MESCAL

Management of End-to-end Quality of Service Across the Internet at Large

IST-2001-37961

D3.1: Specification of Test Campaigns and Experimentation Plans for Performance Analysis and Prototype Validation

Document Identifier: MESCAL/WP3/ALGO/D3.1/Final

Deliverable Type: Report **Deliverable Nature:** Public	**Contractual Date:** 31 May 2004 **Actual Date:** 22 September 2004

Editor: Eleni Mykoniati, Algonet S.A.

Authors:
FTR&D: P.Morand, M.Boucadair
TRT: H.Asgari, R.Egan
UCL: J.Griem, D.Griffin, J.Spencer
UniS: M.Howarth, R.Ho, S.Georgoulas, N.Wang, G.Pavlou
Algo: D.Makris, D.Manikis, K.Kavidopoulos, T.Damilatis, M.Megaloikonomou

Abstract: This document is a key deliverable of the MESCAL project, and is the result of activity AC3.1. It specifies the tests validating the algorithms and protocols built to enable inter-domain Quality of Service (QoS) delivery across the Internet. The performance of the developed prototypes is assessed in terms of benefit/cost, scalability, stability and usability related metrics. Experimentation activities are placed in testbed-based and simulation-based environments. Time and effort plans are presented for the specified experimentation tasks and for the implementation of prototypes, testing platforms and tools.

Keywords: Testbed, Simulation, Service Management, Traffic Engineering, q-BGP, SLS

France Telecom Research and Development	FTR&D	Co-ordinator	France
Thales Research and Technology	TRT	Principal Contractor	UK
University College London	UCL	Principal Contractor	UK
The University of Surrey	UniS	Principal Contractor	UK
Algonet SA	Algo	Principal Contractor	Greece

Project funded by the European Community under the "Information Society Technology" Programme (1998-2002)

Executive Summary

The main theme of this deliverable is the specification of the experiments to be undertaken for testing aspects of the specified functionality of the MESCAL solution to the issue of inter-domain QoS delivery:

- SLS management including pSLS ordering, order handling;
- Off-line intra-domain and inter-domain TE, including multicast;
- q-BGP protocol and related route selection algorithms for QoS-based inter-domain routing;
- SLS admission control;
- Protocol and related mechanisms between PCSs (Path Computation Server) per domain for establishing MPLS-based QoS tunnels across domains.

The tests are to be executed in either the project testbed located in Caen, France, or in simulation environments extending the scope of testbed-based experimentation. The tests described in this deliverable fall under the following categories:

- Validation tests, aiming at verifying the functional validity of the specified functionality.
- Performance assessment tests, aiming at evaluating the performance of the functional aspect (protocol, algorithm, mechanism, sub-system) under test. Various kinds of performance assessment tests are considered, as appropriate for the aspect under test: benefit/cost, scalability, stability and usability assessment tests.

Test specifications are presented following a standard structure and template. Experimentation per functional aspect is rationalised in terms of objectives, controlled and uncontrolled variables influencing the behaviour of the aspect under test, metrics quantifying its performance and a description of the testing environment. Based on this analysis, a number of tests are derived and organised in a number of test suites, each including a number of elementary tests.

The plans and effort per partner for carrying out the specified tests and the required development, of both the components implementing the functional aspects under test and the testing components, are also presented.

The deliverable will be superseded by deliverable D3.2 due at the end of the project (April 2005), which will include the results of and the conclusions drawn from the undertaken tests.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

1.1 Background

MESCAL addresses the problem of IP QoS-based service delivery across different provider domains. MESCAL adopts a hop-by-hop, cascaded model for the interactions between providers, seen both at the service and network (IP) layers. Interactions at the service layer aim at the establishment of agreements for QoS traffic exchange, pSLSes in MESCAL terminology, to allow providers to expand the topological scope of their offered QoS-based services beyond the boundaries of their domains. Interactions at the IP layer are required to enable providers to find, determine and maintain suitable QoS routes for forwarding traffic in the Internet. In addition to appropriate protocols for supporting these interactions, MESCAL has specified the required service management and traffic engineering functionalities per provider domain, to gracefully and effectively meet the requirements emerging from these interactions, while optimising the utilisation of the network resources.

Driven by the different levels of QoS guarantees on packet transfer performance and bandwidth that could be provided to services – loose, statistical and hard contractual QoS guarantees – three corresponding technical solution options have been specified. As such, each solution option suits the needs of different service types, therefore targeting different customer/user segments and pertains to different levels of operations complexity and scalability. Solution option 3, which could also be viewed as an add-on feature to the other two solution options, is suitable for services requiring hard QoS guarantees but with the inherent limitation that cannot scale to the mass market (size of the Internet users/destinations). Following the aggregate philosophy of DiffServ networks, solution option 1 has been designed to provide for loose, qualitative QoS guarantees across the Internet, while solution option 2 delivers statistical guarantees (i.e. not per flow but per flow aggregates) on quantitative QoS targets, in addition to qualitative QoS guarantees. The technical targets, aspects and constraints of the three MESCAL solution options have been presented in [D1.1], while suitable protocols and algorithms are described in [D1.2].

Technical work in the MESCAL project is split over 3 work packages (WPs), and follows a phased approach: a theoretical phase followed by an experimentation-driven design and implementation phase and then an experimentation and dissemination phase. WP1 – Functional Architecture and Algorithms – specifies the inter-domain solution, per-domain architecture and related protocols and algorithms. WP2 – System Design and Implementation – develops aspects of the specified functionality subject to experimentation and required testing components. WP3 – Integration, Validation and Experimentation – sets up the experimentation infrastructure, testbeds and simulators, and conducts experiments with the purpose to validate and assert on performance of the specified functionality.

This document is produced as part of work package 3 – Integration, Validation and Experimentation.

1.2 Organisation of the Deliverable

The rest of this document is structured as follows:

Chapter 2 presents the overall experimentation approach of the project.

Part A addresses simulation-based experimentation and is organised according to the functional aspects under test; chapter 3 is on SLS management, chapter 4 on off-line IP-based intra-domain TE, chapter 5 on off-line inter-domain TE, chapter 6 on q-BGP, chapter 7 on SLS admission handling and chapter 8 on off-line intra-domain multicast TE.

Part B addresses testbed-based experimentation and is organised according to the envisaged test phases; chapters 9, 10, 11 and 12.

Appendix A provides an overview of the topology and configuration of the project testbed and Appendix B a detailed description of the testbed-based validation tests.

2 MESCAL EXPERIMENTATION FRAMEWORK

2.1 Experimentation Activities

Experimentation is an essential aspect of MESCAL work to the end of fulfilling overall project objectives. Table 1 depicts the related activities to be undertaken by the project, in terms of the:

- Functional aspect under test (with reference to the functional architecture in [D1.1] and algorithm/protocol specifications in [D1.2]).
- Type of the environment where experimentation will be undertaken.
- Category of the experiments to be carried out, each aiming at particular objectives to fulfil.

The identified experimentation activities are analysed in terms of their objectives, rationale for their undertaking, related variables, metrics and details of their execution environment in the subsequent chapters (a reference to the corresponding chapter per activity is also shown in Table 1). The following sections elaborate on the types of experimentation environment and categories of experiments considered.

Component/ Sub-system	Environment	Reference Section	Functional	Benefit/Cost	Scalability	Stability	Usability
pSLS Ordering	Simulation	3.1	x	x	x	x	
SLS Order Handling	Simulation	3.2	x		x		
SLS Invocation Handling	Simulation	7	x	x		x	
Traffic Forecast	Simulation	3.2	x		x		
Offline Inter-domain TE	Simulation	5	x	x	x	x	
Offline Intra-domain TE	Simulation	3.1	x	x	x	x	
Offline Intra-domain Multicast TE	Simulation	8	x	x	x	x	
Dynamic Inter-domain TE q-BGP	Simulation	6		x	x	x	
	Phase 2 Testbed	11.1, 11.2	x	x		x	x
Dynamic Inter-domain TE PCS	Phase 3 Testbed	12.1, 12.2	x	x			x
Data Plane	Phase 1 Testbed	10	x				

Table 1: MESCAL Experimentation Activities

2.1.1 Experimentation Environment

Experimentation activities will be carried out both in realistic and simulated network environment, as appropriate to the aspect of the MESCAL work under test and the experimentation objectives. Specifically, experiments will be undertaken:

- In a testbed comprised of Linux-based routers. Commercial routers e.g. Cisco routers could be made available if so required by experimentation. The project testbed is provided by FTR&D and is located in Caen, France.
- In simulators, which, depending on the aspect of the network environment they simulate, can be distinguished into:

  - Dynamic network operation simulation engines, simulating the dynamics of network behaviour at such a level of abstraction as appropriate to the purpose of the experiment e.g. at a packet, flow or protocol or control-plane activity levels.
  - Static network environment simulation tools, simulating the static aspects of the network environment i.e. the context in which the network is to operate; such aspects include network topology, number of supported QoS-classes, established service agreements, aggregate QoS traffic demands and QoS traffic generation patterns.

Three incremental phases of testbed-based experimentation have been identified, as discussed below:

- The objectives of the first phase are to deploy an operational testbed including several ASes exchanging BGP-based inter-domain routing information between them and exercise the notion of meta-QoS-classes within separate autonomous systems.
- The objectives of the second phase are to deploy q-BGP and the associated route selection algorithm, as specified by MESCAL, thus obtaining a prototype of the solution option 1 (loose end-to-end guarantees on multiple meta-QoS-class planes) and experiment with it.
- The objectives of the third phase are to deploy PCSs and their communication protocol above the loose solution option setup of phase 2 and experiment with the machinery for computing QoS paths across domains.

2.1.2 Experimentation Categories

As for their objectives, experimentation activities fall under the following commonly recognised categories:

- Functional validation experiments, aiming at assessing feasibility of implementation and validity of specifications.
- Performance assessment experiments, aiming at assessing the behaviour of the aspect under test in a variety of network operation and environment setups and conditions. Behaviour is assessed in terms of scalability, stability, sensitivity and yielded benefits/incurred cost; as such, corresponding experiments will be carried out. Specifically:
  - Benefit/Cost assessment experiments aim at assessing the benefits/costs that the aspect under test yields/incurs in network performance, as measured through specific metrics in a representative set of network and traffic cases.
  - Scalability assessment experiments aim at calculating and verifying the resource requirements and/or computational performance of the aspect under test as a function of various uncontrollable variables, to see if it can be used in a large scale deployment
  - Stability assessment tests verify that the aspect under test, given its specified dynamics/responsiveness, is operating in a way that drives the network to a stable state of operation, in a representative set of network and traffic cases.
  - Usability tests demonstrate that the aspect under test can operate as expected (according to its functional objectives) in terms of policy-based and/or tuning parameters upon which it may depend.

Obviously, experimentation objectives are restricted by the capabilities of the experimentation environment. As such, performance assessment experiments are rather performed in a simulated network environment, static or dynamic, while functional validity experiments fits better in a testbed environment, where network operational credentials prevail those that can possibly be acquired in a simulated network environment, hence the validity of the functionality from network operation perspectives in a realistic network environment is more evidently exhibited.

2.2 Experimentation Structure

The identified experimentation activities are specified in a clear and concise manner using a common structure/template, along the following headings:

Objectives: The question "What do we want to test?" is answered. The aspects under test (specified algorithm, protocol, mechanism) and the particular goals of experimentation are outlined. Specifically, the broad experimentation categories of functional validity and assessment of benefit/cost, scalability, stability and usability are qualified in terms of concrete objectives as appropriate to the functional aspect under test.

Performance Metrics: The metrics inherent to the particular functional aspect under test that quantify the experimentation objectives such as processing time, overhead, throughput, size of etc. are described. How these metrics can be obtained, e.g. through probes in the entity under test or through test tools, is also discussed where appropriate.

Controlled Variables: The configuration parameters of the aspect under test are outlined. The defined performance metrics will be calculated as a function of these configuration parameters.

Uncontrolled Variables: These are the parameters of the external environment where the aspect under test is to operate are defined affecting its behaviour and/or its performance. Such parameters are network topology, volume and symmetry of traffic, number of peers, contracts etc. Generators or models for creating a realistic and representative set of their values are described where appropriate.

Experimentation Environment: The platform and the setup upon which the envisaged experimentation is to be carried out are described in terms of: components of the functional architecture, experimentation platform and required test tools, their capabilities and interactions.

Test Campaigns: Specify the tests to be carried out in achieving the specified objectives. Each of the tests aims at verifying/assessing a particular aspect of the behaviour/performance of the functional aspect under test (quantified by appropriate performance metrics) in a variety of test cases (quantified by appropriate combinations of uncontrolled variables) as a function of its configuration parameters (quantified by appropriate controlled variables). Tests are aggregated in test suites according to the general category they fall in.

PART – I: SIMULATION-BASED EXPERIMENTATION

3 SLS MANAGEMENT TESTING

3.1 pSLS Ordering

3.1.1 Objectives

The role of the *pSLS Ordering* functional block (see section 4.5 of [D1.2]) is to establish the set of pSLS agreements, the most advantageous to the AS with respect to *Traffic Engineering* and business objectives. The *Binding Selection* block places a collective pSLS order which, in addition to the set of pSLSes to pursue, also contains negotiation strategies and agreement restrictions and preferences; we call this enhanced order a *Negotiation Plan*. By increasing the control on the ordering process we aim at optimising the achieved agreements, taking into account the particular ordering objectives.

The *pSLS Ordering* functional block is decomposed into three major functions to experiment upon (see Figure 2): the *Negotiation Plan Translation* function, the *Transactional Negotiation Logic* function and the *Individual Agreement Negotiation* function. Other functions of the *pSLS Ordering*, such as *Negotiation Document Manipulation* and *Reporting* functions are rather supporting, hence they are not subject of experimentation.

The *Transactional Negotiation Logic* function implements the logic for pursuing optimum collective agreements through multi-party negotiations. The *Negotiation Plan Translation* function translates the input *Negotiation Plan* into concrete operational guidelines fed to the *Transactional Negotiation Logic*. The *Individual Agreement Negotiation* function implements the negotiation engine and protocol providing the primitives to the *Transactional Negotiation Logic* to conduct negotiations within the context of a single-party negotiation session that may eventually result in the establishment of one individual agreement.

The high-level experimentation objectives for the *pSLS Ordering* functional block are:

- Functional validation of the prototype implementation;
- Verification of the convergence of the collective agreement optimisation logic;
- Assessment of the impact of environment complexity upon the scalability of the approach;
- Gaining insight on inherent benefit/cost tradeoffs of the collective agreement optimisation.

The performance metrics, the controlled and uncontrolled variables considered for the *pSLS Ordering* experimentation are listed in sections 3.1.2 and 3.1.3 respectively. Following, we outline the experimentation environment and we describe the test campaigns designed to fulfil *pSLS Ordering* experimentation objectives.

3.1.2 Performance Metrics

Performance Metrics	
processing time per function (ProcT.x)	The processing time of a particular *pSLS Ordering* function. *ProcT.Translation*, *ProcT.Logic*, *ProcT.Protocol* correspond to the processing times of *Negotiation Plan Translation*, *Transactional Negotiation Logic* and *Individual Agreement Negotiation* functions respectively.

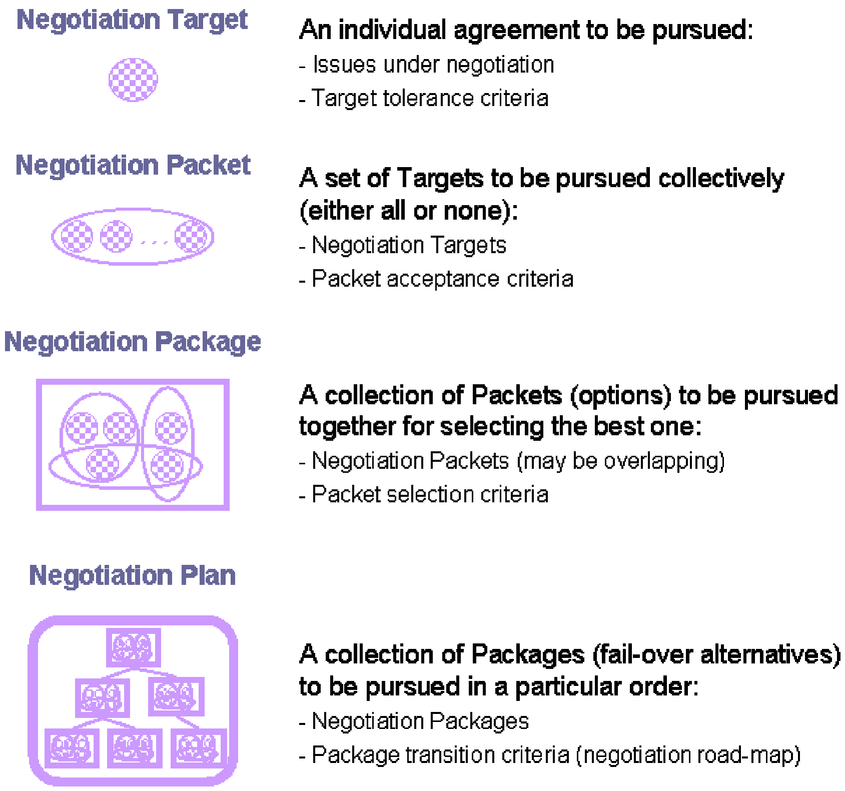
order execution time (OrdExecT)	The time elapsed between the reception of a *Negotiation Plan* and the return of an *Ordering Report*. An alternative metric may be the number of negotiation rounds.
agreement optimality (AgrOptimality)	The optimality of the achieved agreement with respect to the restrictions and preferences of the particular *Negotiation Plan*.

Table 2: pSLS Ordering Performance Metrics

3.1.3 Controlled and Uncontrolled Variables

Controlled Variables		
agreement optimisation convergence directives[1] *(Cvg)*	*maximum number of rounds (MaxRounds)*	The maximum number of negotiation rounds the *Transactional Negotiation Logic* function is permitted to undertake before concluding the execution of an order.
	maximum order execution time (OrdExecT)	The time available to *pSLS Ordering* to conclude the order execution. Approaching the expiration of *Cvg.OrdExecT*, the *Transactional Negotiation Logic* will abort agreement optimisation and consent to the best available at the time agreement.
	maximum time to respond (MaxRespondT)	The time allotted by the *Individual Agreement Negotiation* function to the peer *SLS Order Handling* negotiations server to respond within one negotiation round.

Table 3: pSLS Ordering Controlled Variables



An individual agreement to be pursued:
- Issues under negotiation
- Target tolerance criteria

A set of Targets to be pursued collectively (either all or none):
- Negotiation Targets
- Packet acceptance criteria

A collection of Packets (options) to be pursued together for selecting the best one:
- Negotiation Packets (may be overlapping)
- Packet selection criteria

A collection of Packages (fail-over alternatives) to be pursued in a particular order:
- Negotiation Packages
- Package transition criteria (negotiation road-map)

Figure 1: *Negotiation Plan* complexity

[1] The specified parameters may be used in combination, alternatively or in isolation, depending on the particular test purpose. It is apparent that leaving *Cvg.OrdExecT* unspecified and using a fixed value for *Cvg.MaxRespondT*, convergence directives reduce to specifying *Cvg.MaxRounds*.

Uncontrolled Variables		
Negotiation Plan complexity (NPCplx) (see Figure 1)	*order complexity (OrderCplx)*	The number and combinations of individual agreements (pSLSes) to collectively pursue. Each agreement is considered a *Negotiation Target*. Multiple *Negotiation Targets* to be collectively ordered form a *Negotiation Packet*. Multiple alternative *Negotiation Packets* form a *Negotiation Package*. In case of failure of a *Negotiation Package* an alternative *Negotiation Package* may be pursued following a *Negotiation Plan*. Increasingly adding levels as described above and members within each level results in increasing the *OrderCplx*. We consider the *SingleTarget*, *MultiTarget*, *MultiPacket*, *MultiPackage* ordered discrete values of *OrderCplx*, corresponding to single *Negotiation Target*, multiple *Negotiation Targets* within a single *Negotiation Packet*, multiple *Negotiation Packets* and multiple *Negotiation Packages* respectively.
	agreement criteria complexity (CriteriaCplx)	The decision to establish a collective agreement is based on restriction and preference criteria. Criteria may refer to base or aggregated across *Negotiation Targets* parameters (e.g. the cost of an individual pSLS and the SumOf the costs across pSLSes of the same *Negotiation Packet*). Furthermore parameters may be combined in arithmetic expressions, and/or in logical expressions, allowing for conditional criteria. Each of these aspects contributes to the complexity of both the translation and the execution of a *Negotiation Plan* in an incremental way. We consider the *AcceptanceBase*, *AcceptanceAggregated*, *AcceptanceCombined*, *SelectionCombined* and *Conditional* ordered discrete values of *CriteriaCplx*, corresponding to acceptance criteria on base parameters, acceptance criteria on aggregated parameters, acceptance criteria on combined parameters, acceptance and selection criteria on combined parameters, and conditional acceptance and selection criteria on combined parameters.
SLS Order Handling responses (OrdHdlR)	*availability diversity (Diversity)*	The diversity of the availability of an order denotes the amount and variety of the alternatives presented to the *Transactional Negotiation Logic* from the *SLS Order Handling* servers with respect to the original order request. We distinguish between *minimum*, *limited* and *significant* values. In *minimum* diversity, the servers will only respond with order accepted as is, or order rejected. In *limited* diversity, the servers may further respond with few alternatives close the original order. In *significant* diversity, the servers may respond with multiple alternatives and guidelines, not necessary close to the original order request.
	number of servers (Servers)	The number of *SLS Order Handling* servers involved in an order. We consider a *small* and a *large* set.
	agreement time restriction (TRestriction)	The time restrictions an *SLS Order Handling* server may pose to conclude to an agreement. The agreement optimisation logic will have to decide whether to establish the agreement or drop it counting that a better one will follow without being able to rollback,. The synchronisation of the various time restrictions among different servers affects the performance of the *Transactional Negotiation Logic* function. We consider *none*, *medium* and *tight* types of time restrictions.

Table 4: pSLS Ordering Uncontrolled Variables

To facilitate experimentation we focus to a representative set of test configuration options for *NPCplx* and *OrdHdlR* uncontrolled variables (see Table 5 and Table 6 respectively).

Negotiation Plan complexity (*NPCplx*) test configuration option	***OrderCplx***	***CriteriaCplx***
#1 Base	SingleTarget	AcceptanceBase
#2 Aggregated	MultiTarget	AcceptanceAggregated
#3 AcceptanceCombined	MultiTarget	AcceptanceCombined
#4 SelectionCombined	MultiPacket	SelectionCombined
#5 Conditional	MultiPacket	Conditional
#6 MultiPackage	MultiPackage	Conditional

Table 5: *Negotiation Plan Complexity* Test Configuration Options

SLS Order Handling responses (*OrdHdlR*) test configuration option	***Diversity***	***Servers***	***TRestriction***
#1 MinimumBase	minimum	small	none
#2 MinimumLargeTimeRestricted[Medium/Tight]	minimum	large	medium/tight
#3 LimitedBase	limited	small	none
#4 LimitedLargeTimeRestricted[Medium/Tight]	limited	large	medium/tight
#5 SignificantBase	significant	small	none
#6 SignificantLargeTimeRestricted[Medium/Tight]	significant	large	medium/tight

Table 6: *SLS Order Handling Responses* Test Configuration Options

3.1.4 Experimentation Environment

The test platform is composed by the *pSLS Ordering* prototype, a reduced version of the *SLS Order Handling* prototype, the *Negotiation Plan Generator* acting on behalf of *Binding Selection* block and the *SLS Order Handling Response Generator* testing tools (see Figure 2).

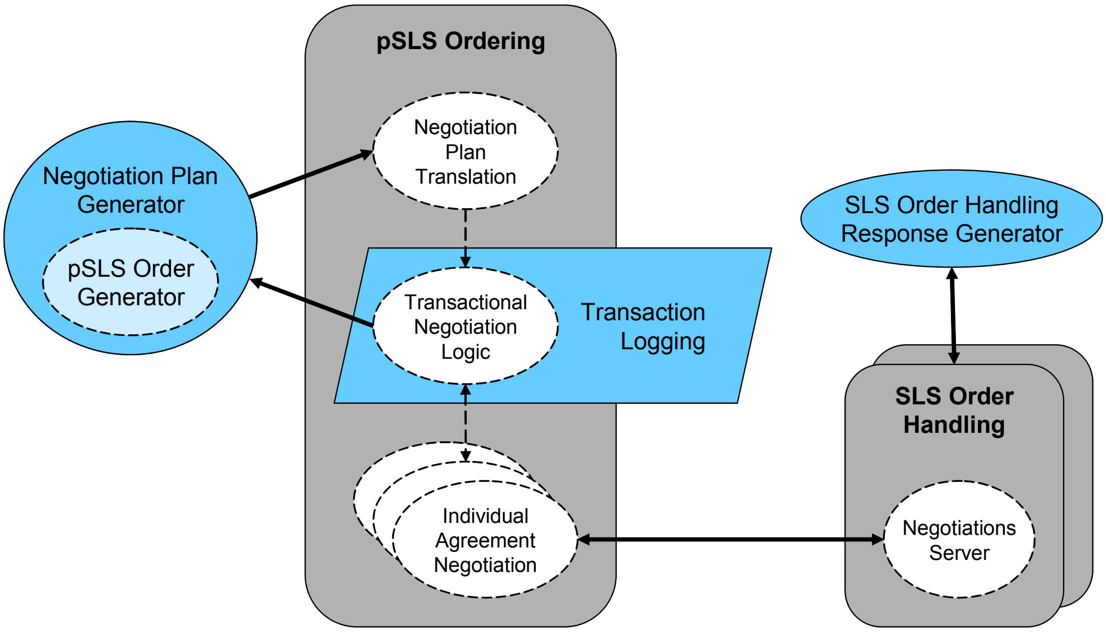

pSLS Ordering
Negotiation Plan Translation
Negotiation Plan Generator
pSLS Order Generator
Transactional Negotiation Logic
Transaction Logging
Individual Agreement Negotiation
SLS Order Handling Response Generator
SLS Order Handling
Negotiations Server

Figure 2: pSLS Ordering Experimentation Environment

An experiment corresponds to the execution of one order, expressed in the *Negotiation Plan*. However, multiple dynamic interactions with a number of *SLS Order Handling* servers may take place for the completion of the order execution and the conclusion of the experiment.

The ***Negotiation Plan Generator*** produces and forwards *Negotiation Plans* to the *pSLS Ordering*. It can be configured to cover orders of increasing complexity for performance assessment testing (*NPCplx*). Besides the negotiation and agreement establishment instructions targeting *pSLS Ordering*, a *Negotiation Plan* also contains the pSLSes to order, handled by the *SLS Order Handling* component. *Negotiation Plan Generator* is therefore enhanced with the *pSLS Order Generator* function to produce all possible pSLS orders and is used in *SLS Order Handling* experimentation (see section 3.2.4).

The *SLS Order Handling* (see section 3.2) prototype is reduced and contains only its *Negotiations Server* engine which, instead of the Admission Logic, it is now controlled by the ***SLS Order Handling Response Generator***. While in the real world, the involved *SLS Order Handling* servers act independently, in order to control their collective effect on *pSLS Ordering*, we need to be able to control the overall server responses. Therefore, there is one instance of the *SLS Order Handling Response Generator* controlling response generation for all involved *SLS Order Handling* servers at the same time. The *SLS Order Handling Response Generator* can instruct the *SLS Order Handling* servers to respond with all possible protocol messages for *Individual Agreement Negotiation* functional testing, even invalid ones. It can be configured to produce responses of increasing complexity to deal with at the *pSLS Ordering* side, increasing *OrdHdlR.Diversity*, different *OrdHdlR.TRestriction* settings, etc.

In order to test the *Transactional Negotiation Logic* function, appropriate ***Transaction Logging*** must be added, so that the logical steps followed towards collective agreement optimisation can be tracked down.

The software platform that will be used for *pSLS Ordering* prototype and testing tools implementation is Java based.

3.1.5 Test Campaigns

The tests undertaken for *pSLS Ordering* are organised under the test suites outlined in Table 7 and described in detail in Table 8. Controlled or uncontrolled variables left unspecified in a test description are set to appropriate fixed values so that they have no impact upon the subject under testing.

Test Suite Id	Objective
pSLSOrd/Funct/Logic	Functional validation of the *convergence directives* implementation in the *Transactional Negotiation Logic* function.
pSLSOrd/Funct/Protocol	Functional validation of the *Individual Agreement Negotiation* function.
pSLSOrd/Scal/NPCplx	Functional validation of *Negotiation Plan Translation* and *Transactional Negotiation Logic* functions. Assessment of the impact of *Negotiation Plan Complexity* on the performance of *Negotiation Plan Translation* and *Transactional Negotiation Logic* functions.
pSLSOrd/Scal/RespCplx	Functional validation of the *Transactional Negotiation Logic* function. Assessment of the impact of *SLS Order Handling responses* on the performance of *Transactional Negotiation Logic* function.
pSLSOrd/BC	Study the effect of bounding *OrdExecT* on *AgrOptimality* as a function of *Negotiation Plan Complexity* and *SLS Order Handling responses*. Verify convergence of the collective agreement optimisation logic of the *Transactional Negotiation Logic* function.

Table 7: pSLS Ordering Test Suites

Test Id	Purpose	Description
pSLSOrd/Funct/Logic/1	Ensure *Cvg.OrdExecT* setting works properly.	Controlled variables: *Cvg.OrdExecT* set to a fixed value Uncontrolled variables: *NPCplx* fixed to *MultiPackage* *OrdHdlR* fixed to *SignificantBase* Test: Check *OrdExecT* does not exceed *Cvg.OrdExecT*.
pSLSOrd/Funct/Logic/2	Ensure *Cvg.MaxRounds* and *Cvg.MaxRespondT* settings work properly.	Controlled variables: *Cvg.MaxRounds* set to a fixed value *Cvg.MaxRespondT* set to a fixed value Uncontrolled variables: *NPCplx* fixed to *MultiPackage* *OrdHdlR* fixed to *SignificantBase* Test: Check no more than *Cvg.MaxRounds* negotiation rounds take place and, leaving out processing time, no more than (*Cvg.MaxRounds* * *Cvg.MaxRespondT*) time elapses till the negotiation completion.
pSLSOrd/Funct/Protocol	Verify FSM implementation is according to specifications.	Uncontrolled variables: *NPCplx* fixed to *Base* *OrdHdlR.Diversity* variable covering all protocol message combinations, invalid ones too *OrdHdlR.Servers* fixed to *small* *OrdHdlR.TRestriction* fixed to *none* Test: Check the negotiation protocol embedded in *Individual Agreement Negotiation* function works as specified.
pSLSOrd/Scal/NPCplx/1-6	Ensure *Negotiation Plan Translation* and *Transactional Negotiation Logic* functions produce expected output for various *NPCplx* values. Assess the impact of *NPCplx* on processing time.	Performance metrics: *ProcT.Translation*, *ProcT.Logic* Uncontrolled variables: *NPCplx* varying from #1 to #6 options *OrdHdlR* fixed to *MinimumBase* Test: Check the translated *Negotiation Plan* produced by *Negotiation Plan Translation* and the order requests logged per negotiation round against specifications. Gather *ProcT.Translation* and *ProcT.Logic* for scalability analysis.
pSLSOrd/Scal/RespCplx/1-6	Ensure *Transactional Negotiation Logic* function produce expected output for various *OrdHdlR* values. Assess the impact of *OrdHdlR* on processing time.	Performance metrics: *ProcT.Logic* Uncontrolled variables: *NPCplx* fixed to *MultiPackage* *OrdHdlR* varying from #1 to #6 options Test: Check the order requests logged per negotiation round against specifications. Gather *ProcT.Logic* for scalability analysis.

pSLSOrd/BC/1-36	Study the *AgrOptimality* curve with respect to *OrdExecT* as a function of *NPCplx* and *OrdHdlR*. Verify *OrdExecT* is finite even with unspecified *Cvg convergence directives*.	Performance metrics: *OrdExecT*, *AgrOptimality* Controlled variables: *Cvg.MaxRounds* set to infinite *Cvg.OrdExecT* set to infinite *Cvg.MaxRespondT* set to a fixed value Uncontrolled variables: *NPCplx* varying from #1 to #6 options *OrdHdlR* varying from #1 to #6 options Test: Check *OrdExecT* is finite. Gather *OrdExecT* and *AgrOptimality* for benefit/cost analysis.

Table 8: pSLS Ordering Tests

3.2 SLS Order Handling

3.2.1 Objectives

The *SLS Order Handling* functional block (see section 4.4 of [D1.2]) conducts negotiations with *pSLS Ordering* so that the best matching between service requests and available resources is achieved. To this end, it uses the *Demand Aggregation and Derivation* function of the *Traffic Forecast* functional block (see section 5.2 in [D1.2]). In addition to pure *SLS Order Handling* this section includes combined *SLS Order Handling* and *Traffic Forecast* tests.

The *SLS Order Handling* is decomposed into four major functions (see Figure 3): the *Negotiations Server*, the *SLS Translation*, the *Admission Logic* and the *SLS Establishment* functions, which together with the *Demand Aggregation and Derivation* function of the *Traffic Forecast* component, constitute the experimentation focus in this section.

The *Negotiations Server* function conducts negotiations for all pending SLS orders from different customers in parallel, using the underlying negotiation protocol. The *SLS Translation* function translates and maps the SLSs contained in an SLS order into their network view; further it performs validity checks against already established SLSs, primarily for ensuring uniqueness of customer/users identification. The *Admission Logic* function calls the *Demand Aggregation and Derivation* function of the *Traffic Forecast* component. The *Demand Aggregation and Derivation* function calculates the aggregated forecasted traffic in the order of *Traffic Trunks* contained in the *Traffic Matrices*, over the existing plus the requested population of SLSs and based on the *Traffic Forecast Parameters* produced by the *Traffic Analysis* functional block. The *Admission Logic* then compares the results against the available resources provided in the *Resource Availability Matrices* and decides accordingly the acceptance of the SLS. The accepted SLSs are established via the *SLS Establishment* function responsible for maintaining the *SLSs* repository and for communicating the relevant information to other system components that need to be updated.

The high-level experimentation objectives for the above mentioned functions are:

- Functional validation of the prototype implementation;
- Assessment of the impact of environment complexity upon the scalability of the approach.

The considered performance metrics, the controlled and uncontrolled variables are listed in sections 3.2.2 and 3.2.3 respectively. Following, we outline the experimentation environment and we describe the test campaigns designed to fulfil the experimentation objectives.

3.2.2 Performance Metrics

Performance Metrics	
processing time per function (ProcT.x)	The processing time of a particular function. *ProcT.Translation*, *ProcT.Forecast* and *ProcT.Admission* correspond to the processing times of the *SLS Translation*, *Demand Aggregation & Derivation* and *Admission Logic* functions respectively.

Table 9: SLS Order Handling Performance Metrics

3.2.3 Controlled and Uncontrolled Variables

Controlled Variables	
Satisfaction Level (SL)	The *Satisfaction Level* operational parameter of the *Admission Control* function (see section 4.4.3.1 of [D1.2]). Its permissible value range is [-1,1], however, for experimentation purpose we consider only the following three values: #1 *NoGuarantees* (-1) #2 *AlmostSatisfied* (0) #3 *FullySatisfied* (+1)

Table 10: SLS Order Handling Controlled Variables

Uncontrolled Variables		
Resource Availability (Availability)		The availability of network resources as provided in the *Resource Availability Matrices*. We consider *low* and *unlimited* settings.
Traffic Matrices size[2] *(TMSize)*	*number of lQCs (lQCs)*	The number of lQCs is one dimension of the internal Traffic Matrix. We only consider the fixed set of three lQCs: *Premium*, *Better-Than-Best-Effort* and *Best-Effort* lQCs.
	number of external interfaces (ExtInterfaces)	The number of external interfaces is the second dimension of the internal Traffic Matrix.
	number of oQCs (oQCs)	The number of oQCs is one dimension of the external Traffic Matrix. We only consider the fixed set of three oQCs: *Premium*, *Better-Than-Best-Effort* and *Best-Effort* oQCs.
	number of external destination prefixes (ExtDestPrefixes)	The number of destination prefixes outside the AS is the second dimension of the external Traffic Matrix. The external destination prefixes result from the established pSLSes, hence their number is analogous to the number of the established pSLSes.
Traffic Forecast Parameters	*number of service classes (SrvClasses)*	Traffic forecast parameters such as *Multiplexing Factor* (MF) and *Aggregation Weight* (AW) refer to a *service class*. A service class groups a homogeneous set of traffic flows allowing for aggregation under the assumed service usage and traffic source patterns. The greater the number of service classes, the more granular the classification of traffic sources, hence the more homogenous the set of traffic flows and the more accurate the traffic forecast result. We consider just *one* or *many* service classes.

[2] The Resource Availability Matrices (see section 5.1.2.3 of [D1.2]) have the same size as the Traffic Matrices (see section 5.1.2.2 of [D1.2]).

SLS Order Requests	*request diversity (Diversity)*	The diversity of an order request denotes the variety of the proposals presented to the *Negotiations Server* for a particular SLS order. We consider *minimum* and *significant* settings. In *minimum*, a customer will present only an initial proposal, while in *significant* diversity, a customer may present additional proposals irrespectively of the server responses.
	number of active customers (Customers)	The number of active ordering customers. We consider just *one* or *many* customers.
SLS Orders	*number of ordered SLSs (SLSsOrdered)*	The total number of non alternative SLSs contained in every SLS order placed during the experiment. Assuming only valid SLSs and *Availability* set to *infinite* then at the end of experiment there will be *SLSsOrdered* number of established SLSs.
	service types (SrvTypes)	The type of the service the SLS order belongs to. The supported service types are: #1 Internet access at loose QoS #2 Loose QoS tunnels in the Internet #3 Traffic inter-exchange at a loose QoS #4 Loose QoS tunnel extension #5 Internet access at a statistically guaranteed QoS #6 Statistically guaranteed QoS tunnels in the Internet The *all* setting signifies all supported service types may be used.

Table 11: SLS Order Handling Uncontrolled Variables

To facilitate experimentation we focus to a representative set of test configuration options for *TMSize* variable (see Table 12).

Traffic Matrices size (*TMsize*) test configuration option	***ExtInterfaces***	***ExtDestPrefixes***
#1 Small	small	small
#2 Medium	medium	medium
#3 LargeAtDestinations	medium	large
#4 LargetAtEdges	large	medium
#5 LargeAll	large	large

Table 12: *Traffic Matrices Size* Test Configuration Options

3.2.4 Experimentation Environment

The test platform comprises the *SLS Order Handling*, the *pSLS Ordering* and the *Traffic Forecast* prototypes, the *Negotiation Plan Generator* and *Service Management Data Generator* testing tools and the accompanying data repositories (see Figure 3).

With no loss of generality we assume no change of the *Resource Provisioning Cycles* (see section 5.3.2 of [D1.2]) throughout one experiment. Consequently, in the course of one experiment, the *Resource Availability Matrices* are static. pSLS orders with destination prefixes not included in the given *Resource Availability Matrices* will fail validity checks and be rejected. Therefore, the destination prefixes of the generated pSLS orders will be limited to those already existing in the *Resource Availability Matrices*, meaning that the *ExtDestPrefixes* will be static too. Assuming fixed *lQCs* and *oQCs* and static number of external interfaces *ExtInterfaces* throughout the experiment, the size of *Traffic Matrices TMSize* becomes static. Finally, the *Traffic Analysis* component is not active in the MESCAL system, hence the *Traffic Forecast Parameters* are also considered static. These static parameters are configured in a consistent way by the *Service Management Data Generator* testing tool.

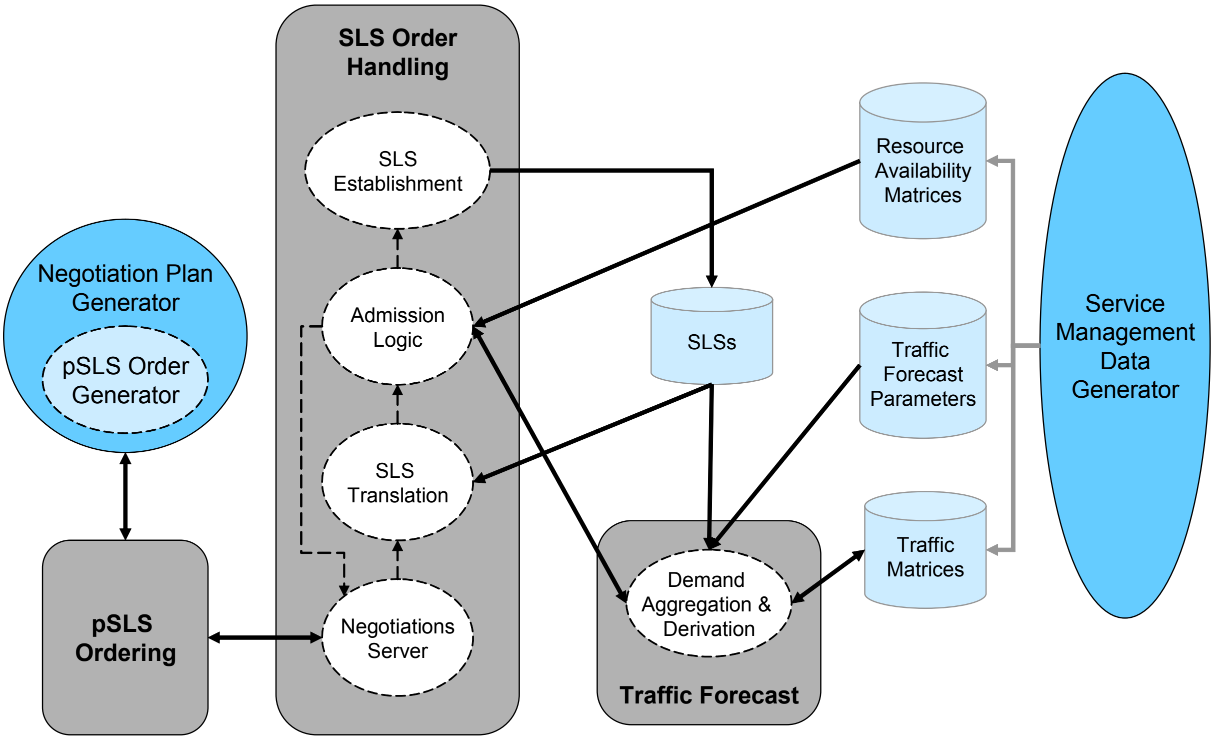


Figure 3: SLS Order Handling Experimentation Environment

The ***Service Management Data Generator*** tool can be configured in terms of *lQCs*, *ExtInterfaces*, *oQCs* and *ExtDestPrefixes*, in terms of *Availability*, of *SrvClasses* and associated traffic forecast parameters. The traffic forecast parameters are stored in the corresponding repository. A set of *ExtDestPrefixes* is created, each associated with one of the created *ExtInterfaces*. Corresponding entries are created in the *Traffic Matrices* and the *Resource Availability Matrices* for each QC. Assuming no SLS is established at the beginning of the experiment forecasted traffic will be set to zero. Resource Availability will be set to either infinite or a low value following *Availability* setting.

An experiment corresponds to the execution, for a given static setup, of a series of SLS orders wrapped in *Negotiation Plans* and fed to the *pSLS Ordering*, thereafter communicated and processed into the *SLS Order Handling* component. The series of SLS orders constitute the runtime variables and are produced by the *Negotiation Plan Generator* tool.

The ***Negotiation Plan Generator*** tool (see section 3.1.4) can be configured in terms of *SrvTypes*, *SLSsOrdered*, *Customers* and *Diversity*. A uniform portion of *SLSsOrdered* number of SLSs is allocated to each of the given *SrvTypes* and the corresponding pSLS orders are produced. For the pSLS orders to be valid the destination prefixes are chosen among the set created during the static experiment setup and the flow identification clauses are populated with unique values.

Resulted pSLS orders are wrapped in *Negotiation Plans*. A basic *Negotiation Plan* will only contain one pSLS Order without any acceptance criteria so that an agreement is always established when the *SLS Order Handling* accepts the initial proposal. A more sophisticated *Negotiation Plan* will contain many pSLS Orders combined in alternative *Negotiation Packages*, resulting in increased request diversity depending on the value of the *Diversity* variable. Finally, if *Customers* is more than just *one*, multiple instances of the *pSLS Ordering* component are activated and the produced *Negotiation Plans* are distributed among them.

The software platform that will be used for *SLS Order Handling* prototype and testing tools implementation is Java based.

3.2.5 Test Campaigns

The tests undertaken are organised under the test suites outlined in Table 13 and described in detail in Table 14. Parameters left unspecified in a test description are set to appropriate fixed values so that they have no impact upon the subject under testing.

Test Suite Id	Objective
SLSOrderH/Funct/NServer	Functional validation of the implementation of the *Negotiations Server* function.
SLSOrderH/Funct/Translation	Functional validation of the service type dependent functions, namely *SLS Translation*, *Demand Aggregation and Derivation* and *SLS Establishment* functions.
SLSOrderH/Funct/Admission	Functional validation of the *Admission Control* function.
SLSOrderH/Scal/SLSs	Assessment of the impact of *SLSs* population on the performance of *SLS Translation* and *Traffic Forecast* functions.
SLSOrderH/Scal/TMSize	Assessment of the impact of *Traffic Matrices Size* on the performance of the *Traffic Forecast* and *Admission Control* functions.

Table 13: SLS Order Handling Test Suites

Test Id	Purpose	Description
SLSOrderH/Funct/NServer	Verify handling of multiple parallel SLS orders. Verify the FSM implementation is according to specifications.	Runtime parameters: *Diversity* fixed to *significant* covering all protocol message combinations, invalid ones too *Customers* fixed to *many* Test: Check the negotiation protocol embedded in *Negotiations Server* function works as specified.
SLSOrderH/Funct/Translation/1	Verify the implementation of SLS translation and validation.	Runtime parameters: *SrvTypes* fixed to *all* Ordered pSLSes flow identification clauses configured to overlap and cause validation failure Test: Check SLSs are translated correctly to the corresponding network view data structures. Ensure validity checks indeed fail.
SLSOrderH/Funct/Translation/2	Verify the implementation of SLS validation and SLS establishment. Verify *Demand Aggregation and Derivation* calculations.	Static parameters: *SrvClasses* fixed to *many* Runtime parameters: *SrvTypes* fixed to *all* Test: Ensure validity checks indeed succeed. Check *Demand Aggregation and Derivation* calculations are in accordance to the provided traffic parameters. Check SLS establishment produces information as expected by the *SLS Invocation Handling* and the *Dynamic Inter-domain TE* components.

SLSOrderH/Funct/Admission/1-6	Verify the implementation of the admission control algorithm is according to specifications.	<u>Static parameters</u>: *Availability* varying between *low* and *unlimited* <u>Runtime parameters</u>: *SL* varying between *NoGuarantees*, *AlmostSatisfied* and *FullySatisfied* values <u>Test</u>: Check requested SLSs are admitted within the availability buffer as resized by the *SL*, while SLSs exceeding that buffer are rejected.
SLSOrderH/Scal/1-5	Assess the impact of *SLSs* and *TMSize* on processing time.	<u>Performance metrics</u>: *ProcT.Translation*, *ProcT.Forecast*, *ProcT.Admission* <u>Static parameters</u>: *TMSize* varying from #1 to #5 options <u>Runtime parameters</u>: *SLSsOrdered* fixed to a large value <u>Test</u>: Gather *ProcT.Translation*, *ProcT.Forecast* and *ProcT.Admission* for scalability analysis.

Table 14: SLS Order Handling Tests

4 OFF-LINE INTRA-DOMAIN IP-TE TESTING

4.1 Objectives

This section presents a detailed description of the test campaigns that have been planned for the Intra-domain Traffic Engineering component described in the MESCAL functional architecture. The campaigns focus predominantly on the *Resource Optimisation* block contained within Intra-domain Traffic Engineering. *Resource Optimisation* contains the essential link weight based IP traffic engineering functionality. In contrast, functionality contained within the *Resource Reconfiguration Scheduler* block is concerned with the efficient implementation of results computed by *Resource Optimisation* and is therefore of secondary concern. The test campaigns have three overall objectives:

1. Functional Validation
2. Algorithm Performance Measurement and Optimisation
3. Algorithm Efficiency Measurement and Optimisation

Functional validation is concerned with ensuring the correct function of the algorithm as well as the simulation software. The outcome of this testing phase will be an assurance in the reliability of later test results. Important tests of this phase include making sure that hop count limits as defined in the l-QC repository are honoured by the algorithm and that bandwidth limits are not exceeded by accommodating more demands than can be supported by the configuration.

Algorithm Performance Measurement and Optimisation is the essential phase during which the traffic engineering capabilities of the algorithm are measured. Different versions of the optimisation algorithm will be compared, as well as OSPF routing cases and more flexible routing cases [tequila] (which would require more flexible packet forwarding techniques like MPLS for implementation in the network). Comparisons will be made between the algorithms to how many traffic demands can be accommodated, as well as the solutions flexibility towards changes in topology and demand. This phase is recursive, as results gained during the phase will be used to further improve the algorithms.

Algorithm Efficiency Measurement and Optimisation is the phase responsible for investigating the algorithms computational properties. Tests will be performed on the various candidate versions of algorithms from the second phase. The test campaigns will focus on resource consumption (i.e. processing requirements), scalability with topology as well as traffic demands and convergence time.

4.2 Performance Metrics

This section lists the metrics that are used to quantify each experiment's objectives and will be measured during the simulation. We distinguish between metrics used during the algorithm performance simulations and those used during the the algorithm efficiency simulations.

Performance Metrics		
Algorithm Performance Metrics	*Average link utilisation*	The average utilisation of all links within the network. Obtained through the algorithm software.
	Standard deviation of link utilisation	The standard deviation from the average link load across the network, obtained as above from the algorithm software.
	Network traffic load (demands)	The load created on the network through the traffic demands passed with the iTM, should be varied between 40-110% but has to be calibrated since the load is an indirect quantity created by demands in the iTM.
	Hop count budget utilisation	An indicative for delay across the network per routing plane. Obtained directly from the algorithm software.
Algorithm Efficiency Metrics	*Algorithm execution time*	Time for the algorithm to complete its run and find a solution. Note that this time depends on settings for cut-off point (depends on many setting, i.e. maximum number of iterations).
	Number of nodes in topology	Nodes in the topology, part of the input data.
	Number of links in topology	Links in the topology, part of input data.
	Cost function	The IP traffic engineering cost function designed to drive towards a good solution. Part of the algorithm software.
	Algorithm iteration	Algorithm iteration counter, part of the algorithm software.

Table 15: Off-line Intra-domain IP-TE Performance Metrics

4.3 Controlled and Uncontrolled Variables

Controlled variables are the configuration parameters that modify the behaviour of the simulation software.

Controlled Variables	
Maximum Number of Iterations	Maximum number of algorithm iterations to find a solution.
Maximum Execution Time	Similar to maximum number of iterations, but defines a time limit instead of an iteration limit.
(Optimisation Cost Function)	The cost function is the weighted sum of all algorithm optimisation criteria. Because of the large impact that changing the cost function has on the algorithm, each modification constitutes a "candidate algorithm".

Table 16: Off-line Intra-domain IP-TE Controlled Variables

Uncontrolled variables are those parameters form the operating environment of the simulation.

Uncontrolled Variables	
Number of nodes	The number of nodes is modified by changing the network topology. The network topology is an input to the algorithm.

Number of links	The number of links is modified by changing the network topology. The network topology is an input to the algorithm.
Network traffic load (demands)	The load created on the network through the traffic demands passed with the iTM, should be varied between 40-110% but has to be calibrated since the load is an indirect quantity created by demands in the iTM.
Network Traffic Load with changing granularity	For this, the granularity of individual iTM traffic demands is varied. Coarse granularity leads to fewer optimisation opportunities for the traffic engineering algorithm.
Modified Network Traffic load	Once a solution has been computed for a specific iTM, this iTM is modified so that the effect of the change can be observed. A modification could be the removal or addition of ingress/egress demands.

Table 17: Off-line Intra-domain IP-TE Uncontrolled Variables

4.4 Experimentation Environment

The primary purpose of experimentation, as stated in the objectives to this section, is to investigate the performance of the traffic engineering algorithms based on IP link weights. These link weight based traffic engineering algorithms are off-line processes (i.e. not at router level) that have specific requirements and require a custom built software application. The application takes as input a network topology as well as a traffic demand matrix in order to calculate a set of link weights that best suite the QoS of the demands. The output to the application is a set of link weights that could potentially be implemented in either testbed or flow/packet level simulator[3]. However, none of the simulations described in this section require packet level capabilities and any flow level simulation required can be carried out within the custom software package. It is therefore argued that implementation of any solutions in either real network or network simulator is not required.

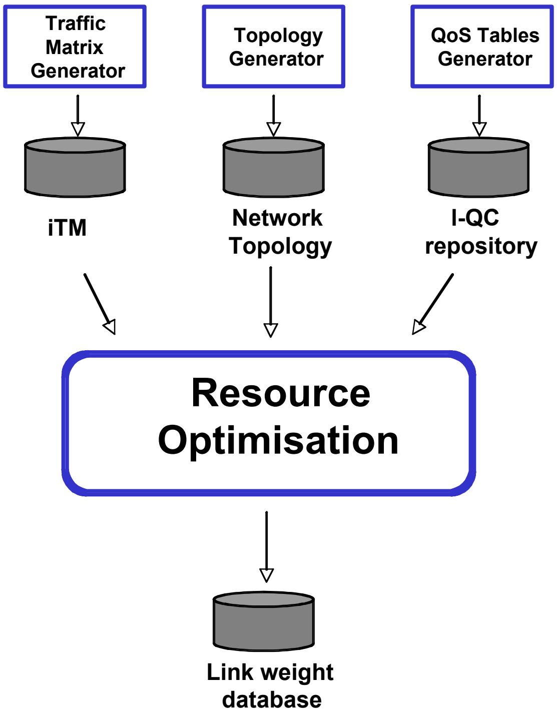


Figure 4: Inputs and Output from Resource Optimisation Software

[3] Direct implementation of a solution is subject to either 1. The routers/simulator having an implementation of per DSCP routing as specified in deliverable [D1.2] or 2. The solution having been limited to a single routing plane.

The software platform is Java based and implements the Dijkstra shortest path algorithm, which is extended with the functionality as described in [D1.2]. In addition to internal settings, the software requires the following input data in order to compute a set of link weights:

- intra-domain network topology from network topology repository
- iTM from the traffic matrix repository
- information on l-QCs deployed from the QoS tables repository

Output generated by the software platform is stored in the link weight repository.

4.5 Test Campaigns

This section details test campaigns for each of the three experimentation phases. The phases are Functional Validation (Funct), Algorithm Performance and Optimisation (Perf) and Algorithm Efficiency and Optimisation (Effic).

Test Id	Purpose	Description
IPTE/Funct/1	Functional validation, ensure correct function of the TE algorithm by checking hop count constraints and link utilisation limits are honoured for a number of different topologies and varying network traffic load.	Performance metrics: hop count limit per l-QC, link utilisation max per l-QC Controlled variables: Uncontrolled variables: *number of nodes in topology,* low, medium and high network traffic load
IPTE/Funct/2		Performance metrics: hop count limit per l-QC, link utilisation max per l-QC Controlled variables: Uncontrolled variables: *total network load* (higher number of demands), small, medium and large number of nodes
IPTE/Perf/1a	Analyses number of demands that can be accommodated by each algorithm. Average link utilisation plotted against network traffic load for each candidate and comparison algorithm.	Performance metrics per candidate algorithm: average link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load (with changing granularity), small, medium and large number of nodes
IPTE/Perf/1b	Investigates and compares the flexibility of traffic engineering solutions by analysing the variations in link utilisation. Shows plots of StdDev on link utilisation against number of traffic demands.	Performance metrics per candidate algorithm: standard deviation of link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load (with changing granularity), small, medium and large number of nodes
IPTE/Perf/2a	As IPTE/Perf/1a, but to the found solutions, apply different network traffic demand matrices in order to assess solution flexibility towards changes in demand.	Performance metrics per candidate algorithm: average link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: modified network traffic load applied to the solutions found in IPTE/Perf/1a no re-computation of solutions, small, medium and large number of nodes

IPTE/Perf/2b	As IPTE/Perf/1b with IPTE/Perf/2a modification.	Performance metrics per candidate algorithm: standard deviation of link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: modified network traffic load applied to the solutions found in IPTE/Perf/1b no re-computation of solutions, small, medium and large number of nodes
IPTE/Perf/3a	As IPTE/Perf/1a, but to the found solutions, apply slightly modified topology. I.e. remove single link or node to demonstrate solutions flexibility towards change in topology.	Performance metrics per candidate algorithm: average link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load applied to the solutions found in IPTE/Perf/1a, removal of singular link or node
IPTE/Perf/3b	As IPTE/Perf/1b with IPTE/Perf/3a modification.	Performance metrics per candidate algorithm: standard deviation of link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load applied to the solutions found in IPTE/Perf/1b removal of singular link or node
IPTE/Perf/4a	Assessment of effectiveness of using different routing planes for traffic engineering. As IPTE/Perf/1a but compute the solutions with demands highly distributed through the different l-QC routing planes and with demands concentrated on a single routing plane.	Performance metrics per candidate algorithm: average link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load (with changing granularity), all routing planes set to map to the same l-QC, small, medium and large number of nodes
IPTE/Perf/4b	Assessment of effectiveness of using different routing planes for traffic engineering. As IPTE/Perf/1b but compute the solutions with demands highly distributed through the different l-QC routing planes and with demands concentrated on a single routing plane.	Performance metrics per candidate algorithm: standard deviation of link utilisation, network traffic load (in terms of traffic demands) Controlled variables: Uncontrolled variables: network traffic load (with changing granularity), all routing planes set to map to the same l-QC, small, medium and large number of nodes
IPTE/Effic/1	Execution time and scalability comparison for different algorithm versions. Comparison with different number of nodes.	Performance metrics per algorithm version: algorithm execution time, number of nodes in topology Controlled variables: Uncontrolled variables: number of nodes in topology, number of links in topology, traffic demands
IPTE/Effic/2	Execution time and scalability comparison for different algorithm versions. Comparison with fine grain and coarse grain traffic demands	Performance metrics per algorithm: execution time, number of links in topology Controlled variables: Uncontrolled variables: number of different traffic demands, network topology

IPTE/Effic/3	Execution time and scalability comparison for different algorithm versions. Varying network load.	Performance metrics per algorithm: execution time, network traffic load Controlled variables: Uncontrolled variables: network traffic load, network topology
IPTE/Effic/4	Execution time and scalability comparison for different algorithm versions focusing on routing plane functionality of the algorithm.	Performance metrics per algorithm: execution time, network traffic load Controlled variables: Uncontrolled variables: network traffic load spread across routing planes and concentrated on one routing plane with 1. Same 2. Different l-QC requirements, network topology
IPTE/Effic/5	Convergence properties of different algorithms. Cost function plotted against iteration number or execution time. The investigation should provide a profile of the convergence rather than just time to convergence.	Performance metrics per algorithm: cost function, algorithm iteration, execution time (different graph) Controlled variables: Uncontrolled variables: network topology, traffic demands (several different scenarios for load and topology changes)
IPTE/Mixt/1a	An investigation into how fast a good result be achieved is a mixture between performance and efficiency tests. It is also useful in adjusting the cost function towards providing faster convergence. The first test specifies average link utilisation plotted against iteration number.	Performance metrics: average link utilisation, algorithm iteration number Controlled variables: Maximum Number of Iterations, Maximum Execution Time Uncontrolled variables: network topology, network traffic load (several different scenarios for load and topology changes)
IPTE/Mixt/1b	The second test specifies standard deviation on link utilisation plotted against iteration number.	Performance metrics: standard deviation on link utilisation, algorithm iteration number Controlled variables: Maximum Number of Iterations, Maximum Execution Time Uncontrolled variables: network topology, network traffic load (several different scenarios for load and topology changes)

Table 18: Off-line Intra-domain IP TE Tests

5 OFF-LINE INTER-DOMAIN TE TESTING

5.1 Objectives

5.1.1 Functional Tests

The Functional Tests confirm that the Inter-domain Traffic Engineering software meets its specification, i.e. that the offline Inter-domain TE algorithms implement the functions described in [D1.2] Section 5.4.1.4 (for Binding Selection) and Section 5.4.3.3 (for Inter-domain Resource Optimisation), and meet the test requirements that were defined in outline in Section 5.4.1.5 (Binding Selection), Section 5.4.2.7 (Binding Activation) and Section 5.4.3.4 (Inter-domain Resource Optimisation) respectively.

The Inter-domain TE software runs for a particular AS (the "AS under test"). In summary, the functional tests are intended to confirm that the software correctly identifies binding candidates (combinations of l-QCs and o-QCs that meet a domain's desired e-QCs), runs sample configurations using as input traffic from the eTM for the time schedule or epoch under consideration, identifies a "best" or set of "near-best" configurations, and identifies changes required to the pSLSes established by the AS under test with its neighbours.

5.1.2 Algorithm Benefit/Cost Performance Tests

The Algorithm Benefit/Cost Performance Tests assess the benefits and costs of each algorithm that has been proposed for the Inter-domain Traffic Engineering software. These tests include the test requirements that were defined in outline in [D1.2] Section 5.4.1.5 (Binding Selection), Section 5.4.2.7 (Binding Activation) and Section 5.4.3.4 (Inter-domain Resource Optimisation) respectively.

The principal algorithms (genetic and heuristic) will be assessed so as to enable their principal parameters to be optimised. The relative performance of all algorithms (random, brute force, genetic and heuristic) will then be compared. Finally, the behaviour of the systems as a function of the parameters that control the interaction between Binding Selection and Inter-domain Resource Optimisation will be investigated.

5.1.3 Scalability Tests

The Scalability Tests assess the computational performance of each algorithm that has been proposed for the Inter-domain Traffic Engineering software. These tests show the extent to which the algorithms are suitable for large scale deployment in large topology networks. The tests include the test requirements that were defined in outline in [D1.2] Section 5.4.1.5 (Binding Selection), Section 5.4.2.7 (Binding Activation) and Section 5.4.3.4 (Inter-domain Resource Optimisation) respectively.

Each of the algorithms will be assessed for scalability, with the emphasis being on the genetic and heuristic algorithms.

5.1.4 Stability Tests

The Stability Tests verify that the Inter-domain TE algorithms are stable, under representative network topologies and traffic. The tests include the test requirements that were defined in outline in [D1.2] Section 5.4.1.5 (Binding Selection), Section 5.4.2.7 (Binding Activation) and Section 5.4.3.4 (Inter-domain Resource Optimisation) respectively.

5.2 Performance Metrics

The performance metrics defined here shall be used to compare the behaviour of each of the Inter-domain TE algorithms.

Performance Metrics		
Algorithm Benefit/Cost Performance Metrics	*Maximum inter-domain link cost*	Calculated comparing all inter-domain links (see [D1.2] Section 5.4.3.3.2.2]).
	Total cost of all inter-domain links	(see [D1.2] Section 5.4.3.3.2.2]).
	Minimum intra-domain link cost	(see [D1.2] Section 5.4.3.3.2.2]).
	Inter-domain link load	Mean and sd across all links.
	Inter-domain link utilisation	Mean and sd across all links.
Scalability Performance Metrics	Computing performance metrics	*run time (elapsed time)*
		computer processing time
		memory requirements

	Algorithm performance metrics	*number of iterations to convergence*
Stability Performance Metrics	Transient response of pSLSes as traffic flows are added and removed	*mean pSLS lifetime*
		pSLS duty cycle
	Route stability	*variability of paths selected for each flow (egress link selected, e-QC assigned, l-QC assigned, o-QC of downstream AS assigned)*
		frequency with which downstream AS changes

Table 19: Off-line Inter-domain TE Performance Metrics

The performance metrics for stability tests shall be based on analysing how flows are engineered in the network from one provisioning cycle to the next, as the eTM varies (i.e. as other flows are added and removed). The tests will therefore be run over multiple provisioning cycles.

5.3 Controlled and Uncontrolled Variables

The controlled variables of the Inter-domain TE software are the configuration parameters that affect the algorithm behaviour. These vary depending on the test being conducted, but in general include:

Controlled Variables
Algorithm parameters (e.g. for the genetic algorithm: N, p_c, p_m)
Binding Selection / Inter-domain Resource Optimisation interaction parameters
Perturbation approaches for optimising required pSLSes
Algorithm / formula for calculation of inter-domain cost Ψ

Table 20: Off-line Inter-domain TE Controlled Variables

Uncontrolled variables are the parameters that define the configuration and capabilities of the network under test, as follows:

Uncontrolled Variables	
Network topology	intra-domain topology, number of peer domains and adjacent AS topology, inter-domain link topology
QoS Capabilities of this domain	number and value of e-QCs and l-QCs supported in the AS under test
QoS Advertisements from neighbouring domains	number of advertisements in total, mean number per AS, number and value of o-QCs supported by all adjacent ASes
Traffic flows, as defined in the eTM and iTM	

Table 21: Off-line Inter-domain TE Uncontrolled Variables

5.4 Experimentation Environment

The test platform shall be the Inter-domain TE software developed by UniS.

5.4.1 Functional Tests

The functional tests shall be performed using the Very Small Network Topology (Figure 5a) and the Small Network Topology (Figure 5b). The former is a degenerate case where there is negligible intra-domain TE and hence the testing of the Inter-domain TE functions is separate from the behaviour of any MESCAL Intra-domain TE software.

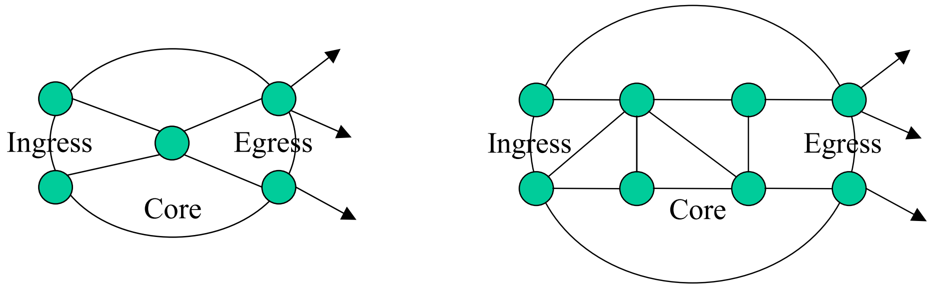


Figure 5a and b: Very Small Topology and Small Topology (Functional Tests)

The functional tests shall be conducted using a small network eTM, iTM, initial pSLS repository, QoS Capabilities repository and QoS Advertisement repository.

Additional test software requirements: for the Binding Selection functional tests, a stub IDRO module will be required. For the IDRO and System functional tests, a stub Traffic Forecast / Intra-domain TE module will be required.

5.4.2 Algorithm Benefit/Cost Performance Tests

The benefit/cost performance assessment tests shall be performed using Medium and Large Topology networks. These networks model a single AS in detail together with its adjacent ASes. The AS under test is modelled as a full router topology (i.e. a set of border routers and core routers, with defined interconnections). The router topology within adjacent ASes is not modelled; however, the inter-domain links are defined; each is represented as a connection between a router within the AS under test and an adjacent AS. In general all border routers will be both ingress and egress (this compares with the functional test Very Small and Small Topologies, where for modelling simplicity each border router only undertakes either ingress or egress functions but not both).

The performance assessments tests shall be conducted using generated eTM, iTM, initial pSLS repositories, QoS Capabilities repository and QoS Advertisement repository.

Additional test software requirements: for these tests, a stub Traffic Forecast / Intra-domain TE module will be required.

5.4.2.1 Internet Topology and Traffic Modelling

For all performance tests the structure of the intra-domain and inter-AS configuration will be based on realistic network topologies. The "Internet model" thus needs to be realistic in the following respects:

- The AS under test should contain a representative number of border and core routers, in a realistic configuration (i.e. realistic intra-domain connection);
- The adjacent AS distribution should be realistic, with an appropriate number of adjacent ASes, each of which has an appropriate number of links to our AS under test. This requires different configurations depending on whether the AS under test is a Tier 1, 2 or 3 provider;
- The set of destination address prefixes advertised (and represented in the model by the QoS Advertisement repository) must be realistic so that any given address can be reached via a realistic number of adjacent ASes;
- The network traffic, represented by the contents of the eTM, must follow the characteristics of realistic inter-domain traffic.

A number of authors have worked on some of these aspects of Internet modelling, and their work is now briefly reviewed. Specifically, we review work on Internet topology and on Internet traffic.

Magoni [Mag01], Hipolito [Hipol03] and Govindan [Govin02] examined the topological and geographical properties of real ASes, considering properties such as numbers of nodes, number of edges, average degree and distance, routing policy, etc. Subramanian [Subra02] classified the Internet

ASes based on commercial relationships (peering or transit); their work illustrates the number of links each AS has, but does not draw conclusions on the number of edge nodes. Our model of a realistic Internet will therefore take account of the findings and data presented in these papers in order to generate realistic graphs for use in our inter- and intra-domain simulation.

Heckmann [Heckmann03] notes that many existing standard network topology models do not model individual routers but instead model ISP points-of-presence (POPs). He also defines some metrics that capture the connectivity properties of the topology graph and which enable comparison of the similarity of real-life and model topologies.

Faloutsos [Falout99] discovered that the degree distribution of the current Internet graph follows a power-law. This implies that, when attempting to generate a realistic Internet graph, it is more important to match the degree distribution than to capture the hierarchical structure of the Internet. This affects our modelling requirement for the AS-level topologies such as the distribution of adjacent ASes, etc.

At the time of writing the number of address prefixes is reported to be of the order of 160 000 [bgp.potaroo.net]. Feamster [Feamster03] showed that a small set of destination prefixes (popular destinations) is responsible for large fraction of the outbound traffic. Thus, we can engineer a large volume of traffic by only manipulating small fraction of destination prefixes. A similar finding has also been reported in non-transit ISPs [Uhlig02]. Rexford [Rexford01] also confirmed that a small fraction of popular destinations is responsible for the bulk of Internet traffic and furthermore found these destinations to be remarkably stable. Uhlig [Uhlig02] looked at the day-to-day stability of the highest sending traffic sources in non-transit ISPs and also showed that they were stable both in terms of their contribution to the total traffic and in terms of their presence among the top number of traffic sources. This point of traffic stability affects how we select a set of destination prefixes to be considered and therefore affects our modelling requirement for the number of destination prefixes to be considered. Quotin [Quoit03] further presented charts that show the AS hop distance of traffic, and concluded that most data packets are exchanged with ASes that are only a few AS hops away. Again, this point also affects our modelling requirement for the network topology, QoS advertisements, and the eTM.

5.4.3 Scalability Tests

The scalability tests shall be performed using Medium and Large Topology networks as described in Section 5.4.2, using input data as also described in that section.

Additional test software requirements: for these tests, a stub Traffic Forecast / Intra-domain TE module will be required.

5.4.4 Stability Tests

The stability tests shall be performed using Small, Medium and Large Topology networks as appropriate, as described in Section 5.4.2, using input data as also described in that section

Additional test software requirements: for these tests, a stub Traffic Forecast / Intra-domain TE module will be required. The eTM generator will need to be extended to produce a set of eTMs for successive provisioning cycles.

5.5 Test Campaigns

Test Id	Purpose	Description
InterTE/Funct/BSel	Binding Selection functions	Confirm correct binding candidates identified, taking into account the time schedule. Confirm that any flow (e-QC / prefix) in the eTM that cannot be met by the available set of l-QCs and o-QCs is correctly identified and flagged. Confirm operation of pSLS perturbation algorithm. Confirm pSLS changes correctly generated (create, change and cease). pSLS Ordering interactions are excluded from these tests.
InterTE/Funct/IDRO/1	General operation of Inter-domain Resource Optimisation	Confirm eRAM correctly calculated and contents of output data structure are correct. Confirm Inter-domain cost function Ψ correctly calculated (based on input intra-domain cost Φ).
InterTE/Funct/IDRO/2	Random algorithm functions	Variation 1 (uses existing pSLSes only): confirm all flows are assigned to egress links and meet QoS and bandwidth requirements. Demonstrate also that where the flows are greater than the available egress capacity, the algorithm fails. Variation 2 (allows pSLSes to be randomly selected): as Variation 1, and also confirm new pSLSes are within the capacity constraints of the egress links.
InterTE/Funct/IDRO/3	Brute force algorithm functions	Confirm that for each configuration tested by the algorithm a cost function is correctly calculated. Confirm that the algorithm returns the lowest cost function configuration found.
InterTE/Funct/IDRO/4	Genetic algorithm functions	Confirm an initial random chromosome population is generated. Confirm crossover and mutation functions performed. Confirm fitness function correctly calculated, and new generation population correctly chosen.
InterTE/Funct/IDRO/5	Heuristic algorithm functions	Confirm that the sorting function is correctly done. Confirm the penalty/cost calculation correctly calculated. Confirm the egress router selection correctly performed without violating any defined capacity/traffic constraints/requirements. Confirm the resource status is correctly updated after every step of the selection.
InterTE/Funct/BAct	Binding Activation functions	Confirm that bandwidths for all pSLSes are correctly calculated. Confirm that optimal configuration is identified and eRAM correctly generated (for transfer to Dynamic Inter-domain TE). Confirm that any flow in the eTM that cannot be supported due to the given set of pSLSes is flagged.
InterTE/Funct/System	System functional tests: interworking between Binding Selection / Binding Activation and Inter-domain Resource Optimisation	Confirm correct binding candidates (e-QCs as combinations of l-QCs and o-QCs) identified in Binding Selection and passed to IDRO. Confirm correct pSLS create / change / cease. Confirm correct parameters passed from Binding Activation to IDRO. Confirm flows assigned within pSLS agreements. Confirm QoS constraints met on all flows, both intra-domain and inter-domain (egress links). Confirm bandwidth limits met on inter-domain links.

InterTE/Perf/1	Performance assessment of genetic algorithm	Assesses the impact of algorithm parameters on the algorithm performance. Parameters to be varied (controlled variables): chromosome population *N*, crossover and mutation probabilities, p_c and p_m. Impact of cost function Ψ calculation. Assess performance for different AS types (Tier 1 / 2 / 3, connections to peer ASes and transit ASes). Record number of iterations to convergence.
InterTE/Perf/2	Performance assessment of heuristic algorithm	Assesses the impact of cost function Ψ calculation on algorithm performance. Impact of cost function parameters, such as α, on the cost function performance are also considered.
InterTE/Perf/3	Performance comparison of algorithms	Assess the relative performance of the algorithms (random, brute force, genetic, heuristic). For comparisons against the brute force algorithm (which is expected to run exceedingly slowly) it is likely that only the Very Small or Small Network Topology could be used. For comparisons against the random algorithm, Medium or Large Topologies will be used.
InterTE/Perf/4	Performance assessment of Binding Selection / Binding Activation / Inter-domain Resource Optimisation interactions	Assess the impact of Binding Selection / Binding Activation / IDRO interaction parameters on the optimality of the solution for each algorithm. Parameters to be varied (controlled variables) include: perturbation approaches, including definition of "significant change" ([D1.2] Section 5.4.1.4.3.2); mechanisms for selection of candidate pSLSes for order / change / cease.
InterTE/Scal/1	Scalability of genetic algorithm	Investigate performance of genetic algorithm as a function of input parameter volumes. Principal controlled variables: number of aggregate flows, number of border routers; number of l-QCs in AS; number of o-QCs and advertisements from adjacent ASes.
InterTE/Scal/2	Scalability of heuristic algorithm	Investigate performance of heuristic algorithm as a function of input parameter volumes. Principal controlled variables: number of aggregate flows, number of border routers; number of l-QCs in AS; number of o-QCs and advertisements from adjacent ASes.
InterTE/Scal/3	Scalability of brute force algorithm	Investigate performance of brute force algorithm as a function of input parameter volumes. Principal controlled variables: number of aggregate flows, number of border routers; number of l-QCs in AS; number of o-QCs and advertisements from adjacent ASes.
InterTE/Scal/4	Scalability of random assignment algorithm	Investigate performance of random assignment algorithm as a function of input parameter volumes. Principal controlled variables: number of aggregate flows, number of border routers; number of l-QCs in AS; number of o-QCs and advertisements from adjacent ASes.
InterTE/Stab/1	Stability of genetic algorithm	Investigate stability performance of genetic algorithm over multiple provisioning cycles as function of eTM.
InterTE/Stab/2	Stability of heuristic algorithm	Investigate stability performance of heuristic algorithm over multiple provisioning cycles as function of eTM
InterTE/Stab/3	Stability of brute force algorithm	Investigate stability performance of brute force algorithm over multiple provisioning cycles as function of eTM

InterTE/Stab/4	Stability of random assignment algorithm	Investigate stability performance of random assignment algorithm over multiple provisioning cycles as function of eTM

Table 22: Off-line Inter-domain TE Tests

6 Q-BGP TESTING

6.1 Objectives

This series of tests aims to examine three major aspects of a QoS-enabled BGP inter-domain routing system in a Solution-Option 1 [D1.1] MESCAL deployment. The aspects considered are:

- ***Scalability***, which aims to examine how the number of q-BGP messages depends on variables such as network size, topology, and traffic demand patterns.
- ***Stability***, which aims to consider the sensitivity of the q-BGP routing algorithms and protocol to changes in the inter-domain network and their ability to settle in a stable state. These changes could include inter-domain link failure or changes in demands.
- ***Efficacy***, which aims to consider the ability of q-BGP routing algorithms to find the optimal routes for a given demand matrix. Optimal is considered to be an inter-domain routing configuration that will accommodate demands with an acceptable level of QoS with minimal resource usage (e.g. inter-domain link usage).

Other aspects of inter-domain routing such as security and authentication are not considered in these tests.

A macro-scale simulation of a large Internet-like network will be performed where we can examine q-BGP's large-scale behaviour – tests which would be unfeasible on the testbed. The simulation will consider aggregate flows and not individual packets as that level of detail is both unnecessary and will take too long to simulate.

A series of scenarios will be simulated and various measurements captured, such as routing table dumps, internal variables and entire topologies, and then a post-simulation off-line analysis will be performed. The intention is to have a modular software approach which will allow us to use input files which are generated by other applications, such as topology generators, and generate output files, which can then be analysed further.

The actual q-BGP systems (the whole protocol, messaging, and management systems) whose scalability, stability and efficacy will be investigated can be split into three phases:

Phase 1: Fixed q-BGP policies – as proposed in the original q-BGP specification [D1.1]. The experimentation would involve an investigation of various QoS attribute parameters (QoS attributes such as available bandwidth, delay, packet loss ratio etc) in the q-BGP messages and whether they are administratively set, or dynamic measurements, or a function of the two. In the case of an unstable network a means of dampening would be investigated. Typical BGP problems would also be considered, such as the injection of false reachability and QoS information, as well as network failures.

Phase 2: Similar to Phase 1 in the examination of the resulting routing, but the route selection algorithms are more sophisticated and the policies aren't fixed, nor globally defined. Whereas in Phase 1 q-BGP followed a strict decision process with a well defined priority level in the QoS attribute based decision process, here the intention is to investigate the added value in having dynamic inter-domain traffic engineering which is local to each AS, and not defined globally for all ASes.

Phase 3: The third phase of q-BGP strategy investigation is considered optional within the scope of MESCAL and a possible future research direction. It is proposed to investigate techniques for providing harder end-to-end QoS guarantees under Solution-Option 1. This would require the definition of acceptable end-to-end quality classes (GQCs, or Global Quality Classes) [D1.1]. ISPs would then make inter-domain routing decisions to meet stricter QoS demands, which are known from the GQC specification and on the role and position of the AS in the end-to-end chain of ASes forming the path between particular source-destination prefixes for that GQC. This is to distinguish from the other phases which have no concept of end-to-end delivered QoS and only consider local QoS treatment and the performance parameters announced by downstream ASes in their q-BGP announcements. Here it is proposed that the AS has some knowledge of where it resides in the QoS delivery ISP chain and adjusts its route selection process and QoS transfer characteristic while maintaining the meta-class bindings. To increase the flexibility yet further load splitting and load balancing will be considered.

In every case, the simulation process is the same. The simulator is initialised with an input inter-AS connectivity topology and a pSLS capacity matrix. The inter-domain network doesn't yet have any demands applied to it, however. The overall simulation procedure is then to apply demands to the network with a series of trigger events. Initially these events would be "add demand" and would entail the addition of demands sequentially by routing each one individually and allowing the network to settle before applying the next one. Having applied the demands a series of other events could be then triggered such as:

- the addition of further demands
- the removal of demands
- the destruction of an inter-domain link
- the creation of an inter-domain link
- an internal change within an AS

When such a trigger event occurs the simulator will then repeatedly perform the functions of each simulation element once in every simulation epoch. These epochs are repeated until either the network state settles, and doesn't change any more, or the simulator goes through a fixed loop of states. For example, an AS would execute the incoming q-BGP message filtering and decision processes and then send out new q-BGP messages within an epoch; messages which will be acted on by the adjacent ASes in the next epoch. The state of the entire network is stored after each simulation epoch so it can then be analysed off-line to investigate issues such as the time (number of epochs) it took to settle in an alternative routing configuration following an inter-domain link failure (as part of the stability test suite).

This approach gives us the flexibility to examine the behaviour in time of various simulated conditions within the same modular environment, as well as make it possible to easily add future events.

It should be noted that we are only simulating q-BGP in Solution-Option 1 and that since each meta-class runs a separate instance of q-BGP, will we only be simulating a single meta-class. It is assumed that there is a partitioning between solution options and meta-classes and therefore there is no interaction.

6.2 Simulation Modules and Experimental Variables

A number of approximations and assumptions are made and some generators are proposed to provide realistic input variables to the simulator. Below is an overview of these to provide a better understanding of the variables involved, while their more specific use is described later when examining each experiment.

6.2.1 Inter-AS Connectivity Topology

As part of the functional validation we use a series of smaller (4 to 10 node) manufactured topologies (such as fish) for which it would be easier to manually find the routing results. For the non-validation experiments, however, power-law compliant topologies generated by BRITE [BRITE] will be used

[D2.1] as they are more similar to the actual Internet topology than other structure-based topology generators [Tang01].

6.2.2 pSLS Capacity Generator

Since we are experimenting on inter-domain routing this is an important variable as it provides us with the scarce resource for which to optimise our routing for. Inter-domain routing in MESCAL is only possible between adjacent ASes between which pSLSes already exist and therefore it is assumed that pSLSes exist on every inter-domain link generated by the inter-AS connectivity topology generator above. However, the available capacity in these pSLSes cannot just be randomly allocated and must be configured in such a way as to provide *usable* end-to-end capacity. This is usually performed by the pSLS handling, RPC and off-line inter-domain TE components, but for these experiments we will generate pSLSes capacity according to the following strategy:

> Given a set of end-to-end demands and an AS topology the demands are routed across the inter-domain topology along either the second, third, or fourth shortest paths. The sum of the bandwidth demands across each inter-domain link is then taken as the minimum capacity of the pSLSes.

This routing configuration from which the pSLS capacities were derived is known as our *reference inter-domain routing configuration* and the pSLS capacity allocation is unlikely to yield any alternative routing configurations for the same set of demands. Since the q-BGP algorithms have different optimisation targets (such as to optimise delay, hop count, utilisation etc) to the above algorithm it is not expected that any q-BGP algorithm will always perform identically to find this reference routing configuration. Therefore, to be able to investigate a q-BGP strategy's ability to find acceptable routing configurations it is proposed to scale all pSLS capacities by a pSLS capacity co-efficient. This co-efficient, or *over-provisioning factor*, will then be a measure of the q-BGP system's ability to find the optimal routing configuration as less over-provisioning will be required by the better q-BGP strategies.

6.2.3 l-QC Generator

For simplicity it is assumed that within each AS there are pre-defined l-QCs between all ASBRs (AS Border Routers). The ASes are assumed to have infinite bandwidth for accommodating demands for the offered l-QCs (since the scarce resource which we are optimising for is the pSLS capacity and l-QCs should be matched anyway to pSLSes), and a fixed QoS transfer characteristic (i.e. a fixed delay). The delay parameter, for example, would be generated with a uniformly random number generator, the bounds of which are realistic for the meta-class plane we are running the q-BGP for and constant for all the l-QCs within a single AS.

6.2.4 Demand Topology Generator

These are the amounts of bandwidth allocated to flows between each edge AS pair. Edge AS pairs are chosen from the AS connectivity topology as nodes which have one or two links. Nodes/ASes with more links are considered to be transit/core ASes. The demand's capacity will be uniformly randomly distributed between *demand_min* and *demand_max*. The demand's end-points are also chosen at random, so there won't necessarily be a demand between every edge AS pair. The demands are expressed as a time-sorted list, so that they can be added to the network in the order specified so the addition of demands could be used as a simulation trigger event (as described in Section 6.1). The use of such time-sorted demands also allows for the simulation of dynamic traffic and the emulation of traffic splitting.

6.2.5 IP Prefix Distribution

This is a variable which is very significant for the scalability testing, as it heavily impacts the ability to aggregate q-BGP advertisements, and therefore scalability. A uniformly random distribution is unrealistic as in the real Internet there are four organisations (APNIC, ARIN, LACNIC and RIPE

NCC) distributing IP address space across four geographic regions [RIR]. These are allocated to larger ISPs who then allocate to smaller ISPs, and there is therefore a hierarchical element to the prefix distribution. It is proposed to emulate this by finding the logical centre of the input AS topology and allocate large segments of address space to the cluster of ASes surrounding it. Then, moving away from the central ASes, along random links away from the centre, segments of address space are split and allocated to further ASes. This is repeated away from the centre until the edge is reached. This can be seen in the diagram below (Figure 6).

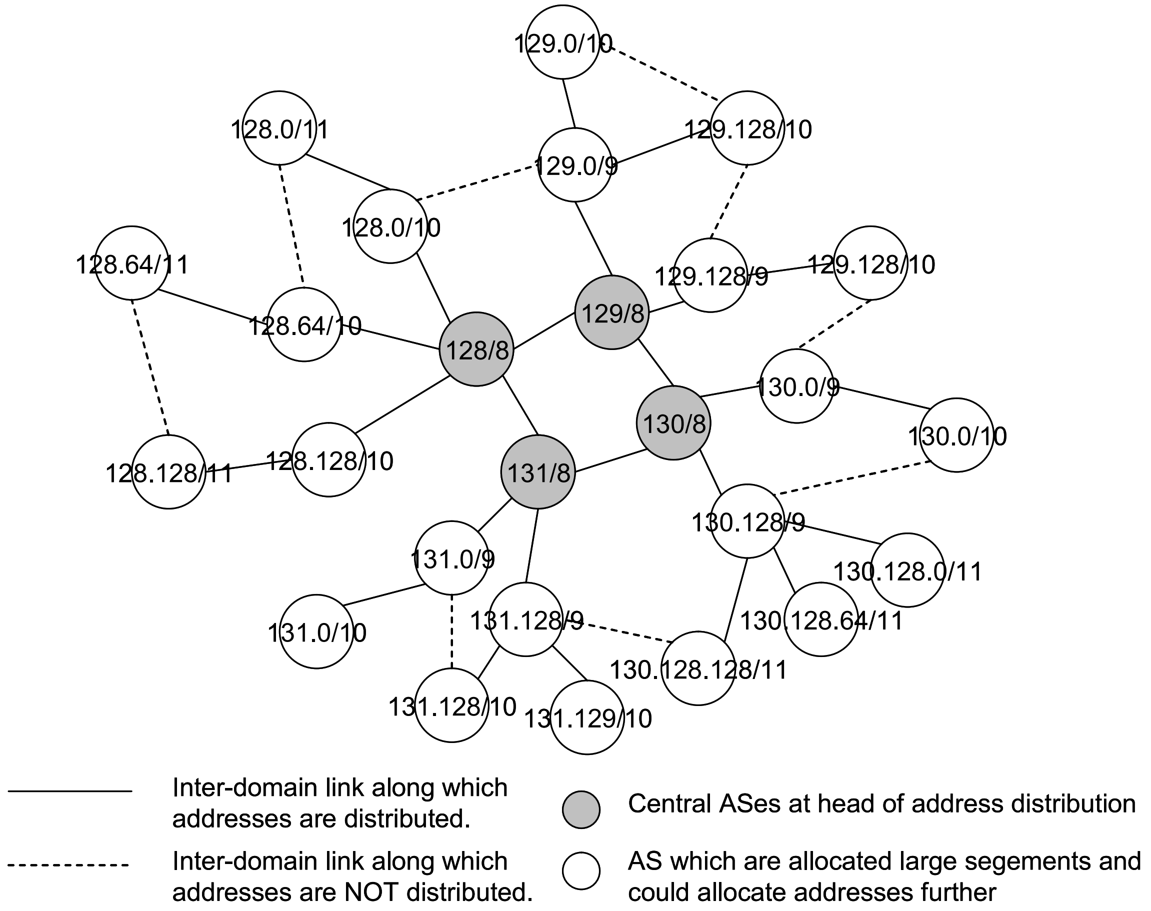

129.0/10
128.0/11
129.128/10
129.0/9
128.0/10
128.64/11
129.128/9
129.128/10
128.64/10
129/8
128/8
130.0/9
130/8
130.0/10
128.128/11
128.128/10
131/8
130.128/9
131.0/9
130.128.0/11
131.0/10
131.128/9
130.128.64/11
130.128.128/11
131.128/10
131.129/10
Inter-domain link along which addresses are distributed.
Central ASes at head of address distribution
Inter-domain link along which addresses are NOT distributed.
AS which are allocated large segements and could allocate addresses further

Figure 6: The Proposed IP Prefix Distribution Technique

It is thought that the use of this algorithm, while probably not ideally realistic, for q-BGP simulations it would still be better than a totally random distribution of prefixes. Also, it should be noted that it won't artificially cause "perfect aggregation" as this will only occur along the route of links along which the addresses were distributed (the solid lines in Figure 6), and aggregation probably won't occur to the same extent on the other links (the dashed lines in Figure 6).

6.2.6 Network Reactivity

Without a packet-level simulation we require some way of simulating the effect of heavy load on delivered QoS. It would, for example, be unrealistic to assume that an inter-AS link working at maximum capacity wouldn't experience any kind of queuing delay. To this purpose it is proposed to assume an M/D/1 output queue on network interfaces and assume the delay is linked to link utilisation. It is therefore assumed that packets arrive with a Poisson distribution and leave the output queue with a deterministic (constant-bit-rate) service rate. This generated delay value is also required for the q-BGP QoS-Attribute (QA) when testing the effect of dynamic measurement information in the q-BGP messages and for the measurement of delivered end-to-end QoS. The average delay caused by queuing can then be calculated by:

$$T = (\mu/\lambda)^2 (2\lambda (1-(\lambda/\mu)))^{-1}$$

Where μ is service rate (a function of pSLS capacity) and λ is mean arrival rate (a function of aggregated demand rate).

6.3 Efficacy Experiments

These experiments aim to compare the ability of various q-BGP policies and strategies to find optimal routes.

6.3.1 Performance Metrics

Performance Metrics	
Delivered QoS *(e.g. DelivDelayAvg, DelivDelaySD, DelivBandwidthFracAvg, DelivBandwidthFracSD)*	This is the actual QoS that the demands receive. This is expressed as the average and standard deviation of a certain QoS attribute, in the usual case, delay. The fraction of the offered bandwidth which was actually delivered, averaged across all demands is also considered. A delivered bandwidth fraction of 1.0 would imply that all the bandwidth of the demands was successfully delivered.
Network Utilisation Average (NetUtilAvg) *Network Utilisation Standard Deviation (NetUtilSD)*	This is a measure of the accuracy and load balancing of the routing solutions created by the routing algorithms. This is expressed as the mean average and standard-deviation of the utilisation of all pSLSes in the network.

Table 23: q-BGP Efficacy Testing Performance Metrics

6.3.2 Uncontrolled Variables

Uncontrolled Variables	
Network Size (NetworkSize)	This is the number of ASes in the network being simulated (AS connectivity topology generator parameter) This value is expected to be fixed at 500 ASes for all efficacy experiments. This isn't close to the size of the internet, but for experiment running time purposes it is limited to 500. It isn't essential that this value is close to the real internet as we are comparing q-BGP policies against each other and not the actual internet. It should be enough to demonstrate large scale behaviour.
Number of inter-domain links per AS (InterDomainLinksPerAS)	The ratio of inter-domain links to the number of ASes. (AS connectivity topology generator parameter) This is expected to take the value of 2.0, which is an approximation based on the average outdegree (outgoing links) of the Internet [Falout99] (the outdegree (3.42, 3.65, 3.76) value is halved to derive the number of uni-directional links per AS).
Demand topology *(demand_min, demand_max)*	As described in section 6.2.4 demand_min is 1 and demand_max is 100. The values aren't critical as the pSLS generator will derive and accommodate the pSLS capacities to these values.
l-QC Delay Range *(e.g. lQCDelayMax ,lQCDelayMin)*	This is the upper and lower bounds of the random values assigned to l-QC delays. lQCDelayMax is set at 100mS and lQCDelayMin is set to 10 mS.

Table 24: q-BGP Efficacy Testing Uncontrolled Variables

6.3.3 Controlled Variables

Controlled Variables	
pSLS over-provisioning factor *(pSLSOverFactor)*	The fraction which the minimum pSLS capacities are multiplied by. Minimum pSLS capacities are generated from demand matrix (controlled variable). Values above 1.0 are therefore over-provisioning, and values below are under-provisioning. Values for this are expected to be 1.0 to 2.4 in steps of 0.2. (eight different values). Higher values may be used if it is found that the q-BGP policy sets are particularly bad at finding good routes and require additional over-provisioning..
qBGPFalseInfoFrac	The fraction of all q-BGP messages which are deliberately falsified.
qBGPFalseInfoCoEff	The co-efficient by which QoS attributes are multiplied to falsify them.
q-BGP Policies and strategies	The q-BGP policy set in use as well as the QoS-attributes used.

Table 25: q-BGP Efficacy Testing Controlled Variables

6.3.4 Experimentation Environment

Since we are examining the behaviour of a number of q-BGP policies against a series of randomly generated networks each experiment should be repeated a number of times with a different random seed and the mean average of the performance metrics considered.

To perform these experiments the inter-AS connectivity topology generator, pSLS capacity generator, l-QC generator and demand topology generator modules are required. Five instances of an AS topology are generated with the required *NetworkSize* (500 ASes) and *InterDomainLinksPerAS* (2.0). For each of these topologies five demand topologies are generated (since the demands must match the edge ASes), producing 25 network/demand configurations. The pSLSes are then generator based on the demand and AS topologies. Also, l-QCs are generated for each AS. The result is a network configuration which is ready to accept demands.

Given the input network/demand configurations each q-BGP policy set will then be applied and the performance metrics measured. The q-BGP policy applied will depend on which phase of experimentation (see section 6.1) is being performed, and would include static q-BGP policies, static policies with QoS-attributes, dynamic q-BGP policies (phase 2) and a possible investigation of harder QoS guarantees (phase 3).

For the functionality testing however a series of manually configured topologies and demands will be used, using typical topologies like fish and their derivatives.

All components are implemented in Java, while the data sets like network and demand topologies are stored in plain texts files or XML files.

6.3.5 Test Campaigns

Test Suite Id	Objective
qBGP/Efficacy/Funct	Functional validation of the q-BGP policies. The input AS topology is manually generated and compared to manually calculated results.

qBGP/Efficacy/DeliveredQoS	An examination of the amount of pSLS over-provisioning required for a given q-BGP policy set to successfully allocate all of the offered demands. A typical results plot would be one of *DelivBandwidthFracAvg* against *pSLSOverFactor*. As well as *DelivBandwidthFracSD* against *pSLSOverFactor* which would tell us the amount of overprovisioning required to successfully allocate all demands (i.e. *DelivBandwidthFracAvg* has approached 1.0 and *DelivBandwidthFracSD* has approached 0.0). A multi-trace plot of *DelivBandwidthFracAvg* against *pSLSOverFactor* for each q-BGP strategy/policy would allow direct comparison of their efficacy. Once all demands have been successfully delivered a further plot of *DelivDelayAvg* against *pSLSOverFactor* would suggest more about the effectiveness of route selection at minimising delay.
qBGP/Efficacy/ResourceUsage	This examines what the distribution of resource usage is like and gives an impression of ability to load balance and use available resources effectively. A typical plot would be *NetUtilAvg* against *pSLSOverFactor.*
qBGP/Efficacy/FalseInfo	This examines the effect of falsified QoS information on the delivered QoS and involves a comparison of performance metric with increasing *qBGPFalseInfoFrac* and *qBGPFalseInfoCoEff.*

Table 26: q-BGP Efficacy Test Suites

6.4 Stability Experiments

These experiments aim to examine the ability of q-BGP to settle in a stable state and its sensitivity to perturbations in the environment.

6.4.1 Performance Metrics

Performance Metrics	
Convergence time (qBGPConvTime)	The number of simulation cycles/epochs required before the network settles in a steady state and no more q-BGP messages are being sent, or internal AS variables are changing. This condition could however never be reached and the network could oscillate, in which case the loop attractor size should be considered.
Loop Attractor size (qBGPLAttrSize)	In the case of the network not finding a stable solution and oscillating through a series of states this is the number of states that it oscillates through. It can be a measure of the extent of oscillation, i.e. whether it affects the entire network, or just a single AS or inter-domain link.
Allocated Capacity Delta (AllocatedCapacityDelta)	This is the change (positive or negative) in capacity usage on a pSLS. It is to differentiate between the oscillation of a small single demand and the oscillation of a major aggregate of traffic, the latter being a more serious stability problem.
Delivered QoS *(e.g. DelivDelayAvg, DelivDelaySD, DelivBandwidthFracAvg, DelivBandwidthFracSD)*	This is the actual QoS that the demands receive. The *DelivBandwidthFracAvg* in particular is a measure of optimality and could be compared to the epoch to see how quickly the network approaches an approximately suitable solution.
numDemandsAffected	The total number of demands whose route has changed during the settling of the AS network into its final state.
numASesAffected	The number of ASes which have received or sent a q-BGP message as a result of the trigger event.

Table 27: q-BGP Stability Testing Performance Metrics

6.4.2 Uncontrolled Variables

Uncontrolled Variables	
pSLS over-provisioning factor (pSLSOverFactor)	The pSLS over-provisioning as chosen by the efficacy tests so as to provide enough pSLS capacity, but not massively over-provisioned so that any route could be taken.
Demand topology(demand_min, demand_max)	As described in section 6.2.4 demand_min is 1 and demand_max is 100.
Network Topology	The topology of the inter-domain AS links isn't as important as the average connectivity, since the average connectivity will affect the number of alternative routes, which in turn can influence stability. However, these variables remain uncontrolled as the comparison is between q-BGP strategies.

Table 28: q-BGP Stability Testing Uncontrolled variables

6.4.3 Controlled Variables

Controlled Variables	
q-BGP Policies and strategies	The q-BGP policy set in use including the QoS-attributes used.
Dampening strategy	This specifies what policies are used to generate and receive q-BGP messages for the purpose of dampening routing oscillations. Variables include qBGP_Update_param_threshold, which would specify the threshold value of incoming QoS parameters (contained in q-BGP update messages) for the message to be considered.

Table 29: q-BGP Stability Testing Controlled Variables

6.4.4 Experimentation Environment

In this series of experiments the focus is on the ability of the q-BGP strategy to settle in a stable state as quickly as possible, its sensitivity to changes within the AS network and how much the network is disturbed during changes. To perform these experiments the inter-AS connectivity topology generator, pSLS capacity generator, l-QC generator and demand topology generator modules are required. A count of the number of epochs elapsed since the trigger event is our measure of how long it takes to settle. However, the definition of settle is more than the lack of q-BGP messages for a few simulator epochs; settle means the lack of any kind of state change within the simulator, including internal variables to q-BGP algorithms and any persistent data stored in the AS simulation entity. Similarly the concept of oscillation includes these additional internal variables. In the case of oscillation the exact number of unique states of the simulator is the loop attractor size (*qBGPLAttrSize*). The tests then investigate, not only the various q-BGP strategies, but also dampening strategies in an attempt to prevent oscillation. Typical dampening strategies may include the artificial manipulation of QoS-attributes in outgoing q-BGP messages to decrease the reaction of adjacent ASes, or the filtering of incoming messages.

6.4.5 Test Campaigns

Test Suite Id	Objective
qBGP/Stability/SettlingSpeed	The time taken in epochs for the simulator to settle in a stable state is considered for the various q-BGP scenarios.
qBGP/Stability/OscillationMagnitude	When a settled state cannot be reached this is an examination of the extent to which the network state changes during the oscillation.

qBGP/Stability/DeliveredQoS	An investigation into the effect of network reconfiguration on the existing demands and whether the settled state delivered QoS is approached suddenly or approached smoothly.

Table 30: q-BGP Stability Test Suites

6.5 Scalability Experiments

These experiments aim to examine the effect of AS topology size and connectivity on the amount of q-BGP messaging and q-FIB and q-RIB sizes.

6.5.1 Performance Metrics

Performance Metrics	
Number of q-BGP messages (numqBGPmsgs)	This is the total number of q-BGP messages sent since network initialisation. The delta of this value is the size of the messaging avalanche caused by a change in the network.
Routing Table Sizes *(numqFIBentries, numqRIBentries)*	This is the number of entries contained in the q-FIBs and q-RIBs. Due to the possible decrease in aggregatability caused by the addition of the QoS attribute it may be the case that q-FIB and q-RIB sizes increase.

Table 31: q-BGP Scalability Testing Performance Metrics

6.5.2 Uncontrolled Variables

Uncontrolled Variables	
pSLS over-provisioning factor (pSLSOverFactor)	The pSLS over-provisioning as chosen by the efficacy tests so as to provide enough pSLS capacity, but not massively over-provision so that any route could be taken.
Demand topology(demand_min, demand_max)	As described in section 6.2.4 demand_min is 1 and demand_max is 100.

Table 32: q-BGP Scalability Testing Uncontrolled Variables

6.5.3 Controlled Variables

Controlled Variables	
Network Topology	Especially network size, but also average connectivity is very significant.
q-BGP Policies and strategies	Specifically which QoS-attributes are used, as they will inhibit the ability to aggregate q-BGP messages.

Table 33: q-BGP Scalability Testing Controlled Variables

6.5.4 Experimentation Environment

This is a set of very computationally expensive tests as they investigate the effect of scale on the magnitude of q-BGP messaging. The number of q-BGP messages isn't intrinsically increased due to the addition of QoS, comparing to a non QoS enabled BGP, as we are only considering a single meta-class (for all meta-classes the total q-BGP messages are expected to scale approximately with the number of metaclasses). The addition of QoS information however could inhibit the ability to aggregate messages and it is this that these experiments mainly aim to examine. The IP prefix generator plays a very significant role as it provides an IP prefix distribution across the network which at least ensures that some aggregation is possible, rather than just randomly allocating prefixes (see Section 6.2.5). The controlled variables for this test suite not only include the network size which is very important, but also the average connectivity because it influences the number of alternative paths, and hence messages.

6.5.5 Test Campaigns

Test Suite Id	Objective
qBGP/Scalability/countqBGPmsgs	An examination of the effect q-BGP strategies and network topology on the number of q-BGP messages (and therefore implicitly q-BGP aggregation).

Table 34: q-BGP Scalability Test Suites

7 SLS INVOCATION HANDLING TESTING

This Section specifies the tests that will be conducted on the SLS Invocation Handling software that implements the SLS Invocation Handling functions defined in [D1.2] Section 4.8.

The tests cover the following three Invocation Handling algorithms:

- Intra-domain cSLS;
- Inter-domain cSLS;
- Inter-domain pSLS.

7.1 Objectives

7.1.1 Functional Tests

The Functional Tests confirm that the SLS Invocation Handling software meets its specification, i.e. that the SLS Invocation Handling algorithms implement the functions described in [D1.2] Section 4.8.5, and meet the test requirements that were defined in outline in Section 4.8.6.

The SLS Invocation Handling software runs for a particular AS. In summary, the functional tests are intended to confirm that the software correctly collects and manipulates the required measurement data, implements the algorithms for the intra/inter-domain cSLS case and for the inter-domain pSLS case and is able to generate the topology and the test traffic with the required volume and characteristics.

7.1.2 Algorithm Benefit/Cost Performance Tests

The Algorithm Benefit/Cost Performance Tests assess the benefits and costs of each algorithm that the SLS Invocation Handling software will implement. These tests include the test requirements that were defined in outline in [D1.2] Section 4.8.6.

The proposed algorithms for intra/inter-domain cSLS and inter-domain pSLS will be assessed in terms of the relevant performance metrics so as to enable their principal parameters to be optimised. In addition to that, the performance of the algorithms will be compared against representative algorithms from the literature.

7.1.3 Scalability Tests

The all-in-one nature of the ns-2 simulator, i.e. the same test platform both generates the test traffic and the invocation requests and implements the invocation handling algorithm, means that ns-2, does not allow for an objective computational performance evaluation in means of computer processing time and memory requirements of the implemented algorithms. Therefore no scalability tests will be considered for the SLS Invocation Handling software.

7.1.4 Stability Tests

The Stability Tests verify that the SLS Invocation Handling algorithms are stable, under increasing traffic load conditions i.e. increasing rate of SLS invocation requests. Since the SLS Invocation Handling algorithms are measurement-based, an increase in the experienced packet loss rate for increasing loading conditions is anticipated. This is because since they rely on measurements, every new invocation request carries the potential of making a wrong decision. That means that a high invocation rate is expected to have a negative effect on performance. The Stability Tests evaluate the extent of the deterioration of the performance of the algorithms as a result of the increasing rate of SLS invocation requests.

In addition to that, the stability of the algorithms will be compared against representative algorithms from the literature. The tests include the test requirements that were defined in outline in [D1.2] Section 4.8.6.

7.2 Performance Metrics

For each algorithm the following performance metrics will be used, as listed in [D1.2] Section 4.8.6:

Performance Metrics
Loss, delay and jitter (real-time traffic)
Throughput/Goodput (elastic traffic)
QC utilization (all types of traffic)

Table 35: SLS Invocation Handling Performance Metrics

7.3 Controlled and Uncontrolled Variables

The controlled variables of the SLS Invocation Handling software are those variables that affect the algorithm performance. These vary depending on the test being conducted, but in general include:

Controlled Variables
Algorithm / formula for calculation of the admission control criterion
Algorithm parameters

Table 36: SLS Invocation Handling Controlled Variables

The uncontrolled variables are as follows:

Uncontrolled Variables	
Network topology	number of peer domains and inter-domain link topology for inter-domain SLS
QoS Capabilities of this domain	packet loss rate of e-QCs and l-QCs
Volume and characteristics of traffic flows	VoIP, Videoconference and elastic traffic

Table 37: SLS Invocation Handling Uncontrolled Variables

7.4 Experimentation Environment

The test platform will be the SLS Invocation Handling software developed by UniS using the Network Simulator (ns-2). The algorithms will be implemented in oTCL, which is the interface language of the simulator. The topology, which will be different for the intra-domain cSLS invocation handling case, for the inter-domain cSLS invocation handling case and the inter-domain pSLS invocation handling case, as well as the traffic flows, will be generated using the built-in functions of the simulator. For the intra-domain cSLS case, a standard dumbbell topology will be used (Figure 7).

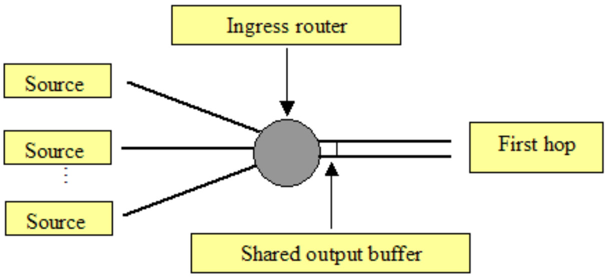


Figure 7: Simulation Topology for the Intra-domain cSLS Case

For the inter-domain case, the topology will be extended to include the inter-domain links.

Figure 8 shows a simple example topology for the inter-domain cSLS case.

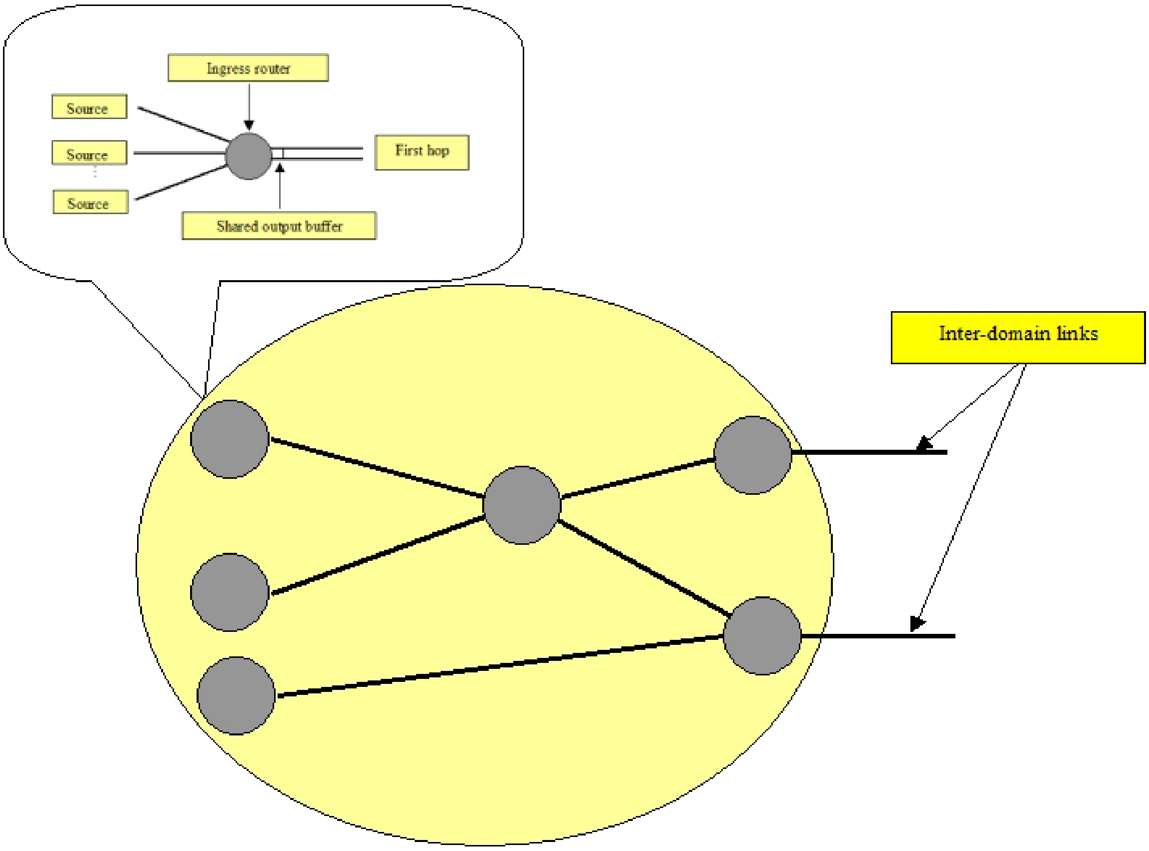


Figure 8: Example Simulation Topology for the Inter-domain cSLS Case

7.5 Test Campaigns

Test Id	Purpose	Description
SLSInv/Funct/1	Topology generation functions	Confirm the topology is created correctly for the intra/inter-domain cSLS and the inter-domain pSLS case.
SLSInv/Funct/2	Test traffic generation functions	Confirm test traffic meets the specified volume and characteristics (VoIP, Videoconference, elastic traffic)

SLSInv/Funct/3	Measurement data collection and manipulation functions	Confirm the measurement data are collected and manipulated correctly (storing of measurement samples and calculation of first and second order statistics).
SLSInv/Funct/Algorithm/1	Intra-domain cSLS invocation handling algorithm functions	Confirm that intra-domain cSLS invocation handling algorithms are implemented correctly (passing of parameters between the various components, calculation of PF and Cest).
SLSInv/Funct/Algorithm/2	Inter-domain cSLS invocation handling algorithm functions	Confirm that inter-domain cSLS invocation handling algorithms are implemented correctly (passing of parameters between the various components, calculation of PF and Cest).
SLSInv/Funct/Algorithm/3	Inter-domain pSLS invocation handling algorithm functions	Confirm that inter-domain pSLS invocation handling algorithms are implemented correctly (passing of parameters between the various components, calculation of PF and Cest).
SLSInv/Perf/1	Performance assessment of the intra-domain cSLS invocation handling algorithms	Assess the performance of the intra-domain cSLS invocation handling algorithms and optimise their principal parameters εref and PF.
SLSInv/Perf/2	Performance assessment of the inter-domain cSLS invocation handling algorithms	Assess the performance of the inter-domain cSLS invocation handling algorithms and optimise their principal parameter εref and PF. Assess the influence of inter-domain link topology on the performance of the algorithm.
SLSInv/Perf/3	Performance assessment of the inter-domain pSLS invocation handling algorithms	Assess the performance of the inter-domain pSLS invocation handling algorithms and optimise their principal parameters εref and PF. Assess the influence of inter-domain link topology and number of peering domains on the performance of the algorithm.
SLSInv/Perf/4	Performance comparison of algorithms	Assess the performance of the algorithms against representative algorithms from the literature
SLSInv/Stab/1	Stability of the intra-domain cSLS invocation handling algorithms	Investigate stability performance of the intra-domain cSLS invocation handling algorithms as function of traffic load.
SLSInv/Stab/2	Stability of the inter-domain cSLS invocation handling algorithms	Investigate stability performance of the inter-domain cSLS invocation handling algorithms as function of traffic load.
SLSInv/Stab/3	Stability of the inter-domain pSLS invocation handling algorithms	Investigate stability performance of the inter-domain pSLS invocation handling algorithms as function of traffic load.
SLSInv/Stab/4	Stability comparison of algorithms	Compare stability performance of the algorithms against representative algorithms from the literature.

Table 38: SLS Invocation Handling Tests

8 OFF-LINE INTRA-DOMAIN MULTICAST TE TESTING

This section specifies the tests that will be conducted on the simulation software that implements the Offline Multicast Traffic Engineering (OMTE) and Dynamic Multicast Routing (DMR) functions defined in [D1.2] Section 2.8. Specifically, we will evaluate the performance of the proposed Genetic Algorithm (GA) based TE algorithm in both dimensioning phase and invocation phase, as well as per-DSCP tree vs. hybrid tree routing in DiffServ environment.

8.1 Objectives

8.1.1 Functional Tests

This set of tests will investigate the functional validity of the proposed offline multicast TE algorithms.

8.1.2 Algorithm Benefit/Cost Performance Tests

This set of tests will investigate the performance of the proposed offline multicast TE algorithms. As we indicate in [D1.2], the objective of the proposed OMTE algorithm is to improve the network service capability and efficiency in resource utilisation with the QoS requirements from end users. Towards this end, our simulation is to evaluate the performance of block rate of group joins, traffic distribution and total consumption of network resources in comparison to conventional approaches. Performance comparison will be carried out between our proposed algorithms and existing IP/MPLS based solutions (shortest hop routing, random link weight setting, Steiner tree etc.).

8.1.3 Scalability Tests

This set of tests will investigate the scalability aspects of the proposed multicast TE algorithms. Here we classify two types of Scalability: (1) Scalability of the algorithm (e.g., computing overhead etc.) and (2) scalability of the configuration (router memory overhead etc). In our experiment, for OMTE-GA we will investigate the running time of the proposed algorithm; For DMR related simulation, we will focus on the router's overhead of multicast group and QoS states maintenance in both Per DSCP tree routing and hybrid tree routing.

8.1.4 Stability Tests

The stability tests verify that the proposed OMTE-GA algorithms are stable when the resulting solutions are configured within the network.

8.2 Performance Metrics

8.2.1 Functional Tests

The performance metrics of our simulation in OMTE can be divided into two categories: (1) offline dimensioned performance and (2) real-time performance after mSLS invocation. These two categories correspond to the two simulation models that will be used in the multicast TE tests respectively. The first type of experiment refers to the anticipated performance assuming that the forecast multicast traffic matrix is accurately activated. The second type of test will generate a scenario/instance of real time traffic invocation, with possible variations to the original traffic matrix, and evaluate the network behaviour according to these different dynamics.

Performance Metrics	
Offline dimensioned performance metrics	*Dimensioned bandwidth consumption*
	Probability of finding feasible solutions
	Traffic distribution in time of congestion
Real-time performance after mSLS invocation	*Block rate of group joins*

Table 39: Off-line Intra-domain Multicast TE Performance Metrics

8.2.1.1 *Offline Dimensioned Performance*

1. Dimensioned bandwidth consumption.

As we have specified in [D1.2], one basic objective of OMTE is to conserve bandwidth resources through Steiner tree based routing. By proper M-ISIS link weight setting, the resulted multicast trees will consume lower bandwidth resources than traditional methods (e.g., hop count routing). To this end, we will evaluate through experiments on how much bandwidth resource can be saved compared to conventional approaches.

2. Probability of finding feasible solutions.

Another important task of OMTE is to increase the network service capability, i.e., to accommodate as many group sessions as possible. When different multicast traffic matrices are derived with end-to-end QoS demands, what we are especially interested in is how many of these instances can be fully accommodated through our proposed OMTE algorithm, while they cannot be handled by conventional approaches?

3. Traffic Distribution in time of congestion.

It should be noted that, it might be the case that no feasible solution exists at all, if external traffic demand exceeds a certain threshold. In this scenario, our focus will shift to the scale of network congestions. In order to evaluate this feature, our simulation will address two inter-related performance matrices, namely proportion of congested links and the utilisation of the most congested link. By studying these two metrics, we can get a clear view on the scope and scale of the anticipated network congestion.

8.2.1.2 *Real-time Performance*

It is also important to evaluate the actual network condition after individual mSLSs are invoked. This real-time performance might be different from the dimensioned one since the actual activation of group joins and leavings are not always consistent to the forecasted traffic matrix. Typically, it is unlikely that all group members for each session are simultaneously joined. In this case, the real-time performance should be better than, if not the same as the dimensioned one. For example, when the dimensioned performance indicates some link congestions based on the derived mSLS, it might not take place at all when individual group sessions are actually invoked. On the other hand, this might not be the case if the behaviours of external sources or group members are different from what has been forecasted.

In our real-time simulation, we mainly study the metric of block rate of group joins. First of all, we will create a series of events (either group join or leaving) with respect of the original mSLS. Thereafter we will calculate how many group joins are rejected due to the real-time network congestions. Of course, it is anticipated that if OMTE has already achieved non-congestion in network dimensioning, then there will be no blocked calls in the result of real-time simulation, if both sources and group members conform to the original mSLS.

8.2.2 Scalability Tests

The major performance metrics in the scalability test of OMTE-GA is the computing overhead when large-scale network and complex multicast traffic matrix are given. The performance metrics in DMR refers to the configuration overhead in multicast group states and QoS states maintained within the network when per DSCP tree routing and hybrid tree routing are applied.

8.2.3 Stability Tests

In this set of tests we will focus on the real-time performance (after invocation) defined in section 8.2.1.2 with traffic variations from the original multicast traffic matrix. That is to say, we will on purpose make the scenario of invocated multicast traffic inconsistent to what has been forecasted, and see whether the proposed solution is still able to guarantee the desired behaviour stably.

8.3 Controlled and Uncontrolled Variables

The following are controlled variables in our functional simulation tests.

Controlled Variables		
GA related control parameters	*Population size (P)*	100-500
	Maximum generation (M)	100-500
	Crossover threshold (K_C)	0.3
	Mutation threshold (K_M)	0.01

Table 40: Off-line Intra-domain Multicast TE Controlled Variables

The uncontrolled parameters are list below:

Uncontrolled Variables
Network topology (number of routers, edge routers etc)
QoS capability of the network (number of QCs and associated bandwidth allocation)
Total number of multicast sessions involved.
Group membership and bandwidth demand: The bandwidth demands are purely random generated within the scope of the maximum demand.
Group density: This parameter is only used in real-time performance study. It indicates the instant fraction of joined members within each group. This value is within the range (0, 1).
Group join/leaving series: This series of events is randomly created with respect to the group density of each multicast session.
QC subscribed: Each multicast group join randomly select one of the QCs as their desired service classes.

Table 41: Off-line Intra-domain Multicast TE Uncontrolled Variables

8.4 Experimentation Environment

The simulations to be conducted are classified into routing level and packet level based experiments. The routing level and packet level simulations are based on Stanford Graph Base (SGB) and Network Simulator (ns-2) respectively. For the functional tests we manually create small-scale network topology with simple multicast traffic matrix.

Same as the functional test scenario, in Benefit/Cost Performance Tests our routing level and packet level simulations are still based on Stanford Graph Base (SGB) and Network Simulator (ns-2) respectively. However, we change to large-scale network topology and more complicated multicast traffic matrix for testing the actual performance of the proposed scheme. Typically the GT-ITM topology generator will be used to create random network with medium/large scale network models. These settings are used for scalability and stability tests as well.

8.5 Test Campaigns

Test Id	Purpose	Description
McastTE/Funct/OMTE-GA/1	The objective of the functional tests is to verify the validity of the proposed GA based offline multicast TE algorithm.	We use small-scaled network model with simple traffic matrix for testing the proposed genetic algorithm. The testing metrics studied are offline dimensioned variables we specified in section 0
McastTE/Funct/DMR/1	The objective of the functional tests is to verify the validity of the proposed OMTE-GA driven DMR at routing level.	For testing the validity of the routing level DMR, we use the same network model and derive the group dynamics from the multicast TM used in OMTE-GA. The result of the DMR should be consistent to that of OMTE-GA. This test will be conducted under both mono-QC and multi-QC (i.e., DiffServ) environment. The testing metrics studied are real-time performance we specified in section 8.2.1.2
McastTE/Funct/DMR/2	The objective of the functional tests is to verify the validity of the proposed DMR algorithm at packet level.	This packet level DMR will be tested using ns-2. We manually set up small network for evaluation of packet level performance of the hybrid tree approach and per DSCP tree approach. The testing metrics studied are real-time performance we specified in section 8.2.1.2
McastTE/Perf/OMTE-GA/1	The task of this performance test of OMTE-GA is to find the best controlled variables (i.e., GA parameters) for obtaining the optimal testing performance.	We use large-scaled network model created by GT-ITM with complex traffic matrix for evaluation of the proposed genetic algorithm based OMTE. We will iteratively try different controlled variables and find the most proper value for the next performance test.
McastTE/Perf/OMTE-GA/2	The task of this performance test of OMTE-GA is to evaluate the cost benefit achieved from the algorithm in comparison to the conventional solutions.	We use large-scaled network model created by GT-ITM with complex traffic matrix for evaluation of the proposed genetic algorithm based OMTE. Performance study will be carried out with the comparison of conventional based solutions, namely: Random link weight setting, shortest hop routing, as well as Steiner tree based solutions that need the help of MPLS tunnelling.
McastTE/Perf/DMR/1	The objective of this performance test is to evaluate the gain obtained from the proposed OMTE-GA driven DMR at routing level.	For testing the performance of the routing level DMR driven by OMTE-GA, we use the same network model as test McastTE/Perf/OMTE-GA/2, and derive the group dynamics from the multicast TM used in OMTE-GA. The result of the DMR should be consistent to that of OMTE-GA. This test will be conducted under both mono-QC and multi-QC (i.e., DiffServ) environment.

McastTE/Perf/DMR/2	The objective of this performance test is to evaluate the inter-QC fairness when the hybrid tree and per DSCP tree approaches are used.	This packet-level DMR will be tested using ns-2. Investigate performance using the same scenario of test McastTE/Perf/DMR/1.
McastTE/Scal/OMTE-GA/1	The major task of this experiment is to evaluate the running time when large scaled network and traffic matrices are given.	Same as the scenario of the performance test in McastTE/Perf/OMTE-GA/1.
McastTE/Scal/DMR/1	The major task of this experiment is to evaluate the scalability issues in group-state and QoS-state maintenance in multi-QC aware routing.	Same as the scenario of the performance test in McastTE/Perf/OMTE-GA/2.
McastTE/Stab/OMTE-GA/1	The purpose of this test is to evaluate the behaviour of the network under the configuration obtained from the proposed OMTE-GA when the actual multicast traffic has some level of variation from the original forecast one.	We will on purpose change the triggered multicast traffic from external group members that are not completely consistent to the original traffic matrix. We will check whether the initial configuration is still able to yield good performance constantly.

Table 42: Off-line Intra-domain Multicast TE Tests

PART – II: TESTBED-BASED EXPERIMENTATION

9 OVERVIEW

The purpose of this section is to describe the tests to be undertaken in the FTR&D testbed. Experimentation that will be undertaken in the FTR&D testbed will focus mainly on q-BGP and PCS implementations. The tests to be carried out are divided into the following types:

- Validation Tests. The emphasis of the Validation Tests is to prove the correct functionality of the protocols/components under test.
- System Level Performance Assessment Tests.

MESCAL testbed experimentation will be done incrementally. Validation Tests and System Performance Assessment Tests have been grouped as follows:

Phase 1) this phase is a baseline for tests that will be carried in further phases. Routing (reachability, BGP configuration, etc) and QoS features (like QoS classes, shaping, policing, etc) will be tested.

Phase 2) for Loose Guarantees Solution Option (**LSO**). q-BGP will be integrated in the platform; functionality and proof of concept tests for the q-BGP implementation, as well as q-BGP and system-level performance tests.

Phase 3) for Hard Guarantees Solution Option (**HSO**). PCS will be integrated in the platform; experimentation focuses on PCS implementation and its conformity to specification detailed in [D1.2] emphasising on the validity of routes computed with desired performance targets.

Each test group is composed of test suites, which are composed of elementary tests. System level elementary tests are described in detail in section 11.2.3.2 and 12.2.3.2. Elementary tests for validation tests are described in detail in Appendix B.

The test id is defined as: *group id / test suite id / test number*.

The group identifications are as follows:

Group Id	Description
TB_P1_Funct	Groups tests related to phase 1: initial testbed setup
TB_P2_Funct	Groups tests related to phase 2: q-BGP testing
TB_P2_Sys	Groups system level tests related to phase 2
TB_P3_Funct	Groups tests related to phase 3: PCS testing
TB_P3_Sys	Groups system level tests related to phase 3

Table 43: Testbed-Based Test Groups

Test suites will be defined in each relevant section.

9.1 System Performance Assessment Testing

The purpose of the MESCAL system level tests is to determine whether the objectives of the MESCAL system have been realised. The MESCAL System Performance Assessment tests have been classified into four categories for each phase: Benefit/Cost, Scalability, Stability and Usability. Table 44 indicates which tests categories are proposed for each test group and will be conducted in the testbed. Scalability tests will only be conducted in simulation environment.

	Phase 1	Phase 2	Phase 3
Benefit/Cost	✔	✔	✔
Scalability	X	X	X
Stability	✔	✔	✔
Usability	✔	✔	✔

Table 44: Test Categories for MESCAL System in the Testbed Environment

9.1.1 Benefit/Cost

The Benefit/Cost tests assess what improvement to network operation is attributable to the MESCAL system, along with a measure of the cost that is incurred in providing the benefit. The Benefit/Cost are measured through specific metrics in a representative set of network configurations and cases. For example, by measuring QoS-related metrics the benefits can be assessed and we expect that a traffic-engineered network should provide better QoS than a conventional best effort network.

MESCAL has a two level approach to Traffic Engineering that combines both offline TE and dynamic TE techniques. Off-line Inter-domain TE provides a mapping of traffic demands onto the network's physical topology, therefore, the resulting configuration may be compared against other alternatives.

Inter-domain dynamic TE reacts to the network state and takes action to maintain the contracted QoS performance (e.g., re-route the traffic). Inter-domain dynamic TE (i.e., q-BGP) aims to offer discrete inter-domain QoS-enabled Planes (parallel QoS Internets), but at the cost of additional complexity, so there are tests to assess the overall cost associated with the use of a q-BGP-based solution.

9.1.2 Scalability

Scalability defines the ability of a system to be deployed and used at large scale, whatever the criteria that define the scale. Scalability in QoS-enabled IP networks has at minimum two aspects: size of network topology and the number and granularity of classes of service supported. Network topologies are characterized by a number of parameters, such as number of nodes and links, speed of links, degree of physical and logical connectivity, network diameter, etc. In QoS-enabled IP networks, supported services are mapped to a number of classes according to the DiffServ model. The scalability of the MESCAL system is the ability of deploying such a system at the scale of large IP networks offering a number of services.

Scalability tests assess the behaviour of the system as certain relevant parameters increase in size. These tests verify that the MESCAL solutions work at a variety of scales. These tests will be conducted in a simulated environment (see Part I sections 3-8).

9.1.3 Stability

Stability assessment tests verify that the system, given its specified dynamics/responsiveness, is operating in a stable state, in a representative set of network and traffic cases.

The stability tests assess the behaviour and performance of the system when subjected to these network stimuli. It will not be possible to conclusively prove stability by such tests, but it should be possible to identify any severe instability. Stability is explicitly related to behaviour of dynamic components of the system and the feedback loops. From this viewpoint, the purpose of these tests is to verify that the expected behaviour is realised without undue oscillations.

As it was stated before, MESCAL has a two level approach to Traffic Engineering that combines both offline TE and dynamic TE techniques. For the off-line TE sub-system, there is little scope for instability, as the dimensioning is done offline. For the Dynamic TE sub-system, q-BGP is a potential source of instability as it reacts to network events, which could result in unstable operation. Changes or fluctuation in BGP/q-BGP routes can cause performance degradation of user traffic, increase processing overheads on routers, and change the user traffic load distributions over the network. q-

BGP route changes happen for a variety of reasons, including link failure, router failure, changes in routing policies, or any reconfigurations that may trigger the closing of an active q-BGP session, forcing each router to withdraw the routes learned from its neighbour. Topology changes may cause some routers to select new q-BGP routes and advertise them to neighbouring ASes. The new advertisements from neighbouring ASes may change the paths that traffic takes through the network. This can cause congestion on certain links in the AS.

q-BGP routing changes can cause performance problems. A link failure can trigger a sequence of update messages as the routers explore alternative paths. Exchanging and processing updates messages also consumes bandwidth and CPU resources on the BGP peers. When routing changes occurs, the q-BGP processes could experience high convergence delay and insert a large number of update messages. Any excessive convergence delay introduces high packet delay and high packet loss to the affected services and destinations. During the q-BGP convergence period, packets may be caught in forwarding loops, leading to increased packet delay/loss.

9.1.4 Usability

The implementation of MESCAL in the WP3 testbed is a "proof of concept" and therefore many of the usability criteria that are associated with "off the shelf" products, such as ease of use, quality of documentation, etc. are inappropriate. The Usability tests are therefore carried out to assess the operational costs of the solution in terms of configuration operations and to demonstrate that the system can operate as expected (according to its functional objectives).

10 TESTBED PHASE-1 EXPERIMENTATION

10.1 Objectives

These tests are carried out to verify that the network is set-up and operates correctly for conducting the tests in Phase 2.

The objectives of the phase one experimentation are to verify that routing and QoS configuration detailed in Appendix A are correctly deployed. In addition, these experimentations aim at verifying if the deployment of policing and shaping are correctly configured in all ASes.

System level performance tests in phase 1 are carried out as baseline tests for comparison purposes. These tests are included in section 11.2.

10.2 Controlled and Uncontrolled Variables

The controlled variables of these tests are variables that affect the sanity of the testbed, especially:

Controlled Variables
Routing configuration parameters
QoS Configuration parameters
Bandwidth values
Algorithms for DSCP marking

Table 45: Phase 1 Controlled Variables

The uncontrolled variables are:

Uncontrolled Variables
Link failures
Process failures
Traffic flows

Table 46: Phase 1 Uncontrolled Variables

10.3 Experimentation Environment

The environment to execute these tests is the MESCAL testbed that is deployed in FTR&D premises. The configuration of the testbed for this phase is detailed in Appendix A. This configuration will be used as it is for executing these tests except there are explicit recommendations in the procedure tag.

10.4 Test Campaigns

10.4.1 Test Suite Structure

TB_P1_FUNCT group contains the following test suites:

Test Suite Id	Objective
TB_P1_FUNCT/ROUT	This group of test aims at verifying the routing features, especially the activation of BGP and reachability aspects.
TB_P1_FUNCT/DSSW	This group of tests aims at verifying the DSCP swapping operations in ingress and egress of a domain.
TB_P1_FUNCT/SHAP	This group test aims at verifying shaping operation.
TB_P1_FUNCT/POLI	This group of test aims at verifying policing issues.
TB_P1_FUNCT/BWMA	This group of test aims at examining the bandwidth management.

Table 47: Phase 1 Test Suites

10.4.2 Elementary Tests

TB_P1_FUNCT group contains the tests in Table 48 (for a detailed description see section 14.1).

Test Id	Purpose
TB_P1_FUNCT/ROUT/1	Validate inter-domain link connectivity.
TB_P1_FUNCT/ROUT/2-12	Validate connectivity between two neighbors when q-BGP process is activated.
TB_P1_FUNCT/ROUT/13	Check the route propagation in a simple scenario.
TB_P1_FUNCT/ROUT/14	Check the reachability of all interfaces.
TB_P1_FUNCT/ROUT/15	Verify reachability status when link failure occurs.
TB_P1_FUNCT/ROUT/16	Verify reachability status when a link failure is re-established.
TB_P1_FUNCT/ROUT/17	Verify intra-domain routing in AS4 domain.
TB_P1_FUNCT/DSSW/1-10	Verify DSCP swapping at egress of MESCAL11, MESCAL21, MESCAL31, MESCAL41, MESCAL42, MESCAL43, MESCAL51, MESCAL61, MESCAL71 and MESCAL81.
TB_P1_FUNCT/DSSW/11-20	Verify DSCP swapping at ingress of MESCAL11, MESCAL21, MESCAL31, MESCAL41, MESCAL42, MESCAL43, MESCAL51, MESCAL61, MESCAL71 and MESCAL81.

TB_P1_FUNCT/DSSW/21	Verify QoS configuration of the whole testbed.
TB_P1_FUNCT/SHAP/1-11	Verify shaping configuration in MESCAL11, MESCAL71, MESCAL81, MESCAL51, MESCAL43, MESCAL41, MESCAL42, MESCAL21, MESCAL31and MESCAL61.
TB_P1_FUNCT/POLI/1-11	Verify policing configuration in MESCAL11, MESCAL71, MESCAL81, MESCAL51, MESCAL43, MESCAL41, MESCAL42, MESCAL21, MESCAL31and MESCAL61.
TB_P1_FUNCT/BWMA/1-11	Verify bandwidth management configuration in MESCAL11, MESCAL71, MESCAL81, MESCAL51, MESCAL43, MESCAL41, MESCAL42, MESCAL21, MESCAL31and MESCAL61.

Table 48: Phase 1 Tests

11 TESTBED PHASE-2 EXPERIMENTATION

11.1 Validation Tests

11.1.1 Objectives

The objectives of this phase 2 experimentations are as follows:

- Test the q-BGP message conformance with what has been specified in [D1.2];
- Validate QoS computation as implemented by q-BGP machinery;
- Validate the route selection process;
- Validate DSCP swapping operations as implemented in q-BGP (especially validate the QoS route-map introduced in ZeboS).

11.1.2 Controlled and Uncontrolled Variables

Controlled variables are in part as follows:

Controlled Variables
Q-BGP parameters: route-maps; local-QoS-classes, policies, etc.
Priority associated with each QoS information

Table 49: Phase 2 Validation Controlled Variables

The uncontrolled variables are as follows:

Uncontrolled Variables
Traffic matrix
Link failures
Operating systems failures

Table 50: Phase 2 Validation Uncontrolled Variables

11.1.3 Experimentation Environment

The environment to execute these tests is the MESCAL testbed that is deployed in FTR&D premises. The configuration of the testbed for this phase is detailed in Appendix A. This configuration will be used as it is for executing these tests except there are explicit recommendations in the procedure tag.

When generating traffic from Smartbit chassis, it is recommended to differentiate source and destination ports per stream. This is a proposal for values to assign per meta-QoS-class stream:

- Destination port = 5001, Source port=9991
- Destination port = 5002, Source port=9992
- Destination port = 5003, Source port=9993
- Destination port = 5004, Source port=9994

11.1.4 Test Campaigns

11.1.4.1 Test Suite Structure

TB_P2_FUNCT group contains the following test suites:

Test Suite Id	Objective
TB_P2_FUNCT/CMES	This group of test aims at verifying the conformance of q-BGP messages such as defined in [D1.2].
TB_P2_FUNCT/DSCP	This group of tests aims at verifying DSCP swapping operations for ingress and egress.
TB_P2_FUNCT/QCMP	This group tests aims at verifying basic computation of QoS information between two peering ASes.
TB_P2_FUNCT/RSEL	This group of tests aims at verifying the behaviour of q-BGP route selection algorithm such as defined in [D1.2]. It also verifies more complex computation of QoS information.
TB_P2_FUNCT/QFIB	This group of tests aims at verifying that QoS-enabled entries are correctly installed in FIB via q-BGP.
TB_P2_FUNCT/INT	This group of tests aims at verifying the interoperability of q-BGP and BGP.

Table 51: Phase 2 Validation Test Suites

11.1.4.2 Elementary Tests

TB_P2_FUNCT group contains the tests in Table 52 (for a detailed description see section 14.2).

Test Id	Purpose
TB_P2_FUNCT/CMES/1	Verify the capability length.
TB_P2_FUNCT/CMES/2	Verify the QoS service capability field length.
TB_P2_FUNCT/CMES/3	Verify that Group 1 QoS service capability is supported.
TB_P2_FUNCT/CMES/4	Verify that Group 2 QoS service capability is supported.
TB_P2_FUNCT/CMES/5	Validate the conformance of QoS information length.
TB_P2_FUNCT/CMES/6	Verify that "Packet Rate QoS Code" and its associated Sub-codes are supported.
TB_P2_FUNCT/CMES/7	Validate that "One Way Delay QoS Code" and its associated Sub-codes are supported.
TB_P2_FUNCT/CMES/8	Validate that "Inter-Packet Delay Variation QoS Code" and its associated Sub-codes are supported.
TB_P2_FUNCT/CMES/9	Validate the QoS information value.
TB_P2_FUNCT/CMES/10	Verify that QoS class identifier can be set to a value that is between 0 and 63.

TB_P2_FUNCT/CMES/11	Validate the QoS Origin field.
TB_P2_FUNCT/CMES/12	Check the validity of Address Family Identifier (AFI).
TB_P2_FUNCT/CMES/13	Check the validity of Subsequent Address Family Identifier (SAFI).
TB_P2_FUNCT/CMES/14	Check the validity of Network Address of Next Hop.
TB_P2_FUNCT/CMES/15	Verify the conformance of NLRI field.
TB_P2_FUNCT/DSCP/1-2	Validate that egress DSCP swapping operation is correctly achieved when receiving BGP UPDATE messages.
TB_P2_FUNCT/QCMP/1	Check that the *reserved-rate* QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/2	Check that invalid *reserved-rate* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/3	Check that the *available-rate* QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/4	Check that invalid *available-rate* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/5	Check that the *min-owd* (minimum one-way-delay) QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/6	Check that invalid *min-owd* values are rejected but the command-line interface.
TB_P2_FUNCT/QCMP/7	Check that the *max-owd* (maximum one-way-delay) QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/8	Check that invalid *max-owd* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/9	Check that the *average-owd* (average one-way-delay) QoS parameter is correctly computed by the receiving ASBR
TB_P2_FUNCT/QCMP/10	Check that invalid *average-owd* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/11	Check that the *loss-rate* QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/12	Check that invalid *loss-rate* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/13	Check that the *jitter* QoS parameter is correctly computed by the receiving ASBR.
TB_P2_FUNCT/QCMP/14	Check that invalid *jitter* values are rejected by the command-line interface.
TB_P2_FUNCT/QCMP/15	Check that the receiving ASBR is able to compute multiple QoS parameters contained in an announcement.
TB_P2_FUNCT/QCMP/16	Check that the receiving ASBR is able to compute multiple QoS parameters for a same prefix announced within different meta-QoS-planes.
TB_P2_FUNCT/RSEL/1	Check that several ASes involved in the loose service option are able to exchange route updates containing correctly computed QoS information.
TB_P2_FUNCT/RSEL/2	Check, in simple Scenarios, that the route selection process takes into account the priority level of each QoS attribute.
TB_P2_FUNCT/RSEL/3	Check that the route selection process takes into account the QoS attributes which have a lower priority when the previous attributes (with higher priority) are equivalent.
TB_P2_FUNCT/RSEL/4	Check that the precision command line parameter is correctly handled for the reserved-rate QoS attribute.

TB_P2_FUNCT/RSEL/5	Check that the precision command line parameter is correctly handled for the available-rate QoS attribute.
TB_P2_FUNCT/RSEL/6	Check that the precision command line parameter is correctly handled for the loss-rate QoS attribute.
TB_P2_FUNCT/RSEL/7	Check that the precision command line parameter is correctly handled for the min-owd QoS attribute.
TB_P2_FUNCT/RSEL/8	Check that the precision command line parameter is correctly handled for the max-owd QoS attribute.
TB_P2_FUNCT/RSEL/9	Check that the precision command line parameter is correctly handled for the average-owd QoS attribute.
TB_P2_FUNCT/RSEL/10	Check that the precision command line parameter is correctly handled for the jitter QoS attribute.
TB_P2_FUNCT/RSEL/11-17	Validate the behavior of q-BGP when mandatory parameters aren't received.
TB_P2_FUNCT/RSEL/18-24	Validate the behavior of q-BGP when optional parameters aren't received.
TB_P2_FUNCT/QFIB	A series of elementary tests will be carried out in order to verify the correct installation of QoS-based routes in the q-FIB table.
TB_P2_FUNCT/INT/1	Validate the behavior of a BGP speaker when receiving unrecognized optional parameters.
TB_P2_FUNCT/INT/2	Validate the behavior of a q-BGP speaker when receiving notification set to unsupported capabilities from BGP speaker.
TB_P2_FUNCT/INT/3	Validate the q-BGP router installs routes received from BGP speaker in best effort plane.
TB_P2_FUNCT/INT/4	Validate the BGP router installs routes received from q-BGP speaker.

Table 52: Phase 2 Validation Tests

11.2 System Performance Assessment Tests

11.2.1 Performance Metrics

Performance Metrics	
QoS metric, Delay (average or percentile) (Delay)	Average delay measured between two specified points per MC Plane.
QoS metric, Average loss probability (LossP)	Average loss ratio measured between two specified points per MC Plane.
Overall throughput (Throughput)	Overall throughput between two specified points per MC Plane.
Network utilisation (Utilisation)	Network utilisation per MC plane.
QoS outage ratio (Outage)	QoS outage to up-time ratio.
Overhead load (Overhead)	The control plane overhead imposed by q-BGP in terms of packet rate and traffic volume.
Lookup time (LookupT)	The q-RIB/q-FIB lookup time by the routing processes.
Lookup table size (LookupTS)	The q-RIB/q-FIB lookup table sizes.

Convergence time (ConvergenceT)	Convergence time is calculated from the time difference between occurrence of a failure and when the QoS-enabled route is re-established. This is the time taken by the routers of the network to get and process the BGP/q-BGP messages, to compute and select new route, to load their routing tables, and to enforce the route. This is subject to the assumption that alternative pSLSes are already in place.
Configurations enforcement time (ConfT)	Time to apply the configurations.
Configurations number (ConfN)	Number of configuration operations.

Table 53: Phase 2 System Level Performance Metrics

11.2.2 Controlled and Uncontrolled Variables

Controlled Variables	
QoS capabilities, *number of M-QCs (M-QC-num)*	QoS capabilities of each domain, number of M-QCs
QoS capabilities, *number of l-QCs (l-QC-num)*	QoS capabilities of each domain, number and values of l-QCs
frequency of network failures (FailureFrq)	The frequency of network failures.

Table 54: Phase 2 System Level Controlled Variables

Uncontrolled Variables	
traffic flows (Flows)	User traffic flows produced by Traffic Generators.
traffic load demand (Demand)	Traffic load demand.
network topology (Topology)	Network topology as specified in FTR&D testbed.
other	These tests depend on the q-BGP configuration e.g., the values associated to timers such as hold timers, scan timer. Results that could be obtained based on two different configurations will be different. Therefore, the same initial conditions must be applied on all network and q-BGP configurations.

Table 55: Phase 2 System Level Uncontrolled Variables

11.2.3 Test Campaigns

11.2.3.1 Test Suite Structure

TB_P2_SYS group is composed of the sub-groups shown in Table 56.

Test Suite Id	Objective
TB_P2_SYS/BC	This group of tests assesses the performance at the system level, including the creation of a baseline for comparison.
TB_P2_SYS/Stab	This group of tests assesses the stability of system under different conditions.
TB_P2_SYS/Usab	This group test assesses the usability of MESCAL system.

Table 56: Phase 2 System Level Test Suites

A number of tests are organised as shown in Table 57 for test groups in Phase 2. The baseline tests are the tests conducted in a similar environment to the current Internet (i.e., single Meta-QoS-Class

Plane). Further tests will be conducted by introducing one or more Meta-QoS-Class (MC) Planes and the results will be compared with the baseline tests. Two environments will be under investigation:

1. When the topological instability is low and generally the network is stable
2. When the topological instability is high and the network failures occur.

Network Configuration		MC0 (BE)		MC0 + MC1		MC0 + MC1 + MC2	
Topological Instability		Low	High	Low	High	Low	High
Tests	Purpose						
BC	QoS Performance	***Delay, LossP, …***					
	Offered Load	***Throughput***					
	Route Selection	***ConvergenceT***					
	q-RIB/q-FIB process	***LookupT*** ***LookupTS***					
	q-BGP Protocol	***Overhead***					
Stab	QoS maintained?	***Delay, LossP***					
	Stable states re-established?	***Outage***					
Usab	Operational cost	***ConfN***					

Table 57: Phase 2 System Level Test Campaigns

11.2.3.2 Elementary Tests

The elementary tests for LSO Phase 2 for performance assessments are shown in Table 58.

Test Id	Purpose	Description
TB_P2_SYS/BC/1	"Proof of Concept" for MC QoS Planes	Ad hoc configurations for network provisioning are used. Each configuration is planned to satisfy the QoS requirements of Meta-QoS-classes. The performance results from different configurations are checked to see if they conform with the MC plane QoS definitions.
TB_P2_SYS/BC/2	Performance comparison	The results obtained from configuration provided by the MESCAL off-line TE system are compared with the result from ad-hoc configuration.
TB_P2_SYS/BC/3	Qualify the q-BGP operational cost	The convergence time, table look-up time, and overhead related to q-BGP operation are measured and the results are analysed in a comparative manner.
TB_P2_SYS/Stab/1	To see whether the QoS is maintained after the establishment of new route.	q-BGP can be the potential source of instability as it reacts to network stimuli, which could result in unstable operation. Before and after the re-establishment of stable state the QoS performance values are measured.
TB_P2_SYS/Stab/2	To access the q-BGP stability and resiliency to perturbations.	The main aspects for the these stability tests are as follows: - How BGP process copes with failures under stress conditions. - Whether stable states are re-established between the failures.
TB_P2_SYS/Usab/1	Operational cost of managing the LSO	The benefit of having the MESCAL TE system to assist network provisioning will be examined.

Table 58: Phase 2 System Level Tests

11.2.3.2.1 TB_P2_SYS/BC/1

Objectives: The objective is to prove the concept of MC QoS Planes by conducting a series of tests in order to compare the performance achieved by different logical configurations. The objectives are to show that MESCAL logic in LSO Phase 2 works and it is beneficial in terms of the total demand that is served by the network while being able to achieve the required level of QoS.

The approach for network provisioning is the ad hoc configurations, where the administrator configures the network according to an ad-hoc policy decision. The administrator can calculate a network configuration by taking into account a predicted traffic demand. The configuration must satisfy the QoS requirements of Meta-QoS-Classes.

Performance metrics: *Delay*, *LossP*, *Throughput*, *Utilisation*

Controlled variables: *M-QC-num*, *l-QC-num*

Uncontrolled variables: *Demand*, *Flows*, *Topology*

Test campaigns: A series of tests, see Table 57, is conducted in order to compare the performance achieved by different logical configurations where:

I) Only the baseline configuration is considered (MC0). MC0 is for all traffic.

II) A new MC plane is introduced (MC1). MC0 is for BE Traffic and MC1 for better than BE traffic.

III) Two/more MC planes are introduced (MC0, MC1, MC2). MC0 is for BE Traffic, MC1 is for AF1 and MC2 is for premium traffic.

Comparisons should be made between the results obtained from various configurations on the basis of the above metrics.

11.2.3.2.2 TB_P2_SYS/BC/2

Objectives: Similar test campaigns to "TB_P2_SYS/BC/1"are conducted to measure the performance achieved by the configuration provided by the MESCAL off-line TE system. The results from different configurations provided by the MESCAL off-line TE system configurations will be compared with the results from the ad-hoc approach for network provisioning. The objectives are to show that MESCAL off-line TE system is beneficial in being able to satisfy the level of QoS, while utilising network resources efficiently.

Performance metrics: *Delay*, *LossP*, *Throughput*, *Utilisation*

Controlled variables: *M-QC-num*, *l-QC-num*

Uncontrolled variables: *Demand*, *Flows*, *Topology*

Test campaigns: A series of tests is conducted in order to compare the performance achieved by different logical configurations conducted in "TB_P2_SYS/BC/1" test campaigns. We expect that the configurations give similar performance figures to the ones in administratively fine tuned ad-hoc configurations.

11.2.3.2.3 TB_P2_SYS/BC/3

Objectives: Using q-BGP introduces additional costs. The amount of q-BGP messages injected into the network for QoS-based routing purposes should be controlled. The q-BGP load should be small if not, then the q-BGP traffic will affect the performance of the router and the network. The objective is to conduct a series of tests for measuring the costs associated with MESCAL system related to Phase 2. To qualify the q-BGP operational cost, these tests will rely upon the following measurements: convergence time, look-up time, and q-BGP overhead.

Performance metrics: *Delay*, *Overhead*, *ConvergenceT*, *LookupTS*, *LookupT*

Controlled variables: *M-QC-num*

Uncontrolled variables: *Topology*[4]

Experimentation environment: Given the reference topology, the topology related change could occur in terms of link failure, router failure, creation of a new pSLS, withdrawal of a pSLS, etc. This triggers the calculation of the above metrics.

Test campaigns: A series of similar tests is conducted in order to compare the measured metrics in terms of convergence time, look-up time, overhead load, and table sizes by different logical configurations when a single link failure is occurred and where:

I) Only the baseline configuration is considered (MC0).

II) A new MC plane is introduced (MC0, MC1).

III) Two/more MC planes are introduced (MC0, MC1, MC2).

These results can be used to extrapolate the trend when the network topology gets larger and the number of MC Planes is increased.

11.2.3.2.4 TB_P2_SYS/Stab/1

Objectives: In the "TB_P2_SYS/BC/3" tests we examined the convergence time under various network configurations. The purpose of the following tests is to verify that QoS is maintained after the establishment of a new route when an inter-domain route changes occurs.

Performance metrics: *Delay*, *LossP*, *Throughput*

Controlled variables: *M-QC-num*

Uncontrolled variables: *Topology*

Test campaigns: A series of similar tests to the test campaigns in "TB_P2_SYS/BC/2" is conducted in order to see whether the QoS is maintained after the new route is established. The QoS performance values will be measured in term of delay and loss after a new route is established and they will be compared with the results obtained in "TB_P2_SYS/BC/2".

11.2.3.2.5 TB_P2_SYS/Stab/2

Objectives: These tests are to assess the LSO is stable and resilient to perturbations. q-BGP is the potential source of instability as it reacts to network stimuli, which could result in unstable operation. The main aspects for the these stability tests are as follows:

- Speed of q-BGP reaction, otherwise the reaction may be too late to be effective
- Stable state must be re-established.

For q-BGP, the convergence time has to be as short as possible. At least two factors impact this convergence time: the number of events that trigger q-BGP and the q-BGP message propagation time across the network. The more often q-BGP messages are triggered, the worst the stability of the network is.

Performance metrics: *Outage*, *LossP*

Controlled variables: *M-QC-num*, *FailureFrq*

Uncontrolled variables: *Topology*

Test campaigns: A series of tests is conducted in order to observe:

1. How BGP process can cope with failures
2. Whether stable states are re-established between the failures.

[4] The network configurations must satisfy the QoS requirements of MC-QoS-classes.

11.2.3.2.6 TB_P2_SYS/Usab/1

Objectives: This test is to assess the operational cost of managing LSO and to show the complexity involved in configuring the testbed in ad-hoc fashion compared with automatic configuration performed through MESCAL Service/Management/Control Planes. This test will measure the benefit of having the MESCAL TE system to assist network provisioning.

Performance metrics: *ConfT*, *ConfN*

Controlled variables: *M-QC-num*

Uncontrolled variables: *Topology*

Test campaigns: These tests, see Table 57, are to demonstrate the complexity of manual configurations, modification to configurations, and the ease of configuring the network through the MESCAL system.

12 TESTBED PHASE-3 EXPERIMENTATION

12.1 Validation Tests

12.1.1 Objectives

The objectives of this phase 3 experimentations are as follows:

- Test the PCP message conformance with what has been specified in [D1.2]
- Validate the QoS computation as implemented by PCS machinery
- Validate the route selection process
- Validate the interface between q-BGP and PCS

12.1.2 Controlled and Uncontrolled Variables

Controlled variables are in part as follows:

Controlled Variables
PCS algorithm parameters.
Priority associated with each QoS information

Table 59: Phase 3 Validation Controlled Variables

The uncontrolled variables are as follows:

Uncontrolled Variables
Traffic matrix
Link failures
Operating systems failures

Table 60: Phase 3 Validation Uncontrolled Variables

12.1.3 Experimentation Environment

The environment to execute these tests is the MESCAL testbed that is deployed in FTR&D premises. The configuration of the testbed for this phase is detailed in Appendix A. This configuration will be used as it is for executing these tests except there are explicit recommendations in the procedure of the test.

12.1.4 Test Campaigns

12.1.4.1 Test Suite Structure

TB_P3_FUNCT group contains the following test suites:

Test Suite Id	Objective
TB_P3_FUNCT/CMES	This group of test aims at verifying message conformance of PCP to what is specified in [D1.2].
TB_P3_FUNCT/QAGG	This group of tests aims at verifying QoS aggregation operations as achieved by PCS entities.

Table 61: Phase 3 Validation Test Suites

12.1.4.2 Elementary Tests

TB_P3_FUNCT group contains the tests in Table 62 (for a detailed description see section 14.3).

Test Id	Purpose
TB_P3_FUNCT/CMES/1	Check the format of OPEN, CLOSE and ACCEPT messages.
TB_P3_FUNCT/CMES/2	Check the format of REQUEST, RESPONSE PATH-ERROR and ACKNOWLEDGE messages.
TB_P3_FUNCT/CMES/3	Validate the REQ-REFERNCE-ID and PATH-COMPUTATION-ID.
TB_P3_FUNCT/CMES/4	Validate QoS information contained in REQUEST-PATH message.
TB_P3_FUNCT/CMES/5	Validate QoS information contained in RESPONSE-PATH message.
TB_P3_FUNCT/CMES/6	Check the format of PATH-ERROR and messages.
TB_P3_FUNCT/CMES/7	Check the format of CANCEL and messages.
TB_P3_FUNCT/CMES/8	Check operational behaviors when receiving REQUEST messages.
TB_P3_FUNCT/QAGG/1	Check QoS aggregation operation.

Table 62: Phase 3 Validation Tests

12.2 System Performance Assessment Tests

12.2.1 Performance Metrics

Performance Metrics	
Delay (average) (Delay)	Average delay measured between two specified points per MC Plane.
Path calculation delay (ClcDelay)	Path calculation delay.
Overhead load (Overhead)	The control plane overhead associated with path computation.
Contiguous path requirement (ContiguousPReq)	The loose path must be a contiguous hop-by-hop (at the AS level) path.
Single MC plane path requirement (SingleMCPReq)	The loose path must satisfy the MC concept that a single MC Plane is used for the entire path.
QoS satisfaction requirement (QoSSatisfactionReq)	The performance target for the LSP based on the administrator settings should satisfy the QoS performance values requested by the customer.

Table 63: Phase 3 System Level Performance Metrics

12.2.2 Controlled and Uncontrolled Variables

Controlled Variables	
path endpoints (EndpointsP)	Path endpoints (i.e. number of PCSs involved in computation).
path QoS requirements (QoSReqsP)	Path QoS requirements.

Table 64: Phase 3 System Level Controlled Variables

Uncontrolled Variables	
network topology (Topology)	Network topology as specified in FTR&D testbed.

Table 65: Phase 3 System Level Uncontrolled Variables

12.2.3 Test Campaigns

12.2.3.1 Test Suite Structure

TB_P3_SYS group is composed of the following sub-groups:

Test Suite Id	Objective
TB_P3_SYS/BC	This group of tests assesses the performance at the system level, including the creation of a baseline for comparison.
TB_P3_SYS/Usab	This group test assesses the usability of MESCAL system.

Table 66: Phase 3 System Level Test Suites

12.2.3.2 Elementary Tests

The elementary tests for HSO for performance assessments are shown in Table 67.

Test Id	Purpose	Description
TB_P3_SYS/BC/1	"Proof of Concept" for HSO in finding end-to-end loose paths	To access whether the combination of q-BGP and PCS operations provide a valid Loose Path for LSP.
TB_P3_SYS/BC/2	Qualify the QoS performance for end-to-end LSPs	To access whether the combination of q-BGP and PCS operations meet the target performance figures for the LSPs.
TB_P3_SYS/BC/3	Performance Assessment of Path Computation Server	To assess the performance of the PCS, when operating in an end-to-end scenario e.g. computation times for calculation of LSP.
TB_P3_SYS/Usab/1	Operational cost of managing PCS	Similar tests as to "TB_P2_SYS/Usab/1"

Table 67: Phase 3 System Level Tests

12.2.3.2.1 TB_P3_SYS/BC/1

Objectives: This test assesses whether the combination of q-BGP and PCS operations can calculate valid "Loose Paths" for LSPs.

Performance metrics: *ContiguousPReq*, *SingleMCPReq*

Uncontrolled variables: *Topology*

Test campaigns: These tests are to demonstrate that MESCAL HSO can calculate an end-to-end path for establishing end-to-end LSPs.

12.2.3.2.2 TB_P3_SYS/BC/2

Objectives: This test assesses whether the combination of q-BGP and PCS operations can meet the target performance figures for the LSP.

Performance metrics: *Delay*, *QoSSatisfactionReq*

Uncontrolled variables: *Topology*

Test campaigns: These tests are to demonstrate that MESCAL HSO can calculate hard guaranteed QoS for end-to-end LSPs.

12.2.3.2.3 TB_P3_SYS/BC/3

Objectives: This test assesses the performance of the Path Computation Server.

Performance metrics: *ClcDelay*, *Overhead*, others [for further study]

Controlled variables: *EndpointsP*, *QoSReqsP*

Uncontrolled variables: *topology*

Test campaigns: The test campaign will measure a number of metrics related to the performance of the PCS in calculating end-to-end LSPs.

References

[BRITE] http://www.cs.bu.edu/brite/

[D1.1] P. Flegas et al., D1.1*: Specification of Business Models and a Functional Architecture for Inter-domain QoS Delivery*, http://www.mescal.org/deliverables/MESCAL-D11-final.pdf, June 2003.

[D1.2] M. Howarth et al., D1.2*: Initial specification of protocols and algorithms for inter-domain SLS management and traffic engineering for QoS-based IP service delivery and their test requirements*, http://www.mescal.org/deliverables/MESCAL-D12-public-final.pdf, January 2004.

[D2.1] D. Griffin et al., *D2.1: Implementation plan and high level engineering design of simulation models and testbed prototypes for inter-domain QoS delivery*, http://www.mescal.org/deliverables/MESCAL-D21-final.pdf, January 2004.

[Falout99] M. Faloutsos, P. Faloutsos and C. Faloutsos, *On Power-Law Relationships of the Internet Topology*, Proc. ACM SIGCOMM 1999, 30 August - 3 September 1999, Cambridge, MA, USA.

[Feamster03] N. Feamster, J. Borkenhagen and J. Rexford, *Guidelines for Interdomain Traffic Engineering*, ACM Computer Communication Review, October 2003.

[Govin02] R. Govindan and P. Radoslavov, *An Analysis of The Internal Structure of Large Autonomous Systems*, Technical Report 02-777, CS Dept, University of Southern California, 2002.

[Heckm03] O. Heckmann et al., *On realistic network topologies for simulation*, Proc. ACM SIGCOMM 2003 Workshops, 25-27 August 2003, Karlsruhe, Germany.

[Hipol03] J.I. Nieto-Hipolito and J.M. Barcelo, *Analysis on Autonomous Systems*, Technical Report, Polytechnic University of Catalonia, Spain.

[Mag01] D. Magoni and J.J. Pansiot, *Analysis of the Autonomous System Network Topology*, ACM Computer Communication Review, July 2001.

[Quoit03] B. Quoitin, C. Pelsser and L. Swinnen, *Interdomain traffic engineering with BGP*, IEEE Communications Magazine, Vol. 41, No.5, pp.122-128, May 2003.

[Rexford01] J. Rexford, J. Wang, Z. Xiao and Y.Zhang, *BGP Routing Stability of Popular Destinations*, Proc. ACM Internet Measurement Workshop, 2002.

[RIP] http://www.ripe.net/ripencc/about/regional/rir-system.html

[Subra02] L. Subramanian, S. Agarwal, J. Rexford and R.H. Katz, *Characterizing the Internet Hierarchy from Multiple Vantage Points*, Proc. IEEE INFOCOM 2002, pp.618-627.

[Tang01] H. Tangmunarunkit et al, *Network topology generators: degree-based vs structural*, SIGCOMM, IEEE Conference on Communication 2001.

[Uhlig02] S. Uhlig and O. Bonaventure, *Implications of Interdomain Traffic Characteristics on Traffic Engineering*, European Transactions on Telecommunications Journal, Special Issue on Traffic Engineering, January 2002.

Appendix A

13 TESTBED CONFIGURATION

13.1 Introduction

This document describes the MESCAL testbed that will be used during the development and validation phases 1, 2 and 3 defined in [D2.1]. This testbed is located in FTR&D premises, in Caen, France.

The same platform will be used for carrying out these 3 phases. From phase 1 to phase 3 the core of the platform (AS topology, links, addressing, core technologies) will remain the same, but will progressively evolve, mainly in terms of configuration and software deployed in order to support the specific constraints and requirements of each of these phases.

Phase 1 aims at validating the overall testing environment including: hardware, software, traffic generators, BGP and QoS configuration. This phase needs a large number of Linux and non-Linux features to be activated but does not include any of the new feature MESCAL has to develop. This phase will allow verifying that the selected environment is suitable for supporting phases 2 and 3 and will definitively confirm the pertinence of MESCAL technical choices for this testbed. Additionally, the experience gained during this phase will help to identify and/or improve all appropriate tools for maintaining the testbed in an efficient way.

In particular, this phase will allow:

- To validate the inter-domain routing with ZebOS running on Linux PCs by setting-up several ASes and by configuring BGP between those domains.
- To validate, at the data plane level, the DSCP swapping (marking/remarking) between ASes in order to signal a given meta-QoS-class.
- To validate the implementation of l-QCs in each domain using Linux traffic control features.

Phase 2 aims at validating the loose service option. The same testbed will be used for validating q-BGP implementation and the enhanced Linux IP forwarding. Depending on the tests that will be performed the number and the definition of local-QoS-classes in each AS will be adapted together with the pSLS definitions. This will be on per test basis.

Phase 3 aims at validating the Path Computation Server (PCS) function. For this purpose, the PCS implementation will be uploaded in an appropriate set of Linux routers but the overall infrastructure will remain the same as for the phase 1. No MPLS features will be deployed since the project will focus only on the computation of inter-AS paths for establishing inter-domain LSPs.

More precisely, this document provides information on:

- The AS topology
- The deployed hardware,
- The links setup between routers,
- The IP addressing scheme,
- The BGP and q-BGP configurations,
- The QoS policies defined for each domain, including a description of the local-QoS-classes deployed in each AS,
- The tools developed for maintaining, configuring and monitoring the platform,
- The tools used for generating and analysing IP traffic.

This document will be regularly updated to reflect any changes made on the testbed during the lifetime of the project.

13.2 Autonomous System Topology

The testbed supports 8 autonomous systems. A single router implements each AS except AS4, which is composed of 3 routers. AS4 and AS6 represent Tier-one providers and have more network resources allocated. AS1 represents a Tier-3 provider. The remaining ASes are Tier-two providers.

In phase 1, BGP is activated at the boundaries of each domain and iBGP is activated within AS4 domain. In phase 2 q-BGP and q-iBGP will be activated instead.

These ASes are interconnected as roughly shown in Figure 9 and depicted in more details in Figure 10.

Some inter-AS interconnections have been doubled in order to allow more sophisticated inter-domain routing tests and to evaluate eventual load balancing features especially between:

- AS4 and AS6
- AS3 and AS4

Even if this topology is far from the real Internet it will nevertheless allow learning different QoS routes thanks to ad-hoc activation of pSLS. If necessary, a maximum of 6 crossed AS can be reached provided the appropriate configuration.

Figure 11 gives a more accurate view of the testbed. It groups on the same picture different level of information:

- AS numbers
- Links between ASes
- IP address of the interfaces, with their interface number on each machine
- DS code point values used to signal local-QoS-classes
- Agreed DS code point values used to signal meta-QoS-classes between domains together with the capacity provisioned for each class.
- Role of the remaining interface: management interface or customer interface

Smartbit interfaces are connected to each router via a dedicated interface. These interfaces are mainly used for injecting load traffic in the testbed. Some of the routers (AS1 and AS8) have additional Smartbit interfaces connected in order to inject customer traffic used for measurement purposes.

Finally, Figure 12 provides a synthetic view of the backside of each router and allows locating quickly the interfaces using their allocated Linux interface number.

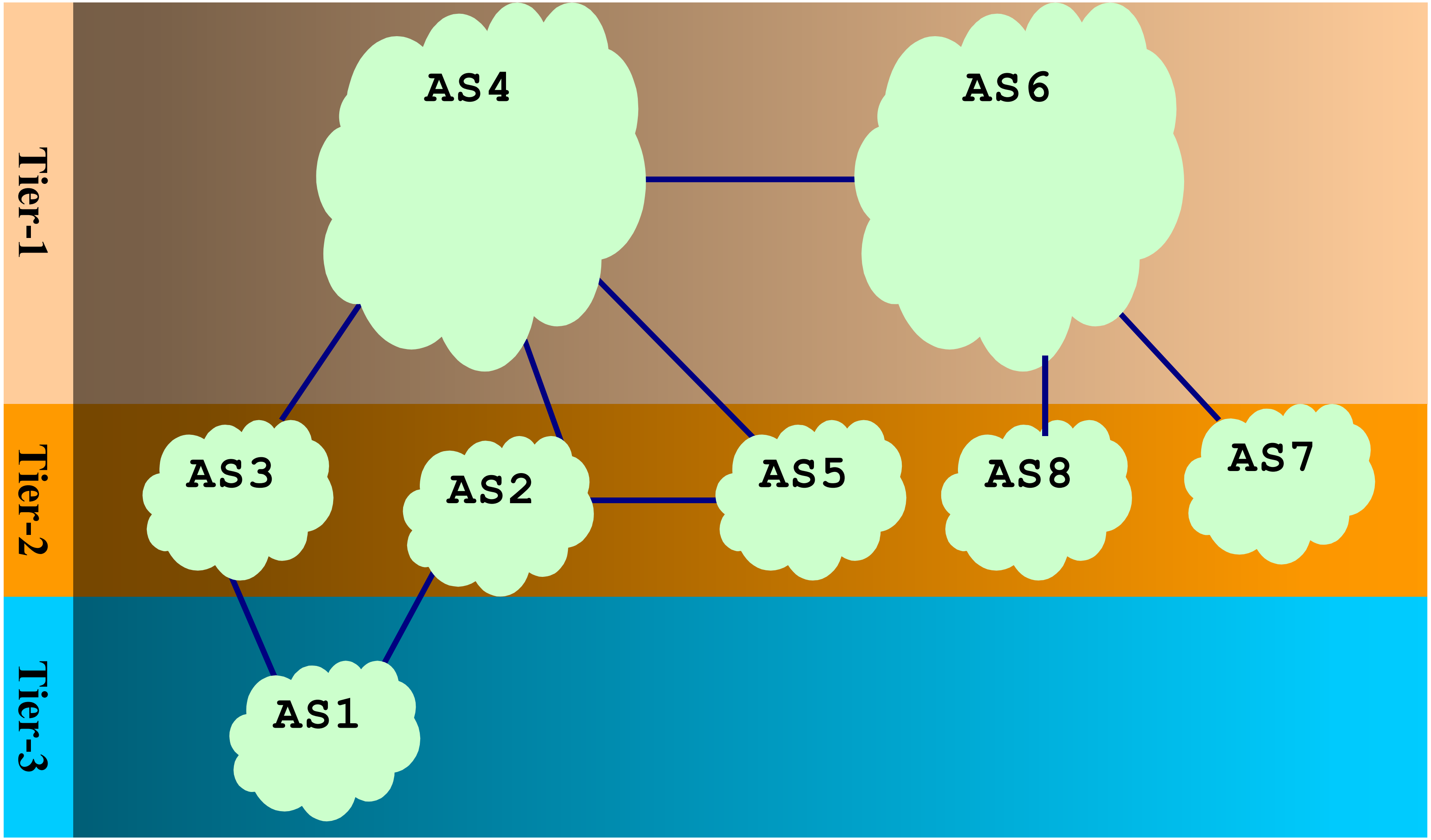


Figure 9: FTR&D MESCAL Testbed Hierarchical View

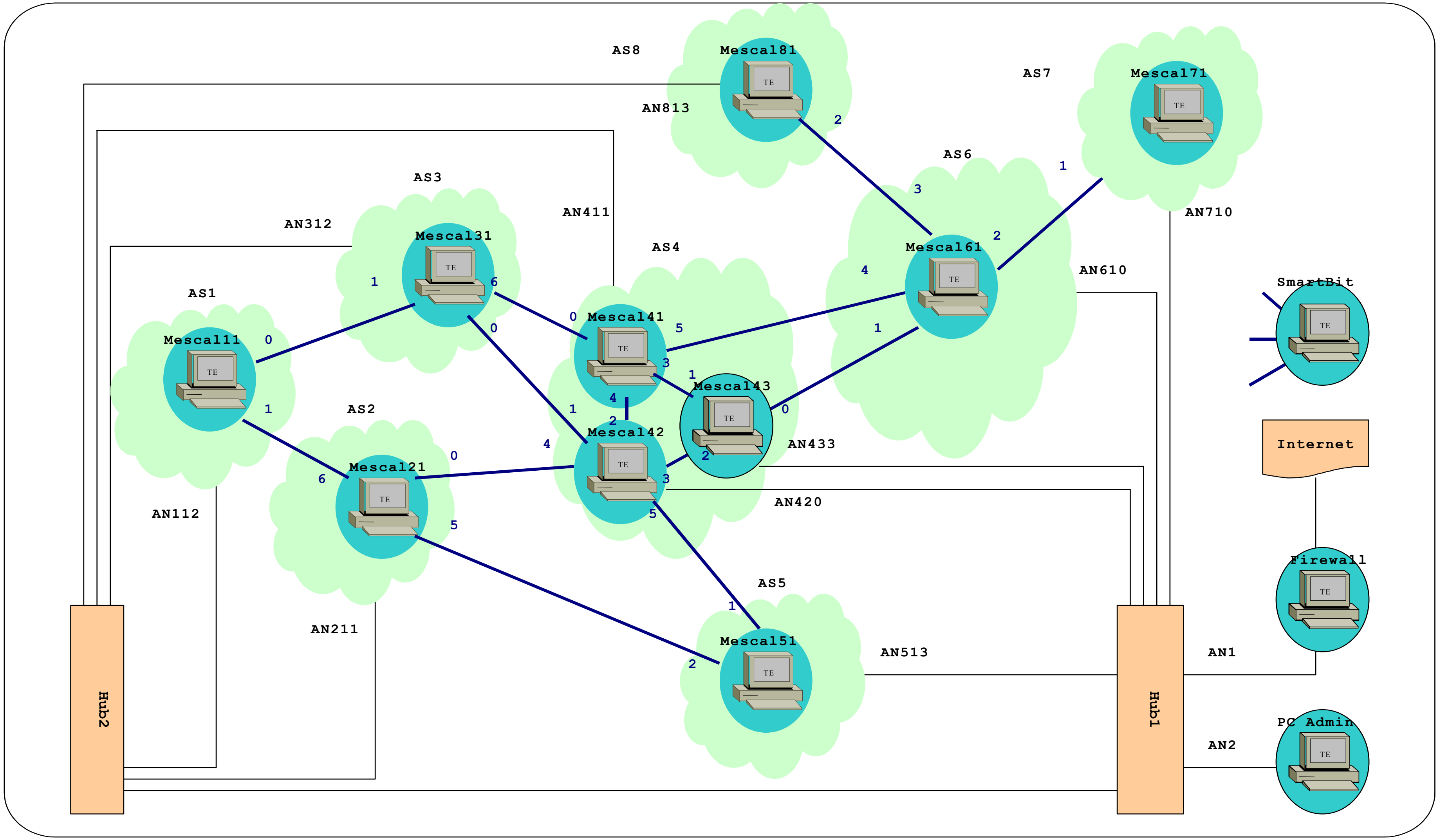


Figure 10: FTR&D MESCAL Testbed Detailed Architecture -1

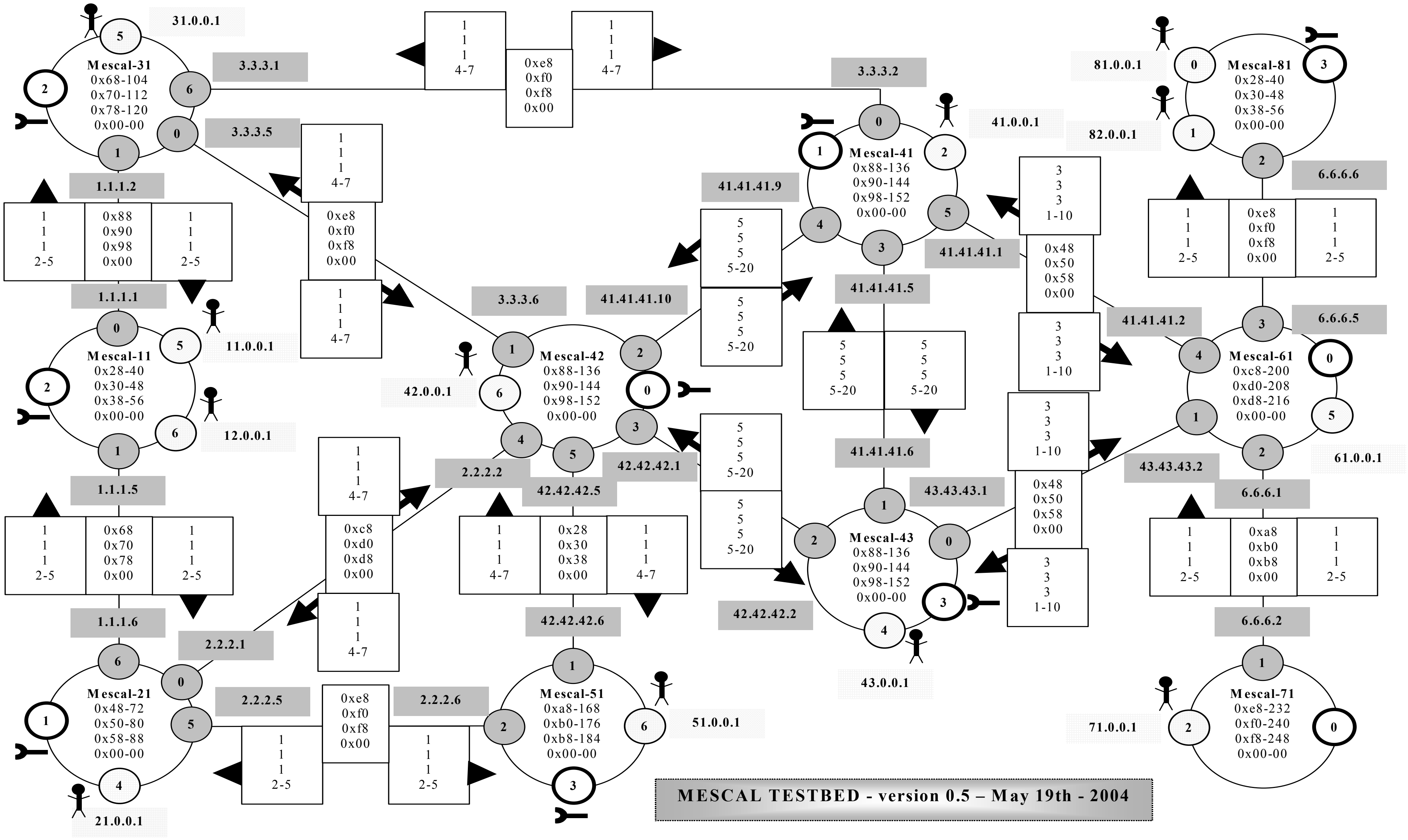


Figure 11: FTR&D MESCAL Testbed Detailed Architecture -2

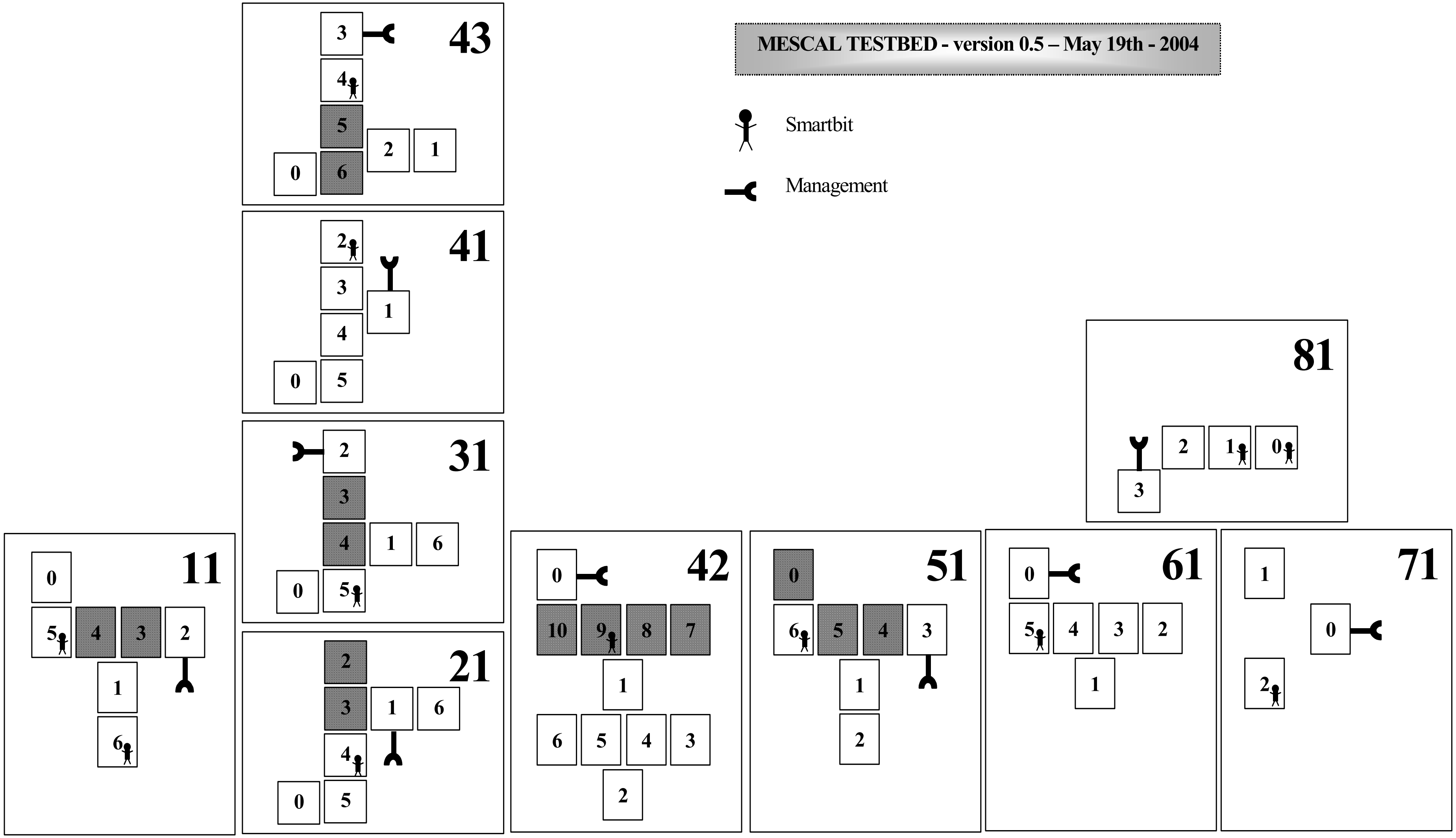


Figure 12: FTR&D MESCAL Testbed Network Interfaces Schema

13.3 Testbed Components

The testbed deployed by France Telecom R&D premises contains many devices including network elements (routers, hubs…), workstations and test equipments. In this current section, we list and describe both the hardware and software components used for building the FTR&D MESCAL testbed.

13.3.1 Hardware Components

13.3.1.1 PCs

PCs used in the testbed are of four types:

- Type 1: Intel® Xeo, CPU 1.7 GHz, 17 GB hard disk drive, 512 MB RAM
- Type 2: Intel® PIII, CPU 1.0 GHz, 20 GB hard disk drive, 512 MB RAM
- Type 3: Intel® P4, CPU 1.7 GHz, 20 GB hard disk drive, 512 MB RAM
- Type 4: Intel® PIII, CPU 500 MHz, 8 GB hard disk drive, 256 MB RAM

Name	Type	Available PCI Slots
Mescal11	1	4
Mescal21	3	3
Mescal31	3	3
Mescal41	2	3
Mescal42	1	4
Mescal43	2	3
Mescal51	1	4
Mescal61	1	4
Mescal71	1	4
Mescal81	3	3
PC-admin	4	4

Table 68: PC Characteristics

Except PC-admin, all these PCs are used as PC-based routers.

Additional PCs could be added to the testbed and would be used to emulate customer premises.

13.3.1.2 Traffic Generators

13.3.1.2.1 Smartbits

Two *SmartBits* chassis (SMB 2000 and SMB600) are available in the lab. These equipments are used to inject traffic in the network and to carry out measurements.

The SMB 2000 is equipped with 20 Ethernet cards:

- 10 of them are 10/100 Mbit/s full duplex cards,
- The others are 10 Mbit/s half duplex cards.

The SMB 6000 is equipped with 2 cards of 2 x 10/100 Mbit/s Ethernet cards. Those cards support additional features the SMB 2000 cards do not support and should be preferred when QoS measurements need to be carried out.

13.3.1.2.2 QARobots

In addition, *QARobot* will be used to generate BGP messages in order to test routing features. This tool will be used to validate the behaviour of q-BGP and more especially the conformity of BGP messages.

13.3.2 Software Components

13.3.2.1 Operating System

Linux Red Hat version 9 (kernel version of 2.4.20-8) is installed on all PC-based routers and PC-admin.

13.3.2.2 Software Information

Software	Version	Company/Package	Description
GCC	3.2.2	GNU	C++ Compiler
ZebOS	5.3.1	IPInfusion	Routing stack including BGP
Ethereal	0.9.8		Traffic analyzer
MGEN	4.2	Naval Research Laboratory	Traffic generator
Jnettop	0.9	GNU	Real-time interface bandwidth measurement
NTP	4.2		Time server

Table 69: Software Information

During phase 1, the ZebOS routing stack is used without any modification. During phase 2, it is enhanced to support QoS related messages and information. Phase 3 relies on the q-BGP implementation realised during phase 2.

MGEN and *TG* are traffic generators that could be used as a complement to the *Smartbits*. These tools can generate UDP and TCP traffic (especially *TG*). They can set the DS code point on a per flow basis.

13.4 Configuration for Phase 1

13.4.1.1 Folders Structure

13.4.1.1.1 Routers

The same following folder structure has been setup in each PC-based router.

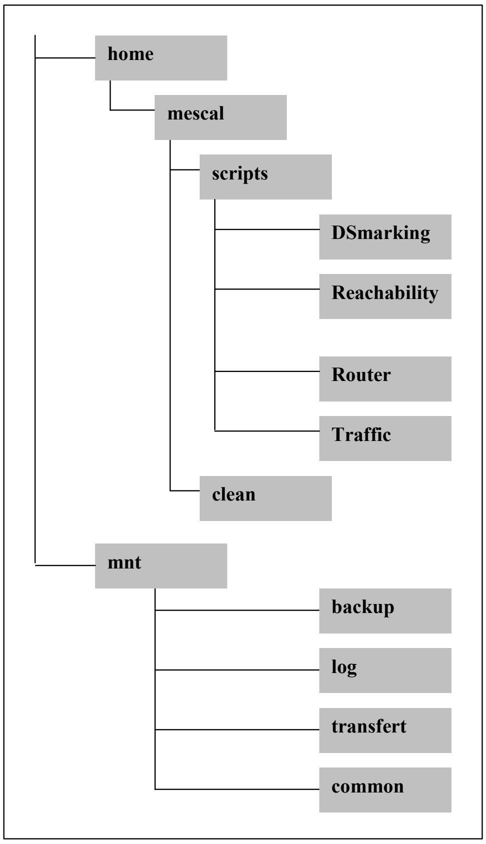


Figure 13: Router Folder Structure

A brief description of each folder and the files it contains is hereafter provided:

- **`DSmarking`**: groups all Traffic Control (TC) related configuration files and scripts for a given router.
  - **`qsi-X`**: This file contains ingress related DiffServ policy instructions for configuring the interface X of the router.
  - **`qsHTB-X`**: This file contains egress related DiffServ policy instructions for configuring interface X of the router.

  In addition, the following files are used to configure in once all the interfaces of the router:

  - **`qsi`**: This script configures the ingress DiffServ policy for all the interfaces of the router.
  - **`qse`**: This script configures the egress DiffServ policy for all the interfaces of the router.
  - **`qsa`**: This script configures both the ingress and the egress DiffServ policy for all the interfaces of the router.
  - **`qsdel`**: This script deletes all ingress and egress policies from all interfaces of the router.
  - **`qsstat`**: This script displays real-time egress TC related statistics.
  - **`qsstati`**: This script displays real-time ingress TC related statistics.
  - **`ifstat`**: This script displays the interface's statistics as reported by the kernel.
  - **`ifstatistics`**: This file stores a snapshot of the interfaces statistics. It is generated and then used by **`ifstat`** in order to compute delta of traffic drops, errors, etc.

The **DSmarking** folder also contains a **stat** folder where temporary data related to interfaces statistics are stored. **qsstat** and **qsstati** generate and use these data in order to compute deltas of traffic between updates.

- **Reachability**: This folder groups a set of scripts used to verify the reachability of a set of destinations.
  - **cgping**: This script displays the reachability status of customer interfaces for a list of meta-QoS-class planes which is provided in the **metaqosclass** file. The list of destinations if stored in the **customers** file.
  - **pfgping**: This script displays the reachability status of router interconnection interfaces for a list of meta-QoS-class planes which is provided in the **metaqosclass** file. The list of destinations if stored in the **interfaces** file.
  - **metaqosclass**: This file contains the list of DSCP values the **cgping/pfgping** command must consider. Each DSCP entry takes the value of a local-QoS-class that implements a given meta-QoS-classe.
  - **customers**: This file contains the list of customer' addresses the **cgping** command has to monitor.
  - **interfaces**: This file contains the list of testbed interconnection interface addresses the **pfgping** command has to monitor.
- **Router**: This folder groups the scripts used for routing purposes.
  - **bgpstart**: This script starts the **bgpd** and **nsm** processes.
  - **bgpstop**: This script stops the **bgpd** and **nsm** processes.
  - **isalive**: This script reports the status of **bgpd** and **nsm** processes.
  - **svty**: This script launches the configuration interface of the router.
  - **pcm**: This script mounts the **common, transfert, log** and **backup** folders.
  - **pcu**: This script un-mounts the **common, transfert, log** and **backup** folders
- **Traffic**: This folder contains scripts used to generate UDP and TCP traffic.
- **backup**: This folder stores backups from all routers. A dedicated sub-folder bas been created for each router.
- **log**: This folder contains log files storing configuration operations traces. Each router has its own dedicated log file.
- **transfer** : This folder is used to transfer files between the machines.
- **common:** This folder contains scripts used by all router.

The **backup**, **log**, **transfer** and **common** folders of each machine are symbolic links towards PC-admin.

13.4.1.1.2 PC-admin

PC-admin is used for managing and maintaining the routers involved in the testbed. Common scripts used by the routers are stored in this machine. The folder structure is shown by the figure below:

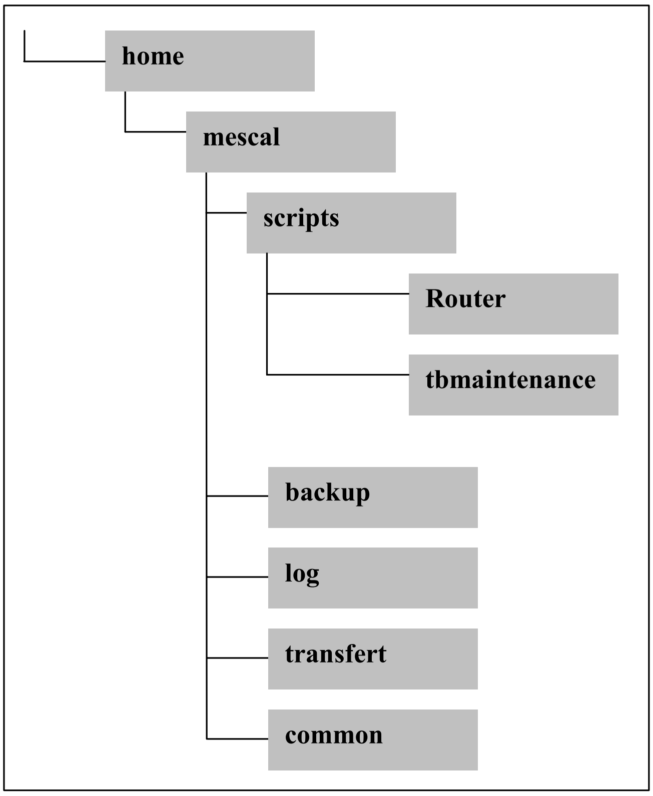


Figure 14: PC-admin Folder Structure

13.4.1.2 User Accounts

Only two accounts have been created in all PCs.

Passwords will be provided by FTR&D on a per request basis.

13.4.1.3 Remote Connection

Remote connections to the routers are achieved via SSH. There are no restrictions between testbed components. From outside the testbed, connections are filtered by an external firewall. Only **PC-admin** has been made accessible. The firewall ensures a network address translation for this latter machine. The corresponding public IP address will be provided by FTR&D on request. Firewall rules will be updated to allow external connexions from MESCAL partners.

In addition, the **`ftpd`** service has been enabled in **PC-admin** and in the routers.

13.4.1.4 Internet Access

An Internet access is configured in all machines involved in the testbed. All Internet connexions go through **PC-admin**, which acts as a HTTP/FTP proxy.

13.4.1.5 Firewall Rules

Firewall rules have been added in order to control communications from/to testbed machines.

13.4.1.6 Time Synchronisation

PC-admin is configured as a NTP server. It synchronizes its clock with a public NTP server. All routers in the testbed synchronise their clock with PC-admin.

13.4.1.7 *Printer*

A postscript printer, called **Gutenberg**, is available for all the machines of the testbed.

13.4.1.8 *AS Identifiers*

Hereafter is listed the AS number affected to each AS. This AS number will be used when configuring BGP instances.

AS	AS number
AS1	1
AS2	2
AS3	3
AS4	4
AS5	5
AS6	6
AS7	7
AS8	8

Table 70: AS Numbers

13.4.1.9 *Addressing*

The table below provides addressing information related to the routers used in the testbed. For details about the architecture of the testbed refer to Figure 9.

The Following abbreviations are used in the table below:

- ANijk: is used to refer to the Administrative Network used for maintaining and configuring the tested. The 3-uple identifies a unique interface in the testbed: "ij" identifies a router whereas "k" identifies a unique interface of the router "ij".
- PINijk-rst: (Provider Interconnection network) is used to identify in a unique way an inter-domain link between two border routers. "ijk" and "rst" are two distinct interfaces.

Router	Type	Alias/Interface	Address	Description
Mescal11	Linux-based router			
		Mescal110 /Eth0	1.1.1.1	PIN110-311
		Mescal111 /Eth1	1.1.1.5	PIN112-216
		Mescal112 /Eth2	192.168.66.65	AN112
		Mescal113 /Eth3		
		Mescal114 /Eth4		
		Mescal115 /Eth5	11.0.0.1	Smart bit
		Mescal116 /Eth6	12.0.0.1	Smart bit
Mescal21	Linux-based router			
		Mescal210 /Eth0	2.2.2.1	PIN210-424
		Mescal211 /Eth1	192.168.66.66	AN211
		Mescal212 /Eth2		
		Mescal213 /Eth3		
		Mescal214 /Eth4	21.0.0.1	Smart bit
		Mescal215 /Eth5	2.2.2.5	PIN215-512
		Mescal216 /Eth6	1.1.1.6	PIN111-216
Mescal31	Linux-based router			
		Mescal310 /Eth0	3.3.3.5	PIN310-421
		Mescal311 /Eth1	1.1.1.2	PIN110-311
		Mescal312 /Eth2	192.168.66.67	AN312
		Mescal313 /Eth3		
		Mescal314 /Eth4		
		Mescal315 /Eth5	31.0.0.1	Smart bit

		Mescal316 /Eth6	3.3.3.1	PIN316-410
Mescal41	Linux-based router			
		Mescal410 /Eth0	3.3.3.2	PIN316-410
		Mescal411 /Eth1	192.168.66.68	AN411
		Mescal412 /Eth2	41.0.0.1	Smart bit
		Mescal413 /Eth3	41.41.41.5	PIN413-431
		Mescal414 /Eth4	41.41.41.9	PIN414-422
		Mescal415 /Eth5	41.41.41.1	PIN415-614
Mescal42	Linux-based router			
		Mescal420 /Eth0	192.168.66.69	AN420
		Mescal421 /Eth1	3.3.3.6	PIN310-411
		Mescal422 /Eth2	41.41.41.10	PIN414-422
		Mescal423 /Eth3	42.42.42.1	PIN423-432
		Mescal424 /Eth4	2.2.2.2	PIN210-424
		Mescal425 /Eth5	42.42.42.5	PIN425-511
		Mescal426 /Eth6	42.0.0.1	Smart bit
		Mescal427 /Eth7		
		Mescal428 /Eth8		
		Mescal429 /Eth9		
		Mescal4210 /Eth10		
Mescal43	Linux-based router			
		Mescal430 /Eth0	43.43.43.1	PIN430-611
		Mescal431 /Eth1	41.41.41.6	PIN413-431
		Mescal432 /Eth2	42.42.42.2	PIN423-432
		Mescal433 /Eth3	192.168.66.63	AN433
		Mescal434 /Eth4	43.0.0.1	Smart bit
		Mescal435 /Eth5		
		Mescal436 /Eth6		
Mescal51	Linux-based router			
		Mescal510 /Eth0		
		Mescal511 /Eth1	42.42.42.6	PIN425-511
		Mescal512 /Eth2	2.2.2.6	PIN212-512
		Mescal513 /Eth3	192.168.66.70	AN513
		Mescal514 /Eth4		
		Mescal515 /Eth5		
		Mescal516 /Eth6	51.0.0.1	Smart bit
Mescal61	Linux-based router			
		Mescal610 /Eth0	192.168.66.71	AN610
		Mescal611 /Eth1	43.43.43.2	PIN430-611
		Mescal612 /Eth2	6.6.6.1	PIN612-711
		Mescal613 /Eth3	6.6.6.5	PIN613-812
		Mescal614 /Eth4	41.41.41.2	PIN415-614
		Mescal615 /Eth5	61.0.0.1	Smart bit
Mescal71	Linux-based router			
		Mescal710 /Eth0	192.168.66.72	AN710
		Mescal711 /Eth1	6.6.6.2	PIN612-711
		Mescal712 /Eth2	71.0.0.1	Smart bit
Mescal81	Linux-based router			
		Mescal810 /Eth0	81.0.0.0	Smart bit
		Mescal811 /Eth1	82.0.0.0	Smart bit
		Mescal812 /Eth2	6.6.6.6	PIN613-812
		Mescal813 /Eth3	192.168.66.64	AN813

Table 71: Routers Addressing

13.4.1.10 LANs

This table summarizes the addressing plan for each local area network.

Subnet	Address/Mask	Device/Interface	Address	Description
Administrative Network	192.168.66.0/24			
		PC Admin /Eth0	.73	AN2
		Firewall	.1	AN1

Table 72: Administrative Network Addressing

13.4.1.11 Customer Addresses

The following table lists the IP network addresses used by customers connected to each AS (via a direct physical connection to each AS).

AS	Customers	IP addresses realm
AS1	Customer11	11.0.0.0/10
	Customer12	12.0.0.0/10
	Customer13	13.0.0.0/10
	Customer14	14.0.0.0/10
AS2	Customer21	21.0.0.0/10
	Customer22	22.0.0.0/10
	Customer23	23.0.0.0/10
	Customer24	24.0.0.0/10
AS3	Customer31	31.0.0.0/10
	Customer32	32.0.0.0/10
	Customer33	33.0.0.0/10
	Customer34	34.0.0.0/10
AS4	Customer41	41.0.0.0/10
	Customer42	42.0.0.0/10
	Customer43	43.0.0.0/10
	Customer44	44.0.0.0/10
AS5	Customer51	51.0.0.0/10
	Customer52	52.0.0.0/10
	Customer53	53.0.0.0/10
	Customer54	54.0.0.0/10
AS6	Customer61	61.0.0.0/10
	Customer62	62.0.0.0/10
	Customer63	63.0.0.0/10
	Customer64	64.0.0.0/10
AS7	Customer71	71.0.0.0/10
	Customer72	72.0.0.0/10
	Customer73	73.0.0.0/10
	Customer74	74.0.0.0/10
AS8	Customer81	81.0.0.0/10
	Customer82	82.0.0.0/10
	Customer83	83.0.0.0/10
	Customer84	84.0.0.0/10

Table 73: Customers IP Address Realms

13.4.1.12 Network Addresses Announced by each AS

This table lists the IP network addresses announced by each AS.

AS	Customers	IP addresses realm
AS1	Network11	193.251.128.0/19

	Network12	212.167.0.0/21
	Network13	62.42.0.0/16
	Network14	193.251.240.0/20
AS2	Network21	194.52.168.0/21
	Network22	202.167.22.0/24
	Network23	194.199.98.0/24
	Network24	128.223.60.102/20
	Network25	193.41.227.0/24
	Network26	192.176.123.0/24
	Network27	192.94.149.0/24
AS3	Network31	216.191.64.0/20
	Network32	154.101.11.10/24
	Network33	132.150.224.0/24
	Network34	155.234.165.12/19
	Network35	199.79.131.0/24
	Network36	198.26.215.0/24
	Network37	196.11.196.0/24
AS4	Network41	62.177.143.254/20
	Network42	62.216.31.254/19
	Network43	146.188.61.109/22
	Network44	195.69.144.12/18
	Network45	192.70.132.0/24
	Network46	216.169.114.0/24
	Network47	216.116.175.0/24
	Network48	214.3.214.0/24
	Network49	205.237.35.0/24
	Network410	204.222.17.0/24
	Network411	204.116.187.0/24
	Network412	203.116.188.0/24
	Network413	216.103.190.0/24
	Network414	216.84.141.0/24
	Network415	213.239.59.0/24
	Network416	213.205.25.0/30
	Network417	198.205.10.0/24
AS5	Network51	198.32.247.85/20
	Network52	12.129.192.134/20
	Network53	170.170.0.0/16
	Network54	203.34.233.0/24
AS6	Network61	205.204.1.0/24
	Network62	216.255.39.0/24
	Network63	193.100.167.0/24
	Network64	193.19.208.0/24
	Network65	198.129.78.23/20
	Network66	193.111.167.0/24
	Network67	192.197.12.0/24
	Network68	192.58.243.0/24
	Network69	192.43.226.0/24
	Network610	200.108.174.0/24
	Network611	200.10.200.0/24
	Network612	194.215.31.0/24
	Network613	195.177.120.0/24
	Network614	193.108.167.0/24
	Network615	170.163.0.0/16
	Network616	217.108.191.0/24
	Network617	217.10.217.0/24
	Network618	216.95.141.0/24
	Network619	212.78.144.0/24
AS7	Network71	209.215.12.0/24
	Network72	208.147.73.0/24

	Network73	207.245.48.0/27
	Network74	192.228.46.0/24
AS8	Network81	198.69.130.0/24
	Network82	193.16.208.0/24
	Network83	192.35.226.0/24
	Network84	192.236.46.0/24

Table 74: Customers IP Address Realms

13.4.1.13 Routing Configuration

This section provides routing information, as currently configured in the testbed. The routing protocol used in the network is BGP (Border Gateway Protocol). Each AS has at least one e-BGP session with its adjacent ASes. i-BGP sessions have been established between all AS4 border routers.

Hereafter is an example of a BGP configuration (example of **`MESCAL420`**)

```
!
!
! Config for ZebOS version 5.3.1:03312003-Main (i686-pc-linux-gnu)
! Written 2004/05/19 17:59:55
!

!
banner motd Welcome to MESCAL42 Router :))
if-arbiter
!
interface lo
 ip address 42.1.1.1/32
!
interface eth0
 ip address 192.168.66.69/24
!
interface eth1
 ip address 3.3.3.6/30
!
interface eth2
 ip address 41.41.41.10/30
!
interface eth3
 ip address 42.42.42.1/30
!
interface eth4
 ip address 2.2.2.2/30
!
interface eth5
 ip address 42.42.42.5/30
!
interface eth6
!
router bgp 4
 bgp router-id 42.42.42.1
 bgp log-neighbor-changes
 bgp scan-time 10
 network 41.0.0.0/10
 network 42.0.0.0/10
 network 43.0.0.0/10
 network 44.0.0.0/10
 network 62.177.128.0/20
 network 62.216.0.0/19
 network 146.188.60.0/22
 network 192.70.132.0/24
 network 195.69.128.0/18
 network 198.205.10.0/24
 network 203.116.188.0/24
 network 204.116.187.0/24
 network 204.222.17.0/24
 network 205.237.35.0/24
 network 213.205.25.0/30
 network 213.239.59.0/24
 network 214.3.214.0/24
 network 216.84.141.0/24
 network 216.103.190.0/24
 network 216.116.175.0/24
 network 216.169.114.0/24
 redistribute connected
 redistribute static
 timers bgp 10 15
 neighbor 2.2.2.1 remote-as 2
 neighbor 2.2.2.1 advertisement-interval 5
 neighbor 2.2.2.1 prefix-list FLAN out
 neighbor 3.3.3.5 remote-as 3
 neighbor 3.3.3.5 advertisement-interval 5
 neighbor 3.3.3.5 prefix-list FLAN out
 neighbor 41.41.41.9 remote-as 4
 neighbor 41.41.41.9 advertisement-interval 5
 neighbor 41.41.41.9 prefix-list FLAN1 out
 neighbor 42.42.42.2 remote-as 4
 neighbor 42.42.42.2 advertisement-interval 5
 neighbor 42.42.42.2 prefix-list FLAN2 out
 neighbor 42.42.42.6 remote-as 5
 neighbor 42.42.42.6 advertisement-interval 5
 neighbor 42.42.42.6 prefix-list FLAN out
!
ip route 3.3.3.0/30 41.41.41.9 2
ip route 3.3.3.0/30 42.42.42.2 3
ip route 41.0.0.0/30 41.41.41.9 2
ip route 41.0.0.0/30 42.42.42.2 3
ip route 41.41.41.0/30 41.41.41.9 2
ip route 41.41.41.0/30 42.42.42.2 3
ip route 41.41.41.4/30 41.41.41.9 2
ip route 41.41.41.4/30 42.42.42.2 2
ip route 43.0.0.0/30 42.42.42.2 2
ip route 43.0.0.0/30 41.41.41.9 3
ip route 43.43.43.0/30 42.42.42.2 2
ip route 43.43.43.0/30 41.41.41.9 3
ip route 192.168.1.0/24 192.168.66.1
!
ip prefix-list FLAN seq 5 deny 192.168.66.0/24
ip prefix-list FLAN seq 10 deny 192.168.1.0/24
ip prefix-list FLAN seq 15 permit any
ip prefix-list FLAN1 seq 5 deny 3.3.3.0/30
ip prefix-list FLAN1 seq 10 deny 41.41.41.0/30
ip prefix-list FLAN1 seq 15 deny 192.168.66.0/24
ip prefix-list FLAN1 seq 20 deny 192.168.1.0/24
ip prefix-list FLAN1 seq 25 permit any
ip prefix-list FLAN2 seq 5 deny 43.43.43.0/30
ip prefix-list FLAN2 seq 10 deny 192.168.66.0/24
ip prefix-list FLAN2 seq 15 deny 192.168.1.0/24
ip prefix-list FLAN2 seq 20 permit any
```

13.4.1.13.1 e-bgp

An e-BGP session is configured in ZebOS routers as follows (example of AS4 declaring AS2 as neighbour):

```
router bgp 4
 bgp router-id 42.42.42.1
 neighbor 2.2.2.1 remote-as 2
```

13.4.1.13.2 i-bgp

In the testbed, AS4 is made of three routers. Each router has to declare the other two routers as i-BGP neighbours, to do so the following configuration has to be added (example of **MESCAL420**):

```
router bgp 4
 bgp router-id 42.42.42.1
 neighbor 41.41.41.9 remote-as 4
 neighbor 42.42.42.2 remote-as 4
```

13.4.1.13.3 Networks

In order to advertise networks prefixes, the command "**network**" is used as listed below:

```
router bgp 4
 network 214.3.214.0/24
 network 216.84.141.0/24
 network 216.103.190.0/24
 network 216.116.175.0/24
 network 216.169.114.0/24
```

13.4.1.13.4 Static Routes

We made the decision not to activate an IGP protocol within AS4 domain. This is motivated by the fact that we prefer not having to modify IGP related processes feeding the FIBs when q-BGP is deployed and enabled. Thus, we will make use of static routes in order to simplify the development phase.

Static routes can be configured with the command "**ip route**". The example below illustrates the static routes that have been configured in **MESCAL420** for joining other AS4 intra-domain destination.

```
ip route 41.41.41.0/30 42.42.42.2 3
ip route 41.41.41.4/30 41.41.41.9 2
ip route 41.41.41.4/30 42.42.42.2 2
ip route 43.0.0.0/30 42.42.42.2 2
ip route 43.0.0.0/30 41.41.41.9 3
ip route 43.43.43.0/30 42.42.42.2 2
ip route 43.43.43.0/30 41.41.41.9 3
```

13.4.1.13.5 Prefix List

The ZebOS implementation can filter network prefixes announcements on a peer-by-peer basis thanks to the use of the "**prefix-list**" command. The configuration bellow allows to send all configured network prefixes except the 43.43.43.0/30, 192.168.66.0/24 and 192.168.1.0/24 Addresses.

```
ip prefix-list FLAN2 seq 5 deny 43.43.43.0/30
ip prefix-list FLAN2 seq 10 deny 192.168.66.0/24
ip prefix-list FLAN2 seq 15 deny 192.168.1.0/24
ip prefix-list FLAN2 seq 20 permit any
```

13.4.1.13.6 Fast Link Failover Detection

ZebOS has been configured to support fast link failover detection.

13.4.1.13.7 BGP Timers

The BGP timers that can be configured are: "**keepalive**", "**holdtime**" and "**connect**". We configured the two first timers to 10s and 15s respectively. This is achieved with the following command:

```
Bgp router 4
 timers bgp 10 15
```

The motivation behind these values is to decrease the time needed for the detection of invalid routes.

13.4.1.13.8 Route Selection Process

ZebOS allows selecting a type of route selection process. We selected the "**rfc-1771-path-selection**" for all routers.

13.4.1.14 Local QoS class DSCP values

This table summarizes the l-QC DSCP values that are used in each AS.

AS	Local QC	DSCP	b-DSCP	x-DSCP	d-DSCP	b-DS	x-DS	d-DS
AS1	l-QC11	001010	0000-1010	0xa	10	0010-1000	0x28	40
	l-QC12	001100	0000-1100	0xc	12	0011-0000	0x30	48
	l-QC13	001110	0000-1110	0xe	14	0011-1000	0x38	56
	l-QC14	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS2	l-QC21	010010	0001-0010	0x12	18	0100-0100	0x48	72
	l-QC22	010100	0001-0100	0x14	20	0101-0000	0x50	80
	l-QC23	010110	0001-0110	0x16	22	0101-1000	0x58	88
	l-QC24	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS3	l-QC31	011010	0001-1010	0x1a	26	0110-1000	0x68	104
	l-QC32	011100	0001-1100	0x1c	28	0111-0000	0x70	112
	l-QC33	011110	0001-1110	0x1e	30	0111-1000	0x78	120
	l-QC34	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS4	l-QC41	100010	0010-0010	0x22	34	1000-1000	0x88	136
	l-QC42	100100	0010-0100	0x24	36	1001-0000	0x90	144
	l-QC43	100110	0010-0110	0x26	38	1001-1000	0x98	152
	l-QC44	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS5	l-QC51	101010	0010-1010	0x2a	42	1010-1000	0xa8	168
	l-QC52	101100	0010-1100	0x2c	44	1011-0000	0xb0	176
	l-QC53	101110	0010-1110	0x2e	46	1011-1000	0xb8	184
	l-QC54	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS6	l-QC61	110010	0011-0010	0x32	50	1100-1000	0xc8	200
	l-QC62	110100	0011-0100	0x34	52	1101-0000	0xd0	208
	l-QC63	110110	0011-0110	0x36	54	1101-1000	0xd8	216
	l-QC64	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS7	l-QC71	111010	0011-1010	0x3a	58	1110-1000	0xe8	232
	l-QC72	111100	0011-1100	0x3c	60	1111-0000	0xf0	240
	l-QC73	111110	0011-1110	0x3e	62	1111-1000	0xf8	248
	l-QC74	000000	0000-0000	0x0	0	0000-0000	0x0	0
AS8	l-QC11	001010	0000-1010	0xa	10	0010-1000	0x28	40
	l-QC12	001100	0000-1100	0xc	12	0011-0000	0x30	48
	l-QC13	001110	0000-1110	0xe	14	0011-1000	0x38	56
	l-QC14	000000	0000-0000	0x0	0	0000-0000	0x0	0

Table 75: l-QC DSCP Values

13.4.1.15 Inter-domain Meta-QoS-classes DSCP values

This table summarizes DSCP values used between ASes in order to signal meta-QoS-classes. These values are used in both directions (i.e. upstream and downstream).

AS-AS	MC	b-DSCP	b-DSCP	x-DSCP	d-DSCP	b-DS	x-DS	d-DS
AS1-AS2	MC1	011010	0001-1010	0x1a	26	01101000	0x68	104
	MC2	011100	0001-1100	0x1c	28	01110000	0x70	112
	MC3	011110	0001-1110	0x1e	30	01111000	0x78	120
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS1-AS3	MC1	100010	0010-0010	0x22	34	10001000	0x88	136
	MC2	100100	0010-0100	0x24	36	10010000	0x90	144
	MC3	100110	0010-0110	0x26	38	10011000	0x98	152
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS2-AS4	MC1	110010	0011-0010	0x32	50	11001000	0xc8	200
	MC2	110100	0011-0100	0x34	52	11010000	0xd0	208
	MC3	110110	0011-0110	0x36	54	11011000	0xd8	216
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS2-AS5	MC1	111010	0011-1010	0x3a	58	11101000	0xe8	232
	MC2	111100	0011-1100	0x3c	60	11110000	0xf0	240
	MC3	111110	0011-1110	0x3e	62	11111000	0xf8	248
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS3-AS4	MC1	111010	0011-1010	0x3a	58	11101000	0xe8	232
	MC2	111100	0011-1100	0x3c	60	11110000	0xf0	240
	MC3	111110	0011-1110	0x3e	62	11111000	0xf8	248
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS4-AS5	MC1	001010	0000-1010	0xa	10	00101000	0x28	40
	MC2	001100	0000-1100	0xc	12	00110000	0x30	48
	MC3	001110	0000-1110	0xe	14	00111000	0x38	56
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS4-AS6	MC1	010010	0001-0010	0x12	18	01001000	0x48	72
	MC2	010100	0001-0100	0x14	20	01010000	0x50	80
	MC3	010110	0001-0110	0x16	22	01011000	0x58	88
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS6-AS7	MC1	101010	0010-1010	0x2a	42	10101000	0xa8	168
	MC2	101100	0010-1100	0x2c	44	10110000	0xb0	176
	MC3	101110	0010-1110	0x2e	46	10111000	0xb8	184
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0
AS6-AS8	MC1	111010	0011-1010	0x3a	58	11101000	0xe8	232
	MC2	111100	0011-1100	0x3c	60	11110000	0xf0	240
	MC3	111110	0011-1110	0x3e	62	11111000	0xf8	248
	MC4	000000	0000-0000	0x0	0	00000000	0x0	0

Table 76: Inter-domain meta-QoS-class DSCP values

13.4.1.16 Bandwidth Thresholds per Meta-QoS-class

This table illustrates the amount of bandwidth that is negotiated between two adjacent ASes and per direction. This amount is expressed in Mbit/s.

AS-AS	MC	BW (Mbit/s) →	BW (Mbit/s) ←
AS1-AS2	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS1-AS3	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS2-AS4	MC1	1	1
	MC2	1	1

	MC3	1	1
	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS2-AS5	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS3-AS4	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS4-AS5	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS4-AS6	MC1	3	3
	MC2	3	3
	MC3	3	3
	MC4	Max is 10 Min is 1	Max is 10 Min is 1
AS6-AS7	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS6-AS8	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2

Table 77: Bandwidth Threshold per meta-QoS-class and per pSLS

13.4.1.17 Maximum Bandwidth per local-QoS-class

This table summarizes the amount of bandwidth allocated per local QoS class. AS4 only is concerned since it is the sole AS in the platform to have intra-domain links.

AS	l-QC	BW in Mbit/s
AS4	l-QC1	5
	l-QC2	5
	l-QC3	5
	l-QC4	Maximum is 20 Min is 5

Table 78: Bandwidth Threshold per local-QoS-class

13.4.1.18 DiffServ-related Configuration

The implementation of the classes of service in the testbed will be achieved thanks to the activation of the HTB (Hierarchical Token Bucket), or the priority queuing discipline coupled with a HTB.

Preliminary tests we achieved shown that the Linux CBQ implementation had difficulties to handle more than 1.5Mbit/s of IP traffic. In addition, the TBF Linux implementation has a 1Mbit/s limitation.

In order to ease the configuration and the debugging operations, the following structure is followed for all routers present in the testbed.

Two files are created for each interface: qsi-ethx and qsHTB-ethx.

- **`qsi-ethx`**: this file contains the ingress related DiffServ policy configuration
- **`qsHTB-ethx`**: this file contains the egress related DiffServ policy configuration

In addition, the following files are used to execute the configuration of all interfaces.

- **`qsi`**: This script launches the ingress related DiffServ policy configuration for all relevant interfaces.
- **`qse`**: This script launches the egress related DiffServ policy configuration for all relevant interfaces.
- **`qsa`**: This script launches the ingress and the egress related DiffServ policy configuration for all relevant interfaces.
- **`qsdel`**: This script deletes all ingress and egress policies.

Below are listed some of these files that are used to configure DiffServ policies on **`MESCAL110`**:

13.4.1.18.1 qsa

```
#!/bin/bash

HOME_DSMARK=/home/mescal/scripts/DSmarking
$HOME_DSMARK/qsdel
$HOME_DSMARK/qsi
$HOME_DSMARK/qse
```

13.4.1.18.2 qsi

```
#!/bin/bash

HOME_DSMARK=/home/mescal/scripts/DSmarking
$HOME_DSMARK/qsi-eth1
$HOME_DSMARK/qsi-eth2
```

13.4.1.18.3 qse

```
#!/bin/bash

HOME_DSMARK=/home/mescal/scripts/DSmarking
$HOME_DSMARK/qshtb-eth1
$HOME_DSMARK/qshtb-eth2
```

13.4.1.18.4 qsdel

```
#!/bin/bash
tc qdisc del dev eth1 root
tc qdisc del dev eth2 root
tc qdisc del dev eth1 ingress
tc qdisc del dev eth2 ingress
```

13.4.1.18.5 qsi-eth1

```
#!/bin/bash

####
#Interfaces AS1-AS2
#______________________________
INGRESS=eth1
```

```
####
# Masks
#_____________________________
MASK1=0xfc
MASK2=0x03
MASK3=0xff

####
# Local QoS classes
#_____________________________
lQC1=0x28
lQC2=0x30
lQC3=0x38
lQC0=0x00

###
# Meta-QoS-classes as signaled by peers
#_____________________________
ICP210=0x00
ICP211=0x68
ICP212=0x70
ICP213=0x78

####
# Meta-QoS-classes as should be signaled if similar codes are
# used in both ways
#_____________________________
ICP120=0x00
ICP121=0x68
ICP122=0x70
ICP123=0x78

####
# Rates for policing purposes
#_____________________________
RATE_Total=10Mbit
RATE_ICP210=2.5Mbit
RATE_ICP211=2.5Mbit
RATE_ICP212=2.5Mbit
RATE_ICP213=2.5Mbit

####
# Attach an ingress qdisc to the $INGRESS interfaces
#_____________________________
echo $INGRESS "ingress policies configuration started........."
tc qdisc add dev $INGRESS handle ffff: ingress

#####
# Definition of filter that will put MC1 traffic in the relevant
# class
#_____________________________
tc filter add dev $INGRESS parent ffff: protocol ip prio 1 u32 \
match ip tos $ICP211 $MASK1 \
police rate $RATE_ICP211 buffer 10k \
drop flowid :1

#####
# Definition of filter that will put MC2 traffic in the relevant
# class. This class is dedicated to TCP traffic
#_____________________________
tc filter add dev $INGRESS parent ffff: protocol ip prio 1 u32 \
match ip tos $ICP212 $MASK1 \
police rate $RATE_ICP212 buffer 10k \
drop flowid :2
#match ip protocol 6 0xff \

#####
# Definition of filter that will put MC3 traffic in the relevant
# class. This class is dedicated to UDP traffic
#_____________________________
tc filter add dev $INGRESS parent ffff: protocol ip prio 1 u32 \
match ip tos $ICP213 $MASK1 \
police rate $RATE_ICP213 buffer 10k \
drop flowid :3

#####
# Definition of filter that will put MC0 traffic in the relevant
# class.
#_____________________________
tc filter add dev $INGRESS parent ffff: protocol ip prio 1 u32 \
match ip tos $ICP210 $MASK1 \
flowid :4

#####
# Definition of filter that will drop all other types of traffic
#_____________________________
tc filter add dev $INGRESS parent ffff: protocol ip prio 2 u32 match ip tos
0x0 0x0 police mtu 1 drop flowid :4

echo $INGRESS "....................................finished"
```

13.4.1.18.6 qsHTB-eth1

```
#!/bin/bash

####
#Interfaces AS1-AS2
#_____________________________
EGRESS=eth1

####
# Masks
#_____________________________
MASK1=0xfc
MASK2=0x03
MASK3=0xff

####
# Local QoS classes
#_____________________________
lQC1=0x28
lQC2=0x30
lQC3=0x38
lQC0=0x00

###
# Meta-QoS-classes as signaled by peers
#_____________________________
ICP210=0x00
ICP211=0x68
ICP212=0x70
ICP213=0x78
```

```
####
# Meta-QoS-classes as should be signaled to peers if similar codes are
# used in both ways
#______________________________
ICP120=0x00
ICP121=0x68
ICP122=0x70
ICP123=0x78

ICP130=0x00
ICP131=0x88
ICP132=0x90
ICP133=0x98
####
# Rates for policing purposes
#______________________________
RATE_Total=10Mbit

RATE_ICP120=2500Kbit
RATE_ICP121=2500Kbit
RATE_ICP122=2500Kbit
RATE_ICP123=2500Kbit

echo $EGRESS "egress policies configuration started....."
#####
# Attach a dsmark to the eth1 interface
#______________________________
tc qdisc add dev $EGRESS handle 1:0 root dsmark indices 8

#####
# Definition of four classes: MC1, MC2, MC3 and MC0
#______________________________
tc class change dev $EGRESS classid 1:1 dsmark mask $MASK2 value $ICP121
tc class change dev $EGRESS classid 1:2 dsmark mask $MASK2 value $ICP122
tc class change dev $EGRESS classid 1:3 dsmark mask $MASK2 value $ICP123
tc class change dev $EGRESS classid 1:4 dsmark mask $MASK2 value $ICP120


####
# Definition of filters that will be invoked in order to put the ingress
# traffic in the right class
#______________________________
tc filter add dev $EGRESS parent 1:0 prio 1 prot ip tcindex pass_on
tc filter add dev $EGRESS parent 1:0 prio 1 prot ip handle 1 tcindex
classid 1:1
tc filter add dev $EGRESS parent 1:0 prio 1 prot ip handle 2 tcindex
classid 1:2
tc filter add dev $EGRESS parent 1:0 prio 1 prot ip handle 3 tcindex
classid 1:3
tc filter add dev $EGRESS parent 1:0 prio 1 prot ip handle 4 tcindex
classid 1:4

####
# Definition of filters that will be invoked in order to put the local
# traffic in the right class
#______________________________
tc filter add dev $EGRESS parent 1:0 protocol ip prio 3 u32 match ip tos
$lQC1 $MASK1 classid 1:1
tc filter add dev $EGRESS parent 1:0 protocol ip prio 3 u32 match ip tos
$lQC2 $MASK1 classid 1:2
```

```
tc filter add dev $EGRESS parent 1:0 protocol ip prio 3 u32 match ip tos
$lQC3 $MASK1 classid 1:3
tc filter add dev $EGRESS parent 1:0 protocol ip prio 3 u32 match ip tos
$lQC0 $MASK1 classid 1:4

#####
# Definition of filters that will be used in order to put the traffic
# generated by local interfaces in the relevant egress classes
# Must add other realms if used to identify local interfaces
# Case of 1.1.1.0/30 and 1.1.1.4/30 realm
#______________________________
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP121 $MASK1 match ip src 1.1.1.1/30 classid 1:1
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP122 $MASK1 match ip src 1.1.1.1/30 classid 1:2
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP123 $MASK1 match ip src 1.1.1.1/30 classid 1:3
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP120 $MASK1 match ip src 1.1.1.1/30 classid 1:4

tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP121 $MASK1 match ip src 1.1.1.5/30 classid 1:1
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP122 $MASK1 match ip src 1.1.1.5/30 classid 1:2
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP123 $MASK1 match ip src 1.1.1.5/30 classid 1:3
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP120 $MASK1 match ip src 1.1.1.5/30 classid 1:4

tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP131 $MASK1 match ip src 1.1.1.1/30 classid 1:1
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP132 $MASK1 match ip src 1.1.1.1/30 classid 1:2
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP133 $MASK1 match ip src 1.1.1.1/30 classid 1:3
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP130 $MASK1 match ip src 1.1.1.1/30 classid 1:4

tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP131 $MASK1 match ip src 1.1.1.5/30 classid 1:1
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP132 $MASK1 match ip src 1.1.1.5/30 classid 1:2
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP133 $MASK1 match ip src 1.1.1.5/30 classid 1:3
tc filter add dev $EGRESS parent 1:0 protocol ip prio 4 u32 match ip tos
$ICP130 $MASK1 match ip src 1.1.1.5/30 classid 1:4
####
# A generic filter that will put other traffic in the BE class
#______________________________
tc filter add dev $EGRESS parent 1:0 prio 5 prot ip handle 0 tcindex
classid 1:4

####
# Definition of an HTB qdisc that is used to simulate a virtual link
#______________________________
tc qdisc add dev $EGRESS parent 1:0 handle 2:0 htb default 1
tc class add dev $EGRESS parent 2:0 classid 2:1 htb rate $RATE_Total burst
15kb

####
# Definition of an HTB qdisc that will be used to share bw between
# classes
#______________________________
tc qdisc add dev $EGRESS parent 2:1 handle 3:0 htb default 1
tc class add dev $EGRESS parent 3:0 classid 3:1 htb rate $RATE_Total burst
10kb

####
# Definition of an HTB class in order to shape the MC1 traffic.
# A pfifo/bfifo/sfq is attached to this class
#______________________________
tc class add dev $EGRESS parent 3:1 classid 3:10 htb rate $RATE_ICP121
burst 1500b prio 0
tc qdisc add dev $EGRESS parent 3:10 handle a:0 pfifo limit 128

#####
# Definition of an HTB class in order to shape the MC2 traffic.
# A pfifo/bfifo/sfq is attached to this class
#______________________________
tc class add dev $EGRESS parent 3:1 classid 3:11 htb rate $RATE_ICP122
burst 1500b prio 1
tc qdisc add dev $EGRESS parent 3:11 handle b:0 pfifo limit 128

####
# Definition of an HTB class in order to shape the MC3 traffic.
# A pfifo/bfifo/sfq is attached to this class
#______________________________
tc class add dev $EGRESS parent 3:1 classid 3:12 htb rate $RATE_ICP123
burst 1500b prio 2
tc qdisc add dev $EGRESS parent 3:12 handle c:0 pfifo limit 128

####
# Definition of an HTB class in order to shape the MC0 traffic.
# A pfifo/bfifo/sfq is attached to this class
#______________________________
tc class add dev $EGRESS parent 3:1 classid 3:13 htb rate $RATE_ICP120 ceil
$RATE_Total burst 1500b prio 3
tc qdisc add dev $EGRESS parent 3:13 handle d:0 pfifo limit 128

#####
# Definition of filter that will put the traffic in the relevant class
#______________________________
tc filter add dev $EGRESS parent 3:0 protocol ip prio 1 handle 1 tcindex
classid 3:10
tc filter add dev $EGRESS parent 3:0 protocol ip prio 1 handle 2 tcindex
classid 3:11
tc filter add dev $EGRESS parent 3:0 protocol ip prio 1 handle 3 tcindex
classid 3:12
tc filter add dev $EGRESS parent 3:0 protocol ip prio 1 handle 4 tcindex
classid 3:13

echo $EGRESS "................................finished"
```

13.4.1.19 Backup

In order to limit the impact of a system failure or a possible configuration error, we put in place a two-level backup procedure, which save most of the configuration files of the testbed.

The shared directory **`/home/mescal/backup`** has been created in **`PC-admin`** for this purpose. This folder is mounted in every router under: **`/mnt/backup`**. Routers data are saved using the **`backup`** command, which can be executed from each router.

In other hand, the **PC Admin** data is saved in **MESCAL110**.

On **PC-admin** side the "**backupall**" command will achieve a backup operation of all routers.

The **/home/mescal/backup** folder contains one sub-folder per router identified with the hostname of the router.

This folder contains again sub-folders containing data saved during a single backup operation. The name of each of these latter folders respects the following structure: **xxxx-DATE-TIME.** "**xxxx**" is set to "local" or "global" depending on the type of backup operation invoked. When backups are achieved with the **backup** command from a single router, **xxxx** takes the value "local" otherwise it is a general backup and it takes the value "global".

The **backup** command saves:

- The **DSmarking** folder
- The **Zebos.conf** file
- The **hosts** file

The second backup level consists in saving all **PC-admin** data in **MESCAL110** router. The following data are saved:

- The **common** folder
- The **scripts** folder
- The **hosts** file
- The **backup** folder

13.4.1.20 Logs

In order to verify quickly that all routers are correctly running (interface configured and services launched), every router is configured to report its BGP (bgpd and nsm daemons) and TC status.

A **crontab** list has been configured in each router that triggers a reporting every 10 min.

An example of the log file is provided below (example of **MESCAL110** router):

```
Tue May 25 17:21:00 CEST 2004-MESCAL110: bgpd running...
Tue May 25 17:21:00 CEST 2004-MESCAL110: nsm running...
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface lo
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface lo
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK qdisc hasn't been
configured for the interface lo
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured HTB root
classes is OK for interface eth0
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured HTB classes
is OK for the interface eth0
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured DSMARK
classes is OK for the interface eth0
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured HTB root
classes is OK for interface eth1
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured HTB classes
is OK for the interface eth1
Tue May 25 17:21:01 CEST 2004-MESCAL110: Number of configured DSMARK
classes is OK for the interface eth1
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface eth2
```

```
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface eth2
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK  qdisc hasn't
been configured for the interface eth2
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface eth3
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface eth3
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK  qdisc hasn't
been configured for the interface eth3
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface eth4
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface eth4
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK  qdisc hasn't
been configured for the interface eth4
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface eth5
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface eth5
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK qdisc hasn't been
configured for the interface eth5
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning An HTB root hasn't been
configured in interface eth6
Tue May 25 17:21:01 CEST 2004-MESCAL110: a Warning An HTB class hasn't been
configured for the interface eth6
Tue May 25 17:21:01 CEST 2004-MESCAL110: Warning A DSMARK qdisc hasn't been
configured for the interface eth6
```

13.4.1.21 Check the sanity of the test bed

In order to verify that all routers in the testbed are configured as expected, two scripts that check the status of the configuration have been developed: **`pfc`** and **`pfcheck`**. These scripts must be executed from **`PC-admin`**.

13.4.1.22 Configuration scripts

The table below lists useful scripts that are used for configuring the testbed. A description of the service they provide is also given.

Location	Script	description
Routers	pcm	This script allows mounting the log, transfer, common and backup folders.
	pcu	This script allows un-mounting the log, transfer, common and backup folders.
	qsa	This script configures all ingress and egress policies for a given router.
	qsdel	This script deletes all ingress and egress policies for a given router.
	qse	This script configures egress policies for a given router.
	qsi	This script configures ingress policies for a given router.
	qsi-ethj	This script configures ingress policies of the interface ethj of a given router.
	qshtb-ethj	This script configures egress policies of the interface ethj of a given router.
	qsstat	This script displays real-time egress TC related statistics

	qsstati	This script displays real-time ingress TC related statistics
	ifstat	This script displays the interface's statistics as reported by the kernel. If the argument "-s" is given the script sets to zero all statistics.
	bgpstart	This script starts the bgpd and nsm daemons
	bgpstop	This script stops the bgpd and nsm daemons
	isalive	This scripts returns the status of the bgpd and nsm daemons
	svty	This script launches the VTY. If the argument "-l" is given, the scripts loads the configuration stored in /usr/local/etc/ZeboS.conf
	pfgping	This scripts displays reachability information of all operational interfaces
	cgping	This scripts displays reachability information for all customers
	confstatus	This scripts returns the status of the configuration of a router
	backup	This script achieved a backup operation.
	pcabackup	This script can only be executed from MESCAL110. It achieved a PC-admin backup operation
	clean	This script recursively deletes all files ending with "~" in the /home/mescal folder.
PC-admin	lup	This script activates a given interface of a router
	ldown	This script deactivates a given interface of a router
	lstatus	This script returns the status of all routers interfaces
	rtrupdate	This script achieves an automatic update for a remote router
	updateall	This script achieves an automatic update for all routers
	rtrbackup	This script achieves an automatic backup for a remote router
	backupall	This script achieves an automatic backup for all routers
	pbackup	This script achieves a PC-admin backup
	pfc	This script displays the status of the testbed configuration without details and based on differed log files.
	pfcheck	This script displays the detailed status of the testbed configuration.
	initqos	This script initializes the QoS configuration of all routers
	delqos	This script deletes the QoS configuration of all routers
	egqos	This script initializes the egress QoS configuration of all routers
	ingqos	This script initializes the ingress QoS configuration of all routers
	iifstat	This script initializes the interface statistics of all routers

	lcap	This script configures the capabilities of Ethernet cards: lcap -s: restarts a negotiation between two back to back interfaces lcap -10: sets an interface to 10Mbit full duplex. lcap -100: sets an interfaces to 100Mbit full duplex.

Table 79: List of Useful Scripts

13.5 Specific Configuration for Phase 2

In the phase 2, the testbed architecture will be the same. The major difference will be the q-BGP activation.

The following scripts have been created for phase 2 purposes:

- initq-BGP: this scripts initialises all q-BGP routers.
- stopq-BGP: this scripts stops all q-BGP routers.
- q-BGPstart: this scripts initialises a q-BGP router.
- q-BGPstop: this scripts stops a q-BGP router.
- qisalive: this scripts tests if a q-BGP process is active.
- qsvty: this scripts launches an VTTY terminal for configuring q-BGP router.
- createPSLS: this scripts manages pSLSes.
- psls2q-BGP: this scripts configures q-BGP router according to a given pSLS.

13.6 Specific Configuration for Phase 3

In this phase, the configuration will be the same as for the phase 2. Nevertheless, we will install a PCS (Path Computation Server) in every AS. The table below shows the locations of these PCS:

PCS id	Router ID
PCS11	MESCAL110
PCS21	MESCAL210
PCS31	MESCAL310
PCS41	MESCAL410
PCS51	MESCAL510
PCS61	MESCAL610
PCS71	MESCAL710
PCS81	MESCAL810

Table 80: PCS Locations

Appendix B

14 DETAILED TESTBED VALIDATION TESTS

14.1 Phase 1

14.1.1 TB_P1_FUNCT/ROUT

Test Reference : **TB_P1_FUNCT/ROUT/1**

Test Purpose : Validate inter-domain link connectivity.

Procedure : Log into MESCAL11 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 1: 1.1.1.2
- Scenario 2: 1.1.1.6

Log into MESCAL21 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 3: 1.1.1.5
- Scenario 4: 2.2.2.2
- Scenario 5: 2.2.2.6

Log into MESCAL31 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 6: 1.1.1.1
- Scenario 7: 3.3.3.2
- Scenario 8: 3.3.3.6

Log into MESCAL41 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 9: 3.3.3.1
- Scenario 10: 41.41.41.10
- Scenario 11: 41.41.41.6
- Scenario 12: 41.41.41.2

Log into MESCAL42 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 13: 3.3.3.5
- Scenario 14: 2.2.2.1
- Scenario 15: 42.42.42.6
- Scenario 16: 42.42.42.2
- Scenario 17: 41.41.41.9

Log into MESCAL43 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 18: 42.42.42.1
- Scenario 19: 41.41.41.5
- Scenario 20: 43.43.43.2

Log into MESCAL51 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 21: 42.42.42.5

- Scenario 22: 2.2.2.5

Log into MESCAL61 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 23: 6.6.6.6
- Scenario 24: 41.41.41.1
- Scenario 25: 43.43.43.1
- Scenario 26: 6.6.6.2

Log into MESCAL71 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 27: 6.6.6.1

Log into MESCAL81 and verify that BGP process is disabled. Ping the following addresses:

- Scenario 28: 6.6.6.5

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success
- Scenario 12: success
- Scenario 13: success
- Scenario 14: success
- Scenario 15: success
- Scenario 16: success
- Scenario 17: success
- Scenario 18: success
- Scenario 19: success
- Scenario 20: success
- Scenario 21: success
- Scenario 22: success
- Scenario 23: success
- Scenario 24: success
- Scenario 25: success
- Scenario 26: success
- Scenario 27: success
- Scenario 28: success

Test Reference : **TB_P1_FUNCT/ROUT/2**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL11 and MESCAL21. Log to MESCAL 21 and configure MESCAL11 as a neighbor of MESCAL21. And log to MESCAL 11 and configure MESCAL21 as a neighbor of MESCAL11.

From MESCAL11 Ping the following addresses:

- Scenario 1: 1.1.1.6
- Scenario 2: 21.0.0.1
- Scenario 3: 2.2.2.5
- Scenario 4: 2.2.2.1
- Scenario 5: 2.2.2.2

From MESCAL21 Ping the following addresses:

- Scenario 6: 1.1.1.5
- Scenario 7: 1.1.1.1
- Scenario 8: 11.0.0.1
- Scenario 9: 11.0.0.2
- Scenario 10: 12.0.0.1
- Scenario 11: 12.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success

Test Reference : **TB_P1_FUNCT/ROUT/3**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL11 and MESCAL31. Log to MESCAL 31 and configure MESCAL11 as a neighbor. Log to MESCAL11 and configure MESCAL31 as a neighbor of MESCAL11.

From MESCAL11 Ping the following addresses:

- Scenario 1: 31.0.0.1
- Scenario 2: 3.3.3.1
- Scenario 3: 3.3.3.5
- Scenario 4: 1.1.1.2
- Scenario 5: 31.0.0.2

From MESCAL31 Ping the following addresses:

- Scenario 6: 1.1.1.5
- Scenario 7: 1.1.1.1
- Scenario 8: 11.0.0.1
- Scenario 9: 11.0.0.2
- Scenario 10: 12.0.0.1
- Scenario 11: 12.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success

- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success

Test Reference : **TB_P1_FUNCT/ROUT/4**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL21 and MESCAL42. Log to MESCAL 21 and configure MESCAL42 as a neighbor. And log to MESCAL42 and configure MESCAL21 as a neighbor.

From MESCAL42 Ping the following addresses:

- Scenario 1: 1.1.1.6
- Scenario 2: 21.0.0.1
- Scenario 3: 2.2.2.5
- Scenario 4: 2.2.2.1
- Scenario 5: 2.2.2.2

From MESCAL21 Ping the following addresses:

- Scenario 6: 3.3.3.6
- Scenario 7: 41.41.41.10
- Scenario 8: 42.42.42.1
- Scenario 9: 42.42.42.5
- Scenario 10: 2.2.2.2
- Scenario 11: 42.0.0.1
- Scenario 12: 42.0.0.2

Expected result : Results of ping request must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success
- Scenario 12: success

Test Reference : **TB_P1_FUNCT/ROUT/5**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL21 and MESCAL51. Log to MESCAL 21 and configure MESCAL51 as a neighbor. And log to MESCAL51 and configure MESCAL21 as a neighbor.

From MESCAL51 Ping the following addresses:

- Scenario 1: 1.1.1.6
- Scenario 2: 21.0.0.1
- Scenario 3: 2.2.2.5
- Scenario 4: 2.2.2.1
- Scenario 5: 2.2.2.2

From MESCAL21 Ping the following addresses:

- Scenario 6: 2.2.2.6
- Scenario 7: 42.42.42.6
- Scenario 8: 51.0.0.1
- Scenario 9: 51.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success

Test Reference : **TB_P1_FUNCT/ROUT/6**

Test Purpose : Validate connectivity between two neighbors when BGP process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL31 and MESCAL42. Log to MESCAL 31 and configure MESCAL42 as a neighbor. And log to MESCAL42 and configure MESCAL31 as a neighbor.

From MESCAL42 Ping the following addresses:

- Scenario 1: 31.0.0.1
- Scenario 2: 3.3.3.1
- Scenario 3: 3.3.3.5
- Scenario 4: 1.1.1.2
- Scenario 5: 31.0.0.2

From MESCAL31 Ping the following addresses:

- Scenario 6: 3.3.3.6
- Scenario 7: 41.41.41.10
- Scenario 8: 42.42.42.1
- Scenario 9: 42.42.42.5
- Scenario 10: 2.2.2.2
- Scenario 11: 42.0.0.1
- Scenario 12: 42.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success

- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success
- Scenario 12: success

Test Reference : **TB_P1_FUNCT/ROUT/7**

Test Purpose : Validate connectivity between two neighbors when BGP process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL31 and MESCAL41. Log to MESCAL 31 and configure MESCAL41 as a neighbor. And log to MESCAL41 and configure MESCAL31 as a neighbor.

From MESCAL41 Ping the following addresses:

- Scenario 1: 31.0.0.1
- Scenario 2: 3.3.3.1
- Scenario 3: 3.3.3.5
- Scenario 4: 1.1.1.2
- Scenario 5: 31.0.0.2

From MESCAL31 Ping the following addresses:

- Scenario 6: 3.3.3.2
- Scenario 7: 41.41.41.9
- Scenario 8: 41.41.41.5
- Scenario 9: 41.41.41.1
- Scenario 10: 41.0.0.1
- Scenario 11: 41.0.0.2

Expected result : Results of ping operation must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success

Test Reference : **TB_P1_FUNCT/ROUT/8**

Test Purpose : Validate connectivity between two neighbors when BGP process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL61 and MESCAL41. Log to MESCAL 61 and configure MESCAL41 as a neighbor. And log to MESCAL41 and configure MESCAL61 as a neighbor.

From MESCAL41 Ping the following addresses:

- Scenario 1: 61.0.0.1
- Scenario 2: 6.6.6.5
- Scenario 3: 6.6.6.1
- Scenario 4: 41.41.41.2
- Scenario 5: 43.43.43.2

From MESCAL61 Ping the following addresses:

- Scenario 6: 3.3.3.2
- Scenario 7: 41.41.41.9
- Scenario 8: 41.41.41.5
- Scenario 9: 41.41.41.1
- Scenario 10: 41.0.0.1
- Scenario 11: 41.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success

Test Reference : **TB_P1_FUNCT/ROUT/9**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL61 and MESCAL43. Log to MESCAL 61 and configure MESCAL43 as a neighbor. And log to MESCAL43 and configure MESCAL61 as a neighbor.

From MESCAL43 Ping the following addresses:

- Scenario 1: 61.0.0.1
- Scenario 2: 6.6.6.5
- Scenario 3: 6.6.6.1
- Scenario 4: 41.41.41.2
- Scenario 5: 43.43.43.2

From MESCAL61 Ping the following addresses:

- Scenario 6: 43.43.43.1
- Scenario 7: 41.41.41.6
- Scenario 8: 42.42.42.2
- Scenario 9: 43.0.0.1
- Scenario 10: 43.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success

Test Reference : **TB_P1_FUNCT/ROUT/10**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL51 and MESCAL42. Log to MESCAL 51 and configure MESCAL42 as a neighbor. And log to MESCAL42 and configure MESCAL51 as a neighbor.

From MESCAL42 Ping the following addresses:

- Scenario 1: 2.2.2.6
- Scenario 2: 42.42.42.6
- Scenario 3: 51.0.0.1
- Scenario 4: 51.0.0.2

From MESCAL51 Ping the following addresses:

- Scenario 5: 3.3.3.6
- Scenario 6: 41.41.41.10
- Scenario 7: 42.42.42.1
- Scenario 8: 42.42.42.5
- Scenario 9: 2.2.2.2
- Scenario 10: 42.0.0.1
- Scenario 11: 42.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success
- Scenario 11: success

Test Reference : **TB_P1_FUNCT/ROUT/11**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL61 and MESCAL71. Log to MESCAL 61 and configure MESCAL71 as a neighbor. And log to MESCAL71 and configure MESCAL61 as a neighbor.

From MESCAL71 Ping the following addresses:

- Scenario 1: 61.0.0.1
- Scenario 2: 6.6.6.5
- Scenario 3: 6.6.6.1
- Scenario 4: 41.41.41.2
- Scenario 5: 43.43.43.2

From MESCAL61 Ping the following addresses:

- Scenario 6: 6.6.6.2
- Scenario 7: 71.0.0.1
- Scenario 8: 71.0.0.2

Expected result : Results of ping operation must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success

Test Reference : **TB_P1_FUNCT/ROUT/12**

Test Purpose : Validate connectivity between two neighbors when BGPD process is activated.

Procedure : Activate BGPD and NSM processes in both MESCAL61 and MESCAL81. Log to MESCAL 61 and configure MESCAL81 as a neighbor. And log to MESCAL81 and configure MESCAL61 as a neighbor.

From MESCAL81 Ping the following addresses:

- Scenario 1: 61.0.0.1
- Scenario 2: 6.6.6.5
- Scenario 3: 6.6.6.1
- Scenario 4: 41.41.41.2
- Scenario 5: 43.43.43.2

From MESCAL61 Ping the following addresses:

- Scenario 6: 6.6.6.6
- Scenario 7: 81.0.0.1
- Scenario 8: 81.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success

Test Reference : **TB_P1_FUNCT/ROUT/13**
Test Purpose : Check the route propagation in a simple Scenario.
Procedure : Configure MESCAL11 to be a neighbor of MESCAL21 and MESCAL31. Configure MESCAL31 and MESCAL21 to be neighbors of MESCAL11. Activate BGPD and NSM processes in MESCAL11, MESCAL21 and MESCAL31.

From MESCAL31 Ping the following addresses:

- Scenario 1: 1.1.1.6
- Scenario 2: 21.0.0.1
- Scenario 3: 2.2.2.5
- Scenario 4: 2.2.2.1
- Scenario 5: 2.2.2.2

From MESCAL21 Ping the following addresses:

- Scenario 6: 1.1.1.2
- Scenario 7: 3.3.3.5
- Scenario 8: 3.3.3.1
- Scenario 9: 31.0.0.1
- Scenario 10: 31.0.0.2

Expected result : Results of ping requests must be as follows:

- Scenario 1: success
- Scenario 2: success
- Scenario 3: success
- Scenario 4: success
- Scenario 5: success
- Scenario 6: success
- Scenario 7: success
- Scenario 8: success
- Scenario 9: success
- Scenario 10: success

Test Reference : **TB_P1_FUNCT/ROUT/14**
Test Purpose : Check the reachability of all interfaces.
Procedure : Activate BGPD and NSM in all routers present in the testbed. BGP configuration is as described in Appendix A.

Log into the following routers and execute from bash prompt "**`pfgping`**":

- Scenario 1: MESCAL11
- Scenario 2: MESCAL21
- Scenario 3: MESCAL31
- Scenario 4: MESCAL41
- Scenario 5: MESCAL42
- Scenario 6: MESCAL43
- Scenario 7: MESCAL51
- Scenario 8: MESCAL61
- Scenario 9: MESCAL71
- Scenario 10: MESCAL81

Expected result : Results of "**`pfgping`**" must be as follows:

- Scenario 1: All destinations are successfully reached.
- Scenario 2: All destinations are successfully reached.

- Scenario 3: All destinations are successfully reached.
- Scenario 4: All destinations are successfully reached.
- Scenario 5: All destinations are successfully reached.
- Scenario 6: All destinations are successfully reached.
- Scenario 7: All destinations are successfully reached.
- Scenario 8: All destinations are successfully reached.
- Scenario 9: All destinations are successfully reached.
- Scenario 10: All destinations are successfully reached.

Test Reference : **TB_P1_FUNCT/ROUT/15**

Test Purpose : Verify the reachability when link failure occurs.

Procedure : Activate BGPD and NSM in all routers present in the testbed. BGP configuration is as described in Appendix A.

Execute the following operations:

- Scenario 1: Disable interface eth6 of MESCAL21, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 2: Disable interface eth1 of MESCAL31, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 3: Disable interface eth1 of MESCAL42, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 4: Disable interface eth4 of MESCAL42, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 5: Disable interfaces eth4 and eth1 of MESCAL42, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 6: Disable interfaces eth2 and eth3 of MESCAL42, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 7: Disable interfaces eth5 of MESCAL41, and from bash prompt of MESCAL11 execute "**`pfgping`**"
- Scenario 8: Disable interfaces eth5 of MESCAL41 and eth0 of MESCAL43, and from bash prompt of MESCAL11 execute "**`pfgping`**"

Expected result : Results of "**`pfgping`**" must be as follows:

- Scenario 1: All destinations are successfully reached except interface eth6 of MESCAL21.
- Scenario 2: All destinations are successfully reached except interface eth1 of MESCAL31.
- Scenario 3: All destinations are successfully reached except interface eth1 of MESCAL42.
- Scenario 4: All destinations are successfully reached except interface eth4 of MESCAL42.
- Scenario 5: All destinations are successfully reached except interface eth1 and eth4 of MESCAL42.
- Scenario 6: All destinations are successfully reached except interface eth2 and eth3 of MESCAL42.
- Scenario 7: All destinations are successfully reached except interface eth5 of MESCAL41.
- Scenario 8: All destinations are successfully reached except:
  Interface eth5 of MESCAL41
  Interface eth0 of MESCAL43
  All interfaces of MESCAL61
  All interfaces of MESCAL71
  All interfaces of MESCAL81

Test Reference : **TB_P1_FUNCT/ROUT/16**
Test Purpose : Verify the reachability when a link failure is re-established.
Procedure : Activate BGPD and NSM in all routers present in the testbed. BGP configuration is as described in Appendix A.

Execute the following operations:

- Scenario 1: Disable interface eth6 of MESCAL21. In few minutes activate this interface, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 2: Disable interface eth1 of MESCAL31. In few minutes activate this interface, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 3: Disable interface eth1 of MESCAL42. In few minutes activate this interface, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 4: Disable interface eth4 of MESCAL42. In few minutes activate this interface, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 5: Disable interfaces eth4 and eth1 of MESCAL42. In few minutes activate these interfaces, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 6: Disable interfaces eth2 and eth3 of MESCAL42. In few minutes activate these interfaces, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 7: Disable interfaces eth5 of MESCAL41. In few minutes activate this interface, and from bash prompt of MESCAL11 execute "`pfgping`"
- Scenario 8: Disable interfaces eth5 of MESCAL41 and eth0 of MESCAL43. In few minutes activate these interfaces, and from bash prompt of MESCAL11 execute "`pfgping`"

Expected result : Results of "`pfgping`" must be as follows:

- Scenario 1: All destinations are successfully reached.
- Scenario 2: All destinations are successfully reached.
- Scenario 3: All destinations are successfully reached.
- Scenario 4: All destinations are successfully reached.
- Scenario 5: All destinations are successfully reached.
- Scenario 6: All destinations are successfully reached.
- Scenario 7: All destinations are successfully reached.
- Scenario 8: All destinations are successfully reached.
- Scenario 9: All destinations are successfully reached.
- Scenario 10: All destinations are successfully reached.

Test Reference : **TB_P1_FUNCT/ROUT/17**
Test Purpose : Verify intra-domain routing in AS4.
Procedure :

Configure the following static routes in MESCAL42:

- 41.41.41.1/30 via 41.41.41.9 metric 1
- 41.41.41.1/30 via 42.42.42.2 metric 2
- 43.43.43.1/30 via 41.41.41.9 metric 2
- 43.43.43.1/30 via 42.42.42.9 metric 1

Configure the following static routes in MESCAL41:

- 43.43.43.1/30 via 41.41.41.10 metric 2
- 43.43.43.1/30 via 41.41.41.6 metric 1

Configure the following static routes in MESCAL43:

- 41.41.41.1/30 via 41.41.41.5 metric 1
- 41.41.41.1/30 via 42.42.42.1 metric 2

Log into MESCAL42. From a bash prompt, ping the following addresses:

- Scenario 1: 41.41.41.1
- Scenario 2: 43.43.43.1

Disable interface eth2 of MESCAL43, and then ping the following addresses:

- Scenario 3: 41.41.41.1
- Scenario 4: 43.43.43.1

Disable interface eth4 of MESCAL41, and then ping the following addresses:

- Scenario 5: 41.41.41.1
- Scenario 6: 43.43.43.1

Enable interface eth2 of MESCAL43, and then ping the following addresses:

- Scenario 7: 41.41.41.1
- Scenario 8: 43.43.43.1

Disable interface eth3 of MESCAL41, and then ping the following addresses:

- Scenario 9: 41.41.41.1
- Scenario 10: 43.43.43.1

Expected result : Results of ping requests must be as follows:

- Scenario 1: Success.
- Scenario 2: Success.
- Scenario 3: Success.
- Scenario 4: Success.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: Success.
- Scenario 8: Success.
- Scenario 9: Success.
- Scenario 10: Unreachable destination.

14.1.2 TB_P1_FUNCT/DSSW

In this section, we list tests that aim at verifying DSCP swapping operations in both ingress and egress of ASes. Before starting executing these tests, it is recommended to verify that the DSCP values that have been assigned in the testbed configuration are correctly settled in **`qsi`** and **`qse`** scripts in all routers of the testbed.

Test Reference : **TB_P1_FUNCT/DSSW/1**
Test Purpose : Verify DSCP swapping at egress of MESCAL11.
Procedure : Configure local-QoS-classes of MESCAL11 as defined in testbed Configuration section. Practically, execute **`qsa`** script in MESCAL11.

Launch a traffic analyzer in:

- MESCAL31 that captures traffic going through eth1.
- MESCAL21 that captures traffic going through eth6.

Execute successively the following commands from a bash terminal of MESCAL11:

- Scenario 1: ping –Q 0x28 1.1.1.2
- Scenario 2: ping –Q 0x30 1.1.1.2
- Scenario 3: ping –Q 0x38 1.1.1.2
- Scenario 4: ping 1.1.1.2
- Scenario 5: ping –Q 0x45 1.1.1.2
- Scenario 6: ping –Q 0x144 1.1.1.2
- Scenario 7: ping –Q 0x28 1.1.1.6
- Scenario 8: ping –Q 0x30 1.1.1.6
- Scenario 9: ping –Q 0x38 1.1.1.6
- Scenario 10: ping 1.1.1.6
- Scenario 11: ping –Q 0x98 1.1.1.6
- Scenario 12: ping –Q 0x144 1.1.1.6

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x88.
- Scenario 2: ping requests are received with a TOS value of 0x90.
- Scenario 3: ping requests are received with a TOS value of 0x98.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0x68.
- Scenario 8: ping requests are received with a TOS value of 0x70.
- Scenario 9: ping requests are received with a TOS value of 0x78.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/2**
Test Purpose : Verify DSCP swapping at egress of MESCAL21.
Procedure : Configure local-QoS-classes of MESCAL21 as defined in testbed Configuration section. Practically, execute `qsa` script in MESCAL21.

Launch a traffic analyzer in:

- MESCAL11 that captures traffic going through eth1.
- MESCAL42 that captures traffic going through eth4.
- MESCAL51 that captures traffic going through eth2.

Execute successively the following commands from a bash terminal of MESCAL21:

- Scenario 1: ping –Q 0x48 1.1.1.5
- Scenario 2: ping –Q 0x50 1.1.1.5
- Scenario 3: ping –Q 0x58 1.1.1.5
- Scenario 4: ping 1.1.1.5
- Scenario 5: ping –Q 0x98 1.1.1.5
- Scenario 6: ping –Q 0x144 1.1.1.5
- Scenario 7: ping –Q 0x48 2.2.2.2
- Scenario 8: ping –Q 0x50 2.2.2.2
- Scenario 9: ping –Q 0x58 2.2.2.2
- Scenario 10: ping 2.2.2.2
- Scenario 11: ping –Q 0x98 2.2.2.2
- Scenario 12: ping –Q 0x144 2.2.2.2
- Scenario 13: ping –Q 0x48 2.2.2.6
- Scenario 14: ping –Q 0x50 2.2.2.6

- Scenario 15: ping –Q 0x58 2.2.2.6
- Scenario 16: ping 2.2.2.6
- Scenario 17: ping –Q 0x98 2.2.2.6
- Scenario 18: ping –Q 0x144 2.2.2.6

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x68.
- Scenario 2: ping requests are received with a TOS value of 0x70.
- Scenario 3: ping requests are received with a TOS value of 0x78.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0xc8.
- Scenario 8: ping requests are received with a TOS value of 0xd0.
- Scenario 9: ping requests are received with a TOS value of 0xd8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0xe8.
- Scenario 14: ping requests are received with a TOS value of 0xf0.
- Scenario 15: ping requests are received with a TOS value of 0xf8.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/3**
Test Purpose : Verify DSCP swapping at egress of MESCAL31.
Procedure : Configure local-QoS-classes of MESCAL31 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL31.

Launch a traffic analyzer in:

- MESCAL11 that captures traffic going through eth0.
- MESCAL42 that captures traffic going through eth1.
- MESCAL41 that captures traffic going through eth0.

Execute successively the following commands from a bash terminal of MESCAL31:

- Scenario 1: ping –Q 0x68 1.1.1.1
- Scenario 2: ping –Q 0x70 1.1.1.1
- Scenario 3: ping –Q 0x78 1.1.1.1
- Scenario 4: ping 1.1.1.1
- Scenario 5: ping –Q 0x98 1.1.1.1
- Scenario 6: ping –Q 0x144 1.1.1.1
- Scenario 7: ping –Q 0x68 3.3.3.6
- Scenario 8: ping –Q 0x70 3.3.3.6
- Scenario 9: ping –Q 0x78 3.3.3.6
- Scenario 10: ping 3.3.3.6
- Scenario 11: ping –Q 0x98 3.3.3.6
- Scenario 12: ping –Q 0x144 3.3.3.6
- Scenario 13: ping –Q 0x68 3.3.3.2
- Scenario 14: ping –Q 0x70 3.3.3.2
- Scenario 15: ping –Q 0x78 3.3.3.2
- Scenario 16: ping 3.3.3.2
- Scenario 17: ping –Q 0x98 3.3.3.2

- Scenario 18: ping –Q 0x144 3.3.3.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x88.
- Scenario 2: ping requests are received with a TOS value of 0x90.
- Scenario 3: ping requests are received with a TOS value of 0x98.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0xe8.
- Scenario 8: ping requests are received with a TOS value of 0xf0.
- Scenario 9: ping requests are received with a TOS value of 0xf8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0xe8.
- Scenario 14: ping requests are received with a TOS value of 0xf0.
- Scenario 15: ping requests are received with a TOS value of 0xf8.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/4**
Test Purpose : Verify DSCP swapping at egress of MESCAL41.
Procedure : Configure local-QoS-classes of MESCAL41 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL41.

Launch a traffic analyzer in:

- MESCAL31 that captures traffic going through eth6.
- MESCAL42 that captures traffic going through eth2.
- MESCAL43 that captures traffic going through eth1.
- MESCAL61 that captures traffic going through eth4.

Execute successively the following commands from a bash terminal of MESCAL41:

- Scenario 1: ping –Q 0x88 3.3.3.1
- Scenario 2: ping –Q 0x90 3.3.3.1
- Scenario 3: ping –Q 0x98 3.3.3.1
- Scenario 4: ping 3.3.3.1
- Scenario 5: ping –Q 0x55 3.3.3.1
- Scenario 6: ping –Q 0x144 3.3.3.1
- Scenario 7: ping –Q 0x88 41.41.41.10
- Scenario 8: ping –Q 0x90 41.41.41.10
- Scenario 9: ping –Q 0x98 41.41.41.10
- Scenario 10: ping 41.41.41.10
- Scenario 11: ping –Q 0x55 41.41.41.10
- Scenario 12: ping –Q 0x144 41.41.41.10
- Scenario 13: ping –Q 0x88 41.41.41.6
- Scenario 14: ping –Q 0x90 41.41.41.6
- Scenario 15: ping –Q 0x98 41.41.41.6
- Scenario 16: ping 41.41.41.6
- Scenario 17: ping –Q 0x38 41.41.41.6
- Scenario 18: ping –Q 0x144 41.41.41.6
- Scenario 19: ping –Q 0x88 41.41.41.2
- Scenario 20: ping –Q 0x90 41.41.41.2

- Scenario 21: ping –Q 0x98 41.41.41.2
- Scenario 22: ping 41.41.41.2
- Scenario 23: ping –Q 0x38 41.41.41.2
- Scenario 24: ping –Q 0x144 41.41.41.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0x88.
- Scenario 8: ping requests are received with a TOS value of 0x90.
- Scenario 9: ping requests are received with a TOS value of 0x98.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0x88.
- Scenario 14: ping requests are received with a TOS value of 0x90.
- Scenario 15: ping requests are received with a TOS value of 0x98.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.
- Scenario 19: ping requests are received with a TOS value of 0x48.
- Scenario 20: ping requests are received with a TOS value of 0x50.
- Scenario 21: ping requests are received with a TOS value of 0x58.
- Scenario 22: ping requests are received with a TOS value of 0x00.
- Scenario 23: ping requests are received with a TOS value of 0x00.
- Scenario 24: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/5**
Test Purpose : Verify DSCP swapping at egress of MESCAL42.
Procedure : Configure local-QoS-classes of MESCAL42 as defined in testbed Configuration section. Practically, execute **`qsa`** script in MESCAL42.

Launch a traffic analyzer in:

- MESCAL21 that captures traffic going through eth0.
- MESCAL31 that captures traffic going through eth0.
- MESCAL41 that captures traffic going through eth4.
- MESCAL43 that captures traffic going through eth2.
- MESCAL51 that captures traffic going through eth1.

Execute successively the following commands from a bash terminal of MESCAL42:

- Scenario 1: ping –Q 0x88 2.2.2.1
- Scenario 2: ping –Q 0x90 2.2.2.1
- Scenario 3: ping –Q 0x98 2.2.2.1
- Scenario 4: ping 2.2.2.1
- Scenario 5: ping –Q 0x55 2.2.2.1
- Scenario 6: ping –Q 0x144 2.2.2.1
- Scenario 7: ping –Q 0x88 3.3.3.5
- Scenario 8: ping –Q 0x90 3.3.3.5
- Scenario 9: ping –Q 0x98 3.3.3.5
- Scenario 10: ping 3.3.3.5

- Scenario 11: ping –Q 0x55 3.3.3.5
- Scenario 12: ping –Q 0x144 3.3.3.5
- Scenario 13: ping –Q 0x88 41.41.41.9
- Scenario 14: ping –Q 0x90 41.41.41.9
- Scenario 15: ping –Q 0x98 41.41.41.9
- Scenario 16: ping 41.41.41.9
- Scenario 17: ping –Q 0x38 41.41.41.9
- Scenario 18: ping –Q 0x144 41.41.41.9
- Scenario 19: ping –Q 0x88 42.42.42.1
- Scenario 20: ping –Q 0x90 42.42.42.1
- Scenario 21: ping –Q 0x98 42.42.42.1
- Scenario 22: ping 42.42.42.1
- Scenario 23: ping –Q 0x38 42.42.42.1
- Scenario 24: ping –Q 0x144 42.42.42.1
- Scenario 25: ping –Q 0x88 42.42.42.6
- Scenario 26: ping –Q 0x90 42.42.42.6
- Scenario 27: ping –Q 0x98 42.42.42.6
- Scenario 28: ping 42.42.42.6
- Scenario 29: ping –Q 0x38 42.42.42.6
- Scenario 30: ping –Q 0x144 42.42.42.6

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xc8.
- Scenario 2: ping requests are received with a TOS value of 0xd0.
- Scenario 3: ping requests are received with a TOS value of 0xd8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0xe8.
- Scenario 8: ping requests are received with a TOS value of 0xf0.
- Scenario 9: ping requests are received with a TOS value of 0xf8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0x88.
- Scenario 14: ping requests are received with a TOS value of 0x90.
- Scenario 15: ping requests are received with a TOS value of 0x98.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.
- Scenario 19: ping requests are received with a TOS value of 0x88.
- Scenario 20: ping requests are received with a TOS value of 0x80.
- Scenario 21: ping requests are received with a TOS value of 0x98.
- Scenario 22: ping requests are received with a TOS value of 0x00.
- Scenario 23: ping requests are received with a TOS value of 0x00.
- Scenario 24: ping requests are received with a TOS value of 0x00.
- Scenario 25: ping requests are received with a TOS value of 0x28.
- Scenario 26: ping requests are received with a TOS value of 0x30.
- Scenario 27: ping requests are received with a TOS value of 0x38.
- Scenario 28: ping requests are received with a TOS value of 0x00.
- Scenario 29: ping requests are received with a TOS value of 0x00.
- Scenario 30: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/6**

Test Purpose : Verify DSCP swapping at egress of MESCAL43.
Procedure : Configure local-QoS-classes of MESCAL43 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL43.

Launch a traffic analyzer in:

- MESCAL41 that captures traffic going through eth3.
- MESCAL42 that captures traffic going through eth3.
- MESCAL61 that captures traffic going through eth1.

Execute successively the following commands from a bash terminal of MESCAL43:

- Scenario 1: ping –Q 0x88 41.41.41.5
- Scenario 2: ping –Q 0x90 41.41.41.5
- Scenario 3: ping –Q 0x98 41.41.41.5
- Scenario 4: ping 41.41.41.5
- Scenario 5: ping –Q 0x55 41.41.41.5
- Scenario 6: ping –Q 0x144 41.41.41.5
- Scenario 7: ping –Q 0x88 42.42.42.1
- Scenario 8: ping –Q 0x90 42.42.42.1
- Scenario 9: ping –Q 0x98 42.42.42.1
- Scenario 10: ping 42.42.42.1
- Scenario 11: ping –Q 0x55 42.42.42.1
- Scenario 12: ping –Q 0x144 42.42.42.1
- Scenario 13: ping –Q 0x88 43.43.43.2
- Scenario 14: ping –Q 0x90 43.43.43.2
- Scenario 15: ping –Q 0x98 43.43.43.2
- Scenario 16: ping 43.43.43.2
- Scenario 17: ping –Q 0x38 43.43.43.2
- Scenario 18: ping –Q 0x144 43.43.43.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x88.
- Scenario 2: ping requests are received with a TOS value of 0x90.
- Scenario 3: ping requests are received with a TOS value of 0x98.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0x88.
- Scenario 8: ping requests are received with a TOS value of 0x90.
- Scenario 9: ping requests are received with a TOS value of 0x98.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0x48.
- Scenario 14: ping requests are received with a TOS value of 0x50.
- Scenario 15: ping requests are received with a TOS value of 0x58.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/7**
Test Purpose : Verify DSCP swapping at egress of MESCAL51.
Procedure : Configure local-QoS-classes of MESCAL51 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL51.

Launch a traffic analyzer in:

- MESCAL21 that captures traffic going through eth5.
- MESCAL42 that captures traffic going through eth5.

Execute successively the following commands from a bash terminal of MESCAL51:

- Scenario 1: ping –Q 0xa8 2.2.2.5
- Scenario 2: ping –Q 0xb0 2.2.2.5
- Scenario 3: ping –Q 0xb8 2.2.2.5
- Scenario 4: ping 2.2.2.5
- Scenario 5: ping –Q 0x55 2.2.2.5
- Scenario 6: ping –Q 0x144 2.2.2.5
- Scenario 7: ping –Q 0xa8 42.42.42.5
- Scenario 8: ping –Q 0xb0 42.42.42.5
- Scenario 9: ping –Q 0xb8 42.42.42.5
- Scenario 10: ping 42.42.42.5
- Scenario 11: ping –Q 0x55 42.42.42.5
- Scenario 12: ping –Q 0x144 42.42.42.5

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0x28.
- Scenario 8: ping requests are received with a TOS value of 0x30.
- Scenario 9: ping requests are received with a TOS value of 0x38.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/8**

Test Purpose : Verify DSCP swapping at egress of MESCAL61.

Procedure : Configure local-QoS-classes of MESCAL61 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL61.

Launch a traffic analyzer in:

- MESCAL41 that captures traffic going through eth5.
- MESCAL43 that captures traffic going through eth0.
- MESCAL71 that captures traffic going through eth1.
- MESCAL81 that captures traffic going through eth2.

Execute successively the following commands from a bash terminal of MESCAL61:

- Scenario 1: ping –Q 0xc8 41.41.41.1
- Scenario 2: ping –Q 0xd0 41.41.41.1
- Scenario 3: ping –Q 0xd8 41.41.41.1
- Scenario 4: ping 41.41.41.1
- Scenario 5: ping –Q 0x55 41.41.41.1
- Scenario 6: ping –Q 0x144 41.41.41.1
- Scenario 7: ping –Q 0xc8 43.43.43.1
- Scenario 8: ping –Q 0xd0 43.43.43.1
- Scenario 9: ping –Q 0xd8 43.43.43.1

- Scenario 10: ping 43.43.43.1
- Scenario 11: ping –Q 0x55 43.43.43.1
- Scenario 12: ping –Q 0x144 43.43.43.1
- Scenario 13: ping –Q 0xc8 6.6.6.2
- Scenario 14: ping –Q 0xd0 6.6.6.2
- Scenario 15: ping –Q 0xd8 6.6.6.2
- Scenario 16: ping 6.6.6.2
- Scenario 17: ping –Q 0x55 6.6.6.2
- Scenario 18: ping –Q 0x144 6.6.6.2
- Scenario 19: ping –Q 0xc8 6.6.6.6
- Scenario 20: ping –Q 0xd0 6.6.6.6
- Scenario 21: ping –Q 0xd8 6.6.6.6
- Scenario 22: ping 6.6.6.6
- Scenario 23: ping –Q 0x55 6.6.6.6
- Scenario 24: ping –Q 0x144 6.6.6.6

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x48.
- Scenario 2: ping requests are received with a TOS value of 0x50.
- Scenario 3: ping requests are received with a TOS value of 0x58.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.
- Scenario 7: ping requests are received with a TOS value of 0x48.
- Scenario 8: ping requests are received with a TOS value of 0x50.
- Scenario 9: ping requests are received with a TOS value of 0x58.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: ping requests are received with a TOS value of 0x00.
- Scenario 12: ping requests are received with a TOS value of 0x00.
- Scenario 13: ping requests are received with a TOS value of 0xa8.
- Scenario 14: ping requests are received with a TOS value of 0xb0.
- Scenario 15: ping requests are received with a TOS value of 0xb8.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: ping requests are received with a TOS value of 0x00.
- Scenario 18: ping requests are received with a TOS value of 0x00.
- Scenario 19: ping requests are received with a TOS value of 0xe8.
- Scenario 20: ping requests are received with a TOS value of 0xf0.
- Scenario 21: ping requests are received with a TOS value of 0xf8.
- Scenario 22: ping requests are received with a TOS value of 0x00.
- Scenario 23: ping requests are received with a TOS value of 0x00.
- Scenario 24: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/9**
Test Purpose : Verify DSCP swapping at egress of MESCAL71.
Procedure : Configure local-QoS-classes of MESCAL71 as defined in testbed Configuration section. Practically, execute **qsa** script in MESCAL71.

Launch a traffic analyzer in:

- MESCAL61 that captures traffic going through eth2.

Execute successively the following commands from a bash terminal of MESCAL71:

- Scenario 1: ping –Q 0xe8 6.6.6.1
- Scenario 2: ping –Q 0xf0 6.6.6.1
- Scenario 3: ping –Q 0xf8 6.6.6.1

- Scenario 4: ping 6.6.6.1
- Scenario 5: ping –Q 0x55 6.6.6.1
- Scenario 6: ping –Q 0x144 6.6.6.1

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xa8.
- Scenario 2: ping requests are received with a TOS value of 0xb0.
- Scenario 3: ping requests are received with a TOS value of 0xb8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/10**

Test Purpose : Verify DSCP swapping at egress of MESCAL81.

Procedure : Configure local-QoS-classes of MESCAL81 as defined in testbed Configuration section. Practically, execute **`qsa`** script in MESCAL81.

Launch a traffic analyzer in:

- MESCAL61 that captures traffic going through eth3.

Execute successively the following commands from a bash terminal of MESCAL81:

- Scenario 1: ping –Q 0xe8 6.6.6.5
- Scenario 2: ping –Q 0xf0 6.6.6.5
- Scenario 3: ping –Q 0xf8 6.6.6.5
- Scenario 4: ping 6.6.6.5
- Scenario 5: ping –Q 0x55 6.6.6.5
- Scenario 6: ping –Q 0x144 6.6.6.5

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: ping requests are received with a TOS value of 0x00.
- Scenario 6: ping requests are received with a TOS value of 0x00.

Test Reference : **TB_P1_FUNCT/DSSW/11**

Test Purpose : Verify DSCP swapping at ingress of MESCAL11.

Procedure : execute **`qsdel`** in both MESCAL31 and MESCAL21 and **`qsi`** in MESCAL11. Launch a traffic analyzer in MESCAL21 that captures traffic going through eth6. Also launch a traffic analyzer in MESCAL31 that captures traffic going through eth1. Ensure that BGPD is started between AS1 and AS2, AS1 and AS3.

Execute successively the following commands in MESCAL31; examine traffic going through interface eth0 of MESCAL11:

- Scenario 1: ping –Q 0x88 1.1.1.6
- Scenario 2: ping –Q 0x90 1.1.1.6
- Scenario 3: ping –Q 0x98 1.1.1.6
- Scenario 4: ping 1.1.1.6
- Scenario 5: ping –Q 0x53 1.1.1.6
- Scenario 6: ping –Q 0x144 1.1.1.6

Execute successively the following commands in MESCAL21; examine traffic going through interface eth1 of MESCAL11:

- Scenario 7: ping –Q 0x68 1.1.1.2
- Scenario 8: ping –Q 0x70 1.1.1.2
- Scenario 9: ping –Q 0x78 1.1.1.2
- Scenario 10: ping 1.1.1.6
- Scenario 11: ping –Q 0x98 1.1.1.2
- Scenario 12: ping –Q 0x144 1.1.1.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x28.
- Scenario 2: ping requests are received with a TOS value of 0x30.
- Scenario 3: ping requests are received with a TOS value of 0x38.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0x28.
- Scenario 8: ping requests are received with a TOS value of 0x30.
- Scenario 9: ping requests are received with a TOS value of 0x38.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/12**

Test Purpose : Verify DSCP swapping at ingress of MESCAL21.

Procedure : Ensure that BGP sessions are activated between AS2 and AS1, AS2 and AS4 and AS2 and AS5. Execute **`qsa`** in MESCAL21 and **`qsdel`** script in:

- MESCAL11
- MESCAL42
- MESCAL51

Launch a traffic analyzer in the following interfaces:

- Eth1 of MESCAL11
- Eth4 of MESCAL42

Execute successively the following commands from a bash terminal of MESCAL11; examine traffic going through eth4 of MESCAL42:

- Scenario 1: ping –Q 0x68 2.2.2.2
- Scenario 2: ping –Q 0x70 2.2.2.2
- Scenario 3: ping –Q 0x78 2.2.2.2
- Scenario 4: ping 2.2.2.2
- Scenario 5: ping –Q 0x98 2.2.2.2
- Scenario 6: ping –Q 0x144 2.2.2.2

Execute successively the following commands from a bash terminal of MESCAL42; examine traffic going through eth1 of MESCAL1:

- Scenario 7: ping –Q 0xc8 1.1.1.5
- Scenario 8: ping –Q 0xd0 1.1.1.5
- Scenario 9: ping –Q 0xd8 1.1.1.5
- Scenario 10: ping 2.2.2.1

- Scenario 11: ping –Q 0x98 1.1.1.5
- Scenario 12: ping –Q 0x144 1.1.1.5

Execute successively the following commands from a bash terminal of MESCAL51; examine traffic going through eth1 of MESCAL11:

- Scenario 13: ping –Q 0xe8 1.1.1.5
- Scenario 14: ping –Q 0xf0 1.1.1.5
- Scenario 15: ping –Q 0xf8 1.1.1.5
- Scenario 16: ping 1.1.1.5
- Scenario 17: ping –Q 0x98 1.1.1.5
- Scenario 18: ping –Q 0x144 1.1.1.5

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xc8.
- Scenario 2: ping requests are received with a TOS value of 0xd0.
- Scenario 3: ping requests are received with a TOS value of 0xd8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0x68.
- Scenario 8: ping requests are received with a TOS value of 0x70.
- Scenario 9: ping requests are received with a TOS value of 0x78.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.
- Scenario 13: ping requests are received with a TOS value of 0x68.
- Scenario 14: ping requests are received with a TOS value of 0x70.
- Scenario 15: ping requests are received with a TOS value of 0x78.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: Unreachable destination.
- Scenario 18: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/13**

Test Purpose : Verify DSCP swapping at ingress of MESCAL31.

Procedure : Ensure that BGP sessions are activated between AS3 and AS1 and AS3 and AS4. Execute **qsa** in MESCAL21 and **qsdel** script in:

- MESCAL11
- MESCAL42
- MESCAL41

Launch a traffic analyzer in the following interfaces:

- Eth0 of MESCAL11
- Eth1 of MESCAL42

Execute successively the following commands from a bash terminal of MESCAL11; examine traffic going through eth1 of MESCAL42:

- Scenario 1: ping –Q 0x88 3.3.3.6
- Scenario 2: ping –Q 0x90 3.3.3.6
- Scenario 3: ping –Q 0x98 3.3.3.6
- Scenario 4: ping 3.3.3.6
- Scenario 5: ping –Q 0x55 3.3.3.6
- Scenario 6: ping –Q 0x144 3.3.3.6

Execute successively the following commands from a bash terminal of MESCAL41; examine traffic going through eth0 of MESCAL11:

- Scenario 7: ping –Q 0xe8 1.1.1.1
- Scenario 8: ping –Q 0xf0 1.1.1.1
- Scenario 9: ping –Q 0xf8 1.1.1.1
- Scenario 10: ping 1.1.1.1
- Scenario 11: ping –Q 0x98 1.1.1.1
- Scenario 12: ping –Q 0x144 1.1.1.1

Execute successively the following commands from a bash terminal of MESCAL42; examine traffic going through eth0 of MESCAL11:

- Scenario 13: ping –Q 0xe8 1.1.1.1
- Scenario 14: ping –Q 0xf0 1.1.1.1
- Scenario 15: ping –Q 0xf8 1.1.1.1
- Scenario 16: ping 1.1.1.1
- Scenario 17: ping –Q 0x98 1.1.1.1
- Scenario 18: ping –Q 0x144 1.1.1.1

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0x88.
- Scenario 8: ping requests are received with a TOS value of 0x90.
- Scenario 9: ping requests are received with a TOS value of 0x98.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.
- Scenario 13: ping requests are received with a TOS value of 0x88.
- Scenario 14: ping requests are received with a TOS value of 0x90.
- Scenario 15: ping requests are received with a TOS value of 0x98.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: Unreachable destination.
- Scenario 18: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/14**

Test Purpose : Verify DSCP swapping at ingress of MESCAL41.

Procedure : Ensure that BGP is activated between AS4 and AS AS4 and AS6. Execute **`qsdel`** in MESCAL31 and MESCAL61. And execute **`qsa`** in MESCAL41.

Execute successively the following commands from a bash terminal of MESCAL61, check then what is received in eth6 of MESCAL31:

- Scenario 1: ping –Q 0x48 3.3.3.1
- Scenario 2: ping –Q 0x50 3.3.3.1
- Scenario 3: ping –Q 0x58 3.3.3.1
- Scenario 4: ping 3.3.3.1
- Scenario 5: ping –Q 0x70 3.3.3.1
- Scenario 6: ping –Q 0x144 3.3.3.1

Execute successively the following commands from a bash terminal of MESCAL31, check then what is received in eth4 of MESCAL61:

- Scenario 7: ping –Q 0xe8 41.41.41.2
- Scenario 8: ping –Q 0xf0 41.41.41.2
- Scenario 9: ping –Q 0xf8 41.41.41.2
- Scenario 10: ping 41.41.41.2
- Scenario 11: ping –Q 0x55 41.41.41.2
- Scenario 12: ping –Q 0x144 41.41.41.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0x48.
- Scenario 8: ping requests are received with a TOS value of 0x50.
- Scenario 9: ping requests are received with a TOS value of 0x58.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/15**

Test Purpose : Verify DSCP swapping at ingress of MESCAL42.

Procedure : Ensure that BGP is activated between AS4 and AS3, AS4 and AS2 and AS4 and AS5. Execute **`qsdel`** in MESCAL21, MESCAL31 and MESCAL51. And execute **`qsa`** in MESCAL42.

Execute successively the following commands from a bash terminal of MESCAL31, check then what is received in eth0 of MESCAL21:

- Scenario 1: ping –Q 0xe8 2.2.2.1
- Scenario 2: ping –Q 0xf0 2.2.2.1
- Scenario 3: ping –Q 0xf8 2.2.2.1
- Scenario 4: ping 2.2.2.1
- Scenario 5: ping –Q 0x55 2.2.2.1
- Scenario 6: ping –Q 0x144 2.2.2.1

Execute successively the following commands from a bash terminal of MESCAL21, check then what is received in eth0 of MESCAL31:

- Scenario 7: ping –Q 0xc8 3.3.3.5
- Scenario 8: ping –Q 0xd0 3.3.3.5
- Scenario 9: ping –Q 0xd8 3.3.3.5
- Scenario 10: ping 2.2.2.1
- Scenario 11: ping –Q 0x55 3.3.3.5
- Scenario 12: ping –Q 0x144 3.3.3.5

Execute successively the following commands from a bash terminal of MESCAL51, and check what is received in eth0 of MESCAL31:

- Scenario 7: ping –Q 0x28 3.3.3.5
- Scenario 8: ping –Q 0x30 3.3.3.5
- Scenario 9: ping –Q 0x38 3.3.3.5

- Scenario 10: ping 3.3.3.5
- Scenario 11: ping –Q 0x55 3.3.3.5
- Scenario 12: ping –Q 0x144 3.3.3.5

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xc8.
- Scenario 2: ping requests are received with a TOS value of 0xd0.
- Scenario 3: ping requests are received with a TOS value of 0xc8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0xe8.
- Scenario 8: ping requests are received with a TOS value of 0xf0.
- Scenario 9: ping requests are received with a TOS value of 0xf8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.
- Scenario 13: ping requests are received with a TOS value of 0xe8.
- Scenario 14: ping requests are received with a TOS value of 0xf0.
- Scenario 15: ping requests are received with a TOS value of 0xf8.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: Unreachable destination.
- Scenario 18: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/16**

Test Purpose : Verify DSCP swapping at ingress of MESCAL43.

Procedure : Ensure that BGP is activated between AS4 and AS6. Execute **`qsdel`** in MESCAL61 and **`qsa`** in MESCAL43.

Execute successively the following commands from a bash terminal of MESCAL61, and check what is received in eth3 of MESCAL41:

- Scenario 1: ping –Q 0x48 41.41.41.5
- Scenario 2: ping –Q 0x50 41.41.41.5
- Scenario 3: ping –Q 0x58 41.41.41.5
- Scenario 4: ping 41.41.41.5
- Scenario 5: ping –Q 0x55 41.41.41.5
- Scenario 6: ping –Q 0x144 41.41.41.5

Execute successively the following commands from a bash terminal of MESCAL61, and check what is received in eth3 of MESCAL42:

- Scenario 7: ping –Q 0x48 42.42.42.1
- Scenario 8: ping –Q 0x50 42.42.42.1
- Scenario 9: ping –Q 0x58 42.42.42.1
- Scenario 10: ping 42.42.42.1
- Scenario 11: ping –Q 0x55 42.42.42.1
- Scenario 12: ping –Q 0x144 42.42.42.1

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x88.
- Scenario 2: ping requests are received with a TOS value of 0x90.
- Scenario 3: ping requests are received with a TOS value of 0x98.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.

- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0x88
- Scenario 8: ping requests are received with a TOS value of 0x98
- Scenario 9: ping requests are received with a TOS value of 0x98
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/17**

Test Purpose : Verify DSCP swapping at egress of MESCAL51.

Procedure : Ensure that BGP is activated between AS5 and AS2 and between AS4 and AS5. Execute **`qsdel`** in MESCAL21 and MESCAL42. And execute **`qsa`** in MESCAL51.

Execute successively the following commands from a bash terminal of MESCAL21, and check what is received in eth5 of MESCAL42:

- Scenario 1: ping –Q 0xe8 42.42.42.5
- Scenario 2: ping –Q 0xf0 42.42.42.5
- Scenario 3: ping –Q 0xf8 42.42.42.5
- Scenario 4: ping 42.42.42.5
- Scenario 5: ping –Q 0x55 42.42.42.5
- Scenario 6: ping –Q 0x144 42.42.42.5

Execute successively the following commands from a bash terminal of MESCAL42, and check what is received in eth5 of MESCAL21:

- Scenario 1: ping –Q 0x28 2.2.2.5
- Scenario 2: ping –Q 0x30 2.2.2.5
- Scenario 3: ping –Q 0x38 2.2.2.5
- Scenario 4: ping 2.2.2.5
- Scenario 5: ping –Q 0x55 2.2.2.5
- Scenario 6: ping –Q 0x144 2.2.2.5

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0x28.
- Scenario 2: ping requests are received with a TOS value of 0x30.
- Scenario 3: ping requests are received with a TOS value of 0x38.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0xe8.
- Scenario 8: ping requests are received with a TOS value of 0xf0.
- Scenario 9: ping requests are received with a TOS value of 0xf8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/18**

Test Purpose : Verify DSCP swapping at egress of MESCAL61.

Procedure : Ensure that BGP is activated between AS6 and AS7, AS6 and AS8 and between AS4 and AS6. Execute **`qsdel`** in MESCAL41, MESCAL43, MESCAL71 and MESCAL81. And execute **`qsa`** in MESCAL61.

Execute successively the following commands from a bash terminal of MESCAL41, and check what is received in eth2 of MESCAL81:

- Scenario 1: ping –Q 0x48 6.6.6.6.6
- Scenario 2: ping –Q 0x50 6.6.6.6.6
- Scenario 3: ping –Q 0x58 6.6.6.6.6
- Scenario 4: ping 6.6.6.6.6
- Scenario 5: ping –Q 0x55 6.6.6.6.6
- Scenario 6: ping –Q 0x144 6.6.6.6.6

Execute successively the following commands from a bash terminal of MESCAL43, and check what is received in eth2 of MESCAL81:

- Scenario 7: ping –Q 0x48 6.6.6.6.6
- Scenario 8: ping –Q 0x50 6.6.6.6.6
- Scenario 9: ping –Q 0x58 6.6.6.6.6
- Scenario 10: ping 6.6.6.6.6
- Scenario 11: ping –Q 0x55 6.6.6.6.6
- Scenario 12: ping –Q 0x144 6.6.6.6.6

Execute successively the following commands from a bash terminal of MESCAL71, and check what is received in eth2 of MESCAL81:

- Scenario 13: ping –Q 0xa8 6.6.6.6
- Scenario 14: ping –Q 0xa0 6.6.6.6
- Scenario 15: ping –Q 0xa8 6.6.6.6
- Scenario 16: ping 6.6.6.6
- Scenario 17: ping –Q 0x55 6.6.6.6
- Scenario 18: ping –Q 0x144 6.6.6.6

Execute successively the following commands from a bash terminal of MESCAL81, and check what is received in eth1 of MESCAL71:

- Scenario 19: ping –Q 0xe8 6.6.6.2
- Scenario 20: ping –Q 0xf0 6.6.6.2
- Scenario 21: ping –Q 0xf8 6.6.6.2
- Scenario 22: ping 6.6.6.2
- Scenario 23: ping –Q 0x55 6.6.6.2
- Scenario 24: ping –Q 0x144 6.6.6.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.
- Scenario 7: ping requests are received with a TOS value of 0xe8.
- Scenario 8: ping requests are received with a TOS value of 0xf0.
- Scenario 9: ping requests are received with a TOS value of 0xf8.
- Scenario 10: ping requests are received with a TOS value of 0x00.
- Scenario 11: Unreachable destination.
- Scenario 12: Unreachable destination.
- Scenario 13: ping requests are received with a TOS value of 0xe8.
- Scenario 14: ping requests are received with a TOS value of 0xf0.
- Scenario 15: ping requests are received with a TOS value of 0xf8.
- Scenario 16: ping requests are received with a TOS value of 0x00.
- Scenario 17: Unreachable destination.

- Scenario 18: Unreachable destination.
- Scenario 19: ping requests are received with a TOS value of 0xa8.
- Scenario 20: ping requests are received with a TOS value of 0xb0.
- Scenario 21: ping requests are received with a TOS value of 0xb8.
- Scenario 22: ping requests are received with a TOS value of 0x00.
- Scenario 23: Unreachable destination.
- Scenario 24: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/19**

Test Purpose : Verify DSCP swapping at egress of MESCAL71.

Procedure : Ensure that BGP is activated between AS6 and AS7. Execute `qsdel` in MESCAL61. And execute `qsa` in MESCAL71.

Execute successively the following commands from a bash terminal of MESCAL61:

- Scenario 1: ping –Q 0xa8 6.6.6.2
- Scenario 2: ping –Q 0xb0 6.6.6.2
- Scenario 3: ping –Q 0xb8 6.6.6.2
- Scenario 4: ping 6.6.6.2
- Scenario 5: ping –Q 0x55 6.6.6.2
- Scenario 6: ping –Q 0x144 6.6.6.2

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xa8.
- Scenario 2: ping requests are received with a TOS value of 0xb0.
- Scenario 3: ping requests are received with a TOS value of 0xb8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/20**

Test Purpose : Verify DSCP swapping at egress of MESCAL81.

Procedure : execute `qsdel` in MESCAL61. And execute `qsa` in MESCAL81.

Execute successively the following commands from a bash terminal of MESCAL61:

- Scenario 1: ping –Q 0xe8 6.6.6.6
- Scenario 2: ping –Q 0xf0 6.6.6.6
- Scenario 3: ping –Q 0xf8 6.6.6.6
- Scenario 4: ping 6.6.6.6
- Scenario 5: ping –Q 0x55 6.6.6.6
- Scenario 6: ping –Q 0x144 6.6.6.6

Expected result : Within the traffic analyzer, following results must be obtained:

- Scenario 1: ping requests are received with a TOS value of 0xe8.
- Scenario 2: ping requests are received with a TOS value of 0xf0.
- Scenario 3: ping requests are received with a TOS value of 0xf8.
- Scenario 4: ping requests are received with a TOS value of 0x00.
- Scenario 5: Unreachable destination.
- Scenario 6: Unreachable destination.

Test Reference : **TB_P1_FUNCT/DSSW/21**
Test Purpose : Verify the QoS configuration of the whole testbed.

Procedure :

Log to MESCAL11, then execute the **pfgping** script. The scenarios are linked together; the starting configuration of scenario N+1 is the one for Scenario N.

- Scenario 1: Execute **qsa** script in all router, or execute **initqos** from MESCAL_ADM.
- Scenario 2: execute **qsdel** in MESCAL31 and MESCAL21.
- Scenario 3: execute **qsi-eth1** in MESCAL31 and **qsi-eth6** in MESCAL21.
- Scenario 4: execute **qsa** in MESCAL31 and MESCAL21.

Expected result : the following results must be obtained as output of **pfgping** script.

- Scenario 1: All destinations are reachable in all meta-QoS-class planes.
- Scenario 2: All destinations are reachable in best effort plane. But only MESCAL31 and MESCAL21 interfaces are reachable in the rest of meta-QoS-class planes.
- Scenario 3: All destinations are reachable in best effort plane. But only MESCAL31 and MESCAL21 interfaces are reachable in the rest of meta-QoS-class planes.
- Scenario 4: All destinations are reached in all meta-QoS-class planes.

14.1.3 TB_P1_FUNCT/SHAP

Tests that are to be carried in this section assume that the following bandwidth configuration is deployed in testbed. Before beginning executing these tests, verify that TC scripts are conform to this configuration.

AS-AS	MC	BW (Mbit/s) →	BW (Mbit/s) ←
AS1-AS2	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS1-AS3	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS2-AS4	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS2-AS5	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS3-AS4	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS4-AS5	MC1	1	1
	MC2	1	1
	MC3	1	1

	MC4	Maximum is 7 Min is 4	Maximum is 7 Min is 4
AS4-AS6	MC1	3	3
	MC2	3	3
	MC3	3	3
	MC4	Max is 10 Min is 1	Max is 10 Min is 1
AS6-AS7	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2
AS6-AS8	MC1	1	1
	MC2	1	1
	MC3	1	1
	MC4	Maximum is 5 Min is 2	Maximum is 5 Min is 2

Table 81 - Bandwidth Threshold per meta-QoS-class

Test Reference : **TB_P1_FUNCT/SHAP/1**
Test Purpose : Verify shaping configuration in MESCAL11.
Procedure : Log to MESCAL11, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 2: Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 3: Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 4: Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 6: Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 7: Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 8: Stream with tos=0x38 and throughput = 5Mbit/s towards 31.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 10: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 11: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 12: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 14: Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 15: Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2

- Scenario 16: Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 18: Stream with tos=0x28 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 19: Stream with tos=0x30 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 20: Stream with tos=0x38 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 22: Stream with tos=0x28 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 23: Stream with tos=0x30 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 24: Stream with tos=0x38 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 26: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 27: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 28: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 30: Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 31: Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 32: Stream with tos=0x38 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 34: Stream with tos=0x28 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 35: Stream with tos=0x30 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 36: Stream with tos=0x38 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 38: Stream with tos=0x28 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 39: Stream with tos=0x30 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 40: Stream with tos=0x38 and throughput = 2Mbit/s towards 31.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result : The following results must be obtained:

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 5Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/2**

Test Purpose : Verify shaping configuration in MESCAL71.

Procedure : Log to MESCAL71, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0xe8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 3: Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 4: Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2

Execute **qsa** script from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0xe8 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0xf0 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0xf8 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0xe8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 11: Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 12: Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0xe8 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0xf0 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0xf8 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0xe8 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0xf0 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0xf8 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.

- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 1Mbit/s
- Scenario 11: Accepted traffic has a rate of 1Mbit/s
- Scenario 12: Accepted traffic has a rate of 1Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 5Mbit/s.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s.
- Scenario 19: Accepted traffic has a rate of 1Mbit/s.
- Scenario 20: Accepted traffic has a rate of 1Mbit/s.

Test Reference : **TB_P1_FUNCT/SHAP/3**

Test Purpose : Verify shaping configuration in MESCAL81.

Procedure : Log to MESCAL81, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 3: Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4: Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 11: Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 12: Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0x28 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0x30 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0x38 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0x28 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0x30 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0x38 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 1Mbit/s
- Scenario 11: Accepted traffic has a rate of 1Mbit/s
- Scenario 12: Accepted traffic has a rate of 1Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 5Mbit/s.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s.
- Scenario 19: Accepted traffic has a rate of 1Mbit/s.
- Scenario 20: Accepted traffic has a rate of 1Mbit/s.

Test Reference : **TB_P1_FUNCT/SHAP/4**

Test Purpose : Verify shaping configuration in MESCAL51.

Procedure : Log to MESCAL51, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 2: Stream with tos=0xa8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 3: Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 4: Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 6: Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2

Execute **qsa** script from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 10: Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 11: Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 12: Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 14: Stream with tos=0xa8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 15: Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 16: Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 18: Stream with tos=0xa8 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 19: Stream with tos=0xb0 and throughput = 1Mbit/s towards 21.0.0.2

- Scenario 20: Stream with tos=0xb8 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 22: Stream with tos=0xa8 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 23: Stream with tos=0xb0 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 24: Stream with tos=0xb8 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 26: Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 27: Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 28: Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 30: Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 31: Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0xa8 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0xb0 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0xb8 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 8Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0xa8 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 39: Stream with tos=0xb0 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0xb8 and throughput = 2Mbit/s towards 42.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 7Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/5**

Test Purpose : Verify shaping configuration in MESCAL43.

Procedure : Log to MESCAL43, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 3: Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 4: Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2

Execute **qsa** script from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 9Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 11: Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 12: Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 15Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 4Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 4Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 4Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.

- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 3Mbit/s
- Scenario 11: Accepted traffic has a rate of 3Mbit/s
- Scenario 12: Accepted traffic has a rate of 3Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 10Mbit/s
- Scenario 18: Accepted traffic has a rate of 3Mbit/s
- Scenario 19: Accepted traffic has a rate of 3Mbit/s
- Scenario 20: Accepted traffic has a rate of 3Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/6**

Test Purpose : Verify shaping configuration in MESCAL41.

Procedure : Log to MESCAL41, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 3: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 4: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 11: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2

- Scenario 12: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
- Scenario 22: Stream with tos=0x88 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 23: Stream with tos=0x90 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 24: Stream with tos=0x98 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 26: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 27: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 28: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 30: Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 31: Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 32: Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2

- Scenario 34: Stream with tos=0x88 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 35: Stream with tos=0x90 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 36: Stream with tos=0x98 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 10Mbit/s towards 61.0.0.2
- Scenario 38: Stream with tos=0x88 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 39: Stream with tos=0x90 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 40: Stream with tos=0x98 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 7Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s.
- Scenario 24: Accepted traffic has a rate of 1Mbit/s.

- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 3Mbit/s
- Scenario 31: Accepted traffic has a rate of 3Mbit/s
- Scenario 32: Accepted traffic has a rate of 3Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 7Mbit/s
- Scenario 38: Accepted traffic has a rate of 3Mbit/s
- Scenario 39: Accepted traffic has a rate of 3Mbit/s
- Scenario 40: Accepted traffic has a rate of 3Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/7**

Test Purpose : Verify shaping configuration in MESCAL42.

Procedure : Log to MESCAL42, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 3: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 4: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 8: Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2

- Scenario 11: Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 12: Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 22: Stream with tos=0x88 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 23: Stream with tos=0x90 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 24: Stream with tos=0x98 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
- Scenario 26: Stream with tos=0x88 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 27: Stream with tos=0x90 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 28: Stream with tos=0x98 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 29: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 31: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 21.0.0.2

- Scenario 32: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 34: Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 35: Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 36: Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 38: Stream with tos=0x88 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 39: Stream with tos=0x90 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 40: Stream with tos=0x98 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
- Scenario 42: Stream with tos=0x88 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 43: Stream with tos=0x90 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 44: Stream with tos=0x98 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 46: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 47: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 48: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 50: Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 51: Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 52: Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 51.0.0.2

- Scenario 54: Stream with tos=0x88 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 55: Stream with tos=0x90 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 56: Stream with tos=0x98 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
- Scenario 58: Stream with tos=0x88 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 59: Stream with tos=0x90 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 60: Stream with tos=0x98 and throughput = 2Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result : Following results must be obtained:

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s
- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.
- Scenario 23: No drop.
- Scenario 24: No drop.

- Scenario 25: Accepted traffic has a rate of 7Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop.
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s
- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s
- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop.
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s
- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.
- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s
- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/8**

Test Purpose : Verify shaping configuration in MESCAL21.

Procedure : Log to MESCAL21, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 2: Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 3: Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 4: Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 6: Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 10: Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 11: Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 12: Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 14: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 15: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 16: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 18: Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 19: Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 20: Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2

- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 22: Stream with tos=0x48 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 23: Stream with tos=0x50 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 24: Stream with tos=0x58 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 26: Stream with tos=0x48 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 27: Stream with tos=0x50 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 28: Stream with tos=0x58 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 31: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0x48 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 39: Stream with tos=0x50 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0x58 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 10Mbit/s towards 42.0.0.2
- Scenario 42: Stream with tos=0x48 and throughput = 2Mbit/s towards 42.0.0.2

- Scenario 43: Stream with tos=0x50 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 44: Stream with tos=0x58 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 46: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 47: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 48: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 50: Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 51: Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 52: Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 54: Stream with tos=0x48 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 55: Stream with tos=0x50 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 56: Stream with tos=0x58 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 10Mbit/s towards 51.0.0.2
- Scenario 58: Stream with tos=0x48 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 59: Stream with tos=0x50 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 60: Stream with tos=0x58 and throughput = 2Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.

- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s
- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.
- Scenario 23: No drop.
- Scenario 24: No drop.
- Scenario 25: Accepted traffic has a rate of 5Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s
- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s

- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s
- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.
- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s
- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/9**

Test Purpose : Verify shaping configuration in MESCAL31.

Procedure : Log to MESCAL31, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 2: Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 3: Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 4: Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 6: Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2

- Scenario 10: Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 11: Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 12: Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2

Execute **qsa** script from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 14: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 15: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 16: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 5Mbit/s towards 11.0.0.2
- Scenario 18: Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 19: Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 20: Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 22: Stream with tos=0x68 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 23: Stream with tos=0x70 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 24: Stream with tos=0x78 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 26: Stream with tos=0x68 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 27: Stream with tos=0x70 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 28: Stream with tos=0x78 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 42.0.0.2

- Scenario 31: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0x68 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 39: Stream with tos=0x70 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0x78 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 10Mbit/s towards 42.0.0.2
- Scenario 42: Stream with tos=0x68 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 43: Stream with tos=0x70 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 44: Stream with tos=0x78 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 46: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 47: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 48: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 50: Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 51: Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 52: Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2

- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 54: Stream with tos=0x68 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 55: Stream with tos=0x70 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 56: Stream with tos=0x78 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 58: Stream with tos=0x68 and throughput = 2Mbit/s towards 41.0.0.2
- Scenario 59: Stream with tos=0x70 and throughput = 2Mbit/s towards 41.0.0.2
- Scenario 60: Stream with tos=0x78 and throughput = 2Mbit/s towards 41.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s
- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.

- Scenario 23: No drop.
- Scenario 24: No drop.
- Scenario 25: Accepted traffic has a rate of 5Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s
- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s
- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s
- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.
- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s

- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/10**

Test Purpose : Verify shaping configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 2: Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 3: Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 4: Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 6: Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 7: Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 8: Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2

Execute **`qsa`** script from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 10: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 11: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 12: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 14: Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 15: Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 16: Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 18: Stream with tos=0xc8 and throughput = 1Mbit/s towards 71.0.0.2

- Scenario 19: Stream with tos=0xd0and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 20: Stream with tos=0xd8 and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
- Scenario 22: Stream with tos=0xc8 and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 23: Stream with tos=0xd0and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 24: Stream with tos=0xd8 and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 26: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 27: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 28: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 30: Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 31: Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 32: Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 34: Stream with tos=0xc8 and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 35: Stream with tos=0xd0and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 36: Stream with tos=0xd8 and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 38: Stream with tos=0xc8 and throughput = 2Mbit/s towards 81.0.0.2
- Scenario 39: Stream with tos=0xd0and throughput = 2Mbit/s towards 81.0.0.2
- Scenario 40: Stream with tos=0xd8 and throughput = 2Mbit/s towards 81.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.

- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 2Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/SHAP/11**

Test Purpose : Verify shaping configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 2: Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 3: Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 4: Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 6: Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 7: Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 8: Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 10: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 11: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 12: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 10Mbit/s towards 41.0.0.2
- Scenario 14: Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 15: Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 16: Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2

- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 18: Stream with tos=0xc8 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 19: Stream with tos=0xd0and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 20: Stream with tos=0xd8 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
- Scenario 22: Stream with tos=0xc8 and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 23: Stream with tos=0xd0and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 24: Stream with tos=0xd8 and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 26: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 27: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 28: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 30: Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 31: Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 32: Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 34: Stream with tos=0xc8 and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 35: Stream with tos=0xd0and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 36: Stream with tos=0xd8 and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 15Mbit/s towards 43.0.0.2
- Scenario 38: Stream with tos=0xc8 and throughput = 6Mbit/s towards 43.0.0.2

- Scenario 39: Stream with tos=0xd0and throughput = 6Mbit/s towards 43.0.0.2
- Scenario 40: Stream with tos=0xd8 and throughput = 6Mbit/s towards 43.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 3Mbit/s
- Scenario 15: Accepted traffic has a rate of 3Mbit/s
- Scenario 16: Accepted traffic has a rate of 3Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 10Mbit/s
- Scenario 22: Accepted traffic has a rate of 3Mbit/s
- Scenario 23: Accepted traffic has a rate of 3Mbit/s
- Scenario 24: Accepted traffic has a rate of 3Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop
- Scenario 30: Accepted traffic has a rate of 3Mbit/s
- Scenario 31: Accepted traffic has a rate of 3Mbit/s
- Scenario 32: Accepted traffic has a rate of 3Mbit/s

- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 10Mbit/s
- Scenario 38: Accepted traffic has a rate of 3Mbit/s
- Scenario 39: Accepted traffic has a rate of 3Mbit/s
- Scenario 40: Accepted traffic has a rate of 3Mbit/s

14.1.4 TB_P1_FUNCT/POLI

Test Reference : **TB_P1_FUNCT/POLI/1**

Test Purpose : Verify policing configuration in MESCAL11.

Procedure : Log to MESCAL11, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 2: Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 3: Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 4: Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 6: Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 7: Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 8: Stream with tos=0x38 and throughput = 5Mbit/s towards 31.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 10: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 11: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 12: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 14: Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2

- Scenario 15: Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 16: Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 18: Stream with tos=0x28 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 19: Stream with tos=0x30 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 20: Stream with tos=0x38 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 22: Stream with tos=0x28 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 23: Stream with tos=0x30 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 24: Stream with tos=0x38 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 26: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 27: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 28: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 30: Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 31: Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 32: Stream with tos=0x38 and throughput = 5Mbit/s towards 31.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 34: Stream with tos=0x28 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 35: Stream with tos=0x30 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 36: Stream with tos=0x38 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 38: Stream with tos=0x28 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 39: Stream with tos=0x30 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 40: Stream with tos=0x38 and throughput = 2Mbit/s towards 31.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result : The following results must be obtained:

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 5Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/2**

Test Purpose : Verify policing configuration in MESCAL71.

Procedure : Log to MESCAL71, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0xe8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 3: Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 4: Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0xe8 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0xf0 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0xf8 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0xe8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 11: Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 12: Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0xe8 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0xf0 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0xf8 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0xe8 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0xf0 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0xf8 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.

- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 1Mbit/s
- Scenario 11: Accepted traffic has a rate of 1Mbit/s
- Scenario 12: Accepted traffic has a rate of 1Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 5Mbit/s.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s.
- Scenario 19: Accepted traffic has a rate of 1Mbit/s.
- Scenario 20: Accepted traffic has a rate of 1Mbit/s.

Test Reference : **TB_P1_FUNCT/POLI/3**

Test Purpose : Verify policing configuration in MESCAL81.

Procedure : Log to MESCAL81, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 3: Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4: Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x28 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x30 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x38 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 11: Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 12: Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0x28 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0x30 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0x38 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0x28 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0x30 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0x38 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 1Mbit/s
- Scenario 11: Accepted traffic has a rate of 1Mbit/s
- Scenario 12: Accepted traffic has a rate of 1Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 5Mbit/s.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s.
- Scenario 19: Accepted traffic has a rate of 1Mbit/s.

- Scenario 20: Accepted traffic has a rate of 1Mbit/s.

Test Reference : **TB_P1_FUNCT/POLI/4**

Test Purpose : Verify policing configuration in MESCAL51.

Procedure : Log to MESCAL51, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 2: Stream with tos=0xa8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 3: Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 4: Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 6: Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 10: Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 11: Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 12: Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 14: Stream with tos=0xa8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 15: Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 16: Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 18: Stream with tos=0xa8 and throughput = 1Mbit/s towards 21.0.0.2

- Scenario 19: Stream with tos=0xb0 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 20: Stream with tos=0xb8 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 22: Stream with tos=0xa8 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 23: Stream with tos=0xb0 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 24: Stream with tos=0xb8 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 26: Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 27: Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 28: Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 30: Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 31: Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0xa8 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0xb0 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0xb8 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 8Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0xa8 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 39: Stream with tos=0xb0 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0xb8 and throughput = 2Mbit/s towards 42.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 7Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/5**

Test Purpose : Verify policing configuration in MESCAL43.

Procedure : Log to MESCAL43, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2

- Scenario 3: Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4: Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2

Execute **qsa** script in immediate neighbors and **qsdel** in local router from a bash prompt.

- Scenario 5: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 9Mbit/s towards 61.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 11: Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 12: Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 15Mbit/s towards 61.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 4Mbit/s towards 61.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 4Mbit/s towards 61.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 4Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.

- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: Accepted traffic has a rate of 3Mbit/s
- Scenario 11: Accepted traffic has a rate of 3Mbit/s
- Scenario 12: Accepted traffic has a rate of 3Mbit/s
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: Accepted traffic has a rate of 10Mbit/s
- Scenario 18: Accepted traffic has a rate of 3Mbit/s
- Scenario 19: Accepted traffic has a rate of 3Mbit/s
- Scenario 20: Accepted traffic has a rate of 3Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/6**

Test Purpose : Verify policing configuration in MESCAL41.

Procedure : Log to MESCAL41, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 3: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 4: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 8: Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2

Execute `qsa` script in immediate neighbors and `qsdel` in local router from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 11: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 12: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
- Scenario 22: Stream with tos=0x88 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 23: Stream with tos=0x90 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 24: Stream with tos=0x98 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 26: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 27: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 28: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 30: Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2

- Scenario 31: Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 32: Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 34: Stream with tos=0x88 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 35: Stream with tos=0x90 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 36: Stream with tos=0x98 and throughput = 1Mbit/s towards 61.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 10Mbit/s towards 61.0.0.2
- Scenario 38: Stream with tos=0x88 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 39: Stream with tos=0x90 and throughput = 2Mbit/s towards 61.0.0.2
- Scenario 40: Stream with tos=0x98 and throughput = 2Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.

- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 7Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s.
- Scenario 24: Accepted traffic has a rate of 1Mbit/s.
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.
- Scenario 30: Accepted traffic has a rate of 3Mbit/s
- Scenario 31: Accepted traffic has a rate of 3Mbit/s
- Scenario 32: Accepted traffic has a rate of 3Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 7Mbit/s
- Scenario 38: Accepted traffic has a rate of 3Mbit/s
- Scenario 39: Accepted traffic has a rate of 3Mbit/s
- Scenario 40: Accepted traffic has a rate of 3Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/7**

Test Purpose : Verify policing configuration in MESCAL42.

Procedure : Log to MESCAL42, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 2: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 3: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 4: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 6: Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 7: Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2

- Scenario 8: Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 10: Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 11: Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 12: Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 14: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 15: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 16: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 18: Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 19: Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 20: Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 22: Stream with tos=0x88 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 23: Stream with tos=0x90 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 24: Stream with tos=0x98 and throughput = 1Mbit/s towards 31.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
- Scenario 26: Stream with tos=0x88 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 27: Stream with tos=0x90 and throughput = 2Mbit/s towards 31.0.0.2
- Scenario 28: Stream with tos=0x98 and throughput = 2Mbit/s towards 31.0.0.2

- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 29: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 31: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 32: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 34: Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 35: Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 36: Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 38: Stream with tos=0x88 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 39: Stream with tos=0x90 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 40: Stream with tos=0x98 and throughput = 1Mbit/s towards 21.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
- Scenario 42: Stream with tos=0x88 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 43: Stream with tos=0x90 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 44: Stream with tos=0x98 and throughput = 2Mbit/s towards 21.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 46: Stream with tos=0x88 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 47: Stream with tos=0x90 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 48: Stream with tos=0x98 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 50: Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2

- Scenario 51: Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 52: Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 54: Stream with tos=0x88 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 55: Stream with tos=0x90 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 56: Stream with tos=0x98 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
- Scenario 58: Stream with tos=0x88 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 59: Stream with tos=0x90 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 60: Stream with tos=0x98 and throughput = 2Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result : Following results must be obtained:

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s

- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.
- Scenario 23: No drop.
- Scenario 24: No drop.
- Scenario 25: Accepted traffic has a rate of 7Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop.
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s
- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s
- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop.
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s
- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.

- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s
- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/8**

Test Purpose : Verify policing configuration in MESCAL21.

Procedure : Log to MESCAL21, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 2: Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 3: Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 4: Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 6: Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 10: Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 11: Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 12: Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 14: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 15: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 16: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 11.0.0.2

- Scenario 17: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 18: Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 19: Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 20: Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 22: Stream with tos=0x48 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 23: Stream with tos=0x50 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 24: Stream with tos=0x58 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 26: Stream with tos=0x48 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 27: Stream with tos=0x50 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 28: Stream with tos=0x58 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 31: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0x48 and throughput = 1Mbit/s towards 42.0.0.2

- Scenario 39: Stream with tos=0x50 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0x58 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 10Mbit/s towards 42.0.0.2
- Scenario 42: Stream with tos=0x48 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 43: Stream with tos=0x50 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 44: Stream with tos=0x58 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 46: Stream with tos=0x48 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 47: Stream with tos=0x50 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 48: Stream with tos=0x58 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 50: Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 51: Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 52: Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 54: Stream with tos=0x48 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 55: Stream with tos=0x50 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 56: Stream with tos=0x58 and throughput = 1Mbit/s towards 51.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 10Mbit/s towards 51.0.0.2
- Scenario 58: Stream with tos=0x48 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 59: Stream with tos=0x50 and throughput = 2Mbit/s towards 51.0.0.2
- Scenario 60: Stream with tos=0x58 and throughput = 2Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.
- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s
- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.
- Scenario 23: No drop.
- Scenario 24: No drop.
- Scenario 25: Accepted traffic has a rate of 5Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s

- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s
- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s
- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.
- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s
- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/9**

Test Purpose : Verify policing configuration in MESCAL31.

Procedure : Log to MESCAL31, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 2: Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 3: Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 4: Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2

- Scenario 6: Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 7: Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 8: Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 9: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 10: Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 11: Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 12: Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 13: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 14: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 15: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 16: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 5Mbit/s towards 11.0.0.2
- Scenario 18: Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 19: Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 20: Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 22: Stream with tos=0x68 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 23: Stream with tos=0x70 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 24: Stream with tos=0x78 and throughput = 1Mbit/s towards 11.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 26: Stream with tos=0x68 and throughput = 2Mbit/s towards 11.0.0.2

- Scenario 27: Stream with tos=0x70 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 28: Stream with tos=0x78 and throughput = 2Mbit/s towards 11.0.0.2
- Scenario 30: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 29: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 31: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 32: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 34: Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 35: Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 36: Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 38: Stream with tos=0x68 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 39: Stream with tos=0x70 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 40: Stream with tos=0x78 and throughput = 1Mbit/s towards 42.0.0.2
- Scenario 41: Stream with tos=0x00 and throughput = 10Mbit/s towards 42.0.0.2
- Scenario 42: Stream with tos=0x68 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 43: Stream with tos=0x70 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 44: Stream with tos=0x78 and throughput = 2Mbit/s towards 42.0.0.2
- Scenario 45: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 46: Stream with tos=0x68 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 47: Stream with tos=0x70 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 48: Stream with tos=0x78 and throughput = 0.5Mbit/s towards 41.0.0.2

- Scenario 49: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 50: Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 51: Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 52: Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 53: Stream with tos=0x00 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 54: Stream with tos=0x68 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 55: Stream with tos=0x70 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 56: Stream with tos=0x78 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 57: Stream with tos=0x00 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 58: Stream with tos=0x68 and throughput = 2Mbit/s towards 41.0.0.2
- Scenario 59: Stream with tos=0x70 and throughput = 2Mbit/s towards 41.0.0.2
- Scenario 60: Stream with tos=0x78 and throughput = 2Mbit/s towards 41.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: No drop.
- Scenario 15: No drop.

- Scenario 16: No drop.
- Scenario 17: No drop.
- Scenario 18: Accepted traffic has a rate of 1Mbit/s
- Scenario 19: Accepted traffic has a rate of 1Mbit/s
- Scenario 20: Accepted traffic has a rate of 1Mbit/s
- Scenario 21: No drop.
- Scenario 22: No drop.
- Scenario 23: No drop.
- Scenario 24: No drop.
- Scenario 25: Accepted traffic has a rate of 5Mbit/s
- Scenario 26: Accepted traffic has a rate of 1Mbit/s
- Scenario 27: Accepted traffic has a rate of 1Mbit/s
- Scenario 28: Accepted traffic has a rate of 1Mbit/s
- Scenario 29: No drop.
- Scenario 30: No drop.
- Scenario 31: No drop.
- Scenario 32: No drop.
- Scenario 33: No drop
- Scenario 34: Accepted traffic has a rate of 1Mbit/s
- Scenario 35: Accepted traffic has a rate of 1Mbit/s
- Scenario 36: Accepted traffic has a rate of 1Mbit/s
- Scenario 37: No drop.
- Scenario 38: No drop.
- Scenario 39: No drop.
- Scenario 40: No drop.
- Scenario 41: Accepted traffic has a rate of 7Mbit/s
- Scenario 42: Accepted traffic has a rate of 1Mbit/s
- Scenario 43: Accepted traffic has a rate of 1Mbit/s
- Scenario 44: Accepted traffic has a rate of 1Mbit/s
- Scenario 45: No drop.
- Scenario 46: No drop.
- Scenario 47: No drop.
- Scenario 48: No drop.
- Scenario 49: No drop
- Scenario 50: Accepted traffic has a rate of 1Mbit/s
- Scenario 51: Accepted traffic has a rate of 1Mbit/s
- Scenario 52: Accepted traffic has a rate of 1Mbit/s

- Scenario 53: No drop.
- Scenario 54: No drop.
- Scenario 55: No drop.
- Scenario 56: No drop.
- Scenario 57: Accepted traffic has a rate of 7Mbit/s
- Scenario 58: Accepted traffic has a rate of 1Mbit/s
- Scenario 59: Accepted traffic has a rate of 1Mbit/s
- Scenario 60: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/10**

Test Purpose : Verify policing configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 2: Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 3: Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 4: Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 6: Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 7: Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 8: Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 10: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 11: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 12: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 13: Stream with tos=0x00 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 14: Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2

- Scenario 15: Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 16: Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 18: Stream with tos=0xc8 and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 19: Stream with tos=0xd0and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 20: Stream with tos=0xd8 and throughput = 1Mbit/s towards 71.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
- Scenario 22: Stream with tos=0xc8 and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 23: Stream with tos=0xd0and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 24: Stream with tos=0xd8 and throughput = 2Mbit/s towards 71.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 26: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 27: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 28: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 30: Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 31: Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 32: Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 34: Stream with tos=0xc8 and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 35: Stream with tos=0xd0and throughput = 1Mbit/s towards 81.0.0.2
- Scenario 36: Stream with tos=0xd8 and throughput = 1Mbit/s towards 81.0.0.2

- Scenario 37: Stream with tos=0x00 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 38: Stream with tos=0xc8 and throughput = 2Mbit/s towards 81.0.0.2
- Scenario 39: Stream with tos=0xd0and throughput = 2Mbit/s towards 81.0.0.2
- Scenario 40: Stream with tos=0xd8 and throughput = 2Mbit/s towards 81.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 1Mbit/s
- Scenario 15: Accepted traffic has a rate of 1Mbit/s
- Scenario 16: Accepted traffic has a rate of 1Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 5Mbit/s
- Scenario 22: Accepted traffic has a rate of 1Mbit/s
- Scenario 23: Accepted traffic has a rate of 1Mbit/s
- Scenario 24: Accepted traffic has a rate of 1Mbit/s
- Scenario 25: No drop.
- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop.

- Scenario 30: Accepted traffic has a rate of 1Mbit/s
- Scenario 31: Accepted traffic has a rate of 1Mbit/s
- Scenario 32: Accepted traffic has a rate of 1Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 2Mbit/s
- Scenario 38: Accepted traffic has a rate of 1Mbit/s
- Scenario 39: Accepted traffic has a rate of 1Mbit/s
- Scenario 40: Accepted traffic has a rate of 1Mbit/s

Test Reference : **TB_P1_FUNCT/POLI/11**

Test Purpose : Verify policing configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1: Stream with tos=0x00 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 2: Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 3: Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 4: Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 5: Stream with tos=0x00 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 6: Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 7: Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 8: Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2

Execute **`qsa`** script in immediate neighbors and **`qsdel`** in local router from a bash prompt.

- Scenario 9: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 10: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 11: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 12: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 41.0.0.2

- Scenario 13: Stream with tos=0x00 and throughput = 10Mbit/s towards 41.0.0.2
- Scenario 14: Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 15: Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 16: Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 17: Stream with tos=0x00 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 18: Stream with tos=0xc8 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 19: Stream with tos=0xd0and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 20: Stream with tos=0xd8 and throughput = 1Mbit/s towards 41.0.0.2
- Scenario 21: Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
- Scenario 22: Stream with tos=0xc8 and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 23: Stream with tos=0xd0and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 24: Stream with tos=0xd8 and throughput = 4Mbit/s towards 41.0.0.2
- Scenario 25: Stream with tos=0x00 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 26: Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 27: Stream with tos=0xd0and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 28: Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 43.0.0.2
- Scenario 29: Stream with tos=0x00 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 30: Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 31: Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 32: Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2
- Scenario 33: Stream with tos=0x00 and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 34: Stream with tos=0xc8 and throughput = 1Mbit/s towards 43.0.0.2

- Scenario 35: Stream with tos=0xd0and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 36: Stream with tos=0xd8 and throughput = 1Mbit/s towards 43.0.0.2
- Scenario 37: Stream with tos=0x00 and throughput = 15Mbit/s towards 43.0.0.2
- Scenario 38: Stream with tos=0xc8 and throughput = 6Mbit/s towards 43.0.0.2
- Scenario 39: Stream with tos=0xd0and throughput = 6Mbit/s towards 43.0.0.2
- Scenario 40: Stream with tos=0xd8 and throughput = 6Mbit/s towards 43.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5: No drop.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8: No drop.
- Scenario 9: No drop.
- Scenario 10: No drop.
- Scenario 11: No drop.
- Scenario 12: No drop.
- Scenario 13: No drop.
- Scenario 14: Accepted traffic has a rate of 3Mbit/s
- Scenario 15: Accepted traffic has a rate of 3Mbit/s
- Scenario 16: Accepted traffic has a rate of 3Mbit/s
- Scenario 17: No drop.
- Scenario 18: No drop.
- Scenario 19: No drop.
- Scenario 20: No drop.
- Scenario 21: Accepted traffic has a rate of 10Mbit/s
- Scenario 22: Accepted traffic has a rate of 3Mbit/s
- Scenario 23: Accepted traffic has a rate of 3Mbit/s
- Scenario 24: Accepted traffic has a rate of 3Mbit/s
- Scenario 25: No drop.

- Scenario 26: No drop.
- Scenario 27: No drop.
- Scenario 28: No drop.
- Scenario 29: No drop
- Scenario 30: Accepted traffic has a rate of 3Mbit/s
- Scenario 31: Accepted traffic has a rate of 3Mbit/s
- Scenario 32: Accepted traffic has a rate of 3Mbit/s
- Scenario 33: No drop.
- Scenario 34: No drop.
- Scenario 35: No drop.
- Scenario 36: No drop.
- Scenario 37: Accepted traffic has a rate of 10Mbit/s
- Scenario 38: Accepted traffic has a rate of 3Mbit/s
- Scenario 39: Accepted traffic has a rate of 3Mbit/s
- Scenario 40: Accepted traffic has a rate of 3Mbit/s

14.1.5 TB_P1_FUNCT/BWMA

Test Reference : **TB_P1_FUNCT/BWMA/1**

Test Purpose : Verify bandwidth management configuration in MESCAL11.

Procedure : Log to MESCAL11, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 8Mbit/s towards 21.0.0.2
  - Stream with tos=0x28 and throughput = 8Mbit/s towards 21.0.0.2
  - Stream with tos=0x30 and throughput = 8Mbit/s towards 21.0.0.2
  - Stream with tos=0x38 and throughput = 8Mbit/s towards 21.0.0.2
- Scenario 2:
  - Stream with tos=0x00 and throughput = 8Mbit/s towards 31.0.0.2
  - Stream with tos=0x28 and throughput = 8Mbit/s towards 31.0.0.2
  - Stream with tos=0x30 and throughput = 8Mbit/s towards 31.0.0.2
  - Stream with tos=0x38 and throughput = 8Mbit/s towards 31.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 3:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2
  - Stream with tos=0x28 and throughput = 0.5Mbit/s towards 21.0.0.2

    - Stream with tos=0x30 and throughput = 0.5Mbit/s towards 21.0.0.2
    - Stream with tos=0x38 and throughput = 0.5Mbit/s towards 21.0.0.2
  - Scenario 4:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
  - Scenario 5:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x30 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x38 and throughput = 5Mbit/s towards 21.0.0.2
  - Scenario 6:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
    - Stream with tos=0x28 and throughput = 5Mbit/s towards 21.0.0.2
  - Scenario 7:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x28 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x30 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x38 and throughput = 0.5Mbit/s towards 31.0.0.2
  - Scenario 8:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x38 and throughput = 5Mbit/s towards 31.0.0.2
  - Scenario 9:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x30 and throughput = 5Mbit/s towards 31.0.0.2
  - Scenario 10:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 31.0.0.2
    - Stream with tos=0x28 and throughput = 5Mbit/s towards 31.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 5:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 6:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s
- Scenario 7: No drop.
- Scenario 8:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 9:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 10:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/2**

Test Purpose : Verify bandwidth management configuration in MESCAL71.

Procedure : Log to MESCAL71, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 8Mbit/s towards 61.0.0.2
  - Stream with tos=0xe8 and throughput = 8Mbit/s towards 61.0.0.2
  - Stream with tos=0xf0 and throughput = 8Mbit/s towards 61.0.0.2
  - Stream with tos=0xf8 and throughput = 8Mbit/s towards 61.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 2:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0xe8 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf0 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf8 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 3:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xe8 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf0 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xf8 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0xe8 and throughput = 1Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3:
  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 2Mbit/s
  - Stream with tos=0xe8 is received with a throughput of 1Mbit/s
  - Stream with tos=0xf0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xf8 is received with a throughput of 1Mbit/s
- Scenario 4:

  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 3Mbit/s
  - Stream with tos=0xf0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xf8 is received with a throughput of 1Mbit/s
- Scenario 5:
  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 4Mbit/s
  - Stream with tos=0xe8 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/3**

Test Purpose : Verify bandwidth management configuration in MESCAL81.

Procedure : Log to MESCAL81, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 2:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x28 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x30 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x38 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 3:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x30 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x38 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
  - Stream with tos=0x28 and throughput = 5Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 4:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x30 is received with a throughput of 1Mbit/s
    - Stream with tos=0x38 is received with a throughput of 1Mbit/s
- Scenario 5:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x28 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/4**

Test Purpose : Verify bandwidth management configuration in MESCAL51.

Procedure : Log to MESCAL51, Configure Smartbit to generate traffic as follows:

- Scenario 1:
    - Stream with tos=0x00 and throughput = 8Mbit/s towards 21.0.0.2
    - Stream with tos=0xa8 and throughput = 8Mbit/s towards 21.0.0.2
    - Stream with tos=0xb0 and throughput = 8Mbit/s towards 21.0.0.2
    - Stream with tos=0xb8 and throughput = 8Mbit/s towards 21.0.0.2
- Scenario 2:
    - Stream with tos=0x00 and throughput = 8Mbit/s towards 42.0.0.2
    - Stream with tos=0xa8 and throughput = 8Mbit/s towards 42.0.0.2
    - Stream with tos=0xb0 and throughput = 8Mbit/s towards 42.0.0.2
    - Stream with tos=0xb8 and throughput = 8Mbit/s towards 42.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 3:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2

  - Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 21.0.0.2
- Scenario 4:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xa8 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb0 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xb8 and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 6:
  - Stream with tos=0x00 and throughput = 5Mbit/s towards 21.0.0.2
  - Stream with tos=0xa8and throughput = 5Mbit/s towards 21.0.0.2
- Scenario 7:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0xa8 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0xb0 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0xb8 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 8:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 9:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xb0 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xb8 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 10:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0xa8 and throughput = 7Mbit/s towards 42.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4:
  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 2Mbit/s
  - Stream with tos=0xa8 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb8 is received with a throughput of 1Mbit/s
- Scenario 5:
  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 3Mbit/s
  - Stream with tos=0xb0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb8 is received with a throughput of 1Mbit/s
- Scenario 6:
  - Received traffic has a throughput of 5Mbit/s
  - Stream with tos=0x00 is received with a throughput of 4Mbit/s
  - Stream with tos=0xa8 is received with a throughput of 1Mbit/s
- Scenario 7: No drop.
- Scenario 8:
  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 4Mbit/s
  - Stream with tos=0xa8 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb8 is received with a throughput of 1Mbit/s
- Scenario 9:
  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 5Mbit/s
  - Stream with tos=0xb0 is received with a throughput of 1Mbit/s
  - Stream with tos=0xb8 is received with a throughput of 1Mbit/s
- Scenario 10:
  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 6Mbit/s
  - Stream with tos=0xa8 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/5**

Test Purpose : Verify bandwidth management configuration in MESCAL43.

Procedure : Log to MESCAL43, Configure Smartbit to generate traffic as follows:

- Scenario 1:
    - Stream with tos=0x00 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 2:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
    - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
    - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
    - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 3:
    - Stream with tos=0x00 and throughput = 10Mbit/s towards 61.0.0.2
    - Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 4:
    - Stream with tos=0x00 and throughput = 10Mbit/s towards 61.0.0.2
    - Stream with tos=0x90 and throughput = 5Mbit/s towards 61.0.0.2
    - Stream with tos=0x98 and throughput = 5Mbit/s towards 61.0.0.2
- Scenario 5:
    - Stream with tos=0x00 and throughput = 10Mbit/s towards 61.0.0.2
    - Stream with tos=0x88 and throughput = 5Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 1Mbit/s
    - Stream with tos=0x88 is received with a throughput of 3Mbit/s

        - Stream with tos=0x90 is received with a throughput of 3Mbit/s
        - Stream with tos=0x98 is received with a throughput of 3Mbit/s
    - Scenario 4:
        - Received traffic has a throughput of 10Mbit/s
        - Stream with tos=0x00 is received with a throughput of 4Mbit/s
        - Stream with tos=0x90 is received with a throughput of 3Mbit/s
        - Stream with tos=0x98 is received with a throughput of 3Mbit/s
    - Scenario 5:
        - Received traffic has a throughput of 10Mbit/s
        - Stream with tos=0x00 is received with a throughput of 7Mbit/s
        - Stream with tos=0x88 is received with a throughput of 3Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/6**

Test Purpose : Verify bandwidth management configuration in MESCAL41.

Procedure : Log to MESCAL41, Configure Smartbit to generate traffic as follows:

- Scenario 1:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 2:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 61.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 3:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
- Scenario 4:
    - Stream with tos=0x00 and throughput = 10Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2

  - Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
  - Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 10Mbit/s towards 31.0.0.2
  - Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
  - Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 6:
  - Stream with tos=0x00 and throughput = 10Mbit/s towards 31.0.0.2
  - Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
- Scenario 7:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 61.0.0.2
  - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 61.0.0.2
- Scenario 8:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 61.0.0.2
  - Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2
  - Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
  - Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 9:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 61.0.0.2
  - Stream with tos=0x90 and throughput = 7Mbit/s towards 61.0.0.2
  - Stream with tos=0x98 and throughput = 7Mbit/s towards 61.0.0.2
- Scenario 10:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 61.0.0.2
  - Stream with tos=0x88 and throughput = 7Mbit/s towards 61.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3:

  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 4Mbit/s
  - Stream with tos=0x88 is received with a throughput of 1Mbit/s
  - Stream with tos=0x90 is received with a throughput of 1Mbit/s
  - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 4:
  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 5Mbit/s
  - Stream with tos=0x90 is received with a throughput of 1Mbit/s
  - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 5:
  - Received traffic has a throughput of 7Mbit/s
  - Stream with tos=0x00 is received with a throughput of 6Mbit/s
  - Stream with tos=0x88 is received with a throughput of 1Mbit/s.
- Scenario 6: No drop.
- Scenario 7: No drop.
- Scenario 8:
  - Received traffic has a throughput of 10Mbit/s
  - Stream with tos=0x00 is received with a throughput of 1Mbit/s
  - Stream with tos=0x88 is received with a throughput of 3Mbit/s
  - Stream with tos=0x90 is received with a throughput of 3Mbit/s
  - Stream with tos=0x98 is received with a throughput of 3Mbit/s
- Scenario 9:
  - Received traffic has a throughput of 10Mbit/s
  - Stream with tos=0x00 is received with a throughput of 4Mbit/s
  - Stream with tos=0x90 is received with a throughput of 3Mbit/s
  - Stream with tos=0x98 is received with a throughput of 3Mbit/s
- Scenario 10:
  - Received traffic has a throughput of 10Mbit/s
  - Stream with tos=0x00 is received with a throughput of 7Mbit/s
  - Stream with tos=0x88 is received with a throughput of 3Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/7**

Test Purpose : Verify bandwidth management configuration in MESCAL42.

Procedure : Log to MESCAL42, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2

    - Stream with tos=0x88 and throughput = 9Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 9Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 9Mbit/s towards 31.0.0.2
  - Scenario 2:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x88 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x90 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x98 and throughput = 9Mbit/s towards 21.0.0.2
  - Scenario 3:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x88 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x90 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x98 and throughput = 9Mbit/s towards 51.0.0.2

Execute `qsa` script from a bash prompt.

  - Scenario 4:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 31.0.0.2
  - Scenario 5:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
  - Scenario 6:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 31.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 31.0.0.2
  - Scenario 7:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 31.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 31.0.0.2
  - Scenario 8:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 21.0.0.2

    - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 21.0.0.2
    - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 21.0.0.2
    - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 21.0.0.2
  - Scenario 9:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
  - Scenario 10:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 21.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 21.0.0.2
  - Scenario 11:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 21.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 21.0.0.2
  - Scenario 12:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x88 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x90 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x98 and throughput = 0.5Mbit/s towards 51.0.0.2
  - Scenario 13:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2
  - Scenario 14:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x90 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x98 and throughput = 7Mbit/s towards 51.0.0.2
  - Scenario 15:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x88 and throughput = 7Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result : Following results must be obtained:

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 6:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 7:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s
- Scenario 8: No drop.
- Scenario 9:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 10:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 11:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s
- Scenario 12: No drop.

- Scenario 13:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 14:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x90 is received with a throughput of 1Mbit/s
    - Stream with tos=0x98 is received with a throughput of 1Mbit/s
- Scenario 15:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s
    - Stream with tos=0x88 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/8**

Test Purpose : Verify bandwidth management configuration in MESCAL21.

Procedure : Log to MESCAL21, Configure Smartbit to generate traffic as follows:

- Scenario 1:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 11.0.0.2
    - Stream with tos=0x48 and throughput = 9Mbit/s towards 11.0.0.2
    - Stream with tos=0x50 and throughput = 9Mbit/s towards 11.0.0.2
    - Stream with tos=0x58 and throughput = 9Mbit/s towards 11.0.0.2
- Scenario 2:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x48 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x50 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x58 and throughput = 9Mbit/s towards 42.0.0.2
- Scenario 3:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x48 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x50 and throughput = 9Mbit/s towards 51.0.0.2
    - Stream with tos=0x58 and throughput = 9Mbit/s towards 51.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 4:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2

  - Stream with tos=0x48 and throughput = 0.5Mbit/s towards 11.0.0.2
  - Stream with tos=0x50 and throughput = 0.5Mbit/s towards 11.0.0.2
  - Stream with tos=0x58 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 6:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x50 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x58 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 7:
  - Stream with tos=0x00 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x48 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 8:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0x48 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0x50 and throughput = 0.5Mbit/s towards 42.0.0.2
  - Stream with tos=0x58 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 9:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 10:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x50 and throughput = 7Mbit/s towards 42.0.0.2
  - Stream with tos=0x58 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 11:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x48 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 12:

    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x48 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x50 and throughput = 0.5Mbit/s towards 51.0.0.2
    - Stream with tos=0x58 and throughput = 0.5Mbit/s towards 51.0.0.2
- Scenario 13:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 14:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x50 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x58 and throughput = 7Mbit/s towards 51.0.0.2
- Scenario 15:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 51.0.0.2
    - Stream with tos=0x48 and throughput = 7Mbit/s towards 51.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 6:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 7:
    - Received traffic has a throughput of 5Mbit/s

    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
  - Scenario 8: No drop.
  - Scenario 9:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
  - Scenario 10:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
  - Scenario 11:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
  - Scenario 12: No drop.
  - Scenario 13:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
  - Scenario 14:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
  - Scenario 15:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/9**

Test Purpose : Verify bandwidth management configuration in MESCAL31.

Procedure : Log to MESCAL31, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x68 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x70 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x78 and throughput = 9Mbit/s towards 11.0.0.2
- Scenario 2:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x68 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x70 and throughput = 9Mbit/s towards 42.0.0.2
  - Stream with tos=0x78 and throughput = 9Mbit/s towards 42.0.0.2
- Scenario 3:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 41.0.0.2
  - Stream with tos=0x68 and throughput = 9Mbit/s towards 41.0.0.2
  - Stream with tos=0x70 and throughput = 9Mbit/s towards 41.0.0.2
  - Stream with tos=0x78 and throughput = 9Mbit/s towards 41.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 4:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 11.0.0.2
  - Stream with tos=0x68 and throughput = 0.5Mbit/s towards 11.0.0.2
  - Stream with tos=0x70 and throughput = 0.5Mbit/s towards 11.0.0.2
  - Stream with tos=0x78 and throughput = 0.5Mbit/s towards 11.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 6:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x70 and throughput = 7Mbit/s towards 11.0.0.2
  - Stream with tos=0x78 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 7:
  - Stream with tos=0x00 and throughput = 9Mbit/s towards 11.0.0.2
  - Stream with tos=0x68 and throughput = 7Mbit/s towards 11.0.0.2
- Scenario 8:

    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 42.0.0.2
    - Stream with tos=0x68 and throughput = 0.5Mbit/s towards 42.0.0.2
    - Stream with tos=0x70 and throughput = 0.5Mbit/s towards 42.0.0.2
    - Stream with tos=0x78 and throughput = 0.5Mbit/s towards 42.0.0.2
- Scenario 9:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
    - Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
    - Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 10:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x70 and throughput = 7Mbit/s towards 42.0.0.2
    - Stream with tos=0x78 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 11:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 42.0.0.2
    - Stream with tos=0x68 and throughput = 7Mbit/s towards 42.0.0.2
- Scenario 12:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
    - Stream with tos=0x68 and throughput = 0.5Mbit/s towards 41.0.0.2
    - Stream with tos=0x70 and throughput = 0.5Mbit/s towards 41.0.0.2
    - Stream with tos=0x78 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 13:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 41.0.0.2
    - Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2
    - Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
    - Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 14:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 41.0.0.2
    - Stream with tos=0x70 and throughput = 7Mbit/s towards 41.0.0.2
    - Stream with tos=0x78 and throughput = 7Mbit/s towards 41.0.0.2
- Scenario 15:
    - Stream with tos=0x00 and throughput = 9Mbit/s towards 41.0.0.2

    - Stream with tos=0x68 and throughput = 7Mbit/s towards 41.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4: No drop.
- Scenario 5:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 6:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 7:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
- Scenario 8: No drop.
- Scenario 9:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 10:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 11:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s

    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
- Scenario 12: No drop.
- Scenario 13:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 14:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 5Mbit/s
    - Stream with tos=0x50 is received with a throughput of 1Mbit/s
    - Stream with tos=0x58 is received with a throughput of 1Mbit/s
- Scenario 15:
    - Received traffic has a throughput of 7Mbit/s
    - Stream with tos=0x00 is received with a throughput of 6Mbit/s
    - Stream with tos=0x48 is received with a throughput of 1Mbit/s

---

Test Reference : **TB_P1_FUNCT/BWMA/10**

Test Purpose : Verify bandwidth management configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xc8 and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xd0and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xd8 and throughput = 7Mbit/s towards 71.0.0.2
- Scenario 2:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xc8 and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xd0and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xd8 and throughput = 7Mbit/s towards 81.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 3:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 71.0.0.2
    - Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 71.0.0.2

    - Stream with tos=0xd0and throughput = 0.5Mbit/s towards 71.0.0.2
    - Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 71.0.0.2
- Scenario 4:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 5:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 71.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 6:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 71.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 71.0.0.2
- Scenario 7:
    - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 81.0.0.2
    - Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 81.0.0.2
    - Stream with tos=0xd0and throughput = 0.5Mbit/s towards 81.0.0.2
    - Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 81.0.0.2
- Scenario 8:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 9:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 81.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 81.0.0.2
- Scenario 10:
    - Stream with tos=0x00 and throughput = 7Mbit/s towards 81.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 81.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

- Scenario 1: No drop.
- Scenario 2: No drop.
- Scenario 3: No drop.
- Scenario 4:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 1Mbit/s
- Scenario 5:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 1Mbit/s
- Scenario 6:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 1Mbit/s
- Scenario 7: No drop.
- Scenario 8:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 2Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 1Mbit/s
- Scenario 9:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 1Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 1Mbit/s
- Scenario 10:
    - Received traffic has a throughput of 5Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 1Mbit/s.

Test Reference : **TB_P1_FUNCT/BWMA/11**

Test Purpose : Verify bandwidth management configuration in MESCAL61.

Procedure : Log to MESCAL61, Configure Smartbit to generate traffic as follows:

- Scenario 1:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
  - Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 2:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 43.0.0.2
  - Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
  - Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
  - Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2

Execute `qsa` script from a bash prompt.

- Scenario 3:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 41.0.0.2
  - Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd0and throughput = 0.5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 41.0.0.2
- Scenario 4:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
  - Stream with tos=0xc8 and throughput = 5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 5:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
  - Stream with tos=0xd0and throughput = 5Mbit/s towards 41.0.0.2
  - Stream with tos=0xd8 and throughput = 5Mbit/s towards 41.0.0.2
- Scenario 6:
  - Stream with tos=0x00 and throughput = 12Mbit/s towards 41.0.0.2
  - Stream with tos=0xc8 and throughput = 8Mbit/s towards 41.0.0.2
- Scenario 7:
  - Stream with tos=0x00 and throughput = 0.5Mbit/s towards 43.0.0.2

    - Stream with tos=0xc8 and throughput = 0.5Mbit/s towards 43.0.0.2
    - Stream with tos=0xd0and throughput = 0.5Mbit/s towards 43.0.0.2
    - Stream with tos=0xd8 and throughput = 0.5Mbit/s towards 43.0.0.2
  - Scenario 8:
    - Stream with tos=0x00 and throughput = 12Mbit/s towards 43.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2
  - Scenario 9:
    - Stream with tos=0x00 and throughput = 12Mbit/s towards 43.0.0.2
    - Stream with tos=0xd0and throughput = 5Mbit/s towards 43.0.0.2
    - Stream with tos=0xd8 and throughput = 5Mbit/s towards 43.0.0.2
  - Scenario 10:
    - Stream with tos=0x00 and throughput = 12Mbit/s towards 43.0.0.2
    - Stream with tos=0xc8 and throughput = 5Mbit/s towards 43.0.0.2

Check with Smartbit statistics if there is traffic drops.

Expected result :

  - Scenario 1: No drop.
  - Scenario 2: No drop.
  - Scenario 3: No drop.
  - Scenario 4:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 1Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 3Mbit/s
  - Scenario 5:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 3Mbit/s
  - Scenario 6:
    - Received traffic has a throughput of 10Mbit/s

    - Stream with tos=0x00 is received with a throughput of 7Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 3Mbit/s
  - Scenario 7: No drop.
  - Scenario 8:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 1Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 3Mbit/s
  - Scenario 9:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 4Mbit/s
    - Stream with tos=0xd0 is received with a throughput of 3Mbit/s
    - Stream with tos=0xd8 is received with a throughput of 3Mbit/s
  - Scenario 10:
    - Received traffic has a throughput of 10Mbit/s
    - Stream with tos=0x00 is received with a throughput of 7Mbit/s
    - Stream with tos=0xc8 is received with a throughput of 3Mbit/s

14.2 Phase 2

The "null" value means no value.

14.2.1 TB_P2_FUNCT/CMES

Test Reference : **TB_P2_FUNCT/CMES/1**
Test Purpose : Verify the capability length.
Procedure : Configure MESCAL11 to send the following QoS service capability to MESCAL21

- Group 1
- Group 2

Expected result : The capability length must be set to 2.

Test Reference : **TB_P2_FUNCT/CMES/2**
Test Purpose : Verify the QoS service capability field length.
Procedure : Configure MESCAL11 to send the following QoS service capability to MESCAL21

- Group 1
- Group 2

Expected result : The length of the QoS service capability must be 2 bytes.

Test Reference : **TB_P2_FUNCT/CMES/3**
Test Purpose : Verify that Group 1 QoS service capability is supported.

Procedure : Configure MESCAL11 to send a QoS service capability G1 to MESCAL21.

Expected result : QoS service capability field is two bytes. The first byte must be set to 0xFF.

Test Reference : **TB_P2_FUNCT/CMES/4**
Test Purpose : Verify that Group 2 QoS service capability is supported.
Procedure : Configure MESCAL11 to send a QoS service capability G2 to MESCAL21.

Expected result : QoS service capability field is two bytes. The second byte is set to 0xFF.

Test Reference : **TB_P2_FUNCT/CMES/5**
Test Purpose : Validate the conformance of QoS information length.
Procedure : Configure MESCAL11 to send BGP UPDATE message with QoS information codes to MESCAL21.

The following QoS information is to be configured. The value of the QoS Information length is to be checked with a traffic analyzer:

- Scenario 1: no QoS information
- Scenario 2: reserved rate 52
- Scenario 3: loss rate 60
- Scenario 4: reserved rate 52 and loss rate 60
- Scenario 5: minimum one way delay 50 and reserved rate 52
- Scenario 6: minimum one way delay 50 and loss rate 60
- Scenario 7: minimum one way delay 50, loss rate 60 and reserved rate 52
- Scenario 8: minimum one way delay 50, maximum one way delay 150 and loss rate 60
- Scenario 9: minimum one way delay 50, average one way delay 100 and loss rate 60
- Scenario 10: minimum one way delay 50, average one way delay 100 and reserved rate 52
- Scenario 11: minimum one way delay 50, average one way delay 100, maximum one way delay 150 and loss rate 60
- Scenario 12: minimum one way delay 50, average one way delay 100, loss rate 60 and reserved rate 52
- Scenario 13: minimum one way delay 50, maximum one way delay 150, average one way delay 100, loss rate 60 and reserved rate 52
- Scenario 14: minimum one way delay 50, maximum one way delay 150, average one way delay 100, loss rate 60 and jitter 5

Expected result : QoS information length is one byte field. The value checked within a traffic analyzer must be set as follows:

- Scenario 1: 0
- Scenario 2: 1
- Scenario 3: 1
- Scenario 4: 2
- Scenario 5: 2
- Scenario 6: 2
- Scenario 7: 3
- Scenario 8: 3
- Scenario 9: 3
- Scenario 10: 3
- Scenario 11: 4
- Scenario 12: 4
- Scenario 13: 5

- Scenario 14: 5

Test Reference : **TB_P2_FUNCT/CMES/6**

Test Purpose : Verify that "Packet Rate QoS Code" and its associated Sub-codes are supported.

Procedure : Configure MESCAL11 to send BGP UPDATE message with following QoS information to MESCAL21:

- Scenario 1: loss rate 30
- Scenario 2: reserved rate 25
- Scenario 3: available rate 25
- Scenario 4: loss rate 45 and reserved rate 454
- Scenario 5: loss rate 55 and available rate 100
- Scenario 6: reserved rate 52 and available rate 100
- Scenario 7: loss rate 60, available rate 100 and reserved rate 52

A traffic analyzer must be launched in the egress of MESCAL11 in order to examine the values of QoS information Codes and QoS information Sub-Codes.

Expected result : In all Scenario listed above, QoS information field is coded in 4 bits and QoS information Sub Code is also coded in 4 bits. The values of these fields must be as follows depending on the scenarios listed above:

- Scenario 1: QoS information Code = 1, QoS information Sub Code = 3
- Scenario 2: QoS information Code = 1, QoS information Sub Code = 1
- Scenario 3: QoS information Code = 1, QoS information Sub Code = 2
- Scenario 4:
  - QoS information Code = 1, QoS information Sub Code = 3
  - QoS information Code = 1, QoS information Sub Code = 1
- Scenario 5:
  - QoS information Code = 1, QoS information Sub Code = 3
  - QoS information Code = 1, QoS information Sub Code = 2
- Scenario 6:
  - QoS information Code = 1, QoS information Sub Code = 1
  - QoS information Code = 1, QoS information Sub Code = 2
- Scenario 7:
  - QoS information Code = 1, QoS information Sub Code = 1
  - QoS information Code = 1, QoS information Sub Code = 2
  - QoS information Code = 1, QoS information Sub Code = 3

Test Reference : **TB_P2_FUNCT/CMES/7**

Test Purpose : Validate that "One Way Delay QoS Code" and its associated Sub-codes are supported.

Procedure : Configure MESCAL11 to send BGP UPDATE message with following QoS information to MESCAL21.

- Scenario 1: minimum one-way delay 50
- Scenario 2: maximum one-way delay 150
- Scenario 3: average one-way delay 100
- Scenario 4: minimum one-way delay 50 and maximum one-way delay 150
- Scenario 5: minimum one-way delay 50 and average one-way delay 100
- Scenario 6: maximum one-way delay 150 and average one-way delay 100
- Scenario 7: minimum one-way delay 50, maximum one-way delay 150 and average one-way delay 100.

A traffic analyzer must be launched in the egress of MESCAL11 in order to examine the values that are set in QoS information Code and QoS information Sub-Code.

Expected result : In all Scenario listed above, QoS information field is coded in 4 bits and QoS information Sub Code is also coded in 4 bits. The values of these fields must be as follows depending on the scenarios listed above:

- Scenario 1: QoS information Code = 2, QoS information Sub Code = 4
- Scenario 2: QoS information Code = 2, QoS information Sub Code = 5
- Scenario 3: QoS information Code = 2, QoS information Sub Code = 6
- Scenario 4:
  - QoS information Code = 2, QoS information Sub Code = 4
  - QoS information Code = 2, QoS information Sub Code = 5
- Scenario 5:
  - QoS information Code = 2, QoS information Sub Code = 4
  - QoS information Code = 2, QoS information Sub Code = 6
- Scenario 6:
  - QoS information Code = 2, QoS information Sub Code = 5
  - QoS information Code = 2, QoS information Sub Code = 6
- Scenario 7:
  - QoS information Code = 2, QoS information Sub Code = 4
  - QoS information Code = 2, QoS information Sub Code = 5
  - QoS information Code = 2, QoS information Sub Code = 6

Test Reference : **TB_P2_FUNCT/CMES/8**
Test Purpose : Validate that "Inter-Packet Delay Variation QoS Code" and its associated Sub-codes are supported.

Procedure : Configure MESCAL11 to send BGP UPDATE message with following QoS information to MESCAL21: jitter 5

Expected result : QoS information field is coded in 4 bits and QoS information Sub Code is also coded in 4 bits. The value of these two fields must be:

- QoS information Code = 3, QoS information Sub Code = 0

Test Reference : **TB_P2_FUNCT/CMES/9**
Test Purpose : Validate the QoS information value.
Procedure : Configure MESCAL11 to send BGP UPDATE message with following QoS information to MESCAL21.

- Scenario 1: minimum one-way delay 50, maximum one-way delay 150 and average one-way delay 100
- Scenario 2: reserved rate 100kbps, available rate 100bps
- Scenario 3: reserved rate 1Mbps
- Scenario 4: jitter 5
- Scenario 5: minimum one-way delay -50, maximum one-way delay -150 and average one-way delay -100
- Scenario 6: reserved rate -100kbps, available rate -100bps

A traffic analyzer must be launched in the egress of MESCAL11 in order to examine the values of QoS information Value field of QoS_NLRI attribute.

Expected result : QoS Information value must be coded in 2 bytes. The value of this field must be as follows (depending on the scenarios listed above)

- Scenario 1:

  - QoS Information Value = 50
  - QoS Information Value = 150
  - QoS Information Value = 100
- Scenario 2:
  - QoS Information Value = 100
  - QoS Information Value = 100
- Scenario 3: QoS Information Value = 1
- Scenario 4: jitter 5
- Scenario 5: Error messages must be returned.
- Scenario 6: Error messages must be returned.

Test Reference : **TB_P2_FUNCT/CMES/10**

Test Purpose : Verify that QoS class identifier can be set to a value that is between 0 and 63.

Procedure : Configure MESCAL11 to send BGP UPDATE messages to MESCAL21 in the following meta-QoS-class plans:

- Plan 1: 0
- Plan 2: -1
- Plan 3: 25
- Plan 4: 63
- Plan 5: 70
- Plan 6: 55
- Plan 7: 33

Traffic must be sniffed in the egress of MESCAL11 or in ingress of MESCAL21 to verify the value of QoS class identifier in BGP UPDATE messages.

Expected result : QoS class identifier is coded in one byte and must be set to the value that is indicated bellow or an error message must be returned to the administrator:

- Plan 1: 0
- Plan 2: An error message must be returned to the administrator
- Plan 3: 25
- Plan 4: 63
- Plan 5: An error message must be returned to the administrator
- Plan 6: 55
- Plan 7: 33

Test Reference : **TB_P2_FUNCT/CMES/11**

Test Purpose : Validate the QoS Origin field.

Procedure : Clear the network prefixes announced by MESCAL11. And then make the following operations:

- Configure a route static towards 193.251.128.0/19 via eth1
- Add the following command lines in MESCAL11 BGP configuration "`network 212.167.0.0/21`" and "`network 62.42.0.0/16`"
- Add the following commands "`redistribute connected`" and "`redistribute static`"
- Under BGP router configuration, add the following line

```
Neighbour 1.1.1.6 route-map SetOrigin out
```

- Define this prefix-list:

```
ip prefix-list test11 permit 62.42.0.0/16
ip prefix-list test11 deny any
```

- Define this route-map:

```
route-map SetOrigin permit 10
 match ip address prefix-list test11
 set origin egp
route-map SetOrigin permit 20
```

Note that a BGP session must be activated between MESCAL11 and MESCAL12.

Expected result : Execute this CLI command in MESCAL11: "`sh ip bgp`". The value of the origin value must be positioned as follows:

- For `212.167.0.0/21` entry origin must be IGP
- For `62.42.0.0/16` entry origin value must be IGP
- For `1.1.1.0/30` entry origin value must be incomplete
- For `193.251.128.0/19` origin value must be incomplete

Execute this CLI command in MESCAL21: "`sh ip bgp`". The value of the origin value must be positioned as follows:

- For `212.167.0.0/21` entry origin must be IGP
- For `62.42.0.0/16` entry origin value must be EGP
- For `193.251.128.0/19` origin value must be incomplete

Test Reference : **TB_P2_FUNCT/CMES/12**

Test Purpose : Check the validity of Address Family Identifier (AFI). For information, the value of this field could be as follows:

Number	Description
0	Reserved
1	IP (IP version 4)
2	IP6 (IP version 6)
3	NSAP
4	HDLC (8-bit multidrop)
5	BBN 1822
6	802
7	E.163
8	E.164 (SMDS, Frame Relay, ATM)
9	F.69 (Telex)
10	X.121 (X.25, Frame Relay)
11	IPX
12	Appletalk
13	Decnet IV
14	Banyan Vines
65535	Reserved

Procedure : Configuration of MESCAL11 is the same as for the previous test. Launch a traffic analyzer in MESCAL11. Execute the following command "`clear ip bgp *`".

Expected result : Within the traffic analyzer, Check the value of AFI field of QoS_NLRI attribute. The value of AFI must be: 1.

Test Reference : **TB_P2_FUNCT/CMES/13**

Test Purpose : Check the validity of Subsequent Address Family Identifier (SAFI). For information, the value of this field could be as follows:

- **`1: Network Layer Reachability Information used for unicast forwarding`**
- **`2: Network Layer Reachability Information used for multicast forwarding`**
- **`3: Network Layer Reachability Information used for both unicast and multicast forwarding`**

Procedure : Configuration of MESCAL11 is the same as for the previous test. Launch a traffic analyzer in MESCAL11. Execute the following command "**`clear ip bgp *`**".

Expected result : Within the traffic analyzer, Check the value of SAFI field of QoS_NLRI attribute. The value of this field must be: 1. Multicast is out of scope.

Test Reference : **TB_P2_FUNCT/CMES/14**

Test Purpose : Check the validity of Network Address of Next Hop.

Procedure : Configuration of MESCAL11 is the same as for the previous test. Log to MESCAL21 and execute the following command line "**`sh ip bgp`**".

Configure MESCAL31 to send the following network "**`216.191.64.0/20`**". Log to MESCAL41 and execute the following command line "**`sh ip bgp`**". Also log to MESCAL43 and execute the following command line "**`sh ip bgp`**". The BGP session between MESCAL31 and MESCAL42 must be down.

Expected result : When executing the command above in MESCAL21, in Next Hop column, the following IP address must present:

- For **`212.167.0.0/21`** entry **`1.1.1.5`**
- For **`62.42.0.0/16`** entry **`1.1.1.5`**
- For **`1.1.1.0/30`** entry **`1.1.1.5`**

When executing the command above in MESCAL41, in Next Hop column, the following IP address must present:

- For **`216.191.64.0/20`** entry **`3.3.3.1`**

When executing the command above in MESCAL43, in Next Hop column, the following IP address must present:

- For **`216.191.64.0/20`** entry **`3.3.3.1`**

Test Reference : **TB_P2_FUNCT/CMES/15**

Test Purpose : Verify the conformance of NLRI field.

Procedure : Log to MESCAL11 and define three local-QoS-classes for MESCAL11.

Scenario 1: Add the following commands:

- **`193.251.128.0/19 local-qos-class 0`**
- **`212.167.0.0/21 local-qos-class 0`**
- **`62.42.0.0/16 local-qos-class 0`**
- **`193.251.240.0/20 local-qos-class 0`**

Scenario 2: Add the following commands:

- **`193.251.128.0/19 local-qos-class 1`**
- **`212.167.0.0/21 local-qos-class 1`**

- **`62.42.0.0/16 local-qos-class 1`**
- **`193.251.240.0/20 local-qos-class 1`**

Scenario 3: Add the following commands:

- **`193.251.128.0/19 local-qos-class 0 1 2`**
- **`212.167.0.0/21 local-qos-class 0 1 2`**
- **`62.42.0.0/16 local-qos-class 0 1 2`**
- **`193.251.240.0/20 local-qos-class 0 1 2`**

Scenario 4: Add the following commands:

- **`193.251.128.0/19 local-qos-class 0`**
- **`212.167.0.0/21 local-qos-class 1 2`**
- **`62.42.0.0/16 local-qos-class 2`**
- **`193.251.240.0/20 local-qos-class 0 1`**

A traffic analyser must be launched in MESCAL11 and capture the traffic that is sent in interface used to connect to MESCAL21.

Expected result : With the traffic analyzer, we must visualize the following results according to the Scenario we are studing:

- Scenario 1: Only one QoS_NLRI message is sent to MESCAL21. The NLRI field contains the following prefixes: **`193251.0/19`**, **`212.167.0.0/21`**, **`62.42.0.0/16`** and **`193.251.240.0/20`**.
- Scenario 2: Only one QoS_NLRI message is sent to MESCAL21. The NLRI field contains the following prefixes: **`193251.0/19, 212.167.0.0/21, 62.42.0.0/16`** and **`193.251.240.0/20.`**
- Scenario 3: Three QoS_NLRI messages are sent to MESCAL21. The NLRI field of each QoS_NLRI message contains the following prefixes: **`193251.0/19, 212.167.0.0/21, 62.42.0.0/16`** and **`193.251.240.0/20.`**
- Scenario 4: Three QoS_NLRI messages are sent to MESCAL21.
  - The NLRI field of the first message contains **`193251.0/19`** and **`193.251.240.0/20.`**
  - The NLRI field of the second message contains **`212.167.0.0/21`** and **`62.42.0.0/16.`**
  - The NLRI field of the third message contains **`212.167.0.0/21`** and **`193.251.240.0/20.`**

14.2.2 TB_P2_FUNCT/DSCP

Test Reference : **TB_P2_FUNCT/DSCP/1**

Test Purpose : Validate that egress DSCP swapping operation is correctly achieved when receiving BGP UPDATE messages.

Procedure : Configure MESCAL11'q-BGP process to be instantiated in the local QoS classes bellow:

- lqc1.dscp = 0
- lqc2.dscp = 10
- lqc3.dscp = 12
- lqc4.dscp = 14
- lqc5.dscp = 26
- lqc6.dscp = 13

- lqc7.dscp = 63

In addition, the following mapping between local QoS classes and meta-QoS-classes is also configured in MESCAL11:

- 0 is mapped to 0
- 10 is mapped to 26
- 12 is mapped to 28
- 14 is mapped to 30

MESCAL11 Network prefixes are to be announced to MESCAL21 in the local QoS classes as specified bellow:

- Scenario 1: **193.251.128.0/19** is to be announced in lqc1
- Scenario 2: **193.251.128.0/19** is to be announced in lqc1, lqc2, lqc3 and lqc4
- Scenario 3: **193.251.128.0/19** is to be announced in lqc5, lqc6 and lqc7
- Scenario 4: **193.251.128.0/19** is to be announced in lqc1, lqc2, lqc3 and lqc7

A traffic analyzer must be used in the egress of MESCAL11 in order to verify the value carried in q-BGP UPDATE messages.

Expected result : In egress of MESCAL11, QoS class identifier of QoS_NLRI attribute must be set to the value that is listed below:

- Scenario 1: QoS_NLRI message must have a QoS class identifier set to 0 for the prefix **193.251.128.0/19**
- Scenario 2: Four q-BGP UPDATE messages for the prefix **193.251.128.0/19** must be sent to MESCAL21 with different QoS_NLRI attributes. QoS class identifier of the first q-BGP UPDATE message is set to 0. QoS class identifier of the second q-BGP UPDATE message is set to 26. QoS class identifier of the third q-BGP UPDATE message is set to 28 and QoS class identifier of the fourth q-BGP UPDATE message is set to 30.
- Scenario 3: Only one q-BGP UPDATE message for the prefix **193.251.128.0/19** must be sent to MESCAL21. QoS class identifier valued in QoS_NLRI attribute is 0.
- Scenario 4: Three q-BGP UPDATE messages for the prefix **193.251.128.0/19** must be sent to MESCAL21 with different QoS_NLRI attributes. QoS class identifier of the first q-BGP UPDATE message is set to 0. QoS class identifier of the second q-BGP UPDATE message is set to 26. QoS class identifier of the third q-BGP UPDATE message is set to 28.

Test Reference : **TB_P2_FUNCT/DSCP/2**
Test Purpose : Validate that ingress DSCP swapping operation is correctly achieved when receiving BGP UPDATE messages.

Procedure : Configure MESCAL21'q-BGP process to be instantiated in the local QoS classes bellow:

- lqc1.dscp = 0
- lqc2.dscp = 18
- lqc3.dscp = 20
- lqc4.dscp = 22
- lqc5.dscp = 26
- lqc6.dscp = 13

- lqc7.dscp = 63

Configure MESCAL11'q-BGP process to be instantiated in the local QoS classes bellow:

- lqc1.dscp = 0
- lqc2.dscp = 26
- lqc3.dscp = 28
- lqc4.dscp = 30
- lqc5.dscp = 44
- lqc6.dscp = 63
- lqc7.dscp = 55

The following mapping between local QoS classes and meta-QoS-classes is also configured in MESCAL21:

- 0 is mapped to 0
- 18 is mapped to 26
- 20 is mapped to 28
- 22 is mapped to 30

Network prefixes are to be announced to MESCAL21 in the local QoS classes as specified bellow:

- Scenario 1: **`193.251.128.0/19`** is to be announced in lqc1
- Scenario 2: **`193.251.128.0/19`** is to be announced in lqc1, lqc2, lqc3 and lqc4
- Scenario 3: **`193.251.128.0/19`** is to be announced in lqc5, lqc6 and lqc7
- Scenario 4: **`193.251.128.0/19`** is to be announced in lqc1, lqc2, lqc3 and lqc7

A traffic analyzer must be used in the egress of MESCAL11 in order to verify the value carried in q-BGP UPDATE messages.

In MESCAL21, the following '**`sh ip route`**' or '**`sh ip bgp`**' commands must be executed in order to verify the existence of the network prefix announced by MESCAL11 in the appropriate local QoS class.

Expected result : when executing the '**`sh ip route`**' command in MESCAL21, the **`193.251.128.0/19`** entry must be in the appropriate local QoS class.

- Scenario 1: The prefix **`193.251.128.0/19`** exists only in the local QoS class 0
- Scenario 2: The prefix **`193.251.128.0/19`** exists in the following local-QoS-class planes:
  - Lqc0= 0
  - Lqc1= 18
  - Lqc2= 20
  - Lqc3= 22
- Scenario 3: The prefix **`193.251.128.0/19`** exists only in the local QoS class 0
- Scenario 4: The prefix c exists in the following local-QoS-class planes:
  - Lqc0= 0
  - Lqc1= 18
  - Lqc2= 20

14.2.3 TB_P2_FUNCT/QCMP

Conformance status is optional for all attributes in the following tests.

Test Reference : **TB_P2_FUNCT/QCMP/1**

Test Purpose :

Check that the **reserved-rate** QoS parameter is correctly computed by the receiving ASBR.

Procedure :

Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic. Successively carry out the following elementary tests with the following parameters values:
A - Set l-QC11.reserved-rate = 500 and l-QC21.reserved-rate = 250.
B - Set l-QC11.reserved-rate = 250 and l-QC21.reserved-rate = 500.
C - Set l-QC11.reserved-rate = 0 and l-QC21.reserved-rate = 250.
D - Set l-QC11.reserved-rate = 500 and l-QC21.reserved-rate = 0.
E - Set l-QC11.reserved-rate = 65535 and l-QC21.reserved-rate = 250.
F - Set l-QC11.reserved-rate = 500 and l-QC21.reserved-rate = 65535.
G - Set l-QC11.reserved-rate = 65535 and l-QC21.reserved-rate = 65535.
H - Set l-QC11.reserved-rate = 0 and l-QC21.reserved-rate = 65535.
I - Set l-QC11.reserved-rate = null and l-QC21.reserved-rate = 500.
J - Set l-QC11.reserved-rate = 500 and l-QC21.reserved-rate = null.
K - Set l-QC11.reserved-rate = null and l-QC21.reserved-rate = null.
All other QoS parameters are un-valued. Configure q-BGP so that AS2 announces network 194.52.168.0/21. Note on AS1 the value of the computed reserved-rate QoS parameter for this network.

Expected result :

A - reserved-rate = 250
B - reserved-rate = 250
C - reserved-rate = 0
D - reserved-rate = 0
E - reserved-rate = 250
F - reserved-rate = 500
G - reserved-rate = 65535
H - reserved-rate = 0
I - reserved-rate = null
J - reserved-rate = null
K - reserved-rate = null

Test Reference : **TB_P2_FUNCT/QCMP/2**

Test Purpose :

Check that invalid **reserved-rate** values are rejected by the command-line interface.

Procedure :

Specify the following invalid values on AS1:
A - Set l-QC11.reserved-rate = 65736
B - Set l-QC11.reserved-rate = 99999999999999999
C - Set l-QC11.reserved-rate = -1
D - Set l-QC11.reserved-rate = -65736
E - Set l-QC11.reserved-rate = string
F - Set l-QC11.reserved-rate = 1string

Expected result :

All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/3**

Test Purpose :

Check that the **available-rate** QoS parameter is correctly computed by the receiving ASBR.

Procedure :

Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic. Successively carry out the following elementary tests with the following parameters values:
A - Set l-QC11.available-rate = 500 and l-QC21.available-rate = 250.
B - Set l-QC11.available-rate = 250 and l-QC21.available-rate = 500.
C - Set l-QC11.available-rate = 0 and l-QC21.available-rate = 250.
D - Set l-QC11.available-rate = 500 and l-QC21.available-rate = 0.
E - Set l-QC11.available-rate = 65535 and l-QC21.available-rate = 250.
F - Set l-QC11.available-rate = 500 and l-QC21.available-rate = 65535.
G - Set l-QC11.available-rate = 65535 and l-QC21.available-rate = 65535.
H - Set l-QC11.available-rate = 0 and l-QC21.available-rate = 65535.
I - Set l-QC11.available-rate = null and l-QC21.available-rate = 500.
J - Set l-QC11.available-rate = 500 and l-QC21.available-rate = null.
K - Set l-QC11.available-rate = null and l-QC21.available-rate = null.
All other QoS parameters are un-valued. Configure q-BGP so that AS2 announces network 194.52.168.0/21. Note on AS1 the value of the computed available-rate QoS parameter for this network.

Expected result :

A - available-rate = 250
B - available-rate = 250
C - available-rate = 0
D - available-rate = 0
E - available-rate = 250
F - available-rate = 500
G - available-rate = 65535
H - available-rate = 0
I - available-rate = null
J - available-rate = null
K - available-rate = null

Test Reference : **TB_P2_FUNCT/QCMP/4**

Test Purpose :

Check that invalid **available-rate** values are rejected by the command-line interface.

Procedure :

Specify the following invalid values on AS1:
A - Set l-QC11.available-rate = 65736
B - Set l-QC11.available-rate = 99999999999999999
C - Set l-QC11.available-rate = -1
D - Set l-QC11.available-rate = -65736
E - Set l-QC11.available-rate = string
F - Set l-QC11.available-rate = 1string

Expected result :

All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/5**

Test Purpose :

Check that the **min-owd** (minimum one-way-delay) QoS parameter is correctly computed by the receiving ASBR.

Procedure :

Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic. Successively carry out the following elementary tests with the following parameters values:
A - Set l-QC11.min-owd = 500 and l-QC21.min-owd = 250.
B - Set l-QC11.min-owd = 250 and l-QC21.min-owd = 500.
C - Set l-QC11.min-owd = 0 and l-QC21.min-owd = 250.

D - Set l-QC11.min-owd = 500 and l-QC21.min-owd = 0.
E - Set l-QC11.min-owd = 65535 and l-QC21.min-owd = 250.
F - Set l-QC11.min-owd = 500 and l-QC21.min-owd = 65535.
G - Set l-QC11.min-owd = 65535 and l-QC21.min-owd = 65535.
H - Set l-QC11.min-owd = 0 and l-QC21.min-owd = 65535.
I - Set l-QC11.min-owd = null and l-QC21.min-owd = 500.
J - Set l-QC11.min-owd = 500 and l-QC21.min-owd = null.
K - Set l-QC11.min-owd = null and l-QC21.min-owd = null.
All other QoS parameters are un-valued. Configure q-BGP so that AS2 announces network 194.52.168.0/21. Note on AS1 the value of the computed min-owd QoS parameter for this network.

Expected result :

A - min-owd = 750
B - min-owd = 750
C - min-owd = 250
D - min-owd = 500
E - min-owd = 65535
F - min-owd = 65535
G - min-owd = 65535
H - min-owd = 65535
I - min-owd = null
J - min-owd = null
K - min-owd = null

Test Reference : **TB_P2_FUNCT/QCMP/6**

Test Purpose :

Check that invalid **min-owd** values are rejected but the command-line interface.

Procedure :

Specify the following invalid values on AS1:
A - Set l-QC11.min-owd = 65736
B - Set l-QC11.min-owd = 99999999999999999
C - Set l-QC11.min-owd = -1
D - Set l-QC11.min-owd = -65736
E - Set l-QC11.min-owd = string
F - Set l-QC11.min-owd = 1string

Expected result :

All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/7**

Test Purpose :

Check that the **max-owd** (maximum one-way-delay) QoS parameter is correctly computed by the receiving ASBR.

Procedure :

Same procedure as TB_P2_FUNCT/QCMP/5. Replace min-owd by **max-owd**.

Expected result :

A - max-owd = 750
B - max-owd = 750
C - max-owd = 250
D - max-owd = 500
E - max-owd = 65535
F - max-owd = 65535
G - max-owd = 65535
H - max-owd = 65535

I - max-owd = null
J - max-owd = null
K - max-owd = null

Test Reference : **TB_P2_FUNCT/QCMP/8**
Test Purpose :
Check that invalid **max-owd** values are rejected by the command-line interface.
Procedure :
Same procedure as TB_P2_FUNCT/QCMP/6. Replace min-owd by **max-owd**.
Expected result :
All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/9**
Test Purpose :
Check that the **average-owd** (average one-way-delay) QoS parameter is correctly computed by the receiving ASBR.
Procedure :
Same procedure as TB_P2_FUNCT/QCMP/5. Replace min-owd by **average-owd**.
Expected result :
A - average-owd = 750
B - average -owd = 750
C - average -owd = 250
D - average -owd = 500
E - average -owd = 65535
F - average -owd = 65535
G - average -owd = 65535
H - average -owd = 65535
I - average -owd = null
J - average -owd = null
K - average -owd = null

Test Reference : **TB_P2_FUNCT/QCMP/10**
Test Purpose :
Check that invalid **average-owd** values are rejected by the command-line interface.
Procedure :
Same procedure as TB_P2_FUNCT/QCMP/6. Replace min-owd by **average-owd**.
Expected result :
All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/11**
Test Purpose :
Check that the **loss-rate** QoS parameter is correctly computed by the receiving ASBR.
Procedure :
Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic. Successively carry out the following elementary tests with the following parameters values:
A - Set l-QC11.loss-rate = 500 and l-QC21.loss-rate = 250.
B - Set l-QC11.loss-rate = 250 and l-QC21.loss-rate = 500.
C - Set l-QC11.loss-rate = 0 and l-QC21.loss-rate = 250.
D - Set l-QC11.loss-rate = 500 and l-QC21.loss-rate = 0.
E - Set l-QC11.loss-rate = 65535 and l-QC21.loss-rate = 250.
F - Set l-QC11.loss-rate = 500 and l-QC21.loss-rate = 65535.

G - Set l-QC11.loss-rate = 65535 and l-QC21.loss-rate = 65535.
H - Set l-QC11.loss-rate = 0 and l-QC21.loss-rate = 65535.
I - Set l-QC11.loss-rate = null and l-QC21.loss-rate = 500.
J - Set l-QC11.loss-rate = 500 and l-QC21.loss-rate = null.
K - Set l-QC11.loss-rate = null and l-QC21.loss-rate = null.
All other QoS parameters are un-valued. Configure q-BGP so that AS2 announces network 194.52.168.0/21. Note on AS1 the value of the computed loss-rate QoS parameter for this network.
A loss-rate = 1 means 0.001%

Expected result :
A - loss-rate = 748 or 749 depending on the round-off
B - loss-rate = 748 or 749 depending on the round-off
C - loss-rate = 250
D - loss-rate = 500
E - loss-rate = 65535
F - loss-rate = 65535
G - loss-rate = 65535
H - loss-rate = 65535
I - loss-rate = null
J - loss-rate = null
K - loss-rate = null

Test Reference : **TB_P2_FUNCT/QCMP/12**
Test Purpose :
Check that invalid **loss-rate** values are rejected by the command-line interface.
Procedure :
Specify the following invalid values on AS1:
A - Set l-QC11.loss-rate to 65736
B - Set l-QC11.loss-rate to 99999999999999999
C - Set l-QC11.loss-rate to -1
D - Set l-QC11.loss-rate to -65736
E - Set l-QC11.loss-rate to string
F - Set l-QC11.loss-rate to 1string
Expected result :
All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/13**
Test Purpose :
Check that the **jitter** QoS parameter is correctly computed by the receiving ASBR.
Procedure :
Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic. Successively carry out the following elementary tests with the following parameters values:
A - Set l-QC11.jitter = 500 and l-QC21.jitter = 250.
B - Set l-QC11.jitter = 250 and l-QC21.jitter = 500.
C - Set l-QC11.jitter = 0 and l-QC21.jitter = 250.
D - Set l-QC11.jitter = 500 and l-QC21.jitter = 0.
E - Set l-QC11.jitter = 65535 and l-QC21.jitter = 250.
F - Set l-QC11.jitter = 500 and l-QC21.jitter = 65535.
G - Set l-QC11.jitter = 65535 and l-QC21.jitter = 65535.
H - Set l-QC11.jitter = 0 and l-QC21.jitter = 65535.
I - Set l-QC11.jitter = null and l-QC21.jitter = 500.
J - Set l-QC11.jitter = 500 and l-QC21.jitter = null.
K - Set l-QC11.jitter = null and l-QC21.jitter = null.
All other QoS parameters are un-valued. Configure q-BGP so that AS2 announces

network 194.52.168.0/21. Note on AS1 the value of the computed jitter QoS parameter for this network.

Expected result :

- A - jitter = 750
- B - jitter = 750
- C - jitter = 250
- D - jitter = 500
- E - jitter = 65535
- F - jitter = 65535
- G - jitter = 65535
- H - jitter = 65535
- I - jitter = null
- J - jitter = null
- K - jitter = null

Test Reference : **TB_P2_FUNCT/QCMP/14**

Test Purpose :

Check that invalid **jitter** values are rejected by the command-line interface.

Procedure :

Specify the following invalid values on AS1:

- A - Set l-QC11.jitter = 65736
- B - Set l-QC11.jitter = 99999999999999999
- C - Set l-QC11.jitter = -1
- D - Set l-QC11.jitter = -65736
- E - Set l-QC11.jitter = string
- F - Set l-QC11.jitter = 1string

Expected result :

All the above rejected values must be rejected by the command-line interface.

Test Reference : **TB_P2_FUNCT/QCMP/15**

Test Purpose :

Check that the receiving ASBR is able to compute multiple QoS parameters contained in an announcement.

Procedure :

Establish a pSLS between AS1 and AS2 for exchanging MC1 traffic.

Set the l-QC11 parameter values as follow:
Set l-QC11.reserved-rate = 500.
Set l-QC11.available-rate = 200.
Set l-QC11.loss-rate = 1000.
Set l-QC11.min-owd = 10.
Set l-QC11.max-owd = 25.
Set l-QC11.average-owd = 15.
Set l-QC11.jitter = 5.

Set the l-QC21 parameter values as follow:
Set l-QC21.reserved-rate = 300.
Set l-QC21.available-rate = 250.
Set l-QC21.loss-rate = 2500.
Set l-QC21.min-owd = 30.
Set l-QC21.max-owd = 66.
Set l-QC21.average-owd = 55.
Set l-QC21.jitter = 15.

Configure q-BGP so that AS2 announces network 194.52.168.0/21. Note on AS1 the value of the computed QoS parameters for this network.

Expected result :

reserved-rate = 300.
available-rate = 200.
loss-rate = 3475.
min-owd = 40.
max-owd = 91.
average-owd = 70.
jitter = 20.

Test Reference : **TB_P2_FUNCT/QCMP/16**

Test Purpose :

Check that the receiving ASBR is able to compute multiple QoS parameters for a same prefix announced within different meta-QoS-planes.

Procedure :

- Establish a pSLS between AS1 and AS2 for exchanging MC1, MC2 and MC3 traffic.
- Configure l-QC12 and l-QC13 with the same QoS parameter values as l-QC11.
- Configure l-QC22 and l-QC23 with the same QoS parameter values as l-QC21.
- Configure q-BGP so that AS2 announces network 194.52.168.0/21 on meta-QoS-planes MC1, MC2 and MC3.
- Note on AS1 the value of the computed QoS parameters for this network and per meta-QoS-plane.

Expected result :

Announcements received for this prefix within the 3 meta-QoS-class planes must have the same values as expected in **TB_P2_FUNCT/QCMP/15.**

14.2.4 TB_P2_FUNCT/RSEL

Sub-group preamble

Unless specified, tests hereafter described involve AS1, AS2, AS3 and AS4. Peering need to be setup between: AS1 & AS2, AS1 & AS3, AS3 & AS4, AS2 & AS4 as described in the testbed configuration section. Only Mescal-42 ASBR from AS4 is involved in this series of tests.

pSLSes established between ASes concern MC1 and Best-effort only.

AS1 announces only the network prefix 193.251.128.0/19 within MC1. Others ASes propagate the route but do not announce any of their own networks.

QoS guarantees of l-QC implementing MC1 in each AS are defined as follow:

Attributes	Priority	l-QC11	l-QC21	l-QC31	l-QC41
Reserved-rate	1	600	800	400	2000
Available-rate	2	500	400	600	2000
Loss-rate	3	1000	500	1500	200
Min-owd	4	10	15	25	5
Max-owd	5	50	25	10	15
Average-owd	6	30	20	15	10
Jitter	7	20	4	5	5

Table 82: Local QoS Class Characteristics

The priority level and the conformance (mandatory/optional) status of each QoS attribute is specified by each test.

The precision is set to 0%, for all attributes of all l-QCs, unless it is explicitly specified.

Group 2 QoS Service Capability is also configured unless explicit related configuration is recommended by a given test.

Test Reference : **TB_P2_FUNCT/RSEL/1**

Test Purpose :

Check that several ASes involved in the loose service option are able to exchange route updates containing correctly computed QoS information.

Procedure :

Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Check the number of route received by AS4 for the network prefix `193.251.128.0/19`.
Check that a route has been selected.
Check that QoS parameters have the expected values.

Expected result :

Route431 via AS4,AS3,AS1:

reserved-rate = 400.
available-rate = 500.
loss-rate = 2680.
min-owd = 40.
max-owd = 75.
average-owd = 55.
jitter = 30.

Route421 via AS4,AS2,AS1:

reserved-rate = 600.
available-rate = 400.
loss-rate = 1692.
min-owd = 30.
max-owd = 90.
average-owd = 60.
jitter = 29.

AS4 must select **route421**

Test Reference : **TB_P2_FUNCT/RSEL/2**

Test Purpose :

Check, in simple Scenarios, that the route selection process takes into account the priority level of each QoS attribute.

Procedure :

As defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Using the table described in the preamble as a reference for each sub-test, exchange the priority value of:

- Scenario A1 - Reserved-rate and the available-rate QoS parameter
- Scenario B1 - Reserved-rate and the loss-rate QoS parameter
- Scenario C1 - Reserved-rate and the min-owd QoS parameter
- Scenario D1 - Reserved-rate and the max-owd QoS parameter
- Scenario E1 - Reserved-rate and the average-owd QoS parameter
- Scenario F1 - Reserved-rate and the jitter QoS parameter

One performed, exchange the QoS attribute values of l-QC21 and l-QC31. Perform the 6 tests again (A2-F2)

Expected result :

- Scenario A1 - Route431 is selected
- Scenario B1 - Route421 is selected
- Scenario C1 - Route421 is selected
- Scenario D1 - Route431 is selected
- Scenario E1 - Route431 is selected
- Scenario F1 - Route421 is selected
- Scenario A2 - Route421 is selected
- Scenario B2 - Route431 is selected
- Scenario C2 - Route431 is selected
- Scenario D2 - Route421 is selected
- Scenario E2 - Route421 is selected
- Scenario F2 - Route431 is selected

Test Reference : **TB_P2_FUNCT/RSEL/3**

Test Purpose :

Check that the route selection process takes into account the QoS attributes which have a lower priority when the previous attributes (with higher priority) are equivalent.

Procedure :

Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Give l-QC31 the same values as l-QC21. Then, perform the following sub-tests:

- Scenario A – Set l-QC31.available-rate=450

- Scenario B – Set l-QC31.available-rate=350
- Scenario C – Set l-QC31.available-rate=400 and set l-QC31.loss-rate=400
- Scenario D – Set l-QC31.available-rate=400 and set l-QC31.loss-rate=600
- Scenario E – Set l-QC31.loss-rate=500 and set l-QC31.min-owd=20
- Scenario F – Set l-QC31.loss-rate=500 and set l-QC31.min-owd=10
- Scenario G – Set l-QC31.min-owd=15 and set l-QC31.max-owd=30
- Scenario H – Set l-QC31.min-owd=15 and set l-QC31.max-owd=20
- Scenario I – Set l-QC31.max-owd=25 and set l-QC31.average-owd=25
- Scenario J – Set l-QC31.max-owd=25 and set l-QC31.average-owd=15
- Scenario K – Set l-QC31.average-owd=20 and set l-QC31.jitter=5
- Scenario L – Set l-QC31.average-owd=20 and set l-QC31.jitter=3

Expected result :

For each sub-test, check that AS4 received 2 routes with the same QoS attribute values except one of them. One of them must have selected.

- Scenario A – Available-rate is different - Route via AS3 must be selected
- Scenario B – Available-rate is different - Route via AS2 must be selected
- Scenario C – Loss-rate is different - Route via AS3 must be selected
- Scenario D – Loss-rate is different - Route via AS2 must be selected
- Scenario E – Min-owd is different - Route via AS2 must be selected
- Scenario F – Min-owd is different - Route via AS3 must be selected
- Scenario G – Max-owd is different - Route via AS2 must be selected
- Scenario H – Max-owd is different - Route via AS3 must be selected
- Scenario I – Average-owd is different - Route via AS2 must be selected
- Scenario J – Average-owd is different - Route via AS3 must be selected
- Scenario K – Jitter is different - Route via AS2 must be selected
- Scenario L – Jitter is different - Route via AS3 must be selected

---

Test Reference : **TB_P2_FUNCT/RSEL/4**

Test Purpose :

Check that the precision command line parameter is correctly handled for the reserved-rate QoS attribute.

Procedure :

Start as defined by this sub-group preamble.

Conformance status is mandatory for all attributes.

- Scenario 1: Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.reserved-rate=10%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result :

- Scenario 1: Route via AS3 must be selected (the decision is enforced by the available-rate)
- Scenario 2: Route via AS2 must be selected (the decision is enforced by the reserved-rate)

Test Reference : **TB_P2_FUNCT/RSEL/5**

Test Purpose : Check that the precision command line parameter is correctly handled for the available-rate QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.available-rate=10%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: Route via AS2 must be selected (the decision is enforced by the loss-rate)
- Scenario 2: Route via AS3 must be selected (the decision is enforced by the available-rate)

Test Reference : **TB_P2_FUNCT/RSEL/6**

Test Purpose : Check that the precision command line parameter is correctly handled for the loss-rate QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.loss-rate=50%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.loss-rate=10%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: Route via AS2 must be selected (the decision is enforced by the min-owd)
- Scenario 2: Route via AS2 must be selected (the decision is enforced by the loss-rate)

Test Reference : **TB_P2_FUNCT/RSEL/7**

Test Purpose : Check that the precision command line parameter is correctly handled for the min-owd QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.loss-rate=50%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.min-owd=20%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.min-owd=5%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: Route via AS3 must be selected (the decision is enforced by the max-owd)
- Scenario 2: Route via AS2 must be selected (the decision is enforced by the min-owd)

Test Reference : **TB_P2_FUNCT/RSEL/8**

Test Purpose : Check that the precision command line parameter is correctly handled for the max-owd QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.loss-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.min-owd=20%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.max-owd=50%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.max-owd=5%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: Route via AS3 must be selected (the decision is enforced by the average-owd)

- Scenario 2: Route via AS3 must be selected (the decision is enforced by the max-owd)

Test Reference : **TB_P2_FUNCT/RSEL/9**

Test Purpose : Check that the precision command line parameter is correctly handled for the average-owd QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.loss-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.min-owd=20%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.max-owd=50%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.average-owd=50%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.average-owd=2%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: Route via AS2 must be selected (the decision is enforced by the jitter)
- Scenario 2: Route via AS3 must be selected (the decision is enforced by the average-owd)

Test Reference : **TB_P2_FUNCT/RSEL/10**

Test Purpose : Check that the precision command line parameter is correctly handled for the jitter QoS attribute.

Procedure : Start as defined by this sub-group preamble.
Conformance status is mandatory for all attributes.

Set the precision for l-QC41.reserved-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.available-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.loss-rate=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.min-owd=20%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.max-owd=50%. The related QoS attribute of the 2 routes overlap.

Set the precision for l-QC41.average-owd=50%. The related QoS attribute of the 2 routes overlap.

- Scenario 1: Set the precision for l-QC41.jitter=10%. The related QoS attribute of the 2 routes overlap.
- Scenario 2: Set the precision for l-QC41.jitter=0.5%. The related QoS attribute of the 2 routes DO NOT overlap.

Expected result : The following results must be obtained:

- Scenario 1: The same Route in best effort and in l-QC41 plan must be selected (depends on BGP route selection process)
- Scenario 2: Route via AS2 must be selected (the decision is enforced by the jitter)

Test Reference : **TB_P2_FUNCT/RSEL/11**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of reserved rate to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send reserved-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send reserved-rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send reserved-rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send reserved-rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Selected route is R421.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/12**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of available rate to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send available-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send available -rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send available -rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send available -rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/13**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of loss rate to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send loss-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send loss-rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send loss-rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send loss-rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/14**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :

q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of min-owd to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send min-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send min-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send min-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send min-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/15**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :

q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of max-owd to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send max-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send max-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send max-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send max-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/16**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of average-owd to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send average-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send average-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send average-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send average-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/17**
Test Purpose : Validate the behavior of q-BGP when mandatory parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of jitter to mandatory:

- Scenario 1: Configure AS1, AS2 and AS3 to send jitter as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send jitter as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send jitter as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send jitter as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. Route Route421 is selected.
- Scenario 2: Only one Route is received from AS3. This route is selected.
- Scenario 3: Only one Route is received from AS2. This route is selected.
- Scenario 4: No route has been received for the specified prefix.

Test Reference : **TB_P2_FUNCT/RSEL/18**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of reserved rate to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send reserved-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send reserved-rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send reserved-rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send reserved-rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R431 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R431 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/19**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of available rate to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send available-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send available -rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send available -rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send available -rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/20**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of loss rate to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send loss-rate as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send loss-rate as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send loss-rate as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send loss-rate as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/21**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of min-owd to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send min-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send min-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send min-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send min-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/22**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of max-owd to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send max-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send max-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send max-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send max-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/23**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of average-owd to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send average-owd as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send average-owd as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send average-owd as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send average-owd as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

Test Reference : **TB_P2_FUNCT/RSEL/24**
Test Purpose : Validate the behavior of q-BGP when optional parameters aren't received.

Procedure :
q-BGP configuration will be update as specified in the description of each scenario:
Set the conformance status of jitter to optional:

- Scenario 1: Configure AS1, AS2 and AS3 to send jitter as described in the introduction of this test group.
- Scenario 2: Configure only AS1 and AS3 to send jitter as described in the introduction of this test group.
- Scenario 3: Configure only AS1 and AS2 to send jitter as described in the introduction of this test group.
- Scenario 4: Configure only AS2 and AS3 to send jitter as described in the introduction of this test group.

Expected result : The following results must be obtained:

- Scenario 1: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 2: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 3: Two routes are received by AS4, from AS3 and AS2. R421 is selected.
- Scenario 4: Two routes are received by AS4, from AS3 and AS2. R421 is selected.

14.2.5 TB_P2_FUNCT/INT

Test Reference : **TB_P2_FUNCT/INT/1**
Test Purpose : Validate the behavior of a BGP speaker when receiving unrecognized optional parameters.

Procedure :
q-BGPD process is activated in MESCAL11. BGPD is activated in MESCAL11. Configure MESCAL21 to not support capabilities negotiation. MESCAL11 is configured to negotiate capabilities.

- Scenario 1: Configure MESCAL11 as a neighbor of MESCAL21. MESCAL21 is to be configured in MESCAL11 as a neighbor. Check if the BGP session has been established.

Configure MESCAL21 to support capabilities negotiation. MESCAL11 is configured to negotiate capabilities.

- Scenario 2: Configure MESCAL11 as a neighbor of MESCAL21. MESCAL21 is to be configured in MESCAL11 as a neighbor. Configure MESCAL11 to support QoS Service Capability Group 1. Check the messages exchanged between MESCAL21 and MESCAL31 with a traffic analyzer.
- Scenario 3: Configure MESCAL11 as a neighbor of MESCAL21. MESCAL21 is to be configured in MESCAL11 as a neighbor. Configure MESCAL11 to support QoS Service Capability Group 2. Check the messages exchanged between MESCAL21 and MESCAL31 with a traffic analyzer.
- Scenario 4: Configure MESCAL11 as a neighbor of MESCAL21. MESCAL21 is to be configured in MESCAL11 as a neighbor. Configure MESCAL11 to support QoS Service Capability Group 1 and Group 2. Check the messages exchanged between MESCAL21 and MESCAL31 with a traffic analyzer.

Expected result : The following results must be obtained:

- Scenario 1: MESCAL21 has to close the BGP session.
- Scenario 2: MESCAL12 has to send a notification message with the Error Sub Code set to Unsupported Optional Parameter. MESCAL11 should re-attempt to open a BGP session with MESCAL21 but with without sending to the peer the Capabilities Optional Parameter.
- Scenario 3: MESCAL12 has to send a notification message with the Error Sub Code set to Unsupported Optional Parameter. MESCAL11 should re-attempt to open a BGP session with MESCAL21 but with without sending to the peer the Capabilities Optional Parameter.
- Scenario 4: MESCAL12 has to send a notification message with the Error Sub Code set to Unsupported Optional Parameter. MESCAL11 should re-attempt to open a BGP session with MESCAL21 but with without sending to the peer the Capabilities Optional Parameter.

Test Reference : **TB_P2_FUNCT/INT/2**
Test Purpose : Validate the behavior of a q-BGP speaker when receiving notification set to unsupported capabilities from BGP speaker.

Procedure :
q-BGPD process is activated in MESCAL11. BGPD is activated in MESCAL11.
Configure MESCAL21 and MESCAL11 to support capabilities negotiation.

Configure local-QoS-class in MESCAL11 as specified in the introduction of the previous test group. Also AS1 networks that are listed in testbed configuration are also to be announced by MESCAL11 in l-QC1 and best effort.

AS2 Networks that are listed in testbed configuration are also to be announced by MESCAL21.

Launch a traffic analyzer in eth1 of MESCAL11. Check the messages that are exchanges between MESCAL11 and MESCAL12.

Expected result : MESCAL12 must not send QoS_NLRI messages to MESCAL12.
MESCAL12 must send NLRI information to MESCAL11.
MESCAL11 must send NLRI information to MESCAL12.

Test Reference : **TB_P2_FUNCT/INT/3**
Test Purpose : Validate the q-BGP router installs routes received from BGP speaker in best effort plane.

Procedure :

q-BGPD process is activated in MESCAL11. BGPD is activated in MESCAL11. Configure MESCAL21 and MESCAL11 to support capabilities negotiation.

Configure local-QoS-class in MESCAL11 as specified in the introduction of the previous test group. Also AS1 networks that are listed in testbed configuration are also to be announced by MESCAL11 in l-QC1 and best effort.

AS2 Networks that are listed in testbed configuration are also to be announced by MESCAL21.

Log to MESCAL11 and execute this command line: "**`sh ip bgp`**". Check if MESCAL21 networks are listed.

Expected result : All MESCAL21's networks must be present in the MESCAL11 best effort RIB.

Test Reference : **TB_P2_FUNCT/INT/4**
Test Purpose : Validate the BGP router installs routes received from q-BGP speaker.

Procedure :

q-BGPD process is activated in MESCAL11. BGPD is activated in MESCAL11. Configure MESCAL21 and MESCAL11 to support capabilities negotiation.

Configure local-QoS-class in MESCAL11 as specified in the introduction of the previous test group. Also AS1 networks that are listed in testbed configuration are also to be announced by MESCAL11 in l-QC1 and best effort.

AS2 Networks that are listed in testbed configuration are also to be announced by MESCAL21.

Log to MESCAL12 and execute this command line: "**`sh ip bgp`**". Check if MESCAL11 networks are listed.

Expected result : All MESCAL11's networks must be present in the MESCAL21 BGP RIB.

14.3 Phase 3

The "null" value means no value.

14.3.1 TB_P3_FUNCT/CMES

14.3.1.1 Reminder

Each PCP message consists of the PCP header followed by a number of arguments depending on the nature of the operation.

```
          0               1               2               3
    +-------------+-------------+-------------+-------------+
    |   Version    |    Op Code   |        Message Length        |
 +--------------+--------------+--------------+--------------+
```

14.3.1.1.1 Open message

```
         0             1              2             3
   +-------------+-------------+-------------+-------------+
   |                                                       |
   |                         PCSID                         |
   |                                                       |
   |                                                       |
   +-------------+-------------+-------------+-------------+
```

14.3.1.1.2 Accept message

```
         0             1              2             3
   +-------------+-------------+-------------+-------------+
   |         KA-Timer          |///////////////////////////|
   +-------------+-------------+-------------+-------------+
```

14.3.1.1.3 Close message

```
          0             1              2             3
   +--------------+--------------+--------------+--------------+
   |         Error-Code          | ////////////////////////////|
   +--------------+--------------+--------------+--------------+
```

14.3.1.1.4 Path Error message

```
       +-------------+-------------+
2 bytes|            L1             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-COMPUTATION-ID                  |
       |----------------------------------//-------------------|
2 bytes|                    REQ-REFERENCE-ID                   |
       |----------------------------------//-------------------|
1 bytes|        REASON-CODE        |
       +-------------+-------------+
```

14.3.1.1.5 Cancel message

```
       +-------------+-------------+
2 bytes|            L1             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-COMPUTATION-ID                  |
       |----------------------------------//-------------------|
2 bytes|                    REQ-REFERENCE-ID                   |
       |-------------------------------------------------------|
```

14.3.1.1.6 Acknowledge message

```
       +-------------+-------------+
2 bytes|            L1             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-COMPUTATION-ID                  |
       |----------------------------------//-------------------|
2 bytes|                    REQ-REFERENCE-ID                   |
       |-------------------------------------------------------|
```

14.3.1.1.7 Request message

```
       +-------------+
1 byte |    TTL      |
       +-------------+
1 byte |     L0      |
       +-------------+-------------+-------------+-------------+
2 bytes|                       AS-NUMBER                       |
       +-------------+-------------+-------------+-------------+
       //                                                      //
       +-------------+-------------+-------------+-------------+
2 bytes|                       AS-NUMBER                       |
       +-------------+-------------+-------------+-------------+
2 bytes|            L1             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-COMPUTATION-ID                  |
       |-----------------------------------//------------------|
2 bytes|            L2             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-REFERENCE-ID                    |
       +-------------+-------------+-------//----+-------------+
2 bytes|                  REQ-REFERENCE-ID                     |
       +-------------+-------------+-------------+-------------+
1 byte |   ADD-TYPE  |
       +-------------+-------------+-------//----+-------------+
       |                  HEAD-END-ADDRESS                     |
       +-------------+-------------+-------//----+-------------+
       |                  TAIL-END-ADDRESS                     |
       +-------------+-------------+-------//----+-------------+
1 byte |  NUMBER-OF-QC-CONSTRAINT  +
       +-------------+-------------+
2 bytes|   QC-CONSTRAINT-LENGTH    +
       +-------------+-------------+
1 byte |   QOS-CLASS-IDENTIFIER    +
       +-------------+-------------+-------------+-------------+
1 byte |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
2 bytes|                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
       |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
       |                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
       |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
       |                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
```

14.3.1.1.8 Response message

```
       +-------------+-------------+
2 bytes|            L1             |
       +-------------+-------------+-------//----+-------------+
       |                  PATH-COMPUTATION-ID                  |
       |-----------------------------------//------------------|
2 bytes|                    REQ-REFERENCE-ID                   |
       |-----------------------------------//------------------|
1 bytes| PATH-LENGTH |
       +-------------+
1 byte |   ADD-TYPE  |
       +-------------+-------------+-------//----+-------------+
       |                       NEXT-HOP                        |
       +-------------+-------------+-------//----+-------------+
       //                                                     //
       +-------------+-------------+-------//----+-------------+
       |                       NEXT-HOP                        |
       +-------------+-------------+-------//----+-------------+
8 bytes|       VALIDITY-DATE       +
       +-------------+-------------+
1 byte |  NUMBER-OF-QC-CONSTRAINT  +
       +-------------+-------------+
2 bytes|   QC-CONSTRAINT-LENGTH    +
       +-------------+-------------+
1 byte |   QOS-CLASS-IDENTIFIER    +
       +-------------+-------------+---------------------------+
1 byte |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
2 bytes|                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
       |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
       |                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
       |   QOS-INFO-CODE           +   QOS-INFO-SUB-CODE       |
       +-------------+-------------+-------------+-------------+
       |                  QOS-INFO-VALUE                       |
       +-------------+-------------+-------------+-------------+
```

14.3.1.2 pSLS agreement

Configure the following pSLS between AS1 and AS2 (AS2 offers this pSLS to AS1):

- List of Meta-QoS-classes and bandwidth:
  - MQ1: TOS value 0x68 bandwidth 1Mbit
  - MQ2: TOS value 0x71 bandwidth 1Mbit
  - MQ3: TOS value 0x78 bandwidth 1Mbit
  - MQ4: TOS value 0x00 bandwidth 2Mbit
- Total bandwidth: 10Mbit
- MESCAL11'PCSID: 11.0.0.11
- MESCAL11'PCSID: 22.0.0.22

Test Reference : **TB_P3_FUNCT/CMES/1**

Test Purpose : Check the format of OPEN, CLOSE and ACCEPT messages.

Procedure :

Launch a traffic analyzer that captures MESCAL11' eth1 traffic.

Establish a PCP session between MESCAL11'PCS and MESCAL21'PCS.

When session is accepted by MESCAL12'PCS, close the session.

Verify that OPEN, CLOSE and ACCEPT messages are conform to [D1.2] specifications. For more details see introduction of this test group.

Expected result :

MESCAL12'PCS should answer with CLOSE or ACCEPT message to OPEN request received from MESCAL11'PCS.

OPEN, CLOSE and ACCEPT messages should have a format as indicated in introduction of this test group.

Op Code contained in common header is:

- 1: OPEN
- 2: ACCEPT
- 3: CLOSE

Test Reference : **TB_P3_FUNCT/CMES/2**

Test Purpose : Check the format of REQUEST, RESPONSE PATH-ERROR and ACKNOWLEDGE messages.

Procedure :

Launch a traffic analyzer that captures MESCAL11' eth1 traffic.

Establish a PCP session between MESCAL11'PCS and MESCAL21'PCS.

- Scenario 1: When PCP session is accepted by MESCAL12'PCS, configure MESCAL11'PCS to send a request for an LSP terminating in 2.2.2.1 in MC1 and a bandwidth of 0.5Mbit. No QoS parameters to be included in the request.

- Scenario 2: Configure MESCAL11'PCS to send a request for an LSP terminating in 2.2.2.1 in MC2 and a bandwidth of 2Mbit. No QoS parameters to be included in the request.

  Verify that REQUEST, RESPONSE, PATH-ERROR and ACKNOWLEDGE messages are conform to [D1.2] specifications. For more details see introduction of this test group.

Expected result :

- Scenario 1:
  - MESCAL12'PCS should answer with CLOSE or ACCEPT message to OPEN request received from MESCAL11'PCS.
  - MESCAL12'PCS should send a RESPONSE-PATH to MESCAL11
  - REQUEST, RESPONSE and ACKNOWLEDGE messages should have a format as indicated in introduction of this test group. Op Code contained in common header is:
    - 4: REQUEST
    - 5: RESPONSE
    - 8: ACKNOWLEDGE
- Scenario 2:

- MESCAL12'PCS should send a EEROR-PATH to MESCAL11.
- REQUEST, RESPONSE and ACKNOWLEDGE messages should have a format as indicated in introduction of this test group.

Test Reference : **TB_P3_FUNCT/CMES/3**

Test Purpose : Validate the REQ-REFERNCE-ID and PATH-COMPUTATION-ID

Procedure :
Same configure as for scenario 1 of **TB_P3_FUNCT/CMES/2**

Expected result :

REQ-REFERNCE-ID and PATH-COMPUTATION-ID that have been inserted in CANCEL, ACKNOWLEDGE and RESPONSE-PATH messages received from MESCAL12'PCS are identical to what have been inserted in REQUEST-PATH sent by MESCAL11'PCS.

Test Reference : **TB_P3_FUNCT/CMES/4**

Test Purpose : Validate QoS information contained in REQUEST-PATH message

Procedure :
Configure MESCAL11' PCS to send an REQUEST message to MESCAL12' PCS.

Execute tests **TB_P2_FUNCT/CMES/5** until **TB_P2_FUNCT/CMES/10**

Expected result : Same results as **TB_P2_FUNCT/CMES/5** until **TB_P2_FUNCT/CMES/10**

Test Reference : **TB_P3_FUNCT/CMES/5**

Test Purpose : Validate QoS information contained in RESPONSE-PATH message

Procedure :
Configure MESCAL11' PCS to send a REQUEST message to MESCAL12' PCS.

Execute tests **TB_P2_FUNCT/CMES/5** until **TB_P2_FUNCT/CMES/10**

Expected result : Same results as **TB_P2_FUNCT/CMES/5** until **TB_P2_FUNCT/CMES/10**

Test Reference : **TB_P3_FUNCT/CMES/6**

Test Purpose : Check the format of PATH-ERROR and messages.

Procedure :
Establish a PCP communication between MESCAL11'PSCA and MESCAL21'PCS.

Scenario 1: Force PATH-COMPUTATION-ID in REQUEST-PATH message to a value that already exists between the two peers.

Scenario 2: Force REF-COMPUTATION-ID in REQUEST-PATH message to a value already that has been handled.

Launch a traffic analyzer that captures MESCAL11' eth1 traffic.

Expected result : in both scenarios, PATH-ERROR message should be sent by MESCAL21'PCS to MESCAL11'PCS.

Test Reference : **TB_P3_FUNCT/CMES/7**

Test Purpose : Check the format of CANCEL and messages.

Procedure :

Establish a PCP communication between MESCAL11'PSCA and MESCAL21'PCS.

Configuration is the same as for scenario 1 of **TB_P3_FUNCT/CMES/2.**

Configure MESCAL12'PCS to send a CANCEL message to MESCAL11'PCS during path negotiation.

Launch a traffic analyzer that captures MESCAL11' eth1 traffic.

Expected result : CANCEL message should be as specified in introduction if this test group. PATH-COMPUTATION-ID and REQ-REFERENCE-ID must be the same as what have been used in the REQUEST message.

Test Reference : **TB_P3_FUNCT/CMES/8**

Test Purpose : Check operational behaviors when receiving REQUEST messages.

Procedure :

Launch a traffic analyzer that captures MESCAL11' eth1 traffic.

Establish a PCP session between MESCAL11'PCS and MESCAL21'PCS.

- Scenario 1: When PCP session is accepted by MESCAL12'PCS, configure MESCAL11'PCS to send a request for an LSP with a head-end-address 1.1.1.6 and tail-end-address in 2.2.2.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 0.
- Scenario 2: When PCP session is accepted by MESCAL12'PCS, configure MESCAL11'PCS to send a request for an LSP with a head-end-address 3.3.3.1 and tail-end-address in 2.2.2.1 in MC1 and a bandwidth of 0.5Mbit.
- Scenario 3: When PCP session is accepted by MESCAL12'PCS, configure MESCAL11'PCS to send a request for an LSP with a head-end-address 1.1.1.6 and tail-end-address in 2.2.2.1 in MC1 and a bandwidth of 5Mbit.

  Examine the messages exchanged between the two PCS.

Expected result :

- Scenario 1: MESCAL12's must send a ERROR-PATH message to MESCAL11'PCS with TTL expired error (value = 5)
- Scenario 2: MESCAL12's must send a ERROR-PATH message to MESCAL11'PCS (value = 3)
- Scenario 3: MESCAL12's must send a ERROR-PATH message to MESCAL11'PCS (value = 1)

14.3.2 TB_P3_FUNCT/QAGG

Only AS1, AS2 and AS3 are used to run this tests group. pSLSes are established between AS1 and AS and between AS1 and AS3 in order to extend Hard Solution Option of each domain.

- AS1offers a pSLS to AS2
  - MQC1: 0x68 bandwidth: 1Mbit
  - MQC0: 0x00 minimum bandwidth: 2Mbit maximum bandwidth: 3Mbit
- AS3 offers a pSLS to AS1
  - MQC1: 0x88 bandwidth: 1Mbit
  - MQC0: 0x00 minimum bandwidth: 2Mbit maximum bandwidth: 3Mbit

QoS capabilities of each AS are those described in 14.2.4

Test Reference : **TB_P3_FUNCT/QAGG/1**

Test Purpose : Check QoS aggregation operation

Procedure :

Launch a traffic analyzer that captures MESCAL11' eth1 and MESCAL11' eth0 traffic.

Establish a PCP session between MESCAL11'PCS and MESCAL21'PCS.
Establish a PCP session between MESCAL11'PCS and MESCAL31'PCS.

- Scenario 1: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. No QoS parameters are to be inserted in the request.
- Scenario 2: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 5Mbit. TTL is to be set to 30. No QoS parameters are to be inserted in the request.
- Scenario 3: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set minimum one-way delay to 45.
- Scenario 4: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set minimum one-way delay to 25.
- Scenario 5: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set maximum one-way delay to 80.
- Scenario 6: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set maximum one-way delay to 60.

- Scenario 7: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set average one-way delay to 40.
- Scenario 8: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set average one-way delay to 50.
- Scenario 9: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set jitter to 20.
- Scenario 10: When PCP session is established between MESCAL11'PCS and MESCAL12'PCS, configure MESCAL21'PCS to send a request for an LSP with a head-end-address 1.1.1.5 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Set jitter one-way delay to 30.

  Examine the messages exchanged between MESCAL11 and MESCAL21 and between MESCAL11 and MESCAL31.

Expected result :

- Scenario 1: MESCAL11'PCS must forward this request to MESCAL31'PCS. This request is for an LSP with a head-end-address 1.1.1.2 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. No QoS parameters are to be inserted in the request.
- Scenario 2: MESCAL11's must send a ERROR-PATH message to MESCAL12'PCS (value = 1)
- Scenario 3: MESCAL11'PCS must forward this request to MESCAL31'PCS. This request is for an LSP with a head-end-address 1.1.1.2 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Minimum one-way delay is set to a value greater than 35.
- Scenario 4: MESCAL11's must send a ERROR-PATH message to MESCAL12'PCS (value = 1)
- Scenario 5: MESCAL11'PCS must forward this request to MESCAL31'PCS. This request is for an LSP with a head-end-address 1.1.1.2 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Maximum one-way delay is set to a value less than 30.
- Scenario 6: MESCAL11's must send a ERROR-PATH message to MESCAL12'PCS (value = 1)
- Scenario 7: MESCAL11's must send a ERROR-PATH message to MESCAL12'PCS (value = 1)
- Scenario 8: MESCAL11'PCS must forward this request to MESCAL31'PCS. This request is for an LSP with a head-end-address 1.1.1.2 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Average one-way delay is set to a value less than 20.
- Scenario 9: MESCAL11's must send a ERROR-PATH message to MESCAL12'PCS (value = 1)

- Scenario 10: MESCAL11'PCS must forward this request to MESCAL31'PCS. This request is for an LSP with a head-end-address 1.1.1.2 and tail-end-address in 3.3.3.1 in MC1 and a bandwidth of 0.5Mbit. TTL is to be set to 30. Jitter is set to a value less than 10.