Stateless Paradigm for Resiliency in Beyond 5G Networks

Savita Sthawarmath*‡, Eric Renault†, Thierry Lejkin‡
*Telecom SudParis, Evry, France
†LIGM, Univ. Gustave Eiffel, CNRS, ESIEE Paris, Noisy-le-Grand, France
‡Orange Innovation Networks, Chatillon, France

***Abstract*—An unprecedented surge in communication capabilities to things, in general, is challenging the traditional internet service providers. Telecommunication operators played a major role in connecting people to the internet and are now compelled to accommodate communicating things with traffic demands that are diverse and unpredictable in nature. To that end, sofwarization and virtualization of network entities have extensively helped to achieve a high degree of flexibility and scalability. Complementing this, separating the computing from the storage as a second degree of decoupling is required to make network functions highly resilient. Our work introduces the stateless network function paradigm by proposing a *Quasi-Local model* which is a fetch and cache model in order to achieve resiliency. We justify the proposed model with the state analysis, design, and derivation of state metrics. Furthermore, we assess various network architectures suitable for future stateless network functions to maintain the End-to-End delay budget of diverse telecom use cases.**

***Index Terms*—Stateless Network Functions, Network Function Virtualization, In-memory data-stores, Resilience, Future networks, 5G.**

I. Introduction

The promising objectives of next-generation networks as defined by the global telecom standardization body have exposed endless potential use-cases by embedding communicating capabilities in things. Internet of Things (IoT), Autonomous vehicles (V2X), Industry 4.0, and enhanced mobile broadband (eMBB) are some of the popularly heard buzzwords in today's communication society. A plethora of use-cases envisaged is in function of latency, bandwidth, and coverage requirements, in line with the promises of the future 5G network. Industry 4.0 may require continuous monitoring of robots and their critical parts, though the requirement of coverage is not significant for these sensors, any compromise in latency would affect revenue. So, is the case in telesurgery, where latency could equate to potential fatality. Some applications like autonomous vehicles have strict requirements of latency, coverage, and bandwidth to be able to sense its environment and operate without human involvement. These diverse applications of communicating things along with existing human adoption of mobile devices will likely change the service demand distribution from previously established Poisson behaviour. The fusion of traffic consumption distribution by various applications is going to define and test the robustness factor of the core network.

Fig. 1: Network entity evolution.

It is time to accept the fact that the traditional telecom core network was never designed for foreseen use-cases of 5G and thus incorporating the new technological advancement is the new mandate. To meet the requirements of 5G, telecom operators are adopting an openness attitude with the content producers (CPs) as well as equipment vendors. By facilitating Mobile Edge Computing (MEC) platforms as an Infrastructure as a Service (IaaS) at the edge of the core network, CPs are allowed to operate close to content consumers (CCs) thereby meeting the latency demands. Encouraging equipment vendors to softwarize the network and further virtualize the core network entities meets the scalability and resiliency demands. These changes make the current core network simpler and in line with the Internet architecture with Service Based Architecture (SBA) over HTTP for communication between network entities. Virtualization has successfully demonstrated the ways of exploitation of the underlying hardware for various applications at a given time, virtualizing network functions breaks this tight dependence of network entities with the hardware by being able to deploy on commodity hardware. We define these new generation of network functions as Virtual Network Functions (VNFs).

Though virtualization best addresses the scalability of future networks, network resilience can be further improved, for example by breaking the tight coupling between the computation and the data in a VNF. Introducing a novel data-plane by limiting VNFs to computation-only aims to address the network resilience effectively. This adds one more step in the network entity evolution, as shown in Fig. 1. Designing the data plane needs a) very good knowledge about control traffic volume between network functions for service provisioning and also the data traffic, b) defining the various data models coherent with the functioning of network functions, c) careful selection of data store providers in function to rate of change

of control data.

In this paper, we first highlight some interesting efforts by various research teams with similar objectives, as discussed in section II. We try to answer the basic questions required for the design of stateless network functions in section III. Later in section III-B, we do traffic analysis and estimation between various network entities considering 4G specifications. In section IV, we discuss our experimental scenario, highlighting a few critical parameters and estimating the latency due to network entity failure considering the best possible values. We conclude our discussion in section V

II. Related Work

The traffic speed promised by 5G demands a highly robust core and access network. To that end, the ability of the solutions and architectures to ensure traffic continuity at the lowest possible end-to-end latency defined as *Resilience* is of paramount importance for the network operators.

A. Resilience

Conventional methods used in mobile core networks generally include Active/Standby and Active/Active redundant pair systems. In both cases, an active node continuously synchronizes the memory to a standby node or dedicated partner active node. In the event of a failure of an active node, the standby node or partner active node takes over the traffic to ensure high availability of service. Both these solutions prove to be expensive as every active network entity needs a backup and the active node always need to be engaged in the synchronization process with its standby node consuming significant amount of computational resources [1].

In addition, auto-healing or restoration is considered in virtualized environments, in which redundant systems are rebuilt by removing failed elements and adding new Virtual Machines (VMs) as standby nodes [2]. However, restoration of Network Function (NF) failure impacts a potential number of users, leading to signalling overhead.

In a stateless environment, the resilience of network functions is achieved by decoupling the computational part and data part, as discussed in [3]. In case of network function failure, externalized state is pulled and used, thereby reducing the service disruption time notably.

B. Architecture for stateless Virtual Network Functions

Several architectural choices have been discussed while addressing the operation resilience aspects for VNF, for example, authors in [4] presents the ratio of virtual function versus compute instances as, *1:1*, *1:N*, *N:1*, *N:2* mapping with different characteristics, advantages, and disadvantages.

1:1 mapping, signifies one functional entity of 4G Evolved Packet Core (EPC) network over one VM. For example, one Mobility Management Entity (MME) over one running VM, and one Packet Gateway Entity (P-GW) on another running VM. Though the design is simple, it has some limitations in terms of scalability i.e, scaling up is easy but scaling down is a complex operation as the virtual EPC components are stateful,

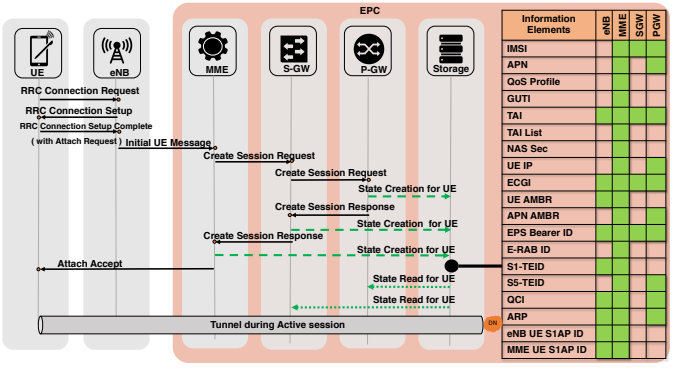

Fig. 2: Initial Attach with transaction level stateless model

and they cannot be simply shut down without disrupting the active user sessions and if a VM fails, all the sessions need to be re-established.

1:N mapping, several replicas of one EPC component are deployed on VMs, but may appear as a single logical entity to other peer components. Stateless EPC components in this implementation ensure the high availability of nodes, but are prone to synchronization issues and latency due to passing through multiple nodes.

N:1 mapping, all the EPC components are merged and deployed over one VM and N:2 is the extension of the N:1 model where control and data plane components are split and with stateless design introducing short delay processing.

1:N architectural choice is discussed in most of the related works as it enables high scalability requirements, availability of the nodes, and the resiliency to failure of nodes. However, the synchronization issues that possibly exists between different instances of VM is addressed in [5] with various state synchronization design choices being *always sync*, *session sync* and *no sync* designed for both control and data plane operations.

We plan to consider 1:1 and 1:N mapping architecture in our discussion and experimentation.

III. Prerequisites

The complexity of telecom networks is such that any upgrade or changes to existing architecture needs several rounds and types of validation. To initiate the *Stateless* concept in telecom network functions, we try to clarify the *3Ws*, i.e, *What*, *When* and *Where* to store?. It is also important to discuss *How* to store?. Answers to these questions are the foundation of the stateless concept.

A. Stateless Model

What to store Decoupling the data from computation at network functions appears to be feasible, but choosing an optimal scheme from various possible configurations is a real challenge. When it comes to data, the first question to be answered is; *What needs to be decoupled and stored?* or *How do we define data or state in a network function?*. Let us consider the Initial Attach Procedure in LTE shown in Fig. 2, the table on the right summarises the information pertaining to a user and bearer associated with the user session as generated

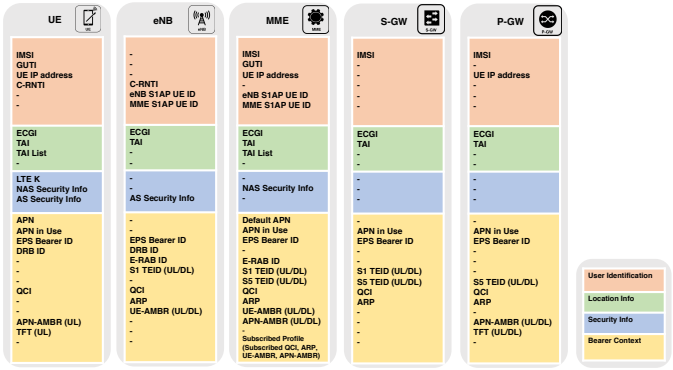

Fig. 3: Context at each network function after Initial Attach Procedure in LTE.

at the end of the attach procedure in 4G LTE network. These user contexts define the state or states being stored locally at each network function needs a state template for pushing the state to a datastore. These states either can be configured as a one big state or grouped as smaller substates as user identity, security, location and bearer context as shown in the Fig. 3. We consider LTE network procedures over 5G for two main reasons, First, to take advantage of availability of real-world network data that helps us to estimate the user-context metrics discussed in Sec. III-B. Secondly, the traffic we considered are mostly human-type communication, which remains as one of the future 5G diverse traffic types.

When to store The stateless concept can be introduced at different instants during the course of a signalling procedure, hence the challenge is identifying the optimal level to introduce statelessness. Possible options that we can think of are at,

- Message level.
- Transaction level (with state update).
- Procedure level.

Each level has its own pros and cons, like persisting the data after every message at a network function is considerable but introduces a lot of update traffic to the data store. Transaction and Procedural level look meaningful, as any failure of network function during a procedure setup does not mandate re-initialization of the whole process.

Where to store 3GPP has standardized the concept of control and user plane separation (CUPS) [9] because of various scalability and flexibility advantages. The core objective is to reduce latency of an application service, hence the network entity that handles user application traffic is made of two functional and independent parts i.e, the control-plane part and the data-plane part. The control part creates and updates the user context, which can later be used for packet processing in the data-plane. A network entity that does not handle user application traffic has their own copy of user context that will be useful in case of session creation error or other failures.

In order to make 5G architecture flexible, 3GPP Technical Specification [10] defines Unstructured Data Storage Function (UDSF) that allows any network entity to store and retrieve unstructured data. Standardization outlines certain core

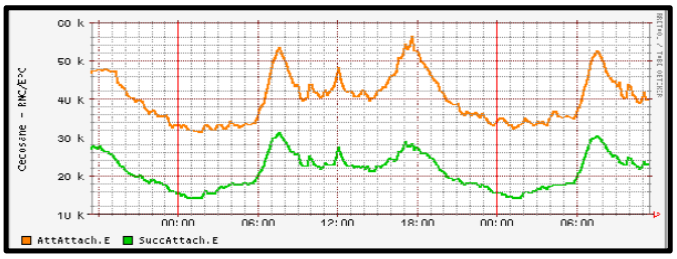

Fig. 4: Volume of Attach Request to a MME on a weekday in some site.

requirements which equipment vendors must adhere to like data concurrency and atomic consistency, subscribable service APIs, ensuring very low latency, multiple logical spaces, guidelines on UDSF sharing by multiple network entities, etc.

Fig. 2 illustrates the potential call flow of the Initial Attach procedure in stateless mode. A state is created by each network function after a transaction, i.e, after every request-response pair, and the final state is written to the data store at the end of the procedure. Once the initial attach is completed and state is pushed to the externalised datastore, the serving data-plane network function processes user packets by reading the user state from the datastore. This process of fetching the state to process each packet can overwhelm the data-plane thereby questioning the network latency and high bandwidth.

Given all the discussion above, we propose our stateless model to have,

- 1:1 or 1:N functional architecture.
- 1:1 user context storage template model.
- Procedure level of context persistence.

Let us call this stateless model as *Quasi-Local* since user context is made available locally (cached) only during the user session. In such a setup, when a network function fails, a new instance can be spinned up to process the redirected packet flow by fetching the user state from the datastore thereby meeting the resilience objective.

B. Traffic Analysis

To validate an optimal stateless scheme for virtual network functions, we need to have a very good understanding of the magnitude of traffic managed by the current core network. For the sake of discussion, we consider the Attach Request procedure in the 4G LTE network, which is the first communication of the User Equipment (UE) with the 4G LTE Evolved Packet System (EPS) to request any service. We try to evaluate the traffic per network entity w.r.t *state metrics* like *Volume*, *Size* and *Frequency* of requests.

Volume of requests can represent 1) the number of data session requests made by the UE to EPS 2). In a stateless network function context, it could mean - the number of datastore operations.

We assume that a UE is attaching itself to the telecom network to request a data session. Let N be the total number of attach requests, n be the number of successful attach attempts in a unit time, and n_w be the number of writes to the datastore. Fig. 4 shows the pattern of attach requests attempted at the

MME of EPS in one location for a weekday. This data is received from a telecom provider just to have an idea about the magnitude of requests. As we see, there are two to three distinguishable peak time of the day where the request is around $50K$ (with a 5K spread) with an average of 40.236 K requests. It is also evident that around 39% of requests are rejected for various reasons, hence average successful attach requests $n = 21.244K$. Considering Procedure wise stateless concept, a success rate of 61% means, on an average $n_w = n$ writes are performed from MME to datastore per second.

Size of Information represents 1) the size in bytes of signalling information per entity when requesting for a procedure 2) In stateless network function context, the size in bytes of the state written to or read from the datastore.
Traffic at MME following Attach Requests can be formulated as,

$$T_{MME} = S_{MME} \times n \times 60$$

where, T_{MME} represents the traffic at MME, S_{MME} represents the Information Elements (IE) size in the attach request. S_{MME} received by MME range around 239 bytes at the lower bound, the actual/exact number might vary depending on the length of the Access Point Name (APN), number of IP addresses used to construct the Traffic Flow Template (TFT), size of the security key, etc. Hence, plugging in the values gives $T_{MME} = 304.6MB/min$ or $18.27GB/hour$.
Similarly, Traffic at S-GW following attach requests can be formulated as,

$$T_{S-GW} = S_{S-GW} \times n \times 60$$

where, T_{S-GW} represents the traffic at S-GW, S_{S-GW} represents the IE size in the Attach Request. To calculate traffic T_{S-GW} at S-GW and T_{S-GW} at P-GW, it is safe to consider the Average Attach Success n. S_{S-GW} received by S-GW is 107 bytes at lower bound. Hence, we estimate traffic T_{S-GW} to be roughly $136MB/min$ or $8.18GB/hour$.

Likewise, Traffic at P-GW following Attach Requests can be formulated as,

$$T_{P-GW} = S_{P-GW} \times n \times 60$$

where, T_{P-GW} represents the traffic at P-GW, S_{P-GW} represents the IE size in the Attach Request. S_{P-GW} received by P-GW is 150 bytes during at lower bound, giving us T_{P-GW} estimate of $191.19MB/min$ or $11.47GB/hour$.

Frequency of information is influenced by both Volume of requests with time and size of information between entities. During the Initial Attach Request for bearer setup and considering procedure wise stateless mode, the number of state equals the number of Attach success i.e. n. Now for three EPS entities i.e, MME, S-GW and P-GW, assuming that they maintain their own copy of user context,

$$n_{w,Total} = \sum(n_{(w,MME)}, n_{(w,S-GW)}, n_{(w,P-GW)}) \times 60$$

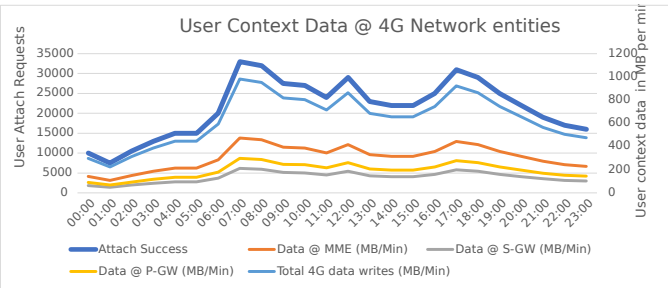

Fig. 5: Volume of information at network entities over 24hours following Initial Attach request.

where, $n_{(w,MME)}, n_{(w,S-GW)}, n_{(w,P-GW)}$ represents the state writes triggered by MME, S-GW and P-GW respectively. This estimates a total writes $n_{w,Total} = 3.823Mwrites/min$.

Now, the traffic to the data store T_{Total} during Attach procedure can be formulated as,

$$T_{Total} = \sum(T_{MME}, T_{S-GW}, T_{P-GW})$$

which gives us an estimate of $631.79MB/min$ or $37.92GB/hour$ at the lower bound considering the values calculated earlier. Although these analyses is limited to a geographic area, it is sufficient for a data store qualification study.

Forecasting the 5G traffic behaviour helps us to adapt to hugely varying patterns of 5G traffic which not only constitutes cellular traffic but also from varied 5G scenarios like IoT, V2X, Industry Automation. Considering these varying sources, the traffic pattern could possibly follow the poisson process of random arrivals as it is composed of some constant behaviour coming out of IoT devices along with the normal periodic/cyclic behaviour of cellular traffic.

Ericsson mobility report 2019 [11] estimates that by 2024, 5G networks will carry 25% of the world's mobile data traffic i.e., mobile data traffic is expected to increase by five folds (x5). Applying this estimation to our 4G traffic analysis in Fig. 4, we can roughly assume that the volume of the 4G traffic will be multiplied by 5 times, which drastically increments the number of bytes to be fetched/written to the data store, i.e,

$$T_{AMF} = T_{MME} \times 5$$

where T_{AMF} is the traffic at 5G Core Access and Mobility Management Function (AMF). Using the earlier numbers gives us T_{AMF} could swell to 1.523 GB/Min or 91.3 GB/hour. In Fig. 5, we present the detailed traffic pattern at network entities versus the number of successful attach requests as presented in Fig. 4.

This could mean that in addition to data store qualification, we may also need to look at data store hierarchical designs to meet future demands.

IV. Results and Discussion

In this section, we highlight the significance of our proposed model with real network data collected from an operator core network around a certain geographic area. Fig. 6 shows the

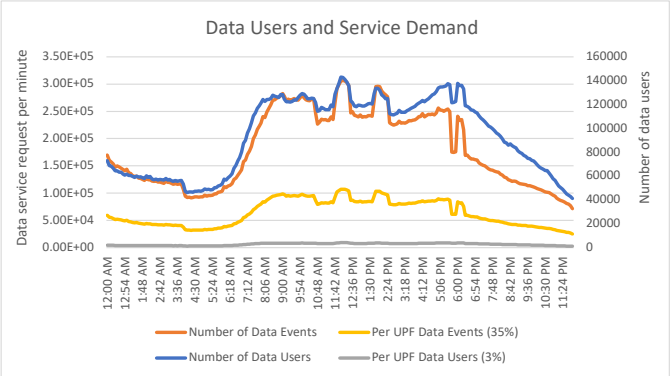

Fig. 6: Data Users and Service Demand

number of data users versus number of data requests made by those users over a period of $24h$. Due to confidentiality reasons, we are unable to publish the exact number of core network entities handling these requests. Due to this reason we assume some values from an interesting study by SK Telecom and Intel Corporation on designing next generation data-plane functions for 5G [12]. In this study, authors prove that a 5G UPF implementation with FD.io Vector Packet Processor (VPP) framework [13] with Data Plane Development Kit (DPDK) [14] plugin shows improved performance in latency and jitter for high priority traffic while still running best effort for lower priority traffic at high throughput rates and high infrastructure utilization. In this experiment, data packets from 50000 users were processed with various combination of packets belonging to high priority and normal traffic category. We consider the best possible values from a profile - where 50% traffic is from high priority group and the rest belonging to normal group. They state that per packet processing time when UPF CPU load is at 50% with average downlink packets (DL) per second per user of 295 packets is 40μs. Assuming that we employ such UPFs in our network, we uniformly distribute the number of data users and data events. In such scenario, each UPF handles around 3% of data users and handles 35% of data requests as shown in the Fig. 6.

Considering the above values, in Fig. 7, we show the number of DL packets at a given time versus the amount of time in seconds required to process those packets with various packet processing schemes. An important assumption here is that buffer-length in a given time does not affect the packet processing time in the next time step, in reality, this will greatly have an impact on the throughput. This assumption is made because of the values considered from [12], in real world telecom network, high performance UPF servers handles data requests and traffic from more than 50000 users. In addition to the values of packet processing, we experimented with various in-memory databases like Redis [15], VoltDB [16] and found that the data fetching times range between 20μs to 200μs. For this comparative study, we consider 20μs as access times by UPF.

Our objective in this study is to compare the packet processing time with three packet processing schemes,

- *True Stateless* - where user context is fetched to process each packet.
- *Quasi-Local Stateless* - where user context is fetched when UPF encounters the user packet for the very first time.
- *Stateful* - where user context is pushed from SMF to UPF as a part of PDU session establishment procedure.

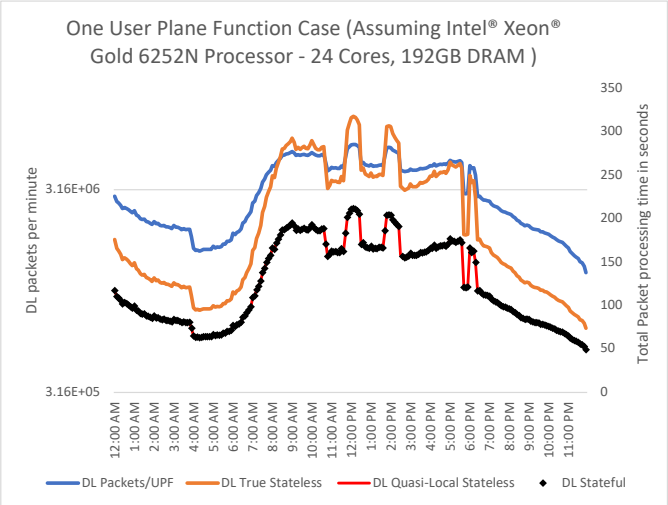

Fig. 7: VPP based User Plane Function case

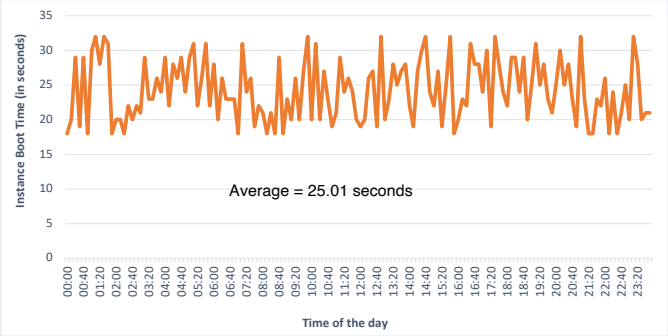

Fig. 8: Instance boot time benchmarking

From the Fig.7, it is clear that *True Stateless* scheme is the most expensive model towards stateless network functions and hence is not recommended for use-cases with critically low delay budget despite offering high degree of service continuity. *Stateful* scheme on the other hand provides the best latency but are not immune to system failures. The best compromise between these two schemes is presented by our *Quasi-Local Stateless* scheme, where UPF fetches the user context only when there is a user packet to process and a cache miss is encountered. Once fetched, user context is cached up to a certain time threshold. Upon failure of a Stateless UPF, a new instance can be booted to resume the operation, hence equating the service restoration time to boot time of a system. To have a realistic idea about the instance boot time, we experimented with Google Cloud Platform [17] considering Google MemoryStore as in-memory database. We achieved the average instance boot time to be around $25s$ as shown in Fig. 8. These values were derived when we instantiated an

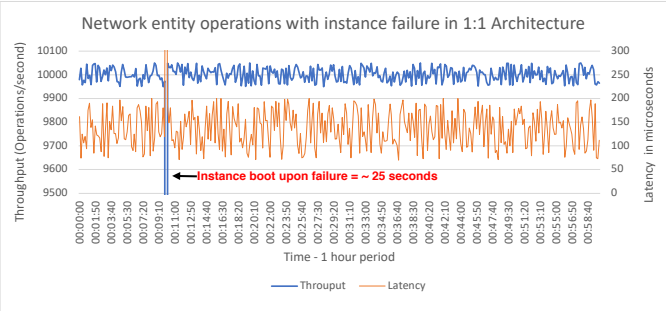

Fig. 9: Failure Scenario with Stateless UPF with 1:1 Architecture

instance at different times of the day with $10min$ frequency over a period of $24h$.

Furthermore, we constructed a functional stateless system to work with Quasi-Local Stateless scheme with 1:1 architecture. The objective of this study is to see the impact of instance downtime on service delivery, so we considered launching a basic non-telecom service that fetches the required data from an in-memory database to process each requests. This triggered around $9K$ database operations for an identical number of service requests. Over the period of $1h$, we simulated a failure event and, we confirmed that service downtime equates to system boot time with infinitesimal small delta fetch time of $20\mu s$ per user as discussed above and shown in Fig. 9.

V. Conclusion and Future Works

In this work, we discussed the significance of stateless network functions to address the resilience aspects in telecom networks. Our proposed *Quasi-Local Stateless Model* can easily be scaled and provides faster failure recovery compared to currently existing expensive solutions like Active-Standby and Active-Active which measures in several seconds. Our experimentation with Google Cloud has presented some promising results that motivate us to port the functional stateless network implementation to telecom network functions.

References

[1] Visibility Architectures: The ABCs of Network Visibility. 2020.

[2] Cisco S-GW Restoration, Ultra Services Platform Release N5.5.

[3] Murad Kablan and Azzam Alsudais and Eric Keller and Franck Le, "Stateless Network Functions: Breaking the Tight Coupling of State and Processing," 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2017).

[4] T. Taleb et al., "EASE: EPC as a service to ease mobile core network deployment over cloud," in IEEE Network, vol. 29, no. 2, pp. 78-88, March-April 2015, doi: 10.1109/MNET.2015.7064907.

[5] P. Satapathy, J. Dave, P. Naik and M. Vutukuru, "Performance comparison of state synchronization techniques in a distributed LTE EPC," 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Berlin, Germany, 2017, pp. 1-7, doi: 10.1109/NFV-SDN.2017.8169832.

[6] Tamura, M., Tetsuya Nakamura, T. Yamazaki and Y. Moritani. "A Study to Achieve High Reliability and Availability on Core Networks with Network Virtualization." (2013).

[7] F. Ojala, A. Rao, H. Flinck and S. Tarkoma, "NoSQL stores for coreless mobile networks," 2017 IEEE Conference on Standards for Communications and Networking (CSCN), Helsinki, Finland, 2017, pp. 200-206, doi: 10.1109/CSCN.2017.8088622.

[8] L. Abdollahi Vayghan, M. A. Saied, M. Toeroe and F. Khendek, "Deploying Microservice Based Applications with Kubernetes: Experiments and Lessons Learned," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2018, pp. 970-973, doi: 10.1109/CLOUD.2018.00148.

[9] 3GPP TS 23.214 Architecture enhancements for control and user plane separation of EPC nodes.

[10] 3GPP Technical Report 29.598 V16.0.0 Unstructured data storage services.

[11] Ericsson, "Ericsson Mobility Report", 2019.

[12] SK Telecom and Intel Corporation; White Paper "Low Latency 5G UPF Using Priority Based Packet Classification" 2020.

[13] FD.io. 2016. Fd.io /dev/boot. 2016.

[14] Intel Corporation. Intel DPDK vSwitch: Performance Report, 2014.

[15] Redis, "Redis benchmark", 2021.

[16] VoltDB, "VoltDB Benchmarks Report", 2018.

[17] Google Cloud Platform, https://cloud.google.com