Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Sebastian Gilje Grøsvik

Towards a Methodology to Assess Quality of 5G Slicing

Master's thesis in Communication Technology
Supervisor: Thomas Zinner
December 2023

NTNU
Norwegian University of
Science and Technology

Sebastian Gilje Grøsvik

Towards a Methodology to Assess Quality of 5G Slicing

Master's thesis in Communication Technology
Supervisor: Thomas Zinner
December 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Title:	Towards a Methodology to Assess Quality of 5G Slicing
Student:	Grøsvik, Sebastian Gilje

Problem description:

5G networks introduces the concept of network slicing. Network slices are multiple logically separated, dynamic and customizable end-to-end networks running on a shared infrastructure. Slicing promises to optimize network data plane behavior based on vertical requirements on an end-to-end basis. It is still unclear how well the slicing concept will enable resource sharing, and to which degree isolation between different slice can be achieved and how that potentially differs between radio access, transport and core networks. Appropriate methods to assess the quality of slicing in different settings are needed. What are the relevant metrics, and how can they be assessed?

The purpose of this project is to investigate performance isolation between separate virtual networks on a common infrastructure in the transport network. More specifically the data plane mechanisms for network resource allocation. The purpose of the investigation is to verify to which degree these mechanisms are able to provide performance isolation.

Approved on:	2023-03-15
Main supervisor:	Zinner, Thomas, NTNU
Co-supervisor:	

Abstract

Network slicing is envisioned to enable multiple services and applications with different service requirements to operate in parallel on a shared network infrastructure. By utilizing software-defined networking (SDN) and network function virtualization (NFV), slicing is expected to provide dedicated end-to-end isolated networks adhering to strict service requirements and performance guarantees. As of 5G release 15, network slicing has been implemented by 3GPP in the standard. This is tempting vertical industries, that have had to use in-house or custom network solutions due to their stringent communication requirements, to consider 5G infrastructure. However, this requires a reliable and stable implementation of network slicing, an implementation able to provide key features such as performance guarantees, slice isolation, dynamic resource reallocation, and scalability.

In this thesis, we have investigated to which extent features of network slicing can be realized by data plane resource allocation mechanisms. We have investigated available mechanisms, created multiple load scenarios, and written router configurations to investigate the effectiveness of the mechanisms. We designed and assembled a physical testbed for measuring the performance isolation between two isolated virtual networks in a Cisco ISR 4221 router. The measurements show that no single mechanism is the one-fits-all solution for providing performance isolation, resource reallocation, and scalability. For the case of only two slices used in this thesis, a best-effort slice and a critical slice, strict priority yields the most optimal solution. The measurements show that the use of strict priority enables both performance isolation and resource reallocation.

Sammendrag

Skivedeling av nettverk muliggjøre at flere tjenester og applikasjoner med ulike ytelseskrav kan operere parallelt på en delt nettverksinfrastruktur. Ved å benytte programvaredefinert nettverk (SDN) og nettverksfunksjonsvirtualisering (NFV), forventes skivedeling å kunne tilby dedikerte ende-til-ende isolerte nettverk som følger strenge ytelseskrav og garantier. Fra og med 5G, utgivelse 15, har skivedeling av nettverk blitt innført av 3GPP i 5G standarden. Dette åpner for at vertikale bransjer som har vært nødt til å bruke interne eller tilpassede nettverksløsninger på grunn av strenge kommunikasjonskrav nå kan se til 5G. Dette krever imidlertid en pålitelig og stabil implementering av nettverkskiver som kan tilby nøkkelfunksjoner som ytelsesgarantier, skiveisolasjon, dynamisk ressursomfordeling og skalerbarhet.

I denne avhandlingen har vi undersøkt i hvilken grad mekanismer for skivedeling av nettverk kan realiseres ved bruk av dataplanbaserte ressursallokeringsmekanismer. Vi har undersøkt tilgjengelige mekanismer, opprettet flere belastningsscenarioer og skrevet ruterkonfigurasjoner for å undersøke effektiviteten til mekanismene. Vi designet og satt sammen et fysisk testmiljø for å måle graden av isolasjonen mellom to isolerte virtuelle nettverk i en Cisco ISR 4221-ruter. Målingene viser at ingen enkelt mekanisme er den universelle løsningen for å gi ytelsesisolasjon, ressursomfordeling og skalerbarhet. For tilfellet med bare to skiver brukt i denne avhandlingen, en skive for alminnelig datatrafikk og en skive for kritisk datatrafikk, gir streng prioritet den mest optimale løsningen. Målingene viser at bruk av prioritering muliggjør både ytelsesisolasjon og ressursomfordeling.

Preface

This thesis was written to conclude the two-year master's degree in Communication Technology class of 2023 at the Norwegian University of Science and Technology (NTNU) in Trondheim.

I want to thank my supervisor Thomas Zinner for excellent feedback and guidance, and I want to thank Stanislav Lange for his valuable insights.

Contents

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**

1.1 Motivation 1
1.2 Scope 2
1.3 Research questions 3
1.4 Research goals 3
1.5 Outline 4
1.6 Sustainability 4

2 Background **5**

2.1 Quality-of-Service 5
2.2 Network slicing 6
2.2.1 Definition of network slicing 6
2.2.2 Differentiating network traffic 8
2.2.3 Slice performance requirements 8
2.2.4 Slice isolation 9
2.2.5 Intra-slice characteristics 10
2.2.6 Inter-slice characteristics 10
2.3 Resource allocation mechanisms 11
2.3.1 Token-bucket based maximum throughput 11
2.3.2 Policing 11
2.3.3 Shaping 12
2.3.4 Throughput guarantees 12
2.3.5 Strict priority 13
2.3.6 Probabilistic dropping 13
2.3.7 Tail drop 14
2.4 Related Work 15
2.4.1 Network resource allocation algorithms 15
2.4.2 Isolation in network slicing 16

2.5 Identified research gap 18

3 Methodology **21**
3.1 Testbed setup 21
3.1.1 Hardware 21
3.1.2 Router: Cisco ISR4221/K9 21
3.1.3 Dell Precision 3630 24
3.1.4 Docker 24
3.2 Data plane policies 26
3.2.1 Configuration goals 26
3.2.2 Tail drop 30
3.3 Data collection 31
3.4 Data analysis 32
3.4.1 Throughput 33
3.4.2 Packet loss 33
3.4.3 Latency 33
3.5 Validation of the testbed 33
3.6 Limitations 35

4 Results **37**
4.1 Critical traffic 37
4.1.1 Throughput 37
4.1.2 Packet loss 37
4.1.3 Latency 38
4.2 Best-effort traffic 40
4.2.1 Throughput 41
4.2.2 Packet loss 41
4.2.3 Latency 42
4.3 Bandwidth investigation 43
4.3.1 Latency trade-off 43
4.4 WRED and latency measurements 45
4.5 Enforcing allocation for both traffic classes 46
4.6 Measurement summary 46

5 Discussion **49**
5.1 Performance isolation 49
5.2 Transient and steady state 50
5.3 Scalability 50
5.4 Shared buffer 51
5.5 Net neutrality 52
5.6 UDP and TCP 52

6 Conclusion **55**
6.1 Future work 56

References **57**

List of Figures

2.1 5G use cases and their requirements [FPEM17]. 6
2.2 Soft/hard slice spectrum. 9
2.3 Expected traffic throughput for shaping and policing [Cis23]. 12
2.4 A model of a weighted queue with three traffic classes. 13
2.5 Priority queuing model [KR00]. 14
2.6 Dropping probability for two traffic classes [Net]. 15
2.7 Approximate Hierarchical Allocation of Bandwidth (AHAB) [MCR23]. . 16
2.8 Virtual queues and slicing in the data plane [HPD+21]. 18

3.1 A logical model of the testbed, detailing the interface connections of the testbed. 22
3.2 A model of the internals of the Cisco ISR4221. 23
3.3 Time series comparison of generated load for scenarios. Scenario 6: Number of packets (1) and Mbps (2). Scenario 3: Number of packets (3) and Mbps (4). Scenario 0: Number of packets (5) and Mbps (6). 25
3.4 Example of the text output from an iperf3 server. 32
3.5 Times series of critical latency (1), best-effort latency (2), IAT of generated critical traffic (3), and IAT of critical traffic received by the traffic sink. The delay spikes in the (1) and (2) can be seen in IATs at the sink, but not at the generator. 34
3.6 CDF comparison of generated packet inter-arrival time. 35

4.1 Critical traffic throughput. 38
4.2 Critical traffic packet loss. 38
4.3 Critical traffic latency. 39
4.4 Critical traffic latency, zoomed into the interval of 0 to 1 ms. 39
4.5 CDFs of critical latency measurements. 40
4.6 Best-effort traffic throughput. 41
4.7 Best-effort traffic packet loss. 42
4.8 Best-effort traffic latency. 42
4.9 Best-effort traffic latency, zoomed into the interval of 0 to 4 ms. 43
4.10 Bandwidth mechanism investigation using updated policies: Critical traffic latency, zoomed into the interval of 0 to 1 ms. 44

4.11 Critical and best-effort latency comparison. 44
4.12 Critical and best-effort latency comparison zoomed in. 45
4.13 Critical traffic latency for a bandwidth mechanism based WRED configuration. 45
4.14 Critical traffic latency with allocation on both classes. 46

List of Tables

1.1 A comparison of 4G requirements as presented in 3GPP TR 25.913, and 5G requirements as presented in IMT Vision-Framework. 2

3.1 The loads used in the different measurement scenarios. 24

4.1 Summary of the investigated slicing attributes. 47

Chapter 1
Introduction

1.1 Motivation

The development of mobile network technology is an important enabler for emerging technologies and future applications. We have seen how increased network data rates have led to the convergence of services such as voice, data, and video into a single network infrastructure. The fifth generation of mobile networks is continuing the trend of offering users even faster download and upload speeds [SFVS20]. The general vision for how 5G performance will compare to 4G can be seen in Table 1.1. 5G will offer peak data rates in the gigabit per second range. As with previous generations, latency reduction is being improved upon. User capacity is increased, while more efficient frequency usage enables faster data rates in a larger coverage area. All the while the equipment is set to be 100 times more energy efficient [ATS+18].

Today, mobile network development aims to enable a set of new applications such as autonomous vehicles, smart cities, industrial automation, augmented and virtual reality, and real-time remote-controlled robotics. These applications demand a new standard of high-quality communications channels, and support for various strict performance requirements. The 5G network standard approaches these verticals with a host of new technologies across all domains to enable the smartest network generation yet [3GP19]. The radio access, core, and transport networks are being upgraded. In the transport and core technologies such as Software-Defined Networking (SDN), Network Function Virtualization (NFV), Mobile Edge Computing (MEC) and network slicing are being leveraged to revolutionize data communications.

Network slicing is a networking solution that enables multiple isolated end-to-end virtual networks with custom service attributes such as isolation, throughput, delay, and reliability to operate on a common physical infrastructure. It enables the operators to deliver the differentiated and optimized network services required to support the new era of devices and applications for their users[NGM15]. Network slicing is introduced as a feature that enables the network operator to take ownership of the network resources and conduct the allocation on behalf of all its users in an optimized manner. However, this means that the operator must implement mechanisms that enable effective and fair network resource allocation in the network slicing schemes.

Resource allocation will have to be handled both internally between users in a single slice, referred to as intra-slice behavior, and between different slices, referred to as inter-slice behavior. For intra-slice behavior, the operator should first implement mechanisms that fairly split the slice resources across the users. Secondly, surplus capacity should be allocated to users with demands beyond their share. Inter-slice behavior is complex as the resources firstly need to be split based on slice requirements and secondly, surplus capacity should be fairly allocated to slices that have additional demand beyond their allocated share[DSM+18]. A challenge of 5G network slicing is balancing the reallocation of unused resources from one slice to another, while at the same time making sure the slices are isolated from one another.

4G vs 5G Key Performance Indicators (KPI)		
Requirement	4G (3GPP TR 25.913)	5G (IMT Vision-Framework)
Peak data rates	100 Mbps DL/ 50 Mbps UL	10 Gbps
Latency	Reduced latency	Low latency i.e. 1ms over-the-air
Mobility	Up to 500 km/h	Service continuity with 500 km/h
User density	-	10 Mbps/m2
Spectral efficiency	Improved spectrum efficiency e.g. 2-4 x Rel 6	Three times higher than 4G
Area coverage	Increased bitrates at the edge of cells	100 Mbps data rates for wide area coverage
Connection density	-	1 000 000 devices per km2
Energy efficiency	Reduced capital and operational expenditure	100 times more energy efficient

Table 1.1: A comparison of 4G requirements as presented in 3GPP TR 25.913, and 5G requirements as presented in IMT Vision-Framework.

1.2 Scope

To reduce the complexity of a network slicing implementation this thesis will focus on the slicing of network resources, particularly in the transport network domain. Furthermore, this investigation will primarily focus on the inter-slice aspect of network slicing, where the quality of slicing is seen as the ability to deliver network slices according to a defined service-level agreement (SLA). The scope of this master thesis is to evaluate data plane resource allocation mechanisms in a network slicing context. The goal is to investigate how data plane mechanisms can be used to achieve performance guarantees, by showcasing the mechanisms, observing how they affect traffic, and which degree of inter-slice performance isolation they can achieve. In this project Cisco hardware and Cisco IOS will be used, meaning that the project will only look at Cisco implementations of the data plane mechanisms. The investigation will study and compare the mechanisms in practice. To this end, this project aims to study data plane resource allocation mechanisms, i.e. traffic policing, shaping, prioritization, probabilistic dropping, and throughput allocation, in the context of slicing quality. To investigate whether these mechanisms can provide the necessary service guarantees to uphold slice service agreements. The goal is to evaluate the effectiveness of these mechanisms to achieve desired slice behavior, such as performance isolation and resource reallocation.

1.3 Research questions

This project aims to explore the topic of resource sharing in 5G slicing on transport networks. The overall goal of the project is to contribute to the state-of-the-art research of network slicing through technical implementations and collecting measurement data.

1. Which data plane resource allocation mechanisms are available in Cisco IOS?
2. How can the performance of resource allocation mechanisms be evaluated?
3. Can these mechanisms provide performance isolation and resource reallocation in a scalable manner to network slices?

1.4 Research goals

The task of answering the research questions has been divided into sub goals which will guide the project along. The following research goals have been defined for this project.

1. Assess mechanisms for conducting resource allocation in network slices.
2. Implement a testbed to investigate the resource allocation mechanisms.
3. Define workloads that cause network congestion to provoke the behavior of the resource allocation mechanisms.

4. Evaluate whether the mechanisms are capable of maintaining performance isolation and conducting resource reallocation for proposed workloads.

1.5 Outline

The thesis is structured as follows. In Chapter 2 the relevant background, technologies, and related work are summarized. Chapter 3, describes the method for implementing the testbed and conducting the evaluation. After that, Chapter 4 highlights the results of the evaluation and sets them into the context of sharing resources between critical and non-critical services. This is followed by a discussion of the limitations and potential next steps in Chapter 5. Chapter 6 concludes the thesis.

1.6 Sustainability

The technology of 5G network slicing is an enabler for sustainability efforts in telecommunications through decreasing energy consumption and increasing resource efficiency. Furthermore, by providing tailored solutions to meet the requirements of verticals, network slicing can remove the need to build homegrown industry-specific solutions. Through hosting multiple network slices, an operator can satisfy different application requirements on one common infrastructure which removes the need for custom hardware and parallel physical infrastructures. Reducing or entirely removing the need for physical isolation of network and computation resources can have a significant impact on the need for physical equipment. Furthermore, network slicing has the potential to enable new green initiatives by providing enhanced and cheaper connectivity for smart solutions and sustainability monitoring.

Chapter 2 Background

This chapter will introduce network slicing and present relevant definitions related to slicing. Furthermore, the chapter will begin to answer the first research goal by presenting theory and research on resource allocation mechanisms.

2.1 Quality-of-Service

The two most basic methods of network switching are the connection-oriented circuit switching and the connectionless packet switching. Circuit switching is a method in which two nodes establish a dedicated communication channel throughout the network. The user receives a guaranteed performance, a fixed throughput, uniform latency, and no packet loss. However, the dedicated line can not be accessed by other users, meaning that idle capacity is left unused. Furthermore, if resources on a link are fully booked, new connections must be blocked.

The second method, packet switching, is a method in which communication between two nodes is divided into individually routed packets. These packets share the network resources in a best-effort manner, by using the resources as needed, while idle capacity can be accessed by everyone. This increases the potential for network utilization and enables more users to access the network. However, guaranteeing resources to specific users is quite challenging, throughput is dependent on available capacity, latency varies depending on the queuing delay in the network, and packet loss can occur in cases of network congestion.

To balance the pros and cons of circuit and packet-switched networks Quality-of-Service (QoS) is introduced. QoS refers to a set of technologies and protocols used in packet-switched networks to provide service guarantees for certain users or traffic types. QoS can enable the network operator to provide performance guarantees to a user, without sacrificing the benefits of packet switching.

2.2 Network slicing

Network slicing is a technology introduced in the early 2000s and later defined in the 3GPP 5G standards. Slicing aims to provide an optimal solution to the trade-off between network utilization and performance guarantees. Slicing introduces a solution to the heterogeneity of 5G mobile network use cases, as seen in Figure 2.1, other examples include Internet of Things (IoT) type communications, and industry-specific communications. By slicing the network into slices of homogeneous communication types, performance and security guarantees can be realized for specific services on a shared infrastructure. Network slicing is envisioned to enable custom logical networks for a specific use case or service, with the intent of providing an optimal network service.

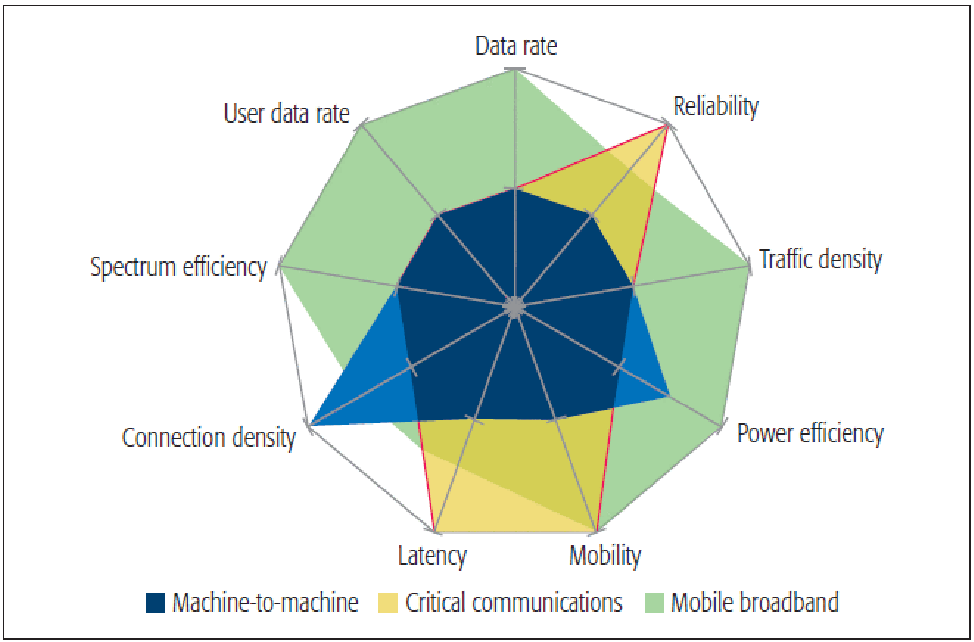


Figure 2.1: 5G use cases and their requirements [FPEM17].

2.2.1 Definition of network slicing

Network slicing is complex, but by studying the definitions we can distill the concept down to its core. In [3GP23b] 3GPP uses the following definition of a network slice: *"Network slice: a set of network functions and corresponding resources necessary to provide the required telecommunication services and network capabilities."* This is a vague definition. It does not provide any specific details as to what a slice is and only specifies that a slice is composed of some network functions and resources, which are needed to provide some services. It does not provide much context as to what these network resources or functions which make up a slice are.

In a later release [3GP23a] 3GPP uses the following definition of network slicing, which is more specific: *"Network slicing is a key feature for 5G. Network slicing is a paradigm where logical networks/partitions are created, with appropriate isolation, resources and optimized topology to serve a purpose or service category (e.g. use case/traffic category, or for MNO internal reasons) or customers (logical system created "on demand")."* This definition introduces slicing as logical networks with some degree of isolation, which have been assigned resources to serve a defined purpose or provide some services. The definition also includes that the slice is created "on demand".

In the research literature, more detailed definitions can be found. In [MCR23] the following definition is used: *"Network slicing is a network feature that allows an operator to divide its network resources into many virtualized networks. Slicing enables operators to rapidly create new service offerings for different markets, while achieving performance isolation and quality-of-service guarantees between different slices. To support slicing, the network needs to implement both intra-slice fairness where different users within the same slice gets a fair share of the slice's bandwidth, as well as inter-slice fairness where each slice gets its share of bandwidth proportional to its specified weight. Meanwhile, the idle capacity from underutilized slices must also be fairly distributed to other over-subscribed slices."* This definition clearly states performance isolation as a requirement of slicing, it specifies that quality-of-service (QoS) guarantees need to be implemented, and the reallocation of available network resources between slices.

An even more specific definition can be found in [MML+22]: *"... the possibility of using the same physical network infrastructure to serve different tenants with different SLA requirements. Two main requirements for the effective use of slicing into networks are isolation of users and deterministic performance guarantees."* In this definition, it is specified that a single shared physical network infrastructure is at the core of slicing. Isolation and deterministic performance guarantees are also included in the definition.

To summarize, network slicing is defined as the process of dividing a single physical network infrastructure into multiple logical partitions in an on-demand and dynamic manner. A network slice is defined to be a single isolated logical partition that is assigned certain deterministic performance guarantees and network functions. The degree of dynamicity slicing should provide is not specified and still remains unclear, in [ATS+18] the SDN controller is the role that takes care of the dynamic network management, but without any requirements to response time.

2.2.2 Differentiating network traffic

To ensure proper mapping of packets to their designated network slice, a network operator must deploy a mapping algorithm. In this section, we present two different approaches.

In [MCR23], the proposed method uses a slice lookup table and the flow representation as an identifier for a user. The flow representation aggregates packets that share their 5-tuple attributes, source IP, destination IP, source port, destination port, and protocol type. In this case, the flow identifies a user, which is mapped to their respective slice. Vlan matching is used in [GSV21], vlan matching follows the same principle, but adds an additional lookup: A flow is mapped to a vlan, and one or more vlans are mapped to a slice.

The second method is service-based classification [SBKT16], which assigns slices based on the specific service the traffic belongs to. Meaning that a single device is allowed to connect to multiple slices for different services. For example, traffic associated with video streaming, voice communication, or IoT devices would be categorized into their respective slices even if it was all coming from a single user equipment. This service-based mapping strategy helps ensure that the traffic is directed to the appropriate slice based on its intended service.

These two methods illustrate that traffic classification is one of the open implementation questions of network slicing, and one that will need to be addressed in a slice implementation.

2.2.3 Slice performance requirements

SLAs and performance requirements are an integral part of network slicing, but which key performance indicators (KPIs) are relevant for slicing? 3GPP [3GP23b] defines the performance requirements and QoS requirements for 5G using the following indicators: Latency, mobility, availability, reliability, and data rates. Reliability is defined in the context of network layer packet transmissions, the percentage value of the packets successfully delivered to a given system entity within the time constraint required by the targeted service out of all the packets transmitted. Since RAN is scoped out of the project mobility will not be considered. That leaves the following KPIs: Throughput, latency, reliability, and availability.

To adhere to the performance guarantees the network operators must implement some form of QoS resource allocation scheme in their network. This resource allocation must be handled in the data plane and the allocation decisions must have a minimal complexity to avoid introducing processing delays. The data plane only serves a limited number of such resource allocation mechanisms.

2.2.4 Slice isolation

Slice isolation is mentioned as one of the two main requirements of network slicing. However, the degree of isolation required for slicing is not specified further than "appropriate" in the presented definitions. One approach to isolation, presented in [HMM+19], is to consider the degree of isolation as a choice of either soft or hard slicing. Soft slicing provides no guarantees to the network during heavy loads, while hard slicing can provide service guarantees through a subdivision of the network. In [ATS+18], three methods for achieving hard isolation are presented. The first is using different physical resources, the second is separation through virtualization of shared physical resources and the third is sharing a resource with an access policy which defines rights for each tenant. 3GPP envisions the degree of isolation depending on the service type (eMBB, URLLC, mIoT). Based on the proposed requirements for critical communications and URRLC slices a hard isolation approach is to be preferred to provide the deterministic service requirements which have been described [ATS+18].

Another approach to slice isolation is to consider isolation as a spectrum, from complete soft-isolation slices on one hand to hard-isolation slices on the other, see Figure 2.2, where the hardest-isolation refers to complete physical separation. While the softest refers to no isolation at all. By considering isolation as a spectrum we can place different isolation functionalities to create a tool for defining a specific isolation level. Using this spectrum approach we can more easily find an appropriate isolation level for a specific use case, without having to resort to the extremes of soft and hard isolation. For this project, we investigate if performance isolation can be

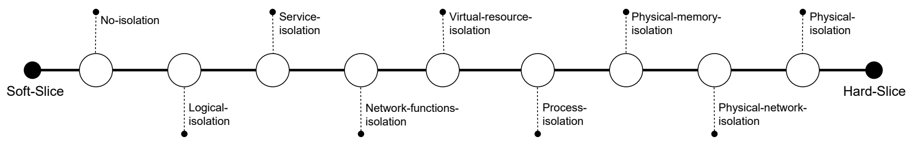

No-isolation
Service-isolation
Virtual-resource-isolation
Physical-memory-isolation
Physical-isolation
Soft-Slice
Hard-Slice
Logical-isolation
Network-functions-isolation
Process-isolation
Physical-network-isolation

Figure 2.2: Soft/hard slice spectrum.

achieved using network virtualization. Performance isolation implies that the SLA KPIs should be met per slice, regardless of the behavior of other slices [HPD+21]. Performance isolation will in this investigation be determined by observing if the service of one slice is affected by the congestion of another.

2.2.5 Intra-slice characteristics

Creating a policy for network slice configurations has two aspects, the first, is the policy for how different slices should behave in relation to each other, referred to as inter-slice behavior. The second is the policy for how users on a specific slice should be treated in relation to each other, referred to as intra-slice behavior.

In [MCR23] intra-slice behavior is characterized by fairness among the users belonging to the same slice. The slice takes care of the service guarantees, which means that the optimal intra-slice behavior is to equally share the resources provided to the slice. If the total bandwidth allocated to the slice is known, then the per-user capacity is equal to the total divided by the amount of users. However, as discussed in [MCR23] a challenge with intra-slice data rate enforcement is the hardware limitations to tracking an accurate state for every user. Depending on the centrality of a networking node the number of users it needs to track in a slice can go from thousands to millions. Scalable enforcement of per-user limits needs to be achieved in a stateless manner.

2.2.6 Inter-slice characteristics

Inter-slice behavior aims to enable efficient and harmonious inter-slice coordination, ensuring that different slices can operate effectively in parallel while meeting their specific service requirements. Inter-slice relations are characterized by the deterministic service guarantee-based resource allocation. Different network slices will have different service agreements that define performance targets and service needs. These service agreements will set the precedence for how network resources should be prioritized and allocated [MCR23][ATS+18].

Inter-slice isolation is required to be on the hard side of the isolation spectrum, but the degree of how hard ideal inter-slice isolation should be is unclear. The slices need at least a minimum of resource isolation to adhere to service guarantees. At the same time, balanced isolation and sharing mechanisms need to be in place to prevent adverse interference between slices and maintain performance. Proper inter-slice coordination and orchestration are needed to enable efficient cross-slice interaction.

To deal with QoS guarantees on an inter-slice basis, the network operators need to implement resource allocation mechanisms that ensure differentiation, allocation, and coordination between the slices. The mechanisms need to be configured based on the service targets of the slices the network supports. To avoid introducing delay into the network, the mechanisms must be implemented with minimal complexity in the data plane.

For throughput, guarantee mechanisms that provide minimum and maximum throughput can be utilized. Providing slices a minimum guaranteed throughput based on a minimum service target, while also enforcing a maximum limit to prevent slices from congesting the network. To guarantee a certain latency the operator has to avoid buffering delay or set a maximum threshold for the time spent in queue. As propagation, transmission, and processing latency are static, buffering delay introduces the greatest variation and therefore challenges to latency guarantees. One way of preventing buffering delay is to prevent queuing in the network. This means dropping low-priority packets that would otherwise be queued. Next, is by utilizing multiple queues with strict priority. By giving one slice strict priority over the rest, packets from this prioritized slice will always be served first and won't be affected by queuing in the other slices.

2.3 Resource allocation mechanisms

In this section data plane resource allocation mechanisms will be presented. The mechanisms presented are all implemented on Cisco routers and can operate at line rate [Cis17]. These mechanisms will be studied with the intent of finding configurations that enable service guarantees and performance isolation in a slicing context.

2.3.1 Token-bucket based maximum throughput

The token bucket is a queue scheduling algorithm used to control the rate of traffic in a queue [Jun23b][Cis23][Cisb]. Traffic is modeled by a stream of tokens that are generated at a fixed rate. The tokens are stored in a bucket with a fixed capacity. When packets arrive in the queue they consume a certain number of tokens from the bucket to be forwarded. If there are not enough tokens the packet can either be dropped, or stored until the token number is sufficient to serve the packet.

2.3.2 Policing

One token-bucket-based mechanism is policing [Jun23a][Cisb]. If policing is enforced the interface drops packets that violate the token-bucket. If traffic conforms to the police rate the interface transmits the data. Using a two-token bucket approach traffic can be categorized into three: conforming, exceeding, and violating. Conforming traffic is transmitted, exceeding traffic is reassigned to a lower priority class, and violating traffic is dropped. As traffic policing drops violating traffic it does not require a queue and can be applied to both incoming and outgoing traffic on an interface. By applying traffic policing to a network interface queuing delays can be avoided. However, in the case of transport protocols with guaranteed delivery, such as TCP, traffic policing adds additional strain on the network as the dropped packets will be retransmitted.

2.3.3 Shaping

Traffic shaping is a token bucket-based mechanism used to limit the data rate on a link. If traffic conforms to the shaping rate the interface transmits the data. However, if the traffic exceeds the shaping rate the traffic is buffered in a queue until the traffic conforms to the shaping rate at which point it is transmitted. The shaping mechanism buffers packets and adds delays to conform to the desired rate. Unlike a policer, a traffic shaper requires a queue and can therefore only be applied to packets leaving an interface. The difference in throughput behavior is visualized in Figure 2.3. Queuing of packets during periods of congestion introduces latency in the network, making this approach unsuitable for real-time data. On the other hand, for applications without time sensitivity, queuing can prevent packet loss, leading to fewer packet retransmissions and less strain on the network.

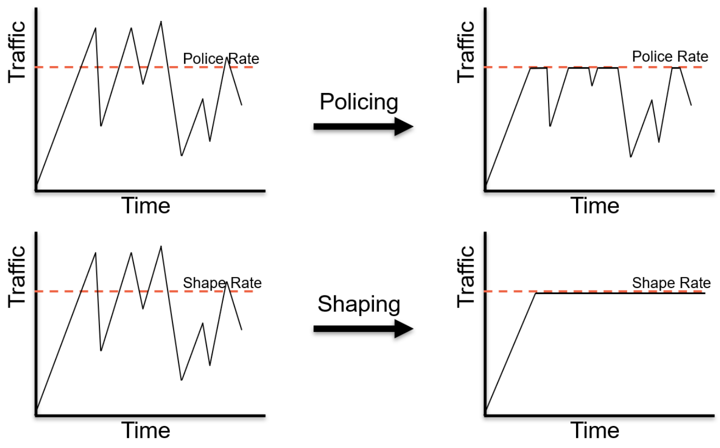


Figure 2.3: Expected traffic throughput for shaping and policing [Cis23].

2.3.4 Throughput guarantees

Minimum throughput guarantees can be used to ensure that a network user receives a minimum throughput in cases of network congestion. Throughput reservations can be configured to ensure these minimum guarantees [Ris01]. Scheduling algorithms such as weighted fair queuing (WFQ), weighted round robin (WRR), or class-based queuing (CBQ) can be used to ensure that the throughput reservation for a specific traffic class is assigned necessary resources, Figure 2.4 illustrates the general model of weighted queuing.

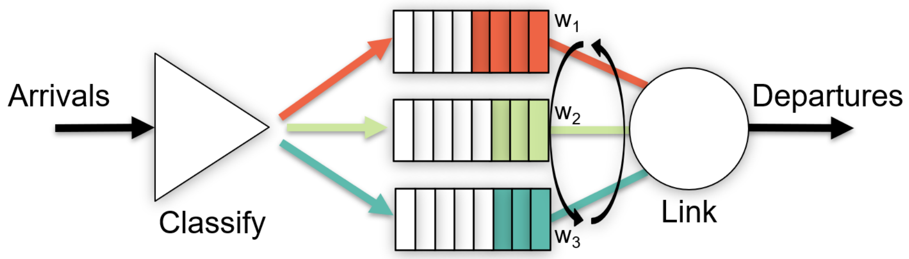


Figure 2.4: A model of a weighted queue with three traffic classes.

2.3.5 Strict priority

Strict priority is employed to allocate various data traffic classes to specific priorities. Packets with high priority are forwarded with minimal delay [Jun21b]. By assigning priorities using a strict priority mechanism it ensures that the traffic is placed in a priority queue, see Figure 2.5. The high-priority queue has precedence over the low-priority, meaning that this queue will be served until it is empty. Strict priority can enable the configuration of a delay-sensitive service such as critical communications [Ris01]. Traffic with higher priority is given precedence over lower-priority traffic, allowing it to be processed and transmitted with minimal added delay. The use of strict priority enables the network to prioritize critical or time-sensitive traffic, such as real-time voice or video, ensuring optimal performance and quality of service for these priority traffic classes. However, poorly implemented priority queuing can introduce bandwidth starvation for other classes [Lee14]. Head-of-the-line blocking occurs when high-priority packets get blocked by a queue of lower-priority packets. Strict priority queuing can include packet preemption, meaning that the processing of a low-priority packet is interrupted if a higher-priority packet arrives, providing immediate service to the high-priority packet. Using strict priority with preemption can ensure minimal delay for high-priority traffic.

2.3.6 Probabilistic dropping

Probabilistic packet dropping is a technique used in network congestion control to manage the flow of network traffic. The interface selectively drops packets from the network when queuing occurs, rather than buffering all packets and allowing the network to become congested [FJ93].

With probabilistic packet dropping, each network device has a certain probability of dropping a packet when queuing occurs [Jun22]. This probability is primarily based on a queue depth and a dropping probability but can be based on various

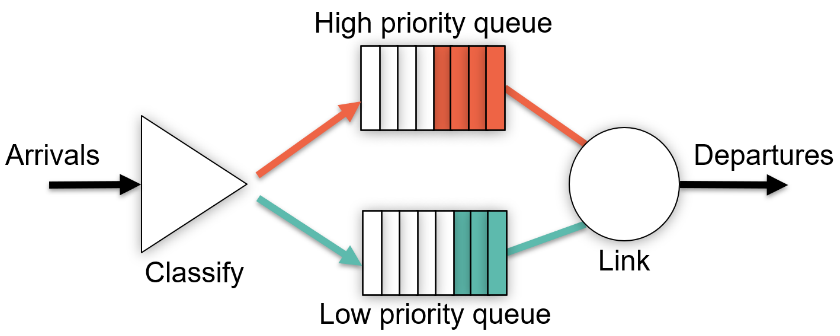


Figure 2.5: Priority queuing model [KR00].

factors, such as the level of congestion in the network, the priority of the packets being transmitted, or the quality of service (QoS) requirements of the traffic. Figure 2.6 shows what this dropping probability can look like for two different traffic classes.

The probability of packet drops is typically calculated based on a random number generator or a weighted algorithm that takes into account the above-mentioned factors. For example, a router might drop a packet with a 50 percent probability if the network is congested, or drop a low-priority packet with a higher probability than a high-priority packet.

By selectively dropping packets, probabilistic packet dropping can help to reduce congestion and improve overall network performance. However, it can also result in high packet loss, which can impact the quality of experience for certain types of applications. Therefore, it is important to carefully tune the parameters of the probabilistic packet-dropping algorithm to balance between network performance and packet loss.

2.3.7 Tail drop

The tail drop mechanism is a simple queue management technique that only considers a full queue condition. When the condition is reached the packets arriving at the end (tail) of the queue are dropped with 100 percent probability [Jun21a]. Meaning that the most recently arrived packets are the ones dropped. Tail drop can lead to "TCP synchronization" [HHTC05]. Which occurs when multiple TCP connections sharing the same congested link experience packet loss at the same time. Leading to a simultaneous and synchronized increase in the sending rate.

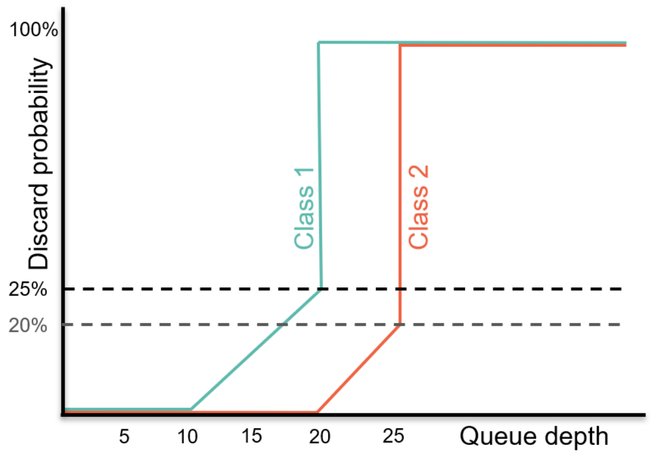


Figure 2.6: Dropping probability for two traffic classes [Net].

2.4 Related Work

Published research on the topic of network slicing has been increasing significantly in recent years. The number of published papers found in the network slicing category has increased from the low hundreds in 2015 to passing a thousand per year in the last 5 years. Several recent works present different approaches to SDN algorithms for transport network slicing. In this section, we present relevant studies on network slicing, slice isolation, and network isolation.

2.4.1 Network resource allocation algorithms

In recent years a number of advanced novel resource allocation algorithms have been proposed. The algorithms are dependent on a programmable data plane, such as P4-enabled switches. Studying the components of these algorithms can provide insights into proposed solutions for achieving slice isolation.

Approximate hierarchical allocation of bandwidth (AHAB), seen in Figure 2.7, is a novel data plane-based slicing algorithm proposed in [MCR23]. AHAB supports enforcement of a per-user bandwidth limit without storing a per-user state. Figure 2.7 illustrates how AHAB functions, first a slice lookup table is used to identify which slice a flow belongs to. Second, by conducting approximate arithmetic operations in the data plane using a TCAM lookup table the per-user bandwidth limit can be approximated. Next, for intra-slice bandwidth limit enforcement, probabilistic dropping is applied using the assigned bandwidth limit. This approach requires the per-user sending rate and the bandwidth limit and can be performed using only a single queue. For tracking the users sending rate AHAB uses an approximate data structure. For inter-slice fairness, the control plane reads the slice demands from the data plane and calculates the new per-slice allocation capacity. This new allocation is then written to the data plane. AHAB running on an Intel Tofino Wedge32-X switch is reported to support up to 16000 slices and 2000 downstream base stations.

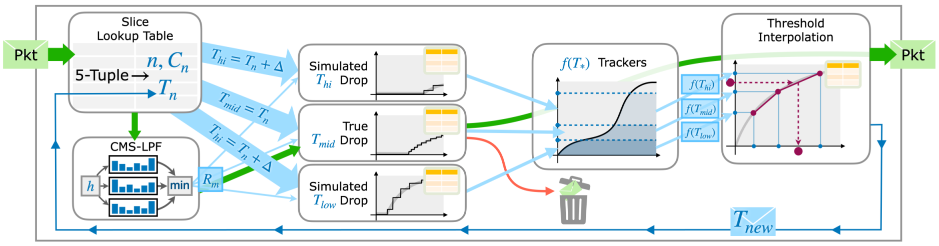


Figure 2.7: Approximate Hierarchical Allocation of Bandwidth (AHAB) [MCR23].

In [YWB+21], Hierarchical core-stateless fair queuing (HCSFQ) is presented. HCSFQ provides hierarchical fair queuing on hardware switches at line rate, while it does not require per-flow states at core switches. HCSFQ has three major components, a rate estimator, probabilistic dropping, and a fair share estimator. The edge nodes use the estimator to tag packets with a flow resource allocation, which other network nodes use to distribute resources fairly throughout the network. Probabilistic dropping is used for limit enforcement in cases of congestion.

A P4 programmable traffic manager is proposed in [HPD+21]. The manager is comprised of a traffic classifier and virtual queue-based traffic management, see Figure 2.8. The classifier uses a match-action table to perform per-flow slice classification, the criteria are not described, but using a VLAN tag is mentioned as an example. The virtual queues are programmed and assigned per slice which enables differentiation of services. Rate limit, priority, and queue management policies are applied per slice. Taildrop and explicit congestion notification (ECN) are the mechanisms used to limit congestion.

2.4.2 Isolation in network slicing

The paper [GOH+20] presents a conceptual approach to isolation for network slices. In the paper, Gonzalez et al. define isolation *"as the property that services in a slice may operate without any direct or indirect influence from activities in other slices, and unsolicited influence of the infrastructure providers."* Isolation is further defined as having three dimensions: Performance, security, and dependability. Performance isolation ensures that slices maintain consistent and predictable performance levels without interference. Security isolation aims to prevent adversarial attacks and ensure confidentiality, integrity, and availability. Dependability isolation involves ensuring resilience and redundancy to prevent failure propagation and misconfigurations.

In [KEKG22] an investigation into security isolation is conducted. The authors propose using over-the-top virtual private network (VPN) tunnels to provide slice isolation. This implementation uses an automated VPN-as-a-service-based approach. The implementation is a contribution towards hard inter-slice security isolation, but does not consider performance isolation or resource allocation. The results show that an over-the-top VPN solution has a significant negative impact on performance, while still being within the limits of performance requirements.

Two examples of performance isolation investigations can be found in the papers [SVK+20][GSV21]. Both papers present SDN-based investigations conducted in an emulated network environment. In [SVK+20] the proposed solution uses police-based inter-slice traffic enforcement, configured with OpenFlow. The investigation results are very coarse-grained as the measurements are conducted on a per-second level. Furthermore, the implemented configuration is based on strict static throughput thresholds and does not take any resource sharing into account. The network utilization and packet loss are not presented. However, the measurements show that a degree of performance isolation is achieved in terms of throughput, while for latency the measurements look inconclusive. Resource sharing is not possible with this configuration. Paper [GSV21] implements three slices in the network with police-based traffic enforcement, but this time takes resource sharing into account. The paper proposes a dynamic solution, where each slice is assigned a per-port elastic throughput threshold, which is periodically updated based on port utilization. The solution uses a hierarchical token-bucket approach consisting of multiple policers. This solution first takes the per-slice static thresholds seen in [SVK+20] into account, but instead of dropping exceeding traffic, the packets are remarked with a lower priority. Remarked packets are then sent to the next policer, where the total port capacity is considered. If there is available capacity on the port the remarked packets are forwarded, however, if the full capacity is used then the remarked packets are dropped. This approach gives each slice a certain minimum guaranteed throughput, while evenly sharing any unused capacity if needed. The paper further presents how latency isolation can be improved by employing a separate priority queue for unmarked packets. However, the measurements are not detailed enough to conclude that the method achieves latency isolation.

In [NLG+15] a virtual network isolation investigation is conducted through measurements in a physical testbed. The investigation is carried out on the Pronto 3290 switch, an OpenFlow-enabled SDN-switch. The goal of the investigation is to verify whether the division of a shared physical queue into two virtual queues can provide isolation between virtual networks. An evaluation of performance isolation is conducted in a testbed setup with two traffic generators, where the flows from each generator are assigned a virtual queue. During periods of congestion on the outgoing switchport, the measurements show that isolation between the virtual networks is violated. The violation is demonstrated in scenarios with flow control both enabled and disabled on the sending devices. The measurements identify overall load and rate guarantees to have a significant impact on the degree of congestion. This investigation does not attempt to schedule or divide the link resources between the two virtual networks, it only allocates space in the shared output queue.

Flex Ethernet (FlexE) based isolation is proposed as a solution for achieving hard isolation in network slicing in [HMM+19]. FlexE uses a TDMA-like resource allocation. A FlexE-based algorithm to be implemented in an SDN controller for slice resource reservation is presented and demonstrated. The demonstration showcases guaranteed hard isolation between the slices. However, the implementation is quite limited in terms of functionality. It is based on reserved strict allocations, without any form of automated readjustment mechanism. Furthermore, the strictness of the reservations does not enable the sharing of spare capacity.

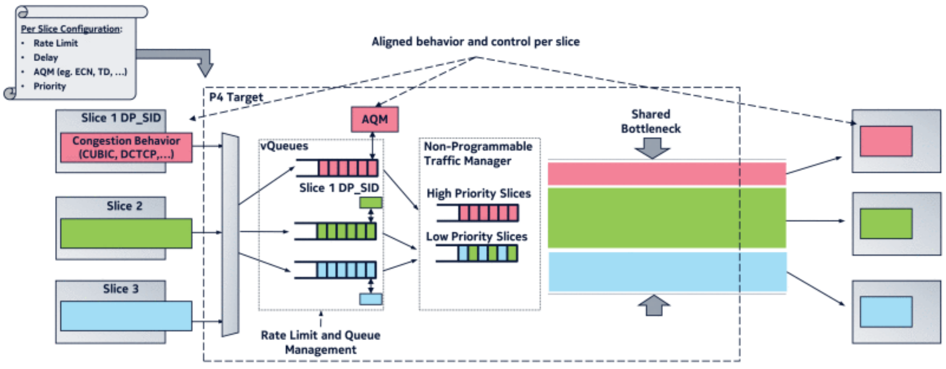


Figure 2.8: Virtual queues and slicing in the data plane [HPD+21].

2.5 Identified research gap

We have seen that novel research into the field of network slicing is software-defined and based on programmable hardware, and in multiple of the presented studies, the researchers seek to address all the challenges of slicing at once with one complex algorithm. This project aims at taking a step back and applying a narrow research scope for network slicing, to remove complexity by focusing on the resource allocation challenge of inter-slice isolation. By combining research on the topic of performance isolation in network slicing with the topic of link capacity allocation techniques in the data plane. This project seeks to evaluate performance isolation in existing data plane resource allocation mechanisms in the context of inter-slice characteristics.

Chapter 3
Methodology

This chapter describes the methodology and implementation employed for answering the research questions. The chapter includes measurement verification and limitations of the testbed implementation. For this thesis, a testbed was designed and built for conducting measurements on data plane performance. The measurements were carried out with predefined workloads to showcase the different attributes of the policy mechanisms. The performance measurements were subsequently analyzed and compared to the defined performance isolation requirements.

3.1 Testbed setup

To evaluate data plane performance isolation a simple testbed was set up for conducting data plane measurements on a single router. Figure 3.1 shows the architecture of the testbed, consisting of a Cisco router and a server hosting the traffic generators, traffic sinks, and the network interface cards used for traffic measurements.

3.1.1 Hardware

The testbed consists of virtual components hosted on two pieces of physical hardware. Specifications of the hardware follow below:

1. Cisco ISR4221/K9
2. Dell Precision 3630 with 2 1-gigabit network interface cards (NIC): Intel Ethernet Connection (7) I219-LM and Intel I200 Gigabit Network Connection

3.1.2 Router: Cisco ISR4221/K9

In this section, we present the router used in the testbed. We present how the router was set up during the measurements: Software, service performance, physical connections, interface configurations, and the network traffic loads.

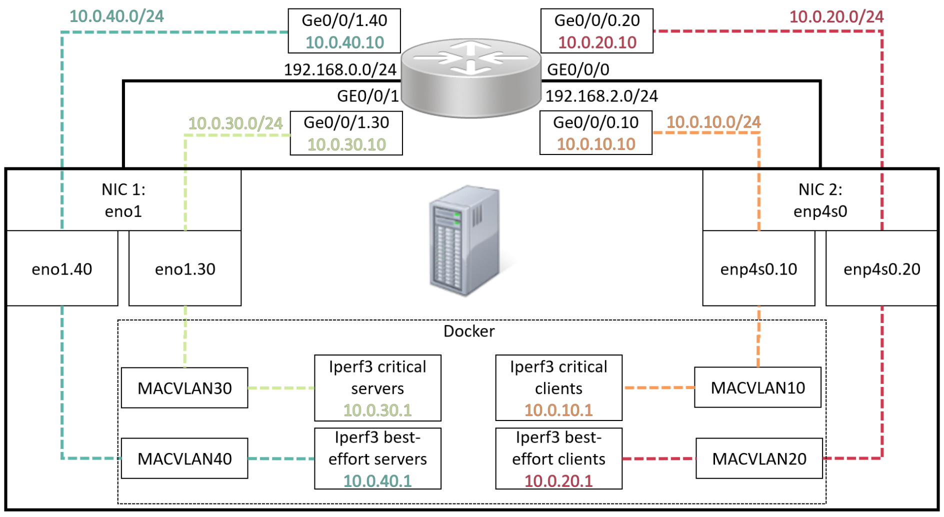


Figure 3.1: A logical model of the testbed, detailing the interface connections of the testbed.

Software

The ISR4221 is running Cisco IOS XE Gibraltar ISR software, version 16.12.04. The ISR is running with a smart license which restricts the throughput to 35000 kbps. Figure 3.2 illustrates the internal router architecture and the presence of a token-bucket that restricts the performance of the router to the licensed capacity.

Physical connection

The testbed with the ISR4221 consists of two physical connections. The first is from NIC 1 to gigabit interface g0/0/1 and the second is from NIC 2 to gigabit interface g0/0/0. Twisted-pair gigabit ethernet cables were used for both connections.

Interface configuration

All four physical interfaces are similarly configured. They have all been assigned IP addresses in the 192.168.x.x network. The connection between the networks was then verified. Next, sub-interfaces were configured. Four sub-interfaces on the router and four on the computer. The sub-interfaces are given matching names. For example, interface g0/0/1.40 and enp2s0f0.40 are connected via the shared 10.0.40.0 network.

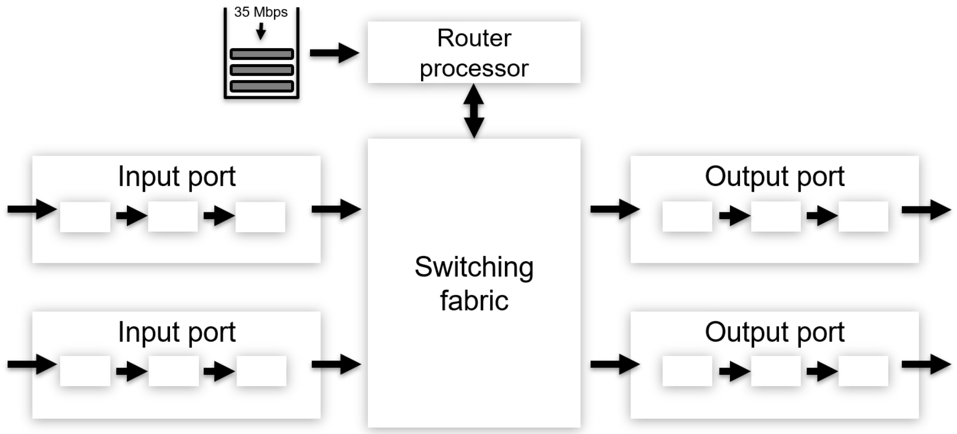


Figure 3.2: A model of the internals of the Cisco ISR4221.

Class-map configuration

Class-maps were used in order to assign the network traffic into different slices. Class-maps map traffic to a certain traffic-class based on a specific configuration. In Cisco IOS a maximum of 1024 class-maps can be configured, however, only 256 can be used in a single policy [Cis17][Cisa]. An implicit class called class-default is always present. VLAN matching was used for the traffic classification. All traffic matching any of the VLANs used for the critical network is assigned to the critical class-map, while traffic matching the best-effort VLANs is assigned to the best-effort class-map. Traffic not assigned to a specific class-map is by default assigned to the default class, class-default. The Cisco configuration used to implement class-maps follows below:

```
class-map match-any critical
 match vlan  10
 match vlan  30
class-map match-any be
 match vlan  20
 match vlan  40
```

Policy-map configuration

Policy-maps were used to implement the resource allocation mechanisms on the router. The purpose of the configured policy-maps was to create a resource allocation between the best-effort and critical traffic based on a single allocation mechanism. This was done in order to study the mechanisms, to determine its qualities and limitations. The configurations which were used are described in detail below. The policy-maps were applied to the physical interfaces of the router in order to take effect. The configurations aim to achieve the following two goals, where the first goal is the priority:

1. Achieve performance isolation for the critical slice.
2. Achieve fair sharing of unused network resources between the slices.

Work loads

The workloads consist of two elements, a static element and an on/off element. The static load is set to 45 percent of the total link capacity for both critical traffic and best-effort traffic, resulting in a total of 90 percent load. The on/off element is configured to last for 1 second, with a 1-second pause in between, it is limited to the best-effort slice. The purpose of this element is to study the effects on the network during periods of congestion in the best-effort slice. The on/off flows are set in a range from 5 percent to 40 percent of link capacity. Seven different measurement scenarios have been defined and are presented in table 3.1.2.

Scenario	Base load	On/off load	Peak total load
0	90%	5%	95%
1	90%	15%	105%
2	90%	20%	110%
3	90%	25%	115%
4	90%	30%	120%
5	90%	35%	125%
6	90%	40%	130%

Table 3.1: The loads used in the different measurement scenarios.

3.1.3 Dell Precision 3630

This section presents the server used in the testbed. We present the docker and flow control configuration used in the measurements.

3.1.4 Docker

Docker is a software tool used to automate the deployment of software in contained packages, known as containers. Containers run on a single Linux instance, but are isolated from each other through namespaces. Docker enables us to host multiple traffic generators and sinks in isolated lightweight containers.

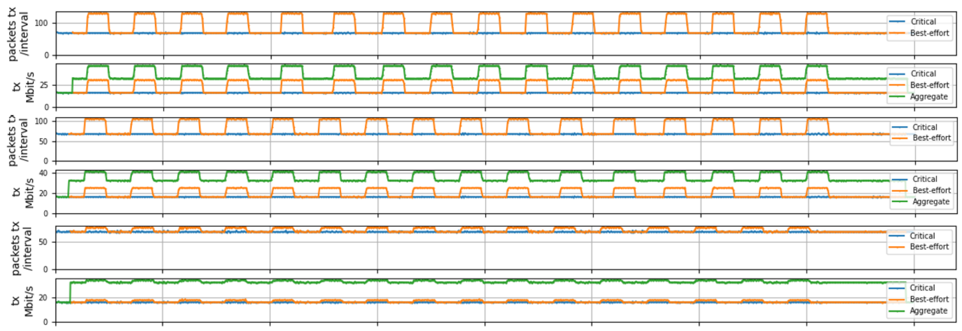


Figure 3.3: Time series comparison of generated load for scenarios. Scenario 6: Number of packets (1) and Mbps (2). Scenario 3: Number of packets (3) and Mbps (4). Scenario 0: Number of packets (5) and Mbps (6).

Docker networks

Docker networking was used to create and isolate the network connections between the docker containers and the router [Doc23]. Macvlans were configured in 802.1q trunk bridge mode. The macvlans were used to route the traffic through an 802.1q sub-interface on the NIC, preventing the traffic from being routed internally in the server. An example configuration of configuring macvlan10 using docker networking:

```
docker network create -d macvlan \
--subnet=10.0.10.0/24 \
--gateway=10.0.10.10 \
-o parent=enp4s0.10 macvlan10
```

The configuration includes the subnet, the gateway, and the sub-interface the macvlan connects to.

Docker images

Docker was used in the testbed to host the traffic generators and sinks. The docker image networkstatic/iperf3 [net23] was used for all containers. Iperf3 clients were used as traffic generators and iperf3 servers as traffic sinks. Multiple of these containers were run in parallel and connected to their assigned virtual networks. The iperf3 docker image lets us choose the transport protocol, and for UDP we can specify the throughput of the generated flow.

IEEE 802.3x flow control

Flow control was disabled on all interfaces during the evaluations to prevent interference with the results. Flow control was disabled at the NICs using ethtool. Firstly, flow control auto-negotiation was disabled. Secondly, flow control was disabled for both rx and tx. The following configuration is used to enable or disable flow control on a specific network interface.

```
ethtool -s eth? autoneg <on|off>
ethtool -A eth? rx <on|off> tx <on|off>
```

3.2 Data plane policies

This section presents how the resource allocation policies were configured.

3.2.1 Configuration goals

There is a need to specify quantifiable and configurable policies based on the high-level policy configuration goals presented earlier. The specific goals must be defined to handle these KPIs in scenarios of network congestion. The following specific goals were identified:

1. Provide the critical slice with a minimum 50 percent share of the capacity.
2. If possible, enable sharing of available capacity between the classes.

One policy is defined for each mechanism we investigate, a policy aims to utilize a single mechanism to achieve the identified goals. To configure a policy on the router a policy-map is used. All policy-map configurations are written to enable a minimum guaranteed 50 percent capacity to the critical slice. Depending on the mechanism under investigation, this is either done by allocating 50 percent to the critical class or restricting the default traffic class to 50 percent.

Default

For the default configuration, no policy-map is configured on the router. The best-effort and critical flows are expected to be treated equally, resulting in comparable performance for both flows regarding throughput, latency, and packet loss.

Shaping

Cisco traffic shaping is configured using the "shape" command. The shape configuration lets the user limit traffic on an interface to a set bandwidth or a percentage.

Traffic that violates the rate limit is queued. Cisco has five different options for shaping: average, peak, adaptive, fecn-adapt and becn-adapt. Adaptive, fecn-adapt, and becn-adapt all use an adaptive algorithm based on network conditions and congestion notifications. The peak and average shaping options on the other hand are static. The shape policy is configured based on three parameters: Committed information rate (CIR), committed burst (Bc), and excess-burst (Be). Bc and Be are optional.

```
shape {average | peak} cir [bc] [be]
```

For shape average, the default burst is limited to the CIR. While for shape peak the default burst is set to 2*CIR. To define an appropriate shaping policy, the difference between shape average and shape peak was investigated. One shape average policy and one shape peak policy with the same CIR of 17.5 Mbps were investigated. The policies were written to shape the best-effort class. A second peak configuration was also written, with a CIR equal to 8.75 Mbps to verify if the performance of this peak configuration and the shape average configuration with a CIR of 17.5 Mbps would yield the same results.

The comparison of the three configurations yielded the following: 1. The shape average policy limited throughput to 17.5 Mbps, with no excess burst allowance. 2. Shape peak of 8.75 Mpbs limited throughput to 17.5 Mbps, yielding identical measurements as the shape average policy. The default size of burst allowance is greater than the size of traffic bursts used in the investigation. 3. The shape peak policy of 17.5 Mbps did not limit the throughput of the best-effort traffic to any degree for the work loads used. Based on these findings, only the shape average configuration was used to represent the shaping mechanism going forward.

Shaping configuration

The shaping policy was configured as follows:

```
policy-map average
 class critical
 class be
  shape average 17500000
!
```

Policing

The police policy is configured based on three parameters: CIR, Bc, and Be. These three parameters set the thresholds for the three states: conforming, exceeding, and violating. Traffic conforms if it falls within CIR + Bc, while it is exceeding if it falls within CIR + Bc + Be, and finally violating if it is greater than CIR + Bc + Be. For each of the three states, actions can be defined. The actions are transmit, drop, or mark. IP precedence, MPLS, DSCP, CoS, and QoS markings can be assigned with the mark action, the marked packet is then transmitted. As this evaluation only contains a single bottleneck, using the marking option is not relevant. The policy is configured to transmit conforming traffic and dropping exceeding traffic, default burst parameters are used.

```
police cir [bc] [be] conform-action {action} \
exceed-action {action} violate-action {action}
```

Policing configuration

The police policy is applied to the best-effort class in order to restrict it to a rate that does not affect the critical traffic. The policy is configured as follows:

```
policy-map policer
 class critical
 class be
  police 17500000 conform-action transmit  exceed-action drop
!
```

Bandwidth

Minimum bandwidth guarantees are configured using the bandwidth command. The bandwidth command takes a data rate as a parameter. For this policy, the critical class is configured with a minimum bandwidth. The Cisco bandwidth mechanism works as a percentage-based allocation or an absolute throughput allocation. The mechanism allows administrators to allocate bandwidth guarantees to specific interfaces or traffic classes and can be applied to both inbound and outbound traffic. However, the bandwidth command does not actually influence the data plane interface. It logically sets the bandwidth and communicates the bandwidth to routing protocols and TCP. Note that the data rate is denoted in kbps, not bps as it is for shaping and policing.

```
bandwidth {kbps | percentage}
```

Bandwidth configuration

The bandwidth policy was configured as follows:

```
policy-map bandwidth
```

```
 class critical
  bandwidth 17500
 class be
!
```

Priority

Strict priority queuing can be configured using the priority command. Cisco implements the low latency queuing (LLQ) mechanism for providing strict priority for certain traffic. LLQ is a QoS mechanism that places prioritized traffic in a separate priority queue. This ensures that the priority traffic is served first, even if other traffic is ahead of it. Cisco has 4 priority index values. 1 is the highest and 4 is the lowest. Index value 3 is the default value. However, in the case of the ISR4221, only 2 levels are available: 1 and 2. For this configuration, the critical class is given priority level 1. Prioritized traffic is placed ahead of non-prioritized traffic in the queue, and the traffic with higher priority is therefore not delayed behind other traffic. This feature is very relevant for communication types that require low latency, but could also lead to bandwidth starvation if there are large amounts of high-priority traffic coming through. However, this can be mitigated by using traffic policing. Shaping and bandwidth cannot be configured along with priority. Priority is configured either for all traffic or for a certain capacity of the link either as a percentage or a kbps value.

```
priority {kbps | percentage | level 1 | level 2}
```

Priority configuration

The following configuration was used for assigning priority:

```
policy-map priority
 class critical
  priority 17500
 class be
!
```

WRED

Cisco enables probabilistic dropping through random early detection (RED) or weighted random early detection (WRED). RED and WRED work by selectively dropping packets based on their queue length and a configurable probability value. When a queue reaches a certain threshold, RED or WRED randomly drops packets from the queue with a probability that increases as the queue length grows. This approach can help to reduce congestion and improve network performance, as it encourages sources to reduce their traffic rates before the network becomes too congested. They can be configured to prioritize certain types of traffic over others and adjust their parameters based on the needs of the network.

WRED is configured based on three parameters: Minimum threshold (min), maximum threshold (max), and mark probability denominator (MPD). The minimum threshold defines the queue length criteria for WRED to take action. The maximum threshold defines the queue length criteria for WRED to take full effect. MPD is used to calculate the probability of packet drop. When the queue length equals the maximum threshold, the probability is 1/(MPD). For example, if the mark probability denominator were set to 20 when the queue depth reaches the maximum threshold, the probability of discard would be 1/20 or 5 percent. WRED uses either IP precedence or DSCP for classifying data. Separate dropping probabilities can be configured for each IP precedence or DSCP marking. UDP iperf3 traffic does not have any specific DSCP marking, meaning that the DSCP default marking, 0, is used when configuring WRED for this traffic. The intention of the WRED configuration is to begin dropping packets from the best-effort network when congestion occurs. A minimum threshold of 1 packet was chosen, and a maximum of 1 packet, at which point there should be a 100 percent drop chance. Meaning an MPD value of 1.

```
random-detect {dscp | precedence} [marking value] [min] [max] [mdp]
```

WRED configuration

The following configuration was used for WRED:

```
policy-map wred
 class critical
 class be
  random-detect dscp-based
  random-detect dscp 0 1 1 1
!
```

3.2.2 Tail drop

The tail drop mechanism is configured based on a threshold queue length. When the length of the queue surpasses the threshold packets are dropped. Queue length can be denoted as a length in bytes, milliseconds, or packets.

```
queue-limit {bytes | ms | packets}
```

Tail drop configuration

The following configuration was used for tail drop:

```
policy-map taildrop
 class critical
 class be
   queue-limit 10 ms
!
```

3.3 Data collection

This section presents how data traffic was generated and collected. Iperf3 was used for generating traffic, while tcpdump was used for collecting traffic data.

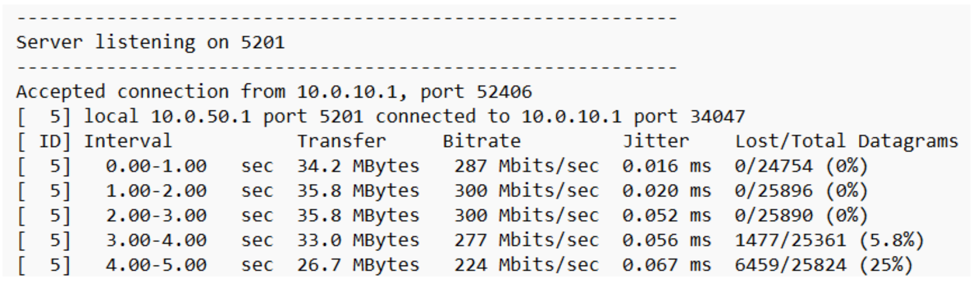
Iperf3

Iperf3 generates packets of 1500 bytes. The packets are marked with a sequence number, making it possible to track a specific packet across a network. The time interval at which the packets are generated is unknown, but based on findings in our measurements, the interval is directly connected to the throughput of the flow. We have observed that packets are generated and sent either in direct succession or with a 1-millisecond interval, the distribution is shown in the cumulative distribution functions (CDF) in Figure 3.6. We have observed that the distribution of packets is dependent on the throughput of the flow.

Iperf3 generates a text output of the measured results on both the server and client side of the flow session. However, the results are given in 1-second intervals, which are considered to be too coarse-grained for this analysis. An example of iperf3 output is shown in Figure 3.4. To reduce the interval size down to the millisecond range tcpdump was chosen as the data collection tool instead.

Tcpdump

Tcpdump is a command-line interface-based network packet analysis tool. It allows the user to conduct packet capture at a specified network interface. The results of the packet capture can be saved to a .pcap file. For this experiment tcpdump was used to save all traffic on the sub-interfaces connecting the Iperf3 clients and servers to the network. Saving the traffic in .pcap files enables measurements to take place at a per-packet level. As all the sub-interfaces are hosted on the same computer, the packet timestamps in all .pcap files will have the same clock source.



```
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 10.0.10.1, port 52406
[  5] local 10.0.50.1 port 5201 connected to 10.0.10.1 port 34047
[ ID] Interval           Transfer     Bitrate         Jitter    Lost/Total Datagrams
[  5]   0.00-1.00   sec  34.2 MBytes   287 Mbits/sec  0.016 ms  0/24754 (0%)
[  5]   1.00-2.00   sec  35.8 MBytes   300 Mbits/sec  0.020 ms  0/25896 (0%)
[  5]   2.00-3.00   sec  35.8 MBytes   300 Mbits/sec  0.052 ms  0/25890 (0%)
[  5]   3.00-4.00   sec  33.0 MBytes   277 Mbits/sec  0.056 ms  1477/25361 (5.8%)
[  5]   4.00-5.00   sec  26.7 MBytes   224 Mbits/sec  0.067 ms  6459/25824 (25%)
```

Figure 3.4: Example of the text output from an iperf3 server.

The following command was used to conduct a tcpdump at sub-interface enp4s0.10, the tcpdump is saved to the file vlan10.pcap:

```
sudo tcpdump --interface enp4s0.10 -w vlan10.pcap
```

3.4 Data analysis

In order to showcase the differences between the allocation mechanisms three performance indicators were used: latency, packet loss, and throughput. For latency measurements, it is possible to track every single packet that is not dropped by the network. By comparing the timestamps on the sending and receiving side, the network latency can be calculated and plotted. For packet loss and throughput, however, a measurement interval is used. The smaller the interval the more data points will be visible. Generating more data points will lead to a higher resolution for the performance indicators. However, using a very small interval with only a few packets per interval can lead to significant fluctuations in the results based on which interval a single packet is placed in. On the other hand, using a longer interval will smooth out these fluctuations, but important details in the data might be missed as it is assimilated into the rest.

This interval for the data analysis was set to 50 ms. Resulting in 20 data points per second.

Python was used to write a script for data visualization. At first, the Pyshark library was used to read the .pcap files, and the Matplotlib library was used to plot the data. Pyshark was later replaced by the DPKT library due to the per-packet processing time. When working with large .pcap files, the per-packet processing of Pyshark was problematic. In [Sha23] the performance of pyshark was found to be 25.067 ms per packet, while 0.011 ms per packet for DPKT. Using DPKT instead of Pyshark cut the total script runtime significantly.

3.4.1 Throughput

Throughput sent was measured by registering the amount of packets sent by the clients per time interval and multiplying this number by the size of the packet in megabits and dividing by the time interval. Received throughput was measured by registering the number of received packets at the server per time interval and multiplying this number with the size of the packet in megabits and dividing by the time interval.

Initially, the timestamps on the sender side were used to place the packets in their designated time intervals for both sent and received throughput. However, this led to some inaccurate measurements during initial network congestion, where the throughput surpassed the service limits of the network due to queue bloating. The timestamps used for received throughput were therefore changed to the receiving side to more accurately reflect the throughput of the network.

3.4.2 Packet loss

Packet loss was computed by comparing the number of packets sent with the number lost. Each packet registered and received on the server side of the network was compared to the packets that were sent on the client side. By comparing the number of packets sent with those received in each time interval the packet loss in that given interval can be calculated. The timestamps on the sender side were used to place the packets in their designated time intervals. An aggregate value was added by using the sum of packets sent across the entire network, meaning both slices, and comparing it with the sum of packets lost across entire the network.

3.4.3 Latency

The one-way latency was computed on the flows with continuous traffic through the length of the evaluation. For these flows delay was calculated for every single packet that made it through the network. For every packet that was registered as received by the NIC on the sink side, the delay of the packet was calculated by comparing the timestamps of when it was sent into the network to when it was received. The delay was plotted using the timestamps on the sender side to place the packets on the x-axis.

3.5 Validation of the testbed

A validation of the testbed was conducted to provide insights into potential weaknesses in the investigation. While conducting measurements on the testbed we noticed seemingly random delay spikes occurring as seen in Figure 3.5. An investigation was carried out to verify whether these spikes could be caused by the measurement setup. We observed that these spikes appeared across both slices, for all scenarios and configurations. We investigated the equipment used, to verify if the spikes could be caused by the traffic generation, the time stamping, or the router. Inter-arrival time measurements at the NICs were used for this purpose. The inter-arrival time of generated packets was measured to verify and assure that the generated traffic patterns are valid and repeatable. In Figure 3.6 we can see the CDFs of the inter-arrival time (IAT) for the generated traffic for scenarios 1 to 6. As can be seen in the figure iperf3 has a distinct traffic generation pattern, where packets are either sent almost simultaneously or with a 1 ms gap. The pattern is similar for all scenarios.

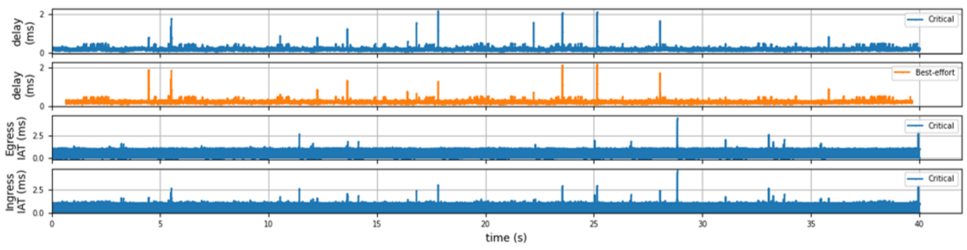


Figure 3.5: Times series of critical latency (1), best-effort latency (2), IAT of generated critical traffic (3), and IAT of critical traffic received by the traffic sink. The delay spikes in the (1) and (2) can be seen in IATs at the sink, but not at the generator.

To further verify the similarities between the generated data. We compared the IAT measurements of generated packets in a pairwise manner using a Kolmogorov-Smirnov test [Mas51]. Running a pairwise Kolmogorov-Smirnov test on all combinations of the curves in the different scenarios can be used to determine the similarity between the distribution of the data sets. All samples have the same size of 54384. Scenario 1 had the largest KS-statistic value. The measured KS statistic was equal to 0.0698 with a p-value of 9.2e-116. The smallest measured appeared in scenario 2 with a KS-statistic equal to 0.0125 and a p-value of 0.00038. The KS-statistic is the maximum vertical distance between two points on the CDF, and the p-value is the probability that the data sets come from the same distribution. To further compare we calculated the mean and variance of the data. For the pair with the largest KS-statistic value, the two means were found to be: 0.73549325 and 0.73549439, with corresponding variances of 0.18466 and 0.18694. These comparisons show that even though the mean and variance are similar, there is a significant degree of discrepancy in the iperf3 traffic generation as shown by the KS-test.

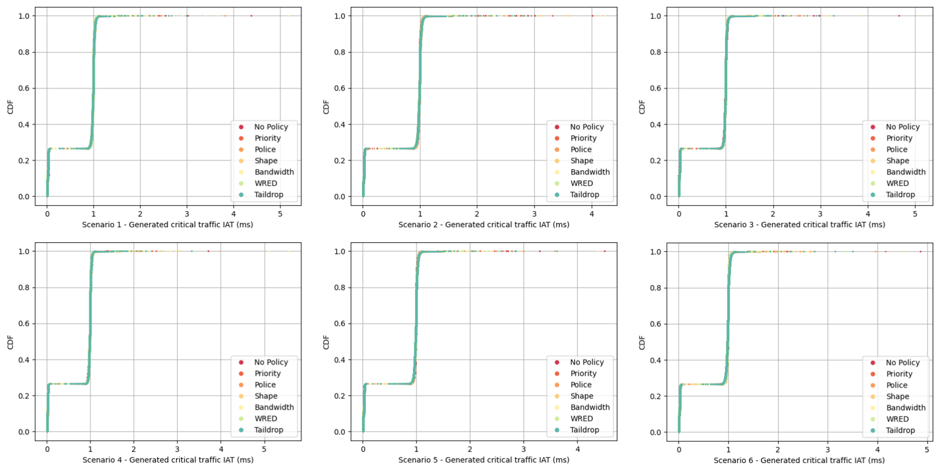


Figure 3.6: CDF comparison of generated packet inter-arrival time.

3.6 Limitations

The evaluations are done using static resource allocation configurations. Using static configurations limits the evaluations in two aspects. The first is that dynamic allocation is an important attribute of network slicing. Changes to the slice configurations should be done dynamically, automatically, and on-demand. The second is the scalability of slices. Statically configuring all slices on all network nodes is not a scalable solution.

The amount of flows is a limitation to the relevancy of the evaluation. If flows are not aggregated, states need to be kept for each flow, resulting in restricted scaling potential due to hardware limitations. An additional evaluation deploying a high number of low-impact flows could be performed to evaluate how the configurations handle this scenario.

The Cisco 4221 ISR, has an IOS set throughput limitation of 35 Mbps. Which results in a software-defined bottleneck. In networking equipment, without a software throughput limitation, either the interface capacity on the device is the bottleneck or the throughput of the switching fabric. This will influence how the resource allocation mechanisms behave, which will potentially be different from the Cisco 4221 ISR.

Chapter 4 Results

This chapter presents the measurement results for the configurations and scenarios presented previously. The measurements have been carried out in multiple iterations, and the ones presented in this section will only be the final version. The figures of the measurements, such as Figure 4.1, present the averages with a 95 percent confidence interval for every scenario with for every policy.

4.1 Critical traffic

This section presents the measured critical traffic performance. In this section, we are primarily studying the measurements to verify whether performance isolation is maintained. We know that the best-effort slice is overbooking the resources allocated to the slice, resulting in congestion of the slice. We therefore need to observe that the critical slice is unaffected by this congestion to verify inter-slice performance isolation.

4.1.1 Throughput

Starting with throughput, the average throughput measurements are presented in Figure 4.1. The results show that all the applied mechanisms can provide stable throughput for the critical traffic even during congestion. From the figure, we can see that only the scenarios without a policy are affected in terms of reduced throughput during congestion. Further, we can observe that the scenarios with a higher degree of congestion lead to lower average throughput.

4.1.2 Packet loss

The average packet loss measurements are presented in Figure 4.2. Again we can observe that all mechanisms provide inter-slice isolation, and can provide zero packet loss for the critical traffic. The mechanisms can provide isolation in terms of packet loss even during network congestion.

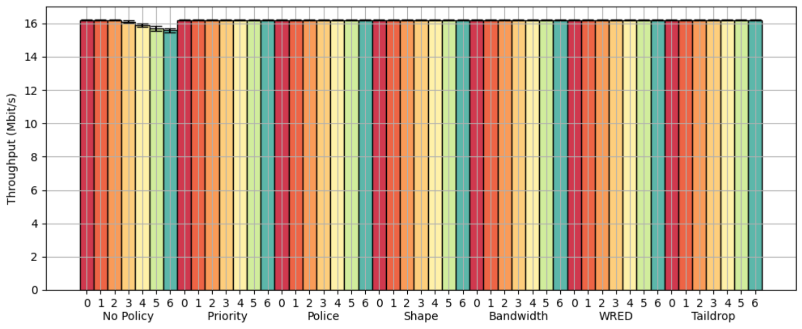


Figure 4.1: Critical traffic throughput.

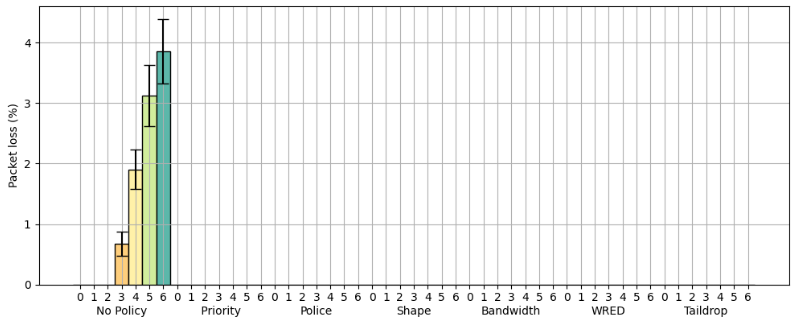


Figure 4.2: Critical traffic packet loss.

4.1.3 Latency

The average latency measurements are first presented in Figure 4.3. Secondly, the same measurements, with a y-axis interval of 0 to 1 are presented in Figure 4.4. From the measurements, we can observe that all mechanisms, except bandwidth, are able to provide a similar average latency for the critical packets in both scenarios with and without congestion. For the bandwidth mechanism, the average latency is significantly higher than the others in scenarios 3, 4, 5, and 6. In scenario 6 the value has passed 0.4 ms, which is an approximate increase of 100 percent compared to the other mechanisms.

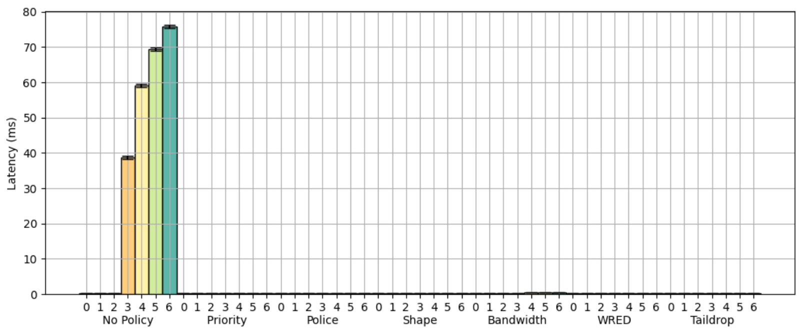


Figure 4.3: Critical traffic latency.

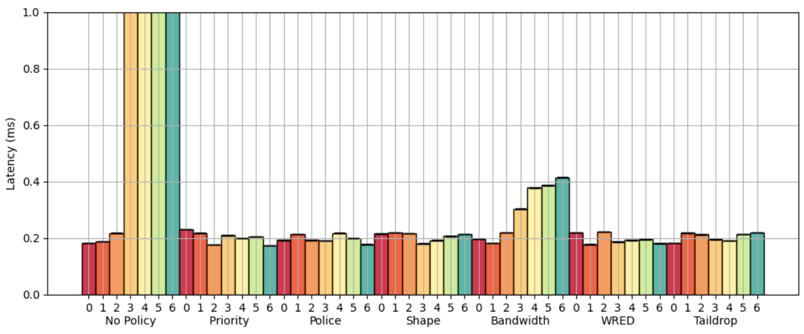


Figure 4.4: Critical traffic latency, zoomed into the interval of 0 to 1 ms.

Cumulative distribution functions

The CDFs of the latency measurements are presented in Figure 4.5. A clear distinction between the bandwidth mechanism and the others can be observed for scenario 3 and beyond, where the peak traffic exceeds a load of 40 Mbps. Furthermore, we can observe how the latency distribution worsens in the case of a greater congestion degree.

Taking a closer look at the latency of the bandwidth mechanism we can observe that a majority of packets are located in the interval of 0.2 ms to 1 ms. A potential reason for this significant shift can be that the critical traffic is affected by the 35 Mbps capacity limit. To investigate this theory we can calculate the theoretical latency in the testbed. The measured latency is the sum of the transmission delay,

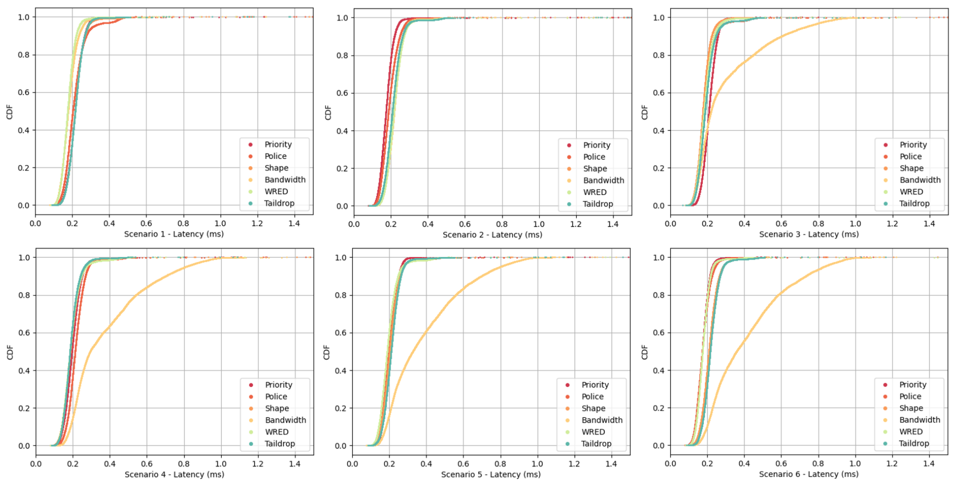


Figure 4.5: CDFs of critical latency measurements.

propagation delay, processing delay, and queuing delay in the testbed. We have observed the packet size of the iperf3 traffic to be 1500 bytes. Both links to the router are gigabit connections. The propagation delay for a packet across a one-gigabit link can be calculated to 0.012 ms. Across two links this delay is 0.024 ms. The transmission delay of the testbed is insignificantly low. Which leaves the processing delay and queuing delays as unknown. The packet with the lowest measured latency across all scenarios had a latency of 0.066 ms. Using this, we can calculate the lowest observed sum of processing latency and other unknown factors to be 0.042 ms. Knowing that the capacity limit of 35 Mbps we can calculate the propagation delay of packets affected by the limit to be 0.34 ms. Adding on an additional propagation delay of 0.012 ms and the processing delay of 0.042 ms equals a minimum latency of 0.394 ms for a packet that is affected by the 35 Mbps limit on the output port. This calculation does not take queuing into consideration. However, as the latency measurements are evenly distributed without any clear thresholds we are not able to conclude with anything.

4.2 Best-effort traffic

This section presents the measurements of the best-effort traffic performance. In this section, we are primarily studying the measurements to verify whether inter-slice resource reallocation is performed. The flows in the critical slice are not utilizing the entire capacity allocation, 10 percent of the allocation is unused. Knowing this, we can observe whether the configurations are able to perform reallocation and utilize the unused capacity.

4.2.1 Throughput

Starting with the throughput measurements we can observe differences between the mechanisms. "No policy" displays the highest throughput results, which makes sense as the best-effort traffic is treated the same as the critical traffic. The equal treatment results in packet loss in the critical slice, allowing additional packets from the best-effort slice to be processed, leading to an increase in best-effort throughput. For the priority and bandwidth mechanisms, the throughput is significantly higher than the other mechanisms. These mechanisms are able to reallocate available capacity to the best-effort slice to the point of full link utilization in scenario 3 and beyond, at which point we can observe a stable average. For the police mechanism, we can observe a greater throughput average than shaping, WRED, and tail drop. This can be explained by a default degree of burst acceptance during congestion, which can be observed during the measurements, resulting in an increased average.

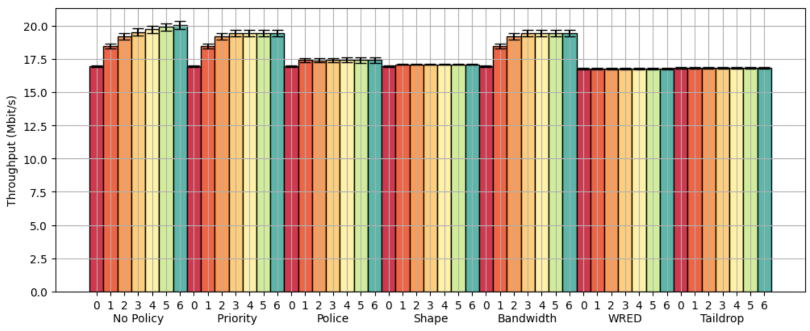


Figure 4.6: Best-effort traffic throughput.

4.2.2 Packet loss

The packet loss measurements are presented in Figure 4.7. As seen in the throughput measurements the packet loss measurements illustrate that the priority and bandwidth mechanisms can reallocate available capacity resulting in a significantly lower degree of packet loss. For the policing mechanism, we can again observe the effect of the permitted burst, in a lower degree of packet loss compared to shaping.

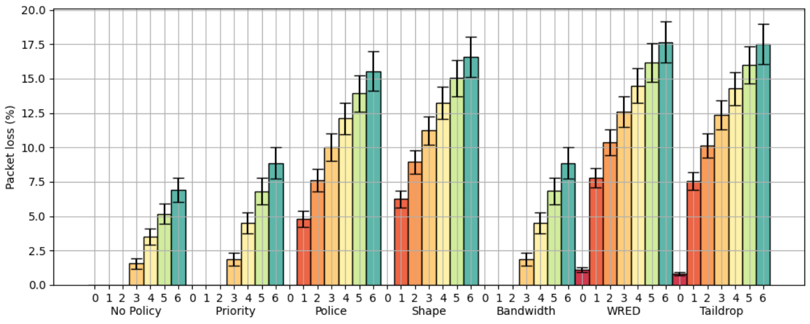


Figure 4.7: Best-effort traffic packet loss.

4.2.3 Latency

The latency measurements are presented in Figure 4.8 and Figure 4.9. The measurements show a clear difference in latency between policies. For no policy, the buffer is split between the two slices without any preemption for the critical slice. Meaning that the average latency is cut down drastically for the best-effort slice. For the priority and bandwidth mechanisms, there are no buffer limitations, and since the critical traffic is preemptively serviced, the best-effort slice utilizes the entire buffer. Resulting in the high latency values seen in the measurements. The policing mechanism does not use the buffer at all, which is reflected in these measurements. As for shaping, WRED, and tail drop, we can observe that the mechanisms use the buffer to varying degrees, which is capped at a certain level.

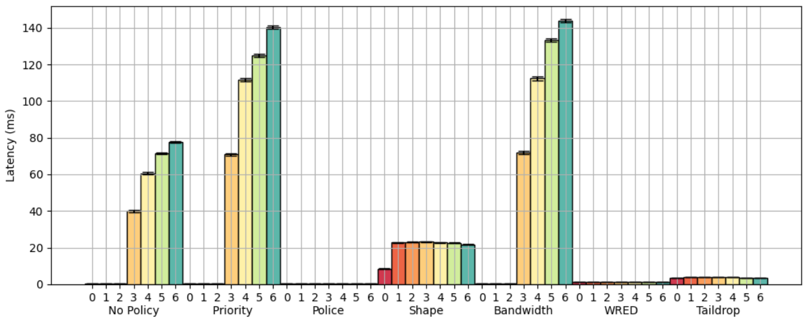


Figure 4.8: Best-effort traffic latency.

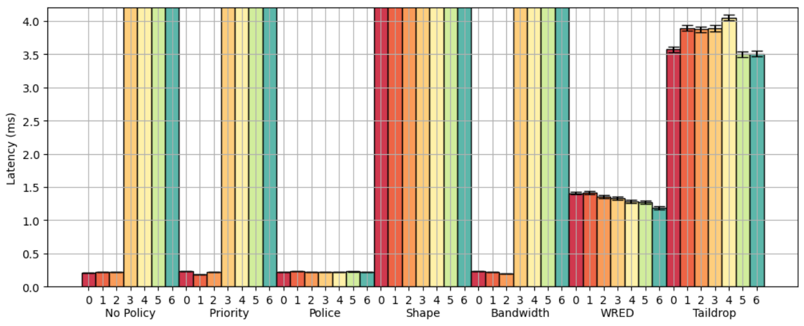


Figure 4.9: Best-effort traffic latency, zoomed into the interval of 0 to 4 ms.

4.3 Bandwidth investigation

In the latency measurements of the critical traffic, we have observed an increased latency for the bandwidth mechanism. To verify if the configuration of the bandwidth mechanism is adding some processing overhead to the router which interferes with the latency of the critical traffic, even when unused, an additional measurement was conducted. For this additional investigation, new policies were defined. The policing, shaping, WRED, and tail drop mechanism configurations were all reconfigured to include a minimum bandwidth of 10 Mbps for the critical traffic. These configurations were used as they are all solely based on the best-effort class. The following configuration was added to all of them:

```
class critical
  bandwidth 10000
```

The other policies were left unchanged. The measurements were then rerun using these new configurations. The new latency measurements for the critical traffic are presented in Figure 4.10. These measurements show that including a minimum bandwidth allocation into the other policies did not yield a similar increased effect on the latency. This shows that implementing the bandwidth mechanism in a policy did not result in any performance degradation, which is useful to narrow down the investigation into the latency isolation breach.

4.3.1 Latency trade-off

To compare the mechanisms in terms of latency, a scatter plot with best-effort latency on one axis and critical latency on the other was plotted. This plot can be useful to

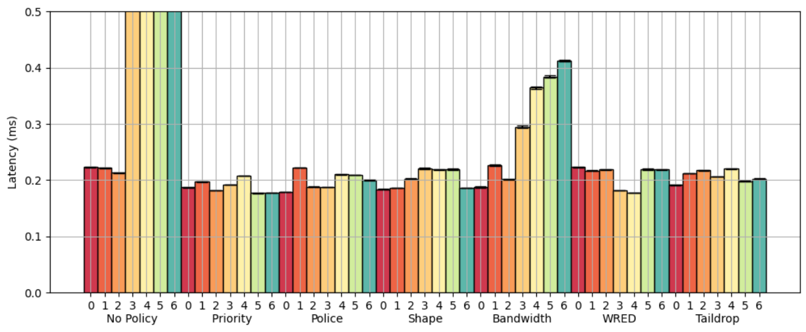


Figure 4.10: Bandwidth mechanism investigation using updated policies: Critical traffic latency, zoomed into the interval of 0 to 1 ms.

determine if there are any trends or trade-offs in terms of latency. Figure 4.11 and Figure 4.12 show the same plot with different axes values. The average latency of both critical and best-effort for each mechanism in every scenario is plotted as a dot. In Figure 4.11 we can observe a trend for the bandwidth mechanism, and based on what we have seen previously we know that increased congestion leads to an increase in both best-effort and critical latency. As for the other mechanisms, there are no similar trends, they are all stable in terms of critical latency. We can observe that the best-effort latency of the priority mechanism increases based on congestion, while for the other mechanisms, it is stable around specific values.

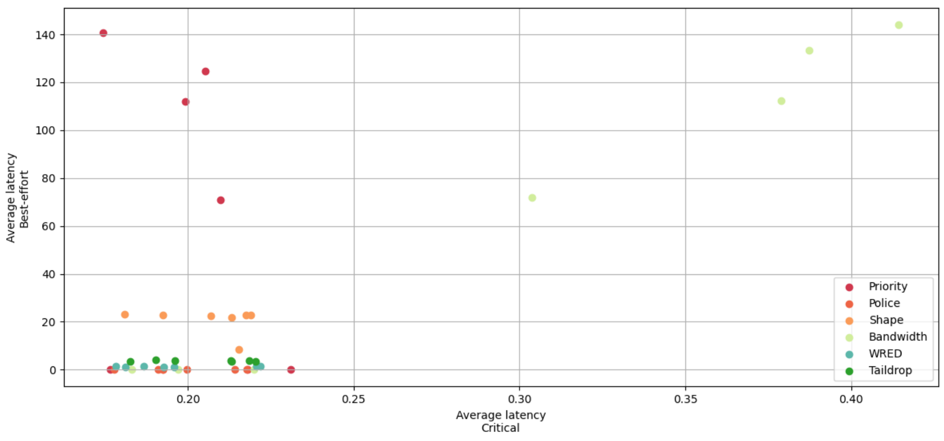


Figure 4.11: Critical and best-effort latency comparison.

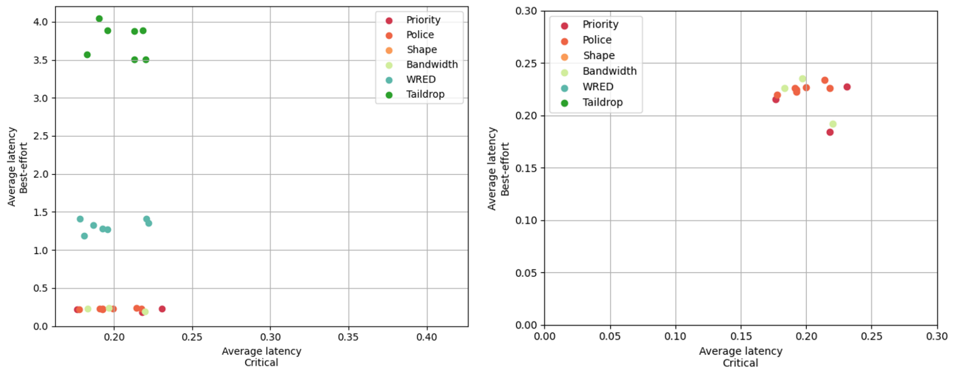


Figure 4.12: Critical and best-effort latency comparison zoomed in.

4.4 WRED and latency measurements

In an early iteration of the measurements we observed how WRED resulted in a breach of latency performance isolation, see Figure 4.13. Similar to the bandwidth mechanism the latency of the critical traffic increased during periods of congestion. However, after rewriting the configuration of the WRED measurement the increase in latency no longer appeared in the measurements. The early version of the WRED configuration was based on using the bandwidth mechanism along WRED instead of the shaping mechanism. Using the bandwidth mechanism along with WRED appears to have breached performance isolation between the two slices.

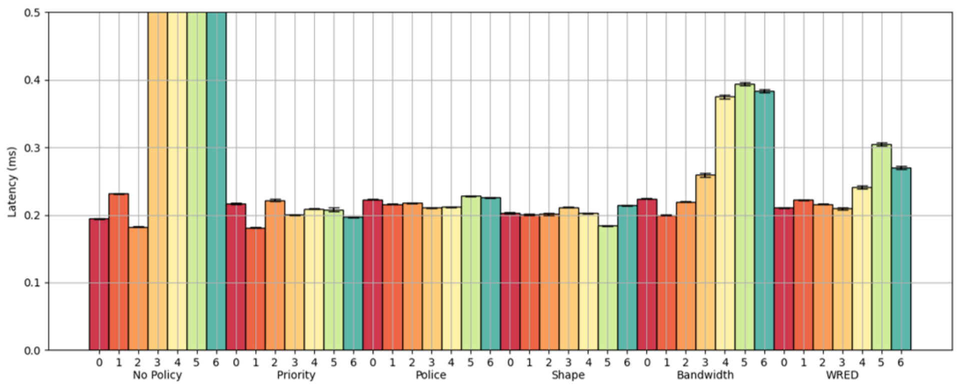


Figure 4.13: Critical traffic latency for a bandwidth mechanism based WRED configuration.

4.5 Enforcing allocation for both traffic classes

To increase the complexity of the configurations and verify if the results are altered, an additional resource allocation is configured for all policies. Meaning that both the critical slice and the best-effort slice are given an allocation. For the priority class, this means that the critical slice is configured with priority level 1 and the best-effort slice is configured with priority level 2. Figure 4.14 shows the latency results of the critical traffic with the added allocations. The only noteworthy result is a lower average critical traffic latency for the bandwidth mechanism. After studying the measurement results, we observe that the buffer is no longer being fully utilized by the best-effort slice. Resulting in more packet loss, but this also leads to empty queues in between the increases in traffic load. As the bandwidth mechanism only experiences increased critical latency during congestion, shorter periods of congestion lead to a lower average latency.

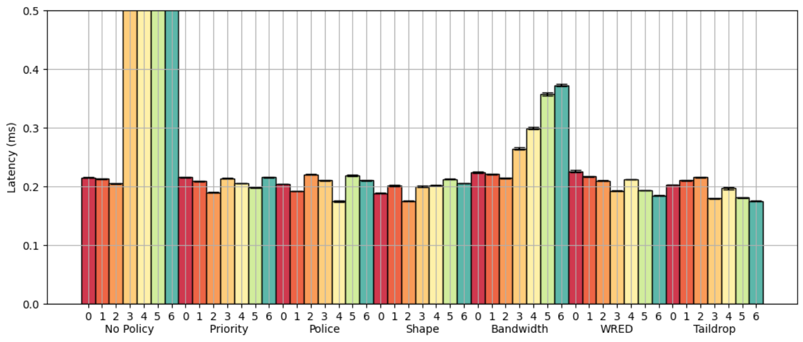


Figure 4.14: Critical traffic latency with allocation on both classes.

4.6 Measurement summary

Table 4.1 summarizes the findings of measurements. The table presents the findings based on three attributes: Performance isolation, resource reallocation, and scalability. We have defined performance isolation to be the undisturbed service of a slice in terms of throughput, latency, and packet loss. A breach in performance isolation will occur if one of the performance indicators is significantly affected by the behavior of another slice. Resource reallocation refers to the sharing of available capacity. For the measurements presented, this is observed primarily in terms of throughput and secondarily as packet loss. Scalability refers to the maximum degree a certain mechanism or configuration can be scaled. This degree can be either defined by hardware limitation or a significant reduction in performance. For the measurements conducted in this thesis we have not studied the effect of scalability on performance, hence we have defined scalability as the maximum hardware threshold. For Cisco equipment, this threshold can be found in the documentation of the mechanisms described in the methodology chapter. Furthermore, we conducted verification of these numbers on the ISR4221. The ISR4221 supports a maximum of 2 priority levels and has a maximum limit of 256 classes per policy. Only one policy can be configured per interface, resulting in a maximum of 256 traffic classes per interface.

	Performance isolation	Resource reallocation	Scalability
No policy	×	✓	No limit
Priority	✓	✓	2 levels (256)
Shaping	✓	×	256
Policing	✓	×	256
Bandwidth	×	✓	256
WRED	✓	×	256
Tail drop	✓	×	256

Table 4.1: Summary of the investigated slicing attributes.

Chapter 5 Discussion

The following chapter is a discussion of the measurement results, the testbed, and potential future work.

5.1 Performance isolation

In a shared transport network infrastructure, different traffic types compete for finite resources such as link capacity and buffer space. Even with prioritization mechanisms, contention arises during periods of high demand, potentially impacting performance in unexpected manners. Networks experience traffic fluctuations, with bursts of data or sudden spikes in demand. Handling these dynamic variations while ensuring isolation for all traffic types is complex and leads to trade-offs. External influences beyond the operator's direct control or awareness, such as unforeseen traffic surges, or malicious attacks can impact isolation despite internal efforts. The hardware limitations of networking equipment can hinder the implementation of isolation mechanisms with large processing overhead. Achieving perfect isolation for all traffic types might involve sacrificing certain aspects, such as increased latency for lower-priority traffic or reduced overall throughput to ensure prioritization of critical traffic. Balancing these trade-offs is a challenge that network slicing has to address.

The results show that all investigated mechanisms can provide performance isolation in the form of throughput and packet loss. This, however, does not hold for latency. As seen in the measurements, the bandwidth mechanism is not able to prevent a misbehaving slice from affecting other slices. In the case of the bandwidth mechanism during periods of network congestion, the performance of the critical slice in terms of latency is severely affected. As for the other mechanisms, they are viable techniques for providing logical performance isolation on a common network. However, the mechanisms are not able to provide redistribution of available resources. Only the priority mechanism shows both performance isolation and resource reallocation, but since the mechanism is limited to only two priority levels it can not be seen as a sole solution either. Another alternative is adding an algorithm for dynamic buffer and throughput allocations to the shape, police, WRED and taildrop configurations, continuously readjusting the allocations based on network conditions, the dynamic allocation algorithm would provide resource reallocation. Shaping, policing and WRED would then potentially fulfill every attribute we have investigated. However, depending on the complexity and overhead of the implementation scalability might be sacrificed. Latency isolation is not considered explicitly in this work, it is a byproduct of the throughput allocations. We have looked at the latency as a performance indicator, but the configured policies do not take latency requirements into account. In working towards a holistic slicing policy that incorporates latency requirements more investigations into slice-specific virtual queues and priority queues are needed. To investigate if these mechanisms can be leveraged to provide both latency isolation and configure latency guarantees is needed.

5.2 Transient and steady state

While conducting the measurements presented, we observed a transient phase in the first 8 seconds of the experiments. During this phase, the router is able to service traffic at a rate greater than the limit of 35 Mbps, after which the service rate stabilizes at 35 Mbps. This increase in throughput during the transient phase increases the average throughput and decreases the packet loss in the case of the priority policy, no policy, and bandwidth policy. This phase is skewing the measured results towards higher throughput and lower packet loss for these three policies, as compared to removing the transient phase from the results. During the transient state, the system does not reflect the typical behavior. Measurements taken during this period do not represent the system's performance or characteristics in stable condition. Thus, using the transient data to conclude leads to a degree of inaccuracy. Measurements taken during transient states might not be directly comparable to those during steady-state conditions.

5.3 Scalability

In the case of the configurations used in this thesis, there are two major scalability restrictions. The first is the amount of class-maps that can be configured into a policy. The second is limitations on priority levels, as the router used in this thesis only provides 2 different levels. The underlying hardware imposes scalability limits depending on the number of slices and users. For instance, the capacity of the forwarding tables, memory limitations, or the available processing power can restrict the scalability of resource allocation mechanisms. Implementing complex resource allocation mechanisms will incur significant processing overhead. As the volume of traffic increases, these mechanisms might struggle to handle the workload efficiently, negatively impacting performance. The more intricate the mechanism in managing individual resources, the more difficult it becomes to scale efficiently for a large number of flows. Furthermore, if the slicing implementation is based on a distributed system the control plane's ability to distribute and manage resource allocation can become a bottleneck as the network scales.

Mechanisms that lack adaptability to dynamic changes in traffic patterns or network conditions will also face scalability issues. As the network evolves or experiences fluctuations in demand, rigid resource allocation mechanisms might struggle to adjust efficiently. Compatibility and interoperability issues between different devices or platforms can further add to this issue. If resource allocation mechanisms are not standardized across different vendors' equipment distributed allocation policies cannot seamlessly work together.

Resource allocation mechanisms that involve real-time decision-making or those sensitive to latency might face scalability constraints. As the network grows, meeting stringent latency requirements across a larger scale can be difficult. Inefficient use and fragmentation of resources can hinder scalability. For instance, if resources are strictly allocated, but not fully utilized or if they are allocated in small, fragmented portions across the network.

The issue of scalability will have a major impact on the implementation of slicing. We have seen that the main factors to consider are the number of slices supported, the size of the slice distribution, the amount of tracked states, and the allocation mechanisms used.

5.4 Shared buffer

In two of the papers presented in related works [GSV21][NLG+15] isolation investigations splitting the buffer into multiple virtual queues are presented. However, the results showed that virtual queues by themselves did not provide performance isolation. The investigation in this thesis does not take the separation of the buffer into account. The buffer is not explicitly configured as part of the policies and is by default shared by both slices. As the evaluation does not study changes in traffic patterns for the critical slice, the buffer is never utilized by the critical traffic while any of the mechanisms are in use. However, if the buffer had been required by critical traffic during the evaluation this would lead to a breach of performance isolation. The buffer would need a specifically configured allocation between the slices to avoid such a breach. This configuration is available for Cisco devices through the policy-map configuration. Future work in this direction could demonstrate if the combination of a virtual queue and link capacity allocation can provide performance isolation in cases where the buffer is used by both traffic types.

5.5 Net neutrality

Net neutrality is the principle that all internet traffic should be treated equally, without any form of discrimination or preferential treatment by network operators and service providers. Implementing mechanisms as we have done in this thesis to achieve network slicing can be a violation of this principle and the use of such mechanisms might be unlawful according to EU regulations. If ISPs or network operators offer paid prioritization schemes, certain content providers can pay for preferential treatment of their traffic. Such practices could create an uneven playing field, favoring larger or wealthier entities while potentially disadvantaging smaller businesses or startups. Net neutrality regulation has been introduced to address these concerns. Regulations and policies governing net neutrality exist in EU regulation. However, as mentioned in [TCC23], a country can under the current European regulation adopt national regulation to enable national use of QoS, priority, and preemption mechanisms for example for a critical communication service.

5.6 UDP and TCP

Many applications rely on TCP for their communication needs due to its reliability and error handling. Conducting network tests only with UDP might not capture how these applications behave in real-world scenarios. Using solely UDP will overlook how other transport protocols handle congestion, manage packet loss, and adjust transmission rates to maintain optimal performance. TCP provides flow control mechanisms to regulate the rate of data transmission between sender and receiver based on network conditions. Without TCP's flow control, UDP tests will not accurately reflect how different network devices handle varying data rates and congestion situations. The generated traffic used for the measurements in this thesis is all UDP-based. The lightweight protocol design of UDP doesn't include mechanisms for error detection, retransmission of lost packets, or acknowledgment of received packets. Therefore, using only UDP flows might not accurately represent how data is transmitted in real-world scenarios. Mechanisms such as TCP congestion control which can have a significant impact on network performance are not taken into consideration. Furthermore, the effect of TCP phenomenons such as buffer bloat and TCP synchronization is not taken into account.

TCP traffic, due to its nature, should potentially be prioritized and handled differently compared to UDP traffic. Testing only with UDP would miss evaluating this type of differential treatment. While UDP is commonly used for real-time applications due to its lower overhead, it does not inherently manage latency or jitter. In summary, while UDP has its advantages in terms of lower overhead and suitability for certain types of applications like real-time streaming or gaming, relying solely on UDP measurements might not provide a comprehensive view of how the network handles various scenarios, especially those involving reliability, congestion management, and application behavior in diverse traffic conditions. Integrating tests with TCP and possibly other protocols can offer a more holistic understanding of network performance and behavior.

However, for conducting measurements focused on performance isolation, UDP traffic is sufficient. We have studied performance isolation at the data plane, at which level realistic traffic conditions are not necessary. Using UDP is useful for eliminating the complexity in TCP traffic behavior by specifically controlling the traffic load in the network. The traffic behavior for the evaluation in this thesis does not need to be realistic for the measurements to be valid, as performance isolation is not affected by the type of transport protocol, only the amount of packets.

Chapter 6

Conclusion

In this thesis, we have studied data plane resource allocation mechanisms in the context of network slicing. To answer our research questions we have designed a testbed that allows us to study two slices of network traffic on a common infrastructure. The testbed was used to conduct measurements and perform mechanism investigations.

The investigations presented in this thesis showcase the differences between the unique types of data plane resource allocation mechanisms that are available in Cisco hardware: Priority, policing, shaping, bandwidth, WRED, and tail drop. As Cisco routers do not provide room for dynamically configuring these mechanisms, they were implemented with predefined static allocation sizes. The evaluations illustrate the benefits and drawbacks of the mechanisms. Revealing which mechanisms perform optimally for use in an inter-slice resource allocation scenario.

Two goals of desirable inter-slice behavior were defined: Performance isolation and resource reallocation. With these goals in mind, seven network load scenarios were created, all with increasing degrees of congestion. During these scenarios, we captured and analyzed network performance. Performance isolation was quantified through measurements of throughput, packet loss, and latency. While resource reallocation was quantified through throughput and packet loss measurements. The results show that all mechanisms, except bandwidth, could provide performance isolation. The results show that the bandwidth mechanism is not able to provide latency performance isolation between slices during periods of congestion.

Furthermore, the results show that the bandwidth and priority mechanisms yield the best results for the reallocation of network resources between slices. These mechanisms can provide a maximized link utilization while maintaining configured performance for the critical slice even when configured statically. However, as noted above, the bandwidth mechanism was unable to provide performance isolation between the slices. Meaning that the priority mechanism is the only mechanism able to provide both performance isolation and resource reallocation. The priority mechanism is in the case of the Cisco ISR4221, limited in terms of scalability as it can only be configured at two levels: high and low.

6.1 Future work

In this thesis, we have conducted measurements on resource allocation mechanisms in a Cisco router and observed which mechanisms can provide performance isolation between two separate virtual networks. There are several paths to extend the work conducted in this thesis. The most prevalent one is to upgrade the hardware used in our testbed to a programmable switch and conduct a similar evaluation. This would contribute towards a more realistic testbed setup for SDN-based network slicing. In this case, the work could also be extended to include mechanisms for automation and dynamic resource reallocation.

Another path is to extend the network testbed to span multiple routers and/or switches to provide realism in the manner of larger networks. This direction could yield insights as to how multiple hops affect resource allocation, providing knowledge of how a controller or orchestrator can optimize network behavior across a larger network.

References

[3GP19] 3GPP. «Release 15 description; summary of rel-15 work items (release 15)». (2019), [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/21_series/21.915/ (last visited: Jun. 6, 2023).

[3GP23a] 3GPP. «Management and orchestration; concepts, use cases and requirements (release 17)». (2023), [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/28_series/28.530/ (last visited: Jun. 6, 2023).

[3GP23b] 3GPP. «Service requirements for the 5g system (release 19)». (2023), [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/22_series/22.261/ (last visited: Jun. 6, 2023).

[ATS+18] I. Afolabi, T. Taleb, *et al.*, «Network slicing and softwarization: A survey on principles, enabling technologies, and solutions», *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[Cisa] Cisco. «Chapter: Classification overview». (), [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios/qos/configuration/guide/12_2sr/qos_12_2sr_book/classification_oview.html (last visited: Jun. 6, 2023).

[Cisb] Cisco. «Chapter: Policing and shaping overview». (), [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/qos_plcshp/configuration/xe-3s/qos-plcshp-xe-3s-book/qos-plcshp-oview.html (last visited: Jun. 6, 2023).

[Cis17] Cisco. «Chapter: Configuring qos». (2017), [Online]. Available: https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst9400/software/release/16-6/configuration_guide/qos/b_166_qos_9400_cg/b_166_qos_9400_cg_chapter_01.html#concept_uzh_d2b_p1b (last visited: Nov. 10, 2023).

[Cis23] Cisco. «Compare traffic policy and traffic shape to limit bandwidth». (2023), [Online]. Available: https://www.cisco.com/c/en/us/support/docs/quality-of-service-qos/qos-policing/19645-policevsshape.html (last visited: Nov. 6, 2023).

[Doc23] Docker. «Docker networking overview». (2023), [Online]. Available: https://docs.docker.com/network/ (last visited: Jun. 6, 2023).

[DSM+18] M. Dighriri, A. Saeed Dayem Alfoudi, *et al.*, «Resource allocation scheme in 5g network slices», in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 2018, pp. 275–280.

[FJ93] S. Floyd and V. Jacobson, «Random early detection gateways for congestion avoidance», *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[FPEM17] X. Foukas, G. Patounas, *et al.*, «Network slicing in 5g: Survey and challenges», *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[GOH+20] A. J. Gonzalez, J. Ordonez-Lucena, *et al.*, «The isolation concept in the 5g network slicing», in *2020 European Conference on Networks and Communications (EuCNC)*, 2020, pp. 12–16.

[GSV21] A. Giorgetti, D. Scano, and L. Valcarenghi, «Guaranteeing slice performance isolation with sdn», *IEEE Communications Letters*, vol. 25, no. 11, pp. 3537–3541, 2021.

[HHTC05] H. Han, C. Hollot, *et al.*, «Synchronization of tcp flows in networks with small droptail buffers», in *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005, pp. 6762–6767.

[HMM+19] N. Huin, P. Medagliani, *et al.*, «Hard-isolation for network slicing», in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 955–956.

[HPD+21] H. Harkous, C. Papagianni, *et al.*, «Virtual queues for p4: A poor man's programmable traffic manager», *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2860–2872, 2021.

[Jun21a] Juniper. «Strict-priority queue overview». (2021), [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/cos-security-devices/topics/concept/cos-strict-priority-queue-security-overview.html (last visited: Nov. 6, 2023).

[Jun21b] Juniper. «Understanding cos tail drop profiles». (2021), [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/cos-ex/topics/concept/cos-ex-series-tail-drop-understanding.html (last visited: Nov. 6, 2023).

[Jun22] Juniper. «Understanding cos congestion management». (2022), [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/cos-ex/topics/concept/cos-congestion-management.html (last visited: Nov. 6, 2023).

[Jun23a] Juniper. «Controlling network access using traffic policing overview». (2023), [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/routing-policy/topics/concept/policer-overview.html (last visited: Nov. 6, 2023).

[Jun23b] Juniper. «Single token bucket algorithm». (2023), [Online]. Available: https://www.juniper.net/documentation/us/en/software/junos/routing-policy/topics/concept/policer-algorithm-single-token-bucket.html (last visited: Nov. 6, 2023).

[KEKG22] S. Kielland, A. Esmaeily, *et al.*, «Secure service implementation with slice isolation and wireguard», in *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2022, pp. 148–153.

[KR00] J. F. Kurose and K. W. Ross. «6.6 scheduling and policing mechanisms». (2000), [Online]. Available: http://www2.ic.uff.br/~michael/kr1999/6-multimedia/6_06-scheduling_and_policing.htm (last visited: Nov. 24, 2023).

[Lee14] G. Lee, «Chapter 3 - switch fabric technology», in *Cloud Networking*, G. Lee, Ed., Boston: Morgan Kaufmann, 2014, pp. 37–64. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128007280000035.

[Mas51] F. J. Massey, «The kolmogorov-smirnov test for goodness of fit», *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951. [Online]. Available: http://www.jstor.org/stable/2280095 (last visited: Dec. 15, 2023).

[MCR23] R. MacDavid, X. Chen, and J. Rexford, «Scalable real-time bandwidth fairness in switches», Jun. 2023. [Online]. Available: https://www.cs.princeton.edu/~jrex/papers/infocom23.pdf.

[MML+22] P. Medagliani, S. Martin, *et al.*, «Resource defragmentation for network slicing», in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2022, pp. 1–2.

[Net] NetworkLessons. «Wred (weighted random early detection)». (), [Online]. Available: https://networklessons.com/cisco/ccie-routing-switching-written/wred-weighted-random-early-detection (last visited: Nov. 10, 2023).

[net23] networkstatic. «Iperf3 docker build for network performance and bandwidth testing». (2023), [Online]. Available: https://hub.docker.com/r/networkstatic/iperf3 (last visited: Jun. 1, 2023).

[NGM15] NGMN. «5g white paper». (2015), [Online]. Available: https://www.ngmn.org/work-programme/5g-white-paper.html (last visited: Jun. 6, 2023).

[NLG+15] A. Nguyen-Ngoc, S. Lange, *et al.*, «Investigating isolation between virtual networks in case of congestion for a pronto 3290 switch», in *2015 International Conference and Workshops on Networked Systems (NetSys)*, 2015, pp. 1–5.

[Ris01] F. Risso, «Decoupling bandwidth and delay properties in class based queuing», in *Proceedings. Sixth IEEE Symposium on Computers and Communications*, 2001, pp. 524–531.

[SBKT16] M. R. Sama, S. Beker, *et al.*, «Service-based slice selection function for 5g», in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[SFVS20] G. Soós, D. Ficzere, *et al.*, «Practical 5g kpi measurement results on a non-standalone architecture», in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–5.

[Sha23] J. Shaw. «Pypcapkit: Comprehensive network packet analysis library». (2023), [Online]. Available: https://pypi.org/project/pypcapkit/ (last visited: Jun. 6, 2023).

[SVK+20] D. Scano, L. Valcarenghi, *et al.*, «Network slicing in sdn networks», in *2020 22nd International Conference on Transparent Optical Networks (ICTON)*, 2020, pp. 1–4.

[TCC23] TCCA. «Legal and regulatory aspects regarding the realisation of qpp in commercial networks». (2023), [Online]. Available: https://critcommsnetwork .com/documents/september-23-lrwg-qpp-white-paper (last visited: Nov. 24, 2023).

[YWB+21] Z. Yu, J. Wu, *et al.*, «Twenty years after: Hierarchical Core-Stateless fair queueing», in *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, USENIX Association, Apr. 2021, pp. 29–45. [Online]. Available: https://www.usenix.org/conference/nsdi21/presentation/yu.