Observability: The Missing Piece of Management in NFV-based Network Environments

Guilherme Werneck de Oliveira
guilherme.oliveira@ifpr.edu.br
Federal Institute of Paraná
Pinhais, Paraná, Brazil

Elias P. Duarte Jr., Vinicius Fulber-Garcia
{elias,vinicius}@inf.ufpr.br
Federal University of Paraná
Curitiba, Paraná, Brazil

Abstract

Networks are dynamic entities, as the demands on their resources, functions, and services vary over time, leading to continuous changes in their state. In recent years, the Network Function Virtualization (NFV) paradigm has emerged as a practical approach to address this dynamic nature, offering high flexibility, elasticity, and mobility for managing network functions and services. To fully leverage this flexibility, it is essential to observe the network environment by monitoring metrics that reflect its state, enabling the inference of complex indicators and trends. Despite its importance, observability in NFV-based networks remains underexplored in the literature. Existing works often lack discussions that explicitly integrate observability concepts into the NFV reference architecture's working domains and operational elements, as well as analyses of the practical implications of enforcing them. This paper presents an investigation of the application of observability in the context of NFV, linking the fundamentals of observability with the latest NFV architectures and technologies. It also demonstrates, through a case study on detecting network state changes caused by malicious activity, the impact of adopting different observability measurement models. Our results show that distinct measurement models present varying overheads on network traffic and network function instances, underscoring the importance of a well-planned integration of observability in NFV-based networks.

CCS Concepts

• **Networks** → **Network measurement**; **Network management**; **Network monitoring**;

Keywords

NFV, Observability, Metrics, Measurements, Monitoring, Management

1 Introduction

Computer networks have become increasingly complex, requiring the implementation and deployment of innovative and sophisticated functions and services to support them. In this context, adaptive networks, which can be reconfigured in response to changes in network state, have emerged as a critical technology [15, 53]. To address the challenges of flexibility, elasticity, and mobility in adaptive networks, softwarization paradigms such as Software-Defined Networking (SDN) and Network Function Virtualization (NFV) have been extensively explored by both academia and industry [20]. In particular, the NFV paradigm has received special attention due to its ability to deploy Virtualized Network Functions (VNF) and orchestrate them through Service Function Chains (SFC) in a highly adaptable manner [13].

Although NFV was developed to provide high flexibility for network environments, leveraging all this potential is not trivial. First of all, it requires effective management of virtual functions and services [16, 18]. Network management must effectively provide a deep understanding of the network state. This challenge becomes even more complex when the network is regarded as a dynamic entity whose state changes through the interactions of its connected nodes [12, 34]. Therefore, we argue that NFV can significantly benefit from observability, which has been successfully applied to support strategic management and decision-making across multiple scenarios [33, 46].

Observability can be defined as the extent to which a system's state can be traced and understood by an observer entity [35]. To simplify observability in NFV, it can be seen as the monitoring of metrics, through measurements that characterize network functions, services, and the overall network state. However, although a vast literature exists on solutions to manage NFV-based networks, keeping them operational and optimized [14, 21, 41, 42], there are few works on metrics and measurement models to provide such solutions with appropriate data aligned to their objectives. Furthermore, there are few insights into where these metrics should be defined and how they can be made available for measurement by observer entities within the NFV reference architecture. At last, every operation in the network incurs a cost, including observing virtualized functions, services, and the network itself. Thus, discussing the costs and consequences of adopting different observability measurement models represents another gap in the literature.

Accordingly, this article discusses observability as a key requirement for managing NFV-based network environments. However, implementing observability requires support from multiple operational elements defined in the NFV reference architecture, which may act as observed entities, observer entities, or both depending on the case. Moreover, adopting different observability measurement models can generate heterogeneous impacts on defining the network state and shape decision-making in management. To discuss these impacts, we conducted an empirical evaluation simulating a scenario in which the network faces malicious traffic, with observability enabling the detection of an ongoing attack. The results revealed distinct operational characteristics and collateral effects for each tested measurement model, indicating that choosing one model over another may lead to overhead on the network traffic and network function instances.

The remainder of this work is organized as follows. Section 2 presents the fundamentals of observability. Section 3 identifies observability enablers within the NFV reference architecture and describes them as observed or observer entities. Section 4 discusses metrics and measurement models, highlighting implementation

opportunities based on state-of-the-art NFV architectures and technologies. Section 5 demonstrates the impact of adopting different observability measurement models in NFV-based networks through a case study. Finally, Section 6 concludes the paper.

2 Observability in a Nutshell

The concept of observability was introduced in 1960 within the framework of control theory to formalize a mathematical approach for inferring a system's internal state from its external outputs [30]. Initially, observability was applied to the study of formal methods and linear algebra, particularly in the analysis of dynamic systems in mechanical and automation engineering.

In networking, observability enables an agent (**observer entity**) to compute metrics and generate indicators from network functions, services, and links (**observed entities**). These metrics and indicators can then be analyzed by management and provisioning elements to assess the evolution of the network state and to support decision-making processes, with the goal of, for example, meeting performance requirements, fulfilling service level agreements, and improving both Quality of Service (QoS) and Quality of Experience (QoE) [10, 38].

Technically, observability can be analyzed from two primary perspectives: **metrics** and **measurements**. Metrics are standardized definitions of values computed in an experiment designed for specific performance-related purposes [25]. Examples include the state of an operational element in terms of CPU and RAM usage, as well as the state of links in terms of latency, throughput, etc. By analyzing these metrics, it is possible to assess the functionality of each operational element and its connections, thereby deriving the network's abstract state. Measurements, on the other hand, refer to the process of executing a series of operations to get the value of a metric within a specific scenario and time frame.

It is important to note that observability extends beyond mere monitoring by providing a holistic view of the network. Observability defines which metrics should be measured and how they should be collected, integrating multiple data sources to provide complex indicators and actionable insights, such as proactive issue detection and resolution. In contrast, monitoring consists of evaluating the state of a network and its functions and services by comparing them against an expected state. Observability does not rely on predefined expected states; instead, it treats the network as a dynamic entity and aims to deeply analyze how its states evolve and what valuable information and actions can be derived from these dynamics. It includes assessing the potential interference introduced by the observation process itself and adapting it to obtain the most appropriate information from the network, balancing freshness, accuracy, and costs.

3 Observability in NFV: A Map

The NFV reference architecture, proposed by the European Telecommunications Standards Institute (ETSI) [8], comprises three working domains: Virtualized Infrastructure (VI), Management and Orchestration (MANO), and Virtualized Network Functions (VNF). The VI domain is responsible for managing the underlying infrastructure that provides the computational resources required to support the deployment and execution of virtualized functions and services.

The MANO domain provides management and orchestration of the NFV environment through three main operational elements: the Virtualized Infrastructure Manager (VIM), which communicates with the VI domain to allocate, release, and monitor computational resources; the VNF Manager (VNFM), which interacts with elements in the VNF domain to manage the lifecycle of individual virtualized function instances; and the NFV Orchestrator (NFVO), which is responsible for managing the lifecycle of complete network services, typically composed of multiple interconnected functions.

Finally, the VNF domain consists of two operational elements: the Virtualized Network Functions (VNFs) themselves, which represent the primary traffic-processing elements by executing network functions over the traffic; and the Element Management System (EMS), a management element that interacts directly with VNF instances, abstracting their complexity and heterogeneity from the MANO.

Some works in the literature address specific aspects of defining which working domains or operational elements of the NFV reference architecture assume particular observability roles in a network. In [4], the authors propose a mechanism for trust assessment in NFV-based networks. Their solution integrates a trust monitor into the MANO working domain, in accordance with ETSI guidelines on NFV trust and security. The architectural discussion underpinning this decision reinforces MANO's role as the central actor in observability, monitoring, and decision-making within the NFV reference architecture, as it manages and orchestrates both the VNF and VI domains (observed entities). Accordingly, MANO and its operational elements can be considered top-level observer entities within the NFV reference architecture.

In [29], the concept of a target entity for NFV observability is introduced, which refers to the elements that must be considered to compute a given metric. For example, when measuring the CPU load, a specific VNF instance serves as both the target and the observed entity. However, this model highlights an important point: computing a single metric may involve multiple target entities. For instance, when measuring the Round-Trip Time (RTT) from a VNF X to a VNF Y, the observed entity is the VNF X (since RTT is not necessarily symmetric). In contrast, both VNF instances (X and Y) are required to compute the metric and are therefore considered target entities. Thus, based on this work, VNF instances are assumed to be the primary observed entities in an NFV environment.

Enabling observer entities within MANO to request measurements and access information from observed entities requires the establishment of appropriate communication interfaces. The ETSI NFV reference architecture defines several interfaces that can be used for this purpose, including the Nf-Vi interface between MANO (VIM) and the VI domain; the Ve-Vnfm-vnf interface between MANO (VNFM) and VNF instances; and the Ve-Vnfm-em interface between MANO and EMS instances. These interfaces, along with their associated communication protocols, must be implemented within each working domain and its operational elements. However, in practice, there is limited standardization regarding the set of supported operations and the technologies used to implement them.

The literature, however, provides architectures and frameworks to enable holistic communication among different implementations

of interfaces provided by observer and observed entities. For instance, the framework presented in [22] enables different orchestrators to communicate to deploy and manage the network functions of a single service, including an abstract implementation of the Nf-Vi interface. Furthermore, the architecture for implementing VNF platforms described in [17] introduces an internal module, referred to as a management agent, which is specifically designed to establish the Ve-Vnfm-vnf interface.

In this sense, the work in [11] presents an architecture for the EMS module. According to the proposed internal organization, the EMS acts as an intermediary via the Ve-Vnfm-em interface, abstracting the heterogeneity involved in accessing VNF instances from the VNFM. In this architecture, the EMS is responsible for directly interacting with VNF instances, including observing them, and thus acts as an observer entity. At the same time, the EMS acts as an observed entity with respect to the VNFM (MANO), as it receives and responds to measurement requests for VNF instances. To support this, the architecture defines three main modules: the VNF subsystem, which establishes communication between the EMS and the VNF instances; the monitoring subsystem, which enables operators to define proactive monitoring routines for VNF instances; and the access subsystem, which implements the Ve-Vnfm-em interface.

By leveraging the observability enablers provided by the NFV paradigm, it is possible to define multiple metrics and heterogeneous measurement models that trigger processes to compute and communicate them. Naturally, an NFV-based network may present a diversity of requirements, making specific metrics and measurement models more suitable for certain scenarios.

4 Metrics and Measurements Tailored to the NFV Paradigm

Observability has the potential to play a critical role in managing NFV-based networks, providing the visibility needed to act on virtualized elements in real time. In modern networks, where virtualized functions and services are deployed on demand to meet varying service requirements, observability enables operators to continuously assess their health, performance, and resource utilization. Furthermore, NFV technology has been also used to deploy arbitrary functions and services in the network [51], multiple services have been built according with this paradigm [9, 44, 49, 50]. Real-time observability enables management and orchestration platforms [23, 43], for example, to automatically scale VNF instances up or down, in or out, based on current demand or to migrate them across different virtualization infrastructures.

However, NFV-based networks are highly dependent on the underlying virtual infrastructure, and their performance can be affected by multiple factors, including resource contention, the state of network links, and the way metrics are measured. Consequently, different metrics and measurement models must be defined and analyzed from the perspective of the NFV paradigm in order to identify the most suitable observability approach for each specific network context.

4.1 Observing NFV-based Networks: Metrics

While a myriad of metrics are commonly used in daily network operations, their definitions are far from straightforward. The design of metrics must ensure they are meaningful, measurable, and interoperable across diverse network environments, enabling consistent measurement and comparison. RFC 6390 [25] outlines a process for designing metrics, which includes the following steps: *i)* problem statement, clearly defining the problem that the new metric aims to address; *ii)* metric definition, creating a precise and unambiguous definition of the metric, including what is being measured and the units of measurement; *iii)* method of measurement, specifying how the metric will be measured, including any relevant algorithms or methodologies; and *iv)* reporting and interpretation, providing guidelines for how the results should be reported and interpreted, ensuring clarity and consistency.

In the context of virtualized network services, the ETSI defines a comprehensive set of metrics related to virtualized network functions and the management of virtualized infrastructures [6, 7]. These metrics address relevant aspects of virtualization-enabled networks, including efficiency (*e.g.*, packet delay, jitter, throughput of delivered packets, interruption, provisioning, and configuration latency), efficacy (*e.g.*, packet loss rate, clock error, placement, and compliance policy), and reliability (*e.g.*, provisioning reliability, broken connections, premature releases, and failed release rates).

The Internet Engineering Task Force (IETF), in turn, has expanded the scope of the Benchmarking Methodology Working Group (BMWG) to include methods for evaluating virtualized network functions and their supporting technologies, such as SDN controllers and virtual switches. The BMWG has developed a specific methodology for benchmarking virtual network performance [26], which includes key metrics such as throughput, packet loss rate, latency, and CPU and RAM consumption.

Other metrics related to NFV performance are outlined in RFC 8172 [27], including the time required to deploy and migrate virtual network functions. This RFC also offers insights into the impact of measurement policies on VNF performance, highlighting that different measurement policies can have heterogeneous effects on VNF performance when processing network traffic.

The academic literature also presents works that highlight metrics related to NFV. In [31], for instance, the authors present performance comparison metrics for SDN and NFV controllers. NFV metrics include vCPU and vMemory (compute-associated), latency and throughput (communication-associated), I/O rate, and VNF recovery time (storage-associated). The work of [19] illustrates the applicability of metrics specific to NFV fault tolerance, such as packet loss, average packet throughput, congestion interconnection, among others. A deeper study of the impacts of environmental metrics on Machine Learning (ML) models used for management is presented in [32] and [5]. The work [32] presents a supervised learning study to predict VNF deployment decisions under changing network conditions. In [5], the authors focus on VNF placement to mitigate DDoS attacks in industrial IoT systems. The observed metrics included environment deployment time, CPU and RAM consumption, latency, throughput, time to attack detection and mitigation, response time, and ML model accuracy.

Table 1 presents a classification of NFV metrics based on the works presented in this section. Three categories of metrics are considered: *i)* metrics for diagnosing the state of virtualized functions, which depend on the implementation of the network function; *ii)* metrics related to the state of virtualized nodes (v-nodes); and *iii)* metrics concerning the overall network state.

Table 1: Categories of metrics for NFV environments.

Metric		
VNF	**V-Node**	**Network**
Setup and response time	CPU, RAM and disk/swap consumption	Latency
Number of rules executed/not-executed	Disk and network I/O	Throughput
Policy enforcement time	Deployment, migration and recovery time	Jitter
Machine learning models metrics	Scalability rate	Delivered/discarded packets
Packet processing time		
Packet and data receiving/transmitting rate		

Finally, it is important to note that, in addition to conventional metrics, other resources can be instrumental for observing the state of applications and services in NFV provider environments. These resources include [35]: *i)* logs, which provide a detailed, timestamped, and immutable record of events; and *ii)* traces, which capture end-to-end temporal events related to specific actions in distributed systems, such as the management of virtualized network services. Notably, tracing is also crucial for addressing the challenge of explainability (understanding how algorithms build their results and make decisions) in dynamic networks.

4.2 Observing NFV-based Networks: Measurements

After designing and implementing metrics in an NFV-based network environment, the next step toward achieving observability is to measure them across network elements and links. Thereby, these measurements must be conducted in a way that does not overload the network or its elements, ensuring that regular traffic processing remains unaffected. For a considerable time, network operators have relied on protocols and technologies such as the Simple Network Management Protocol (SNMP), Command-Line Interface (CLI), or Syslog for network monitoring. However, dynamic network operations require data to be measured from multiple sources with varying granularity and frequencies, a need that legacy protocols and technologies often struggle to meet. Even with advances such as NETCONF [24], gRPC collectors [39], perfSONAR [40], and In-band Network Telemetry (INT) [37], there remain limitations in collecting metrics for v-nodes and VNF instances.

Upon analyzing the literature on observability, it is possible to identify the primary characteristics of measurement models suitable for network environments. Authors in [52] propose an abstract framework for implementing general entities as observable sources of knowledge and services, introducing two models regarding the **dissemination** of measured metrics: an observer entity can get information from an observed entity using a reactive or proactive approach. When an observer entity sends a state request, and the observed entity processes it as soon as it arrives, the dissemination model is called reactive. However, the observed entity may or may not respond immediately. For example, if a response has the same value as the previous one, the observed entity may decide to spare a retransmission. This technique is called ”conditioned responses´´, where replies are sent only when specific rules are met in the observed entity or, generally, when a relevant event occurs.

Unlike the reactive model, the proactive dissemination involves the observed entity notifying the observer entity, which can be implemented using a publish/subscribe scheme [3]. In this case, the observer entity sends a request to register its interest in receiving notifications whenever the value of a specific metric or state from the observed entity changes. This subscription can be carried out directly with the observed party or through an intermediary, such as a message broker. Moreover, to manage the number of requests and responses generated in the network environment, the previously mentioned technique of conditioned responses can also be integrated into this model.

Furthermore, the **encapsulation** of measured metrics for transmission from the observed entity to the observer also defines two models: active and passive. According to [47] and [1], the passive model does not introduce additional traffic into the network. In contrast, the active model involves the transmission of dedicated packets to provide information about network entities, such as links and devices; as a result, it is often intrusive and may disrupt ongoing services due to the additional traffic it generates.

It is important to note that passive and active measurement models can be applied both to reactive dissemination, through reactive responses [52] or query-based polling [28], and proactive dissemination, through proactive reporting [52] or subscription-based pushing [28]. The classification ultimately depends on how the communication of the measurement process is structured. For instance, using dedicated packets to transfer measurement results characterizes an active model, whereas techniques such as piggybacking measurement data onto existing traffic correspond to a passive model.

Another characteristic to consider regarding measurements is the **acquisition** mode, which determines whether measurements are triggered continuously or on demand. Continuous acquisition involves constant monitoring of environmental metrics over time [45], enabling real-time visibility and trend analysis. The continuous model enables early identification of patterns, anomalies, and performance degradation, making it valuable for proactive network management and long-term optimization. In contrast, on-demand acquisitions are triggered by an entity when specific information about the network’s current state is required, providing flexibility and targeted insights without constantly collecting data.

Moreover, the **interval** between measurements is also relevant, since it may affect not only the network but also the observed entities, such as VNF instances [25, 27]. Short sampling intervals offer valuable insights into an entity's performance. However, they can result in an overwhelming number of measurements and an excessive processing load concurrent with the primary function of the observed entity. On the other hand, longer sampling intervals reduce the volume of data collected, which may be insufficient to accurately assess the observed entity performance or the network state, as the metrics being measured might fluctuate significantly over time. In some cases, the percentile-based statistical technique may be used to disregard potential outliers. Table 2 summarizes the described measurement characteristics and models presented in this section.

Table 2: Observability measurement: characteristics and models.

Characteristic	Models	Description
Dissemination	Reactive Proactive	Refers to the strategy by which measurements are requested and delivered from observed to observer entities, either reactively upon request or proactively based on conditions or events.
Encapsulation	Active Passive	Refers to the strategy used to transfer measurements from observed to observer entities, either actively through dedicated packets or passively by, for example, piggybacking on existing traffic.
Acquisition	Continuous On-Demand	Refers to the strategy by which measurements are triggered to define or update metric values, either continuously through constant and regular monitoring or on demand in response to occasional requests.
Interval	Short Long	Refers to the sampling interval strategy that defines how frequently metrics are measured, either short to improve information freshness or long to reduce observability overhead.

From a technical implementation perspective in NFV, there is a strong synergy between observability measurement models and state-of-the-art architectures designed for the operational elements of the paradigm. For example, the VNF platform architecture presented in [17] is prepared to enable reactive dissemination in an active measurement mode through its management agent modules, which create the interface to communicate with EMS and VNFM (the observer entities), and extended agents, which provide specific metrics related to the network function executed within the VNF platform. Notably, there is no restriction on the measurement process for either the acquisition or the interval models. Consequently, the observer entity is fully responsible for determining both the measurement frequency and the strategy for executing requests. Partially or entirely, the VNF platforms ClickOS [36], Click-on-OSv [2], and COVEN [17] follow this architectural approach, supporting the measurement models as described.

The EMS architecture presented in [11], in turn, can leverage most measurement models, as it operates as both an observed and an observer entity. Through its internal modules, the EMS supports reactive measurements via the ETSI Ve-Vnfm-em standard interface implemented in the access subsystem, as well as proactive measurements through a subscription mechanism associated with the monitors of the monitoring subsystem. In this sense, the access subsystem enables top-level observer entities to submit on-demand measurement requests. In contrast, the monitoring subsystem allows the EMS to measure metrics from VNF instances either continuously or on demand, at both short and long intervals. From an architectural perspective, there is no explicit support for the EMS operating in a passive mode with respect to packet encapsulation, with the active model being the most straightforward alternative, as implemented in the HoLMES EMS [11]. Nevertheless, a passive model could also be realized in EMS-based solutions, for example, by piggybacking measurement results onto packets used to confirm the execution of lifecycle operations in VNF instances requested by the VNFM or by operations and business support systems.

Furthermore, top-level observer entities, such as the VNFM, must be capable of triggering measurements and receiving and processing their results, regardless of the measurement models adopted. The ETSI specification provides only a high-level definition of the VNFM and does not define an internal architecture for implementing this operational element within the MANO domain. Nevertheless, the literature offers insights into possible implementations, including the definition of internal modules that enable refined observability operations through the VNFM. For example, the authors in [48] propose a framework to address technical gaps in the VNFM that are not covered by its specification documents. They introduce a Monitoring Module composed of two agents: one responsible for monitoring VNF instances, either directly via the Ve-Vnfm-vnf interface or indirectly through the EMS via the Ve-Vnfm-em interface, and another responsible for monitoring the underlying virtualization infrastructure through the Nf-Vi interface. These agents can execute different models for acquisition and interval, and play a central role in lifecycle management decision-making in an NFV environment.

Another important factor is that observability can have significant implications for both infrastructure security and user privacy. On the one hand, observability enhances threat detection, incident response, and forensic analysis by enabling fine-grained monitoring of anomalous behavior across distributed systems, such as lateral movement within virtualized clusters or suspicious API calls in workloads. Thus automatic and autonomous systems can leverage real-time observations to detect misconfigurations, privilege escalation attempts, or data exfiltration patterns, thereby strengthening resilience and compliance. On the other hand, the same data collection mechanisms that improve visibility may also expand the attack surface and introduce privacy risks. Extensive logging of user interactions, session identifiers, IP addresses, or payload data, particularly without adequate access controls, encryption, or data minimization practices, can inadvertently expose sensitive

personal information. In addition, centralized observability platforms may become high-value targets, as aggregated observations can reveal system architecture details and user behavior patterns. Consequently, while observability is essential for securing modern virtualized infrastructures, it requires rigorous governance frameworks, strict access policies, and privacy-by-design principles to prevent the transformation of monitoring mechanisms into vectors of surveillance or compromise.

Regardless of whether measurement is proactive or reactive, passive or active, continuous or on-demand, and whether it is performed over short or long intervals, measurement techniques may consume significant network resources and generate redundant or sensitive data. Therefore, the enforcement of observability must carefully consider the trade-offs associated with different metrics and measurement models, taking into account network requirements and capacity constraints. This approach enables the tracing of functions, services, and overall network state with an appropriate level of detail, aligned with the objectives of users and operators.

5 The Impacts of Observability: A Case Study

The level of insight derived from metric measurements is an important factor in planning observability in NFV-based networks, since VNF instances are highly dependent on the underlying virtual infrastructure and their performance can be influenced by several factors, such as resource contention, the state of network links, and even the way in which measurements are executed. To demonstrate the impacts of adopting different observability measurement models, the case study illustrated in Figure 1 is presented. The experimental setup consists of a simulated NFV network implemented with Docker containers, running applications written in C. We consider three main containers in the experiment:

- **VNF**: A container that runs an application whose primary functionality is to receive packets from a client and analyze their payloads, searching for malicious signatures in the content, thus simulating a virtual Deep Packet Inspection (vDPI) function. For each detected signature, an attack counter is incremented; this counter is reset upon communication of the measured value resulting from a requested or executed measurement. The VNF provides two types of interfaces. The first is a server-side network socket that waits for requests and responds with the counter value (**reactive dissemination model**). The second is a client-side network socket that proactively forwards signature counter values to the observer application without explicit requests (**proactive dissemination model**);
- **Client**: A container running an application that sends a large volume of predefined packets, which may or may not contain attack signatures identified by the vDPI VNF;
- **Observer**: A container with an application whose purpose is to request metric measurements from the vDPI VNF (reactive dissemination) or to receive metric values automatically (proactive dissemination) from the VNF instance.

In this experiment, all metric measurement requests are responded to using dedicated packets; thus, observability measurements are implemented using an **active encapsulation model**. Moreover, all measurements adopt the same **acquisition model (continuous)** and **interval model (adaptable)**, according to the following policy: after each measurement, the VNF instance (in the proactive case) or the observer (in the reactive case) increases the measurement interval by two (2) seconds when no malicious signatures are identified, or decreases it by two (2) seconds when one or more malicious signatures are detected. The interval is bounded by a minimum of zero (0) seconds and a maximum of eleven (11) seconds. Furthermore, in the proactive dissemination model, the measurement agent running within the VNF platform may not initiate communication with the observer at the end of a measurement interval if no attack activity has been detected during that period (*i.e.*, when the signature counter equals zero).

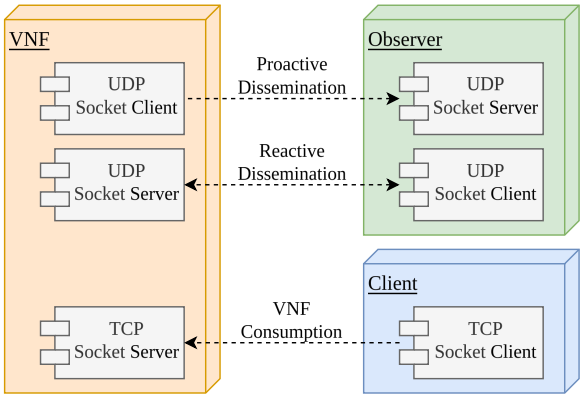

Figure 1: Experimental setup for the case study.

Furthermore, additional metric values are transmitted along with the signature counter with the measurement results, including CPU and RAM utilization percentages, as well as the number of packets and the volume of data received and transmitted per second (rxpck/s, txpck/s, rxkB/s, and txkB/s). These metrics are obtained using the `sysstat` tool[1].

After establishing the experimental setup, we conducted tests using three sets of messages, each containing 10%, 40%, and 90% of the packets with attack signatures (Cases 1, 2, and 3), for a total of 500 million packets per case. To simulate a burst attack, the packets containing the respective signatures were distributed across three equal-sized traffic windows. Each case was executed 30 times to assess the impact of adopting proactive or reactive measurement models on network load and the detection of malicious activity. The developed applications, scripts, and raw data and results are available in a public GitHub repository[2].

The number of packets exchanged between the vDPI VNF instance and the observer was determined for the three testing scenarios. As shown in Figure 2, the reactive measurement model generated approximately 2.25 times more observations than the proactive measurement model, with this difference decreasing as the number of identified malicious packets increased. This behavior occurs because the reactive model always requires a request from the observer to the VNF to trigger a measurement and, subsequently, to update the measurement interval according to the defined policy.

[1] https://github.com/sysstat/sysstat
[2] https://github.com/werneckg/vnf-observability

Since there are no synchronized clocks or fault-detection mechanisms in the system, each request issued by the observer must receive a response, even when the signature counter is zero. In contrast, under the proactive model, the VNF does not communicate with the observer when the attack counter is zero; it only adapts its measurement interval. This distinction is also evident in Figure 3, which presents the cumulative number of requests and responses generated by each approach based on their respective observations.

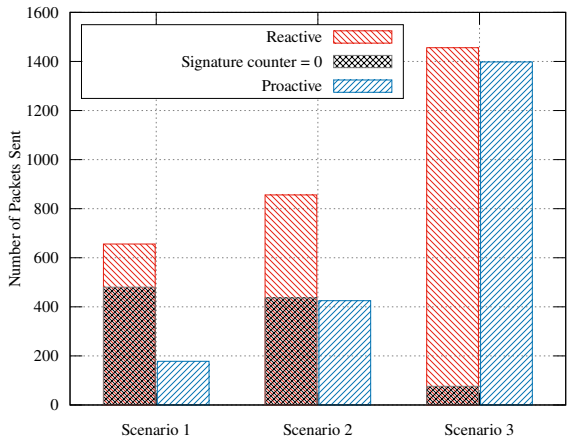

Figure 2: Comparison between reactive and proactive dissemination models according to the number of metric measurements sent from the VNF to the observer.

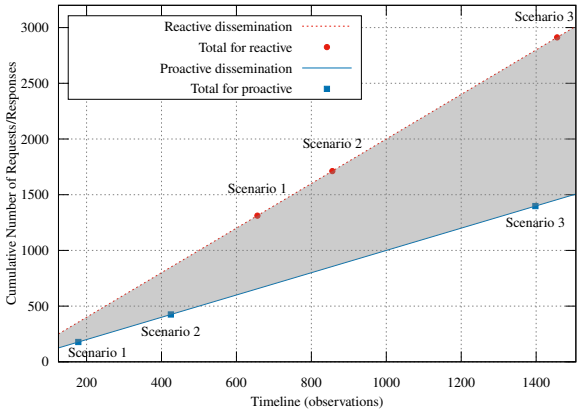

Figure 3: Cumulative number of measurement results sent from the VNF instance to the observer along the three scenarios.

Regarding the performance metrics measured and sent by the VNF, along with the signature counter, a clear difference is observed in the average number of packets per second received and transmitted under the reactive and proactive dissemination models, as shown in Figure 4. The 90th percentile confirms this difference, also shown in Figure 4. Specifically, the proactive model consistently results in fewer received and transmitted packets. This behavior stems from two features of the proactive dissemination model implemented in this case study: it does not require network requests to trigger the measurement process, and it does not transmit measurement packets when the signature counter is zero. In contrast, the reactive model always involves request–response exchanges, resulting in a slightly higher network load.

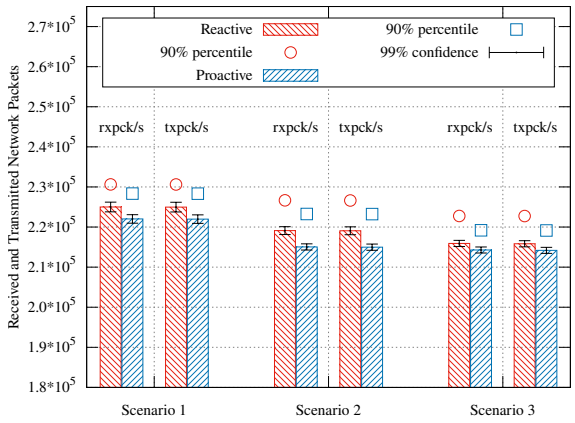

Figure 4: Received and transmitted packet rate from the VNF perspective.

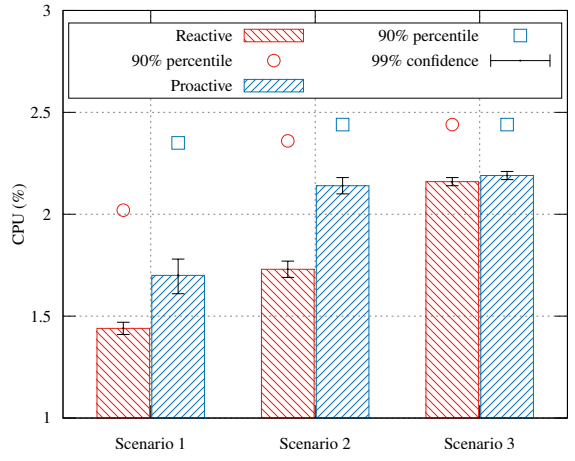

Figure 5: CPU load during the execution of testing scenarios in the VNF instance.

However, to reduce network overhead caused by the reactive dissemination model, proactive dissemination requires implementing a measuring agent within the VNF platform, which consequently consumes computational resources to configure, trigger, execute, and evaluate measurements. As a result, a slightly higher CPU consumption is noted when this model is enforced, as shown in Figure 5. This processing overhead increases as the number of measurement

agents and the complexity of their operations grow. Nevertheless, in the specific case study presented, the increase in CPU consumption, although prominent in percentage terms when compared to the reactive model, is not significant in absolute terms (no more than 0.5% of additional CPU load). Moreover, the difference between the dissemination models decreases as the measurement interval is reduced, a phenomenon that becomes explicit when analyzing the 90th percentile across the scenarios presented in Figure 5.

As demonstrated in this case study, the adopted measurement model can directly affect the network and the functions and services deployed on it. It is worth noting that the experiment was conducted in a controlled infrastructure and does not fully reflect the heterogeneity of demands and VNFs found in real-world environments, where multiple factors can naturally modify the nature and magnitude of these impacts. For instance, depending on a network function's attributes, a passive encapsulation model may be a suitable alternative for reducing the network load associated with reactive dissemination, potentially making it comparable to proactive dissemination. Therefore, there is no golden rule for implementing observability in NFV-based networks. Instead, the specific characteristics of the environment and the metrics of interest must be carefully considered to select the most appropriate model for each measurement characteristic, enabling robust and effective observability.

6 Conclusion

The flexibility introduced by the NFV paradigm has brought several advantages to modern dynamic network environments, making it an efficient approach for guaranteeing quality of service across different contexts. However, the unpredictability of dynamic network environments demands efficient solutions to manage them effectively. Observability, can be seen as a strategy for holistically determining the network state in a precise way. It is the cornerstone for enabling operators to make the best management decisions for the network. However, defining metrics to determine these states and the best measurement models to assess them is far from trivial.

In this paper, we broadly discussed observability in the context of the NFV paradigm. First, we identified the observability enablers of the paradigm, considering the NFV reference architecture proposed by ETSI. We then examined how observability metrics and measurement models can be applied to NFV, from both conceptual and technical perspectives. The operational elements across NFV working domains were characterized as observed or observer entities, and state-of-the-art architectures and frameworks were analyzed with respect to their support for practical observability implementations. Finally, we presented a case study demonstrating that adopting different observability measurement models can lead to distinct impacts in NFV-based network environments. In particular, the chosen dissemination models may introduce additional network overhead or increased CPU consumption at VNF instances.

Implementing observability in conjunction with NFV is inherently complex, as multiple factors must be considered, including the definition of metrics aligned with the objectives of network managers and operators, as well as measurement models that assess metric values in a balanced manner, providing timely information with minimal overhead. Nevertheless, despite this complexity, observability offers significant opportunities to manage and optimize networks by considering both the current network state and predicting future states based on historical observations. Examples of operations that can benefit from refined observability include mapping and placing virtualized services on the underlying infrastructure, preventing and mitigating network bottlenecks, and strategically migrating virtualized functions to improve quality of service. Furthermore, through observability, cost models can be designed and tailored to each type of infrastructure environment based on its behavioral characteristics, such as network traffic patterns, deployed VNFs and services, and end-user demands, enabling detailed analysis and planning for network management.

Acknowledgments

This work was developed with the support of the Federal University of Parana (COFPI/PRPI 19/2025 - 23075.058047/2025-51), the Program of Academic Excellence (PROEX) - Coordination for the Improvement of Higher Education Personnel (CAPES - AUXPE 88881.189840/2025-01) and the National Council for Scientific and Technological Development (CNPq - Project 305108/2025/5).

References

[1] Thomas M. Chen. 2001. Increasing the observability of Internet behavior. *Commuications of the ACM* 44, 1 (2001), 93–98.

[2] Leonardo da Cruz Marcuzzo, Vinicius F Garcia, Vitor Cunha, Daniel Corujo, Joao P Barraca, Rui L Aguiar, Alberto E Schaeffer-Filho, Lisandro Z Granville, and Carlos RP dos Santos. 2017. Click-on-osv: A platform for running click-based middleboxes. In *IFIP/IEEE Symposium on Integrated Network and Service Management*. IFIP/IEEE, Lisbon, Portugal, 885–886.

[3] João Paulo De Araujo, Luciana Arantes, Elias P Duarte, Luiz A Rodrigues, and Pierre Sens. 2017. A publish/subscribe system using causal broadcast over dynamically built spanning trees. In *2017 29th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. IEEE, 161–168.

[4] Marco De Benedictis and Antonio Lioy. 2019. A proposal for trust monitoring in a network functions virtualisation infrastructure. In *IEEE Conference on Network Softwarization*. IEEE, Paris, France, 1–9.

[5] Guilherme Werneck De Oliveira, Michele Nogueira, Aldri Luiz dos Santos, and Daniel Macêdo Batista. 2023. Intelligent VNF Placement to Mitigate DDoS Attacks on Industrial IoT. *IEEE Transactions on Network and Service Management* 20, 2 (2023), 1319–1331.

[6] ETSI Industry Specification Group (ISG) NFV. 2014. Network Functions Virtualisation (NFV); NFV Performance & Portability Best Practices.

[7] ETSI Industry Specification Group (ISG) NFV. 2014. Network Functions Virtualisation (NFV); Service Quality Metrics.

[8] ETSI Industry Specification Group (ISG) NFV. 2024. *Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Architectural Framework Specification*. Group Specification GS NFV 006 v4.5.1. European Telecommunications Standards Institute (ETSI).

[9] Bruno E Farias, José Flauzino, and Elias P Duarte Jr. 2025. VNF-Cache: An In-Network Key-Value Store Cache Based on Network Function Virtualization. *arXiv preprint arXiv:2512.19964* (2025).

[10] Ummay Faseeha, Hassan J Syed, Fahad Samad, Sehar Zehra, and Hamza Ahmed. 2025. Observability in Microservices: An In-Depth Exploration of Frameworks, Challenges, and Deployment Paradigms. *IEEE Access* 13 (2025), 72011–72039.

[11] Vinicius Fulber-Garcia, José Flauzino, Carlos RP Dos Santos, and Elias P Duarte. 2023. An ETSI-compliant Architecture for the Element Management System: The Key for Holistic NFV Management. In *IEEE International Conference on Network and Service Management*. IEEE, Niagara Falls, Canada, 1–9.

[12] Vinicius Fulber-Garcia, José Flauzino, Giovanni Venâncio, Alexandre Huff, and Elias P Duarte Junior. 2024. Breaking the limits: Bio-inspired sfc deployment across multiple domains, clouds and orchestrators. In *2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 1–6.

[13] Vinicius Fulber-Garcia, Alexandre Huff, Carlos R P dos Santos, and Elias P Duarte Jr. 2020. Network service topology: Formalization, taxonomy and the CUSTOM specification model. *Elsevier Computer Networks* 178 (2020), 107337.

[14] Vinicius Fulber-Garcia, Alexandre Huff, Leonardo da C Marcuzzo, Marcelo C Luizelli, Alberto E Schaeffer-Filho, Lisandro Z Granville, Carlos RP dos Santos,

and Elias P Duarte Junior. 2021. Customizable Deployment of NFV Services. *Journal of Network and Systems Management* 29, 3 (2021), 1–27.

[15] Vinicius Fulber-Garcia, Marcelo C Luizelli, Carlos R Paula dos Santos, Eduardo J Spinosa, and Elias P Duarte Jr. 2023. Customizable mapping of virtualized network services in multi-datacenter environments based on genetic metaheuristics. *Journal of Network and Systems Management* 31, 4 (2023), 71.

[16] Vinícius F Garcia, Leonardo C Marcuzzo, Giovanni V Souza, Lucas Bondan, Jéferson C Nobre, Alberto E Schaeffer-Filho, Carlos RP dos Santos, Lisandro Z Granville, and Elias P Duarte Jr. 2019. An nsh-enabled architecture for virtualized network function platforms. In *International Conference on Advanced Information Networking and Applications*. Springer, 376–387.

[17] Vinicius Fulber Garcia, Leonardo da C Marcuzzo, Alexandre Huff, Lucas Bondan, Jeferson C Nobre, Alberto Schaeffer-Filho, Carlos RP dos Santos, Lisandro Z Granville, and Elias P Duarte. 2019. On the design of a flexible architecture for virtualized network function platforms. In *IEEE Global Communications Conference*. IEEE, Big Island, Hawaii, USA, 1–6.

[18] Vinícius Fülber Garcia, Giovanni Venâncio De Souza, Elias Procopio Duarte Jr, Thales Nicolai Tavares, Leonardo Da Cruz Marcuzzo, Carlos RP Dos Santos, Muriel Figueredo Franco, Lucas Bondan, Lisandro Zambenedetti Granville, Alberto Egon Schaeffer-Filho, et al. 2020. On the design and development of emulation platforms for NFV-based infrastructures. *International Journal of Grid and Utility Computing* 11, 2 (2020), 230–242.

[19] Lav Gupta, Tara Salman, Maede Zolanvari, Aiman Erbad, and Raj Jain. 2019. Fault and performance management in multi-cloud virtual network services using AI: A tutorial and a case study. *Computer Networks* 165, C (2019), 22 pages.

[20] Mu He, Alberto Martínez Alba, Arsany Basta, Andreas Blenk, and Wolfgang Kellerer. 2019. Flexibility in softwarized networks: Classifications and research challenges. *IEEE Communications Surveys & Tutorials* 21, 3 (2019), 2600–2636.

[21] Thanh Tung Hoang, Manh Linh Pham, and Hoai Son Nguyen. 2024. Scaling and Dynamic Resource Reallocation in NFV: Challenges and Research Perspectives. *International Journal of Electrical and Computer Engineering Systems* 15, 10 (2024), 851–863.

[22] Alexandre Huff, Giovanni Venâncio, Vinícius Fulber Garcia, and Elias P Duarte. 2020. Building multi-domain service function chains based on multiple nfv orchestrators. In *IEEE Conference on Network Function Virtualization and Software Defined Networks*. IEEE, Virtual, 19–24.

[23] Alexandre Huff, Giovanni Venancio, Leonardo da C Marcuzzo, Vinicius F Garcia, Carlos RP dos Santos, and Elias P Duarte. 2018. A holistic approach to define service chains using click-on-osv on different nfv platforms. In *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.

[24] IETF. 2011. RFC 6241: Network Configuration Protocol (NETCONF). https://datatracker.ietf.org/doc/html/rfc6241

[25] IETF. 2011. RFC 6390: Guidelines for Considering New Performance Metric Development. https://datatracker.ietf.org/doc/html/rfc6390

[26] IETF. 2017. Benchmarking Methodology for Virtualization Network Performance. https://datatracker.ietf.org/doc/html/draft-huang-bmwg-virtual-network-performance-03

[27] IETF. 2017. RFC 8172: Considerations for Benchmarking Virtual Network Functions and Their Infrastructure. https://datatracker.ietf.org/doc/html/rfc8172

[28] IETF. 2022. RFC 9232: Network Telemetry Framework. https://datatracker.ietf.org/doc/html/rfc9232

[29] Wolfgang John, Farnaz Moradi, Bertrand Pechenot, and Pontus Sköldström. 2017. Meeting the observability challenges for VNFs in 5G systems. In *IFIP/IEEE Symposium on Integrated Network and Service Management*. IFIP/IEEE, Lisbon, Portugal, 1127–1130.

[30] R.E. Kalman. 1960. On the general theory of control systems. *International IFAC Congress on Automatic and Remote Control* 1, 1 (1960), 491–502.

[31] Taekhee Kim, Taehwan Koo, and Eunkyoung Paik. 2015. SDN and NFV benchmarking for performance and reliability. In *Asia-Pacific Network Operations and Management Symposium*. IEEE, Busan, Korea, 600–603.

[32] Stanislav Lange, Hee-Gon Kim, Se-Yeon Jeong, Heeyoul Choi, Jae-Hyung Yoo, and James Won-Ki Hong. 2019. Predicting vnf deployment decisions under dynamically changing network conditions. In *IFIP/IEEE/ACM International Conference on Network and Service Management*. IFIP/IEEE/ACM, Halifax, Canada, 1–9.

[33] Yang-Yu Liu, Jean-Jacques Slotine, and Albert-László Barabási. 2013. Observability of complex systems. *Proceedings of the National Academy of Sciences* 110, 7 (2013), 2460–2465.

[34] Marcelo Caggiani Luizelli et al. 2017. The actual cost of software switching for NFV chaining. In *IFIP/IEEE Symposium on Integrated Network and Service Management*. IFIP/IEEE, Lisbon, Portugal, 335–343.

[35] Charity Majors, Liz Fong-Jones, and George Miranda. 2022. *Observability Engineering*. O'Reilly, Springfield, Missouri.

[36] Joao Martins, Mohamed Ahmed, Costin Raiciu, Vladimir Olteanu, Michio Honda, Roberto Bifulco, and Felipe Huici. 2014. {ClickOS} and the Art of Network Function Virtualization. In *USENIX Symposium on Networked Systems Design and Implementation*. USENIX, Seattle, USA, 459–473.

[37] Chris Misa, Ramakrishnan Durairajan, Reza Rejaie, and Walter Willinger. 2021. Revisiting Network Telemetry in COIN: A Case for Runtime Programmability. *IEEE Network* 35, 5 (2021), 14–20.

[38] Arthur N Montanari and Luis A Aguirre. 2020. Observability of network systems: A critical review of recent results. *Journal of Control, Automation and Electrical Systems* 31, 6 (2020), 1348–1374.

[39] OpenConfig Project. 2025. OpenConfig: Vendor-neutral, model-driven network management designed by users. https://www.openconfig.net/

[40] perfSONAR Project. 2025. perfSONAR: performance Service-Oriented Network monitoring ARchitecture. https://www.perfsonar.net/

[41] Guto Leoni Santos, Diego de Freitas Bezerra, Elisson da Silva Rocha, Leylane Ferreira, André Luis Cavalcanti Moreira, Glauco Estácio Gonçalves, Maria Valéria Marquezini, Ákos Recse, Amardeep Mehta, Judith Kelner, et al. 2022. Service function chain placement in distributed scenarios: a systematic review. *Springer Journal of Network and Systems Management* 30, 1 (2022), 4.

[42] Jie Sun, Yi Zhang, Feng Liu, Huandong Wang, Xiaojian Xu, and Yong Li. 2022. A survey on the placement of virtual network functions. *Elsevier Journal of Network and Computer Applications* 202 (2022), 103361.

[43] Thales Nicolai Tavares, Leonardo da Cruz Marcuzzo, Vinícius Fulber Garcia, Giovanni Venâncio de Souza, Muriel Figueredo Franco, Lucas Bondan, Filip De Turck, Lisandro Zambenedetti Granville, Elias Procópio Duarte Junior, Carlos Raniery Paula dos Santos, et al. 2018. Niep: Nfv infrastructure emulation platform. In *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 173–180.

[44] Rogerio C Turchetti and Elias Procopio Duarte. 2015. Implementation of failure detector based on network function virtualization. In *2015 IEEE International Conference on Dependable Systems and Networks Workshops*. IEEE, 19–25.

[45] Rogério C Turchetti, Elias P Duarte Jr, Luciana Arantes, and Pierre Sens. 2016. A QoS-configurable failure detection service for internet applications. *Journal of Internet Services and Applications* 7, 1 (2016), 9.

[46] Muhammad Usman, Simone Ferlin, Anna Brunstrom, and Javid Taheri. 2022. A survey on observability of distributed edge & container-based microservices. *IEEE Access* 10 (2022), 86904–86919.

[47] Niels L. M. van Adrichem, Christian Doerr, and Fernando A. Kuipers. 2014. OpenNetMon: Network monitoring in OpenFlow Software-Defined Networks. In *IEEE Network Operations and Management Symposium*. IEEE, Krakow, Poland, 1–8.

[48] Giovanni Venâncio, Vinícius Fulber Garcia, Leonardo da Cruz Marcuzzo, Thales Nicolai Tavares, Muriel Figueredo Franco, Lucas Bondan, Alberto Egon Schaeffer-Filho, Carlos Raniery Paula dos Santos, Lisandro Zambenedetti Granville, and Elias P. Duarte Jr. 2021. Beyond vnfm: Filling the gaps of the etsi vnf manager to fully support vnf life cycle operations. *International Journal of Network Management* 31, 5 (2021), e2068.

[49] Giovanni Venâncio, Rogério C Turchetti, Edson T Camargo, and Elias P Duarte Jr. 2021. VNF-Consensus: A virtual network function for maintaining a consistent distributed software-defined network control plane. *International Journal of Network Management* 31, 3 (2021), e2124.

[50] Giovanni Venâncio, Rogério C Turchetti, and Elias P Duarte. 2019. Nfv-rbcast: Enabling the network to offer reliable and ordered broadcast services. In *2019 9th Latin-American Symposium on Dependable Computing (LADC)*. IEEE, 1–10.

[51] Giovanni Venâncio, Rogério C Turchetti, and Elias Procópio Duarte Jr. 2022. Nfv-coin: Unleashing the power of in-network computing with virtualization technologies. *Journal of Internet Services and Applications* 13, 1 (2022), 46–53.

[52] Mirko Viroli and Andrea Omicini. 2002. Specifying agent observable behaviour. In *International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM, New York, USA, 712–720.

[53] Qixia Zhang, Fangming Liu, and Chaobing Zeng. 2021. Online Adaptive Interference-Aware VNF Deployment and Migration for 5G Network Slice. *IEEE/ACM Transactions on Networking* 29, 5 (2021), 2115–2128.