electronics

Article

Traffic Classification for Network Slicing in Mobile Networks

Álvaro Gabilondo [1,2,*], Zaloa Fernández [1], Roberto Viola [1], Ángel Martín [1], Mikel Zorrilla [1], Pablo Angueira [2] and Jon Montalbán [3]

1 Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), 20009 San Sebastián, Spain; zfernandez@vicomtech.org (Z.F.); rviola@vicomtech.org (R.V.); amartin@vicomtech.org (Á.M.); mzorrilla@vicomtech.org (M.Z.)
2 Department of Communications Engineering, University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain; pablo.angueira@ehu.eus
3 Department of Electronic Technology, University of the Basque Country (UPV/EHU), 20018 San Sebastián, Spain; jon.montalban@ehu.eus
* Correspondence: agabilondo@vicomtech.org

Abstract: Network slicing is a promising technique used in the smart delivery of traffic and can satisfy the requirements of specific applications or systems based on the features of the 5G network. To this end, an appropriate slice needs to be selected for each data flow to efficiently transmit data for different applications and heterogeneous requirements. To apply the slicing paradigm at the radio segment of a cellular network, this paper presents two approaches for dynamically classifying the traffic types of individual flows and transmitting them through a specific slice with an associated 5G quality-of-service identifier (5QI). Finally, using a 5G standalone (SA) experimental network solution, we apply the radio resource sharing configuration to prioritize traffic that is dispatched through the most suitable slice. The results demonstrate that the use of network slicing allows for higher efficiency and reliability for the most critical data in terms of packet loss or jitter.

Keywords: network slicing; 5QI; traffic classification; nDPI; 5G network

Citation: Gabilondo, Á.; Fernández, Z.; Viola, R.; Martín, Á.; Zorrilla, M.; Angueira, P.; Montalbán, J. Traffic Classification for Network Slicing in Mobile Networks. *Electronics* **2022**, *11*, 1097. https://doi.org/10.3390/electronics11071097

Academic Editors: Bin Han, Simon Pietro Romano and Patrick Seeling

Received: 28 January 2022
Accepted: 29 March 2022
Published: 30 March 2022

1. Introduction

Network slicing is a promising technique that has the potential to accommodate individual traffics and satisfies the quality-of-service (QoS) requirements for the data flows of specific applications in a 5G network [1]. When applied to 5G networks, slicing can be performed at different levels [2]: at the radio segment, allowing for an efficient and dynamic allocation of radio resources; in the multi-access edge computing (MEC) resources, when services are pushed to the edge and need smart hosting to balance performance–cost trade-offs; in the network core, when traffic isolation is required to ensure security; or during the deployment of end-to-end solutions, when multi-domain/site or multi-user vendor platforms come into play in private networks.

In recent years, in both the industrial and automotive sectors, the use of Internet of Things (IoT) connections has increased due to its wide range of applications. Moreover, this increase is expected to reach around 50% of all devices and connections worldwide by 2023 [3]. However, these connections result in cyber-physical scenarios where sensorized data are turned into actionable data, meaning that asymmetric communications will dominate and vast amounts of information will be available on uplinks (UL).

For these sectors, the 5G Infrastructure Public Private Partnership (5G PPP) [4], and the manufacturing (5G-ACIA [5]) and automotive (5GAA [6]) clusters have provided all the key performance indicators (KPIs) for representative applications of each use case, including enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable low-latency communications (URLLC) [7,8].

In this context, the negotiation and control messages and the actual data flows may have different communication needs, so the required QoS granularity goes beyond the sensor device or the application itself. Additionally, the management of individual data flows becomes a major concern for Industrial Internet of Things (IIoT) and telematic control unit (TCU) architectures, in which wired or wireless sensors and systems use gateways that concentrate data flows that are pushed to distant systems or to the Internet. Thus, a single device could simultaneously produce or consume data flows of different types with specific QoS demands, beyond individual devices or specific applications.

Furthermore, due to the variety of applications, protocols, and business logics in private networks within the industrial and automotive sector, the ability to classify individual data flows is essential. Therefore, instead of mapping application flows or individual devices, granularity can be used to classify individual data flows in these environments. In this way, it becomes logical to assign a higher priority to control messages over other data flows where the statistics of the traffic may provide a clear view of the criticality of short messages sent from time to time (likely URLLC), on the bandwidth required for data flows sending high volumes of packets all the time (likely eMBB) and for the high density of messages from multiple parties sending frequent short messages (likely mMTC).

Due to the asymmetry of sensorized communications as well as the existence of flows with different QoS requirements, the use of radio access network (RAN) slicing may be beneficial [9].

The objective is to empower the RAN to become aware of the traffic features of individual data flows, enabling their classification and their dynamic assignment to existing slices. To this end, the major contributions of our work include two original approaches to handling the different types of traffics when applying the network slicing technique in UL. These contributions can be summarized as follows:

- On the one hand, a mechanism based on a traffic classifier is used to identify the class of individual data flows. This allows for subsequent slicing to be dynamic, adaptive and transparent to end users by being able to transmit all types of traffic.
- On the other hand, the proposed approaches assign, in real time, the data flow to one of the available slices associated with a 5G QoS identifier (5QI) value from [10], according to specific policies. Here, the data flow classes considered are the main ones present in the industrial and automotive sectors, including control messaging, video streaming, IoT data, and generic web data.

The contribution of this paper goes beyond the simple use of traffic classification and network slicing techniques. A dynamic and transparent allocation of slices at the radio level was carried out in a real 5G network deployment. This real deployment also marks a difference from other proposed state-of-the-art solutions, which remain at the level of simulation. In order to evaluate the performance of the proposed solutions, a scenario was defined in which UL communications are saturated in a real testbed of a 5G standalone (SA) network. The results were also compared with a reference scenario where no slicing was applied.

The rest of this paper is organized as follows. First, related work is presented in Section 2, covering the concept of network slicing, the traffic types, and the methods of classification. In Section 3, the most relevant software packages for a 5G experimental network with network slicing support are described. Then, Section 4 presents the RAN slicing solution and the algorithms involved in its formulation. Furthermore, the evaluation setup, the system performance analysis, the defined metrics, and the results of the experiment carried out are described in Section 5. Finally, the results are discussed in Section 6, and Section 7 concludes the paper.

2. Background and Related Work

2.1. Network Slicing

Network slicing is a technique introduced by the NGMN (Next Generation Mobile Network) in [11] and allows for multiple virtual networks to run in a single common physical infrastructure in an efficient and cost-effective manner, while satisfying different sets of QoS. A network slice is defined as a set of available resources (network, computation, and radio) assigned to a virtualized network service to satisfy specific requirements associated with the service. In turn, network slicing consists of defining a set of policies and mechanisms to identify the traffic flows associated with each slice. Taking this into account, a common physical infrastructure provides access to multiple slices simultaneously in order to satisfy the QoS of devices, users, or applications, as shown in Figure 1.

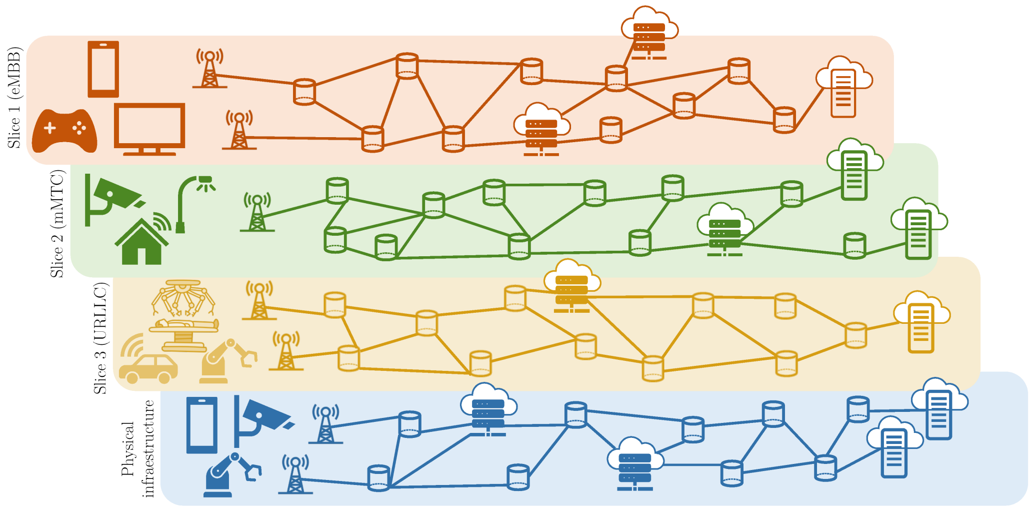


Figure 1. 5G network slicing architecture.

Currently, each user equipment (UE) element can be served by eight slices, and the identifier for each slice is called the Single Network Slice Selection Assistance Information (S-NSSAI), where an NSSAI is a collection of these slices. This identifier is employed by the 5G Core (5GC), the 5G-RAN, and the UE elements of a 5G network.

In the same way, each identifier consists of two parts. First, the slice/service type (SST) identifies the slice in terms of its characteristics and services. Second, the slice differentiator (SD) allows for differentiating multiple slices from the same SST. While the SST parameter is mandatory, the SD is optional and is used to differentiate between two slices with the same SST. Moreover, some SST standard values have been set by the 5G use cases, as Table 1 shows.

Table 1. Standardized SST values [10].

SST	Slice/Service Type
1	eMBB
2	URLLC
3	Massive IoT (MIoT)
4	Vehicle to everything (V2X)
5	High-Performance Machine-Type Communications (HMTC)

The 5G end-to-end network slicing requires this concept to be applied in the radio, edge, core, and transport networks.

From the core network perspective, network slicing mainly provides the possibility of deploying multiple instances of virtual 5GCs concurrently on a single common physical infrastructure [12]. Each of these instances is configured to satisfy different service requirements. In 5G networks, the design of the core networks is implemented as virtualized network functions (VNF), following the network function virtualization (NFV)/software-defined network (SDN) paradigm [13]. Moreover, NFV and SDN are the pillars that enable network slicing in 5G networks and create slicing in a transport network segment to meet the requirements and the QoS requirements of the application and services [14].

From the RAN point of view, the network slicing concept is implemented considering dynamic resource management mechanisms. These mechanisms allow for an efficient and dynamic allocation of radio resources, i.e., through smart schedulers at the media access control (MAC) level. To match the requirements, this resource allocation must consider the KPIs of the different services/applications that are served by the slice.

In order to exemplify the application of slices in the previously defined network segments on the 5GC, the edge, and the transport network segments, the CPU resources, the network topology used, or the traffic characteristics can be considered, i.e., each slice can allocate CPU resources or obtain its topology. Moreover, each traffic type can be associated with a slice isolated from the rest of the network [15]. In contrast, on the 5G RAN segment, the available bandwidth resources can be assigned efficiently among different slices while considering their requirements.

Furthermore, in terms of resources, there are three available options concerning the slice isolation: (i) fully shared resources, (ii) partially shared resources, and (iii) completely dedicated resources [12,16]. Here, different slices are isolated as long as actions performed on one slice do not affect the performance of another.

In the case of completely dedicated resources, each slice has a set of radio resources assigned in the control and user plane as well as in the MAC-level scheduler and in the radio spectrum. In this case, each slice has access to a percentage of dedicated physical resource blocks (PRBs), which are defined as minimum units of resources that a base station can allocate to a UE, guaranteeing both traffic isolation between slices and that the QoS requirements (e.g., delay and capacity constraints) are met. However, this reduces flexibility as it will not allow resources to be moved from one slice to another, and the resources will be wasted if a slice is not used.

In this paper, we focus on the adoption of fully shared resources for network slicing in the RAN segment. In the fully shared resource option, all the slices share the radio spectrum, the MAC-level scheduler, and the 5G network control plane. The PRBs are managed by a common scheduler, which allocates resources to each slice according to the requirements and QoS required by their services, applications, or use cases. It should be noted that neither the traffic isolation nor the target QoS levels are ensured in the fully shared resources option. However, since the resources are shared among all slices, it is a more flexible option, since it allows PRBs to be dynamically allocated according to the needs of the application.

With this in mind, several EU initiatives have focused on defining 5G network slicing architectures. Some examples include 5G NORMA [17], SliceNet [18], and 5GTANGO [19]. In the case of 5G NORMA, a multi-tenant and multi-service 5G system architecture based on the concept of network slicing is proposed [20]. SliceNet aims to achieve end-to-end network slicing through control, management, and orchestration mechanisms [21]. In the case of 5GTANGO, a network slicing resource allocation and monitoring framework over multiple clouds and networks was created [22]. In contrast to the focus of this paper, which is RAN slicing, all of these initiatives focused more on the application of network slicing using SDN and NFV, ignoring the radio paradigm.

Several papers have considered the challenges and research issues raised due to the application of the network slicing concept in a 5G network [1,12,13,16,23–26]. Early contributions related to the concept of network slicing were focused on LTE [27], but this trend has shifted in recent years to 5G communications. However, it must be underlined that current network slicing in 5G networks has mainly been carried out and evaluated in the context of the 5GC segment, thus neglecting the radio segment, which is the focus of this paper [28]. Furthermore, many of the contributions at the RAN level focus on theoretical analyses or simulation-level evaluations [29–32]. Thus, the evaluation of the solutions is not performed in a real 5G architecture.

One of the open issues is the lack of analysis adapted to types of data flows in order to define the resources required in each slice, according to the characteristics of the type of traffic to be transmitted. For this reason, the focus of this paper is the map of data flow types to specific 5QI slices and its evaluation in a real 5G network with network slicing techniques at the RAN level. It should be noted that the solution presented in this paper is also in line with the core challenges described in [33].

2.2. Traffic Types

This section describes the types of traffic widely considered in the literature; the characteristics are summarized in Table 2 [34] and can be described as follows:

- Video traffic is the streaming of video data between endpoints. This type of traffic is loss-tolerant but sensitive to delays in real-time streaming.
- Audio/voice traffic is the transmission of audio data between endpoints. Similar to video traffic, this type of traffic is loss-tolerant but sensitive to delays in real-time streaming.
- IoT traffic is the streaming of cyber-physical data and information about the environment collected by hundreds or thousands of devices and transmitted periodically. This kind of traffic is not very loss-tolerant. However, ensuring a high number of connections while guaranteeing bandwidth and latency is very important.
- WebData traffic is the amount of data sent and received by visitors to a website. The typical size of the data is variable, as it mainly depends on the type of website from which the data originates. By contrast, the main focus for this traffic is on bandwidth.
- Data transfer traffic refers to the transmission of files of varying sizes between endpoints. For this type of traffic, the most important requirement is the reliability of the content upon reception, with longer delays being more tolerable than for other types of traffic.
- Control data traffic contains network control messages. These messages are usually produced in low volumes with very low packet sizes but have strong delivery requirements, for example, a very long delay or the network control frames experiencing a loss can lead to loss of network functions.
- Best-effort traffic comprises traffic that has no specific requirements to be fulfilled.

Table 2. Traffic-type characteristics [34].

	Traffic Characteristics				
Traffic	**Tolerance to Loss**	**Network Requirement Guarantee**	**Packet Size (Bytes)**	**Criticality**	**Periodicity**
Video	Yes	Latency	High (1000∼1500)	Medium	Periodic
Audio/Voice	Yes	Latency	Variable (20∼1500)	Medium	Periodic
IoT	No	Nº of connections	Low (50∼500)	Low	Periodic
WebData	No	Bandwidth	Variable (30∼1500)	Low	Sporadic
Data Transfer	No	Bandwidth	Variable (30∼1500)	Low	Sporadic
Control Data	No	Latency	Ultra low (30∼150)	High	Periodic
Best-Effort	Yes	None	Variable (30∼1500)	None	Sporadic

2.3. Traffic Classification

This section describes the automatic traffic classification process, the different groups into which the algorithms can be divided, and the set of open-source alternatives considered in this paper. Choosing the proper traffic classification algorithm is indispensable to obtaining a proper and efficient traffic classification.

In general, the traffic classification process carried out by these algorithms involves four steps. First, the traffic coming from the network is collected to form a dataset. The second step extracts and selects particular features from the dataset obtained in the previous step. The third step takes into account the features obtained in the previous step to identify the traffic category. To do so, it uses patterns or model training in conjunction with machine learning algorithms. Finally, in the last step, the obtained results are verified.

Taking this into account, the algorithms can be divided into five main groups [35]:

- Statistics-based classification: this uses statistical information related to the traffic without analyzing the packet payload. Algorithms belonging to this group have a high computational overhead as an analysis of heuristics for individual packets is required.
- Correlation-based classification: this uses the correlation of flows combined with the statistical information of the traffic. As in the case of the statistics-based classification group, these also have a high computational overhead.
- Behavior-based classification: this uses host interaction and connection data to classify traffic. Despite the good results in terms of accuracy and the lightweight processing demand, the traffic classification result does not provide much detail.
- Payload-based classification: this uses the content of the payload or some specific fields of the payload to perform traffic classification. Therefore, the accuracy obtained by these algorithms is the highest among the groups mentioned in this paper. This group can be further divided into deep packet inspection (DPI) algorithms and stochastic packet inspection (SPI) algorithms. The difference between them is the way in which they inspect the content of the packet. While DPI algorithms are generally not able to classify encrypted packets, SPI algorithms can deal with encrypted traffic. Both have high computational overheads.
- Port-based classification: this uses only the port to classify the traffic. This method is the least accurate among all of the above groups since many traffic sources use dynamic ports or two traffic sources may use the same port to transmit data with different protocols. Therefore, in these cases, the results of the classification may not be correct.

Considering the aforementioned advantages and disadvantages of the algorithms listed, DPI is the most accurate classification algorithm, with an acceptable computational burden. Therefore, this paper focuses on DPI algorithms. Many products, both commercial and open-source, rely on some form of DPI to perform the classification process. This paper emphasizes the use of open-source alternatives because of the flexibility and low cost offered during implementation. More precisely, the paper focuses on five open-source tools: L7-filter, OpenDPI, Libprotoident, nDPI, and NFStream.

L7-filter [36]: this classifier identifies packets based on application layer data to determine which protocols are being used, thus complementing classifiers that match IP addresses and port numbers. This classifier is more demanding in terms of processing and memory than others, so its use is recommended only when it is necessary to map protocols that use unpredictable ports, to match traffic on non-standard ports, or to distinguish between protocols that share a port. However, the project has now been discontinued and is considered closed.

OpenDPI [37]: this is an open-source version of an early version of PACE, a commercial classifier, which has been stripped of encrypted protocols, making it a more limited classifier with fewer supported protocols and slower performance. As in the previous case, this project is closed and considered obsolete.

Libprotoident [38]: this classifier, similar to the others mentioned in this section, performs payload-based classification, but with one relevant difference. This tool examines only the first four bytes of the payload in each direction. This minimizes the storage capacity needed to store the packet traces, as well as the computational load required for classification. Payload pattern matching, payload size, port numbers, and IP address matching are used when performing the classification.

nDPI [39]: this is an extension of OpenDPI, with further optimization of both performance and speed, as well as a larger number of protocols, including encrypted protocols with the addition of a decoder for a secure sockets layer (SSL). It is based on a traffic classification library using both packet header and payload. Moreover, it allows for decoding open system interconnection (OSI) layers 3 and 4 of the packets to improve the classification. It is a project with constant updates and a large community. In addition, the output of the results is presented in a simple form for further processing.

NFStream [40]: this is based on the nDPI classifier, although it is based on Python, unlike previous tools that are based on C or C++. This nDPI-based classification allows NFStream to perform a reliable identification of encrypted applications and metadata fingerprinting. In addition, this classifier is flexible and allows for the creation of new features by adding lines and Python code in a simple way.

A comparison of the multiple classifiers presented is shown in Table 3.

Table 3. Comparison between different DPI algorithms [41].

Name	Released	Updates	Language	Apps/Protocols Identified
L7-filter	2009	Deprecated	C++	∼110
OpenDPI	2011	Deprecated	C	∼100
Libprotoident	2013	2–3 quarter	C++	∼250
nDPI	2014	Few days	C	∼170
NFStream	2019	Few days	Phyton	∼180

Although the OpenDPI and L7-filter projects were abandoned and their development stopped several years ago, they have been included in this paper for reference because many scientific papers base their results and conclusions on these two classifiers.

Among all the options mentioned above, the open-source classifiers that dig deep into the packets for classification are nDPI and NFStream. Both of them allow a large number of protocols to be classified, including those necessary for the types of traffic studied in the previous section, such as Real Time Protocol (RTP) for Video, Message Queue Telemetry Transport (MQTT) for IoT, and Hypertext Transfer Protocol (HTTP) for WebData. They also have a large community that offers support when problems arise and are updated from time to time. However, although NFStream is somewhat more flexible in terms of data output, the nDPI classifier displays more detailed information and allows for better processing.

3. Network Slicing Solutions

In this section, we briefly describe the most relevant software for deploying 5G SA experimental networks in which a network slicing deployment can be performed. Table 4 shows the analysis of solutions for testbeds.

Table 4. Analysis of network slicing solutions.

Name	Description	Pros	Cons
Open5GCore [42]	Commercial software for 5GC [43]	Allows virtualization of the software	The same level of bandwidth, throughput, latency, etc., for each slice. So, no difference between slices. License needed
Open5Gs [44]	Open-source software for 5GC with C language	Allows for virtualization of the software. The number of UEs per slice is not limited	Only 5GC deployment without RAN
Amarisoft [45]	Commercial products for 5GC, RAN, and UEs	Supports virtualization depending on the license purchased. Allows end-to-end 5G network deployment with network slicing	Radio network slicing is applied in the scheduler of the base station called gNodeB (gNB) through traffic prioritization

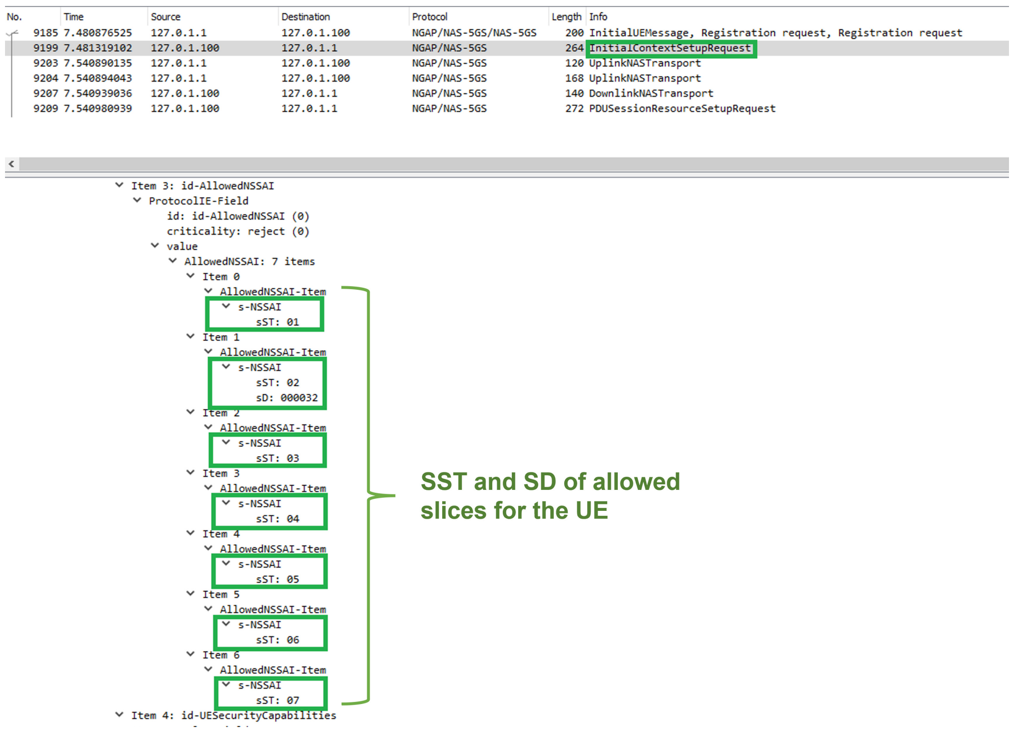
Other relevant open-source solutions for academic and research communities include 5G OpenAirInterface (OAI) [46] and free5GC [47]. Nevertheless, although they allow for the deployment of 5G SA networks—end to end in the case of 5G OAI and only the core system in the case of free5GC—in contrast to the other software solutions mentioned above, they cannot currently implement functional network slicing.

Considering the previous analysis, the commercial software Amarisoft has been selected as it supports network slicing based on traffic prioritization at the RAN level and offers a 5GC, a RAN, and UEs—all with network slicing support—unlike the other solutions mentioned above.

Furthermore, the deployment of the network slicing offered by Amarisoft includes a mechanism at the gNB scheduler that prioritizes traffic—both UL and downlink (DL)—based on the 5QI associated with each slice. The 5QI is an indicator of a set of QoS characteristics such as the priority level, packet delay, packet error rate, etc. These QoS characteristics can either be standardized or non-standardized. The standardized 5QI values can be seen in [10], while the non-standardized 5QI can be used for ad hoc configurations. Moreover, the lower the priority value assigned by the 5QI, the higher the priority of the slice.

For the application of network slicing in the RAN segment, not only should the RAN be aware of the characteristics and parameters of each of the slices but the UE must also know them. Therefore, the UE needs to discover both the identifiers of the slices (SST and SD) and the 5QI parameter of each of them, which will be used for the prioritization of different slices. For this, the sequence of messages for discovering these parameters is as follows:

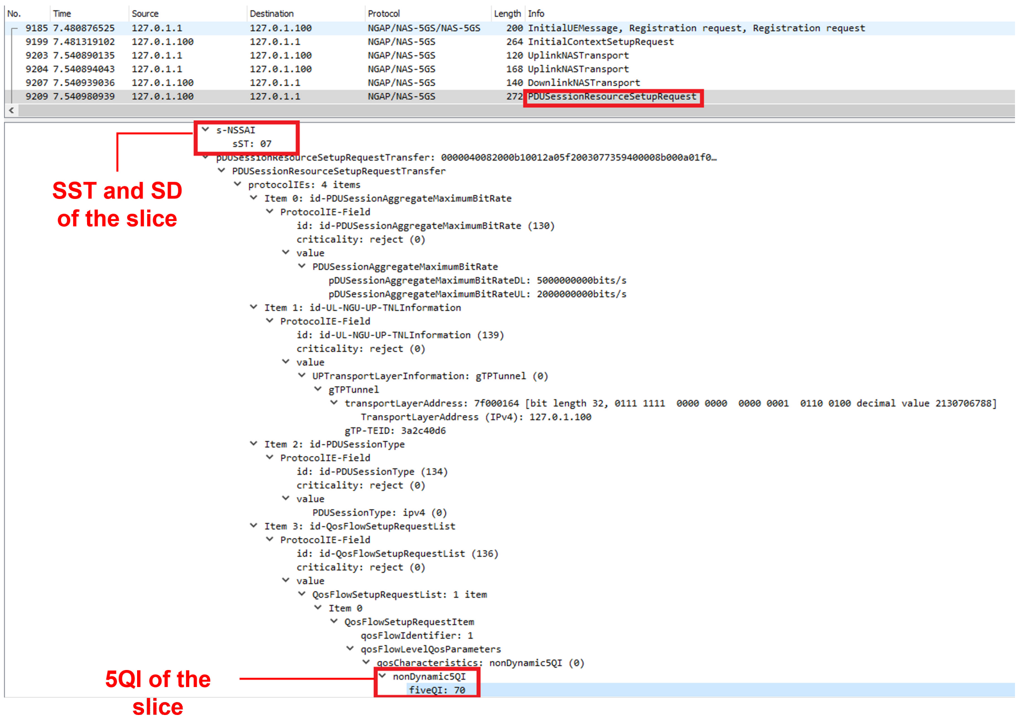
1. The context is required to find out which slices are in the 5GC database for that specific UE. These slices, as shown in Figure 2, are identified with their corresponding SST and SD using the NG Application Protocol (NGAP) *Initial Context Setup Request message*.
2. The connection to a slice is established, as in Figure 3—in this example, a slice with an SST 7 and an associated 5QI of 70—using a message *PDU Session Resource Setup Request*, where the UE receives the 5QI parameter of that slice. Therefore, to compile all of the 5QI parameters of the available slices, it would be necessary to previously connect to each of them.



```
No.    Time          Source        Destination   Protocol             Length Info
  9185 7.480876525   127.0.1.1     127.0.1.100   NGAP/NAS-5GS/NAS-5GS    200 InitialUEMessage, Registration request, Registration request
  9199 7.481319102   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            264 InitialContextSetupRequest
  9203 7.540890135   127.0.1.1     127.0.1.100   NGAP/NAS-5GS            120 UplinkNASTransport
  9204 7.540894043   127.0.1.1     127.0.1.100   NGAP/NAS-5GS            168 UplinkNASTransport
  9207 7.540939036   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            140 DownlinkNASTransport
  9209 7.540980939   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            272 PDUSessionResourceSetupRequest

  Item 3: id-AllowedNSSAI
    ProtocolIE-Field
      id: id-AllowedNSSAI (0)
      criticality: reject (0)
      value
        AllowedNSSAI: 7 items
          Item 0
            AllowedNSSAI-Item
              s-NSSAI
                sST: 01
          Item 1
            AllowedNSSAI-Item
              s-NSSAI
                sST: 02
                sD: 000032
          Item 2
            AllowedNSSAI-Item
              s-NSSAI
                sST: 03
          Item 3
            AllowedNSSAI-Item
              s-NSSAI
                sST: 04
          Item 4
            AllowedNSSAI-Item
              s-NSSAI
                sST: 05
          Item 5
            AllowedNSSAI-Item
              s-NSSAI
                sST: 06
          Item 6
            AllowedNSSAI-Item
              s-NSSAI
                sST: 07
  Item 4: id-UESecurityCapabilities
```

Figure 2. *Initial Context Setup Request* message with the allowed SST.



```
No.    Time          Source        Destination   Protocol             Length Info
  9185 7.480876525   127.0.1.1     127.0.1.100   NGAP/NAS-5GS/NAS-5GS    200 InitialUEMessage, Registration request, Registration request
  9199 7.481319102   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            264 InitialContextSetupRequest
  9203 7.540890135   127.0.1.1     127.0.1.100   NGAP/NAS-5GS            120 UplinkNASTransport
  9204 7.540894043   127.0.1.1     127.0.1.100   NGAP/NAS-5GS            168 UplinkNASTransport
  9207 7.540939036   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            140 DownlinkNASTransport
  9209 7.540980939   127.0.1.100   127.0.1.1     NGAP/NAS-5GS            272 PDUSessionResourceSetupRequest

  s-NSSAI
    sST: 07
  pDUSessionResourceSetupRequestTransfer: 0000040082000b10012a05f2003077359400008b000a01f0…
    PDUSessionResourceSetupRequestTransfer
      protocolIEs: 4 items
        Item 0: id-PDUSessionAggregateMaximumBitRate
          ProtocolIE-Field
            id: id-PDUSessionAggregateMaximumBitRate (130)
            criticality: reject (0)
            value
              PDUSessionAggregateMaximumBitRate
                pDUSessionAggregateMaximumBitRateDL: 5000000000bits/s
                pDUSessionAggregateMaximumBitRateUL: 2000000000bits/s
        Item 1: id-UL-NGU-UP-TNLInformation
          ProtocolIE-Field
            id: id-UL-NGU-UP-TNLInformation (139)
            criticality: reject (0)
            value
              UPTransportLayerInformation: gTPTunnel (0)
                gTPTunnel
                  transportLayerAddress: 7f000164 [bit length 32, 0111 1111  0000 0000  0000 0001  0110 0100 decimal value 2130706788]
                    TransportLayerAddress (IPv4): 127.0.1.100
                  gTP-TEID: 3a2c40d6
        Item 2: id-PDUSessionType
          ProtocolIE-Field
            id: id-PDUSessionType (134)
            criticality: reject (0)
            value
              PDUSessionType: ipv4 (0)
        Item 3: id-QosFlowSetupRequestList
          ProtocolIE-Field
            id: id-QosFlowSetupRequestList (136)
            criticality: reject (0)
            value
              QosFlowSetupRequestList: 1 item
                Item 0
                  QosFlowSetupRequestItem
                    qosFlowIdentifier: 1
                    qosFlowLevelQosParameters
                      qosCharacteristics: nonDynamic5QI (0)
                        nonDynamic5QI
                          fiveQI: 70
```

Figure 3. *PDU Session Resource Setup Request* message with the selected SST and associated 5QI.

For more information on the message chain for making connections in 5G networks, please refer to the technical report found in [48].

4. Proposed Approaches for Network Slicing

4.1. Target Scenario

The approach proposed for the application of network slicing in this paper is explained below. The approach presented here uses network slicing in a scenario where a UE is assumed to be a generic modem/gateway, which receives different types of traffic from external sources and connects to the Internet through the RAN. This approach allows 5G features to be used in multi-source and scalable scenarios such as factories, office buildings, etc., as well as in scenarios with difficulties in bridging wired connections.

Figure 4 shows a scenario where multiple types of traffic, from multiple external sources such as devices connected to a Wi-Fi router, computers connected to the subnet, machines in factories, etc., are connected to a UE that relays all of the traffic to the Internet through a RAN. However, for the external sources, this connection to the 5G network, as well as the use of network slicing and the traffic classification, will be transparent. Traffic will be sent to the same point, i.e., the same UE, regardless of the type of traffic, and it will be the mechanism deployed that is responsible for the transmission of the traffic to the 5G network.

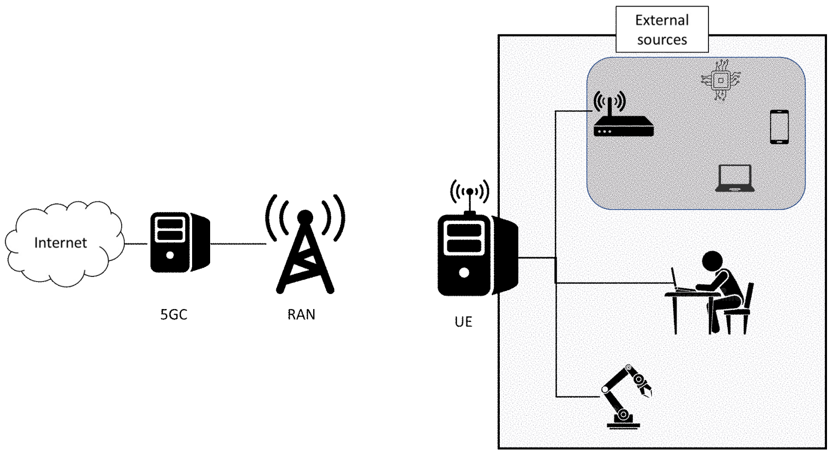


Figure 4. Target scenario scheme.

4.2. Radio Slicing Mechanism

The scope of the approach in this paper focuses on radio network slicing in the UL, although it could also be used for DL traffic. Here, different types of traffic come from external sources, while the UE has no a priori knowledge of the types of traffic that will be forwarded. Therefore, the traffic classification mechanism is applied to select the slice that meets the traffic needs for each data flow during communication. The workflow of this dynamic slice selection mechanism is based on three steps:

1. The UE receives all types of traffic and forwards them to the Internet through the main interface.
2. The classifier runs on the main interface and classifies the different traffic flows.
3. The slicing policy selects the best slice based on the previous classification. Accordingly, the UE sends the traffic through the chosen slice.

The selection of the slice for the type of traffic is dynamic, so the workflow continuously iterates the second and third steps.

In turn, the selection and re-transmission of content varies, depending on the approach used. This paper presents two original approaches for the selection and transmission of traffic over a slice. In the first approach, the UE connects to just one slice and the classifier is used to classify all traffic reaching the UE as a whole to decide the appropriate slice. Based on this information, the algorithm selects the best slice, connects to it, and disconnects from the previous one. In the second approach, the UE initially connects to all available slices and the algorithm uses the classifiers to identify individual traffic types and then redirects all packets through the selected slice for each type of traffic.

The following sections explain these proposed traffic-forwarding approaches.

4.2.1. Approach 1: Dynamic Slice Selection

The first approach to performing slice selection is described in Algorithm 1. This approach focuses on dynamically changing the slice depending on the dominant type of traffic in all the data flows gatewayed by the UE. This dominant type of traffic refers to the type of traffic with the highest number of packets detected by the classifier in each analysis.

Algorithm 1 Algorithm for dynamic slice selection.

```
procedure SLICESELECTION( $\{traffic_{map}\}_k$ )           ▷ Dynamic slice selection
Input: $\{traffic_{map}\}_k = \{\{type\}_k, \{5qi\}_k\}$      ▷ Mapping of target traffic types and 5QI
    connectUE($slice_0$)                                 ▷ Connect UE to default slice
    slice == $slice_0$                                   ▷ Save current slice
    {slice}$_n$ ← getAllSlices()                         ▷ Get $SST$ and $SD$ of all available slices
    for all {slice}$_n$ do                               ▷ Loop over $n$ slices
        5QI$_i$ ← connectUE($slice_i$)                   ▷ Connect UE to $slice_i$ & get attached 5QI
        {5QI}$_i$ = 5QI$_i$                              ▷ Store 5QI from $slice_i$
        disconnectUE($slice_i$)                          ▷ Disconnect UE from $slice_i$
    end for
    for all $\{5qi\}_k$ do                               ▷ Loop over $k$ target 5QI
        for all {5QI}$_n$ do                             ▷ Loop over $n$ available 5QI
            if $\{5qi\}_i$ == {5QI}$_j$ then             ▷ Check if $\{type\}_i$ has available slice
                {type$^s$}$_m$,{slice$^s$}$_m$ ← add($i$)   ▷ Traffic is supported by a slice
            else
                default($i$)                             ▷ Assign traffic as Best-Effort/default
            end if
        end for
    end for
    while true do
        type$_{main}$ ← performClassifier($t$)           ▷ Get dominant traffic after $t$ seconds
        if hasChanged(type$_{main}$) then                ▷ Check dominant traffic shift
            for all {type$^s$}$_m$ do                    ▷ Loop over $m$ supported types
                if type$_{main}$ == {type$^s$}$_i$ then  ▷ Check support of dominant type
                    connectUE($slice_i$)                 ▷ Connect UE to new slice
                    disconnectUE(slice)                  ▷ Disconnect UE from previous slice
                    slice == $slice_i$                   ▷ Current slice
                    break
                end if
            end for
        end if
    end while
end procedure
```

In the first step, the UE connects to the default slice, defined as the one with the lowest priority. Then, through the messages sent by the RAN in the DL, the UE obtains the identifiers of each slice (SST and SD) allowed for the UE. Once the identifiers are collected, the UE connects to each slice, obtaining the 5QI parameters and then disconnecting from the slice.

In a second step, the algorithm takes the traffic types and their 5QI parameters, which were initialized by the user, and compares these 5QI parameters with those obtained from the slices. Accordingly, the algorithm decides whether this type of traffic is supported by the UE.

Finally, the classifier is launched in a loop to continuously analyze the traffic. When the algorithm detects that the dominant traffic has changed, it connects the UE to the appropriate slice, depending on the 5QI parameter, and disconnects from the previous one.

This approach makes it possible to adapt the UE to increased traffic. This is effective in scenarios where there are large numbers of packets of heterogeneous traffic. However, in situations where there is priority traffic but with a small number of packets, this approach is not effective, because those packets, even if they are captured by the classifier, will not have enough weight to adapt to the slice change and could be hidden. For this reason, the second approach was proposed.

4.2.2. Approach 2: Dynamic Traffic Redirection

This second approach was developed after observing the drawbacks of the approach explained above. In this approach, the UE initially connects to all possible slices simultaneously, i.e., those included in the *Initial Context Setup Request* message. Then, after a learning period in which the algorithm identifies all types of traffic passing through the UE, it can redirect each traffic independently through the appropriate slice. This way, if high priority traffic arises, it can be delivered without experiencing congestion by existing traffic in the network. This approach, focusing on dynamic redirection based on traffic type, is described in Algorithm 2.

Algorithm 2 Algorithm for dynamic traffic redirection.

```
  procedure SLICESELECTION( $\{traffic_{map}\}_k$ )      ▷ Dynamic slice selection
Input: $\{traffic_{map}\}_k = \{\{type\}_k , \{5qi\}_k\}$      ▷ Mapping of target traffic types and 5QI
    connectUE($slice_0$)      ▷ Connect UE to default slice
    $\{\text{slice}\}_n \leftarrow$ getAllSlices()      ▷ Get $SST$ and $SD$ of all available slices
    for all $\{\text{slice}\}_n$ do      ▷ Loop over $n$ slices
        $\text{5QI}_i \leftarrow$ connectUE($slice_i$)      ▷ Connect UE to $slice_i$ & get attached 5QI
        $\{\text{5QI}\}_i = \text{5QI}_i$      ▷ Store 5QI from $slice_i$
    end for
    for all $\{5qi\}_k$ do      ▷ Loop over $k$ target 5QI
        for all $\{\text{5QI}\}_n$ do      ▷ Loop over $n$ available 5QI
            if $\{5qi\}_i == \{\text{5QI}\}_j$ then      ▷ Check if $\{type\}_i$ has available slice
                $\{\text{type}^s\}_m, \{\text{slice}^s\}_m \leftarrow$ add($i$)      ▷ Traffic is supported by a slice
            else
                default($i$)      ▷ Assign traffic as Best-Effort/default
            end if
        end for
    end for
    while true do
        $\{\text{type}_{class}\}_h \leftarrow$ performClassifier($t$)      ▷ Get all traffic after $t$ seconds
        for all $\{\text{type}_{class}\}_h$ do      ▷ Loop over $h$ present classes
            for all $\{\text{type}^s\}_m$ do      ▷ Loop over $m$ supported types
                if $\{\text{type}_{class}\}_i == \{\text{type}^s\}_j$ then      ▷ Check support of traffic type $i$
                    redirectTraffic($\{\text{slice}^s\}_j$)      ▷ Redirect the traffic through the $\{\text{slice}^s\}_j$
                    break
                end if
            end for
        end for
    end while
  end procedure
```

Similarly to the first algorithm, in the first step the UE connects to the default slice and, through the messages sent by the RAN in the DL, the UE obtains the SSTs and SDs of all of the slices allowed for this UE. With them, the UE connects to each slice and obtains its 5QI parameters. The main difference of this approach is that the UE keeps these connections alive. In a second step, the algorithm takes the traffic types declared, initialized by the user with their corresponding 5QI parameters, and then the algorithm compares them with the parameters obtained from the slices to decide whether the traffic type can be supported by the UE.

Finally, the classifier is launched by capturing all the traffic that is gatewayed through the default slice. For this traffic, the algorithm considers whether it is a new traffic type or one that has not appeared before and then redirects it to the appropriate slice, to which the UE is already connected.

As can be seen in the pseudo-code shown, in this approach, each type of traffic is redirected as it is classified, so for the best performance, training is preferably conducted at the beginning, where the types of traffic passing through the UE are classified and redirected. However, although this approach needs a larger number of resources, i.e., as many IP addresses as slices, it solves the problem of the previous approach by allowing for the transmission of individual traffic types over the appropriate slice with different levels of prioritization, regardless of the number of packets or their size.

Initially, this second approach showed greater control and accuracy than the first one in scenarios with heterogeneous traffic; it treats each type of traffic passing through the UE independently, allowing it to accurately adapt to situations with heterogeneous traffic types, such as mixed IoT device environments with Streaming traffic and Control traffic for network and device control. However, in situations where networks do not have many resources, such as networks with few available IP addresses, or in scenarios where it is not known what type of traffic is used, the first approach is a better option.

5. Performance Evaluation

5.1. Experimental Setup and Evaluation Metrics

This section presents the setup deployed to evaluate the different solutions proposed. Figure 5 shows the setup deployed in the laboratory, which consisted of the following:

- An Amarisoft Callbox Pro device in order to deploy the Core and RAN segments of the 5G SA network;
- An Amarisoft UE Simbox unit, which allows for the deployment of multiple simultaneous 5G UEs in the same hardware;
- A virtual instance, which remotely manages the UEs offered by Amarisoft acting as a traffic injector in the 5G network and the various algorithms that perform the selection of the slice or the dynamic traffic redirection through the corresponding slice. In our setup, the virtual instance was hosted on an OpenStack server.

Regarding the environment where the setup was deployed, this was a closed environment (a room) where there was a direct view between the RAN and the UE, which were 5 m apart.

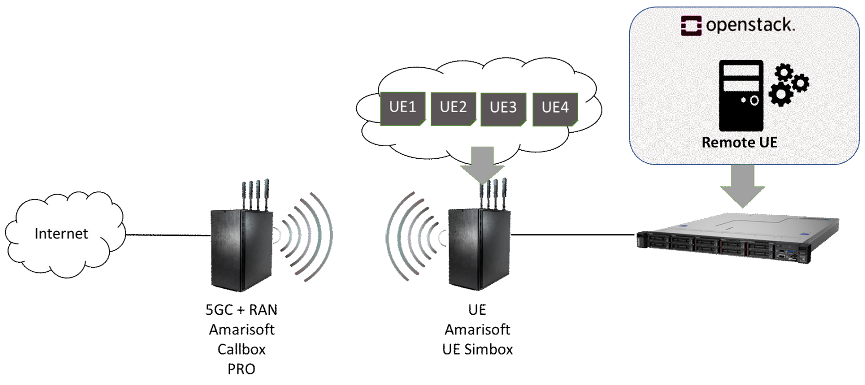


Figure 5. Experimental setup in the laboratory.

The evaluation of the proposed algorithms for the slice selection, along with the traffic classification algorithm, was carried out using Amarisoft software for the whole 5G network, i.e., 5GC, RAN, and UEs. Regarding the classifier, for all of the reasons explained in Section 2.3, as well as for the comparison made in [41,49], the nDPI traffic classifier was selected. In turn, five types of traffic were defined: Best-effort/default traffic, Control data traffic, Video traffic, IoT traffic, and WebData traffic. We selected these types of traffic due to, among all the reasons explained in Section 2.2, our laboratory capabilities, since these are the five that we could generate. Each type of traffic was assigned a slice (defined by its SST ID) associated with a 5QI.

As previously stated, the 5QI is a high-level indication of the QoS. At present, there is no one-to-one mapping or clear explanation in the 3GPP requirements on how Packet Delay Budget, Packet Error Rate, Default Maximum Data Burst Volume, Default Averaging Window—which are the parameters taken into account when defining the standardized 5QI values in [10]—should be implemented. Therefore, in the case of Amarisoft, the way the 5QI is translated into useful parameters is based on implementation choices. For this purpose, Amarisoft allows the experimenter to adjust the 5QI configuration to the target traffic types.

Since in this paper we are focused on validating the application and operation of the slices at RAN level, we decided to take the predefined 5QI configurations of Amarisoft and to modify the priority level at the MAC layer for each of them, mapping the defined 5QI to a slice. In the Amarisoft equipment, the priority level spans from 1 to 16. Therefore, we assigned a higher priority (1) to the slice where the most critical traffic is transmitted, in our case, the Control data traffic, and a lower priority (16) to the slice where the least critical traffic is transmitted, in our case, the Best-effort/default traffic. The characteristics of the five types of traffic considered, as well as the associated slices and 5QI values can be observed in Table 5.

Regarding the IP addresses shown in Table 5, these refer to established protocol data unit (PDU) sessions. In the tests carried out, these sessions were established at a rate of one per slice, so that when sending the different types of traffic through the corresponding slice, these IP addresses of the PDU sessions, associated with the specific slice, were used .

Table 5. Selected traffic per slice.

Slice/ SST ID	5QI	Traffic Type	Protocol	Periodicity	Priority	IP Address of the PDU Session
1	5	Control	DNS/ ICMP	Variable	1 (Ultra high)	192.168.3.6
2	6	Video	RTP/ RTCP	25 frames per second	5 (High)	192.168.3.10
3	9	IoT	MQTT	0.5 s	9 (Medium)	192.168.3.14
4	8	WebData	HTTP/ HTTPS	Variable	13 (Low)	192.168.3.18
7	70	Best-effort/ default	Non specified	Variable	16 (Ultra low)	192.168.3.2

These types of traffic were generated in our setup through different tools and software. For Control data traffic, the Ping tool was used to generate Internet Control Message Protocol (ICMP) requests and responses; for Streaming, the FFmpeg software was used to transmit a video from the UE; for IoT, MQTT messages were transmitted using the open-source Mosquitto tool; and finally, for WebData, the Wget tool was used to download content through the HTTP protocol. Meanwhile, the rest of the traffic was considered Best-effort/default traffic.

Moreover, to obtain a proper and efficient traffic classification, it is important to determine the volume of packets to collect from the network to conclude an accurate classification while also quickly adapting to traffic-type shifts. The time interval is dependent on the application or protocol to be detected. In the case of the nDPI traffic classifier, a single packet is usually necessary to determine most User Datagram Protocol (UDP)-based protocols widely employed in the industry and in automotive applications. However, this is not always the case. According to the literature [50], there is a rule stating that only eight packets per direction allow the nDPI algorithm to accurately identify the protocol or application. In our tests, carried out for the five types of traffic defined in Table 5, it was concluded that using 1 s as the window of time for the evaluation algorithm (the minimum collection interval allowed by the nDPI traffic classifier) was enough to accurately detect traffic, but depending on the protocol/application, more time may be necessary for correct detection. Since in our scenario this collection time is not critical for the correct operation of the slicing selection algorithms, it was established that the traffic classifier would collect traffic at an interval of 10 s, to ensure high efficacy and accuracy in the decisions.

To evaluate the approaches, the following experiment was performed. The experiment was based on measuring the network saturation and the performance of the different slices presented in Table 5. These saturation and performance measurements were carried out using network metrics such as the following:

- Bandwidth, which allows for the level of saturation suffered by each slice to be known;
- Percentage of packets lost, which provides a measure of the reliability, complementing the bandwidth metric;
- Jitter, which allows the level of stability of the different scenarios to be known and for which the higher its value, the greater the time variations in packet reception.

To carry out the measurement of these metrics, the active measurement tool Iperf [51] was used to obtain the maximum achievable bandwidth in IP networks. This tool was selected because it is a popular command-line tool used to diagnose network problems by measuring the maximum network throughput. In order to use it to achieve network saturation and to obtain metrics, each traffic is first generated using the aforementioned tools (Ping, FFmpeg, MQTT and Wget). In this way, the proposed approaches characterize the traffic and map it to the corresponding slices. Subsequently, a series of redirection rules are added, only for this saturation test, due to the fact that the saturation of a network with real IoT traffic is not easy because they have a small packet size. With these rules, when the Iperf tool sends packets to the same ports used by the already-characterized traffic, these data flows are redirected to the previously mapped slices. In this way, the traffic sent by Iperf can be treated as Control, Streaming, IoT and WebData traffic.

The measurements were captured for three scenarios:

- A scenario in which there is no network slicing, as an initial reference scenario that outlines a baseline performance and behavior;
- A scenario where each UE is associated with a single slice, representing the scenario of the first approach;
- A scenario where each UE is connected to all available slices, representing the scenario of the second approach.

In order to conducted the saturation test using the Iperf tool, in each scenario, four Iperf servers were deployed in the RAN element, each one in the port associated with each characterized traffic. Regarding the Iperf clients, in both the first and second scenarios, an Iperf client was deployed in each of the UEs, attacking different ports. In the last scenario, the same number of Iperf clients were deployed in each UE, attacking a different port in each one, with the approach, together with the added rules, being responsible for redirecting the traffic through each slice.

After measuring the bandwidth offered by the experimental 5G network in UL, the UEs achieved around 105 Mbps of available bandwidth to be shared among all those connecting. Taking this and the fact that network slicing is analyzed on the four slices mentioned in Table 5 into account, about 20 Mbps was transmitted continuously for each slice. Therefore, initially, the network was not saturated. This saturation was performed for each slice independently in different steps. The default slice was not taken into account because it transmits Best-effort traffic.

The duration of the test was set at 5 min and consisted of saturating the different slices one by one, adding 20 more Mbps every 20 s to observe the behavior of the rest of the slices in each case. Table 6 shows how many Mbps per slice were transmitted for each time slot.

Table 6. Bandwidth transmitted in the saturation test.

Time (s)	**Bandwidth (Mbps)**			
	Slice 1	**Slice 2**	**Slice 3**	**Slice 4**
0–20	20	20	20	20
20–40	40	20	20	20
40–60	60	20	20	20
60–80	80	20	20	20
80–100	100	20	20	20
100–120	20	20	20	20
120–140	20	40	20	20
140–160	20	60	20	20
160–180	20	80	20	20
180–200	20	20	20	20
200–220	20	20	40	20
220–240	20	20	60	20
240–260	20	20	20	20
260–280	20	20	20	40
280–300	20	20	20	20

This test was performed ten times to minimize any randomness and was carried out in the three mentioned scenarios. The number of repetitions was selected after observing that, even with a larger number of tests, the variability was not appreciable. Therefore, we concluded that these ten tests were sufficient to obtain real and accurate results in each scenario. In the first scenario, which works as a reference, the network slicing technique was not used; therefore, four different UEs were connected to the RAN and congested one by one, according to the methodology already explained. In the second scenario, the same four UEs were connected to the RAN. Here, each one was associated with a different slice, which was congested depending on the prioritization of the slices. Finally, in the last scenario, one UE was connected to the same four slices of the previous scenario, congested in the same manner as in the second scenario.

5.2. Experimental Results

In this section, the performance of the two proposed approaches described in Section 4 in terms of bandwidth, packet loss, and jitter are evaluated using the saturation test described in the UL. As mentioned, to compare the results, a reference scenario in which network slicing was not considered was also evaluated. All evaluations were performed on a 5G SA network, as stated previously.

5.2.1. Bandwidth

Figures 6–8 show the bandwidth obtained every second for each slice and each scenario. Figure 6 shows the bandwidth measured by four equal UEs in a scenario where the network slicing concept was not applied, i.e., where all users and traffic had the same priority. As can be seen, the performance of each user was the same; all users share the bandwidth and when a user, regardless of who they are, asks for more, they can use the extra bandwidth available. In the same way, if a saturation situation is reached, the rest of the users are not affected, as seen throughout Figure 6, in which 20 Mbps are guaranteed in each slice.

In this first scenario, although the total bandwidth shared by all the UEs should be around 105 Mbps, as mentioned above, it can be seen that when a UE transmits more than the initial 20 Mbps transmitted per slice and, at some points of the test, exceeds that theoretical limit, only 6–7 Mbps can be obtained in addition to the initial 20 Mbps transmitted. This implies that the total bandwidth of the network is around 85–90 Mbps. The reason for this behavior is that, when sharing bandwidth between multiple users, considering how scheduling works in 3rd Generation Partnership Project (3GPP) systems, it is not as simple as dividing the radio resources equally. At the same time, it is necessary to take into account that the tests were carried out in a wireless environment using an experimental 5G network deployment, where there is more instability to consider than in a wired environment.

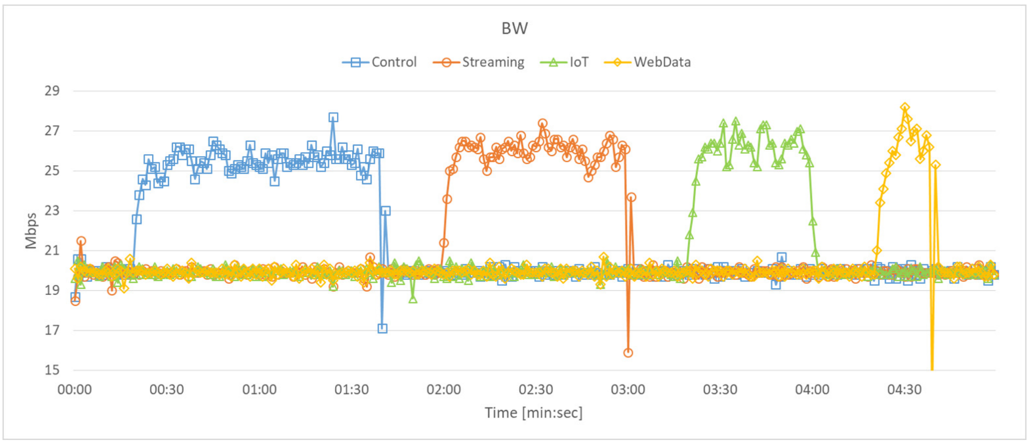


Figure 6. Bandwidth obtained from scenario without prioritization.

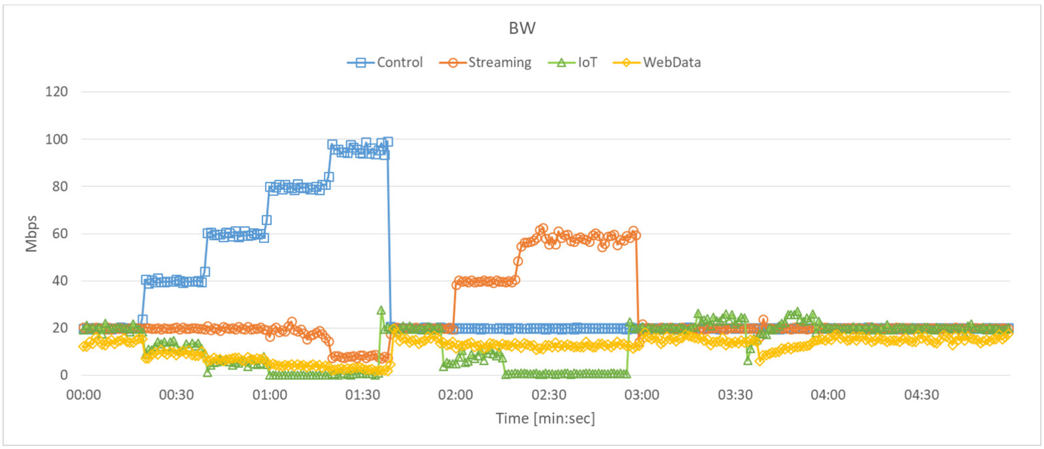


Figure 7. Bandwidth obtained from the scenario using the first approach.

Figure 7 also represents the bandwidth measured in the same four UEs but in a scenario where the network slicing concept was applied. More specifically, each UE is associated with a different slice. In this figure, it can be seen that the performance of each user is different. The user with the highest priority, associated with the Control slice, has the expected performance and can request more bandwidth by taking bandwidth away from the other UEs/slices. When the user with the second-highest priority, connected to the Streaming slice, requests 40 and 60 Mbps (from 2:00 to 2:20 and from 2:20 to 2:40 min, respectively), they take them from the third and fourth users, connected to the IoT and WebData slices, respectively, without affecting the highest priority slice. However, when they ask for 80 Mbps (from 2:40 to 3:00 min), they cannot obtain more bandwidth as the traffic has become congested. Regarding the third slice, when a user asks for more bandwidth, they take away part of the lowest priority slice, which is only about 30 Mbps in the best case. For fourth slice, when a user asks for more bandwidth, they cannot take any from the first, second, or third slices since these have higher priorities.

This behavior, where the second slice cannot ask for more than 80 Mbps and the third and fourth slices cannot ask for more bandwidth, provides the same response as the previous scenario, since having to share the bandwidth among several UEs makes it impossible to achieve the full nominal bandwidth of 105 Mbps. The only case where a slice comes close to reaching this score is in the highest priority slice, as this can take bandwidth from the others. However, even this situation has some instabilities when transmitting 100 Mbps and cannot maintain this level of transmission.

Finally, Figure 8 shows the bandwidth obtained in the third scenario, where a single UE is connected to the same four slices used in the second scenario, i.e., Control, Streaming, IoT and WebData. This figure clearly shows the behavior of traffic prioritization in the different slices. It can be seen that, when each slice requires more bandwidth, it obtains the bandwidth from lower priority slices. Thus, the Control slice reaches the saturation limits as the channel in 100 Mbps, once it obtains bandwidth from other slices. The Streaming slice obtains its bandwidth from the IoT and WebData slices, at a maximum of 80 Mbps, but it does not affect the bandwidth of the Control slice. The IoT slice obtains its bandwidth from the WebData slice, with a maximum of 60 Mbps, without affecting the Control and Streaming slices. Finally, the WebData slice is the most unstable because it can only access available bandwidth from the throughput, not that in use. Therefore, it can only obtain 40 Mbps at most.

As can be seen, in this case, a bandwidth of more than 100 Mbps is obtained, since, unlike the other scenarios, this bandwidth is dedicated to a single UE, acting as a gateway for the different data flows distributed among the different slices.

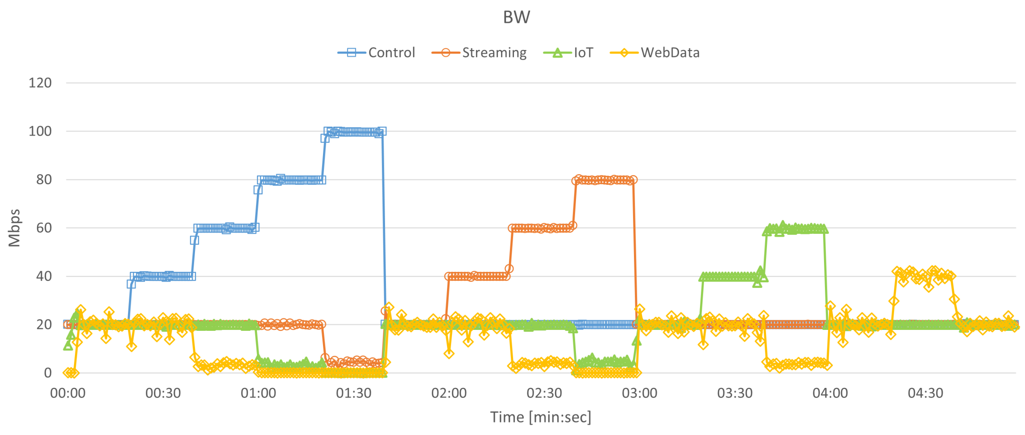


Figure 8. Bandwidth obtained from the scenario using the second approach.

5.2.2. Packet Loss

The figures shown in this subsection (Figures 9–11) show the percentage of packets lost from each slice in each of the scenarios. However, at some specific times, when the different slices are severely congested, long delays occur and the reception is erratic when measuring packet loss; therefore, fewer packets are measured as sent. This makes the results difficult to interpret, mainly in the lower priority slices such as IoT and WebData. Due to this, and in order to obtain a more uniform representation for comparison, smoothing was carried out by obtaining and modeling the average value of the results for each slice every 5 s.

Figure 9 presents the packet losses in the first scenario without applying the network slicing concept. In this figure, it can be seen that the losses increase for each user each time that the user requests more bandwidth. This is consistent with the results shown in Figure 6. As mentioned above, in this scenario, each user cannot obtain more than 26–27 Mbps in total, so when a user requests 40 Mbps, the losses for that user are around 30% and the others are not affected. When one of the users asks for 60 Mbps, the losses are around 55%; when the demand increases to 80 Mbps, the losses are around 66%, and when the user asks for 100 Mbps, the losses are around 75%. This behavior is the same for all users.

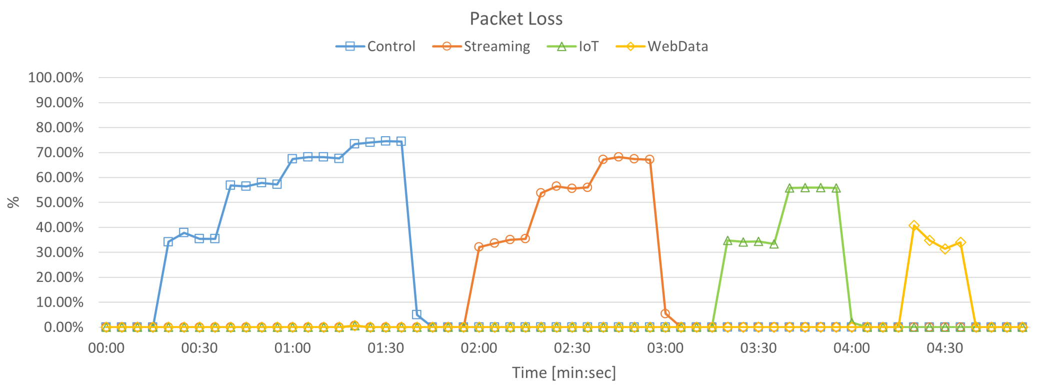


Figure 9. Packet loss obtained from the scenario without prioritization.

Figure 10 shows the packet loss in the second scenario, where each user is connected to a different slice. First, regarding the representation of IoT and WebData, we observe that there are intervals (1:00–1:40 for IoT and 2:40–03:00 for WebData) in which, although the slices are saturated, the packet losses are lower than those for other higher priority slices, such as for the Streaming slice in the first interval or for the IoT slice in the second interval. This is because, although the losses indicate a lower percentage, the total number of packets received in the slices during those intervals is also lower, implying a higher loss. An example of this behavior is at 1:20 in the Streaming slice. In this case, there is a loss percentage of 53% over a total of 1506 datagrams received, while in the IoT slice, there is a loss percentage of 33% over a total of 66 datagrams received. This shows that the total saturation of the higher priority slice means that many of the packets transmitted by the lower priority slices do not even reach the Iperf server, thus increasing packet losses. It is worth remembering that both slices try to send 20 Mbps during this saturation condition in this time interval.

Taking this into account when analyzing this case, it can be seen that the losses are also consistent with the behavior shown in Figure 7, which shows the results related to bandwidth in this second scenario. It can be seen that, as a user becomes saturated, the other lower priority users lose more packets, starting with the lowest priority user. It is also worth noting the percentages of packet loss for the Control slice at 1:20–1:40, for the Streaming slice at 2:40–3:00, for the IoT slice at 3:20–4:00, and for the WebData slice at 4:20–4:40 are due to the aforementioned sharing of available bandwidth between more than one user.

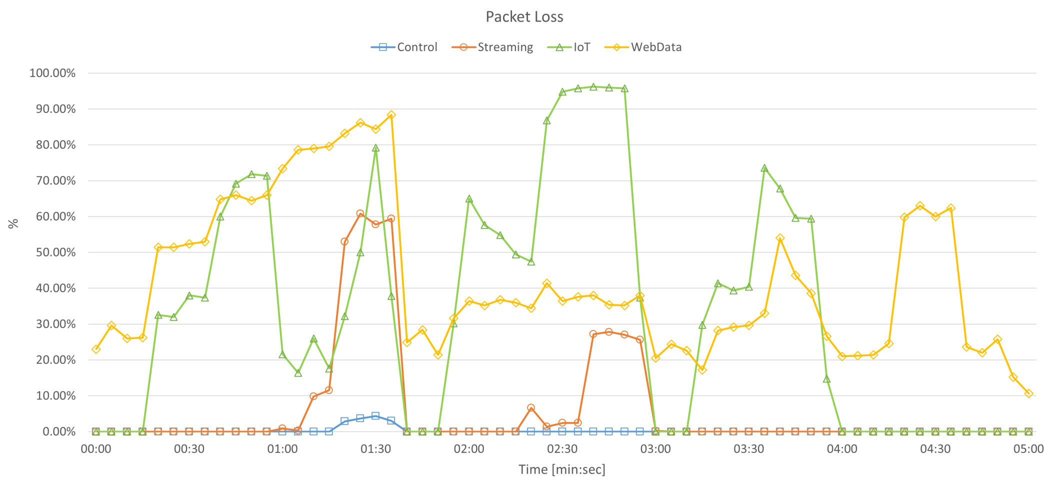


Figure 10. Packet loss obtained from the scenario using the first approach.

Finally, Figure 11 shows the packet losses in the third scenario, where one user is connected to four slices. In this last scenario, it can be observed that when a slice, regardless of its priority, demands more bandwidth, the packet losses affect only the slices that have a lower priority. Therefore, in this case, unlike the other scenarios, the slice asking for more bandwidth never loses packets. This allows the highest priority slice to achieve high stability and reliability by not being saturated at any point in the test. Additionally, as in the previous scenario, at a higher saturation level, the lower priority slices detect fewer packets and perceive fewer packets as being lost. As an example, at 1:30, the Streaming slice has 77% losses out of 1923 datagrams detected, while IoT and WebData have 72% and 52% losses out of 136 and 33 datagrams detected, respectively.

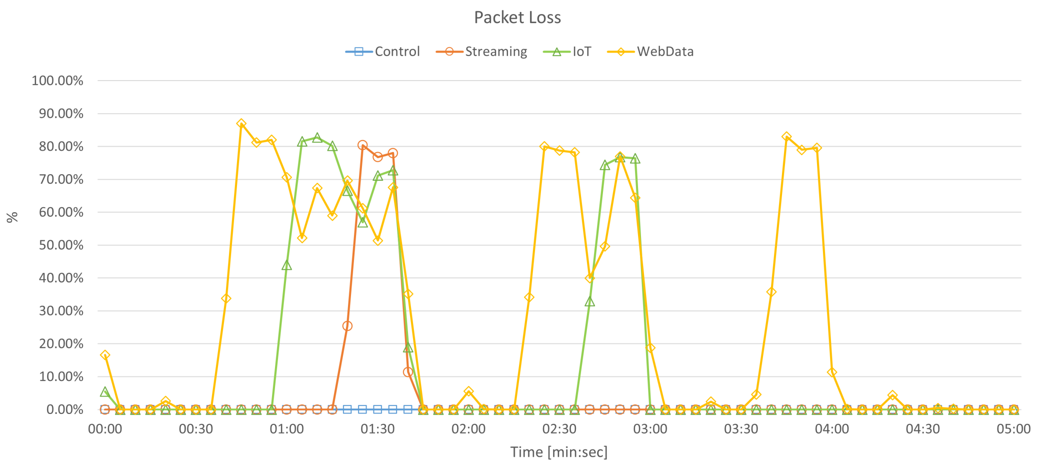


Figure 11. Packet loss obtained from the scenario using the second approach.

5.2.3. Jitter

In this last part of the results, Figures 12 and 13, as well as Tables 7 and 8 show comparisons between the three scenarios considered. The comparisons focus on the jitter obtained for both the Control and Streaming slices. The jitter obtained refers to the variation in delays in arrival between packets. It should also be noted that these results only show comparisons for the two highest priority slices, because the other slices have very high jitter values—more than 100 ms when higher priority slices demand bandwidth—with high variations, due to all the saturation moments they suffer. Therefore, we considered it relevant to only show the jitter obtained for the cases with more critical traffic.

Table 7 presents the mean jitter values obtained numerically for the Control slice, highlighting the intervals where the slice requests a greater bandwidth, and Figure 12 plots the mean, median, and deviation values of the jitter obtained in each 20 s interval for the Control slice in each of the three scenarios. It can be seen that all of the values shown are low, never reaching 3.5 ms of jitter. Regarding the first scenario, it can be observed that during the interval where the Control user demands more bandwidth, the jitter worsens, increasing its mean and deviation values and reaching values of 2.435 ms, but it remains stable throughout the rest of the test, with most of its mean values between 0.8 and 0.9 ms. However, Figure 12 shows that there are many atypical points outside the range of the deviation during the whole test. Concerning the second scenario, i.e., the first approach, the highest median, mean, and deviation values are obtained, with atypical points higher than 2.5 ms. This leads to unstable behavior, especially when it is compared with other scenarios, where most of the mean values are between 1.4 and 1.7 ms. However, this range of values can still be considered stable. Additionally, during the period when more bandwidth is needed, it can be seen that the prioritization carried out reduced the mean values of jitter in each interval by approximately 0.83 ms. Finally, the last scenario, i.e., the second approach, shows that the values are very stable throughout the test, with most of the values maintained between 1.1 and 1.2 ms. As with the first approach, when the greatest bandwidth is requested, the jitter is reduced in each interval, obtaining the best results among all scenarios.

Table 7. Mean jitter values measured in each scenario in the Control slice.

Times (s)	**Without Prioritization (ms)**	**First Approach (ms)**	**Second Approach (ms)**
0–20	0.86865	1.69075	0.8868
20–40	1.3488	1.3746	0.8973
40–60	1.75745	1.21645	0.8373
60–80	2.1286	0.9007	0.7713
80–100	2.43515	0.8378	0.79385
100–120	0.85055	1.5392	0.9122
120–140	0.83605	1.6702	1.03025
140–160	0.86075	1.4451	1.1007
160–180	0.9496	1.4847	1.1278
180–200	0.8391	1.70245	1.20035
200–220	0.82355	1.30005	1.1501
220–240	0.85865	1.48455	1.1113
240–260	0.83575	1.50085	1.108
260–280	0.83655	1.7163	1.13495
280–300	0.89005	1.64653	1.25858

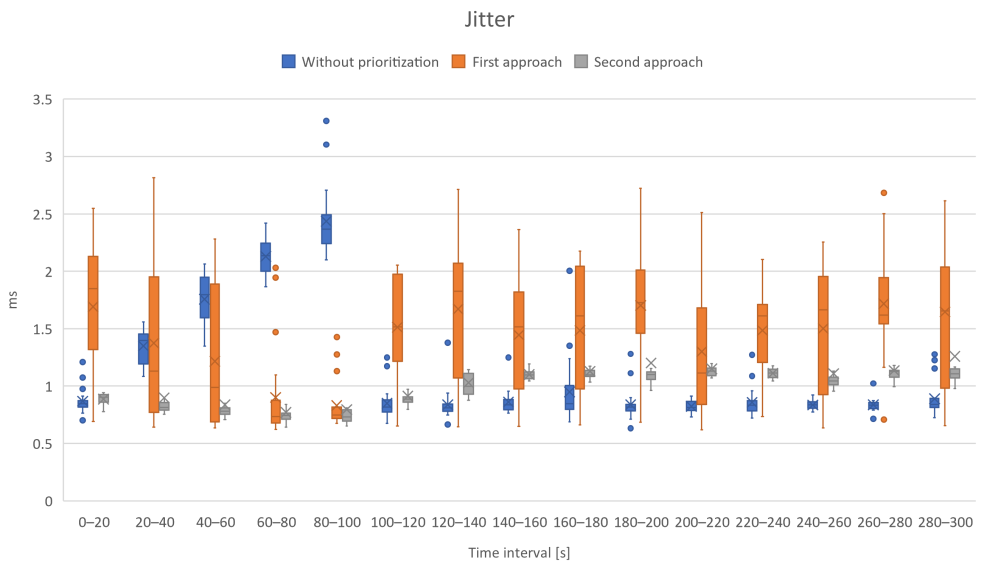


Figure 12. Comparison of the jitter obtained between the approaches considered (Control traffic slice).

As in the previous case, Table 8 presents the mean jitter values obtained numerically in the Streaming slice, highlighting the intervals where the slice requests a greater bandwidth, and Figure 13 shows the mean, median, and deviation values of the jitter for the Streaming slice. In the first scenario, it can be seen that the values remain stable, with values between 0.8 and 0.9, and increase slightly in the intervals corresponding to the moment of saturation of the slice. On the other hand, in both the first and the second approaches, these values are stable throughout the whole test—in the same way as in the Control slice and taking into account that the mean, median, and variance values are wider in the first approach—with values for the second scenario between 1.3 and 1.9 ms and values for the third scenario between 1 and 1.2 ms. The only moment when it can be seen that both approaches lose stability and have much higher mean jitter values is in the interval where the highest priority slice, i.e., the Control slice, requests practically all of the available bandwidth and therefore affects the Streaming slice (80–100 s interval). Finally, it is worth noting that during the period in which more bandwidth is needed, that the mean jitter values in each interval are also reduced.

Table 8. Mean jitter values measured in each scenario in the Streaming slice.

Times (s)	Without Prioritization (ms)	First Approach (ms)	Second Approach (ms)
0–20	0.8396	1.681	0.8614
20–40	0.8445	1.65525	0.9488
40–60	0.80605	1.92275	1.0087
60–80	0.82645	1.69285	1.3972
80–100	0.94615	2.8613	5.6984
100–120	0.90295	1.3117	0.89665
120–140	1.2251	1.17465	0.88205
140–160	1.7204	1.0637	0.84345
160–180	1.8813	1.39505	0.81155
180–200	0.8682	1.58875	1.13335
200–220	0.82765	1.37105	1.03115
220–240	0.83425	1.5474	1.05305
240–260	0.8319	1.6727	1.12325
260–280	0.80655	1.33285	1.0661
280–300	0.86074	1.37711	1.20058

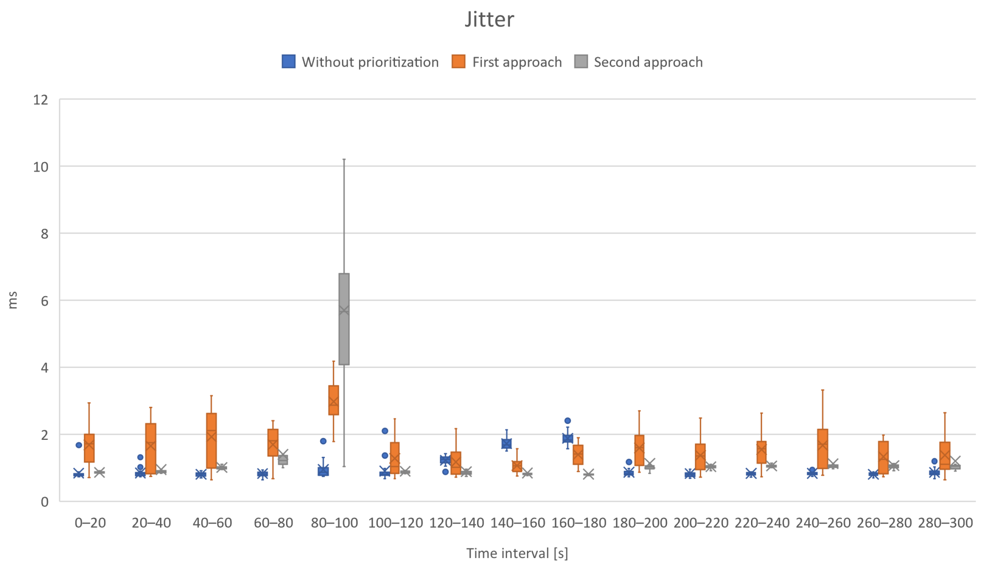


Figure 13. Comparison of the jitter obtained between the approaches considered (Streaming traffic slice).

6. Discussion

Taking into account the results obtained in Section 5, a comparison was made between the three scenarios considered.

Before going into the details, we show how the use of the network slicing concept clearly obtains better results in terms of guaranteeing the transmission of critical traffic such as Control traffic, which corresponds to the highest priority slice in this paper. It is clearly seen that, regardless of the network slicing approaches proposed in this paper—scenarios in which a single UE is associated with a single slice (first approach) or scenarios in which each UE is connected to all available slices (second approach)—the result is satisfactory for the higher priority slices. This applies to the three metrics obtained, which are bandwidth, packet loss, and jitter.

If we focus on the obtained metrics, it is clear that if we seek to guarantee that all types of traffic, regardless of their characteristics, can obtain an equitable amount of traffic transmitted, the scenario in which no network slicing approach is contemplated is the most suitable. Despite guaranteeing bandwidth for all UEs involved in the traffic transmission, the percentage of packet losses and jitter will inevitably be high in those UEs that want to transmit information that exceeds the saturation level of the network. That is why this approach is not valid for industrial or automotive use cases, which require the most critical data to be received correctly and with reduced jitter, regardless of other types of lower priority traffic trying to be sent [52,53].

Therefore, in these types of applications or use cases, in which the satisfactory sending of critical traffic prevails over non-priority traffic, there is a benefit from the use of either of the two network slicing approaches proposed in this paper, as observed in the results. Both network slicing approaches are able to prioritize the highest priority traffic over the rest of the traffic. Nevertheless, in all senses, i.e., in the maximum bandwidth obtained, the percentage of packet loss, and the amount of jitter, the second approach obtained better results. It should also be remembered that the first approach is not effective in situations where there is priority traffic but with a small number of packets, as the traffic may be hidden under the low priority but dominant type of traffic. Once again, for this type of application, the second approach is the most effective, because a UE is considered to be a generic modem/gateway, which receives different types of traffic from external sources and is able to treat them separately by correctly assigning the corresponding slice to the traffic category.

Taking this into account, regarding bandwidth, even though the traffic transmitted based on the highest priority slice in the two network slicing approaches proposed in this paper manages to reach almost 105 Mbps, which is the maximum bandwidth provided by our experimental 5G SA network—the scenario without network slicing does not even reach 30 Mbps—in the case of the first approach, the rest of the slices do not reach the maximum bandwidth possible. For example, the second slice, corresponding to Streaming traffic, only reaches 60 Mbps in the first approach, compared with the 80 Mbps reached by this slice in the second approach. This difference between the bandwidths obtained is even greater for the other lower priority slices, which is a favorable result in the case of the second approach.

Regarding the packet losses, the most remarkable observation obtained from the comparison of these three scenarios is that, with the second approach, no packet is lost in the highest priority slice throughout the test. However, for the scenario where network slicing is not considered and for the first approach, this is not the case.

Finally, the jitter results obtained also show the superiority of the second approach with respect to the other scenarios considered. The jitter values obtained in this case are very stable throughout the test, without being affected by saturation and without exceeding 1.26 ms jitter at any point for the highest priority slice.

7. Conclusions

This paper evaluates the potential of network slicing applied to UL at RAN level in a dynamic environment with multiple data flows and different requirements from different external sources. For this purpose, two approaches were proposed to generate a dynamic, adaptive, and transparent slicing for the end users based on traffic classification. For both cases, a traffic classification mechanism was initially used to classify each type. In a second step, in the first approach this information was used to obtain the dominant traffic and to transmit all flows through the most suitable slice for this dominant traffic. In contrast, in the second approach this information was used to obtain the classification of each of the data flows and to redirect them independently through the slice associated with that category. However, irrespective of whether the first or the second approach was used, this classification and the use of slices were transparent to the external sources sending traffic to the UE, which acts as a gateway, regardless of the type of traffic. The algorithm generated was responsible for classification, slice selection, and transmission of traffic to the 5G network.

To evaluate the performance of such approaches, this paper deployed an experimental testbed of a 5G SA network.

The results obtained from these approaches, along with a reference scenario in which no network slicing was considered, showed that the use of network slicing techniques allows for higher efficiency and reliability for the most critical data, compared with those that do not require guarantees in terms of requirements, such as packet loss or jitter.

In addition, it can be seen that the second proposed approach showed an advantage over the first by obtaining greater control and accuracy in scenarios with heterogeneous traffic, by treating each type of traffic that passed through the UE independently, allowing for the highest priority data flows to be received correctly even in situations where heterogeneous traffic dominates. However, the first approach can be more useful when the networks do not have many resources, such as in networks with few available IP addresses or in scenarios where the type of traffic is not known but the traffic is known to be homogeneous.

Future lines of research should focus on the application of these approaches to end-to-end network slicing to ensure higher reliability throughout the 5G network. Likewise, better isolation of radio resources will also be sought, in order to avoid sharing resources between slices. Therefore, each category of traffic will have its own independent network. Finally, we will seek to improve the dynamic selection of slices by means of machine learning techniques.

Author Contributions: Conceptualization and methodology, Á.G., Z.F. and R.V.; software and investigation, Á.G. and Z.F.; writing—original draft preparation, Á.G. and Z.F.; writing—review and editing, Á.G., Z.F., R.V., Á.M., M.Z., P.A. and J.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Spanish Centre for the Development of Industrial Technology (CDTI) and the Ministry of Economy, Industry and Competitiveness under grant/project CER-20191015/Open, Virtualized Technology Demonstrators for Smart Networks (Open-VERSO).

Data Availability Statement: Experimental data are available upon request.

Acknowledgments: This research was supported by the Spanish Centre for the Development of Industrial Technology (CDTI) and the Ministry of Economy, Industry and Competitiveness under grant/project CER-20191015/Open, Virtualized Technology Demonstrators for Smart Networks (Open-VERSO) and the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957360 (5GMETA project).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

3GPP	3rd Generation Partnership Project
5GC	5G Core
5GPPP	5G Infrastructure Public Private Partnership
5QI	5G QoS identifier
DL	Downlink
DPI	Deep packet inspection
eMBB	Enhanced mobile broadband
gNB	gNodeB
HMTC	High-performance machine-type communications
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IoT	Internet of Things
IIoT	Industrial Internet of Things
KPI	Key performance indicators
MAC	Media access control
MEC	Multi-access edge computing
MIoT	Massive IoT
mMTC	Massive machine-type communications
MQTT	Message Queue Telemetry Transport
NGAP	NG Application Protocol

NGMN	Next Generation Mobile Network
NFV	Network function virtualization
OAI	OpenAirInterface
OSI	Open systems interconnection
PDU	Protocol data unit
PRB	Physical resource block
QoS	Quality of service
RAN	Radio access network
RTP	Real Time Protocol
SA	Standalone
SD	Slice differentiator
SDN	Software-defined network
SPI	Stochastic packet inspection
SSL	Secure sockets layer
SST	Slice/service type
S-NSSAI	Single Network Slice Selection Assistance Information
TCU	Telematic control unit
UDP	User Datagram Protocol
UE	User equipment
UL	Uplink
URLLC	Ultra-reliable low-latency communications
V2X	Vehicle to everything
VNF	Virtualized network functions

References

1. Shu, Z.; Taleb, T. A novel QoS framework for network slicing in 5G and beyond networks based on SDN and NFV. *IEEE Netw.* **2020**, *34*, 256–263. [CrossRef]
2. Sohaib, R.M.; Onireti, O.; Sambo, Y.; Imran, M.A. Network Slicing for Beyond 5G Systems: An Overview of the Smart Port Use Case. *Electronics* **2021**, *10*, 1090. [CrossRef]
3. Cisco, U. Cisco Annual Internet Report (2018–2023) White Paper. 2020. Available online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed on 15 January 2022).
4. 5GPPP: The 5G Infrastructure Public Private Partnership. Available online: https://5g-ppp.eu/ (accessed on 7 January 2022).
5. 5G-ACIA: 5G Alliance for Connected Industries and Automation. Available online: https://5g-acia.org/ (accessed on 7 January 2022).
6. 5G-AA: 5G Automotive Association. Available online: https://5gaa.org/ (accessed on 7 January 2022).
7. Siddiqi, M.A.; Yu, H.; Joung, J. 5G ultra-reliable low-latency communication implementation challenges and operational issues with IoT devices. *Electronics* **2019**, *8*, 981. [CrossRef]
8. Rinaldi, F.; Raschella, A.; Pizzi, S. 5G NR system design: A concise survey of key features and capabilities. *Wirel. Netw.* **2021**, *27*, 5173–5188. [CrossRef]
9. Teng, Y.; Yan, M.; Liu, D.; Han, Z.; Song, M. Distributed learning solution for uplink traffic control in energy harvesting massive machine-type communications. *IEEE Wirel. Commun. Lett.* **2019**, *9*, 485–489. [CrossRef]
10. 3GPP. System Architecture for the 5G System (5GS). TS 23.501 V17.2.0. September 2021. Available online: https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-h20.zip (accessed on 22 December 2021).
11. NGMN Alliance. 5G White Paper. In *Next Generation Mobile Networks, White Paper*; NGMN Alliance: Frankfurt am Main, Germany, 2015; Volume 1. Available online: https://ngmn.org/wp-content/uploads/NGMN_5G_White_Paper_V1_0.pdf (accessed on 20 January 2022).
12. Afolabi, I.; Taleb, T.; Samdanis, K.; Ksentini, A.; Flinck, H. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2429–2453. [CrossRef]
13. Barakabitze, A.A.; Ahmad, A.; Mijumbi, R.; Hines, A. 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Comput. Netw.* **2020**, *167*, 106984. [CrossRef]
14. Afolabi, I.; Taleb, T.; Frangoudis, P.A.; Bagaa, M.; Ksentini, A. Network slicing-based customization of 5G mobile services. *IEEE Netw.* **2019**, *33*, 134–141. [CrossRef]
15. Granelli, F. Chapter 3-Network slicing. In *Computing in Communication Networks*; Fitzek, F.H., Granelli, F., Seeling, P., Eds.; Academic Press: Cambridge, MA, USA, 2020; pp. 63–76. [CrossRef]
16. Elayoubi, S.E.; Jemaa, S.B.; Altman, Z.; Galindo-Serrano, A. 5G RAN slicing for verticals: Enablers and challenges. *IEEE Commun. Mag.* **2019**, *57*, 28–34. [CrossRef]
17. 5G-NORMA—5G Novel Radio Multiservice Adaptive Network Architecture. Available online: https://www.it.uc3m.es/wnl/5gnorma/ (accessed on 7 January 2022).
18. SLICENET: End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks. Available online: https://slicenet.eu/ (accessed on 7 January 2022).

19. 5Gtango: 5G Development and Validation Platform for Global Industry— Specific Network Services and Apps. Available online: https://www.5gtango.eu/ (accessed on 7 January 2022).
20. Crippa, M.R.; Arnold, P.; Friderikos, V.; Gajic, B.; Guerrero, C.; Holland, O.; Pavon, I.L.; Sciancalepore, V.; Von Hugo, D.; Wong, S.; et al. Resource Sharing for a 5G Multi-tenant and Multi-service Architecture. In Proceedings of the 23th European Wireless Conference 2017, Dresden, Germany, 17–19 May 2017; pp. 1–6.
21. Gavras, A.; Weiss, M.B.; Wang, Q.; Calero, J.M.A. SliceNet: E2E cognitive network slicing and slice management in 5G networks. In Proceedings of the European Conference on Networks and Communications 2018, Ljubljana, Slovenia, 18–21 June 2018.
22. Martínez, R.; Vilalta, R.; Casellas, R.; Muñoz, R.; Fei, L.; Tang, P.; López, V. Network slicing resource allocation and monitoring over multiple clouds and networks. In *Optical Fiber Communication Conference*; Optical Society of America: San Diego, CA, USA, 2018; p. Tu3D-11.
23. Banchs, A.; de Veciana, G.; Sciancalepore, V.; Costa-Perez, X. Resource Allocation for Network Slicing in Mobile Networks. *IEEE Access* **2020**, *8*, 214696–214706. [CrossRef]
24. Khan, L.U.; Yaqoob, I.; Tran, N.H.; Han, Z.; Hong, C.S. Network slicing: Recent advances, taxonomy, requirements, and open research challenges. *IEEE Access* **2020**, *8*, 36009–36028. [CrossRef]
25. Wijethilaka, S.; Liyanage, M. Survey on network slicing for Internet of Things realization in 5G networks. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 957–994. [CrossRef]
26. Akyildiz, I.F.; Khorov, E.; Kiryanov, A.; Kovkov, D.; Krasilov, A.; Liubogoshchev, M.; Shmelkin, D.; Tang, S. XStream: A new platform enabling communication between applications and the 5G network. In Proceedings of the 2018 IEEE Globecom Workshops (GC Wkshps), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.
27. Van Giang, N.; Kim, Y.H. Slicing the next mobile packet core network. In Proceedings of the 2014 11th International Symposium on Wireless Communications Systems (ISWCS), Barcelona, Spain, 26–29 August 2014; pp. 901–904.
28. Subedi, P.; Alsadoon, A.; Prasad, P.; Rehman, S.; Giweli, N.; Imran, M.; Arif, S. Network slicing: A next generation 5G perspective. *EURASIP J. Wirel. Commun. Netw.* **2021**, *2021*, 102. [CrossRef]
29. Sallent, O.; Perez-Romero, J.; Ferrus, R.; Agusti, R. On radio access network slicing from a radio resource management perspective. *IEEE Wirel. Commun.* **2017**, *24*, 166–174. [CrossRef]
30. Chang, C.Y.; Nikaein, N. RAN runtime slicing system for flexible and dynamic service execution environment. *IEEE Access* **2018**, *6*, 34018–34042. [CrossRef]
31. Ferrus, R.; Sallent, O.; Perez-Romero, J.; Agusti, R. On 5G radio access network slicing: Radio interface protocol features and configuration. *IEEE Commun. Mag.* **2018**, *56*, 184–192. [CrossRef]
32. Liu, Y.J.; Feng, G.; Sun, Y.; Qin, S.; Liang, Y.C. Device association for RAN slicing based on hybrid federated deep reinforcement learning. *IEEE Trans. Veh. Technol.* **2020**, *69*, 15731–15745. [CrossRef]
33. Mazied, E.A.; Liu, L.; Midkiff, S.F. Towards Intelligent RAN Slicing for B5G: Opportunities and Challenges. *arXiv* **2021**, arXiv:2103.00227.
34. Consortium, I.I. *Time Sensitive Networks for Flexible Manufacturing Testbed Characterization and Mapping of Converged Traffic Types*; Industrial Internet Consortium: Boston, MA, USA, 2019.
35. Zhao, J.; Jing, X.; Yan, Z.; Pedrycz, W. Network traffic classification for data fusion: A survey. *Inf. Fusion* **2021**, *72*, 22–47. [CrossRef]
36. Application Layer Packet Classifier for Linux. Available online: http://l7-filter.sourceforge.net/ (accessed on 30 November 2021).
37. OpenDPI. Available online: https://github.com/thomasbhatia/OpenDPI (accessed on 30 November 2021).
38. Libprotoident. Available online: https://github.com/wanduow/libprotoident (accessed on 30 November 2021).
39. nDPI. Available online: https://github.com/ntop/nDPI (accessed on 30 November 2021).
40. NFStream. Available online: https://github.com/nfstream/nfstream (accessed on 30 November 2021).
41. Bujlow, T.; Carela-Español, V.; Barlet-Ros, P. Independent comparison of popular DPI tools for traffic classification. *Comput. Netw.* **2015**, *76*, 75–89. [CrossRef]
42. Open5GCore. Available online: https://www.open5gcore.org (accessed on 2 December 2021).
43. Fokus. Available online: https://www.fokus.fraunhofer.de/en (accessed on 2 December 2021).
44. Open5GS. Available online: https://open5gs.org (accessed on 2 December 2021).
45. Amarisoft. Available online: https://www.amarisoft.com (accessed on 2 December 2021).
46. OpenAirInterface Software Alliance, OpenAirInterface (OAI). Available online: https://openairinterface.org (accessed on 2 December 2021).
47. free5GC. Available online: https://free5gc.org (accessed on 2 December 2021).
48. 3GPP. Non-Access-Stratum (NAS) Protocol for 5G System (5GS). TS 24.501 V17.4.1. September 2021. Available online: https://www.3gpp.org/ftp/Specs/archive/24_series/24.501/24501-h41.zip (accessed on 14 January 2022).
49. Alcock, S.; Nelson, R. Measuring the accuracy of open-source payload-based traffic classifiers using popular internet applications. In Proceedings of the 38th Annual IEEE Conference on Local Computer Networks-Workshops, Sydney, NSW, Australia, 21–24 October 2013; pp. 956–963.
50. Deri, L.; Martinelli, M.; Bujlow, T.; Cardigliano, A. ndpi: Open-source high-speed deep packet inspection. In Proceedings of the 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), Nicosia, Cyprus, 4–8 August 2014; pp. 617–622.

51. iPerf. Available online: https://iperf.fr/ (accessed on 25 January 2022).
52. Schulz, P.; Matthe, M.; Klessig, H.; Simsek, M.; Fettweis, G.; Ansari, J.; Ashraf, S.A.; Almeroth, B.; Voigt, J.; Riedel, I.; et al. Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture. *IEEE Commun. Mag.* **2017**, *55*, 70–78. [CrossRef]
53. Ma, Z.; Xiao, M.; Xiao, Y.; Pang, Z.; Poor, H.V.; Vucetic, B. High-reliability and low-latency wireless communication for internet of things: Challenges, fundamentals, and enabling technologies. *IEEE Internet Things J.* **2019**, *6*, 7946–7970. [CrossRef]