REARM: Renewable Energy Based Resilient Deployment of Virtual Network Functions

Sameer G Kulkarni*, Mayutan Arumaithurai*, K.K. Ramakrishnan† and Xiaoming Fu*
*University of Göttingen, Germany, †University of California, Riverside.
{skulkar, arumaithurai, fu}@cs.uni-goettingen.de; kk@cs.ucr.edu.

Abstract—Network Function Virtualization (NFV) is becoming more prevalent in Data Center, Telecommunication and Enterprise networks, enabling the Virtual Network Functions (VNFs) to fast replace the traditional dedicated hardware based middleboxes. Ensuring high availability and fault tolerance of VNFs is cardinal to meet the performance and service level agreement requirements. Also, with the increasing electricity demands in the Information and Communications Technology (ICT) sector, especially for the data centers, the inclination towards employing renewable (green) resources to power up the data centers is also increasing. Mitigating the carbon footprint and curbing the energy costs have been the driving factors for push towards employing the green energy resources. However, the Green energy supply is rather intermittent and unstable.

In this work, we study the impact of deploying VNFs in Green Data Centers (GDCs) and make a case for addressing the VNF reliability and high availability to effectively tackle the stability concerns of GDC. To this extent, we present REARM, which adopts the concept of Transient VNFs that rely on a very short advance warning time to seamlessly migrate the VNFs from GDC to a more reliable and stable Data Centers (SDCs). Our experiments with container based VNFs demonstrate that adaptive state transfer mechanism results into significant reduction in both computation and communication overheads for maintaining the NF replica, and warning time of 30ms is sufficient to failover VNFs (serving 1K flows) within a data center to ensure high availability of NFV services.

I. INTRODUCTION

Network functions such as Firewalls, Intrusion Detection Systems (IDS), cache optimization, load balancing, etc. have become an integral part of large scale enterprise and Data center (DC) networks. Network Function Virtualization (NFV) enables the deployment of software based middleboxes also known as the Virtualized Network Functions (VNFs) on top of the commercial off-the-shelf (COTS) hardware rather than using the dedicated hardware appliances. This allows for flexible realization of network services with greater cost optimization both in-terms of capital expenses (CapEx) and operational expenses (OpEx). More importantly, NFV catsers towards better energy efficiency due to consolidation of compute and network resources [1]. Hence, VNFs are fast replacing the traditional middleboxes (dedicated hardware appliances). Typical use cases in DCs, and Telecommunication require the network flows to be subject to different policies based on the security requirement, service level agreements and desired quality-of-service. These policies are often realized through a chain of network-resident services that require to be processed in a specific sequence. This construct is referred to as Service Function Chaining (SFC) [2], which is fundamental to most of Glian, Enterprise and Data center networks. It has been observed that middlebox failures can result in service outage and incur significant loss of revenue [3, 4], hence the need for high availability and fault-tolerance is important for the middleboxes and NFVs.

It has been studied that the DC industry accounts to over 30 Gigawatts of energy per year [5], accounting to roughly 21% of energy accounted by Information and Communications Technology (ICT) [6], and the demand keeps increasing every year. The carbon footprint of a medium 10 Megawatt data center can range from 3,000,000 to over 130,000,000 kilograms of CO2 [7]. Depending on the electric grid region, Power Usage Effectiveness (PUE) improvements can eliminate millions of pounds of CO2 emissions [8]. These factors have led to tremendous increase in the widespread adoption of renewable resources for powering the data centers. The recent study [9] indicates a phenomenal increase in the investments ($285.9 billion) for harnessing renewable energy, which is more than double ($130.6 billion) the investments on non-renewable energy resources in 2016. It is also noteworthy that the amount of renewable energy generation capacity has increased by nearly 56 percent over last two years. Greenpeace report [6] indicates that already the companies like Apple, Facebook and Google have started adopting renewable energy to power the data centers.

Despite growth, the nature of renewable resource based power is i) not sufficient to fully power the large data centers, ii) highly intermittent and unstable [10]; hence pose a greater challenge in adopting them for the large data centers which require stable and sustained power resources in-order to avoid any service disruptions. However, the Green energy could be used to adequately power a small DC with reasonable degree of reliability [11].

To this end, we present our work REARM1, that aims to enable running the VNFs in renewable energy backed data centers while providing sufficient degree of reliability and high availability. REARM makes use of the most preferred active-standby redundancy mechanism with many-to-many2 backup model [4], [12] to address the sustained VNF availability even in the unstable and intermittent power outage issues faced by GDCs. We have also built custom stateful NFs to demonstrate practical benefits of REARM. The key contributions of our work include:

---

1REARM: Renewable Energy Based Resilient Deployment of VNFs.
2Multiple NF instances of same type in GDC can be backed-up by one or more NF instance in SDC.
to VMs that range in several giga bytes [14]. VNF is minimal (order of few bytes to kilo bytes) compared to VMs that range in several giga bytes. Hence, stateless VNFs only need the routing state also needs to be updated for reliable processing of subsequent packets. Therefore, stateless VNFs only need the routing state also needs to be updated for reliable processing of subsequent packets. This diversity not only hinders portability - since the Docker based and process based VNFs need to be backed-up on nodes matching the hardware and Operating system requirements, but also pose a challenge towards achieving generalized framework for replication as the needs and means to snapshot and back-up the VNFs significantly differ. However, the promising part of the VNF diversity, especially with the Unikernel based VM’s and containers is that the amount of data that need to be backed-up is significantly lowered (order of few mega bytes) compared to the traditional VMs that range in giga bytes.

Service Function Chaining: The flows served by the VNFs are typically subject to more than one network functions, processed in a specific order, e.g., NAT, Firewall, IDS, and Load-balancer. This implies that in-order for the flow/packets to be processed consistently across the replicas, the VNFs cannot be treated in isolation, but the chain (ordered list) of VNFs that a flow/packets go through need to be treated as a group. Hence to maintain the VNF state consistency across the chain, the back-up and snapshot mechanism should consider the periodicity for group of VNFs.

VNF State Anatomy: We leverage the study and analysis of earlier works [4], [14], [15] in discerning the anatomy of internal states. Based on the study of existing VNFs, they can be broadly classified into i) stateless VNFs (those that do not maintain any state for VNF packet processing) e.g., stateless firewalls. ii) stateful NFs - that maintain and store the state for the packets/flows being processed by the NFs. e.g., IDS. Further, the stateful NFs can be sub categorized into i) VNFs with per flow status - that maintain and update states for each of the new flows e.g., Application Delivery controllers and stateful Firewalls ii) VNFs with per packet status - that maintain and update state for every individual packet processed by the VNF e.g., IDS.

In general, the state maintained by the VNFs can be broadly categorized into i) Internal - or the ephemeral state, which do not affect the consistency and may deviate across the replicas, ii) External state - typically constitute the flow specific information and counters, which needs to be kept updated across replica, and iii) Coherent state constitute the global counters and configuration parameters, which also needs to be kept consistent across multiple replicas [14]. Hence, to maintain the replica, different types of VNFs demand different kinds of state with different levels of state synchronization. We take advantage of this aspect in our work to account for periodicity of state updates.

B. Data Center Power Infrastructures

State-of-the-art data center power delivery infrastructure can support multiple power sources, allowing some of the racks in data center to be completely powered by the renewable resources and part of the racks to be powered by the non-renewable based sources [16], [17]. We note that, with such configurations, the standby (backup) nodes can be maintained within the same data center on the racks powered by non-renewables. In the case of GDC that is fully powered by

II. BACKGROUND AND MOTIVATION

A. VNF Diversity: Challenges and Opportunities

Virtual machines (VMs) are generally the application processing engines, characterized by the application states, whereas the middleboxes or the VNFs cater towards diverse set of use cases and applications like the network services, e.g., NAT, web proxies etc.; security services, e.g., firewalls, intrusion detection and prevention, encryption etc.; and the value added services like parental control, WAN optimizer and HTTP enrichment etc., resulting in diverse complexity and processing requirements. The ETSI standards [13] broadly classifies the VNF types into four classes namely i) data plane, ii) control plane, iii) signal processing, and iv) storage, based on the resource (computation, communication, memory and storage) requirements and performance (throughput, latency, jitter, and packet loss) requirements. VNFs are essentially the high speed packet processing engines that maintain flow/packet specific states and tend to serve millions of packets per second at 10G/40G/100Gbps line rates. This implies that the frequency at which the VNFs state change can occur is too high and the downtime in the order of milliseconds can lead to severe service disruptions. Hence, to achieve high-availability, consistent updates need to be done more frequently than compared to the traditional VMs.

We also note that, in addition to the VNF state, the network routing state also needs to be updated for reliable processing of subsequent packets. Hence, stateless VNFs only need the routing state update, but the stateful class of VNFs additionally need to snapshot their internal states. Also, the amount of internal state that need to be transferred to back-up the stateful VNF is minimal (order of few bytes to kilo bytes) compared to VMs that range in several giga bytes [14].

Fig. 1: Different NFV Deployment Approaches

- Distinction of NFV heterogeneity, different deployment models and the associated challenges in providing resiliency and high availability.
- Introduce REARM, that presents a generalized NFV framework to migrate VNFs efficiently and seamlessly in order to achieve high availability and resiliency to power instabilities of GDCs.
- Design and implementation of novel push-pull and adaptive threshold based state migration techniques, that enables to lower the warning times on the order of milliseconds.
- Preliminary evaluations on a prototype testbed using the controlled experiments to demonstrate the benefits of employing REARM.
the renewables, the backups will have to be maintained in another SDC which is powered by brown/non-renewable power resources.

As REARM intends to support for both the kinds of power infrastructure, we refer to the VNFs in GDC or the VNFs placed on a node that is powered by renewables as the Transient VNFs and the standby VNFs that are placed on a node powered by non-renewables either in SDC or in the same data center as stable VNFs or the replica VNFs.

Our main goal is to provide generalized framework for VNF state replication that exploits the opportunities and overcome the challenges (Refer II-A) to deploy the VNFs in GDC and to provide high availability of NFV services through efficient state replication mechanism.

III. RELATED WORK

We observe that the prospects of employing Green energy for the VNFs is a less studied topic. Nonetheless, we discuss some relevant work in the following overlapping categories.

**NF Migration:** Split/Merge [14] presents a FreeFlow system, and shared library APIs to access and update the internal state of the VNFs. It relies on the ability to identify per-flow state in middleboxes that need to be migrated. OpenNF [15], is a control plane architecture that facilitates for loss free state transfer of VNFs, but impose high per packet latency due to controller based orchestration and event buffering mechanism. We leverage the concept of [14] and present optimization to ensure critical state transfers are performed without incurring any additional latency.

**Fault Tolerance and High Availability:** Pico Replication [3] is a high availability, VNF state preserving framework built on top of Split/Merge by providing fine-grained flow level state replication. FTMB [4] is a replay based framework that employs periodic check-pointing and logging of determinants (shared variables in VNF that account to non-determinism) to restore the states on the replica. However, neither of the work account for SFC, REARM fills this gap by addressing chain level replication.

**Green Energy and Energy Efficiency:** In [1], authors analyze the prospects of energy efficiency that can be achieved by employing the VNFs for the Evolved Packet Core, Customer Premise Equipment, and Radio Access Network in telecommunication networks. This study is seminal in terms of establishing the energy efficiency prospects of VNFs. In [18], authors analyze and present the energy efficiency implications of NFV for different packet processing mechanisms. In contrast, we consider a more broader perspective and target towards achieving energy efficient network infrastructure that can be powered by renewable resources and still be able to meet the high availability and resiliency requirements.

IV. REARM ARCHITECTURE AND DESIGN

A. REARM: Architecture

We envision SDN based network with NFV management and Orchestrator to provision the VNFs on physical nodes. REARM’s architecture is based on the ETSI-NFV framework [13]. Figure 2 shows the high level architecture and key components of REARM. We briefly describe the role of the key components and concepts of REARM’s architecture.

![Fig. 2: REARM Architecture.](image)

1) **NFV Orchestrator:** NFV Orchestrator (NFVO) is responsible to manage, coordinate and communicate with the VNF managers to handle snapshot and restoration of stateful VNFs in a data center.

2) **VNF Managers:** VNF Manager is responsible to instantiate the VNFs and to perform snapshot and restoration of VNFs on a physical node.

3) **SDN Controller:** SDN Controller is responsible to implement the flow policies, i.e., to configure the service chain for the flows and to setup the forwarding rules/route for the flows to traverse through the desired chain of VNFs within the data center. Also, the SDN controller is responsible to migrate and re-route the flows to the appropriate VNFs in the data center.

4) **Advance Warning Time:** We leverage the YANK [11] concept of advance warning time (AWT) to backup the transient VNFs on the stable nodes. AWT refers to the estimated period until when the renewable energy backed universal power supply (UPS) is expected to last and sustain running the nodes reliably. The computation of AWT depends on the UPS capacity and the estimated power consumption of the rack. NFV orchestrator is responsible to estimate the warning time and communicate to the VNF managers and SDN controller.

5) **Snapshot and Restore Mechanism:** Both the stateless and stateful VNFs are backed-up on stable nodes and only the stateful NFs need to be periodically updated with the state changes. VNF manager is responsible to periodically transfer and synchronize the the VNF states of the Transient VNFs with the standby VNF replicas. With multiplexed backup scheme, REARM optimizes the VNF state transfer by combining the state of all the VNFs that are replicated on the same remote node.

**Multiplexed Backups:** We make use of the N:1 backup model, such that more than one transient VNFs in GDC can be backed up on a single node in the brown energy powered data center. In general, our backup scheme across DC is N:M, where N>M. This configuration helps to trade performance with power for the duration the VNFs are expected to run on brown energy. With multiple VNFs backed on single node in SDC, ensures to save on brown energy, but at the cost of increased latency and lower throughput for short period of time. Once the power levels are back to normal operating conditions, NFVO triggers for the restoration of transient VNFs from the stable data center.

---

3AWT need not be accurate, a rough estimate of around 40-50% of the total expected UPS discharge time is sufficient.
Addressing SFC: In order to account the service chains, we optimize by batching the updates for an entire chain of VNFs, so that the state of VNFs in a service chain is coherently synchronized and backed-up on stable services, instead of updating the state of each NF individually, which significantly reduces the communication overheads involved in the state transfer.

6) Routing update: When both the transient VNF and standby nodes are in the same data center, the routing update is simple and incurs less overhead, as it only involves setting new set of forwarding rules to redirect flows towards the new instance. However, routing across data-centers is more complex and incurs delay in the order of seconds [19].

B. Design

Figure 3 illustrates the VNF state migration mechanism of REARM platform. We summarize below the key steps illustrating the working of REARM.

VNF Setup (1A, 1B): When the VNFs are provisioned, the NFVO sets up the active (Transient VNFs) on nodes powered by renewables and the associated back-up nodes. Depending on the DC power Infrastructure, these can be mapped either on different nodes in same data center or on GDC and SDC respectively. VNF Manager on each of the host node is responsible to periodically extract the state updates of the transient VNFs and communicate to the VNF manager that hosts the standby VNF node. SDN controller is responsible to setup the forwarding rules for the flows that need to be processed by the VNF instances accordingly.

VNF State Tracking (2A, 2B): \textit{vnflib} periodically extracts (pull mode) the VNFs state change and sends the information to the VNF Manager. Also, the VNF can explicitly trigger to communicate the state changes at the end of packet processing. This is useful to save critical state changes instantaneously, especially in the case of NFs that track per-flow state and update critical states during processing of intermittent packets like stateful firewalls and IDS.

**VNF State Migration and updating (2C, 2D):** VNF Manager, upon receiving updates from the \textit{vnflib}, transfers the state to the back-up nodes. Upon receiving the packets, the restore manager on the standby node identifies the corresponding VNF and notifies the \textit{vnflib} to trigger for the state update.

Handling Downtime and restoration (3A-3D): Once the warning time is delivered by the UPS, the NFV Orchestrator (NFVO), communicates the warning time to all the VNF managers and waits for the notification of state transfer completion. VNF managers perform best-effort approach to complete the state transfers within the specified warning time. Upon completion notification, or timeout, NFVO directs the SDN controller to setup and update the forwarding rules to redirect the traffic towards the VNFs in SDC. In case of LAN (within same data center), the SDN controller updates the forwarding rules in the switches to route the packets to the destination node. But, in the case of wide area networks (WAN), the Multi-Protocol Label Switching (MPLS) based VPNs are created as an abstraction for private network address space shared across multiple data centers. NFVO in SDC also initiates the SDN controller and VNF managers on stable node to switch from standby to active mode. Once the power levels resume back to normal, the restoration from SDC to GDC is triggered and VNFs switch back their roles.

V. Implementation

We have implemented REARM using a DPDK-based NFV platform [20] for fast data plane processing, and leverage POX SDN controller to serve as REARM controller.

We also implemented custom stateful VNF variants (Monitor and packet logger) that maintain per flow and per packet states, with aggregate state information size of 8KB and 32KB, supporting upto 1024 flows as shown in Table I. Overall implementation\(^4\) of REARM components VNF Manager, \textit{vnflib}, excluding the custom VNFs is \(\sim 1200\) lines of code. All the VNFs link with the \textit{vnflib} that facilitates for VNF state import and export functionality.

**Communication Mechanism:** REARM provides a simple library called \textit{vnflib}, which abstracts the communication with the VNF Manager from the VNF implementation. \textit{vnflib} is responsible to transfer and notify the VNF state updates with the VNF Manager using the shared memory buffers. The \textit{vnflib} APIs allow the application code (VNFs) to export and import the VNF specific internal states are shown in Listing 4. At the time of initialization, VNF must register the callback function \texttt{callback\_fn()} with \textit{vnflib} library. As the state characteristics are intrinsic and distinct for each VNF, the payload to export the state is treated as opaque pointer by \textit{vnflib}.

**VNF Manager** implements the socket based communication protocol with the VNF Manager on the remote nodes to transfer the VNF states. In order to accommodate variable sized state information of multiple VNFs to be transferred in a single packet, we package each VNF’s state with tag-length-value format encapsulated in the UDP payload.

**Adaptive Threshold and Push-Pull based State Transfer:** \textit{vnflib} periodically (every 100\(\mu\)s) pulls the VNF state and

\[^4\]We only account the implementation added to support VNF state migration and communication framework.
TABLE II: Performance analysis using Apache bench, 10K web requests 32KB files with 500 concurrent requests

<table>
<thead>
<tr>
<th>Variant</th>
<th>Total Time</th>
<th>Mean Latency</th>
<th>Transfer rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (Monitor)</td>
<td>47.19s</td>
<td>4.712ms</td>
<td>6842.08 KBps</td>
</tr>
<tr>
<td>Monitor (MON-2)</td>
<td>49.52s</td>
<td>4.952ms</td>
<td>6510.12 KBps</td>
</tr>
</tbody>
</table>

export to VNF Manager. VNF Manager buffers it until state of all NFs in the chain is obtained. But, when any VNF updates state with push based API, the state is immediately transferred to ensure that critical VNF states are synchronized. Threshold is dynamically computed by VNF Manager, which determines the total bytes of data that can be buffered and transferred for each service chain. Threshold computation is based on the number of service function chains served by VNF manager, measured round trip time to update VNF state on the replica, and previously issued AWT.

VI. Evaluation

We performed preliminary evaluation on our university testbed. Our testbed has three Intel(R) Xeon(R) CPU E5-2697 v3 @ 2.60GHz servers, with dual-port 10Gbps DPDK compatible NICs, running Ubuntu SMP Linux kernel 3.19.0-39-lowlatency. We designate one each for traffic generation, GDC node and SDC node respectively. We make use of Moongen [21] and Apache bench to generate line rate traffic and HTTP web traffic with varying numbers of flows.

Our primary focus of evaluation is i) to analyze the overhead of REARM in terms of processing cost and throughput, necessary to perform VNF state transfer and update the replica ii) to quantify the effectiveness of REARM for different service chain lengths and iii) demonstrate how quickly the VNFs can be replicated and restored on the remote nodes.

A. Overhead analysis

Latency and Throughput: First, we evaluate the latency and throughput overhead for running HTTP web requests using Apache bench. In this setup, we launch a total of 10K web requests, running 500 concurrent sessions, where each flow routed through a VNF, downloads 32KB file from the nginx web server. We compare VNF that updates state for every processed packet with the baseline (without vnflib state transfer). Table II shows the impact on mean latency (measured across all the concurrent sessions, has an increase of 240 μs), and the aggregate transfer rate (measured across all the web requests dropped by 4.8%) are very minimal.

Next, we run the same experiment on a chain of 3NFs i.e., 1 VNFs with per-flow (MON-1) and 2 per-packet (MON-2 and PLOG) state update VNFs and capture the CDF of flow completion time as shown in Figure 5a. We can observe that the latency impact measured in terms of flow completion time is minimal and remains consistent without adding any additional tail latency i.e., the relative flow completion time of 90th, 95th, 99th percentile of flows is not impacted.

Computation overhead: Figure 5b shows the CPU overhead incurred by VNF using the vnflib for exporting the VNF state to the VNF Manager. For this experiment, we use the variants of Monitor VNF (baseline) and compare with the stateless, per-flow and per-packet state update to estimate the computation cost overhead added in each of these cases for performing the VNF state transfers. We set packet size of 500 bytes and vary the load using the DPDK Pktgen tool, and record the average CPU utilization of the core pinned for executing the VNF using the mpstat tool. We report the average CPU utilization reported over a period of 10 seconds for five different runs. We observe that CPU utilization for the stateless VNF is same as that of the Baseline VNF. And, compared to Baseline VNF, MON-2 (per-packet state update VNF) incurs highest CPU overhead, but even at 100% load (10Gbps), the average increase in CPU utilization is less than 5%, indicating the computation overhead of REARM is negligible.

B. NFV Resiliency and Warning Time Analysis

In Figure 6, we evaluate the VNF state transfer completion time for different service chain lengths, after issuing the AWT, which is set to 10 seconds based on the solar and wind energy powered data center traces in [11]. Setup for chain length 1,3 are same as before, and we add two more VNFs (MON-1 and MON-2) for the chain length of 5. We launch long running flows and vary the number of flows served by VNF chain, and measure the time taken to completely transfer the VNF state.

We can observe that the state transfer time increases with the increasing number of flows, and is in the order of few (1-30) milliseconds. The increase is primarily due to state transfers performed for each flow. Second, with the increasing chain length for a given number of flows, we observe that the state transfer time remains almost the same (only marginally increases) due to service chain based optimization. Also, we notice that 26 milliseconds is sufficient for 5NF chain serving 1000 flows to complete the VNF state transfer.

VII. Conclusion & Future Work

To summarize, we have characterized and analyzed the benefits and challenges in employing the Green Data Centers (GDCs) for VNFs. We have designed and implemented REARM, especially to cater towards the special needs of VNFs migration, and introduced the concept of Transient VNFs, which can be sufficiently restored to stable powered nodes within the specified advance warning time.

With prototype based evaluation on our SDN/NFV testbed, we have demonstrated the potential benefits of incorporating...
REARM towards achieving high availability and resiliency with VNFs, with the known warning times for the Green Data Centers. REARM exhibits extremely low computation and communication overheads (less than 5%), and is able to efficiently migrate VNF states for service chains serving thousands of flows in less than 30 milliseconds. We acknowledge that the network routing and communication latency for migrating the VNFs across WAN would be more demanding and challenging to address. Hence, we plan to extend our evaluation to extensively study the associated trade-offs and conduct cross-site and large scale data center topology based evaluation with real traffic traces.

ACKNOWLEDGEMENT

This work was supported by EU FP7 Marie Curie Actions CleanSky ITN project Grant No. 607584 and the joint EU Horizon2020/NICT ICN2020 Project, Contract No. 723014, and NICT No. 184.

REFERENCES