LOCOMOTIVE 5G Core for 6G ready Resilient and Highly Available Network Slices and SFCs

Sourav Sarkar, Shwetha Vittal, Antony Franklin A
Department of Computer Science and Engineering, Indian Institute of Technology Hyderabad
Email: {cs19mtech11031, cs19resch01001, antony.franklin}@iith.ac.in

***Abstract*—The presence of a Load Balancer (LB)s is much significant to keep up the High Availability (HA) and resilience of the scalable 5G Core (5GC). The whole system may collapse just because of inefficient LB at any NF, resulting in total disruption to the High Availability (HA) service. In this paper, we present the LOCOMOTIVE 5GC which outperforms the traditional hot standby in both HA and resilience during various dynamic conditions. LOCOMOTIVE serves** 16% **(at least) more user requests compared to hot standby in the control plane while handling unexpected overloaded conditions (without the failure of LB). During the failures of LB, it drops** 22% **lesser user requests than hot standby. With this outstanding resilience, LOCOMOTIVE even achieves** 4% **better availability than the hot standby in an active-active cluster configuration. To prove the feasibility of LOCOMOTIVE and to encourage further research works in the world of LBs, we developed its entire framework in a 3GPP compliant 5G test-bed system along with eXpress Data Path (XDP) and extended Berkeley Packet Filter (eBPF) framework.**

I. Introduction

Service Based Architecture (SBA) based 5G Core (5GC) (see Fig. 1) comprises a set of Network Function (NF)s namely, Access and Mobility Management Function (AMF), Authentication Server Function (AuSF), Unified Data Management (UDM), Network Repository Function (NRF), and Session Management Function (SMF) in the control plane. These NFs can be grouped together to compose a network slice (see Fig. 1) or a Service Function Chain (SFC) [1] (see Fig. 2) to serve different User Service Requests (USR)s like User Equipment (UE) registration, Packet Data Unit (PDU) session establishment, and modification. Additionally, Network Function Virtualization (NFV) [2] enables these NFs of 5GC to run as compute-intensive Virtual Network Function (VNF)s on commodity hardware (like cloud servers) to achieve higher performance in each of the slice service types [3]. However, the overwhelming control plane traffic in the 5G network urges the operator to design and deploy a scalable 5GC. As shown in Fig. 1, the scalable 5GC has a Load Balancer (LB) leading from the front and multiple instances of the NF in the back end, to serve the User Service Requests (USRs) arriving through the Radio Access Network (RAN). Any unanticipated fluctuations in the traffic and unforeseen failure of the LB at each of the NFs could disturb the High Availability (HA) of the service for the respective users. Though the requirements for 6G are not released, researchers [4] envision enhancing resilience and security in 6G.

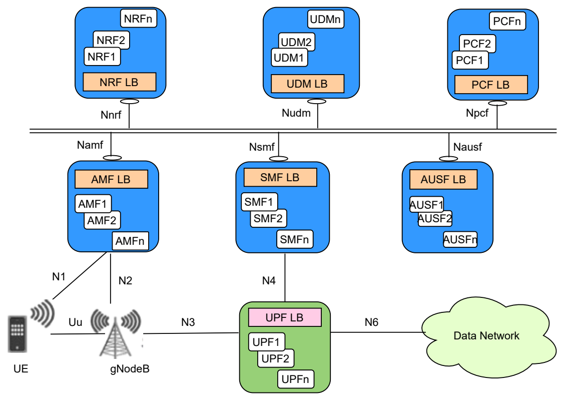


Fig. 1: Scalable 5GC SBA from [5] for a network slice.

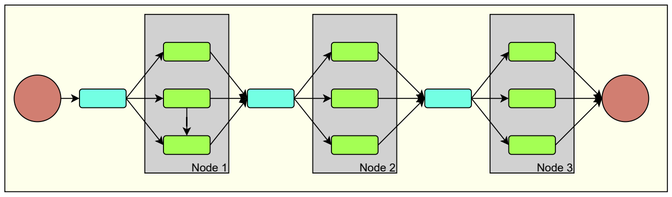


Fig. 2: SFC architecture with VNFs and LBs from [1].

In this regard, we present a LOCOMOTIVE 5GC, having higher resilience to unexpected situations on the control plane functions, by revisiting the narrow frontier between the LB and serving instances of every NF in the 5GC. The key idea of LOCOMOTIVE is to leverage the resources of hot standby LB to serve more USRs, instead of keeping them in idle. Additionally, we claim that an NF instance can take up the role of an LB whenever needed. i.e., if the primary LB fails unexpectedly, to keep up the HA of the slice, the LOCOMOTIVE approach enables one of the serving instances of the NF to serve as the primary LB, before a new LB instance gets ready. To prove these key points, we build the complete proof of concept LOCOMOTIVE framework prototype on our in-house 3GPP compliant 5GC testbed. In addition to this, to encourage further research works in the world of LBs, we leverage eXpress Data Path (XDP) [6] and extended Berkeley Packet Filter (eBPF) [7] framework. For this, we place LB functionality in the kernel driver, availing full benefits of XDP's secure kernel mode execution of LB and avoiding busy poll loops on the receive path. Therefore,

978-3-903176-51-5 © 2022 IFIP

TABLE I: Conventions in rest of the paper

Term	Meaning and Interpretation
5GC	Only the Control Plane of 5G Core in this work.
Slice or SFC	Set of resources in the form NFs to offer a specific service.
USR	User Service Requests arriving at 5GC through RAN.
NF	Control plane Network Functions like AMF, SMF, AuSF, UDM, NRF, and PCF.
NFI	Serving instance of an NF.
primary LB	Primary Load Balancer is the current LB, which actively handles the arriving USRs at the entry of NF.

to summarize, the key contributions of this work are:

- Design of resilient LOCOMOTIVE 5GC to serve additional USRs than the popular hot standby approach.
- Addressing the unexpected failure(s) of one or multiple (even the chained too) primary LBs, by leveraging the capacity of the serving NF instance.
- Proof of concept of the LOCOMOTIVE 5GC with XDP and eBPF to demonstrate its feasibility.

II. Motivation and Related Work

A. High Availability and Resilience of 5G Core

The High Availability of 5GC directly relates to its capability to be available in order to deliver the service in conformance with Service Level Agreement requirements. However, the resilience of 5GC directly relates to its capability to work and recover up to the best potential during and after unforeseen failure(s) of LB and NF instances. Therefore, the scope of availability and resilience of 5GC relates to the presence of the individual NFs [8] of 5GC and the associated LB of the respective NF. Hence, the availability of an NF includes the availability of its multiple instances and the associated LB. For example, A_{AMF} encompasses the total availability from one or more serving AMF instances and the respective LB. Therefore, the overall availability of an NF is given by

$$A_{NF} = A_{NF-LB} * (1 - (1 - A_{NFI})^n)) \qquad (1)$$

where NFI represents the NF Instance (serving instance of NF) and 'n' is the total number of such NFIs. A_{NF-LB} represents the availability of the respective NF's LB. Finally, the availability of each of these participating Network Entities (NE = NF instance or the LB) is given by

$$A_{NE} = \frac{Uptime\ of\ NE}{Uptime\ of\ NE + Downtime\ of\ NE} \qquad (2)$$

B. Traditional High Availability Provisioning

3GPP [3] motivates the need for stateless network functions in 5GC by designating a central network entity equipped with a database named Unstructured Data Storage Function (UDSF) [3]. So, all other NFs can function in a stateless mode, by storing the information they require and up-to-date UE context at UDSF. This functioning has been discussed in detail in our previous work [5]. So, continuing in the same context of 5GC, we now discuss the most relevant traditional cold standby and hot standby HA methods [9]. As detailed in [5], we assume the following is consistent across all the methods.

- Each of the 5GC NFs has multiple instances (NFIs) to serve the USRs. Every such NFI is of equal capacity in terms of storage and computing for processing the USRs.
- A dedicated UDSF is reserved for the LB. It stores the scheduling information for LB to aid the stateless functionality of LB.
- The LB at every NF is capable of handling the overloaded situation. But, it can fail independently of the LB at other NF (see Fig. 1).

*1) **Cold Standby**:* As shown in Fig. 3a, in the cold standby method, a new instance of LB is instantiated after the failure of the primary LB is detected. Once this new instance gets active, it fetches the up-to-date load balancing information from the UDSF and then it is ready to schedule the USRs as a primary LB. Therefore, the HA of NF in the cold standby can be realized from Eqn. 1.

*2) **Hot Standby**:* In the hot standby method, as shown in Fig. 3b, another instance of LB keeps running in the background. UDSF is aware of this standby LB. This standby keeps its load balancing information up-to-date by fetching it from the UDSF regularly. Whenever the failure of the primary LB is detected, UDSF immediately triggers this standby LB to get into the active mode. Therefore, HA in hot standby for 'n' number of NFIs (serving instances of NF) can be realized as

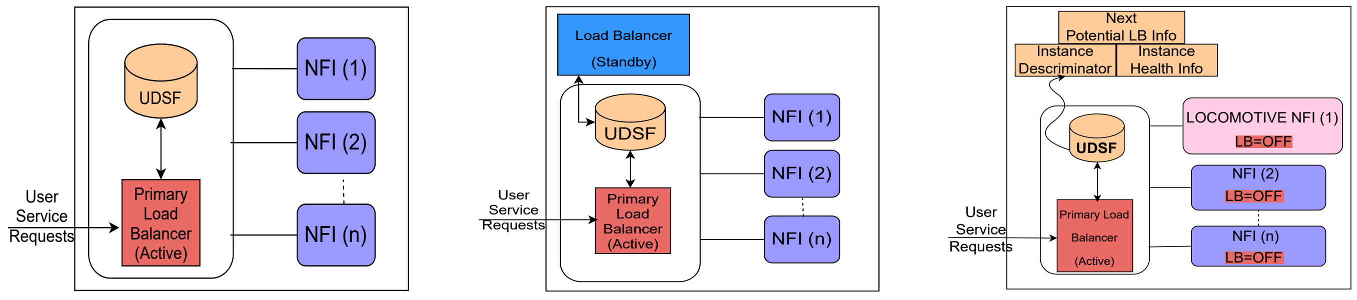
$$A_{NF} = (1 - (1 - A_{NF-LB})^2) * (1 - (1 - A_{NFI})^n)) \qquad (3)$$

C. Rationale of Load Balancer for High Availability Service

The presence of LB for every NF is a must in the scaled framework of the 5GC (see Fig. 1), to achieve higher performance and the total availability (see Eqn. 1) five 9s '(99.999 %)' representing carrier-grade HA service [8] in the NFV domain. To ascertain these facts in HA service with resilience, we capture the total time of commissioning the LB of individual NFs from our previous work [10]. In the experiment, this commissioning time is about '80' seconds for LB on Virtual Machine (VM)s using OpenStack. Any unforeseen failure of LB makes the slice service unavailable by dropping the USRs arriving during such situations. Hence, it is important to keep up the HA provisioning and resilience of the overall slice. Though the hot standby can solve the situation quickly (see Fig. 3b), unanticipated subsequent failures of LB in a closed interval lead to a catastrophic collapse of the slice. Moreover, in the hot standby, the resources of standby LB are totally untapped (unlike the cold standby, where the resources of standby LB are conserved) until the primary LB fails.

D. Related Work

In [11], the authors emphasize the resilient mechanisms to be implanted into VNFs, to gracefully handle unexpected failures in order to maintain the desired HA service level in the NFV domain. [12] sets a goal for delivery of HA in NFV



(a) (b) (c)

Fig. 3: Traditional High Availability provisioning in 5GC a) Cold Standby, b) Hot Standby, and c) Proposed LOCOMOTIVE.

during resource failures and bursty traffic conditions. Works in [13] and [14] predict and classify the failure events using Artificial Intelligence (AI) methods and then design proactive actions against such failures. AI is definitely a crucial enabler for building 6G [4] networks. However, the sudden failure of LB or a network entity is still an open issue. Hence, HA with resilience is the critical challenge for 6G networks [4], due to which proactive hot standby approach [9] of provisioning HA service is still in use currently. In this context, authors in [1] (see Fig. 2), propose an SFC sub-chaining method to enhance the reliability of an SFC in softwarized 5G networks. This method can be incorporated if there is sufficient time to build the sub-chain, like during regular system updates and reconfiguration. But, the authors have not performed any such practical experiments in the 5GC prototype which could indicate the total time of sub-chaining. Therefore, these works listed here and the factors in the motivation (section II-C), inspired us to dive deep into addressing the resilience and HA of the 5GC during unexpected overloads and single-point failure of LB. Hence, we propose LOCOMOTIVE 5G Core.

III. LOCOMOTIVE 5G Core

We design the proactive LOCOMOTIVE framework shown in Fig. 3c to offer multiple functions in a single box. First, we empower one of the serving instances of NF, with an additional capacity of LB. i.e., in terms of resources required by the LB to serve as primary LB upon the first failure of the current LB. We term this NFI as **LOCOMOTIVE NFI**. Upon the first failure of the primary LB, the LOCOMOTIVE NFI takes over the role of the primary LB. But, now its serving capability as an NFI will limit to the same as that of other NFIs. Additionally, we place the LB's functionality in each of the other NFIs too. We do this to activate one among them as a primary LB instantly when required, (like multiple failures of primary LB in a close time interval) to survive the situation(s) for HA. The chosen potential LB during these conditions serves as an *LBonly*, sacrificing itself so that other NFIs can continue to serve the USRs. However, since such situations are very rare, these NFIs are not additionally powered like the LOCOMOTIVE NFI. This helps in saving the unnecessary wastage of resources.

The HA of NF in the LOCOMOTIVE is given as

$$A_{NF} = (1-(1-A_{NF-LB})^{1+n}) * (1-(1-A_{NFI})^{n})) \quad (4)$$

for 'n' number of NFIs (serving instances of NF).

The actual flow of LOCOMOTIVE is driven by Algorithms 1 and 2 (that function as per the framework shown in Fig. 3c). They run at UDSF and are detailed in the steps below.

Algorithm 1: Find the next potential LB.

```
Input: $availMemOfNFInstances$ and
       $availCPUOfNFInstances$ at time $i$,
bool $isLocomotiveNFIAvailable$;
1  if $isLocomotiveNFIAvailable$ then
2      return 1;
3  $optimalPotentialLBIndex = -1$;
4  $prevMinWeight = \infty$;
5  $alpha = 0.5$;
6  for $nfIndex \leftarrow 2$ to $numberOfAvailNFInstances$
     by 1 do
7      $availMem \leftarrow$
         getNormVal($availMemOfNFInstances[nfIndex]$);
8      $availCPU \leftarrow$
         getNormVal($availCPUOfNFInstances[nfIndex]$);
9      if ($availMemOfNFInstances[nfIndex] <$
         $minMemoryOfLB$) then
10         continue;
11     $weight \leftarrow$
         $(alpha * availMem) + (1 - alpha) * availCPU$;
12     if $weight < prevMinWeight$ then
13         $optimalPotentialLBIndex = nfIndex$;
14         $prevMinWeight = weight$;
15 return $optimalPotentialLBIndex$;
```

1) Algorithm 1 runs periodically to keep the next potential LB ready based on the current load information update by every NFI to UDSF. Here, we define the load as the availability of sufficient resources at the NF instance i.e., memory and CPU, to become the next potential LB. Index of LOCOMOTIVE NFI ('1') is returned if it

Algorithm 2: Handling the failure of primary LB.

```
Input: bool $isLocomotiveNFIAvailable$
1 $isPLBAvailable \leftarrow$ PLB.healthStatus();
2 if !$isPLBAvailable$ then
3     $nfRole$ = NFAndLB
4     if !$isLocomotiveNFIAvailable$ then
5         $nfRole$ = LBOnly;
6     $potentialLBInfo$ =
        setLB($optimalPotentialLBIndex$);
7     notifyNFInstanceForPLB($nfRole$,
        $potentialLBInfo$);
8     sendLBInfoToNRF($potentialLBInfo$);
```

is available. Otherwise, it runs through the rest of the available NFIs. $alpha$ is a smoothing factor. We choose it as '0.5', giving equal weightage to the available memory and available CPU of the respective NFI, in order to ensure finding the most eligible next potential LB among all the available NFIs.
The next potential LB should have $memory >= minMemoryOfLB$ as it will serve as the primary LB only. This way, during the unusual situations of multiple failures of primary LB, one of the NFIs is sacrificed (selected) as the primary LB temporarily, to survive the situation so that the other NFIs can continue to serve the USRs. But, the CPU is not considered here in a similar way as that of memory. This is because, the selected NFI is sacrificed and hence it no longer serves the USRs in the role of NFI but leverages its processing power to serve as primary LB only. However, the operator can tune the $alpha$ if one among memory and CPU needs to be prioritized. It is to be noted that we have used a centralized approach (i.e., using the load update procedure from every NFI to UDSF in Algorithm 1) to select the potential LB. Instead, Consensus[1] algorithms like Paxos or Raft can also be used to find the next potential LB.

2) Further, in Algorithm 2, when the primary LB suddenly goes unavailable, UDSF detects it with the heartbeat[2] and immediately notifies the LOCOMOTIVE NFI if it is available. Otherwise, it notifies the ready-to-use new potential LB ('$optimalPotentialLB$') obtained from the output of Algorithm 1. Finally, a notification is also sent to NRF about the updated LB.

In the whole process of fail-over, we depict the total fail-over time as

$$T_{FO} = T_{DF} + T_{FS} \quad (5)$$

Where T_{FO} is the total fail-over time consumed to get a new primary LB ready to operate. T_{DF} is the time to detect the failure of the current primary LB, and T_{FS} is the time taken to synchronize the scheduling state information at the new primary LB.

IV. LOCOMOTIVE Prototype in 5G Core

To evaluate the performance of the proposed LOCOMOTIVE, we build its complete prototype on AMF [3], as AMF is the single entry point at 5GC for handling all the control plane USRs. Each of the entities in the LOCOMOTIVE framework are hosted on Intel® Xeon® CPU E5-1650 v4 @ 3.60GHz with 6 CPU cores and 32GB RAM. The memory and processing power of the LOCOMOTIVE AMF instance (LOCOMOTIVE NFI) is the same as other AMF instances (NFIs) as they are sufficient to show the benefits of the LOCOMOTIVE.

A. 5GC SBA Development

For 5GC SBA, we have used the in-house 5GC prototype based on 3GPP Release 15 [3] presented in our previous work [10] along with the RAN+UE emulator. The RAN+UE emulator here emulates the UE behaviour with the USRs and the respective provisioning by RAN towards the 5GC. While we plan to open source our 5GC testbed code soon, these development experiments can be reproducible with existing open source 5GC projects like free5GC, OAI 5GC, Open5GS, etc.

B. LOCOMOTIVE AMF and Load Balancer

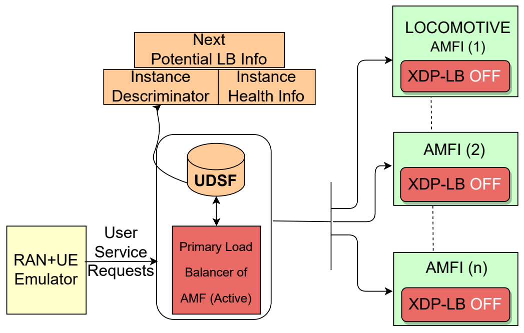
Fig. 4a shows the end-to-end framework of LOCOMOTIVE setup before the failure of primary LB of AMF. Here, we extended our previous work [5] on designing an autonomous LB. So, we leverage XDP/eBPF [6] availing full benefits of its performance acceleration for load balancing [6] versus Linux Kernel based IPVS. We developed the LOCOMOTIVE LB in an allowed restricted 'C' language, compile it into custom byte code, and load it into the kernel.

We used the minimum XDP/eBPF supportive vanilla Linux 5.6 kernel version Intel® Xeon® CPU E5-1650 v4 @ 3.60GHz system with Ethernet controller Intel Corporation I210 1 Gigabit Network Connection speed 'igb' driver. Fig. 4b, shows the internal view of XDP-eBPF based LOCOMOTIVE AMF. As shown here, we hooked the LOCOMOTIVE AMF LB byte code in XDP driver mode to process the packets immediately after NIC. In the same system, we load the AMF instance into the user space. So, each of the VMs picked for serving USRs, hosts LOCOMOTIVE's AMF instance in the user space to serve the USRs and XDP/eBPF based AMF LB code in the kernel driver, but set to 'OFF'. ('OFF' here indicates that the LB function is not triggered as it just functions as the serving AMF instance till the primary LB failure).

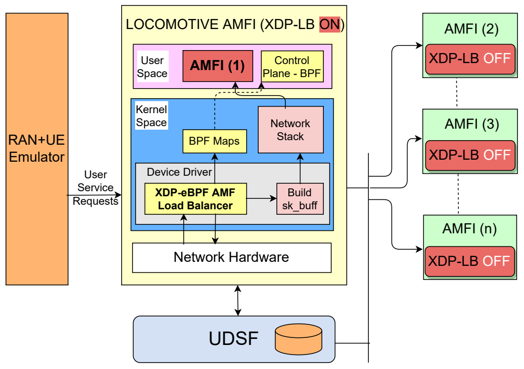
The primary LB schedules every USR received from the RAN to an appropriate AMF instance. The XDP driver code in the AMF instance simply passes the USR to the user space. The AMF instance in the user space handles the USR and responds to the RAN directly. Upon the failure of primary

[1]Consensus is the process of agreeing on one result, where all the NFIs can vote for the leader election and select an NFI as the next potential LB.

[2]Heartbeat based failure detector is a popular approach to detect the failure or unreachability of a network entity.



(a)



(b)

Fig. 4: End-to-End setup with LOCOMOTIVE framework a) before and b) after the failure of primary LB of AMF.

LB, UDSF detects it (as we use TCP heartbeat here) and notifies LOCOMOTIVE AMF instance immediately as per Algorithm 2. Fig. 4b shows the experimental setup after the failure of the primary LB of AMF. Based on this immediate indication, the XDP/eBPF in the driver marks the code to take up the role of LB. And hence, the LB code in the driver gets executed for every USR received from the RAN, making use of the scheduling information from UDSF regularly. The remaining AMF instances continue to serve the USRs as before, with their XDP/eBPF code for AMF LB set to 'OFF'.

V. Performance Evaluation

A. Experimental Setup

To prove the key design aspects of LOCOMOTIVE we evaluate its working for resilience, resource utilization, and HA using the end-to-end setup shown in Fig. 4a and Fig. 4b. For comparison, we pick the cold standby (see Fig. 3a) and hot standby (see Fig. 3b) methods (see Section II-B).

B. Data Set of User Service Requests

We used the data set from the previous work [15] for three primary slice types (eMBB, uRLLC, and mMTC). Other specific characteristics of the slices are not considered as they are outside the scope of this work. The data set for every slice type has the USRs connecting to the 5GC per second, for one complete day. i.e., a maximum of '100' per second for normal conditions and a maximum of '120' per second representing overloaded situations.

C. Performance of LOCOMOTIVE's Resilience

To measure the resilience during the LB failure, we capture the amount of impacted USRs by referring [16]. Though multiple combinations are possible for the unexpected failure of primary LB, we pick a few instances of such situations tabulated in Table II to evaluate Algorithm 1. The following parameters are set in consistency across all the three methods (cold standby, hot standby, and LOCOMOTIVE) used for comparison.

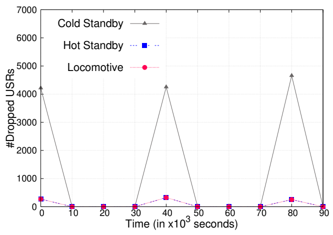
- The ground truth capacity of processing 25 $USRs$ per second at every instance of AMF. This number '25' is derived from the time taken to complete a single UE registration (a type of USR) measured as '40' ms, from our 5GC prototype [10].
- '4' serving AMF instances and a primary LB available at the start of the test. This primary LB is capable of scheduling 120 $USRs$ per second.

TABLE II: Situations of sudden failure of primary LB

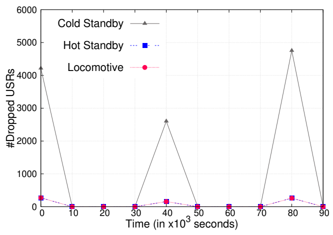
LB Failure Situation	**Number of LB Failures**	**Description**
Normal	1 in a distant time interval.	Number of USRs arriving is within the expected limit.
Overloaded-1	1 in a distant time interval.	More USRs arrive than the expected limit.
Overloaded-2	2 in a close time interval.	More USRs arrive than the expected limit.
Overloaded-3	2 in a close time interval.	Overloaded-2 + One NFI (other than LOCOMOTIVE NFI) fails at the same time as that of 1st failure of LB.

Additionally, we capture the total fail-over time (T_{FO}) using Eqn. 5 from LOCOMOTIVE prototype in Section. IV as '5' seconds. This value remains the same in hot standby too. But, for cold standby, it accumulates to '80' seconds for a new instance to get ready (see Section. II-B1) and Section. II-C).

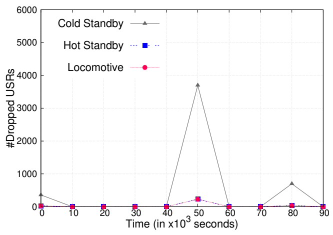
Fig. 5 compares the dropped USRs in cold standby, hot standby, and LOCOMOTIVE for eMBB [Fig. 5a], uRLLC [Fig. 5b], and mMTC [Fig. 5c] over one complete day in the granularity of seconds, with 3 non-consecutive distant failures of primary LB, when the respective slice traffic is peak. Using this, Fig. 6 shows the Cumulative Distribution Function (CDF) of the number of dropped USRs in all three methods, across all three slice types. i.e., for eMBB in Fig. 6a, uRLLC in Fig. 6b, and mMTC in Fig. 6c. This confirms that the cold standby incurs a higher number of dropped USRs while the hot standby and the proposed LOCOMOTIVE show a high probability of dropping less number of USRs across all the slice types.Next, we compare LOCOMOTIVE and hot standby as per Table II. As shown in Fig. 7a, before the primary LB fails at 25th second, in normal and '$overloaded - 1$' situations, LOCOMOTIVE drops a lesser number of USRs compared to the hot standby, using its LOCOMOTIVE NFI



(a)

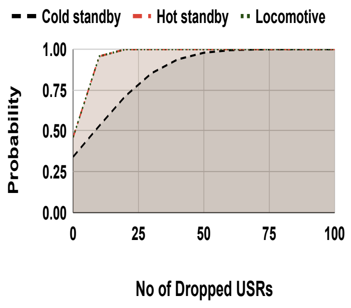


(b)

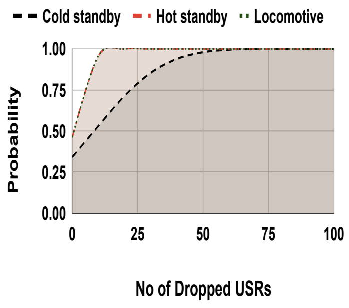


(c)

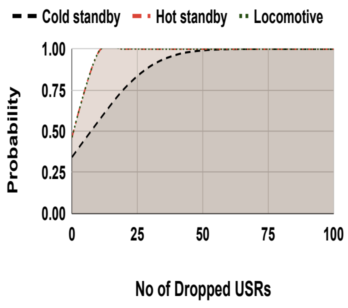
Fig. 5: Dropped USRs when primary LB failed 3 times in a day with peak traffic in a) eMBB, b) uRLLC, and c) mMTC.



(a)



(b)



(c)

Fig. 6: CDFs of number of dropped USRs in a) eMBB slice, b) uRLLC slice, and c) mMTC slice.

(it leverages LB's processing and memory resources to serve additional USRs). However, upon the failure of primary LB, both methods incur the same number of dropped USRs, as their total fail-over time ('5' seconds, see Eqn. 5) is the same.

Further, when the primary LB fails twice in a given time period of '80' seconds, LOCOMOTIVE outperforms the hot standby from the second failure of the primary LB in '$overloaded-2$' [Fig. 7b] and '$overloaded-3$' [Fig. 7c] situations. This is because in LOCOMOTIVE one of the NFIs sacrifices itself to serve as the primary LB so that other NFIs can serve at their maximum capacity and ensure resilience. Nevertheless, the chained failures of LB are very rare. So, to address this, the operator can anticipate traffic conditions like the work [17] and deploy the required number of LOCOMOTIVE NFIs accordingly.

D. *Performance of LOCOMOTIVE in Resource Utilization*

To evaluate the resource utilization by LOCOMOTIVE, we once again use the '$overloaded-1$' situation from Table II, but without failing the primary LB. Fig. 8a compares the number of dropped USRs between hot standby and LOCOMOTIVE. Here too, LOCOMOTIVE outperforms the hot standby by serving at least 16% more USRs. This is because LOCOMOTIVE uses the standby LB's resource in serving additional USRs rather than reserving it for standby LB (see Section III). Hence, LOCOMOTIVE is **super** beneficial in handling the sudden increase in traffic to the best of its capacity, if the primary LB never fails.

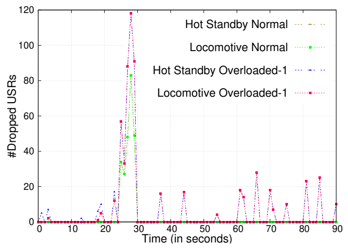
E. *Performance of LOCOMOTIVE's High Availability*

To compare the overall HA achieved between the methods, we trigger the failure of primary LB at 60^{th} second, 110^{th} second, and further at 200^{th} second (in alignment with Table. II), for a total time of '7' minutes testing. Total HA of the AMF is computed for cold standby, hot standby, and LOCOMOTIVE using Eqn. 1, Eqn. 3, and Eqn. 4 respectively. As observed in Fig. 8b, the cold standby delivers no service from 60^{th} second to 140^{th} second and again from 200^{th} second. On the other side, though hot standby is 92% available till 110^{th} second, another failure of primary LB leads to its zero availability from 111^{th} second onwards, till an active instance, comes up at 140^{th} second for the first failure. However, LOCOMOTIVE is resilient throughout, with its availability varying between 99.3% to 96% only as all the instances of AMF contribute to the HA, except dropping at 200^{th} second with a momentary 91% availability, due to highly overloaded conditions.

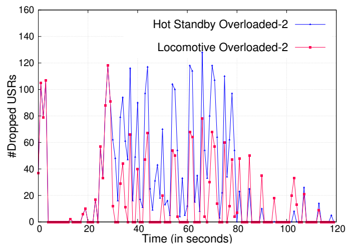
Further, Fig. 8c depicts the total number of VMs consumed versus HA, between hot standby and LOCOMOTIVE, during this testing time of '7' minutes. The total number of VMs is the sum of the number of LB(s) and the number of AMF instances used. As observed here, hot standby consumes an additional VM for standby LB. On the other hand, though LOCOMOTIVE empowers the LOCOMOTIVE AMF instance with additional resources equivalent to the standby LB of hot standby, it uses them efficiently to sustain all the situations.

VI. Conclusion and Future Work

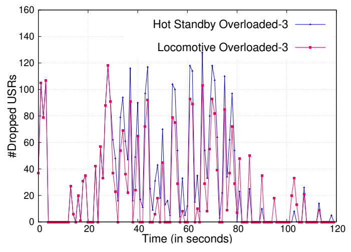
This paper proposed a LOCOMOTIVE 5G Core that is super resilient to provide High Availability service before and



(a)

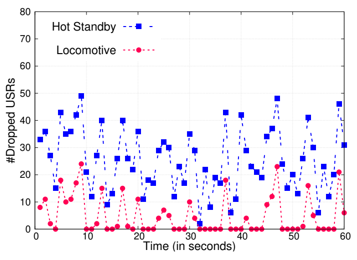


(b)

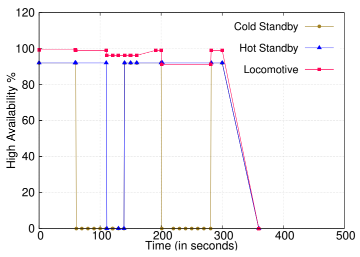


(c)

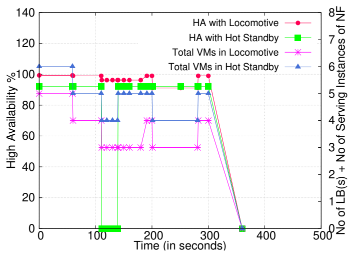
Fig. 7: Dropped USRs with one or more primary LB failures during different overloaded situations.



(a) Dropped USRs in overloaded conditions.



(b) High Availability during primary LB failure(s).



(c) High Availability versus total VMs during primary LB failure(s).

Fig. 8: Resilience and High Availability analysis.

after the failure of primary LB. Our study with experiments on the 5GC testbed has shown that the LOCOMOTIVE outperforms hot standby in multiple ways. It serves at least 16% more USRs than the hot standby while handling unexpected overloaded conditions. Further, it surpasses the hot standby once again, when primary LB fails multiple times. We hope to encourage further research works in the 5GC to aid in building a resilient and smart orchestration of AI focused 6G networks. Therefore, in the future, we plan to build an AI based model to predict the serving schedule of USRs on our proposed LOCOMOTIVE 5GC.

Acknowledgment

This work has been supported by the Department of Telecommunications, Ministry of Communications, India as part of the "Indigenous 5G Test Bed" project.

References

[1] Prabhu Kaliyammal Thiruvasagam, Vijeth J. Kotagi, and Siva Ram Murthy, "A reliability-aware, delay guaranteed, and resource efficient placement of service function chains in softwarized 5g networks", *IEEE Transactions on Cloud Computing*, pp. 1–1, 2020.
[2] L. Ma, X. Wen, L. Wang, Z. Lu, and R. Knopp, "An SDN/NFV based Framework for Management and Deployment of Service Based 5G Core Network", *China Communications*, vol. 15, no. 10, pp. 86–98, Oct 2018.
[3] 3GPP, "System Architecture for the 5G System", Tech. Rep. TS 23.501, 2021.
[4] NGMN, "6G Drivers and Vision", https://www.ngmn.org/wp-content/uploads/NGMN-6G-Drivers-and-Vision-V1.0_final.pdf.
[5] Shwetha Vittal and Antony Franklin A, "HARNESS: High Availability supportive Self Reliant Network Slicing in 5G Networks", *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.
[6] Toke Hoiland-Jorgensen, Jesper Dangaard Brouer, Daniel Borkmann, John Fastabend, Tom Herbert, David Ahern, and David Miller, "The express data path: Fast programmable packet processing in the operating system kernel", in *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies*, New York, NY, USA, 2018, CoNEXT '18, p. 54–66, Association for Computing Machinery.
[7] eBPF Working Group, "eBPF", https://ebpf.io/, 2021.
[8] ETSI, "Network Functions Virtualisation (NFV) Release 3;Reliability;Report on NFV Resiliency for the Support of Network Slicing", Tech. Rep. ETSI GR NFV-REL 010 V3.1.1, 2019.
[9] ETSI, "Network Functions Virtualisation (NFV) Release 3;Reliability;Report on Models and Features for End-to-End Reliability", Tech. Rep. ETSI GS NFV-REL 003 V1.1.2, 2016.
[10] Shwetha Vittal, Aditya Chilukuri, Sourav Sarkar, Akshitha Shinde, and Antony Franklin A, "Performance study of large scale network slice deployment in a 5g core testbed", in *2021 IEEE 4th 5G World Forum (5GWF)*, 2021, pp. 311–316.
[11] Bo Han, Vijay Gopalakrishnan, Gnanavelkandan Kathirvel, and Aman Shaikh, "On the resiliency of virtual network functions", *IEEE Communications Magazine*, vol. 55, no. 7, pp. 152–157, 2017.
[12] ETSI, "Network Functions Virtualisation (NFV) Release 2;Reliability;Report on Scalable Architectures for Reliability Management", Tech. Rep. ETSI GS NFV-REL 002 V1.1.1, 2015.
[13] Zhijing Li, Zihui Ge, Ajay Mahimkar, Jia Wang, Ben Y. Zhao, Haitao Zheng, Joanne Emmons, and Laura Ogden, "Predictive analysis in network function virtualization", in *Proceedings of the Internet Measurement Conference 2018*, New York, NY, USA, 2018, IMC '18, p. 161–167, Association for Computing Machinery.
[14] Junichi Kawasaki, Genichi Mouri, and Yusuke Suzuki, "Comparative analysis of network fault classification using machine learning", in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–6.
[15] Shwetha Vittal and Antony Franklin A, "Self optimizing network slicing in 5g for slice isolation and high availability", in *2021 17th International Conference on Network and Service Management (CNSM)*, 2021, pp. 125–131.
[16] ENISA, "Resilience Metrics and Measurements: Technical Report", https://www.enisa.europa.eu/publications/metrics-tech-report.
[17] Imad Alawe, Yassine Hadjadj-Aoul, Adlen Ksentini, Philippe Bertin, Cesar Viho, and Davy Darche, "Smart scaling of the 5g core network: An rnn-based approach", in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.