Cunha, V. A., Maroulis, N., Papagianni, C., Sacido, J., Jiménez, M., Ubaldi, F., Gharbaoui, M., Chang, C. Y., Koursioumpas, N., Tomakhxi, K., Corujo, D., Barraca, J.P., Barmpounakis, S., Kucherenkoxi, D., Giorgetti, A., Boddi, A., Valcarenghi, L., Kolodiazhnyixi, O., Zabala, A., Salvat, J. X. & García-Saavedra, A. (8-11 June 2021). *5 Growth: Secure and reliable network slicing for verticals* [proceedings]. 2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), Porto, Portugal.

DOI: 10.1109/EuCNC/6GSummit51104.2021.9482536

5Growth: Secure and Reliable Network Slicing for Verticals

Vitor A. Cunha[‡‡‡], Nikolaos Maroulis[§], Chrysa Papagianni[*††], Javier Sacido[‡‡], Manuel A. Jiménez[x], Fabio Ubaldi[†],
Molka Gharbaoui[¶xii], Chia-Yu Chang[*], Nikolaos Koursioumpas[§], Konstantin Tomakh[xi], Daniel Corujo[‡‡‡],
João P. Barraca[‡‡‡], Sokratis Barmpounakis[§], Denys Kucherenko[xi], Alessio Giorgetti[¶‖], Andrea Boddi[†],
Luca Valcarenghi[¶], Oleksii Kolodiazhnyi[xi], Aitor Zabala[‡‡], Josep Xavier Salvat[**], Andres Garcia-Saavedra[**]
[*]Nokia Bell Labs, [†]Ericsson Research, [‡]Instituto de Telecomunicações, [‡‡]Universidade de Aveiro, [xi]MIRANTIS
[§]National and Kapodistrian University of Athens, [¶]Scuola Superiore Sant'Anna, [**]NEC Laboratories Europe,
[††]University of Amsterdam [‖]IEIIT-CNR [‡‡]Telcaria Ideas S.L. [x]IMDEA Networks Institute [xii]CNIT

***Abstract*—Network slicing is well-recognized as a core 5G technology to enable heterogeneous vertical services sharing the same infrastructure. In this context, the H2020 5Growth project extends baseline 5G management and orchestration platforms to manage the life-cycle of real end-to-end, reliable, and secure network slices with performance guarantees. In this paper, we present 5Growth's approaches to (i) attain isolation across network slices, (ii) provide secure interfaces towards third parties, and (iii) exploit AI/ML to achieve reliability through automated anomaly detection. In our quest towards validating full-fledged 5G pilots, we demonstrate our slicing mechanisms in PoCs that include interacting with ICT-17 infrastructure.**

I. Introduction

One of the main goals of 5G is to make its infrastructure available to vertical industries that have traditionally been alien to the telco industry (e.g., automotive, health, construction) as a means to enable new services and boost revenue. In this way, *verticals* would deploy and deliver their services on top of shared 5G infrastructure. To this end, 5G shall support a comprehensive set of heterogeneous use cases, ranging from e-Health applications, with stringent low latency constraints, to streaming services, with substantial bandwidth requirements. Therefore, it is crucial to provide verticals with a secure and isolated environment that is tailored to their needs.

In this way, network slicing arises as a critical technology in 5G. Specifically, network slicing fits into the Infrastructure-as-a-Service (IaaS) paradigm, which enables partitioning 5G infrastructure into multiple virtual networks (*a.k.a.* "slices") that are tailored to the different services' requirements. Network slicing exposes tenants to a standard interface to deploy and configure their services, virtualizing the different network components using Network Functions Virtualization (NFV) and exploiting programmability enabled by Software Defined Networking (SDN). This enables fast deployment and simple lifecycle management of the network slices deployed.

However, network slicing is not *only* about virtualization [1]. To start with, performance guarantees and isolation across slices are required. For example, a resource-voracious slice should not exhaust other slices' resources, disrupting their performance. Moreover, network slices shall also interconnect different domains, potentially owned by different parties, which puts their security at risk. In summary, *network slicing should provide mechanisms that enforce resource guarantees, attain security, and detect/resolve anomalies.*

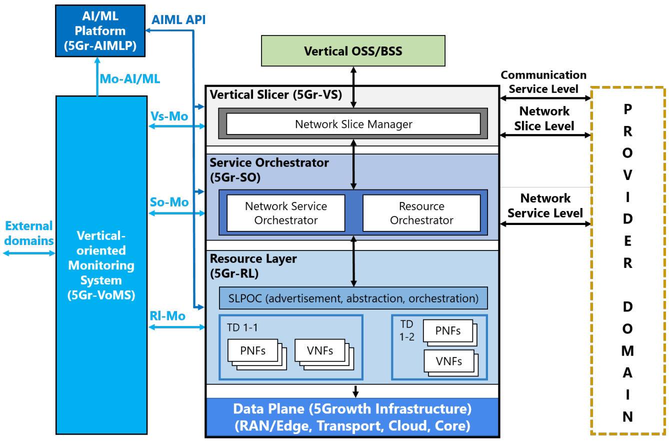
AI/ML Platform (5Gr-AIMLP)
AIML API
Mo-AI/ML
Vertical OSS/BSS
Communication Service Level
Vs-Mo
Vertical Slicer (5Gr-VS)
Network Slice Manager
Network Slice Level
So-Mo
Service Orchestrator (5Gr-SO)
Network Service Orchestrator
Resource Orchestrator
Network Service Level
External domains
Vertical-oriented Monitoring System (5Gr-VoMS)
Resource Layer (5Gr-RL)
SLPOC (advertisement, abstraction, orchestration)
Rl-Mo
TD 1-1
PNFs
VNFs
TD 1-2
PNFs
VNFs
Data Plane (5Growth Infrastructure) (RAN/Edge, Transport, Cloud, Core)
PROVIDER DOMAIN

Fig. 1: 5Growth High-Level Architecture

To this end, the EU project 5Growth extends the 5G-Transformer's platform providing such additional mechanisms for network slicing. In this paper, we present three different mechanisms that 5Growth has integrated into its platform to (i) provide bandwidth guarantees and limits for different slices while increasing the degree of isolation between them (ii); provide an additional layer of security when connecting other domains that belong to different stakeholders; and (iii) provide a method for anomaly detection for fast recovery.

II. Background

The 5Growth platform comprises three main building blocks (Fig. 1): (i) the Vertical Slicer (5Gr-VS), (ii) the Service Orchestrator (5Gr-SO), and (iii) the Resource Layer (5Gr-RL).

The 5Gr-RL layer interacts directly with the underlying infrastructure to provide all the different resources (e.g., computing, networking, storage) to network slices. Multiple *plugins* are used to this end, including: a WIM plugin to control the underlying transport networking, a VIM plugin to govern the available computing resources, a Radio plugin to manage the radio resources, and a MEC plugin to configure MEC services.

The 5Gr-RL abstracts the underlying infrastructure as a Single Logical Point of Contact (SLPOC), so any operation goes through the same standard IFA005-motivated interface.

The 5Gr-SO enables resource and service orchestration and lifecycle management for the deployed network services. In detail, the SO handles the placement, configuration, scaling, and termination of deployed network services interacting with the 5Gr-RL to configure a network service correctly. It also controls the interaction with other administrative domains working all together with neighbor Service Orchestrators. Finally, 5Gr-SO offers an end-to-end view of the deployed service to the 5Gr-VS through its northbound API.

Finally, the 5Gr-Vs is the main point of interaction with vertical parties. Once a vertical party requests the vertical service deployment, the 5Gr-VS maps it to a network slice instance, which provisions all the associated network services interacting with the 5Gr-SO.

Additional building blocks, namely the Vertical-oriented Monitoring System (5Gr-VoMS) and the AI/ML Platform (5Gr-AIMLP), are worth mentioning [2]. The 5Gr-VoMS integrates vertical service monitoring (gathering and storing service metrics) to support new mechanisms such as self-healing and service scaling. In turn, the AI/ML platform enables the management of AI/ML models as a service (e.g., training) to support novel mechanisms such as anomaly detection.

III. Traffic and Bandwidth isolation

The concept of isolation between network slices is a central point for 5G and network slicing. Several possible levels of isolation between network slices exist. Network slice isolation can be considered in areas such as [3] (i) isolation of traffic, meaning that network slices should ensure that the data flow of one slice does not move to another slice sharing the same infrastructure; and (ii) isolation of bandwidth, which implies that slices are allocated a certain bandwidth and should not utilize any assigned to other slices. Going one step further, performance isolation among slices means that service-specific performance requirements are always met on each slice, regardless of the traffic activities and workloads of other slice instances running concurrently. In this study, we focus on the solution adopted for traffic and bandwidth isolation in 5Growth. Moreover, we describe the corresponding extensions of the 5Growth stack to facilitate various levels of network slice isolation. Finally, we validate the efficiency of the proposed framework for traffic and bandwidth isolation in a PoC setup of the 5Growth architecture.

A. Traffic and Bandwidth isolation

This paper considers Layer-2 traffic isolation via the Virtual Private LAN Service (VPLS), a service providing pseudowires based on Ethernet multi-point to multi-point communications. VPLS enables sharing a common broadcast domain among the peers in a virtual private LAN, over the shared network infrastructure. Using VPLS in the transport network has been integrated by making use of the VPLS native ONOS application, which already provides an interface to deploy and manage the lifecycle of multiple VPLSs in SDN/Openflow-based layer 2 networks. However, the current implementation of the application (version 2.4) does not support P4 out of the box, so a re-engineering of the P4 data plane was required to be cross-compatible with OpenFlow switches. Specifically, the P4 pipeline was reduced to a single forwarding table and a single action, representing a chain of actions.

Bandwidth isolation per slice in 5Growth is guaranteed via applying existing OpenFlow [4] or P4 [5] meter-based solutions, depending on the programmability level of the shared data plane. In particular, the per-flow metering can measure and control the data rate of each flow. The adopted solutions ensure bandwidth isolation by rate-limiting the set of flows on a per slice basis, which enables network services planning for the transport network and ensures a rate for every slice. Despite the aforementioned, metering is desirable at the data plane, but the ONOS core can only set up meters over OpenFlow. In contrast, ONOS P4 drivers as of today (version 2.4) are not capable of creating and managing the stateful P4 objects of a P4-programmable data plane dynamically. Therefore, in 5GROWTH, we also introduce P4 metering capabilities at the ONOS level, providing bandwidth isolation seamlessly under a mix of P4 and OpenFlow switches.

B. 5Growth Integration

5Gr RL extensions. The current release of 5Gr-RL [6] is extended to support slice isolation. Specifically, the SLPOC abstract view is connected to an internal slicing isolation algorithm that maps the selected abstract resources to a QoS policy characterizing the network slice. A sample QoS policy for bandwidth management may include parameters like maximum/minimum bandwidth. The 5Gr-RL uses such a policy to allocate the transport resources via the WIM plugin. Therefore the southbound interface is extended accordingly to communicate the QoS policy parameters to the WIM plugin that is responsible for configuring the transport infrastructure.
5Gr WIM plugin. The WIM plugin allows for the interaction between the RL southbound interface and the ONOS SDN controller responsible for managing the network resources in the transport SDN-based network. In fact, the plugin allows to (i) check the current status of the network and (ii) enforce the necessary parameters to create isolated slices. More specifically, the WIM plugin is able to expose a set of parameters to the 5GR-RL that, upon request, can retrieve information regarding the status of the virtualized network resources characterizing the SDN network (e.g., virtual links, available bandwidth, total bandwidth). On the other hand, it also allows for requesting the virtualized network resources' allocation over one or more virtual links used for deploying a specific network service.

Regarding the previous release [7], the plugin has been extended to handle requests that aim to set up network slices with specific QoS characteristics and enforce the bandwidth isolation feature. To do so, new parameters characterizing the slices have been considered for the allocation operation, such as the maximum and minimum bandwidth and the maximum

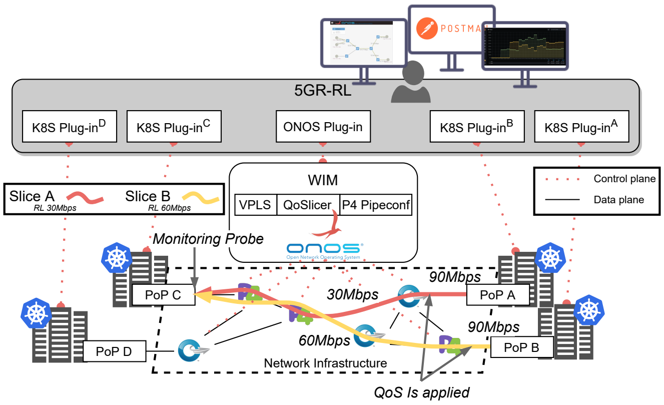

Fig. 2: Performance isolation validation scenario

burst size. Once those parameters are extracted from the request, the plugin interacts with the SDN controller to set up the slice and then enforce the QoS guarantees.

5Gr VIM plugin. The RL manages VIM resources - computing and communication resources inside specific VIM - through the VIM plugin. In the particular example, Kubernetes was used as a VIM, and the corresponding Kubernetes plugin for the 5Gr-RL was utilized. The VIM plugin receives the 5Gr-RL's requests to create compute resources, translates, and sends them to Kubernetes, creating pods inside the Kubernetes cluster. The initial version of the Kubernetes plugin did not support VLAN based networks. Kubernetes must use different container network interfaces (CNI) to support different network technologies. Multus [8] CNI is used to connect the pods to the transport network's external WIM. Multus is a CNI plugin for Kubernetes that enables attaching multiple network interfaces to pods. It provides the possibility to create a VLAN-based network that connects a physical port to the pod. The VIM plugin was extended accordingly to provide the interface for managing the Multus CNI.

C. PoC: Preliminary Results

To validate traffic and bandwidth isolation in the 5Growth, we deploy the network topology depicted in Fig. 2 composed of four Points-of-Presence (PoP), interconnected using network infrastructure including both P4 reference software bmv2 (behavioral model version 2) switches and OpenFlow Virtual Switches (OVSs). Each PoP manages its virtual resources on the control plane using Kubernetes (VIM), controlled by its own 5Gr VIM plugin. The network infrastructure is controlled by a single instance of ONOS SDN controller (WIM), using the 5Gr WIM ONOS plugin. Note that the mixed data path network infrastructure is emulated using Mininet utility. Finally, the 5Gr-RL interacts with the whole set of plugins gluing the whole workflow to deploy isolated network slices.

Monitoring. To gather real-time data plane information (e.g., throughput), we use network probes at the PoP endpoints based on the *pmacct* set of monitoring tools (www.pmacct.net). Thus, we monitor the active data flows and forward monitoring information through a *Kafka* broker (https://kafka.apache.org), where a flow monitoring data processor calculates the throughput per flow. Eventually, this information is scraped and stored by the *Prometheus* monitoring platform (https://prometheus.io) and visualized by the *Grafana* tool (https://grafana.com).

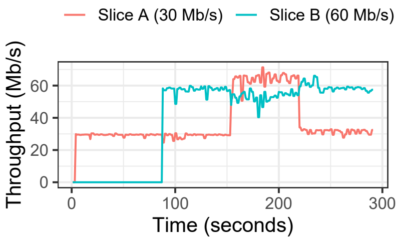

Fig. 3: Bandwidth isolation results

Validation scenario. Leveraging the aforementioned topology and real-time monitoring system in place, we deploy two network slices, as depicted in Fig. 2. The QoS policy employed for both slices enforces traffic policing. For each slice, we dynamically enable/disable the policer and observe the impact on their individual performance. In more detail, we generate UDP traffic at saturation rate (˜100 Mb/s) from (*i*) PoP A to PoP C (Slice A), and (*ii*) PoP B to PoC B (Slice B). Considering the first common link across both slices as bottleneck, a rate-limit of 30 Mb/s (slice A) and 60 Mb/s (slice B) is applied respectively to validate the approach's efficiency for both P4 and OpenFlow solutions. These values are selected so that the sum for both slices falls within the mean capacity of thetransport network (i.e., 100 Mb/s). In the following, we sequentially disable the QoS policy for Slice A and B, while we observe the impact of having policed and/or unpoliced slices running on the shared infrastructure. Finally, we re-enabled the policer for the slices sequentially (Slice A and then B) to validate fast convergence to the target rate.

Results. Fig. 3 depicts the results from the bandwidth isolation scenario by executing the above validation process. We observe that traffic policing has been successfully enforced on both data-paths, seamlessly activating the corresponding rate-limit solution over OVS or bmv2 switches (from 90 to 150 s). Slices coexist without any noticeable interference. However, when we disable the QoS policy in one of the slices, we notice how the un-policed slice (Slice A) consumes resources from the policed slice (Slice B); the throughput of the policed slice is diminished up to one-third of its target rate (from 150 to 180 s). Next, we also disable the policer for slice B and see traffic fluctuations when two un-policed slices coexist (form 180 to 220 s). In the result, we can see that both slices start competing for resources at around 200 s. Finally, we enable the QoS policy for slice A at 220 s, where slice B achieves its peak rate, and also for slice B at 250 s, where the slices resume their isolated operation.

IV. Interdomain Security

The 5Growth stack is designed to span across different administrative domains connected through a network. The ongoing 5Growth demonstrators at the pilot sites, much like other real-world deployments, have different levels of trust between the participating domains. Fig. 4 shows an overview of the control of the different systems in the 5Growth Aveiro pilot site. The site has three main stakeholders: the vertical customer, the ICT-17 infrastructure of 5G-VINNI, and the other domain (a private cloud). Each dashed rectangle represents an administrative domain. We highlight in Fig. 4 two types of trust boundaries. In red, we have the critical trust boundary across different domains through untrusted networks. This trust boundary is caused by the 5Growth stack's objectives and is fundamental for its integrity. In gray, we have the already existing trust boundaries of the typical management interactions within the administrative domains. In this article, we will focus on the boundaries created by the 5Growth stack.

A closer inspection reveals that the communications going through the untrusted network are all client-server interactions with a single 3GPP TS 28.531 compliant API. Therefore, the interdomain communications' attack surface can be reduced to a single server socket in each domain that must expose that API (i.e., the Network Slice as a Service (NSaaS) Providers). Furthermore, the interdomain communications over untrusted networks fit a threat model and resolved defenses that first depend on perimeter defenses. The most commonly used model to describe the attack process under a perimeter defense setting is the Cyber Kill Chain (CKC) [9].

The CKC consists of 7 sequential stages. During stage *(1) reconnaissance*, the attacker enumerates the endpoints, finds its targets, and discovers their characteristics. After acquiring sufficient information during the reconnaissance, the attacker starts the *(2) weaponization* stage. The service characteristics are used to identify a vulnerability and then devise a tailored payload to exploit that vulnerability. Once the payload is built, the attacker enters the *(3) delivery* stage, where that payload must be transferred into the target. After successfully delivering the payload, the attacker can enter the *(4) exploitation* stage where that payload is executed in the target system. That execution gives a foothold to the attacker that allows the *(5) installation* of a more reliable backdoor communication channel. After establishing the more reliable channel, the attacker can embed the victim machine into a *(6) Command & Control (C2)* such as a botnet system. Lastly, the attacker can now perform *(7) actions* upon his objectives.

Various approaches already exist to thwart each of the actionable stages of the CKC. For instance, deploying an Intrusion Detection System (IDS)/Intrusion Protection System (IPS) and suitable access control (e.g., firewall or others) is effective against reconnaissance, delivery, and subsequent exploitation stages. The weaponization stage is not actionable. The defender cannot stop the offline payload development done with the knowledge acquired during the reconnaissance stage. The later stages of the CKC depend on the success of the earlier ones. Detecting and defending against adversary activity in those stages goes beyond perimeter defense. A defense-in-depth approach would involve various types of endpoint protection and malware identification.

Our primary focus is the critical trust boundary of interdomain communications through untrusted networks in the 5Growth stack. We must first address the network bound challenges before proceeding to the defense-in-depth. The 5Growth stack already minimizes the attack surface to its minimum, a single API socket that must be exposed to authorized parties. Nevertheless, the attack surface can never be zero (i.e., we need to expose that socket to deliver functionality), neither are any of the existing defense systems completely failure-proof. Our main contribution has this fallibility in mind and thus researches complementary defense mechanisms that increase the resilience against an attack. Because the attack surface cannot be reduced any further, the next step is using Moving Target Defense (MTD) mechanisms to make that irreducible surface unpredictable to the attacker.

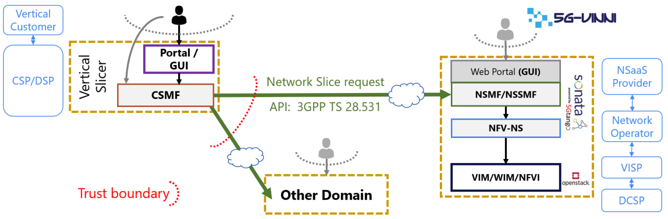

Fig. 4: Trust boundaries

The server's API socket is characterized by three network parameters within TCP/IP network headers: the transport protocol (e.g., TCP), the listening IP address of the host, and the listening port. The application protocol (e.g., REST) and the data that can inflict impactful system changes are in the packet payload. However, delivering that payload depends on the headers mentioned above. The main objective is to counter two main threat types: attacks carried-out by actors that do not have inside knowledge and the attacks performed by advanced threats that can acquire inside information through other intelligence sources. Actors without inside knowledge of the system rely heavily on the reconnaissance processes to acquire actionable intelligence. When the attackers already have some inside knowledge of the system and its vulnerabilities, we must disrupt the malicious payload delivery, thus stopping our stack's further exploitation through this vector. We devised an interdomain communications approach inspired by the framework that used MTD to set network slicing security as a KPI [10]. Our MTD mechanism relies on HMAC [11] to produce reliable mutations that can only be reversed by the authorized parties (i.e., those who hold the shared secret).

The evaluation was done through a PoC using the Aveiro pilot's facilities. We have measured the typical SONATA API request time for the LCM interactions and determined that (on average) the response took a little over ≈ 30 ms. Therefore, we have set the MTD mechanism to perform a mutation every 30 ms. The SONATA API performance values shown in Fig. 5 are within the 95% confidence interval. The bars show the calculated average, and the wishers show the distance to the min-max. The response time measurement was conducted in 500K runs for each of the two cases. We used siege to carry-out the load-test. The load-test values were gathered in 100 runs, each run having 10 workers that will perform 100 requests in parallel, making a total of 100K datapoints. We have also experimented with different parallelism levels, finding that the Sonata performance results were similar despite the changed parameters. We have determined that the MTD

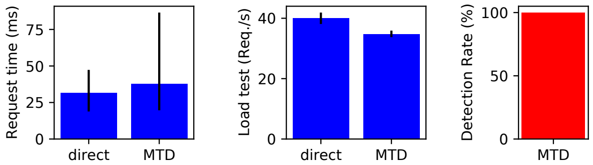

Fig. 5: Interdomain VS-Sonata LCM Requests Performance

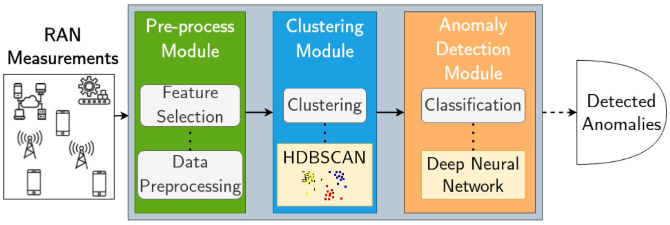

Fig. 6: Anomaly Detection Algorithm Process

mechanism has a measurable impact on the SONATA API's performance, making the average request time $\approx$ 19.53% slower, and reduced the number of requests attainable per second by $\approx$ 15.21%. Despite the initial impressions caused by those numbers, the impact over the average request time ($\approx +6.16$ ms) is not significant to the management processes done through that interface. Furthermore, while there is an impact to the maximum requests per second attainable when stress-testing the SONATA API (≈ -5.29 req./s), that is hardly a typical scenario in the 5Growth stack. We found that the MTD performance is well within the pilots' requirements.

In turn, the security benefits of the MTD mechanism were well noticeable. We have placed an attacker that attempts to exploit an information disclosure vulnerability in the SONATA API. All authentication mechanisms were disabled, leaving just the MTD solution as the only line of defense. We have carried-out 500K runs of an exploit attempt of the SONATA API using the best strategy available (a random port). In Fig. 5, we show that the detection rate was $\approx$ 99.9958%. In the few instances where the attacker evaded detection, we did not record any exploitation: the connection reseted midway because the mutation period was so fast (30 ms) and below the average request time ($\approx$ 37.67 ms with MTD).

V. Anomaly Detection

Due to the heterogeneity of the services that 5G and beyond network environments will support, the current work proposes an AI-based module in order to help administrators and slice tenants to detect and diagnose anomalies among the services of the different slices deployed on the virtualized infrastructure. The proposed AI framework is capable of analyzing aggregated and fine-grained data, such as resource-level data, RAN measurements, service KPIs, as well as traffic and mobility patterns [12], [**?**], in order to identify potential network anomalies.

A. Algorithm description

The proposed algorithm comprises three main phases (Fig. 6). The first phase involves pre-processing the input data, along with the feature extraction step. Table I illustrates the features of the training dataset.

Feature Name	Description
lost_packets	# of lost packets, per service per user
ul_delay	Uplink delay (ms)
dl_delay	Downlink delay (ms)
rsrp	RSRP (dB)
transfer_protocol	TCP or UDP encoded [0,1]
ulrx_cell	UE Bytes received from the Cell

TABLE I: Network KPIs used as Features for the Training Set

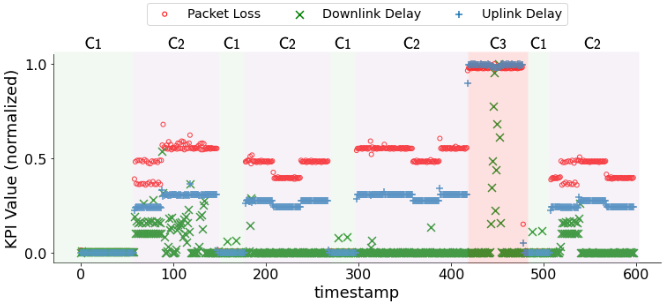

Fig. 7: Indicative network KPIs for clustering evaluation

Next, a clustering method is performed using an Unsupervised Machine Learning Clustering model, where outliers and abnormal patterns of the time series are grouped together. The algorithm used for this step is Hierarchical Density-Based Spatial Clustering (HDBSCAN), which is an extension to DBSCAN by converting it into a hierarchical clustering algorithm. HDBSCAN is a robust clustering approach for anomaly detection in time series, which can capture, as well as predict, clusters of varying densities. The number of clusters is determined beforehand and is based on how many severity levels a network administrator targets, thus increasing the granularity of the prediction. Therefore, by adjusting the $epsilon$ value of the distance metric of the Model, the number of clusters formed is also changed (e.g., a large distance value generates two clusters [-1,1], which means that samples with assigned cluster 1 are normal and -1 anomalies). Fig. 7 illustrates the training dataset's time series, along with its assigned cluster; three potential clusters are defined, namely [C_1: normal, C_2: moderate, C_3: anomalous] behavior. Some KPIs have been selected, namely $lost\_packets$, ul_delay, and dl_delay, to illustrate the validity of the clustering algorithm by comparing the variance of the selected KPIs with the assigned cluster, for each time-step.

Finally, a feed-Forward Deep Neural Network (DNN) is exploited, to perform a classification task, with the training dataset as input, along with each Cluster as the Label/Target value for each sample, as assigned by the previous phase. More specifically, the DNN model consists of 5 layers $X \times 64 \times 64 \times 32 \times Y$, where X is the number of features in each sample and Y the number of clusters determined by the clustering algorithm (one-hot encoded), ReLU activation function, and categorical cross-entropy as a loss function. The aforementioned classification scheme is then used to predict the Anomaly and assign a Cluster to new and unseen data in real-time and hence detect potential system anomalies.

Parameter	Service 1	Service 2
Transfer Protocol	TCP	UDP
Packet interval DL/UL	2/200 ms	1000/3 ms
Packet size DL/UL	500/50 bytes	12/1200 bytes

TABLE II: Service Modeling for the normal behavior UE

Parameter	Idle Mode	Overload Mode
Packet interval DL/UL	5000/5000 ms	1/1 ms
Packet size DL/UL	12/12 bytes	1400/1400 bytes

TABLE III: Traffic Load Modeling of the two modes

B. *Preliminary results*

We perform the evaluation using the NS-3 discrete event simulator. The simulation duration is set to 600 seconds. The logging frequency of the monitored data is set to 1 second. The area of experiments in which the simulation was executed is equal to 200x100 m^2, at the center of which a femtocell is placed. The UEs follow a *Random-Walk* mobility model, with *medium* or *high* velocities, uniformly distributed in the range of $(0.2, 1.2]$ m/s and $(1.2, 2]$ m/s respectively. Table IV summarizes the NS-3 parameters used in the scenario.

Overall, 31 UEs are simulated, one of which is considered the *test* UE and operates under *Normal* behavior running two types of services. The traffic models of the *test* UE are in Table II. The rest UEs are used in order to increase/decrease the network load to simulate anomalous network conditions; the 30 *loaded* UEs operate in 2 modes (*Idle* and *Overload*) in order to control the varying traffic load of the network (see Table III), under the following configuration:

- The UEs become active at t=60s
- Every 30 seconds, each UE chooses a random mode.
- For a specific 60 second time period [420,480]s all UEs operate in a fixed *Overload* mode.
- There are 3x30 second time periods [150,180], [270,300], [480,510] in which, all the UEs are in *Idle* mode.

The evaluation of the proposed scheme involves assigning a cluster to a new unseen and pre-processed (as in phase 1) test dataset, using the already trained Clustering algorithm to act as the ground-truth. The trained DNN model classifies each sample of the test set. The proposed scheme's accuracy is defined by calculating the percentage of correctly classified samples over the clusters assigned in the previous step. The classifier's accuracy varies based on the number of clusters determined at the beginning of the process. Figure 8 shows the accuracy score of the DNN Classifier over the different number of clusters defined, where the y axis shows the percentage of correctly classified samples, averaging out in approximately 87.51% accuracy score. It is worth noting that as the number of clusters increases – and hence, the algorithm's granularity – a trade-off can be observed in the accuracy of the classification prediction. More specifically, for the 2-cluster configuration, the prediction accuracy reaches 89,5%, while increasing to a 6-cluster configuration, the prediction scheme's performance decreases to 82%, i.e., an 8,3% loss. Thus, the configuration in terms of the number of clusters should be dependent on the specific use case and the granularity (i.e., number of levels) of the [*normal-anomalous*] behavior range that the network administrator targets to identify.

Parameter Description	Default Value
# of Normal Behavior UEs	1
# of Traffic Control UEs	30
# of FemtoCells	1
Mobility States	2 (Medium, High)
UEs' Transmission Power	20 dBm
Femto Cells' Transmission Power	20 dBm
Femto Cells Downlink and Uplink Bandwidth	20 MHz

TABLE IV: Parameters Used in the ns-3 Simulated Scenario

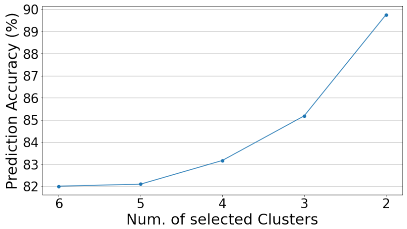

Fig. 8: Anomaly Prediction Accuracy Percentage

VI. Conclusion

This paper introduced a set of innovations that the 5Growth project proposes for better isolation and life cycle management of network slices. Specifically, we have presented how 5Growth (i) isolates traffic between network slices, which enables providing bandwidth guarantees per slice, (ii) securely interconnects different domains for end-to-end network slices, and (iii) exploits AI/ML to integrate anomaly detection in deployed slices. Our results using experimental PoCs at pilot sites, also interacting with ICT-17 infrastructure, validate our approach towards network slicing.

Acknowledgments

This work has been partially supported by EC H2020 5GPPP 5Growth project (Grant 856709).

References

[1] L. Zanzi *et al.*, "Ovnes: Demonstrating 5g network slicing overbooking on real deployments," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops*. IEEE, 2018, pp. 1–2.

[2] C. Papagianni *et al.*, "5Growth: AI-driven 5G for Automation in Vertical Industries," in *2020 European Conference on Networks and Communications (EuCNC)*. IEEE, 2020, pp. 17–22.

[3] Z. Kotulski *et al.*, "On end-to-end approach for slice isolation in 5G networks. Fundamental challenges," in *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sept 2017.

[4] D. Scano *et al.*, "Network Slicing in SDN Networks," in *22nd International Conference on Transparent Optical Networks (ICTON)*, 2020.

[5] Y.-W. Chen *et al.*, "P4-enabled bandwidth management," in *20th Asia-Pacific Network Operations and Management Symposium*, 2019.

[6] 5Growth RL code repository, https://github.com/5growth/5gr-rl.

[7] 5Growth WIM plugin code repository, https://github.com/5growth/5gr-rl/tree/master/rl/plugins/WIM/ONOS-OpenFlow-Slicing/onos-plugin.

[8] Multus CNI code repository, https://github.com/intel/multus-cni.

[9] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," *Proceedings of the 6th International Conference on Information Warfare and Security*, pp. 113–125, 2011.

[10] V. A. Cunha *et al.*, "Moving Target Defense to set Network Slicing Security as a KPI," *Internet Technology Letters*, May 2020.

[11] D. H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," RFC 2104, Feb. 1997.

[12] J. A. Ayala-Romero *et al.*, "vrAIn: A Deep Learning Approach Tailoring Computing and Radio Resources in Virtualized RANs," in *Proc. of ACM MobiCom 2019*.