Managing Failures and Service Quality in the Context of NFV

Siamak Azadiabad
ECE, Concordia University
Montreal, Canada
siamak.azadiabad@concordia.ca

Ferhat Khendek
ECE, Concordia University
Montreal, Canada
ferhat.khendek@concordia.ca

***Abstract*— In the context of network function virtualization (NFV), virtual network functions (VNF) are the building blocks of network services (NS). VNFs are usually distributed applications composed of VNF components (VNFC). A VNFC instance is the actual consumer of resources and it is realized as a virtual machine (VM). Service availability of an NS is one of its important characteristics which has been expansively investigated in the literature. Fault tolerance is the main mechanism used to guarantee service availability. Fault tolerance relies on VNF redundancy and failover operation. It reduces the service outages when a complete failure of a VNF happens. However, complete failures are less frequent than partial failures in which only some VNFC instances of a VNF fail. Partial failures can cause service degradation and annoy tenants who usually expect a guaranteed service quality. To handle partial failures, the failover mechanism may not be ideal since it can cause a complete service outage. We, therefore, propose a solution to determine the redundancy of VNFs to guarantee the required quality of service for an NS and avoid service degradation below a defined level. We propose a framework that includes an architecture and operations to guarantee service quality and avoid potential service outage.**

Keywords*— *Network Function Virtualization (NFV), Network Service (NS), Virtual Network Function (VNF), Availability, Service Degradation, Service Quality.

I. INTRODUCTION

In the context of network function virtualization (NFV), a network service (NS) is composed of virtual network functions (VNFs) interconnected by virtual links (VL) [1]. Figure 1 shows an example of NS realized by interconnecting three types of VNFs (i.e., next-generation firewall (NGFW), load balancer (LB), and web server (WS)) to provide a web service functionality. VNFs and VLs in this figure use the virtual computing, storage, and networking resources of the underlying infrastructure. In the NFV context, a VNF (or VL) instance is created based on a VNF (or VL) type [2, 3]. The same VNF or VL type can have multiple instances in an NS. In the rest of this paper, *VNF* or *VL* represents a *type* of VNF or VL, and to refer to an instance, we use *VNF* or *VL instance*. In Figure 1, there are three instances of the NGFW, two instances of the LB, five WS instances, and one instance of each VL.

A VNF instance can be a monolithic application running on one VM. It can also be a distributed application using multiple Virtual Machines (VM). In the latter case, the VNF instance is composed of VNF components (VNFC) which are interconnected by internal VLs (IntVL). For example, Figure 2 shows the internal components of each NGFW instance in Figure 1. In this example, each NGFW instance is composed of three different VNFC types (i.e., firewall, intrusion prevention system, and network anti-virus) and each of these VNFC types has multiple instances. In this example, the NGFW instance is composed of eight VMs in total. In the rest of this paper, we refer to the VNFC type as *VNFC*, and the *VNFC instance* stands for an instance instantiated according to a VNFC type.

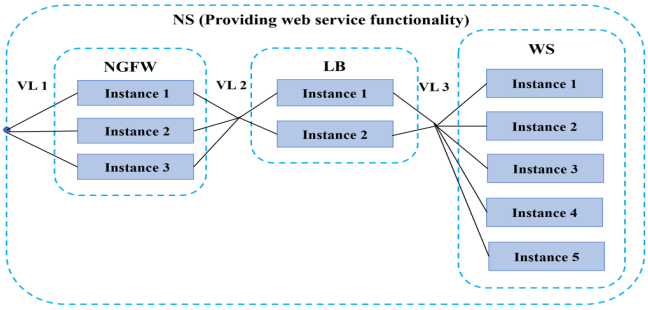

Fig. 1. Example of NS in the context of ETSI NFV.

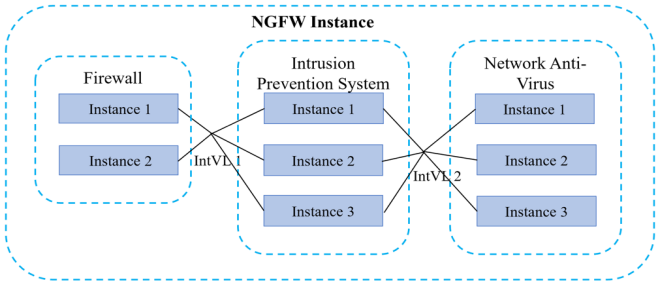

Fig. 2. Example of VNFCs for an NGFW instance in Fig.1.

An NS can have multiple scaling levels with different numbers of VNF instances to handle an increased workload and free resources in the case of decreased workload [2]. Similarly, a VNF instance can have multiple VNF scaling levels with different numbers of VNFC instances [3].

Service availability of an NS is defined as the fraction of the operation time the functionality of the NS is provided [4]. Traditionally, only a complete outage of the service contributes to service unavailability [5]. In other words, if the service of the NS is provided but in a

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

lower-than-expected capacity due to partial failures of VNF instances, it is not considered as service unavailability/outage. Similarly, at the VNF level, if a VNF instance is functional but some of its VNFC instances are failed, it is not considered a service outage. This is considered as service degradation, e.g., the throughput of the NS is reduced, the capacity in terms of requests per second for the web service is reduced, etc.

Service availability of NSs has been expansively investigated in the literature [5, 6, 7, 8, 9, 10, 11, 12, 13]. These works consider complete VNF instance failures in their solutions. A VNF instance is considered as failed if all instances of at least one of its VNFCs are failed [14]. This is an extreme case, but partial failures (e.g., failure of some instances of VNFCs) that cause service degradation are more common in an NFV environment. Considering the example of Figure 2, an NGFW instance fails if all instances of one of its VNFCs fail at the same time. In an NFV environment, the VMs of the NGFW are usually spread on different hosts and the probability of simultaneous failures for all instances of one VNFC is less than the probability of failure of one or some of them.

From the tenant's perspective, partial failures degrade the quality of service. Therefore, tenants are interested in indicating a threshold for the required service quality (RSQ) in addition to the required availability (RA) of the service. Accordingly, NS designers need to determine the required redundancy for VNFs to meet both of these requirements. Moreover, the NFV environment should support the required operations at runtime to maintain the satisfaction of the RSQ and the RA.

Service availability requirements have been addressed in [5, 14]. In this paper, we address the service degradation issue in addition to service availability. We propose a solution that consists of a method to determine the required VNF redundancy and a framework to guarantee the service degradation requirements for NSs. The rest of this paper is organized as follows. Section II introduces the NFV framework. Section III defines the problem at hand. Section IV presents our solution for guaranteeing the RSQ for NSs. Section V discusses related work. We conclude in Section VI.

II. The NFV Framework

The NFV framework manages the virtualization technologies to provide VNFs with virtual resources [1]. The NFV framework includes the NFV management and orchestration (MANO), the NFV infrastructure (NFVI), and the VNFs. Element managers (EM) and operations support system/business support system (OSS/BSS) collaborate with the NFV framework; however, they are not part of the NFV framework.

The MANO manages the lifecycle of VNFs and network services (NS) [15]. It is aware of the virtualization aspects of VNFs and NSs, but it not aware of the functionality of VNFs and NSs or the application-level configuration of VNFs. MANO includes the NFV Orchestrator (NFVO), the VNF Manager (VNFM), and the Virtualized Infrastructure Manager (VIM). The NFVO manages the lifecycle of NSs [15]. The VNFM manages the lifecycle of VNFs [15]. For example, the VNFM monitors the VNFs, and in case of VNF failure, it restarts or respawns the failed VMs. The VIM orchestrates the NFVI resources [15].

The NFVI includes the computing hardware, networking, and storage resources. It also includes the virtual resources that are created and managed by the virtualization layer (e.g., hypervisor). The VIM supports the basic management of virtual resources, like creating, deleting, or resizing VMs [1].

A VNF is a software implementation of a network function that can run on the NFVI [1]. The VNF definition includes both traditional network functions (e.g., firewalls, routers, etc.) and non-traditional network functions, such as web servers and databases [16]. The application configuration and the functionality of VNFs are managed by EMs. This includes fault, configuration, accounting, performance, and security (FCAPS) management [17].

The OSS provides the NFVO with the VNF and NS descriptors to onboard them. It can also send requests to the NFVO to instantiate, alter, or terminate NSs. The BSS supports business management through systems like billing and customer management [17].

III. Problem Statement

An NS encounters service degradation if some of the VNFC instances of its VNFs fail, and at least one instance of all the VNFCs for at least one instance of all the VNFs is healthy. In this paper, we assume that the availability and networking capacity of VLs and IntVLs are enough to meet the RA and RSQ requirements as these can be indicated for each VL/IntVL by the NS designer, and the MANO is expected to deliver them [2]. If there is no VNFC instance failure, there is no service degradation or service outage. If all instances of a VNFC fail, the corresponding VNF instance fails. If this happens to all instances of at least one of the VNFs of the NS, the NS encounters a service outage. Service availability guarantees have been addressed in [5, 14]. In this paper, we address the service degradation problem.

The RA can be expressed by tenants in terms of a percentage of a period (e.g., 99.999% of a year of availability). RSQ can be expressed as the percentage of network throughput, requests per second, or concurrent sessions, etc. For example, a tenant can request an *RA of 99.99% per year* and an *RSQ of 90% of the requests*, meaning the NS should be available for 31,553,796 seconds in a year (1 year = 31,556,952 seconds), and

during its availability time, the NS should be able to serve at least 90% of the end-users requests at any moment. For instance, if the requested capacity of the NS in terms of requests per second is 10000, this capacity should not fall below 9000 requests per second.

For genericity, one can map the *requests per second, concurrent sessions,* and *network throughput* of the NS to its processing capacity (i.e., the number of VNFC and VNF instances). Therefore, in this paper, we assume that tenants express RSQ as the percentage of the processing capacity of the NS.

In the context of NFV, NSs and VNFs can be designed with multiple scaling levels. Scaling levels of a VNF are designed by the VNF vendors and cannot be altered by the NS designers. However, the NS scaling levels are determined by the NS designers based on the processing power requirements of the service. The highest NS scaling level serves the highest expected workload of the NS, resulting in the highest network throughput, for instance. The lower scaling levels can help to improve resource efficiency and free resources for lower workloads. Each NS scaling level usually serves a range of workloads. For example, let us assume the NS of Figure 1 should support a workload between 10 to 1000 requests per second. Also, assume that the NS designer has created three NS scaling levels. Table I shows an example of the range of workloads that each NS scaling level can support.

TABLE I. RANGE OF WORKLOADS SUPPORTED BY DIFFERENT SCALING LEVELS OF THE EXAMPLE NS

	Minimum workload (Requests per second)	Maximum workload (Requests per second)
NS scaling level 1	10	100
NS scaling level 2	101	500
NS scaling level 3	501	1000

For the example of Table I, if an *RSQ* of *90%* is requested by the tenant, the NS can afford to lose at most 10% of its processing capacity for the maximum supported workload of each scaling level due to the failure of some VNFC instances. In other words, the RSQ should be guaranteed for the worst-case scenario of each NS scaling level assuming that the NS is serving the maximum workload of the scaling level. To guarantee the RA and the RSQ of an NS, we may need to add standby/backup instances to VNFs.

When a VNF instance fails (i.e., all instances of at least one of its VNFCs fail), failover can recover the lost capacity (if there are standby instances for the VNF) [18, 19, 20, 21]. The MANO is not aware of the active and standby roles of VNF instances. Usually, failover is managed by the VNF applications. The VNFM in the MANO only detects VM failures and restarts or respawns the failed VMs. The purpose of using the failover mechanism is to recover the service as soon as possible since restarting or respawning can be time-consuming. If only some instances of VNFCs of the VNF instance fail (i.e., we have service degradation), again the failover mechanism can restore the service capacity, but it may not be an ideal operation. Because, if we failover a VNF instance with degraded capacity, we interrupt the portion of the workload which is still being served by the remaining healthy VNFC instances. Failover in this case can result in a complete service outage. Ideally, we prefer to increase the capacity of the VNF to meet the RSQ without imposing complete service unavailability. Therefore, we need a new operation (in addition to failover) to support RSQ in the NFV environment without imposing outages.

IV. REQUIRED SERVICE QUALITY GUARANTEE

In this section, we provide a method to determine the required number of standby instances for VNFs to meet the required RA and RSQ of NSs. We also propose a framework that supports operations needed for RSQ guarantees.

A. VNF Redundancy Calculation

To calculate the required number of standby instances for each VNF of the NS, we should first determine the expected availability of each VNF. Let us assume that for the example NS of Figure 1, an RA of 99.99% and an RSQ of 97% are requested by the tenant. This NS has three different VNFs, and its availability is the product of the availability of its VNFs. Therefore, the expected availability of each VNF is:

$$EA_{VNF} = \sqrt[3]{RA} = 99.9967\% \quad (1)$$

In general, for an NS with n different VNFs, the expected availability of each VNF is:

$$EA_{VNF} = \sqrt[n]{RA} \quad (2)$$

The availability of a VNF is calculated based on the availability of its instances, which in turn is determined based on the availability of its VNFCs and the underlying resources. [14] proposes a method to calculate the availability of one VNF instance. So, in this work, we assume that the availability of instances of each VNF of the NS is known and denoted by A_{vnf}.

For a VNF with N active instances, we should determine the number of standby instances (M), so that the VNF meets its expected availability at least for the RSQ portion of its capacity. In other words, if A_{VNF} denotes the availability of the VNF, we should determine the minimum value of M which can satisfy inequation (3):

$$A_{VNF} \geq RSQ * EA_{VNF} \quad (3)$$

A_{VNF} for a VNF with N active and M standby instances is calculated using equation (4) [5]:

$$A_{VNF} = \sum_{k=0}^{M} \binom{N+M}{N+k} A_{vnf}^{N+k} * \left(1 - A_{vnf}\right)^{M-k} \quad (4)$$

Therefore, to determine the minimum value of M, we start from $M=0$ and calculate A_{VNF} using equation (4).

If A_{VNF} satisfies inequation (3), there is no need to add any standby instance to the VNF. Otherwise, we increment the value of M and recalculate A_{VNF} until inequation (3) is satisfied.

For illustration purposes, let us consider the example in Figure 2 and assume:

- The availability of each instance (VM) of the Firewall VNFC is 99.9 %,
- The availability of each instance of the Intrusion Prevention System VNFC is 99.8%, and
- The availability of each instance of the Anti-Virus VNFC is 99.7%

Let also assume that we want to determine the required number of standby VNF instances for NGFW of Figure 1, and the requirements at the NS level are RA = 99.99% and RSQ = 97%.

First, we should determine the availability of one VNF instance:

$$A_{vnf} = 0.999^2 * 0.998^3 * 0.997^3 = 0.98312 = 98.312\% \quad (5)$$

The expected availability for this VNF is:

$$EA_{VNF} = \sqrt[3]{0.9999} = 99.9967\% \quad (6)$$

According to inequation (3), the availability of the VNF should satisfy the constraint $A_{VNF} \geq 96.9968\%$. To determine the minimum number of required standby instances, we start from $M = 0$ and calculate A_{VNF} for $N = 3$. Using equation (4), we obtin $A_{VNF} = 95.02\%$ which does not satisfy the requirements. So, we recalculate A_{VNF} for $M = 1$ and we get $A_{VNF} = 99.83\%$, which is greater than 96.9968%. Therefore, one standby instance for the active instances of this VNF can satisfy the RA and the RSQ in this case.

B. A Framework to Support the Required Service Quality Guarantees

As mentioned earlier, when service degradation occurs for an NS, if VNF instances are not completely failed, failover can result in a complete service outage. To avoid this service outage, we should only increase the capacity of the affected VNF instead of failover. This can happen by switching the role of some standby instances to active without changing the role of active instances. To perform this runtime adjustment operation, we propose the following steps:

- **Step 1**: Monitor VNF instances at VNFC level.
- **Step 2**: Detect VNFC instances failures.
- **Step 3**: Determine whether corresponding VNF instances failed completely or partially.
- **Step 4**: In the case of complete failures, perform failover.
- **Step 5**: In the case of partial failures, determine if RSQ is violated. If RSQ is not violated, no further action is needed. Note that in case of a VNFC (VM) failure, the VNFC is restarted or respawned by the VNFM.
- **Step 6**: If RSQ is violated, determine the number of standby VNF instances that should be active to restore the processing capacity to the RSQ level.
- **Step 7**: Accordingly, switch the role of standby VNF instances to active.
- **Step 8**: Monitor failed VNFC instances and wait until they become alive (i.e., restarted or respawned). Once failed VNFC instances become alive, the processing capacity of degraded VNF instances is recovered.
- **Step 9**: Switch back the role of VNF instances that became active in Step 7 to standby.

In Step 5 of the runtime adjustment operations, we can also react proactively. Meaning, we do not wait until the RSQ is violated. If the RSQ is 90%, we can set up a threshold of 92%, monitor, and react as soon as this threshold is violated to guarantee the 90%. In this case, the reaction will be to switch one standby VNF instance to active to avoid RSQ violation if more VNFC instances fail. Acting this way, we can avoid the calculations in Step 6, and always activate one standby VNF instance in Step 7.

Note that a tenant can request an RSQ of 100% (i.e., no service degradation is acceptable). In this case, one has to react to every VNFC failure if the NS is dimenionsed exactly to provide the required quality of service. In general, in this case the NS is overdimensionned slightly, and one can follow the aforementioned operations for adjustment.

Service outage or unavailability is one (extreme) case of service degradation. This happens if one does not react to service degradation or when several failures happen simultaneously, which is less likely to occur. Therefore, reacting to service degradation and guaranteeing the satisfaction of RSQ contributes to service availability. On the other hand, service availability does not guarantee or mean satisfaction of the RSQ, because a service can be highly available with lower quality of service than requested.

To perform the runtime adjustment operations, an actor in the NFV architecture should take this responsibility. This actor should be:

- aware of the RSQ,
- able to monitor VNF instances at the VNFC level,
- aware of the current NS scaling level,
- aware of the current VNF scaling level, and
- able to configure application-level properties of VNF instances to assign active/standby roles.

Entities of the MANO are not aware of the application-level properties of VNFs and are not good candidates to perform the runtime adjustment. However, the EM is a candidate for this purpose. The EM can:

- receive the RSQ from the OSS (the OSS receives the RSQ directly from the tenant),
- monitor the internal components of its managed VNFs,
- get notified by the OSS when the NS scaling level changes (the OSS gets this notification from the NFVO), and
- be aware of the VNF scaling level and configure the VNF applications, since it can manage the FCAPS of VNFs.

Figure 3 shows the communications (i.e., texts in green) between NFVO, OSS, and EM to support the EM to perform the runtime adjustment operations. Figure 3 also shows that the VNFM restarts/respawns failed VMs, regardless of RSQ violation.

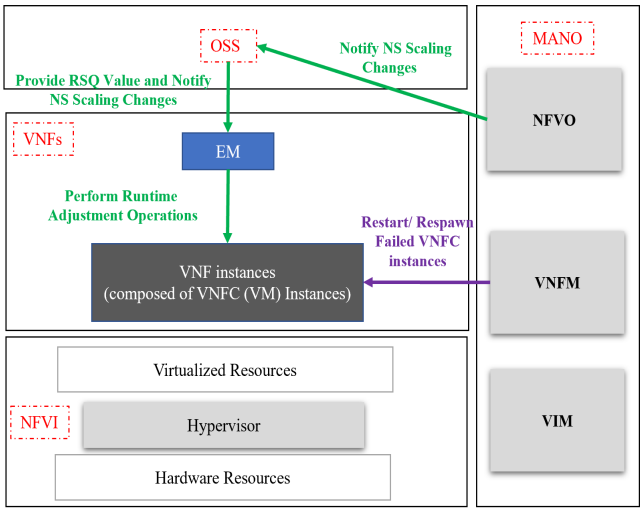

Fig. 3. Interactions between NFV architecture entities to support the EM with runtime adjustment operations.

V. Related Work

To the best of our knowledge, service degradation in the context of NFV has not been tackled in the literature. Related work focuses mainly on service availability and reliability. In addition, partial VNF instance failure (i.e., failure of some VNFC instances) and its impact on the availability and service quality of NSs are not addressed in related work.

[9] proposes an architecture to support the high availability of NSs in the NFV environment. This architecture detects (complete) VNF failures and determines whether restart or failover recovery should be used to meet the availability requirements. [22] proposes a method to determine the availability and reliability of NSs with partial, complete, or zero host-sharing strategies for the VNFs. Authors of [23] provide a VNF placement method to satisfy the reliability requirement of the NS and reduce the resource cost. [24] determines if a backup instance is needed for each active VNF instance of the NS to meet the availability requirements. [25] proposes an approach to determine the optimal number of standby instances for the active instances of each VNF of the NS to meet the availability requirements. The authors in [26] propose a solution for VNF placement that guarantees the required availability of the NS while minimizing the resource cost. In this solution, an active instance of a VNF shares the host with a standby instance of a different VNF. [7] discusses the factors that can affect the availability of NSs and proposes an approach to determine the end-to-end NS availability based on these factors. This work takes into account the topology of the NS, the availability of resources in the infrastructure, and the different failure modes of VNFs. All these solutions address NS availability or reliability, and they only consider complete VNF outages in their approaches.

Existing works mainly consider VNFs as a monolithic applications. VNFC-level failures are addressed only in a few related works [14, 27, 28, 29, 30]. However, the goal of these works is not to tackle service degradation. [14] proposes methods to calculate the availability and reliability of VNF instances based on the availability and reliability of their VNFC instances and the underlying infrastructure. [27] proposes an architecture to monitor containerized VNFs failure (i.e., pods failure) and recover the pod after a failure. Although [27] does not explicitly mention VNFC-level monitoring, a pod can be mapped to a VNFC in the NFV specifications, and their solution can be considered for VNFCs. In [28], a monitoring driver is proposed for OpenStack (an implementation of NFV architecture) to detect failures at the VNFC level. [29] uses an availability management framework in the NFV environment to manage the fault tolerance of VNFs at the VNFC level. [30] proposes a framework for Kubernetes to reserve resources for pods in order to meet the availability requirements.

VI. Conclusion

Service degradation happens more often than service outage, and may lead to service outage. Service outage is an extreme case of service degradation. In addition to meeting the required service quality, reacting to service degradation can certainly avoid service outages.

In this paper, we showed that in case of partial failures (i.e., service degradation), failover can impose a service outage and may interrupt the existing workload. We proposed a method to determine the redundancy of VNFs to meet the service quality requirements of NSs in addition to the availability requirements. We proposed a framework that includes runtime adjustment operations. These operations keep the degraded VNF instances active while they restore the processing capacity to meet the expected service quality. Our proposed framework also includes an architectural solution to monitor the internal components of VNFs and perform the runtime adjustment operations. This framework complies with the ETSI NFV specifications. As future work, we are planning to experiment with the method and the framework in a realistic testbed.

ACKNOWLEDGMENT

This work has been partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

[1] ETSI ISG NFV, "ETSI GS NFV 002 V1.2.1: Architectural Framework," December 2014. https://docbox.etsi.org/isg/nfv/open/Publications_pdf/Specs-Reports/NFV%20002v1.2.1%20-%20GS%20-%20NFV%20Architectural%20Framework.pdf.

[2] ETSI ISG NFV, "ETSI GS NFV-IFA 014 V4.3.1: Network Service Templates Specification," June 2022. https://docbox.etsi.org/isg/nfv/open/Publications_pdf/Specs-Reports/NFV-IFA%20014v4.3.1%20-%20GS%20-%20Network%20Service%20Templates%20Spec.pdf.

[3] ETSI ISG NFV, "ETSI GS NFV-IFA 011 V4.3.1: VNF Descriptor and Packaging Specification," June 2022. https://docbox.etsi.org/isg/nfv/open/Publications_pdf/Specs-Reports/NFV-IFA%20011v4.3.1%20-%20GS%20-%20VNF%20Packaging%20Spec.pdf.

[4] W. H. Von Alven, Reliability Engineering, Englewood Cliffs, N.J.: Prentice-Hall, 1964.

[5] S. Azadiabad, F. Khendek and M. Toeroe, "Availability and Service Disruption of Network Services: From High-level Requirements to Low-level Configuration Constraints," *Computer Standards & Interfaces,* vol. 80, 2022.

[6] J. Fan, et al., "A Framework for Provisioning Availability of NFV in Data Center Networks," *IEEE Journal on Selected Areas in Communications,* vol. 36, no. 10, 2018.

[7] B. Tola, G. Nencioni and B. E. Helvik, "Network-Aware Availability Modeling of an End-to-End NFV-Enabled Service," *IEEE Transactions on Network and Service Management,* vol. 16, no. 4, 2019.

[8] Y. Xu and V. P. Kafle, "An Availability-Enhanced Service Function Chain Placement Scheme in Network Function Virtualization," *Sensor and Actuator Networks,* vol. 8, no. 2, 2019.

[9] H. Yang and Y. Kim, "Design and Implementation of High-Availability Architecture for IoT-Cloud Services," *Sensors,* vol. 19, no. 15, 2019.

[10] N.-T. Dinh and Y. Kim, "An Efficient Availability Guaranteed Deployment Scheme for IoT Service Chains over Fog-Core Cloud Networks," *Sensors,* vol. 18, no. 11, 2018.

[11] D. Li, P. Hong, K. Xue and J. Pei, "Availability Aware VNF Deployment in Datacenter Through Shared Redundancy and Multi-Tenancy," *IEEE Transactions on Network and Service Management,* vol. 16, no. 4, 2019.

[12] T. Critchley, High Availability IT Services, CRC Press, 2015.

[13] E. Bauer, R. Adams and D. Eustace, Beyond Redundancy: How Geographic Redundancy Can Improve Service Availability and Reliability of Computer-based Systems, John Wiley & Sons, 2012.

[14] S. Azadiabad, F. Khendek and M. Toeroe, "Availability and Failure Rate of VNF Instances: Impacting Parameters and Calculation Methods," in *13th Int. Conference on Network of the Future (NoF)*, Ghent, Belgium, 2022.

[15] ETSI ISG NFV, "ETSI GR NFV-MAN 001 V1.2.1: Report on Management and Orchestration Framework," December 2021.https://www.etsi.org/deliver/etsi_gr/NFV-MAN/001_099/001/01.02.01_60/gr_NFV-MAN001v010201p.pdf.

[16] AT&T, "VNF Guidelines for Network Cloud and OpenECOMP," February 2017. https://wiki.onap.org/download/attachments/1015849/VNF%20Guidelines%20for%20Network%20Cloud%20and%20OpenECOMP.pdf?api=v2.

[17] J. Sathyan, Fundamentals of EMS, NMS and OSS/BSS, CRC Press, 2016.

[18] J. Fan, M. Jiang and C. Qiao, "Carrier-grade Availability-aware Mapping of Service Function Chains with On-site Backups," in *IEEE/ACM IWQoS*, Spain, 2017.

[19] Y. Harchol, D. Hay and T. Orenstein, "FTvNF: Fault Tolerant Virtual Network Functions," in *Symposium on Architectures for Networking and Communications Systems*, Ithaca, New York, 2018.

[20] R. Kang, F. He and E. Oki, "Fault-tolerant Resource Allocation Model for Service Function Chains with Joint Diversity and Redundancy," *Computer Networks,* vol. 217, no. 9, 2022.

[21] G. Venâncio and E. P. Duarte Jr., "NHAM: An NFV High Availability Architecture for Building Fault-Tolerant Stateful Virtual Functions and Services," *11th Latin-American Symposium on Dependable Computing*, 2022.

[22] P. Mandal, "Comparison of Placement Variants of Virtual Network Functions From Availability and Reliability Perspective," *IEEE Transactions on Network and Service Management,* vol. 19, no. 2, 2022.

[23] M. Karimzadeh-Farshbafan, et al., "A Dynamic Reliability-Aware Service Placement for Network Function Virtualization (NFV)," *IEEE Journal on Selected Areas in Communications,* vol. 38, no. 2, 2019.

[24] Y. Zhang, F. He and E. Oki, "Availability-Aware Service Chain Provisioning with Sub-chain-enabled Coordinated Protection," in *IFIP/IEEE Int. Symposium on Integrated Network Management (IM)*, Bordeaux, France, 2021.

[25] S. Sharma, A. Engelmann, A. Jukan and A. Gumaste, "VNF Availability and SFC Sizing Model for Service Provider Networks," *IEEE Access,* vol. 8, 2020.

[26] M. Niu, et al., "HARS: A High-Available and Resource-Saving Service Function Chain Placement Approach in Data Center Networks," *IEEE Transactions on Network and Service Management,* vol. 19, no. 2, 2022.

[27] M. Verma, et al., "A Reliability Assurance Framework for Cloud-Native Telco Workloads," in *COMSNETS*, Bangalore, India, 2023.

[28] H. Yang, B. Mutichiro and Y. Kim, "Implementation of VNFC Monitoring Driver in the NFV Architecture," in Proceedings of *ICTC*, Jeju, Korea, 2017.

[29] P. C. Rangarajan, F. Khendek and M. Toeroe, "Managing the Availability of VNFs with the Availability Management Framework," in *13th Int. Conf. on Network and Service Management (CNSM)*, Tokyo, Japan, 2017.

[30] Z. Huang, N. Samaan and A. Karmouch, "A Novel Resource Reliability-Aware Infrastructure Manager for Containerized Network Functions," in *IEEE Int. Conf. on Communications*, Montreal, QC, Canada, 2021.