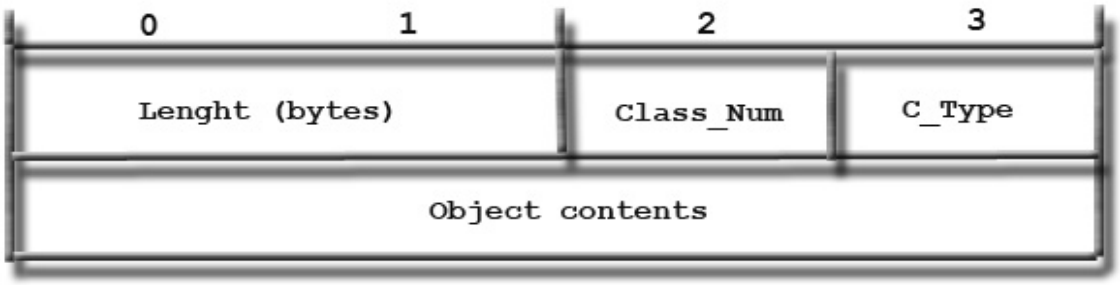

Figure 4.7 Common format of RSVP object

Object header fields

- **Field Length** - 16 bits, contains the length of the object in bytes. Must be at least 4 or a number multiple of 4.
- **Field Class-Num** - shows the object class with its name.

RSVP classes

- **NULL** - its content is only zeros, can occur in a sequence of objects and is ignored by the host part;
- **SESSION** - contains the IP address of the receiver and is a must for any RSVP message;
- **RSVP_HOP** - contains the IP address of the node from which it is received the message and the output logical interface;
- **TIME_VALUES** - contains the period for updating R, used by the originator of the message. Mandatory class for each RSVP message;
- **STYLE** - defines the style, the resource reservation is made and includes style-specific information;
- **FLOWSPEC** - defines the desired quality of services in the Resv message;
- **FILTER_SPEC** - defines a set of packets in a session, requiring the desired quality of services, in the Resv message;
- **SENDER_TEMPLATE** - contains the IP address of the sender and in some cases demultiplexing information, for simpler identification of the transmitting side. Mandatory

class for the Path message;
- **SENDER_TSPEC** - mandatory class for the Path message, defines the characteristics of the data stream sent from the transmitting side;
- **ADSPEC** - carries One Pass with Advertising (OPWA) data, contained in the Path message;
- **ERROR_SPEC** - defines the type of error in PathErr and ResvErr messages, or defines ResvConf confirmation message;
- **POLICY_DATA** - transmits information that allows local policy modules to determine whether the required resources are administratively enabled. May appear in Path, Resv, PathErr and ResvErr messages;
- **INTEGRITY** - carries cryptographic authentication information for the node which sent the message and checking the contents of the RSVP message;
- **SCOPE** - contains an explicit list of transmitting hosts that the message must be sent to. May be part of Resv, ResvErr and ResvTear messages;
- **RESV_CONFIRM** - carries the IP address of the destination host requesting the confirmation, may be part of ResvConf and Resv messages.
- **Field C-Type** - type of the object with a unique Class-Num.

The maximum length of the included objects, in RSVP message, is 65528 bytes. Fields Class-Num and C-Type can be combined to define a unique type for each object. [13]

Path message

Path messages are sent periodically by each transmitting side for each stream of data it sends. Message contains SENDER_TEMPLATE object that defines the format of data packets. Also contains SENDER_TSPEC object, which defines the required flow characteristics, for the transmission of information. The object ADSPEC carrying OPWA information about the data in the stream is optional.

Path message reaches the destination through the same path, which is used by data packets. IP address in the source address field must be the same as the address of transmitting site while the destination address must be the DestAddress of the particular session. These addresses esensure that the message will be correctly forwarded also outside RSVP part of the network [13]. The format of Path message is shown in Figure 4.8.

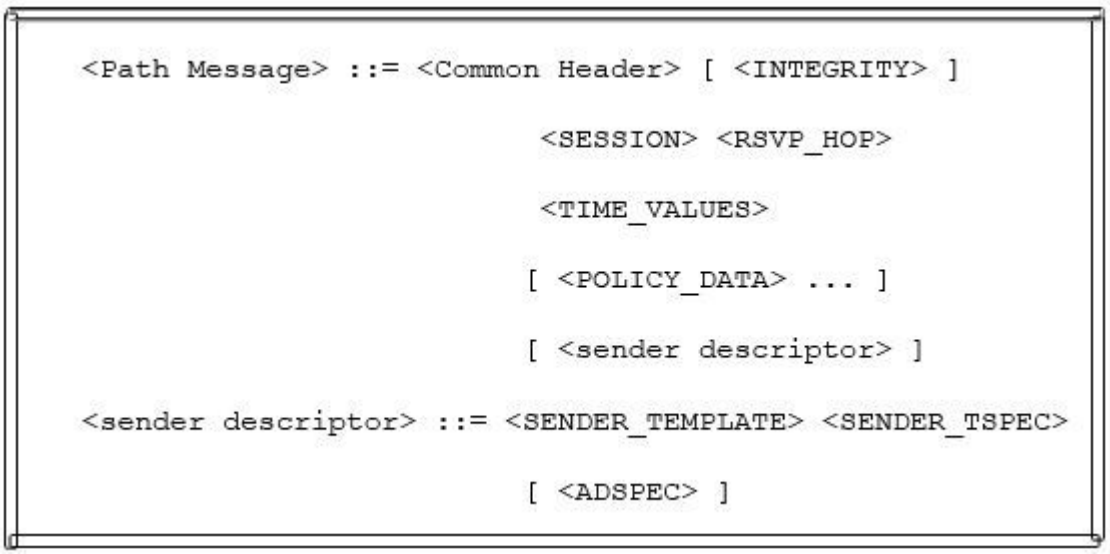

Figure 4.8 Path message format

If a Path message contains INTEGRITY object, the object should be placed immediately after the packet header. There are no requirements to the order of transmission, but the procedure shown in Figure 4.8 is recommended.

There is no limit to the number of POLICY_DATA objects which may be included in the packet.

PHOP object in the message contains information about the IP address of the interface from which the message has been received. This object also contains information about the behavior of the transmitting logical interface (logical interface handle, LIH).

Each RSVP node intercepts Path messages, processes them and forms Path State for the transmitting side. This is done based on the SESSION and SENDER_TEMPLATE objects. If there are POLICY_DATA, SENDER_TSPEC or ADSPEC objects, they also contained in the Pah state for transmitting host. If there are errors in the senders Path message, PathErr message is returned.

RSVP process periodically scans the given node for new Path messages, to be sent to the recipient. Each message contains a description of the transmitting side and carries the IP address of the original sender.

Routing depends on DestAddress field of the session, for protocols such as IP it also depends transmitter address. Usually, the routing information contains a list of zero or more output interfaces to which the Path message can be forwarded. Path messages sent by the different interfaces of the same router have different PHOP addresses, for each interface has its own IP address. ADSPEC objects also vary in messages sent through the different interfaces.

Path state for a given session and transmitting side not always has dedicated PHOP address and input interface. This occurres in two cases:

- **Multicast session** - multicast routing allows multicast distribution tree, wherein the Path message can be received by more than one PHOP address;
- **Unicast session** - for a short period of time when path change occurres, the node may receive a Path message, with a given pair sender - session, from several PHOP

interface. The node can not reliably determine which is the correct address PHOP, although it receives information from only one PHOP at a time. RSVP has to determine which of the PHOP addresses refer to the previous path. This situation is temporary, because the Path state will turn into "time out" or be terminated [13].

Resv message

Resv message carries requests for hop-by-hop resource reservation from the recipients to the transmitter along the data flow way back, for a given session. IP destination address is is the address of the previous node (unicast address in the Resv message) and is learned from the Path state. Sender IP address is the address of the transmitting node [13]. Figure 4.9 shows the Resv message format.

```
<Resv Message> ::= <Common Header> [ <INTEGRITY> ]
                        <SESSION>  <RSVP_HOP>
                        <TIME_VALUES>
                        [ <RESV_CONFIRM> ]  [ <SCOPE> ]
                        [ <POLICY_DATA> ... ]
                        <STYLE> <flow descriptor list>
<flow descriptor list> ::=  <empty> |
                  <flow descriptor list> <flow descriptor>
```

Figure 4.9 Resv message format

If Resv message contains INTEGRITY object, the object must appear right after the common packet header. STYLE object along with the flow descriptor list must be located at the end of the message, and the objects in the flow descriptor list must follow the format specified by the Backus-Naur Form. There are no requirements to the transmission order, but the procedure shown in Figure x.x is recommended.

The NHOP object in the message contains information about the IP address of the sender interface of the Resv message and also LIH interface information from/to resource reservation is required.

The appearance of the RESV_CONFIRM object in the Resv message signales for resource reservation request confirmation and contains the IP address of the recipient, that must receive the ResvConf information. There are restrictions for the number of POLICY_DATA objects that can be included in the packet. [13]

LSP tunnel related messages

The five messages related to LSP tunnels are given in Table 4.1.

Object name	Containing RSVP message
LABEL_REQUEST	Path
LABEL	Resv
EXPLICIT_ROUTE	Path
RECORD_ROUTE	Path, Resv
SESSION_ATTRIBUTE	Path

Table 4.1 LSP tunnels related messages

In addition to these messages new C-Type classes for objects SENDER, FILTER_SPEC and SENDER_TEMPLATE are created. These new types of objects are optional for RSVP and during the protocol implementation it is decided whether to use them or not. Although LABEL_REQUEST and LABEL are required according to reference RFC3209.

LABEL and RECORD_ROUTE objects are specific for each sender. In Resv message these objects are mandatory and have to be associated with the FILTER_SPEC object and also to precede each FILTER_SPEC object.

The recommended placement of the EXPLICIT_ROUTE, LABEL_REQUEST and SESSION_ATTRIBUTE objects, but is not mandatory. For this reason, implementations of RSVP must be ready to accept packets including the specified object types in random order [11].

4.3. Virtual Private Network (VPN) - L2VPN and L3VPN

4.3.1. MPLS Layer 3 VPNs

The design of MPLS-based VPNs is described in Recommendation 2547bis (draft-ietf-ppvpn-rfc2547bis-01.txt) and is considered as BGP/MPLS VPNs.

Data transmission is realized by reading the destination IP address in the header of the packet sent from the transmitting side. The IP address of the receiver is read and the packet is forwarded to a given LSP in the provider network, based on the routing table . To achieve this, the service provider must have access to information about the customer networks. Provider Edge (PE) routers exchange information with the Customer Edge (CE) routers. BGP/MPLS VPNs are similar to the peer-to-peer network model. Paths for a given VPN are transmitted to the other PE routers working with a given CE router. These paths are not

shared with the Provider core (P) routers and forwarded from one PE to another PE through the LSP. P routers do not need information about the customer networks to perform their function for packet switching. When a given PE router learns about new paths to a given VPN from another PE router, it transmits the information to the CE router operating within the given VPN, so the CE router is also informed about the remote parts of the network.

Mechanisms behind BGP / MPLS VPNs, are developed to overcome some drawbacks of standard L3VPNs and these mechanisms have the following goals:

- Maintenance of unique public IP addresses as well as support of private IP address spaces, including overlapping address spaces;
- Maintenance of overlapping VPNs, which may belong to more than one virtual network.

Since this type of virtual private networks relies on routing, these goals can be challenging. Due to the overlapping customer address spaces PE routers maintain a number of routing and forwarding tables, also known as a VPN Routing and Forwarding (VRF) tables, to separate the traffic belonging to different VPNs.

For the different client networks connected to a PE, separate VRF tables are maintained. When several branches of the same VPN are connected to one PE, they use the same VRF. If one site is connected to multiple VPNs, it does not share its VRF table with other sites, unless they are members of the exact same VPNs. The VRF table contains paths for all VPNs the site is connected to.

Another complication of overlapping address spaces, is that the PE receiving updates from its BGP neighbors, may receive conflicting or overlapping paths belonging to different VPNs. To avoid this, the "select the best path and ignore the rest" approach is applied to each prefix. This is achieved by adding eight byte Route Distinguisher (RD) field. The RD field is used to distinguish the paths to the different VPNs, for devices running BGP. As a result of the addition of RD field to the four octets of the IPv4 address 12 byte address is obtained, for which a new VPN-IPv4 family of addresses is created and for their distribution multi-protocol BGP is used.

Route Distinguishers are used to differentiate the separate paths and have no affect to the routes distribution management. RD is associated with a VRF, so this RD field is added to the prefixes in the given VRF. Usually the VRF, containing records of the sides of the same VPN, adds one and the same RD for all roads. RDs are unique for each VPN network, but that does not mean that sides which are related to multiple VPNs have several Route Distinguishers. VRFs, containing records of such networks have one RD.

To avoid receiving routes for foreign VPNs, PE is configured with BGP extended communities, that control the distribution of routes between nodes. Route Target (RT) attribute (of the extended communities) is distributed along with the roads for a given VPN, in order to show which sites the given route belongs to. RTs are unique for each client network and PE keeps track of them, along with the information stored for virtual private networks. When a packet is received, BGP verifies whether a given RT belongs to the networks that it serves - if it does, the packet is processed, if not - it is ignored. This is done to avoid keeping information for all VPNs to all PEs. Figure 4.10 shows topology of a BGP/MPLS L3VPN network [14].

When distributing VPN-IPv4 route, PE adds to the BGP message also an MPLS label message for this route, and in the NEXT_HOP field puts its your own address. As the Service Provider network is MPLS-based, each PE can access the other PEs via LSP. LSP can be RSVP based or can use the LDP protocol.

When PE receives a packet for remote side, it adds to it two MPLS labels in order to be forwarded to the recipient. The first added label indicates the LSP and contains BGP NEXT_HOP field. The internal label is used in the remote network. Its address comes from the BGP updates, received from neighboring routers. On the next step PE sends the frame to the interface associated with the corresponding LSP. Along the way to the receiving site label switching is performed to the packets, eliminating the outer label and replacing it with a new one, based on the internal label. The internal label identifies uniquely the destination and has been removed one step before the packet reaches the recipient. If route summarization occurs, PE reads the internal label and VRF table entries to be able to redirect the packet to the receiver correctly.

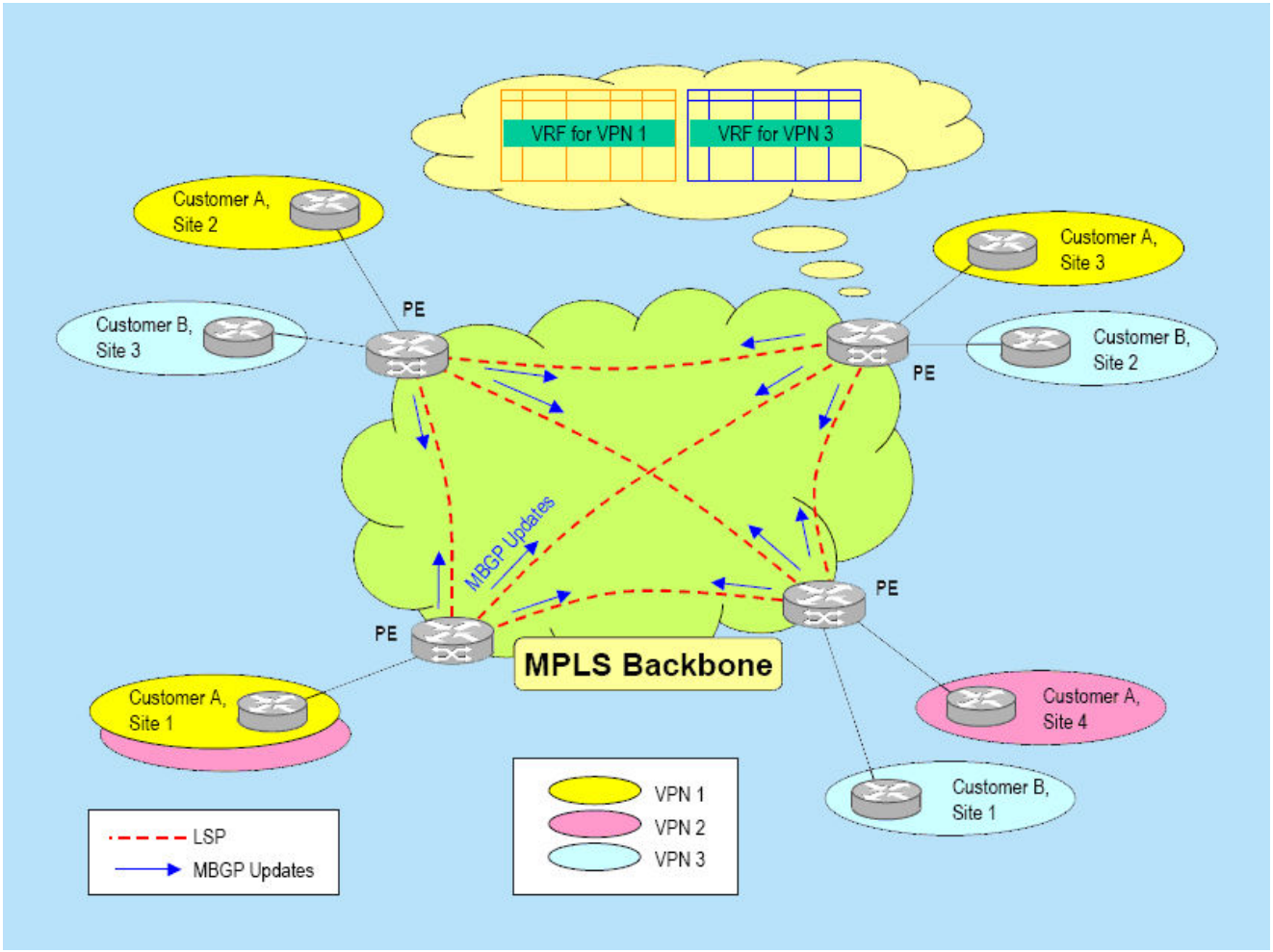

Figure 4.10 BGP/MPLS L3VPN network architecture [14]

Figure 4.11 shows the specific encapsulation recommendation according to RFC2547 [14].

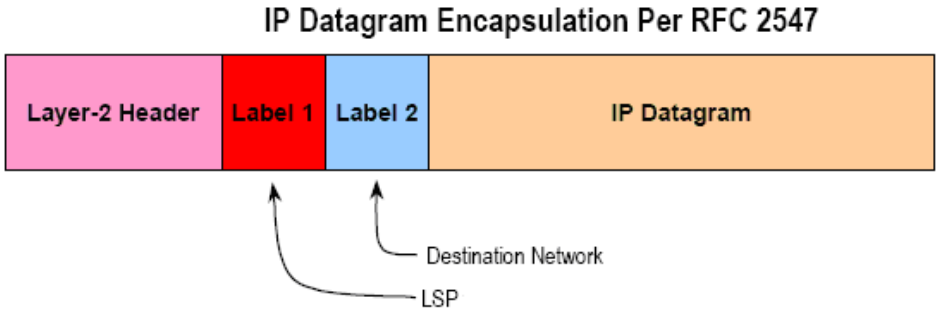

Figure 4.11 RFC2547 referred eccapsulation [15]

4.3.2. MPLS Layer 2 VPNs

Design of MPLS-based VPNs on the Layer 2 offers a switching based solution of the problem. In this approach there is complete separation between the customer network and the service provider network and no exchange of paths between the CE and PE routers is done. L2VPN model is based on the layered approach for VPN design.

Separation between the client part of the network and the service provider part of the network simplifies the implementation of this approach. MPLS L2VPNs offer services to transport the layer-2 frames from one client site to another. This approach is completely transparent to the CE devices. Working with layer-2 frames allows the ISP to provide services that are independent from layer-3 protocols used. In this case, the provider can carry along its network IPv4, IPv6, IPX, DECNet, OSI and other L3 protocols. There are two approaches to build MPLS-based L2VPNs [14]:

- provisioning of Point-to-Point connectivity;
- provisioning of Point-to-Multipoint connectivity.

Point-to-Point Connectivity

Point-to-Point connectivity is described in the Martini drafts [3] "draft-martini-l2circuit-trans-mpls-08.txt" and "draft-martini-l2circuit-encap-mpls-04.txt".

Transfer of layer-2 frames over MPLS backbone network is using the concept of Virtual Circuits (VC). The VC is a LSP, which works within the real LSP tunnel. LSPs act as tunnels that carry several VCs, and these virtual circuits carry user layer-2 frames.

LSP tunnel provides tunnels between PEs, while VCs only carry the client information. VCs are unidirectional - for two-way communication a virtual circuit for each direction is used.

For establishing a hierarchical structure of the network, the user frame is encapsulated with two labels (Figure 4.12) [14]:

- Tunnel label - refers to the tunnel LSP, leading to the final PE;

- Label of the virtual circuit (VC label) - contains information about the virtual circuit that passes the client information to the dedicated final PE.

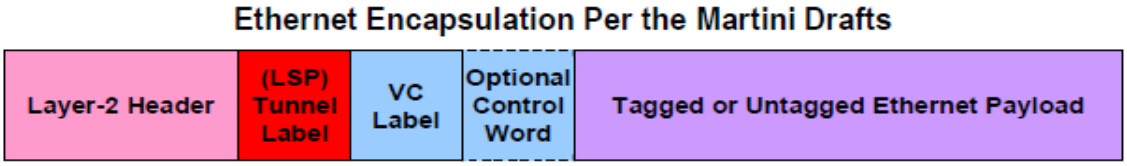

Figure 4.12 L2VPN frame[15]

Tunnels between PE routers can be build based on protocols like RSVP-TE, LDP, and others. The nodes exchange information about the virtual circuits via these protocols. When a packet is sent to the service provider network a VC and tunnel label are added. Then the packet is sent through the LSP tunnel. On the other side of the network the PE router removes the tunnel label and based on the VC label decides to which interface, associated with a customer network, to forward it. Then retrieves the original Layer2 frame and sends it through the selected interface (Figure 4.13) [14].

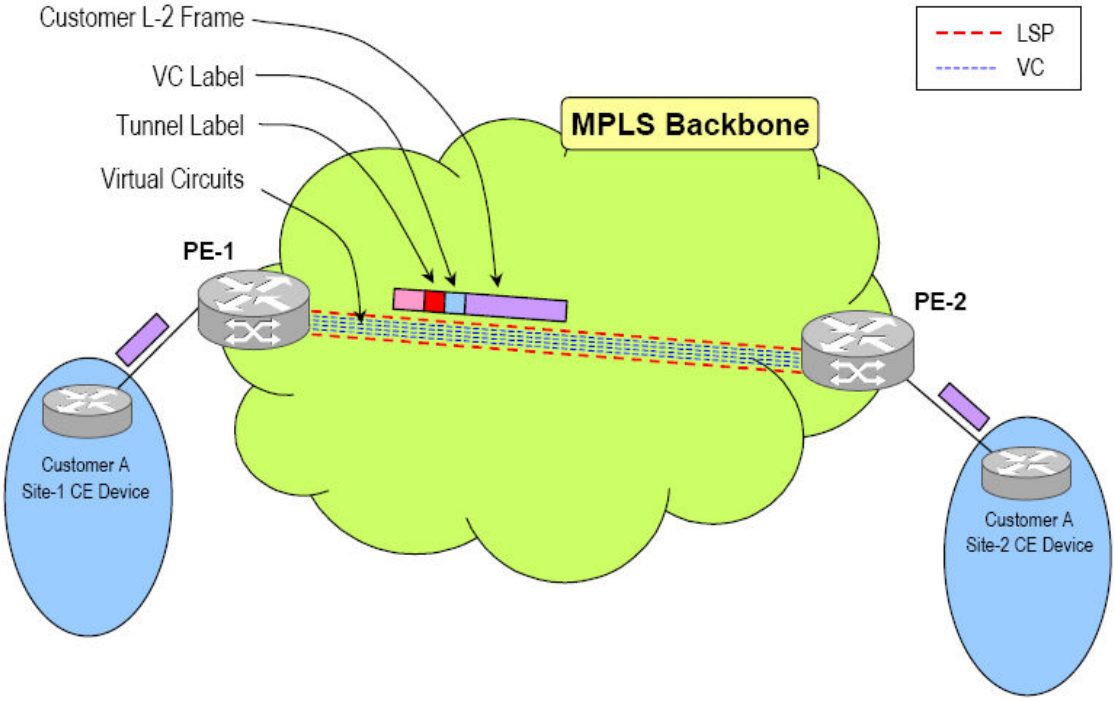

Figure 4.13 MPLS L2VPN network architecture [14]

4.3.3. Multi-Point Connectivity

To build a network topology that provides connectivity to multiple sites, participating in a IP/MPLS VPN based network the service Virtual Private LAN Services (VPLS) is used.

VPLS builds VPNs using full mesh topology (Figure 4.14) of virtual circuits between the

PE nodes, that build a virtual private network. VPLS relies on LDP protocol to exchange VC labels between PEs. Client VPNs have their own unique identifiers - VPN ID. PE routers learn the source (client) MAC address working as a switch, but the difference is that they do it for frames received via virtual circuits. PE maintain Layer2 forwarding table called Virtual Forwarding Instance (VFI), for each VPN, they are connected to. In such a topology Provider (P) routers do not learn MAC addresses, they only maintain switching labels.

Unlike normal Layer2 (L2) switches, routers do not use Spanning-Tree Protocol (STP) for fault tolerance and avoidance of L2 loops in the network provider. For this purpose, VPLS relies on MPLS protection mechanisms to ensure the necessary traffic fault tolerance. VPLS PE routers use the "split horizon" rule for forwarding, when forwarding the client VPNs frames. This is work-around for avoiding L2 frames loops without using STP. This solves the problem of network scalability, that emerges when using normal Layer2 STP topology.

Managing the paths between separate VPNs is responsibility of the client part, because PE routers do not maintain client route records in their forwarding tables.

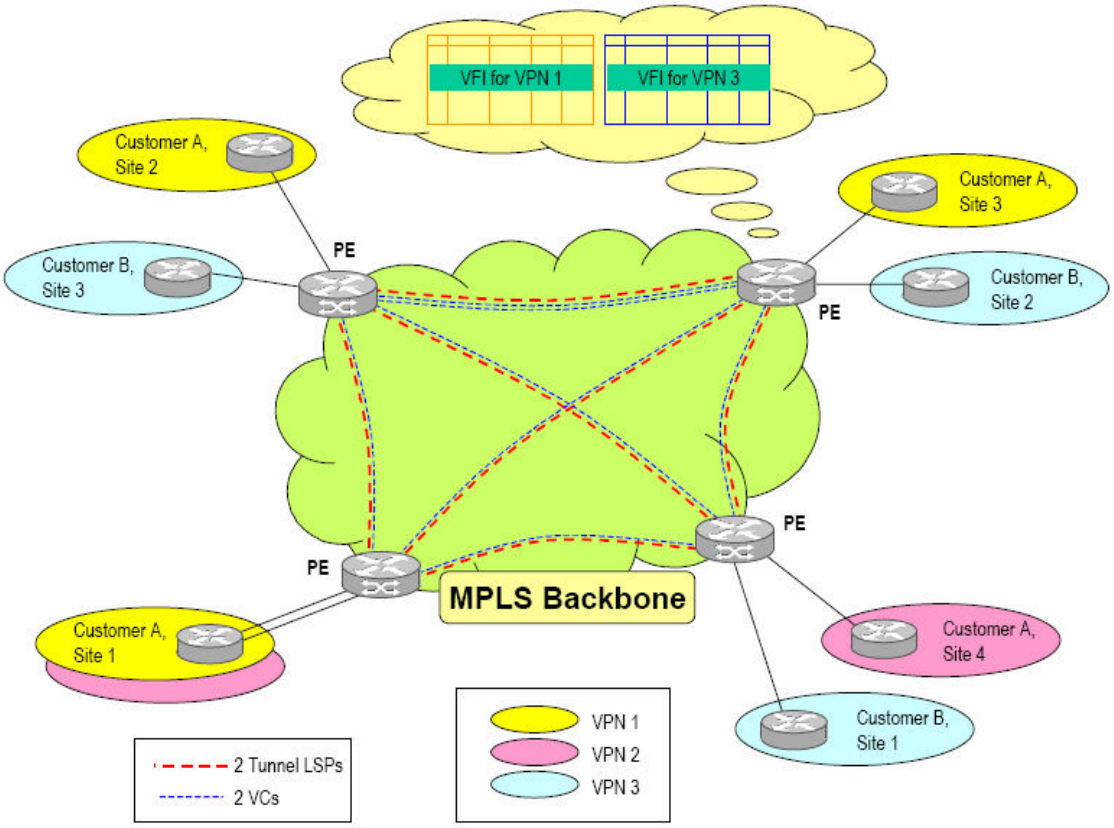

Figure 4.14 MPLS VPLS network architecture [14]

4.4. Dynamic routing protocol Border Gateway Protocol (BGP)

Border Gateway Protocol (BGP) is specified in recommendation RFC1771 [17].The protocol is used for communication and exchange of routing information between neighboring devices (peers).

Relations between routers is determined based on whether they belong to the same Autonomous System (AS). The relations management between the nodes and the sharing of information about routes is based on packet exchange. The devices make routing decisions relying on collected data, which contains the attributes of various routes, and based on local routing policies. Afterwards they share their routing information with their neighbors [15].

4.4.1. BGP Peers

BGP exchanges information between two routers called neighbors (peers) via TCP session. The session is being established between adjacent devices or through a link, which including intermediate devices.

IP connectivity is used when devices are directly connected. When devices are not adjacent IGP must be used. There are two types of neighboring:

- When the routers are in different autonomous systems;
- When the routers are in the same autonomous system.

In both cases, IP connectivity is interpreted differently when establishing the TCP session between devices [15].

4.4.2. External BGP sessions

When two routers are in different autonomous systems the session between them is considered an External BGP session (EBGP), which by default is enabled on directly connected devices. This requirement is necessary because the TTL (Time-To-Live) field is set to "1", so that no intermediate devices can process and redirect BGP packets.

When the connection is established, BGP nodes exchange routing information. All routes learned from internal and external BGP sessions, with neighboring devices, are shared [15].

4.4.3. Internal BGP sessions

The connection between two routers belonging to the same autonomous system is called internal BGP session (iBGP). In iBGP there is no requirement for a direct connection

between the IBGP neighbors. To enable the connection between devices in one AS, the TTL field is set the value of "64".

iBGP neighbors rely on the running IGP in the AS. In order to build a TCP session between iBGP neighbors, the IP routing tables of the intermediary nodes are used. BGP session uses the loopback address of the router, in order to ensure reliability and flexibility. Thus, in case of network failure, the IBGP session remains operational.

When the session is established, BGP neighbors exchange routing information. Only routes learned from EBGP sessions are distributed. For networks running IBGP sessions the use full-mesh topology is recommended [15].

4.4.4. Relations establishment in BGP

BGP uses finite state automata algorithms when establishing connections between neighboring routers, passing through BGP the following conditions:

- **Idle** - initial condition, in which all requests for building a session are denied;
- **Connect** - a condition in which the TCP connections are built;
- **Active** - a state in which TCP sessions are initialized;
- **OpenSent** - after initializing a TCP session, the router is waiting for messages from their neighbors;
- **OpenConfirm** - in this state, after receiving a message from a neighboring node the BGP device sends a confirmation and adjacent devices make exchange of Keepalive messages for session maintenance;
- **Established** - condition of fully operational relation between neighbours [15].

4.4.5. BGP message types

All BGP messages have identical header with a fixed length of 19 bytes (Figure 4.15).

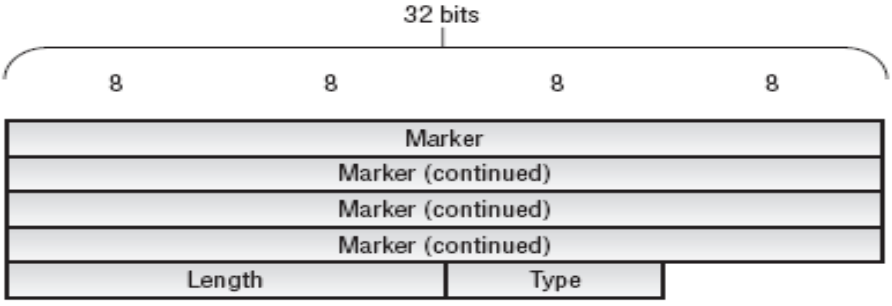

Figure 4.15 BGP protocol header

BGP header fields

- **Marker** - 16 bytes, all set to "1". Serves to detect synchronization loss;
- **Length** - 2 octets containing the full length value of the BGP message. This value can vary between 19 and 4096;
- **Type** - 8 bits, indicates the type of BGP message.

BGP messages

- **Open** - the first message that is sent to BGP neighboring device while establishing the connection. Serves for parameters negotiation of BGP sessions, which are: protocol version, session hold time, data authentication, session renewal and others.
- **Update** - used for sending and receiving routing information updates. It also may contain information for previously sent data that is no longer valid, as well as the kind of information sent to the remote peers. All messages for a certain session contain the same attributes and routes, having the attributes.
- **Notification** - it is sent to a neighboring node when session error is detected. Then the session is terminated immediately on both sides.
- **Keepalive** - only contains the 19 byte header of a BGP message and no other data. It is only sent if there is no exchange of other BGP messages. The hold timer is set to one third of the sessions hold time. [12]

4.5. Dynamic routing protocol - Open Shortest Path First

Open Shortest Path First (OSPF) protocol is defined in recommendation RFC2328 [16] and is an IGP protocol. OSPF uses link-state algorithm to calculate and build the shortest path to a destination. The algorithm works as follows:

- During the initialization or when routing information changes, link-state advertisements (LSA) are generated;
- All routers exchange link-states by "flooding" the OSPF domain;
- Each router that receives link-state information, keeps a copy of it in its link-state database, and forward it to its neighbours;
- When the routing database is updated, routers calculate the shortest paths to all destinations with the Djikstra algorithm.

When there are no changes in the OSPF topology, there is no additional OSPF traffic load in the network. All changes are learned from adjacent routers via link-state packets, and with Djikstra's algorithm shortest paths are recalculated [16].

4.5.1. Shortest path Algorithm

Shortest Path Algorithm is based on the Djikstra algorithm. Each router becomes the root of a "tree" and calculates the shortest paths to each known destination, based on cost of the path to it. Every router has its own view on the network topology [16].

4.5.2. OSPF Path Cost

Cost, also known as the metric of an OSPF interface, shows the overhead needed to send packets through it. The cost of an interface is inversely proportional to the bandwidth of this interface. Higher throughput means smaller cost. The formula for cost calculation is shown in figure 4.16 [14].

$$Cost = \frac{100\,000\,000\,bps}{Link\ Speed}$$

Figure 4.16 OSPF cost calculation

4.5.3. Areas and Area Border Routers

Areas are designed to prevent the enormous flooding with link-state updates. LSA "flooding" and the Djikstra's algorithm calculation are limited within the area. All routers in an area have the same link-state database. Routers belonging to a couple of areas are called area border routers and are assigned to distribute routing information or topology changes between areas (Figure 4.17). There are the following router types:

- - Internal Router (IR) - when its interfaces belong to a single area and AS;
- - Area Border Router (ABR) - when its interfaces are in different areas, but in the same AS .
- - Autonomous System Border Router (ASBR) - when its interfaces belong to different ASs [14].

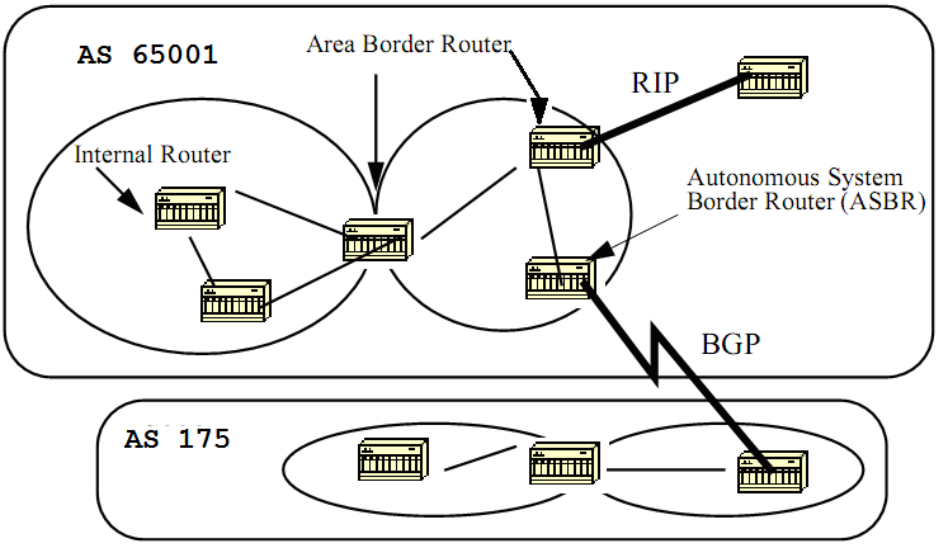

Figure 4.17 OSPF router types

4.5.4. Link-state packet types and OSPF header

The types of link-state packets are described in the following Table 4.2:

Type	Packet Name	Description
1	**Hello Packet**	Neighbor discovery and Connection with neighbors
2	**Database Description - DBD**	Database synchronization
3	**Link-state Request - LSR**	Link-state record request
4	**Link-state Update - LSU**	Link-state record reply
5	**Link-state Acknowledge- LSAck**	Link-state acknowledgement

Table 1.2 Link-state packet types

All OSPF packet types are stored directly in the IP payload. OSPF does not use transport layer protocols (TCP, UDP). Every OSPF packet starts with the same header format, shown in Figure 4.18:

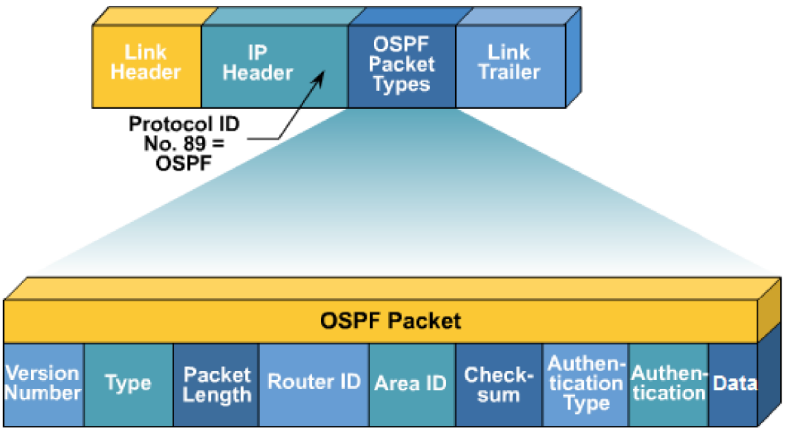

Figure 4.18 OSPF Header

4.5.6. OSPF header fields

- **Version Number** - 8 bits - shows if OSPF version is 2 or 3;
- **Type** - 8 bits - distinguishes 5 types of packets;
- **Packet Length** - 16 bits - length of the packet in bytes;
- **Router ID** - 32 bit - the packet source router;
- **Area ID** - 32 bits - indicates the area this packet was destined to;
- **Checksum** - 16 bits - used to detect errors in the OSPF header;
- **Authentication Type** - 16 bits - specifies the used type of authentication;
- **Data** - variable-length field [16].

4.6. Make-Before-Break rule

Interruption of traffic flow is highly undesirable, moreover when it comes to re-routing, this affects the operation of the network. To achieve smooth and adaptive re-routing it is necessary to have a pre-established LSPs. In case of connection failure the traffic can be transferred through these LSPs before disconnection. This concept is called "make-before-break" rule. Problem may occur if the primary and the secondary LSP tunnels "compete" for resources in a network segment that is common to both. Depending on the resources available, the establishment of the secondary traffic forwarding tunnel can be unsuccessful
For a smooth transition it is recommended, that the primary LSP resources should not be released before the secondary LSP is established and traffic can be transferred through the new tunnel [11].

4.7. Fast Reroute

Fast Reroute (FRR) is a way of protecting MPLS LSPs from link and node failures. This is done by locally repairing the LSPs at the point of failure, allowing data to continue flowing on them, while their headend routers try to establish new end-to-end LSPs to replace them. FRR

repairs the protected LSPs locally, by rerouting them over backup paths that bypass failed links or node [19].

4.7.1. Link Protection

Backup tunnels that evade only one link of the LSP's path provide link protection. By bypassing a failed link, they protect LSPs by rerouting the LSP's traffic to another next hop. These are known as next-hop (NHOP) backup tunnels because they end at the LSP's next hop beyond the point of failure [19].

4.7.2. Node Protection

FRR offers node protection for LSPs. Backup tunnels that evade next-hop nodes in LSP paths are called next-next-hop (NNHOP) backup tunnels because they terminate at the node, evading the next-hop node. They protect LSPs if a node along their path fails. This is provided by enabling the sending node to reroute the LSPs and their traffic around the failed node. FRR supports the use of RSVP Hellos to speed up the node failure detection. NNHOP backup tunnels also offer link failure protection [19].

If LSP is using a backup tunnel and changes occur, and the LSP is no longer suitable for the backup tunnel, the LSP is torn down. Such changes are as follows [19]:

- The backup bandwidth of the backup tunnel is decreased.
- The type of the backup bandwidth of the backup tunnel is changed to a type that is non-compliant with the primary LSP.
- Primary LSP modification causes disabling of the FRR.

4.7.3. Bandwidth Protection

Bandwidth protection for re-routed LSPs is provided by NHOP and NNHOP backup tunnels. This is also known as backup bandwidth. This gives information to the router about the amount of backup bandwidth a given backup tunnel can protect. Only if there is sufficient backup bandwidth an LSP could use given backup tunnel. This is done when a router maps LSPs to backup tunnels. The router chooses the backup tunnels the LSPs can use, in order to assure maximum bandwidth protection. This is done, so the router can determine the best way to map LSPs onto backup tunnels, this way gives maximum number of LSPs that can be protected [19].

4.8. Quality of service

There are numerous ways to implement QoS in the access part of the network.

Depending on the specific requirements, the quality of service in the network can be ensured in various ways [4]:

- Payload-Based classification - packets to be classified on TCP/UDP (User Datagram Protocol) ports basis in the payload fields;
- Packet size based classification – depending on the size of the packets exchanged during the session;
- Statistical classification - statistical analysis of the traffic parameters like inter-packet arrival, session time.
- Deep Packet Inspection - using Signature Analysis to recognize and check different applications, with the help of: 1. Pattern analysis - certain patterns in the payload of the packets can be used to identify protocols. 2. Numerical analysis - analyzing the numerical characteristics of packets (size, offset, etc.); 3. Heuristic and Behavioral analysis - looking closely at the way applications behave can help with the classification; 4. Protocol and State analysis - some protocols have specific request/response messages which can be easily followed and the traffic can be classified based of this communication.

4.9. Class of Service

CoS works by examining traffic at the network entry point. The edge nodes distribute traffic into defined service groups, to assure special treatment of this traffic along the network. When the traffic leaves the network at the far end, the traffic can be reclassified. To support CoS, it should be enabled on each router in the network. Every router inspects the packets that enter it to determine their CoS settings. These settings identify which packets must be transmitted first to the next node. Edge routers might be required to amend the CoS settings of the packets that enter the network.

Traffic Classifiers

Packet classification is referred to the inspection of an incoming packet. This function maps the packet with a particular CoS service level. Classifiers may map incoming packets with a forwarding class and loss priority. After that based on the dedicated forwarding class, assign packets to output queues. Several general types of classifiers are supported. They are explained in the following paragraphs [19].

Behavior aggregate or CoS value traffic classifiers

A behavior aggregate (BA) is an algorithm of classification that manages the packet as it enters the router. The CoS value in the packet header is inspected, and this one field defines the CoS settings applied to the packet. BA classifiers allow the forwarding class and loss priority of a packet to be set, based on the Differentiated Services code point (DSCP) value, DSCP IPv6 value, IP precedence value, MPLS EXP bits, and IEEE 802.1p value. The default classifier takes the value of the IP precedence [19].

4.9.1. Type of Service

The 'Type Of Service' Byte In The IP Header is define in RFC 791 [20]. ToS is one byte field defined in the IP header. This Byte is used to define the quality of service recommended for the datagram and is an incorporation of several factors. These factors contain several fields such as Precedence, Speed, Throughput and Reliability as defined below. The Type of Service byte typically is set to zero, but it comes in use when working with multimedia applications [20].

The Type of Service provides information of the required abstract quality of service parameters. They administer the selection of the actual service parameters when transferring a datagram through a given network. Networks propose service precedence, which processes high precedence traffic as traffic with higher priority, usually by acknowledge only traffic above a stated precedence when there is high load. The main choice is a compromise between low-delay, high-reliability, and high-throughput [20].

BITS							
7	6	5	4	3	2	1	0
PRECEDENCE			D	T	R	0	0

Figure 4.19 ToS IPv4 bits

The IP Type of Service Byte

Bits 0-2: Precedence.
Bit 3: Delay (0 = Normal Delay, 1 = Low Delay)
Bit 4: Throughput (0 = Normal Throughput, 1 = High Throughput)
Bit 5: Reliability (0 = Normal Reliability, 1 = High Reliability)
Bits 6-7: Reserved for Future Use.

The three bit Precedence field is outlined as follows

111 - Network Control
110 - Internetwork Control
101 - CRITIC/ECP
100 - Flash Override
011 - Flash
010 - Immediate
001 - Priority
000 - Routine

The use of the Delay, Throughput, and Reliability indications may rise the price of the service. In many networks improving of the performance for one of the mentioned parameters goes with worsened performance on another. Besides of very exceptional cases at most two of the three indicators should be defined. The ToS is used to define the datagram processing during its transfer through the network [20].

4.9.2. Differentiated Services Codepoint

IP packets contain a Type of Service (ToS) byte that includes a Differentiated Services Codepoint (DSCP) for differential services and IP Precedence bits used for QoS prioritization. The ToS byte is configured by an upstream device or application before the packet enters the switch [20].

Differentiated Services

Differentiated Services use the DSCP in an IP packet received from an upstream device to set the 802.1p priority to the packet. A new DSCP can also be set to the packet. It can be transferred (with the 802.1p priority) in the packet to downstream devices. In this case the DSCP of an input IPv4 packet can be read and,no change of the codepoint, set the 802.1p priority to the packet. Meaning that a priority value of 0-7 must be set for a DSCP before the device executes a QoS match on the packet's DSCP bits. Some codepoints default to the DSCP standard are set in the DSCP Policy Table but can be overwritten [20].

IP Precedence

All IP packets generated by upstream devices contain precedence bits in the ToS byte. The node can use these precedence bits to calculate and set an 802.1p priority. Precedence Bits consist of the upper three bits of the ToS byte. The same DSCP can be used in the IP packets of different applications. In case that an application operates on different devices, the same application could use multiple DSCPs. With using an edge device only certain packets can be selected. Afterwards mark them with predictable DSCPs. The DSCP can be used by downstream nodes to esteem policies set in the edge device [20].

- **Multifield traffic classifiers** - Multifield classifier is a second algorithm for traffic flows classification. Unlike the behavior aggregate, the multifield classifier can inspect multiple fields in the packet. Some fields that a multifield classifier can inspect are the source and destination address of the packet, and the source and destination port numbers of the packet.
- **Forwarding classes** - The forwarding classes have effect on the forwarding, scheduling, and marking policies applied to packets as they do through a router. Per-hop behavior is defined by the combination of the forwarding class and the loss priority. Categories of supported forwarding classes: best effort, assured forwarding, expedited forwarding, and network control.
- **Loss priorities** - They allow the packet dropping priority to be set. Loss priority has

effect on the scheduling of the packets without influencing the packet's comparative ordering. The packet loss priority (PLP) bit can be used as part of a congestion control scheme. The loss priority can be used to identify packets that were congested. Packets exceeding some service level are marked with a high loss priority. PLP is set by enabling a classifier or a policer.

- **Forwarding policy options** - The forwarding classes are mapped with next hops. Forwarding classes are appointed by forwarding policy, which contain classification overrides [19].

Transmission scheduling and rate control—These parameters provide a variety of instruments to control traffic flows [19]:

- **Queuing**—After a packet is sent to the egress interface on a node, it is queued for transmission on the physical media. The period a packet is queued on the device is determined by the availability of the egress physical media as well as the amount of traffic using the interface.
- **Schedulers**—Every router interface handles multiple queues set to store packets. The router defines which queue to process based on a given scheduling algorithm. This process often contains an identification which type of packet should be transmitted earlier.
- **Policers for traffic classes**—Policers have traffic limitation of a given class to a specified bandwidth and burst size as an option. Packets transcending the policer limits can be discarded, or can be put to another forwarding class, an another loss priority,or both. Policers can be define with filters that can be related with ingress or egress interfaces.
- **Rewrite rules**—A rewrite rule assigns the proper CoS bits in the outgoing packet. This enables the next downstream router to classify the packet into the proper service group. Rewriting, or marking, outbound packets is helpful when the router is at the network edge and must change the CoS values to face the policies of the targeted peer.

Default CoS Settings

If no CoS settings are configured on a router, the software executes some CoS functions to guarantee that user traffic and protocol packets are forwarded with minimal delay during the network congestion. Some default mappings are automatically set to each configured logical interface. Another default mappings, such as explicit default classifiers and rewrite rules, are in use only if being explicitly associated with an interface.

4.10. Used equipment for building the network architectures

4.10.1. MX960

MX Series consist of high-performance Ethernet routers with overall multiservice

support, and it operates as a true Universal Edge able to support business, mobile, and residential services. With its switching and security features, the MX Series delivers flexibility and reliability to maintain advanced services and applications. MX Series routers consist of separate control and forwarding plane. This way providing scalability and intelligent service delivery.

MX Series 3D Universal Edge Routers are optimized for Ethernet and are directed for a wide range of deployments and architectures, for both ISPs and enterprise networks.These routers are also ideal for high-performance data centers, mobile and campus networks.

Running Junos OS, the MX Series gives a consistent operating environment that simplifies network operations and improves the availability, performance, and security of services. MX Series features comprise traffic segmentation and virtualization with MPLS, low latency multicast, security and QoS implementations to improve delivery of time sensitive applications and services. The carrier-class reliability and high availability features on the MX Series consist of MPLS fast reroute, a comprehensive OAM toolkit, and service-level flexibility with features such as virtual private LAN service (VPLS) multihoming.

MX Series 3D Universal Edge Routers offer scale, bandwidth, services, and subscribers. MX Series routers are ideal for large applications that demand predictable performance for feature-rich infrastructures. In addition, they are ideal when Switch Control Board (SCB) and Routing Engine redundancy is a must.

With the MX Series, all main modules are field replaceable, that way improving system efficiency and reliability, and decreasing mean time to repair (MTTR).

Juniper Networks MX960 3D Universal Edge Router is a high density Layer 2 and Layer 3 Ethernet platform created for deployment in various enterprise and service provider
Ethernet architectures. For ISPs, Universal Edge applications supported by the MX960 consist of VPLS for multipoint connectivity, virtual leased line for point-to-point services, full support for MPLS VPNs, Ethernet aggregation at the campus/enterprise edge, and Ethernet aggregation at the multiservice edge. In the enterprise, the MX960 can be used for campus, mobile and data center core and aggregation, and also as a WAN gateway [21].

4.10.2. SRX210

SRX Series runs Junos OS with carrier-class routing features of IPv4/IPv6, OSPF, BGP, MPLS, VPLS, IPSec, etc, and multicast. SRX Series provides high level of security, application visibility, tracking and policy enforcement, access control, and network threat visibility and control. The SRX Series also includes features for firewall, IPsec VPN, NAT.

Policy-based VPNs support advanced security platforms that require dynamic addressing and split tunneling. For content security, SRX Series have a complete suite of Unified Threat Management (UTM) services.

SRX Series are secure routers that bring high performance and broad deployment capabilities to enterprises. The variety of options allow design of performance, functionality, and scalability. Ethernet, serial, T1/E1, DS3/E3, xDSL, Wi-Fi, and 3G/4G LTE wireless are the available options for WAN and Internet connectivity to link the sites in the network securely. Multiple form factors allow cost-effective choices for mission-critical implementations.

4.10.3. NTOOLS traffic generator

ntools presents a traffic generator, analyzer and network emulator package running on Linux system. It can generate UDP and TCP streams with configurable characteristics, and supply divers statistics. ntools satisfies the needs of network devices testing. The following items can be measured: performance, latency and Quality of Service. The network emulator tool can simulate several loss, delay and jitter impairments. The package contains the following modules:

NGEN AND NRECV - ngen can produce arbitrary number of static UDP/TCP streams. The manageable stream parameters include packet size, rate, ToS value, source and destination IP address and port. nrecv terminates the flows produced by ngen and gives the following stream statistics: transferred rate, packet loss, packet misordering, delay (latency), and jitter.

NGEN is the UDP/TCP generator module of ntools. It can produce traffic for number of various UDP/TCP streams. The module supports four packet generation profiles:

- **Fix**: fixed packet size and rate;
- **On-Off**: two states with different packet size, rate and duration;
- **Poisson**: fixed packet size, exponential inter-packet time;
- **TCP**: packet generation is controlled by the TCP protocol stack.

Frame sizes and rates are specified on layer 2 - containing the Ethernet overhead. The module sets sequence number and timestamp into the packets, so nrecv can compute loss and delay.

Only UDP flows can contain source IP, destination MAC address, VLAN parameters. The interface, protocol, destination IP and destination port values are required. For UDP flows if the source IP is not given, it is set to the address of the sending interface.

For TCP flows the module tries to send out the packets with the defined rate, but using a normal TCP socket, the sending rate relies on the state of the network, and may be much smaller than the configured value. For delay measurements the clock of the devices using ngen and nrecv must be synchronized [23].

NRECV is the UDP/TCP receiver module of ntools that gives statistics about static UDP and TCP flows produced by ngen. The module supports number of flows which can be defined by filters. The output of the NRECV generates the following fields for every stream:

- ID;
- number of received packets;
- number of lost and misordered packets;
- number of misordered packets;
- L2 (Ethernet-level) rate for UDP streams, and goodput for TCP streams, both in Mbps;
- delay in ms;
- jitter (min delay, max delay, and difference in ms);
- delay percentage in ms;
- flags for special events: F - when foreign frames, not generated by ngen, were received, C - when the measured delay is void, caused by clock sync problem.

The log file keeps one line for every captured packet. The line has the following fields for received packets:

- timestamp;
- stream ID;
- status: OK (normal frame), LOSS (frame loss), DUPLIC (frame duplication), MISORD (frame misorder);
- sequence number placed by ngen;
- number of lost frames before this one;
- Frame length (with Ethernet overhead);

the measured delay in ms (CLOCK indicates clock sync problem) [23].

4.10.4 JDSU testing module

The JDSU T-BERD/MTS-8000 40/100G Transport Module support full mobility. It is designed for carrier Ethernet/packet transport, long-haul, and metro-core networks, as well as government telecommunication centers. This module integrates both copper and fibre Small Form-factor Pluggable (SFP). It provides service activation, traffic testing, troubleshooting, and monitoring capabilities for synchronous Ethernet, SONET/SDH, and optical transport network (OTN) testing.

Layer 1 Test options - pause frame support and basic Layer 1 patterns, such as the IEEE 802.3ba scrambled idle pattern.The optimal level of field test support is provided at Layer 1 with injections and monitoring, including irregular measurements and numerous errors/alarms on a per-lane basis.

Ethernet VLAN, Q-in-Q, and MPLS Technologies - Ethernet tagging and encapsulation is often used to increase the scalability of Ethernet networks by isolating customer traffic. Separate from the encapsulation and tagging used, the device tests class of service to validate key performance indexes - committed information rate (CIR), frame delay (FD), frame delay variation (FDV), and frame loss ratio. Support for virtual local area network tags (VLAN tags), Q-in-Q VLAN tags, and multiprotocol label switching (MPLS) enables the module to test network architectures.

IPv4 and IPv6 Testing - Layer 3 test properties include traffic generation and analysis for both IPv4 and IPv6. Router connectivity is fulfilled through support of the ARP protocol to dynamically identify destination MAC addresses. The device supports ping and traceroute testing. Particular for IPv6, the neighbor discovery protocol ensures support for IPv6 address resolution.

CoS Verification with Multiple Streams - Multistream testing provides traffic generation and analysis at the Ethernet and IP layers so various types of traffic with CoS mappings can be propagated, evaluating the impact of traffic prioritization on the network architecture and to validate the suitable queuing, policing, and shaping. The per-stream parameters - VLAN ID and priority, type of service/differentiated services code point (TOS/DSCP) marking, packet size, stream bandwidth, and source/destination MAC and IP addresses, can be configured. Setting constant, burst, or ramp traffic enables near real-world traffic simulation. This device can measure round-trip latency (150 ns accuracy) [24].

5. Verification of the proposed network architecture approach

For the purpose of the present project a simplified network architecture is built (Figure 5.1). It covers the main steps in designing LTE backhaul network. The NGN network architecture is chosen according to the requirements for the design of networks with service provisioning and implemented End-to-End QoS. The proposed NGN network architecture can be easily scaled with simply appending new devices in the network.

The considered network consists of BTS stations, to which users connect to. BTSs connect to a switch or access router which is at the edge of the network. The BTSs operate with the end user devices. They multiplex numerous user traffic flows and carry them towards the backhaul network. A BTS is controlled by a BSC. BTSs are beyond the scope of the present research.

Access region, which connects to the "backbone" of the network, and establishes the link between the BTS the Core network. For optimization and simplification of the network the access and aggregation regions are consolidated in one Access/Aggregation area in the architecture. In it the QoS is applied to the traffic running through the network. Access/Aggregation region multiplexes numerous BTS and carries their traffic flows to the Core/Edge area via E1 lines.

Core/Edge is the high-speed region of the network. It makes the connection between the Access/Aggregation and Application servers areas. In the analyzed scenario the connections between Core/Edge devices are with 1Gbit/s speed. The edge part is the entry point of the network "backbone". It exchanges information and traffic flows with the Access/Aggregation area, as well as with Application servers area. The Core part consists of the primary nodes in the architecture. They have high capacity and carry large amounts of traffic.

In the proposed solution the Application servers are referred to the management of the user traffic. They control the functionality of the BTSs. Application servers are considered as BSC/RNC, Multi-media entities, Services gateways, Packet Data Network (PDN) gateways (PGW), Database servers, Multimedia servers, Gateway GPRS Support Nodes, Authentication, Authorization, and Accounting servers, etc.

Authentication, authorization, and accounting (AAA) referres to controlling access to network resources, policies enforcement, verification purposes, and providing the required billing information.

MME is the main control node in LTE networks. It is liable for retransmissions and idle mode of end users. MME is part bearer activation and deactivation,choosing SGW for a mobile user by initial attachment and intra-handover of Core Network (CN) node reposition.

SGW handles handovers between eNodeB's neighbors, data transfer, mobility interface networks like 2G/3G, monitoring and maintaining context information about users in their idle state and generates paging requests, replication of user traffic in case of link interruption.

The PGW forwards mobile user traffic towards PDN, works as main node of mobility between 3GPP and non-3GPP networks, provides link from mobile user to external PDN,

enforces policies, packet filtering, billing and connection loss.

Multimedia servers store and manage multimedia objects and deliver data streams in real-time, in response to requests from users .

The GGSN serves as the interface to GPRS networks, accessing external GPRS functions such connection to the 2G, 3G and WCDMA mobile networks.

Database servers handle the records for the network users, which contain data such as user identification, billing plan, etc. Application servers are beyond the scope of this project.

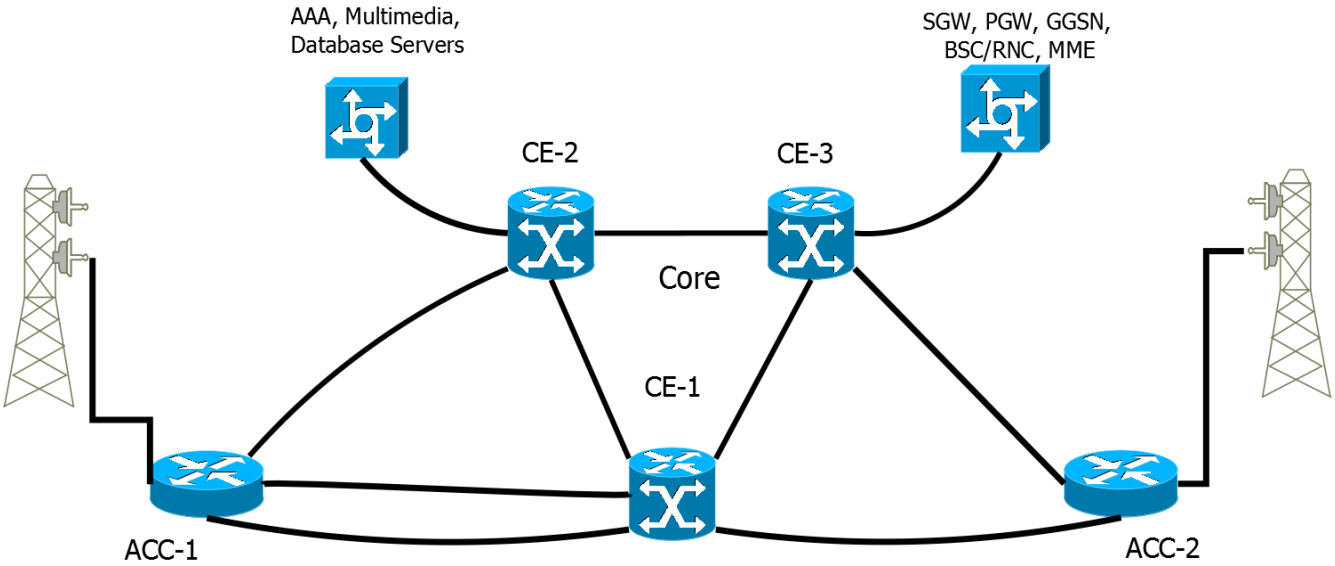

Figure 5.1 Simplified NGN network architecture with End-to-End QoS

The proposed network model is slightly modified for the purposes of the evaluation testing network (Figure 5.2). The BTSs and Application servers are changed with general purpose traffic generators. One of the traffic generators is Linux based software solution with advanced network testing capabilities- NTOOLS and its modules NGEN and NRECV. The other one is hardware based JDSU T-BERD/MTS-8000.

Two network architectures were evaluated. One is Plane IP based with single-area IGP OSPF. The second is IP/MPLS based with RSVP-TE signaling and tunneling. Both architectures use the same networking equipment. The links and interfaces are similar in both Plain IP and IP/MPLS models. The QoS applied to the traffic running through the network is similar in both solutions.

The Core/Edge region of the network is realized with Juniper Networks MX960 series router. The device is logically divided into three logical systems. These systems are acting like separate routers. They have full functionality capabilities of three separate hardware devices. The connections between the three logical systems are made with logical tunnel interfaces ("lt" interfaces). The lt interface speed is set to 1Gbit/s. The links with the other devices in the network are realized with general Gigabit Ethernet interfaces ("ge" interface). The speed of the ge interfaces, connected with the Access/Aggregation area, is set to 2Mbit/s, so emulation of E1 lines is achieved. The Core/Edge nodes connected with the end devices (Application servers) on Fast Ethernet speed of 100Mbit/s.

The Access/Aggregation area is made with two Juniper Networks SRX210 series secure

services routers. They are working in packet-switched mode instead of the default packet-flow, due to the IP/MPLS architecture. The devices are working with Fast Ethernet interfaces ("fe" interfaces). The links with the Core/Edge region are made on E1 link speed of 2Mbit/s. The links with the end devices are with full Fast Ethernet speed of 100Mbit/s. The Access/Aggregation routers apply the QoS to the traffic from the end devices.

The Application servers are exchanged with general purpose traffic generators. For testing the QoS applied on the traffic flow linux NTOOLS modules NGEN and NRECV are used. These modules provide functions as random traffic generation, fixed or non-fixed packet size, simultaneous generation of multiple traffic flows with heterogeneous parameters, definition of Layer 4 protocol used, possibility to define ToS on the generated traffic. The JDSU T-BERD/MTS-8000 is used for testing the latency across the realized evaluation NGN network architectures.

The uniform network entities for both Plain IP an IP/MPLS tested architectures are given in following description. These units are the IP address allocation, the interface connections and configuration, as well as the QoS applied along the network.

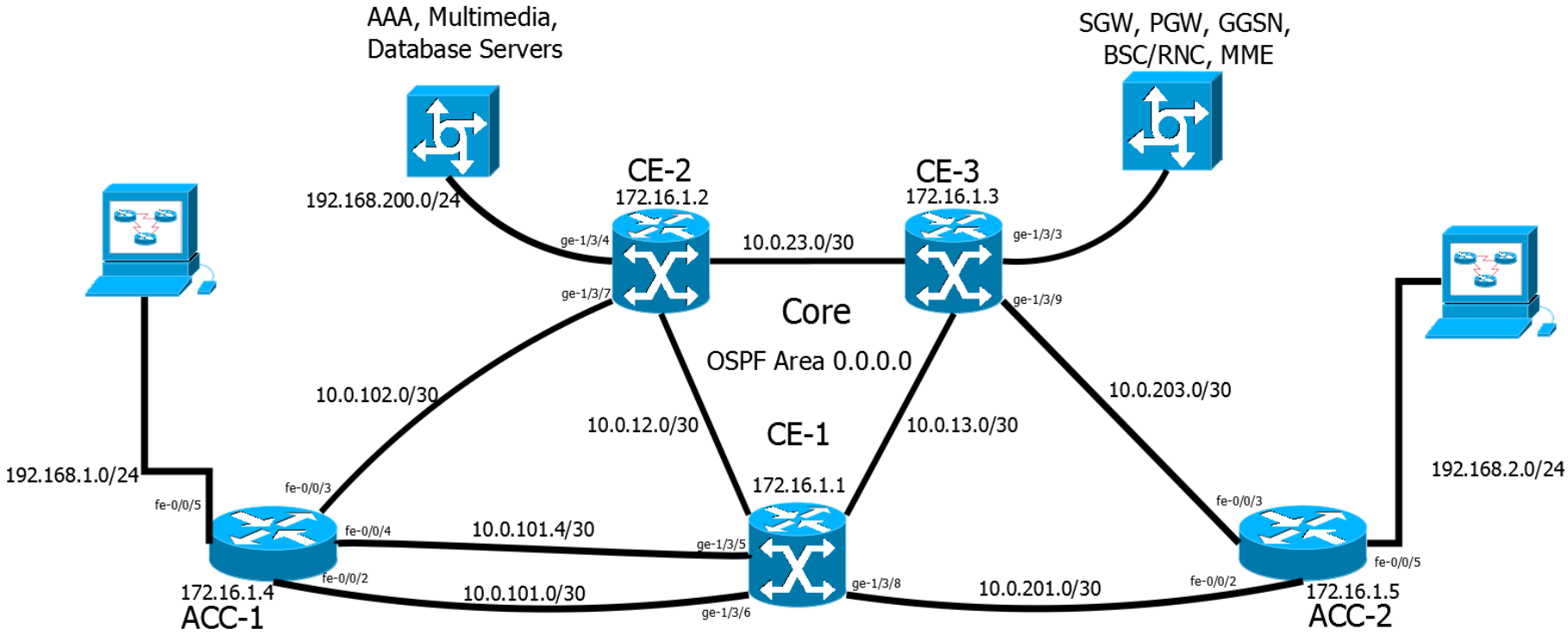

Figure 5.2 Evaluation model of NGN network with End-to-End QoS

5.1. Logical System Configuration

The MX960 router is divided in three logical routers (logical systems). These logical systems are working as three standalone nodes. To achieve this logical routers have tobe configured on the device. This is made by using the following command:

```
[edit]
root# set logical-systems logical-system-name
```

To configure any other features on the given logical system either the statement:

```
[edit]
root# set logical-systems logical-system-name [protocols; interfaces;
routing-options; etc]
```

is used or the configurations are made under [edit logical-systems logical-system-name] configuration hierarchy level. This level is entered via edit logical-systems logical-system-name command.

5.2. Network IP Address Spaces

The IP address spaces in the network are described in the following steps. Most networks are private from Internet's point of view, and they have just a couple of uplink points connected to Internet. These uplink points are dedicated interfaces, which have public IP addresses, and are assumed as external interfaces (external part of the network). This gives opportunity to use private IP address spaces in the internal interfaces in the network.

To simplify IP addressing scheme in the designed network different private IP address spaces are used.

The A-class **10.0.0.0/8** network is dedicated for the connections between Core/Edge devices, and connections between Core/Edge and Access devices. This address space is split into

The B-class **172.16.1.0/16** network is given to the Loopback interfaces of the devices. These interfaces have IP addresses with 32-bit subnet mask.

A couple of C-class address spaces starting with **192.168.1.0/24** are dedicated for the connections between Access and End devices in the architecture.

5.2.1. Access to Core/Edge links and Core/Edge network links IP addresses

Initial Network address – 10.0.0.0 /8
Subnet mask: 255.0.0.0

Starting Network address – 10.0.0.0/30
Number of host bits - 3 bits
Number of hosts - 2 hosts
Number of subnet bits - 22 bits
Number of subnets - 4194304 subnets
Subnet mask: - / 30
Subnet mask: - 255.255.255.252

Connection between CE-1 and CE2 - 10.0.12.0/30

Host Range: - 10.0.12.1 – 10.0.12.2
Broadcast address: - 10.0.12.3

Connection between CE-1 and CE3 - 10.0.13.0/30

Host Range: - 10.0.13.1 – 10.0.13.2
Broadcast address: - 10.0.13.3

Connection between CE-2 and CE3 - 10.0.23.0/30

Host Range: - 10.0.23.1 – 10.0.23.2
Broadcast address: - 10.0.23.3

Connection between CE-1 and ACC-1 - 10.0.101.0/30

Host Range: - 10.0.101.1 – 10.0.101.2
Broadcast address: - 10.0.101.3

Connection between CE-1 and ACC-1 - 10.0.101.4/30

Host Range: - 10.0.101.5 – 10.0.101.6
Broadcast address: - 10.0.101.7

Connection between CE-2 and ACC-1 - 10.0.102.0/30

Host Range: - 10.0.102.1 – 10.0.102.2
Broadcast address: - 10.0.102.3

Connection between CE-3 and ACC-2 - 10.0.203.0/30

Host Range: - 10.0.203.1 – 10.0.203.2
Broadcast address: - 10.0.203.3

Connection between CE-1 and ACC-2 - 10.0.201.0/30

Host Range: - 10.0.201.1 – 10.0.201.2
Broadcast address: - 10.0.201.3

5.2.2. Network Devices Loopback IP Addresses

Initial Network address – 172.16.1.0/16
Subnet mask: 255.255.0.0

Starting Network address	– 172.16.1.0/32
Number of host bits	- 0 bits
Number of hosts	- 1 hosts
Number of subnet bits	- 16 bits
Number of subnets	- 65536 subnets
Subnet mask:	- /32
Subnet mask:	- 255.255.255.255

CE-1 Loopback - 172.16.1.1/32

CE-2 Loopback - 172.16.1.2/32

CE-3 Loopback - 172.16.1.3/32

ACC-1 Loopback - 172.16.1.4/32

ACC-2 Loopback - 172.16.1.5/32

5.2.3. IP address spaces between Access devices and End devices in the network

The IP address spaces for the connections between end devices in the network and the access routers are chosen to be a C-class networks. This gives opportunity for scaling the network with up to 253 end devices connected to each access device.

Initial Network address – 192.168.1.0/24
Subnet mask: 255.255.255.0

Starting Network address	– 192.168.1.0/24
Number of host bits	- 8 bits
Number of hosts	- 254 hosts
Subnet mask:	- /24
Subnet mask:	- 255.255.255.0

Connection between ACC-1 and BSC-1 – 192.168.1.0/24

Host Range:	- 192.168.1.1 – 192.168.1.254
Broadcast address:	- 192.168.1.255

Connection between ACC-1 and BSC-2 – 192.168.2.0/24

Host Range: - 192.168.2.1 – 192.168.2.254
Broadcast address: - 192.168.2.255

Connection between ACC-1 and SRV-1 – 192.168.200.0/24

Host Range: - 192.168.200.1 – 192.168.200.254
Broadcast address: - 192.168.200.255

Connection between ACC-1 and SRV-2 – 192.168.300.0/24

Host Range: - 192.168.300.1 – 192.168.300.254
Broadcast address: - 192.168.300.255

The full IP addressing scheme is given in Table 5.1. The EN-INT in the table is referred to the interfaces of the end node devices.

Interface	Device									
	CE-1	CE-2	CE-3	ACC-1	ACC-2	BSC-1	BSC-2	BSC-3	SRV-1	SRV-2
Lo0	172.16.1.1/32	172.16.1.2/32	172.16.1.3/32	172.16.1.4/32	172.16.1.5/32					
lt-0/0/0.12	10.0.12.1/30									
lt-0/0/0.13	10.13.0.1/30									
lt-0/0/0.21		10.0.12.2/30								
lt-0/0/0.23		10.0.23.1/30								
lt-0/0/0.31			10.0.13.2/30							
lt-0/0/0.32			10.0.23.2/30							
ge-1/3/3			192.168.300.1/24							
ge-1/3/4		192.168.200.1/24								
ge-1/3/5	10.0.101.5/30									
ge-1/3/6	10.0.101.1/30									
ge-1/3/7		10.0.102.1/30								
ge-1/3/8	10.0.201.1/30									
ge-1/3/9			10.0.203.1/30							
fe-0/0/2				10.0.101.2/30	10.0.201.2/30					
fe-0/0/3				10.0.102.2/30	10.0.203.2/30					
fe-0/0/4				10.0.101.6/30	192.168.2.1/30					
fe-0/0/5				192.168.1.1/24						
EN-INT						192.168.1.2/24	192.168.1.3/24	192.168.2.2/24	192.168.200.2/24	192.168.300.2/24

Table 5.1 IP addressing scheme of the designed network architectures

5.3. Interfaces in the designed evaluation network architecture

The interfaces in the network architecture are established with the following steps. The configuration is almost common for all interfaces. Only loopback and lt interfaces have distinct differences in the way of their configuration.

The loopback interfaces can't contain VLAN configuration.

The logical tunnel interfaces must contain peer unit in their configuration in order to establish connection between two lt interfaces. The encapsulation type of the interfaces must be given. It can be one of the following types Ethernet, Ethernet circuit cross-connect (CCC), Ethernet VPLS, Frame Relay, Frame Relay CCC, VLAN, VLAN CCC, or VLAN VPLS. For the purposes of this project Ethernet encapsulation is used

The following command is the most common format to in order to configure given interface:

```
[edit]
root# set interfaces interface-name unit unit family inet address ip-address
```

In Juniper Networks equipment the "unit" number distinguishes whether the configuration feres either physical interface or logical sub-interface. If the unit number is "0", this means the interface is physical. If the number is any other numeral it stands for logical sub-interface.

The fe, ge and lo interfaces are configured by the following commands:

```
[edit]
root# edit interfaces fe-x/x/x
[edit interfaces fe-x/x/x]
root# set unit unit-number family inet address ip-address
[edit]
root# edit interfaces ge-x/x/x
[edit interfaces ge-x/x/x]
root# set unit unit-number family inet address ip-address
[edit]
root# edit interfaces lo0
[edit interfaces lo0]
root# set unit unit-number family inet address ip-address
```

The lt interfaces can be configured either for physical device or logical system on a router, and are configured with the following configuration lines:

```
[edit]
root# edit interfaces lt-x/x/x
```

```
[edit interfaces lt-x/x/x] or
[edit logical-systems logical-system-name interfaces]
root# set interfaces lt-0/1/0 unit 1 description XXXXX
[edit interfaces lt-x/x/x] or
[edit logical-systems logical-system-name interfaces]
root# set interfaces lt-0/1/0 unit 1 encapsulation Ethernet
[edit interfaces lt-x/x/x] or
[edit logical-systems logical-system-name interfaces]
root# set interfaces lt-0/1/0 unit 1 peer-unit 25
[edit interfaces lt-x/x/x] or
[edit logical-systems logical-system-name interfaces]
root# set interfaces lt-0/1/0 unit 1 family inet ip-address
```

The commands configure the following interfaces (Figure 5.2):

- Logical Tunnel interface lt-0/0/0.12 of CE-1;
- Logical Tunnel interface lt-0/0/0.13 of CE-1;
- Logical Tunnel interface lt-0/0/0.21 of CE-2;
- Logical Tunnel interface lt-0/0/0.23 of CE-2;
- Logical Tunnel interface lt-0/0/0.31 of CE-3;
- Logical Tunnel interface lt-0/0/0.32 of CE-3;
- Gigabit Ethernet interface ge-1/3/5 of CE-1;
- Gigabit Ethernet interface ge-1/3/6 of CE-1;
- Gigabit Ethernet interface ge-1/3/8 of CE-1;
- Gigabit Ethernet interface ge-1/3/4 of CE-2;
- Gigabit Ethernet interface ge-1/3/7 of CE-2;
- Gigabit Ethernet interface ge-1/3/9 of CE-3;
- Gigabit Ethernet interface ge-1/3/3 of CE-3;
- Fast Ethernet interface fe-0/0/2 of ACC-1;
- Fast Ethernet interface fe-0/0/3 of ACC-1;
- Fast Ethernet interface fe-0/0/4 of ACC-1;
- Fast Ethernet interface fe-0/0/5 of ACC-1;
- Fast Ethernet interface fe-0/0/2 of ACC-2;
- Fast Ethernet interface fe-0/0/3 of ACC-1;
- Fast Ethernet interface fe-0/0/4 of ACC-1;
- Loopback interface lo.0 of CE-1;
- Loopback interface lo.0 of CE-2;
- Loopback interface lo.0 of CE-3;
- Loopback interface lo.0 of ACC-1;
- Loopback interface lo.0 of ACC-2;

Sample interface configuration for each interface type is shown on Figure 5.3. The extended interface configurations are given in Appendix A.

```
[edit]
root@core-edge# show logical-systems ce-1 interfaces
lt-0/0/0 {
    unit 12 {
        encapsulation ethernet;
        peer-unit 21;
        family inet {
            address 10.0.12.1/30;
        }
        family mpls;
    }
    unit 13 {
        encapsulation ethernet;
        peer-unit 31;
        family inet {
            address 10.0.13.1/30;
        }
        family mpls;
    }
}
ge-1/3/5 {
    unit 0 {
        bandwidth 2m;
        family inet {
            policer {
                input 2m;
                output 2m;
            }
            address 10.0.101.5/30;
        }
        family mpls;
    }
}
ge-1/3/6 {
    unit 0 {
        bandwidth 2m;
        family inet {
            policer {
                input 2m;
                output 2m;
            }
            address 10.0.101.1/30;
        }
        family mpls;
    }
}
ge-1/3/8 {
    unit 0 {
        family inet {
            address 10.0.201.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.1.1/32 {
                primary;
            }
        }
    }
}
```

Figure 5.3 Sample interface configuration

The interfaces between the Access/Aggregation and Core/Edge devices are shaped on 2Mbit/s. This way E1 lines are emulated, so the NGN network architecture is maximally close to a real on-field network. This E1 lines emulation is due to the compliance with the old mobile core infrastructure. This way the NGN model can be employed, and work with the 2G/3G BTSs infrastructure. Because the most 2G/3G mobile antennas are connected to the existing mobile core via E1 lines.

The commands to set this are the following for the SRX routers - Access/Aggregation devices:

```
[edit]
root# edit interfaces fe-x/x/x
[edit interfaces fe-x/x/x]
root# set unit unit-number bandwidth bandwidth
```

For the MX960 routers - Core/Edge the command-line sequence is as follows:

```
[edit]
root# edit interfaces ge-x/x/x
[edit interfaces ge-x/x/x] or
[edit logical-systems logical-system-name interfaces ge-x/x/x]
root# set unit unit-number bandwidth bandwidth
```

```
[edit interfaces ge-x/x/x] or
[edit logical-systems logical-system-name interfaces ge-x/x/x]
root# set unit unit-number family inet policer input speed-in-mbps
[edit interfaces ge-x/x/x] or
[edit logical-systems logical-system-name interfaces ge-x/x/x]
root# set unit unit-number family inet policer output speed-in-mbps
```

On fig 5.4 sample interface bandwidth shaping configuration is shown.

```
[edit]
root@acc-1# show interfaces
fe-0/0/2 {
    fastether-options {
        auto-negotiation;
    }
    unit 0 {
        bandwidth 2m;
        family inet {
            address 10.0.101.2/30;
        }
        family mpls;
    }
}
fe-0/0/3 {
    fastether-options {
        auto-negotiation;
    }
    unit 0 {
        bandwidth 2m;
        family inet {
            address 10.0.102.2/30;
        }
        family mpls;
    }
}
fe-0/0/4 {
    fastether-options {
        auto-negotiation;
    }
    unit 0 {
        bandwidth 2m;
        family inet {
            address 10.0.101.6/30;
        }
        family mpls;
    }
}
```

Figure 5.4 Bandwidth shaping configuration

5.4. Quality of Service in the designed NGN network architectures

The QoS is common for both Plain IP and IP/MPLS architectures. This way both technologies have the same initial conditions for their evaluation.

5.4.1. Configuring CoS-Based Forwarding

CoS-based forwarding (CBF) gives opportunity to control next-hop selection on a packet's class of service basis - the value of the IP packet's precedence bits. When a routing protocol finds equal-cost paths to a destination, it can choose path at random or load-balance between the paths either via hash selection or round robin. CBF permits path selection on

class basis.

CBF is applied to the equal-cost links between CE-1 and ACC-1. The output is shown on Figure 5.5. CBF is configured with the command set:

```
[edit]
root# set policy-options policy-statement policy-statement-name then
load-balance per-packet
[edit]
root# set forwarding-options hash-key family inet layer-4
```

```
policy-options {
    policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
}
```

Figure 5.5 Load-balancing configuration

For CoS next-hop map to be specified, the forwarding-policy statement must be included at the [edit class-of-service] hierarchy level. This is applied on the LSPs between the devices in the network, thus the IP/MPLS architecture can properly carry the traffic through the network. The Plain IP architecture does not differ, whether there is CoS next-hop map applied, for its forwarding decisions.

The CoS next-hop map is implemented with the next lines:

```
[edit class-of-service]
root# set forwarding-policy next-hop-map map-name forwarding-class
class-name lsp-next-hop lsp-regular-expression
```

When CBF is configured with OSPF as the interior gateway protocol (IGP), next hop as an interface name or next-hop alias must be set, not as an IP address. The reason is OSPF appends routes with interface as a next hop, IP address is not considered.

The JUNOS operating system applies the CoS next-hop map to previously defined next hops; the next hops can be situated across any egress interfaces on the node. The CoS next-hop maps are configured on both Access/Aggregation and Core/Edge devices. The following were used for the configuration of classes and next-hop identifiers:

```
[edit class-of-service forwarding-policy]
root# set next-hop-map map-name forwarding-class expedited-forwarding
lsp-next-hop lsp-next-hop-name
[edit class-of-service forwarding-policy]
root# set next-hop-map map-name forwarding-class best-effort lsp-
next-hop lsp-next-hop-name
```

The route filter the load balancing CBF option, on ACC-1 and CE-1, must be applied as a

route filter in the routing-options hierarchy level on the nodes. This is made with the following command set:

```
[edit]
root# set routing-options forwarding-table export policy-statement-name
```

This option causes the routing processes to insert routes to the forwarding table matching the policy-statement-name with the belonging next-hop CBF rules. This algorithm is used when given configuration is applied to a route:

- If the path is an only next-hop route, all traffic will go to that path, thus CBF will not take effect.
- If a next hop shows for the path, but not in the cos-next-hop map, it won't exist in the forwarding table.
- Default forwarding class takes effect if any forwarding classes are not designed in the next-hop map. If Default is not defined, one is picked randomly.

5.4.2. Scheduler-maps

After scheduler is defined, it can be implemented in a scheduler map, which maps a dedicated forwarding class to a scheduler. Scheduler map is implemented to assign a forwarding class to a scheduler, and then employ the scheduler map to any interface that ought to enforce DiffServ CoS. After they are referred to an interface, the scheduler maps influence the hardware queues, packet schedulers, and RED drop profiles.

In this project, scheduler maps are designed and applied on the interfaces of all devices in the network. The scheduler maps are implemented in the configuration of specific interfaces - to fulfill the QoS requirements on the links between Access/Aggregation and Core/Edge devices in the network architecture.

To configure and implement a scheduler map to a device's interface the following command set has been used:

Configure a scheduler map.

```
[edit class-of-service]
root# edit scheduler-maps map-name
```

Configure a best-effort forwarding class and scheduler.

```
[edit class-of-service]
rootт# set forwarding-class best-effort scheduler data
```

Configure an expedited forwarding class and scheduler.

```
[edit class-of-service]
root# set forwarding-class expedited-forwarding scheduler voice
```

Configure a network control class and scheduler.

```
[edit class-of-service]
root# set forwarding-class network-control scheduler nc
```

Apply the scheduler map to an interface.

```
[edit class-of-service]
root# set interfaces xx-x/x/x unit 0 scheduler-map map-name
```

5.4.3. Schedulers

Individual device interface has numerous queues assigned to store packets temporary before transmitting. To identify the order in which the queues are served, the device utilizes a round-robin scheduling method, based on priority and the queue's weighted round-robin (WRR) credits. Schedulers are configured to allocate resources, priorities, and drop profiles to output queues in the designed network architecture.

In this project, best-effort scheduler called "data" is configured. Its priority is set to low. The "data" scheduler has transmit-rate set to remainder. So the data traffic in the network will use the bandwidth left from the high-priority traffic types. The expedited forwarding scheduler called "voice" is set, its priority is medium-high. The transmit-rate of this scheduler is set to 1m for every real-time stream on the Access/Aggregation devices and to transmit-rate percent 50, which means 50 percent of the available bandwidth on the Core/Edge devices.

The network control scheduler called "nc" is configured and its priority is set to strict-high. To this scheduler 5 percent of the available bandwidth have been given.

To configure CoS schedulers with the next command lines (Figure 5.6):

Configure a best-effort scheduler.

```
[edit]
root# edit class-of-service schedulers best-effort
```

Specify a best-effort scheduler priority and buffer size.

```
[edit class-of-service schedulers data]
root# set priority low
[edit class-of-service schedulers data]
root# set transmit-rate remainder
```

Configure an expedited forwarding scheduler.

```
[edit]
root# edit class-of-service schedulers voice
```

Specify an expedited forwarding scheduler priority and transmit rate.

```
[edit class-of-service schedulers voice]
root# set priority medium-high
[edit class-of-service schedulers voice]
root# set transmit-rate 1m
```

Configure an assured forwarding scheduler.

```
[edit]
root# edit class-of-service schedulers nc
```

Specify an assured forwarding scheduler priority and transmit rate.

```
[edit class-of-service schedulers nc]
root# set priority strict-high
[edit class-of-service schedulers nc]
root# set transmit-rate percent 5
```

```
class-of-service {
    forwarding-policy {
        next-hop-map acc1-ce1 {
            forwarding-class expedited-forwarding
                lsp-next-hop access1-to-ce2-voice
            }
        }
    }
    interfaces {
        fe-0/0/2 {
            scheduler-map access;
        }
        fe-0/0/3 {
            scheduler-map access;
        }
        fe-0/0/4 {
            scheduler-map access;
        }
    }
    scheduler-maps {
        access {
            forwarding-class network-control scheduler nc;
            forwarding-class best-effort scheduler data;
            forwarding-class expedited-forwarding scheduler voice;
        }
    }
    schedulers {
        voice {
            transmit-rate 1m;
            priority medium-high;
        }
        data {
            transmit-rate {
                remainder;
            }
            priority low;
        }
        nc {
            transmit-rate percent 5;
            priority strict-high;
        }
    }
}
```

Figure 5.6 Class of Service configuration

5.5. Designing Plain IP architecture

The Plain IP architecture is based on single-area OSPF Protocol. This architecture considers only the traffic shaping implemented in the network. OSPF gives only opportunity to forward the traffic through the link with least cost. The routes, traffic goes through cannot be manipulated manually. This is constraint in Plain IP network architectures. In case two links

have equal costs load balancing via both of them is implemented. Load balancing is applied with routing policy, which states that this operation must be done from the OSPF. The policy is described in the the previous section.
The Interior Gateway Protocol OSPF is configured on all interfaces in the Plain IP network model. All nodes in the network are set in single area, which is area 0. The access interfaces in the architecture are passive, because no OSPF LSA messages exchange is expected from the links with the end nodes.
OSPF protocol is configured on all routers (Figure 5.7). To configure OSPF the following commands are used:

```
[edit]
root# edit protocols ospf
[edit protocols ospf]
root# set area 0
[edit protocols ospf area 0.0.0.0]
root# set interface interface-name
```

For access interfaces the additional command used is:

```
[edit protocols ospf area 0.0.0.0]
root# set interface interface-name passive
```

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fe-0/0/5.0 {
            passive;
        }
    }
}
```

Figure 5.7 OSPF configuration

5.6. Implementing IP/MPLS network model

The IP/MPLS architecture is created, because unlike the Plain IP model, IP/MPLS gives opportunity to affect the way the traffic goes through the network. In this architecture the routes used by the flow are designed and implemented by the administrator. In the IP/MPLS network primary and secondary LSPs and strict nodes in the network can be defined. The IP/MPLS network model uses the same QoS as Plain IP, so the comparison can be made.

The IP/MPLS network model uses OSPF to satisfy the needs of Constrained Shortest Path First algorithm (CSPF), so MPLS can build labels and routes in the network. The RSVP-TE protocol is used to provide signaling of the LSPs, as well as to assure the End-to-End QoS for the traffic flow running through the network. The LSPs are designed to satisfy the QoS requirements of the traffic flows running in the network architecture. BGP is needed to

provide the signaling of a number of Layer 3 VPNs services can operate together in the architecture. The Layer 3 VPNs are used as a service so the end user traffic can be carried in the network with the proper handling and QoS.

5.6.1. Extending the function of OSPF

In order to operate and establish Label Switched Paths, MPLS needs information from Interior Gateway Protocol such as OSPF or IS-IS. In the implemented IP/MPLS network architecture OSPF is used. The command traffic-engineering must be added to the OSPF configuration, in order to generate LSA (Link State Advertisements) messages, which are compatible with MPLS functionality. This is necessary for filling in the Constrained Shortest Path First algorithm (CSPF) database and to calculate the LSP routes. This is done with the following command.

```
[edit protocols ospf]
root# set traffic-engineering
```

5.6.2. Configuring MPLS protocol

All interfaces on nodes in the Access/Aggregation and Core/Edge regions of the network are configured with family mpls, so those interfaces can operate in IP/MPLS environment. All routers in the backhaul network are configured to run MPLS. This is done with the following steps:

- Configuration of the MPLS itself;
- Configuration of the interfaces used in the MPLS region of the network.

The labels assigned to the traffic flows are dynamically assigned. The static label definition option is not used.

Configuring MPLS on all routers is done using the commands (Figure 5.8):

```
[edit protocols]
root# set mpls interface interface-name
[edit interfaces]
root# set interface-name unit unit-number family mpls
```

```
mpls {
    interface ge-1/3/9.0;
    interface lt-0/0/0.31;
    interface lt-0/0/0.32;
}
```

```
fe-0/0/3 {
    unit 0 {
        family inet {
            address 10.0.203.2/30;
        }
        family mpls;
    }
}
```

Figure 5.8 MPLS Configuration

5.6.3. Configuring RSVP-TE protocol

The designed, in the MPLS domain, LSPs require signaling and provisioning of resources for the demanded QoS.For this purpose the signaling protocol RSVP-TE is implemented.

This protocol provides data so the paths in the network can be built. RSVP-TE generates the information needed to build LSP routes different from the ones existing in the from OSPF database, based on certain criteria. Afterwards RSVP-TE allocates its own resources and finds the shortest path to a remote location on the network. This protocol generates its shortest routes taking in consideration also the QoS requirements. RSVP-TE reserves the needed for each traffic flow resources on each device along the path of the given flow in the network architecture. For the links between Access/Aggregation and Core/Edge nodes reference bandwidth is defined, so the protocol is aware of the shaping applied on the interfaces.

For least data loss in the network RSVP-TE provides mechanisms to send the packets, which are already in the network, to their destination. This is done with the "Fast reroute" algorithms, which serves the "Make-Before-Break" rule. This means that the protocol establishes secondary paths to each destination and in case of link or node failure, the packets sent to the given destination are forwarded via these secondary paths. Establishing these detours works in case of a node or link in the LSP fails, and the traffic on the LSP is delivered with minimal packet loss. Extension for fast reroute is the optimize timer. The optimize timer initiates a periodic optimization process that recalculates the fast reroute secondary LSPs to use network resources in more efficient manner.

RSVP-TE is activated on all nodes in the IP/MPLS architecture (Figure 5.9). This is done with the command set:

```
[edit protocols]
root# set rsvp interface interface-name
[edit protocols]
root# set rsvp interface interface-name bandwidth bandwidth-in-mbps
[edit protocols]
root# set rsvp fast-reroute optimize-timer seconds
```

```
[edit]
root@acc-1# show protocols rsvp
fast-reroute optimize-timer 5;
interface fe-0/0/4.0 {
    bandwidth 2m;
}
interface fe-0/0/3.0 {
    bandwidth 2m;
}
interface fe-0/0/2.0 {
    bandwidth 2m;
}
```

Figure 5.9 RSVP configuration

5.6.4. Configuring BGP protocol

For signaling the MPLS Layer 3 VPN services it the IP/MPLS network architecture BGP protocol is used. This approach allows simultaneous signaling of several different types of services in the same BGP session.

Fist an Autonomous System has to be defined on the nodes, it is needed in order for BGP to operate. All nodes in the network model are in the same Autonomous System - in this project the AS number is 65001. This is private AS number, but it can as well work in collaboration with nodes in the Internet. When all devices in the network operate within single AS, this means that internal BGP (iBGP) sessions are used.

For BGP to run smoothly there are a number of parameters have to be defined (Figure 5.10):

- **Group name** - it has to be the same on all nodes;
- **Group type** - in this project it is internal, meaning that iBGP is in use;
- **Neighbor IP addresses** - the IP addresses of all neighbors the node keeps sessions with;
- Local address - the address the node uses when communicating with its neighbors - in the current network model the Loopback addresses are used;
- **Peer AS** - the AS number of the neighboring device - in this project all devices use single AS number;
- **Family IPv4 VPN** – stating that the type of service is Layer 3 VPN.

Overall configuration of IBGP is created with the following commands:

```
[edit routing-options]
root# set autonomous-system as-number
[edit protocols bgp]
root# set group group-name type type
[edit protocols bgp]
root# set group group-name neighbor ip-address
[edit protocols bgp]
root# set group group-name local-address ip-local-address
[edit protocols bgp]
root# set group group-name family inet-vpn unicast
[edit protocols bgp]
root# set peer-as as-number
```

```
[edit]
root@acc-1# show protocols bgp
local-address 172.16.1.4;
group internal {
    type internal;
    family inet-vpn {
        unicast;
    }
    neighbor 172.16.1.1;
    neighbor 172.16.1.2;
    neighbor 172.16.1.3;
    neighbor 172.16.1.5;
}
```

Figure 5.10 Sample BGP configuration

5.6.5. Label Switched Paths implementation

For the transmission of user traffic Label Switched Paths are required. These LSPs are set explicitly on the Access/Aggregation and Core/Edge devices, which communicate directly with end nodes. All other nodes are just intermediary for the LSPs and need no LSP configuration. RSVP-TE gives the ability to define specific parameters on LSPs. Since RSVP-TE by default chooses the shortest path, for traffic engineering LSP priority must be set to influence the data path.

LSPs are designed on Access/Aggregation and Core/Edge devices as follows:

- from ACC-1 to CE-2 and CE-3;
- from ACC-2 to CE-2 and CE-3;
- from CE-2 to ACC-1 and ACC-2;
- from CE-3 to ACC-1 and ACC-2;

The LSPs implemented on these devices are dedicated per each traffic flow. This means that the voice and the data traffic have different LSPs, and go through different paths in the IP/MPLS network architecture. For the voice traffic the path with minimal latency is set as "primary". In case of link or node failure the data paths are as secondary paths for the voice traffic. For the data traffic the path with maximum available bandwidth is set as "primary". In case of link or node failure the voice paths are as secondary paths for the data traffic.

Mechanisms are used for fast-reroute, which provides a rapid response in case of a link or node failure in the network. Detours are established, so the traffic on the LSP can be rerouted with minimal packet loss. During reroute, doubling of resources on links shared by the old and new paths must be evaded. Including "adaptive" command statement makes RSVP use shared explicit (SE) reservation styles and helps for smooth transition during rerouting.

The "bandwidth" command allows the MPLS tunnels to automatically tune up their bandwidth allocation based on the amount of traffic passing through the tunnel. The maximum and minimum volumes of bandwidth are set on each LSP in bits per second (bps).

For smooth service provisioning and for assurance of the QoS requirements, priorities had been set on the LSPs. There are two types of priorities - setup priority and reservation priority. If there is insufficient bandwidth during session establishment, the setup priority is compared with the other setup priorities to determine if some of them should be preempted to install the new session. The sessions, which have lower priorities are preempted.

- **reservation-priority** - it is used to hold a reservation after it has been established. A lower number has a higher priority. The priority must be more than or equal with the setup priority for protection from preemption loops.

Range: 0 through 7, where 0 is the highest and 7 is the lowest priority. Once the session is established with priority 0, no other session can preempt it.

- **setup-priority** - Setup priority.

Range: 0 through 7, where 0 is the highest and 7 is the lowest priority. 7 means that the session cannot preempt other sessions.

For reservation-priority and setup-priority only numbers are defined in the configuration lines. When new LSPs are established after link or node failure, the LSPs with highest priorities are re-established first.

MPLS path is defined when paths in the designed network need to overcome the routes calculated, based on the IGP database. The paths should have given path-name. The service command "strict" indicates that the LSP must go to the next address specified in the path statement without passing through other nodes if possible. If there are intermediary links or nodes in the network, strict states that the LSP should pass through this point, no matter of the other links or nodes along the path.

The LSPs are configured with the following commands and sample configuration is given on Figure 5.11.

```
[edit protocols mpls]
root# set label-switched-path lsp-path-name to remote-ip-address
[edit protocols mpls]
root# set label-switched-path lsp-path-name to remote-ip-address
fast-reroute
[edit protocols mpls]
root# set label-switched-path lsp-path-name adaptive
[edit protocols mpls]
root# set label-switched-path lsp-path-name auto-bandwidth minimum-
bandwidth bandwidth-in-mbps
[edit protocols mpls]
root# set label-switched-path lsp-path-name auto-bandwidth maximum-
bandwidth bandwidth-in-mbps
[edit protocols mpls]
root# set label-switched-path lsp-path-name priority reservation-
priority setup-priority
[edit protocols mpls]
root# set path path-name remote-ip-address strict
```

```
label-switched-path access1-to-ce2-voice {
    to 172.16.1.2;
    priority 4 0;
    adaptive;
    fast-reroute;
    auto-bandwidth {
        minimum-bandwidth 1m;
        maximum-bandwidth 2m;
    }
    primary acc1-ce2;
}
label-switched-path access1-to-ce2-data {
    to 172.16.1.2;
    priority 7 5;
    adaptive;
    fast-reroute;
    auto-bandwidth {
        minimum-bandwidth 100k;
        maximum-bandwidth 2m;
    }
    primary acc1-ce2-data;
}
path acc1-ce2 {
    10.0.102.1 strict;
}
path acc1-ce2-data {
    10.0.101.1 strict;
    10.0.101.5 strict;
}
```

Figure 5.11 LSP and Path configurations

5.6.6. Building Layer3VPN service

A Layer 3 VPN is a number of sites that share the same routing information.The communication between the sites is managed by policies. The sites that create a Layer 3 VPN are connected over the existing IP/MPLS network architecture. VPN routing information in the created model, and MPLS is used to transfer end nodes VPN traffic through the network to the Application servers.

Each Layer 3 VPN has its own dedicated routing table that keeps the routing information for the given VPN. To separate VPN routes from routes of other VPNs, the Access/Aggregation and Core/Edge nodes, connected to end devices, create a dedicated routing table for each Layer 3 VPN named a VPN routing and forwarding (VRF) table. Each site that is attached Layer 3 VPN can access only the routing information in the VRF table for that VPN. Each layer 3 VPN routing instance is identified by route target attribute, it shows the number of sites (VRF tables) to which a Access/Aggregation and Core/Edge routers serve routes. The border nodes use the route target to limit the entry of foreign routes into its VRF tables.

When an ingress device receives paths advertised from a directly connected end node, it compares the received information with the VRF export policy for the given Layer 3 VPN.

The routes for the Layer 3 VPN services are distributed with iBGP sessions. The iBGP implementation was discussed in the previous sections.

To ensure Layer3VPN service is necessary to define routing instance. Instance type should be given - in this project this is Layer 3 VPN.

The description is optional, but for convenience it is set. Afterwards interface on which the service is running must be specified.

Each implemented Layer 3 VPN must have a unique route distinguisher associated with it. L3VPN VRF tables need a route distinguisher, so BGP can identify between probably identical network layer reachability information (NLRI) messages sent to it from different VPNs.

For the propper operation of the Layer 3 VPN service vrf-target should be set. The vrf-target must be identical on all sites participating in given VPN. For mapping of the inner label of a packet to a specific VRF table vrf-table-label should be used. This gives opportunity to examine the encapsulated IP header and check the QoS requirements for this packet. All paths in the given VRF, which have this option set are advertised with the label dedicated per VRF.

A static route is set to carry the traffic from the Layer 3 VPN instance to the forwarding table of the MPLS. BGP is used for signaling the Layer 3 VPN instance in all parts of the network.

For Layer 3 VPN service implementation the following commands are used:

```
[edit]
root# set routing-instances instance-name instance-type type
[edit routing-instances L3VPN]
root# set description description
[edit routing-instances L3VPN]
root# set vrf-table-label
[edit routing-instances L3VPN]
root# set interface interface-name
[edit routing-instances L3VPN]
root# set vrf-target target:number:number
[edit routing-instances L3VPN]
root# set route-distinguisher number:number
[edit routing-instances L3VPN]
root# set routing-options static route ip-address next-hop next-hop
```

Layer 3 VPN configuration is shown in Figure 5.12.

```
[edit]
root@acc-1# show routing-instances
acc1 {
    description acc1-l3vpn;
    instance-type vrf;
    interface fe-0/0/5.0;
    route-distinguisher 172.16.1.4:100;
    vrf-target target:65001:100;
    vrf-table-label;
    routing-options {
        static {
            route 172.16.1.2/32 next-hop [ 10.0.102.1 10.0.101.1 10.0.101.5 ];
        }
    }
}
```

Figure 5.12 Layer 3 VPN configuration

Configurations of ACC-1 and CE-1 are applied in Appendix A. The configurations of all other devices in the proposed network architectures are analogical with these two configurations.

6. Results from the evaluation of the designed network architectures

The proper functioning of the designed Plain IP and IP/ MPLS network architectures include:

- All protocols to be fully operating;
- Proper implementation of the designed QoS;
- Provisioning of the necessary services - ensuring L3VPN operation;
-redundancy of network resources - this includes rerouting in case of link or node failure.

The necessities for fulfilling these requirements will be discussed with the relevant tests for each of them. To be entrusted the proper functioning of the network first the basic components are checked. Then comparison between Plain IP and IP/MPLS network models is made.

6.1 Performance of the network architecture

Proper operation of the model includes checking the functioning of the components used to build it. The operation of each working protocol is tested. Afterwards measurements of the bandwidth, traffic loss and latency are made.

6.2. IGP protocol OSPF

For both architectures first the OSPF operation is checked, since it is one of the basic components of the designed models. Checking protocol for OSPF routing involves testing its routing table, established neighboring, interfaces and the election of the DR and BDR routers, topology database:

- For checking the routing table the command used is (Figure 6.1):

```
router> show route protocols ospf
```

```
ce2@core-edge:ce-2> show route protocol ospf

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

10.0.13.0/30       *[OSPF/10] 6w3d 07:41:31, metric 2
                      to 10.0.12.1 via lt-0/0/0.21
                    > to 10.0.23.2 via lt-0/0/0.23
10.0.101.0/30      *[OSPF/10] 04:08:40, metric 51
                    > to 10.0.12.1 via lt-0/0/0.21
10.0.101.4/30      *[OSPF/10] 04:09:37, metric 51
                    > to 10.0.12.1 via lt-0/0/0.21
10.0.201.0/30      *[OSPF/10] 6w3d 07:41:31, metric 2
                    > to 10.0.12.1 via lt-0/0/0.21
10.0.203.0/30      *[OSPF/10] 6w3d 07:41:36, metric 2
                    > to 10.0.23.2 via lt-0/0/0.23
172.16.1.1/32      *[OSPF/10] 6w3d 07:41:31, metric 1
                    > to 10.0.12.1 via lt-0/0/0.21
172.16.1.3/32      *[OSPF/10] 6w3d 07:41:36, metric 1
                    > to 10.0.23.2 via lt-0/0/0.23
172.16.1.4/32      *[OSPF/10] 04:08:30, metric 50
                    > to 10.0.102.2 via ge-1/3/7.0
172.16.1.5/32      *[OSPF/10] 1d 22:48:41, metric 2
                    > to 10.0.12.1 via lt-0/0/0.21
                      to 10.0.23.2 via lt-0/0/0.23
193.178.153.0/24   *[OSPF/10] 1d 22:48:41, metric 3
                      to 10.0.12.1 via lt-0/0/0.21
                    > to 10.0.23.2 via lt-0/0/0.23
224.0.0.5/32       *[OSPF/10] 6w3d 07:42:28, metric 1
                      MultiRecv

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

ce2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

```
root@acc-1> show route protocol ospf

inet.0: 19 destinations, 20 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.12.0/30       *[OSPF/10] 04:08:21, metric 51
                      to 10.0.101.1 via fe-0/0/2.0
                      to 10.0.102.1 via fe-0/0/3.0
                    > to 10.0.101.5 via fe-0/0/4.0
10.0.13.0/30       *[OSPF/10] 04:08:21, metric 51
                    > to 10.0.101.1 via fe-0/0/2.0
                      to 10.0.101.5 via fe-0/0/4.0
10.0.23.0/30       *[OSPF/10] 04:08:31, metric 51
                    > to 10.0.102.1 via fe-0/0/3.0
10.0.201.0/30      *[OSPF/10] 04:08:21, metric 51
                      to 10.0.101.1 via fe-0/0/2.0
                    > to 10.0.101.5 via fe-0/0/4.0
10.0.203.0/30      *[OSPF/10] 04:08:21, metric 52
                    > to 10.0.101.1 via fe-0/0/2.0
                      to 10.0.102.1 via fe-0/0/3.0
                      to 10.0.101.5 via fe-0/0/4.0
172.16.1.1/32      *[OSPF/10] 04:08:21, metric 50
                    > to 10.0.101.1 via fe-0/0/2.0
                      to 10.0.101.5 via fe-0/0/4.0
172.16.1.2/32      *[OSPF/10] 04:08:31, metric 50
                    > to 10.0.102.1 via fe-0/0/3.0
172.16.1.3/32      *[OSPF/10] 04:08:21, metric 51
                      to 10.0.101.1 via fe-0/0/2.0
                    > to 10.0.102.1 via fe-0/0/3.0
                      to 10.0.101.5 via fe-0/0/4.0
172.16.1.5/32      *[OSPF/10] 04:08:21, metric 51
                      to 10.0.101.1 via fe-0/0/2.0
                    > to 10.0.101.5 via fe-0/0/4.0
193.178.153.0/24    [OSPF/10] 04:08:21, metric 52
                    > to 10.0.101.1 via fe-0/0/2.0
                      to 10.0.101.5 via fe-0/0/4.0
224.0.0.5/32       *[OSPF/10] 04:10:34, metric 1
                      MultiRecv

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

acc1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

Figure 6.1 OSPF routes

- Testing of the topology database (Figure 6.2):

```
router> show ospf database
```

```
root@acc-1> show ospf database

    OSPF database, Area 0.0.0.0
 Type       ID               Adv Rtr           Seq      Age  Opt  Cksum  Len
Router   172.16.1.1       172.16.1.1       0x8000056e  2148  0x22 0x39c1  96
Router   172.16.1.2       172.16.1.2       0x800005a0  2115  0x22 0x8311  72
Router   172.16.1.3       172.16.1.3       0x8000053b  1350  0x22 0xe4c   72
Router  *172.16.1.4       172.16.1.4       0x8000000c  1594  0x22 0x2b41  72
Router   172.16.1.5       172.16.1.5       0x8000054f  2387  0x22 0xe41e  72
Router   193.178.153.60   193.178.153.60   0x800000bd  2068  0x22 0xb399  36
Network  10.0.12.2        172.16.1.2       0x8000051b   617  0x22 0x27a0  32
Network  10.0.13.2        172.16.1.3       0x80000517  2636  0x22 0x28a0  32
Network  10.0.23.2        172.16.1.3       0x80000518   921  0x22 0xc5f6  32
Network *10.0.101.2       172.16.1.4       0x80000003   466  0x22 0x97ef  32
Network *10.0.101.6       172.16.1.4       0x80000002  2301  0x22 0x7113  32
Network *10.0.102.2       172.16.1.4       0x80000003    92  0x22 0x9aea  32
Network  10.0.201.2       172.16.1.5       0x80000519  1887  0x22 0x10f5  32
Network  10.0.203.2       172.16.1.5       0x8000051a  1387  0x22 0x14ec  32
Network  193.178.153.60   193.178.153.60   0x80000061  2068  0x22 0xd364  32
OpaqArea 1.0.0.1          172.16.1.1       0x80000533    12  0x22 0x6515  28
OpaqArea 1.0.0.1          172.16.1.2       0x8000051a  2615  0x22 0x9bf5  28
OpaqArea 1.0.0.1          172.16.1.3       0x80000517  1779  0x22 0xa5ec  28
OpaqArea*1.0.0.1          172.16.1.4       0x80000003  1215  0x22 0xe0c8  28
OpaqArea 1.0.0.1          172.16.1.5       0x8000051c   887  0x22 0xa3e5  28
OpaqArea 1.0.0.3          172.16.1.1       0x80000002  2573  0x22 0x27cd 164
OpaqArea 1.0.0.3          172.16.1.2       0x80000002  1616  0x22 0x510e 124
OpaqArea 1.0.0.3          172.16.1.3       0x800004ce  2207  0x22 0xad9a 164
OpaqArea*1.0.0.3          172.16.1.4       0x80000003   841  0x22 0x2f2e 124
OpaqArea 1.0.0.3          172.16.1.5       0x800004c6   387  0x22 0x5aa2 124
OpaqArea 1.0.0.4          172.16.1.1       0x800004d3   869  0x22 0x4107 164
OpaqArea 1.0.0.4          172.16.1.2       0x800004c4  1117  0x22 0x4a3c 124
OpaqArea 1.0.0.4          172.16.1.3       0x800004c2   493  0x22 0x7a0b 124
OpaqArea*1.0.0.4          172.16.1.4       0x80000003  1947  0x22 0x25c0 124
OpaqArea 1.0.0.4          172.16.1.5       0x800004c4  2887  0x22 0xb841 124
OpaqArea 1.0.0.5          172.16.1.1       0x800004c4  1298  0x22 0x285f 124
OpaqArea 1.0.0.5          172.16.1.2       0x800004c4   117  0x22 0x4a26 124
OpaqArea 1.0.0.5          172.16.1.3       0x800004c2    64  0x22 0x660a 124
OpaqArea*1.0.0.5          172.16.1.4       0x80000002  2654  0x22 0xed66 124
OpaqArea 1.0.0.6          172.16.1.1       0x800004c9   441  0x22 0x56db 124
OpaqArea 1.0.0.7          172.16.1.1       0x80000002  1723  0x22 0xcf19 164
```

Figure 6.2 OSPF database

- Verification of the interfaces and the selection of the DR and BDR routers used is command line (Figure 6.3):

```
router> show ospf interfaces
```

```
root@acc-1> show ospf interface
Interface           State   Area            DR ID           BDR ID          Nbrs
fe-0/0/2.0          DR      0.0.0.0         172.16.1.4      172.16.1.1         1
fe-0/0/3.0          DR      0.0.0.0         172.16.1.4      172.16.1.2         1
fe-0/0/4.0          DR      0.0.0.0         172.16.1.4      172.16.1.1         1
lo0.1               DR      0.0.0.0         172.16.1.4      0.0.0.0            0
sp-0/0/0.0          PtToPt  0.0.0.0         0.0.0.0         0.0.0.0            0


ce1@core-edge:ce-1> show ospf interface
Interface           State   Area            DR ID           BDR ID          Nbrs
ge-1/3/5.0          BDR     0.0.0.0         172.16.1.4      172.16.1.1         1
ge-1/3/6.0          BDR     0.0.0.0         172.16.1.4      172.16.1.1         1
ge-1/3/8.0          BDR     0.0.0.0         172.16.1.5      172.16.1.1         1
lo0.1               DR      0.0.0.0         172.16.1.1      0.0.0.0            0
lt-0/0/0.12         BDR     0.0.0.0         172.16.1.2      172.16.1.1         1
lt-0/0/0.13         BDR     0.0.0.0         172.16.1.3      172.16.1.1         1



ce2@core-edge:ce-2> show ospf interface
Interface           State   Area            DR ID           BDR ID          Nbrs
ge-1/3/7.0          BDR     0.0.0.0         172.16.1.4      172.16.1.2         1
lo0.2               DR      0.0.0.0         172.16.1.2      0.0.0.0            0
lt-0/0/0.21         DR      0.0.0.0         172.16.1.2      172.16.1.1         1
lt-0/0/0.23         BDR     0.0.0.0         172.16.1.3      172.16.1.2         1
```

Figure 6.3 OSPF Interfaces

- Neighboring between devices (Figure 6.4) can be seen using the command:

```
router> show ospf neighbor
```

```
root@acc-1> show ospf neighbor
Address          Interface              State     ID              Pri  Dead
10.0.101.1       fe-0/0/2.0             Full      172.16.1.1      128    35
10.0.102.1       fe-0/0/3.0             Full      172.16.1.2      128    34
10.0.101.5       fe-0/0/4.0             Full      172.16.1.1      128    38

ce1@core-edge:ce-1> show ospf neighbor
Address          Interface              State     ID              Pri  Dead
10.0.101.6       ge-1/3/5.0             Full      172.16.1.4      128    34
10.0.101.2       ge-1/3/6.0             Full      172.16.1.4      128    38
10.0.201.2       ge-1/3/8.0             Full      172.16.1.5      128    33
10.0.12.2        lt-0/0/0.12            Full      172.16.1.2      128    32
10.0.13.2        lt-0/0/0.13            Full      172.16.1.3      128    34

ce2@core-edge:ce-2> show ospf neighbor
Address          Interface              State     ID              Pri  Dead
10.0.102.2       ge-1/3/7.0             Full      172.16.1.4      128    34
10.0.12.1        lt-0/0/0.21            Full      172.16.1.1      128    36
10.0.23.2        lt-0/0/0.23            Full      172.16.1.3      128    35
```

Figure 6.4 OSPF neighbors

- A brief status report (Figure 6.5) is given with the command line:

```
router> show ospf overview
```

```
root@acc-1> show ospf overview
Instance: master
  Router ID: 172.16.1.4
  Route table index: 0
  LSA refresh time: 50 minutes
  Traffic engineering
  Area: 0.0.0.0
    Stub type: Not Stub
    Authentication Type: None
    Area border routers: 0, AS boundary routers: 0
    Neighbors
      Up (in full state): 3
  Topology: default (ID 0)
    Prefix export count: 0
    Full SPF runs: 16
    SPF delay: 0.200000 sec, SPF holddown: 5 sec, SPF rapid runs: 3
    Backup SPF: Not Needed
```

Figure 6.5 OSPF overview

From the output of these commands can be that each router is connected to the others devices loopback addresses, which is an important prerequisite for the proper functioning of the other components of the network.

The outcome of the routers in Figure 6.2 means that the OSPF protocol successfully established its topology database of the network, and built its routing table.

The information about OSPF interfaces (Figure 6.3) shows successful of DR and BDR election. This is important for updating routing information when there is a change of the network topology.

The outcome of the routers (Figure 6.4) shows that they made neighboring with each other, and the links between them are functioning normally.

From the brief information of the protocol (Figure 6.5) can be seen that the establishment of the routing table has passed. It is further understood that the protocol is configured to work with the signaling protocol RSVP.

6.3. Signaling protocol RSVP

To examine the implementation of the RSVP protocol the following commands are used:

```
router> show rsvp interface
router> show rsvp session
```

The outputs of these commands are shown in Figures 6.6 and 6.7.

```
root@acc-1> show rsvp interface
RSVP interface: 3 active
                    Active Subscr- Static      Available   Reserved    Highwater
Interface   State resv   iption  BW          BW          BW          mark
fe-0/0/2.0  Up         0   100%  2Mbps       2Mbps       0bps        0bps
fe-0/0/3.0  Up         2   100%  2Mbps       900kbps     1.1Mbps     1.1Mbps
fe-0/0/4.0  Up         0   100%  2Mbps       2Mbps       0bps        0bps


ce1@core-edge:ce-1> show rsvp interface
RSVP interface: 5 active
                    Active Subscr- Static      Available   Reserved    Highwater
Interface   State resv   iption  BW          BW          BW          mark
ge-1/3/5.0  Up         0   100%  2Mbps       2Mbps       0bps        0bps
ge-1/3/6.0  Up         0   100%  2Mbps       2Mbps       0bps        0bps
ge-1/3/8.0  Up         0   100%  100Mbps     100Mbps     0bps        0bps
lt-0/0/0.12 Up         0   100%  10Gbps      10Gbps      0bps        0bps
lt-0/0/0.13 Up         0   100%  1000Mbps    -           0bps        0bps


ce2@core-edge:ce-2> show rsvp interface
RSVP interface: 3 active
                    Active Subscr- Static      Available   Reserved    Highwater
Interface   State resv   iption  BW          BW          BW          mark
ge-1/3/7.0  Up         2   100%  2Mbps       2Mbps       0bps        0bps
lt-0/0/0.21 Up         0   100%  10Gbps      10Gbps      0bps        0bps
lt-0/0/0.23 Up         0   100%  10Gbps      10Gbps      0bps        0bps
```

Figure 6.6 RSVP interfaces

```
root@acc-1> show rsvp session
Ingress RSVP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.2      172.16.1.4      Up       0  1 SE       -        0 access1-to-ce2-voice
172.16.1.2      172.16.1.4      Up       0  1 SE       -        7 access1-to-ce2-data
Total 2 displayed, Up 2, Down 0

Egress RSVP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.4      172.16.1.2      Up       0  1 SE       3        - ce2-to-access1-voice
Egress RSVP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.4      172.16.1.2      Up       0  1 SE       5        - ce2-to-access1-data
Total 2 displayed, Up 2, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0



ce2@core-edge:ce-2> show rsvp session
Ingress RSVP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.4      172.16.1.2      Up       0  1 SE       -        3 ce2-to-access1-voice
172.16.1.4      172.16.1.2      Up       0  1 SE       -       12 ce2-to-access1-data
Total 2 displayed, Up 2, Down 0

Egress RSVP: 2 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.2      172.16.1.4      Up       0  1 SE       0        - access1-to-ce2-voice
172.16.1.2      172.16.1.4      Up       0  1 SE      14        - access1-to-ce2-data
Total 2 displayed, Up 2, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Figure 6.7 RSVP sessions

From the outputs of these commands can be concluded that the signaling protocol RSVP operates correctly, all interfaces are operational and neighboring are built between the nodes.

An important point is the established sessions between the devices, which are successfully made. This means that the QoS requirements will be served.

6.4. Performance of MPLS

To check the operation of MPLS commands used are:

```
router> show mpls call-admission-control
```

- Displays forwarding information management - Figure 6.8;

```
root@acc-1> show mpls call-admission-control
LSP name: access1-to-ce2-data
  Primary   acc1-ce2-data
  Available bandwidth: 0bps

LSP name: access1-to-ce2-voice
 *Primary   acc1-ce2
  Available bandwidth: 1000kbps


ce2@core-edge:ce-2> show mpls call-admission-control
LSP name: ce2-to-access1-data
  Primary   ce2-acc1-data
  Available bandwidth: 0bps

LSP name: ce2-to-access1-voice
 *Primary   ce2-acc1
  Available bandwidth: 0bps
```

Figure 6.8 MPLS call-admission control

```
router> show mpls interfaces
```

- Displays the status of interfaces - Figure 6.9;

```
root@acc-1> show mpls interface
Interface         State       Administrative groups (x: extended)
fe-0/0/2.0        Up          <none>
fe-0/0/3.0        Up          <none>
fe-0/0/4.0        Up          <none>


ce1@core-edge:ce-1> show mpls interface
Interface         State       Administrative groups (x: extended)
ge-1/3/5.0        Up          <none>
ge-1/3/8.0        Up          <none>
lt-0/0/0.12       Up          <none>
lt-0/0/0.13       Up          <none>
ge-1/3/6.0        Up          <none>


ce2@core-edge:ce-2> show mpls interface
Interface         State       Administrative groups (x: extended)
lt-0/0/0.21       Up          <none>
lt-0/0/0.23       Up          <none>
ge-1/3/7.0        Up          <none>
```

Figure 6.9 MPLS interfaces

```
router> show mpls lsp
```

- Shows the designed LSPs - Figure 6.10;

```
root@acc-1> show mpls lsp
Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
172.16.1.2      172.16.1.4      Up     0 *     acc1-ce2         access1-to-ce2-voice
172.16.1.2      172.16.1.4      Up     0       acc1-ce2-data    access1-to-ce2-data
Total 2 displayed, Up 2, Down 0

Egress LSP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.4      172.16.1.2      Up       0  1 SE       3        - ce2-to-access1-voice
Total 1 displayed, Up 1, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0



ce2@core-edge:ce-2> show mpls lsp
Ingress LSP: 2 sessions
To              From            State Rt P     ActivePath       LSPname
172.16.1.4      172.16.1.2      Up     0 *     ce2-acc1         ce2-to-access1-voice
172.16.1.4      172.16.1.2      Up     0       ce2-acc1-data    ce2-to-access1-data
Total 2 displayed, Up 2, Down 0

Egress LSP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
172.16.1.2      172.16.1.4      Up       0  1 SE       0        - access1-to-ce2-voice
Total 1 displayed, Up 1, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Figure 6.10 MPLS LSPs

```
router> show mpls path
```

- Shows the configured paths- Figure 6.11.

```
root@acc-1> show mpls path
Path name                       Address         strict/loose if-id
acc1-ce2                        10.0.102.1      strict<empty>
acc1-ce2-data                   10.0.101.1      strict<empty>
                                10.0.101.5      strict<empty>

ce2@core-edge:ce-2> show mpls path
Path name                       Address         strict/loose if-id
ce2-acc1                        10.0.102.2      strict<empty>
ce2-acc1-data                   10.0.12.1       strict<empty>
```

Figure 6.11 MPLS Paths

The router is configured to send the information explicitly on the path established via CE-1 (for data traffic) and CE-2 (for voice traffic) (Figure 6.8).

Figure 6.9 shows that the interfaces of the routers are fully functional.

From the outputs of the devices in Figure 6.10 can be concluded that the established LSPs function correctly.

Figure 6.11 shows the explicit paths that are set to the Application servers to redirect the voice data through the low latency connections and the data flows through the high bandwidth connections.

When functioning, MPLS creates routing table entries (Figure 6.12). For the provided Layer 3 VPN service that is provided, MPLS creates separate routing table - the paths have different labels assigned for forwarding data. LSP configured to create their own entries in the routing table that contain information about the metrics of the different paths.

```
inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.2/32      *[RSVP/7/1] 05:01:14, metric 50
                    > to 10.0.102.1 via fe-0/0/3.0, label-switched-path access1-to-ce2-voice

acc1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.1.0/24     *[Direct/0] 06:36:54
                    > via fe-0/0/5.0
192.168.1.1/32     *[Local/0] 06:36:58
                      Local via fe-0/0/5.0
192.168.200.0/24   *[BGP/170] 01:17:15, localpref 100, from 172.16.1.2
                      AS path: I
                    > to 10.0.102.1 via fe-0/0/3.0, label-switched-path access1-to-ce2-voice

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

0                  *[MPLS/0] 06:37:49, metric 1
                      Receive
1                  *[MPLS/0] 06:37:49, metric 1
                      Receive
2                  *[MPLS/0] 06:37:49, metric 1
                      Receive
16                 *[VPN/0] 06:37:17
                      to table acc1.inet.0, Pop

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.2:100:192.168.200.0/24
                   *[BGP/170] 01:17:15, localpref 100, from 172.16.1.2
                      AS path: I
                    > to 10.0.102.1 via fe-0/0/3.0, label-switched-path access1-to-ce2-voice
```

Figure 6.12 MPLS routing table entries

6.5. Protocol BGP

A single AS number is used in the whole network architecture. This way the operation of the network is simplified. Also AS numbers are saved since their count is ceasing fast with the rapid growth of networks and network users.

The BGP operation is checked with the following commands:

```
router> show bgp group
router> show bgp summary
```

```
router> show bgp route
router> show bgp neighbors
router> show route receiving-protocol bgp
router> show route advertising-protocol bgp
```

The outputs of the devices for Layer 3 VPN service are shown in Figures: 6.13, 6.14, 6.15, 6.16, 6.17, and 6.18.

```
ce2@core-edge:ce-2> show bgp group
Group Type: Internal    AS: 65001                 Local AS: 65001
  Name: internal        Index: 0                  Flags: <Export Eval>
  Holdtime: 0
  Total peers: 4        Established: 1
  172.16.1.3
  172.16.1.1
  172.16.1.4+59577
  172.16.1.5
  bgp.l3vpn.0: 1/1/1/0
  ce2.inet.0: 1/1/1/0

Groups: 1  Peers: 4    External: 0    Internal: 4    Down peers: 3   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
ce2.mdt.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       1          1          0          0          0          0
ce2.inet.0
                       1          1          0          0          0          0


root@acc-1> show bgp group
Group Type: Internal    AS: 65001                 Local AS: 65001
  Name: internal        Index: 0                  Flags: <Export Eval>
  Holdtime: 0
  Total peers: 4        Established: 1
  172.16.1.1
  172.16.1.2+179
  172.16.1.3
  172.16.1.5
  bgp.l3vpn.0: 1/1/1/0
  acc1.inet.0: 1/1/1/0

Groups: 1  Peers: 4    External: 0    Internal: 4    Down peers: 3   Flaps: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
acc1.mdt.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       1          1          0          0          0          0
acc1.inet.0
                       1          1          0          0          0          0
```

Figure 6.13 BGP groups

```
ce2@core-edge:ce-2> show bgp neighbor

Peer: 172.16.1.4+59577 AS 65001 Local: 172.16.1.2+179 AS 65001
  Type: Internal    State: Established    Flags: <ImportEval Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.1.2 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 172.16.1.4      Local ID: 172.16.1.2        Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  NLRI of received end-of-rib markers: inet-vpn-unicast
  NLRI of all end-of-rib markers sent: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 65001)
  Peer does not support Addpath
  Table bgp.l3vpn.0
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not advertising
    Active prefixes:              1
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
  Table ce2.inet.0 Bit: 40000
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          1
  Last traffic (seconds): Received 17   Sent 16   Checked 62
  Input messages:  Total 9      Updates 2       Refreshes 0     Octets 287
  Output messages: Total 9      Updates 1       Refreshes 0     Octets 306
  Output Queue[2]: 0
  Output Queue[3]: 0
```

```
root@acc-1> show bgp neighbor

Peer: 172.16.1.2+179 AS 65001  Local: 172.16.1.4+59577 AS 65001
  Type: Internal    State: Established    Flags: <ImportEval Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: Open Message Error
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.1.4 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Error: 'Open Message Error' Sent: 1 Recv: 0
  Peer ID: 172.16.1.2      Local ID: 172.16.1.4        Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  NLRI of received end-of-rib markers: inet-vpn-unicast
  NLRI of all end-of-rib markers sent: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 65001)
  Peer does not support Addpath
  Table bgp.l3vpn.0
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not advertising
    Active prefixes:              1
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
  Table acc1.inet.0 Bit: 40000
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            1
    Accepted prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          1
  Last traffic (seconds): Received 16   Sent 17   Checked 62
  Input messages:  Total 8      Updates 2       Refreshes 0     Octets 228
  Output messages: Total 11     Updates 1       Refreshes 0     Octets 388
  Output Queue[2]: 0
  Output Queue[3]: 0
```

Figure 6.14 BGP neighbors

```
ce2@core-edge:ce-2> show route protocol bgp

inet.0: 18 destinations, 18 routes (18 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

ce2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.1.0/24     *[BGP/170] 00:00:38, localpref 100, from 172.16.1.4
                      AS path: I, validation-state: unverified
                    > to 10.0.102.2 via ge-1/3/7.0, label-switched-path ce2-to-access1-voice

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.4:100:192.168.1.0/24
                   *[BGP/170] 00:00:38, localpref 100, from 172.16.1.4
                      AS path: I, validation-state: unverified
                    > to 10.0.102.2 via ge-1/3/7.0, label-switched-path ce2-to-access1-voice



root@acc-1> show route protocol bgp

inet.0: 19 destinations, 20 routes (19 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

acc1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.200.0/24   *[BGP/170] 00:01:50, localpref 100, from 172.16.1.2
                      AS path: I
                    > to 10.0.102.1 via fe-0/0/3.0, label-switched-path access1-to-ce2-voice

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.2:100:192.168.200.0/24
                   *[BGP/170] 00:01:50, localpref 100, from 172.16.1.2
                      AS path: I
                    > to 10.0.102.1 via fe-0/0/3.0, label-switched-path access1-to-ce2-voice
```

Figure 6.15 BGP routes

```
root@acc-1> show bgp summary
Groups: 1 Peers: 4 Down peers: 3
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State
172.16.1.1            65001          1          4       0       0        7:03 Active
172.16.1.2            65001         16         19       0       0        6:15 Establ
  bgp.l3vpn.0: 1/1/1/0
  acc1.inet.0: 1/1/1/0
172.16.1.3            65001          1          4       0       0        7:03 Active
172.16.1.5            65001          1          4       0       0        7:03 Active


ce1@core-edge:ce-1> show bgp summary
Groups: 1 Peers: 4 Down peers: 2
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State
172.16.1.2            65001          1          2       0       1        6:35 Active
172.16.1.3            65001     143373     143352       0       0      6w3d6h 0/0/0/0
172.16.1.4            65001          1          2       0       5        7:04 Active
172.16.1.5            65001     143317     143336       0       0      6w3d5h 0/0/0/0


ce2@core-edge:ce-2> show bgp summary
Groups: 1 Peers: 4 Down peers: 3
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State
172.16.1.1            65001          1          8       0       0        7:56 Active
172.16.1.3            65001          1          2       0       0        7:56 Active
172.16.1.4            65001         20         20       0       0        7:37 Establ
  bgp.l3vpn.0: 1/1/1/0
  ce2.inet.0: 1/1/1/0
172.16.1.5            65001          1          2       0       0        7:56 Active
```

Figure 6.16 BGP summary

```
root@acc-1> show route receive-protocol bgp 172.16.1.2 extensive

inet.0: 19 destinations, 20 routes (19 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

acc1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
* 192.168.200.0/24 (1 entry, 1 announced)
     Import Accepted
     Route Distinguisher: 172.16.1.2:100
     VPN Label: 16
     Nexthop: 172.16.1.2
     Localpref: 100
     AS path: I
     Communities: target:65001:100

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 172.16.1.2:100:192.168.200.0/24 (1 entry, 0 announced)
     Import Accepted
     Route Distinguisher: 172.16.1.2:100
     VPN Label: 16
     Nexthop: 172.16.1.2
     Localpref: 100
     AS path: I
     Communities: target:65001:100
```

```
ce2@core-edge:ce-2> show route receive-protocol bgp 172.16.1.4 extensive

inet.0: 18 destinations, 18 routes (18 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

ce2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
* 192.168.1.0/24 (1 entry, 1 announced)
     Import Accepted
     Route Distinguisher: 172.16.1.4:100
     VPN Label: 16
     Nexthop: 172.16.1.4
     Localpref: 100
     AS path: I
     Communities: target:65001:100

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 172.16.1.4:100:192.168.1.0/24 (1 entry, 0 announced)
     Import Accepted
     Route Distinguisher: 172.16.1.4:100
     VPN Label: 16
     Nexthop: 172.16.1.4
     Localpref: 100
     AS path: I
     Communities: target:65001:100
```

Figure 6.17 BGP received routes

```
root@acc-1> show route advertising-protocol bgp 172.16.1.2 extensive

acc1.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
* 192.168.1.0/24 (1 entry, 1 announced)
 BGP group internal type Internal
     Route Distinguisher: 172.16.1.4:100
     VPN Label: 16
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [65001] I
     Communities: target:65001:100


ce2@core-edge:ce-2> show route advertising-protocol bgp 172.16.1.4 extensive

ce2.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
* 192.168.200.0/24 (1 entry, 1 announced)
 BGP group internal type Internal
     Route Distinguisher: 172.16.1.2:100
     VPN Label: 16
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [65001] I
     Communities: target:65001:100
```

Figure 6.18 BGP advertised routes

Of the outputs of the nodes show that BGP is fully operational, and has established neighboring. BGP sessions are established. The L3VPN groups are properly signaled. The end nodes traffic is properly forwarded and there is communication between the nodes in the L3VPN services.

6.6. Performance of established Layer 3 VPN Service

Operation of the Layer 3 VPN service is checked with the command:

`router> show route instance` *`instance`* `detail`

The functioning of the L3VPN service is shown in Figure 6.19.

```
root@acc-1> show route instance acc1 detail
acc1:
  Description: acc1-l3vpn
  Router ID: 192.168.1.1
  Type: vrf                 State: Active
  Interfaces:
    fe-0/0/5.0
    lsi.0
  Route-distinguisher: 172.16.1.4:100
  Vrf-import: [ __vrf-import-acc1-internal__ ]
  Vrf-export: [ __vrf-export-acc1-internal__ ]
  Vrf-import-target: [ target:65001:100 ]
  Vrf-export-target: [ target:65001:100 ]
  Fast-reroute-priority: low
  Tables:
    acc1.inet.0             : 3 routes (3 active, 0 holddown, 0 hidden)


ce2@core-edge:ce-2> show route instance ce2 detail
ce2:
  Description: acc1-l3vpn
  Router ID: 192.168.200.1
  Type: vrf                 State: Active
  Interfaces:
    ge-1/3/4.0
    lsi.16777472
  Route-distinguisher: 172.16.1.2:100
  Vrf-import: [ __vrf-import-ce2-internal__ ]
  Vrf-export: [ __vrf-export-ce2-internal__ ]
  Vrf-import-target: [ target:65001:100 ]
  Vrf-export-target: [ target:65001:100 ]
  Fast-reroute-priority: low
  Tables:
    ce2.inet.0              : 3 routes (3 active, 0 holddown, 0 hidden)
```

Figure 6.19 Layer 3 VPN functionality

6.7. Quality of Service and Latency verification of the tested network architectures

6.7.1 QoS evaluation

The applied QoS in both network architectures is checked with the command (Figure 6.20). In this figure the operation of the applied forwarding classes, schedulers, and scheduler maps can be seen.

```
root@acc-1> show class-of-service scheduler-map access
Scheduler map: access, Index: 56851

  Scheduler: data, Forwarding class: best-effort, Index: 35538
    Transmit rate: remainder, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: low
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

  Scheduler: voice, Forwarding class: expedited-forwarding, Index: 12115
    Transmit rate: 1000000 bps, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: medium-high
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

  Scheduler: nc, Forwarding class: network-control, Index: 3491
    Transmit rate: 5 percent, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: strict-high
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>
```

```
root@core-edge> show class-of-service scheduler-map access
Scheduler map: access, Index: 56851

  Scheduler: data, Forwarding class: best-effort, Index: 35538
    Transmit rate: remainder, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: low
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

  Scheduler: voice, Forwarding class: expedited-forwarding, Index: 12115
    Transmit rate: 50 percent, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: medium-high
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

  Scheduler: nc, Forwarding class: network-control, Index: 3491
    Transmit rate: 5 percent, Rate Limit: none, Buffer size: remainder,
    Buffer Limit: none, Priority: strict-high
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>
```

Figure 6.20 Applied QoS

The QoS in the architectures is tested with NTOOLS modules NGEN and NRECV. A couple of scenarios are tested with different traffic streams with different parameters and speeds.

In the first scenario the generated traffic consists of two streams of real-time application traffic streams and two data traffic flows. The first real-time stream is 1mbit/s, the UDP port is set to 5060, which is the standard Session Initiation Protocol (SIP) port, the type of service is set to A0 (hex), which that minimal delay and maximum reliability identifiers are set in the parameters of the traffic. The second real-time flow has the same parameters, but its speed is set to 500kbit/s. The two real-time traffic flows emulate two end nodes connected to the ACC-1 node. The data streams use TCP port 80 and are with speed of 500kbit/s. The test is made between ACC-1 and CE-2 nodes. The results of this trial are given on figures 6.21, 6.22 and 6.23.

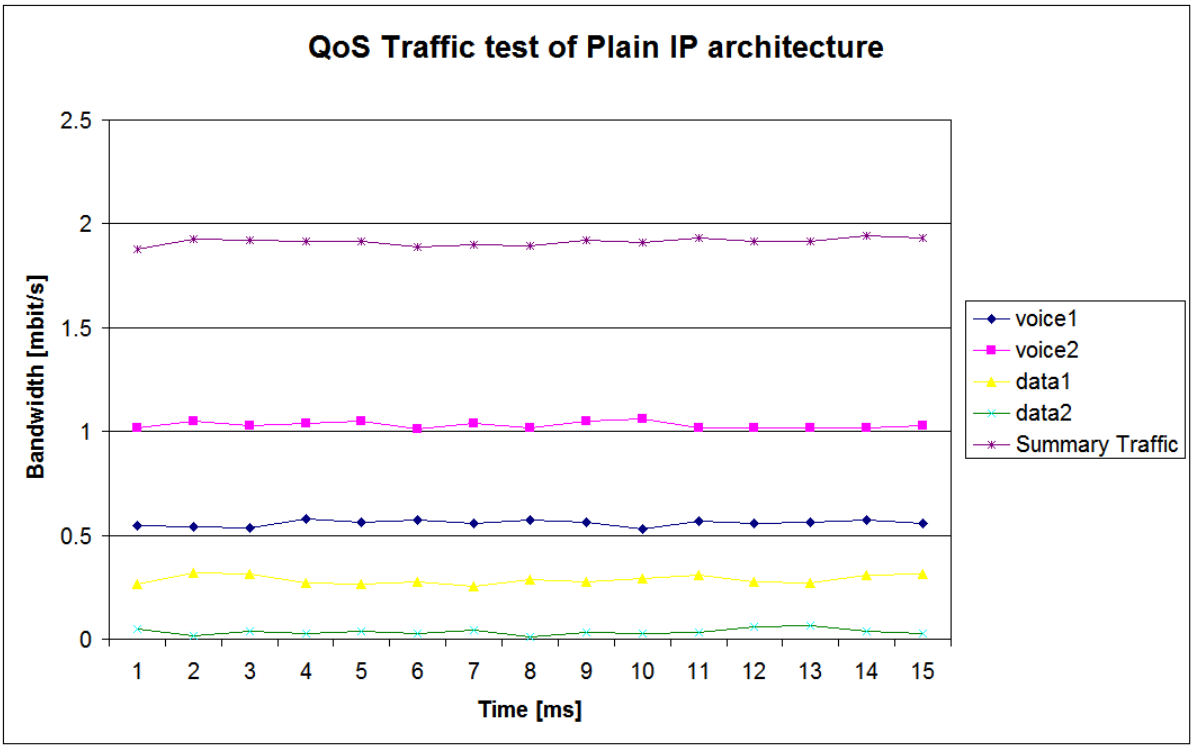

Figure 6.21 QoS test of Plain IP architecture with applied 2.5 Mbit/s traffic

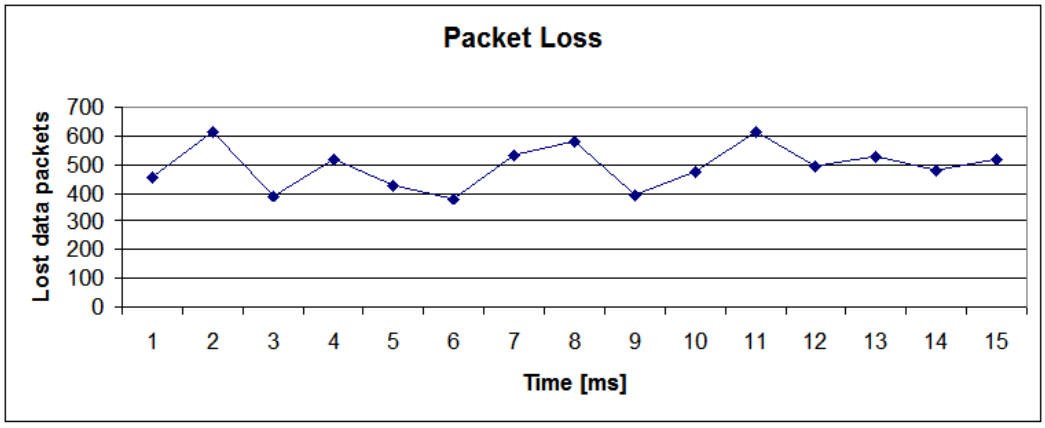

Figure 6.22 Packet loss in Plain IP architecture with applied 2.5 Mbit/s traffic

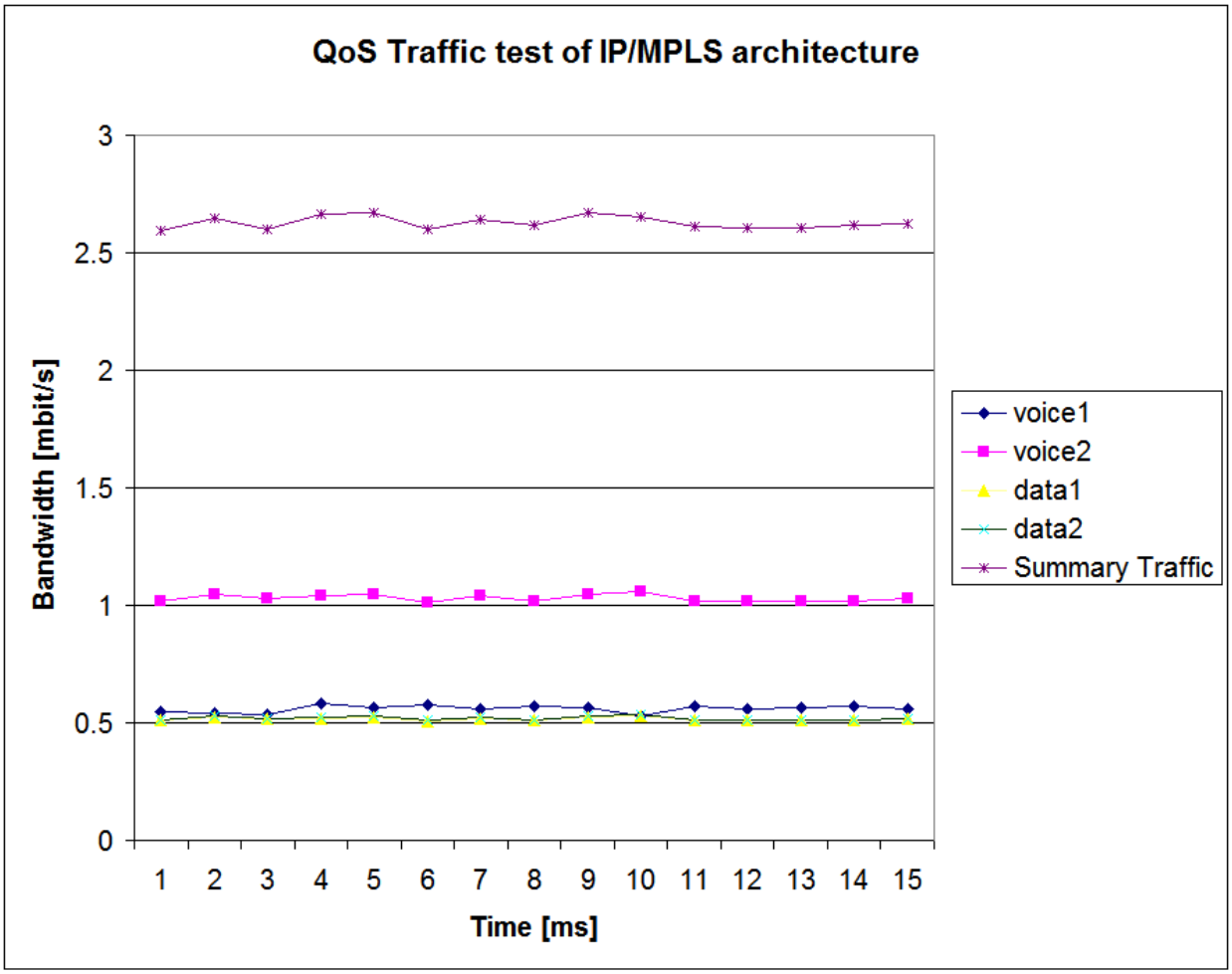

Figure 6.24 QoS test of IP/MPLS architecture with applied 2.5 Mbit/s traffic

From the figures can be seen that the Plain IP architecture suppresses the data stream, because all the traffic is forced to go through the "shortest path" to the destinations. It can be seen that big amount of packet losses can be observed. The packet loss with 2mbit/s traffic flow exists due to the 5 percent bandwidth reserved for network control traffic. In the same time the IP/MPLS model sends all traffic, because the LSPs are designed to send the data traffic through the links with greater latency, but much more available bandwidth. This way all kinds of traffic flows are served with their specific QoS requirements. The IP/MPLS architecture suffers from packet loss when 6mbit/s traffic is streamed due to network control reservation (Figures 6.24 and 6.25).

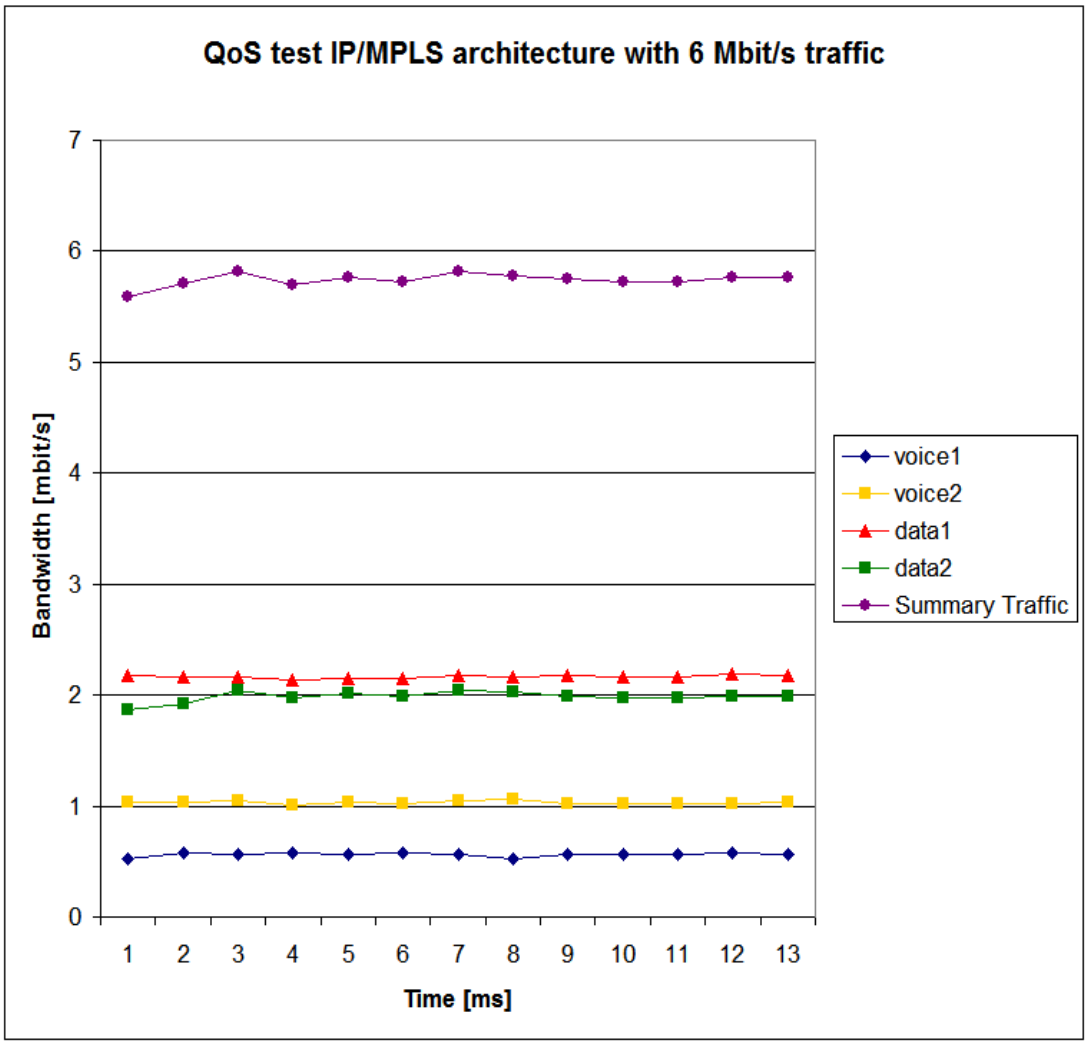

Figure 6.24 QoS test of IP/MPLS architecture with applied 6 Mbit/s traffic

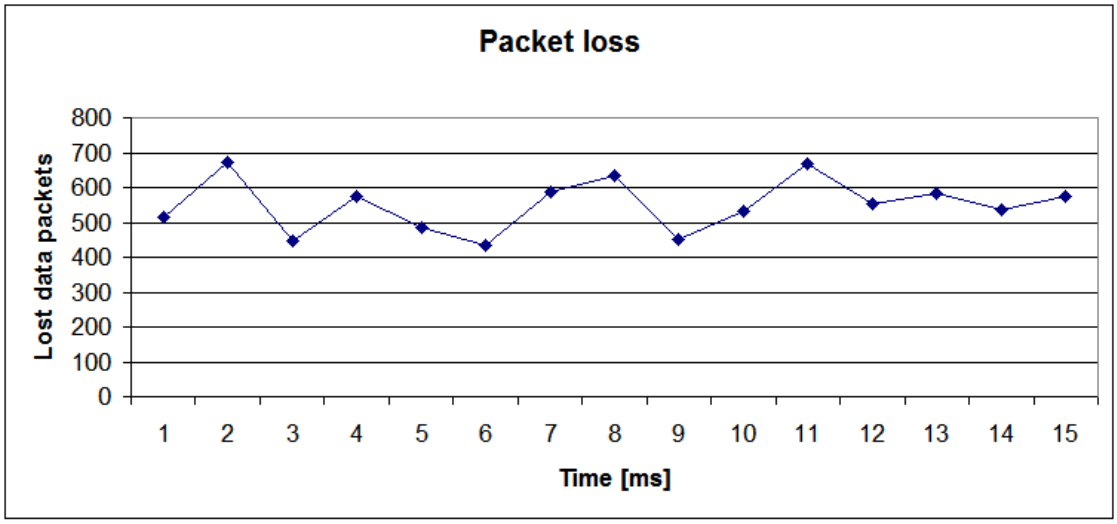

Figure 6.25 Packet loss in IP/MPLS architecture with applied 6 Mbit/s traffic

In case of link failure - the direct connection between ACC-1 and CE-2 both architectures have the same behavior in regard to packet loss. The reason for this is that both models use the load-balanced links between ACC-1 and CE-1 nodes. In this case the Plain IP architecture chooses the path through the double link, because there is no "shorter" path available. IP/MPLS network model uses these links as a secondary for the voice traffic in case of link failure between ACC-1 and CE-2 nodes. Since the available bandwidth for both architectures is the same, they show the same results.

6.7.2. Latency measurements in the built network models

The latency is measured with specialized JDSU tester. It can be seen that the IP/MPLS architecture shows lower latency compared to the Plain IP model. The reason for this is that MPLS doesn't perform route lookup for every packet. MPLS just switches the label and forwards the packet to its next hop in the network. This is shown on figure 6.26.

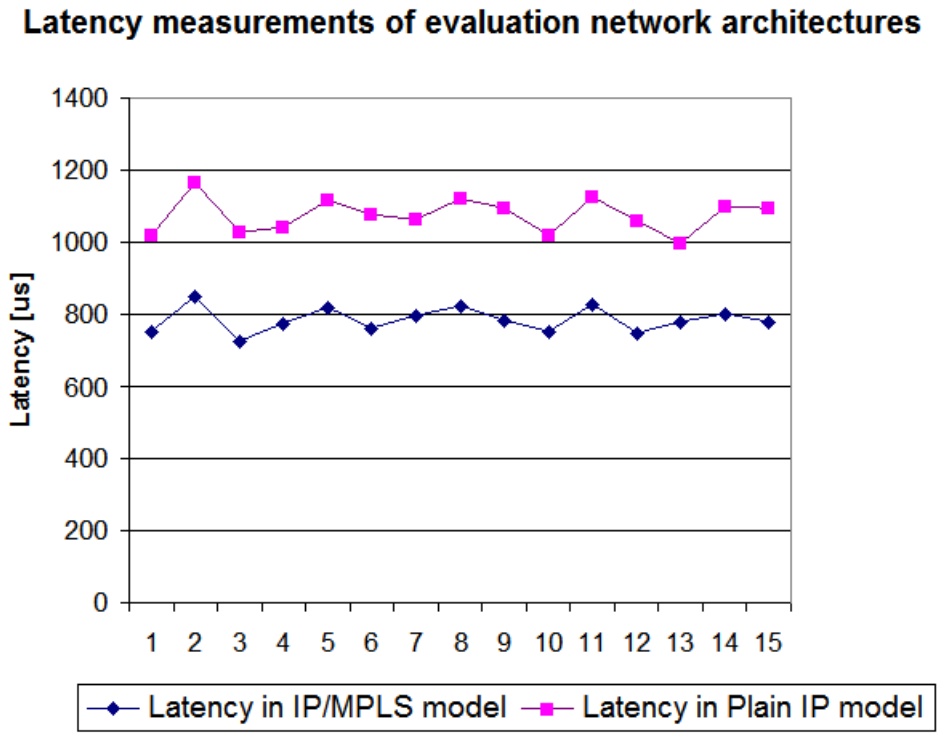

Figure 6.26 Latency measurements comparison

As it can be seen from the evaluation testing of the Plain IP and the IP/MPLS network architectures, the implementation of IP/MPLS has many benefits compared to the Plain IP model.

Conclusion

In this project the new wave technologies for design of NGN networks with End-to-End QoS were revised. This type of networks are suitable for implementation in the LTE backhaul networks.

A simplified network topology was created. Two network architectures were designed, built and evaluated with generic telecommunication equipment. One Plain IP network model was implemented and tested. One IP/MPLS architecture was designed and tested. Optimized End-to-End QoS was designed and implemented in both network models. Implemented were Layer 3 VPN services to handle the traffic from the end nodes in the IP/MPLS architecture. Both network architectures were fully functional. Verification of the applied End-to-End QoS was made, and results were obtained. Latency measurements were made for both network models.

After the whole evaluation was made, it is seen that the IP/MPLS architecture has much more benefits than the Plain IP. This is due to the opportunity for traffic-engineering in the network, which helps for better traffic management and provisioning of suitable End-to-End QoS.

The IP/MPLS architecture could be used in many mission-critical applications. Also the opportunity for easy scalability of the network is in great help in today's rapidly growing networks. This approach is suitable for real-time services, because of the low latency across the network. This way much better network productivity can be achieved.

The designed IP/MPLS network architecture is easy to scale and troubleshoot. The addition of new end devices in the network is simplified and just slight configuration changes are required.

The problem with the fast ceasing number of available ASs is evaded by using single AS number in the whole network architecture.

Because of the implemented failure mechanisms in case of link failure the impact on network flow is ceased. The traffic, which entered the network is delivered through backup routes to minimize the traffic loss, while new paths are built.

With the careful design of the applied QoS the traffic requirements of the implemented applications are served. In the IP/MPLS architecture the all services obtain the required traffic handling.

The designed IP/MPLS network model can easily be used for point-to-point connectivity services, L2/L3 business services in both centralized and distributed architectures. End-to-end MPLS solutions for the mobile backhaul applications are smoothly served.

The proposed approach provides more efficient use of network resources and reduction of the number of backhaul network nodes. It relies on single MPLS forwarding scheme, which simplifies traffic management in the network. This way of service provisioning offers simplicity to the end nodes, and depends more on the intelligent nodes in the core network. At the same time, it's implementation and maintenance are also simplified. The designed IP/MPLS can can easily be implemented in the LTE core network. And afterward can be simply managed, configured and scaled with least efforts and almost without any operational costs.

Future development

In future the network can be extended with more reliability functions. These functions include chassis clustering for Access/Aggregation devices, implementation of high availability features, implementation of LDP DoD protocol for MPLS label downstreaming on demand. Extended DiffServ services with more application specific QoS can be implemented. Layer 2 VPNs and VPLS can be included as a services in the network architecture.

Good future extension is implementation of Self-organizing features, such as self-learning – self-configuration and self-management, self-optimization - prediction of network congestion, prediction of traffic loops. To implement extensions like these in the network, a great help are algorithms for time series prediction. The self-organization can be based on both linear and non-linear neural networks. Algorithms for adaptive training of the network such as Widrow-Hoff algorithm can be in great use for process predictions in operating networks [25].

This way the designed IP/MPLS network architecture can become SON, which save operational and maintenance costs. It'll be able to make self-optimizations, based on collected data from previous network states and based on predictions. This way preventing congestion, failures and loops.

References:

1. http://www.potaroo.net/tools/asn16/
2. MR-238 MMBI White Paper on Use of MPLS in LTE, The Broadband Forum, 2010
3. Building multi-generation scalable networks with End-to-End MPLS, Juniper Networks, 2012
4. WAN and Application Optimization Solution Guide, Cisco Systems, 2008
5. MPLS Enabled Applications, Ina Minei, Julian Lucek, John Wiley & Sons, 2006
6. Traffic Classification On The Fly, Laurent Bernaille, Renata Teixeira, Ismael Akodjenou, Augustin Soule, Kave Salamatian, LIP6, Universit é Pierre et Marie Curie, Thomson Paris Lab, Paris, FRANCE, 2006
7. State of the Art in Traffic Classification: A Research Review, Min Zhang, Wolfgang John, KC Claffy, Nevil Brownlee, PAM2009, April 1-3, 2009, Seoul, Korea
8. Stefanova S. A., Some Applications of the Radial Basis Function Neural Network, Annual Journal of Electronics, International Scientific Conference "Electronics 2011", Sozopol, Bulgaria, 14-16 Sept. 2011, Volume 5, Book 1, pp. 31-33, ISSN 1313-1842.
9. http://tools.ietf.org/html/draft-leymann-mpls-seamless-mpls-03
10. http://www.ietf.org/rfc/rfc3031.txt
11. http://www.ietf.org/rfc/rfc3209.txt
12. http://www.isi.edu/rsvp/overview.html
13. http://www.ietf.org/rfc/rfc2205.txt
14. Ahmed Abdulrahim, IP/MPLS Based VPNs, Foundry Networks Inc., 2002
15. Joseph M. Soricelli, Juniper Networks Certified Internet Associate, Juniper Networks Inc. 2003
16. http://www.cisco.com/en/US/tech/tk365/technologies_white_paper09186a0080094e9e.shtml
17. http://www.ietf.org/rfc/rfc1771.txt
18. Thomas M. Thomas, Doris Pavlichek, Lawrence H. Dwyer, Rajah Chowbay, Wayne W. Downing, James Sonderegger, Juniper Networks® Reference Guide: JUNOS™ Routing, Configuration, and Architecture, Addison Wesley, 2002
19. Santiago Alvarez, QoS for IP/MPLS Networks, Cisco Press, 2005
20. http://tools.ietf.org/html/rfc791
21. MX SERIES 3D UNIVERSAL EDGE ROUTERS
    http://www.juniper.net/us/en/local/pdf/datasheets/1000208-en.pdf
22. SRX SERIES SERVICES GATEWAYS FOR THE BRANCH SRX100, SRX110, SRX210, SRX220, SRX240, SRX550, and SRX650
    http://www.juniper.net/us/en/local/pdf/datasheets/1000281-en.pdf
23. http://norvegh.com/ntools/index.php
24. T-BERD®/MTS-8000 40/100 G Transport Module
    http://www.jdsu.com/ProductLiterature/tbmts800040100G_ds_tfs_tm_ae.pdf
25. Stefanova S. A., Time Series Prediction Using Linear Neural Networks, Annual Journal of Electronics, International Scientific Conference "Electronics 2012", Sozopol, Bulgaria, 19-21 Sept. 2012, Volume 6, Book 1, pp. 160-163, ISSN 1314-0078.

Appendix A

Configurations of ACC-1 and CE-1 devices:

- CE-1 node configurations

```
## Last changed: 2013-05-23 15:28:45 UTC
version 12.2R4.5;
system {
    host-name core-edge;

     login {
        class ce-1 {
            logical-system ce-1;
            permissions all;
        }
        user ce1 {
            uid 2010;
            class ce-1;
            authentication {
                encrypted-password "$1$vFrwJfAW$gefStVpUJqUgmtGRkd.jK."; ## SECRET-DATA
            }
        }
        user ce2 {
            uid 2015;
            class ce-2;
            authentication {
                encrypted-password "$1$psevdMBW$vhGMc.twHb3.WPS0VMev1."; ## SECRET-DATA
            }
        }
        user ce3 {
            uid 2016;
            class ce-3;
            authentication {
                encrypted-password "$1$5EBqiFTy$gNo05sdP4gU5S8ZsCYAyI."; ## SECRET-DATA
            }
        }


    }

logical-systems {
    ce-1 {
        interfaces {
            lt-0/0/0 {
                unit 12 {
                    encapsulation ethernet;
                    peer-unit 21;
                    family inet {
```

```
            address 10.0.12.1/30;
        }
        family mpls;
    }
    unit 13 {
        encapsulation ethernet;
        peer-unit 31;
        family inet {
            address 10.0.13.1/30;
        }
        family mpls;
    }
}
ge-1/3/5 {
    unit 0 {
        bandwidth 2m;
        family inet {
            policer {
                input 2m;
                output 2m;
            }
            address 10.0.101.5/30;
        }
        family mpls;
    }
}
ge-1/3/6 {
    unit 0 {
        bandwidth 2m;
        family inet {
            policer {
                input 2m;
                output 2m;
            }
            address 10.0.101.1/30;
        }
        family mpls;
    }
}
ge-1/3/8 {
    unit 0 {
        family inet {
            address 10.0.201.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 1 {
        family inet {
            address 172.16.1.1/32 {
                primary;
            }
        }
    }
```

```
    }
}
protocols {
    rsvp {
        fast-reroute optimize-timer 5;
        interface all;
        interface ge-1/3/8.0 {
            bandwidth 2m;
        }
        interface lt-0/0/0.13 {
            subscription {
                ct0 25;
                ct1 25;
                ct2 25;
                ct3 25;
            }
            bandwidth 1g;
        }
    }
    mpls {
        diffserv-te {
            bandwidth-model extended-mam;
        }
        interface all;
    }
    bgp {
        local-address 172.16.1.1;
        group internal {
            type internal;
            neighbor 172.16.1.2;
            neighbor 172.16.1.3;
            neighbor 172.16.1.4;
            neighbor 172.16.1.5;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
        }
    }
    ldp {
        interface lo0.1;
    }
}
policy-options {
    policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
}
routing-options {
    autonomous-system 65001;
    forwarding-table {
```

```
                export lb;
            }
        }
    }
chassis {
    fpc 0 {
        pic 0 {
            tunnel-services {
                bandwidth 10g;
            }
        }
    }
}
interfaces {
    ge-1/2/1 {
        vlan-tagging;
    }
    ge-1/2/2 {
        per-unit-scheduler;
        vlan-tagging;
    }
    ge-1/2/3 {
        per-unit-scheduler;
        vlan-tagging;
    }
    ge-1/3/5 {
        speed auto;
        gigether-options {
            auto-negotiation;
        }
    }
    ge-1/3/6 {
        speed 100m;
    }
    ge-1/3/7 {
        speed 100m;
    }
    ge-1/3/8 {
        speed 100m;
    }
    ge-1/3/9 {
        speed 100m;
    }
    fxp0 {
        unit 0 {
            family inet {
                address 193.178.153.85/24;
            }
        }
    }
    lo0 {
        inactive: unit 1 {
            inactive: family inet {
                address 172.16.1.1/32;
            }
```

```
        }
        unit 2 {
            family inet {
                address 172.16.1.2/32;
            }
        }
        unit 3 {
            family inet {
                address 172.16.1.3/32;
            }
        }
    }
}
routing-options {
    forwarding-table {
        export lb;
    }
}
protocols {
    rsvp {
        interface all;
    }
}
policy-options {
    policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
}
inactive: class-of-service {
    forwarding-policy {
        next-hop-map ce2-acc1 {
            forwarding-class expedited-forwarding {
                lsp-next-hop ce2-to-access1-voice;
            }
        }
    }
    interfaces {
        lt-0/0/0 {
            unit * {
                scheduler-map access;
            }
        }
        ge-1/3/* {
            scheduler-map access;
            unit * {
                shaping-rate 2m;
                input-shaping-rate 2m;
                classifiers {
                    inet-precedence default;
                }
                rewrite-rules {
                    inet-precedence default;
                }
```

```
            }
        }
    }
    scheduler-maps {
        access {
            forwarding-class network-control scheduler nc;
            forwarding-class best-effort scheduler data;
            forwarding-class expedited-forwarding scheduler voice;
        }
    }
    schedulers {
        voice {
            transmit-rate percent 50;
            priority medium-high;
        }
        data {
            transmit-rate {
                remainder;
            }
            priority low;
        }
        nc {
            transmit-rate percent 5;
            priority strict-high;
        }
    }
}
firewall {
    policer 2m {
        if-exceeding {
            bandwidth-limit 2m;
            burst-size-limit 625k;
        }
        then discard;
    }
}
```

- ACC-1 node configurations

```
root@acc-1# show
## Last changed: 2013-05-23 14:40:10 UTC
version 11.2R6.3;
system {
    host-name acc-1;
    root-authentication {
        encrypted-password "$1$KJqSFm76$6ogldo8RsymxdWF1QtTYv1"; ## SECRET-DATA
    }
    services {
        ssh {
            root-login allow;
        }
        telnet;
        xnm-clear-text;
```

```
        web-management {
            http {
                interface vlan.0;
            }
            https {
                system-generated-certificate;
                interface vlan.0;
            }
        }
    }
    syslog {
        archive size 100k files 3;
        user * {
            any emergency;
        }
        file messages {
            any critical;
            authorization info;
        }
        file interactive-commands {
            interactive-commands error;
        }
    }
    max-configurations-on-flash 5;
    max-configuration-rollbacks 5;
    license {
        autoupdate {
            url https://ae1.juniper.net/junos/key_retrieval;
        }
    }
}
interfaces {
    traceoptions {
        file interfaces;
        flag all;
    }
    ge-0/0/0 {
        unit 0 {
            family inet {
                address 193.178.153.76/24;
            }
        }
    }
    fe-0/0/2 {
        fastether-options {
            auto-negotiation;
        }
        unit 0 {
            bandwidth 2m;
            family inet {
                address 10.0.101.2/30;
            }
            family mpls;
        }
    }
```

```
    fe-0/0/3 {
        unit 0 {
            bandwidth 2m;
            family inet {
                address 10.0.102.2/30;
            }
            family mpls;
        }
    }
    fe-0/0/4 {
        unit 0 {
            bandwidth 2m;
            family inet {
                address 10.0.101.6/30;
            }
            family mpls;
        }
    }
    fe-0/0/5 {
        unit 0 {
            family inet {
                address 192.168.1.1/24;
            }
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 172.16.1.4/32;
            }
        }
    }
}
forwarding-options {
    hash-key {
        family inet {
            layer-4;
        }
    }
}
routing-options {
    router-id 172.16.1.4;
    autonomous-system 65001;
    forwarding-table {
        export lb;
    }
}
protocols {
    rsvp {
        fast-reroute optimize-timer 5;
        interface fe-0/0/4.0 {
            bandwidth 2m;
        }
        interface fe-0/0/3.0 {
            bandwidth 2m;
```

```
    }
    interface fe-0/0/2.0 {
        bandwidth 2m;
    }
}
mpls {
    inactive: diffserv-te {
        bandwidth-model extended-mam;
    }
    statistics {
        auto-bandwidth;
    }
    label-switched-path access1-to-ce2-voice {
        to 172.16.1.2;
        priority 4 0;
        adaptive;
        fast-reroute;
        auto-bandwidth {
            minimum-bandwidth 1m;
            maximum-bandwidth 2m;
        }
    }
    label-switched-path access1-to-ce2-data {
        to 172.16.1.2;
        priority 7 5;
        adaptive;
        fast-reroute;
        auto-bandwidth {
            minimum-bandwidth 100k;
            maximum-bandwidth 2m;
        }
    }
    path acc1-ce2 {
        10.0.102.1 strict;
    }
    interface fe-0/0/2.0;
    interface fe-0/0/3.0;
    interface fe-0/0/4.0;
}
bgp {
    local-address 172.16.1.4;
    group internal {
        type internal;
        neighbor 172.16.1.1;
        neighbor 172.16.1.2;
        neighbor 172.16.1.3;
        neighbor 172.16.1.5;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface ge-0/0/0.0 {
            disable;
```

```
            }
        }
    }
    ldp {
        interface lo0.1;
    }
}
policy-options {
    policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
}
class-of-service {
    forwarding-policy {
        next-hop-map acc1-ce1 {
            forwarding-class expedited-forwarding {
                lsp-next-hop access1-to-ce2-voice;
            }
        }
    }
    interfaces {
        fe-0/0/2 {
            scheduler-map access;
        }
        fe-0/0/3 {
            scheduler-map access;
        }
        fe-0/0/4 {
            scheduler-map access;
        }
    }
    scheduler-maps {
        access {
            forwarding-class network-control scheduler nc;
            forwarding-class best-effort scheduler data;
            forwarding-class expedited-forwarding scheduler voice;
        }
    }
    schedulers {
        voice {
            transmit-rate 1m;
            priority medium-high;
        }
        data {
            transmit-rate {
                remainder;
            }
            priority low;
        }
        nc {
            transmit-rate percent 5;
            priority strict-high;
        }
```

```
    }
}
security {
    forwarding-options {
        family {
            mpls {
                mode packet-based;
            }
        }
    }
}
routing-instances {
    acc1 {
        description acc1-l3vpn;
        instance-type vrf;
        interface fe-0/0/5.0;
        route-distinguisher 172.16.1.4:100;
        vrf-target target:65001:100;
        vrf-table-label;
    }
}
```