Design of network setup automation using gitops operation

Mafeni Vitumbiko
School of Electronic Engineering
Soongsil University
Seoul, Korea
vitumafeni@dcn.ssu.ac.kr

Younghan Kim
School of Electronic Engineering
Soongsil University
Seoul, Korea
younghak@ssu.ac.kr

***Abstract*—The deployment of cloud-native private 5G networks on public, private, or hybrid cloud platforms is becoming a key trend for operators and vendors. While cloud-native architectures offer rapid and software-driven deployments, they introduce challenges in automation, scaling, and network consistency. Current studies often lack validation in real-world environments. This paper presents a framework that uses open-source projects and GitOps principles to automate the deployment and management of cloud-native private 5G networks on real infrastructure by leveraging Kubernetes and OpenStack. In addition, this paper addresses the fixed network topology issues encountered when transitioning from a test environment to a real production environment. The proposed framework facilitates rapid deployment and automation across multi-cluster environments, providing a practical solution for operators.**

***Index Terms*—Automation, Deployment, Private 5G, Multi-cluster, Kubernetes, GitOps**

I. Introduction

In recent years, researchers have increasingly utilized virtualization platforms like OpenStack and VMware to experiment with network configurations and deployments. Virtual Network Functions (VNFs) are now evolving from Virtual Machines (VMs) to more lightweight and dynamic solutions in the form of containers, known as Cloud-Native Network Functions (CNFs) [1]. Kubernetes is the leading platform for running Cloud-Native Network Function (CNF) applications. However, the growing complexities of cloud-native network applications, particularly with 5G containerized applications, demand the implementation of automation. The scope of these automation processes covers a wide range of tasks, from deployment to immediate reconfiguration of a particular Network Function (NF) [2]. By leveraging the Kubernetes Resource Model (KRM) and extending its API through Custom Resource Definitions (CRDs), network intents or NFs can be defined and managed by the underlying Kubernetes controllers. Often, a single Git repository is used to store these KRM resources, serving as a single source of truth for the entire desired state of the network from a centralized system [3]. GitOps [4] is an operational framework that uses Git as the single source of truth for managing declarative infrastructure and applications. Its principles include: declarative infrastructure to define the desired state, Git as the source of truth for all configurations, automated delivery through CI/CD pipelines, continuous reconciliation to ensure the system matches the desired state, and collaboration via pull requests (PRs) for safe and transparent changes. These principles improve reliability, security, and efficiency by automating deployments and reducing manual intervention. Several other projects and manuscripts exist adhering to GitOps principles, but they do not necessarily focus on the complex management, configuration, and deployment of private 5G networks. Nephio [5] is an open-source project that specifically targets private 5G management while following GitOps principles. Nephio [6] aims to provide carrier-grade, simple, and open Kubernetes-based cloud-native intent automation, along with common automation templates that significantly simplify the deployment and management of multi-vendor cloud infrastructure and network functions across large-scale edge deployments. Regrettably, the Nephio project currently focuses only on running and testing free5GC and OAI 5G use cases on Kubernetes IN Docker (KIND) [7] clusters using a Containerlab [8] environment, rather than on actual Kubernetes clusters. This approach limits the ability to fully assess system performance and behavior in real-world scenarios. Testing on real Kubernetes clusters would provide more accurate insights and validate the system's robustness in practical use cases. Therefore, in this paper, we build upon Nephio and explore how to transition the Proof of Concept (PoC) from running on KIND to real Kubernetes clusters hosted on OpenStack. The key contributions of our paper include:

- Providing an architectural overview of Nephio on real Kubernetes clusters running on OpenStack, along with detailed setup procedures.
- Adapting and optimizing the Nephio private 5G topology from KIND clusters to real Kubernetes clusters, addressing the limitations of KIND environments.
- Integrating with ArgoCD as the GitOps tool for automated package deployment, ensuring efficient and consistent management across multi-cluster environments.

The rest of the article is structured as follows. Section II presents related work. Section III presents the proposed system description. Section IV presents preliminary experiment and evaluation of the proposed work. Finally, section V presents our future planned directs and conclusions.

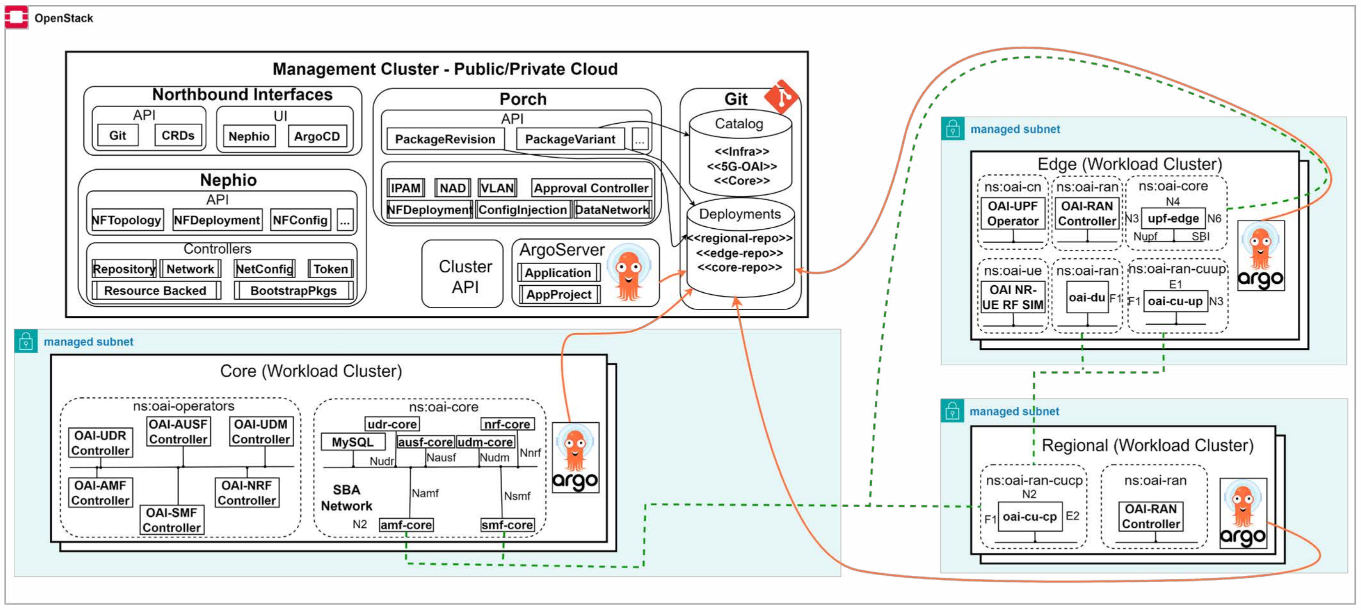

Fig. 1. Proposed Nephio OpenStack Topology

II. Related Work

This section presents various existing studies in relation to our study with distinct components and focus.

The paper [2] focuses specifically on using Kubernetes Operators for Netconf-based configuration management, emphasizing the automation of network configurations using GitOps principles. However, it is somewhat narrow in scope, concentrating primarily on configuration management rather than broader lifecycle management or other aspects of network function orchestration. In [9], the focus is on using the Open Network Automation Platform (ONAP) to automate 5G network slices but it can be overly complicated for operators who need quick, efficient deployment solutions. The paper [10] emphasizes the challenges of containerizing 5G workloads and proposes solutions using the Smart Edge framework. However, while the approach is cloud-native, it relies heavily on specific tools (e.g., Ansible, Helm), which may introduce complexity and reduce flexibility compared to more modern practices like GitOps. Unlike our approach, the paper [11] focuses only on automating the configuration and implementation of the Access and Mobility Function (AMF), making it not fully automated. The article in [12] explores the use of Kubernetes operators and the declarative approach through KRM to automate the management and configuration of edge-located User Plane Functions (UPFs). However, this approach does not leverage the benefits of the GitOps methodology to simply the task.

III. System Description

In this section, we introduce the main conceptual parts of our proposal and select an open source project, Nephio [6], as the base framework software. We utilize OpenStack [13] as the replacement platform to KIND for running our

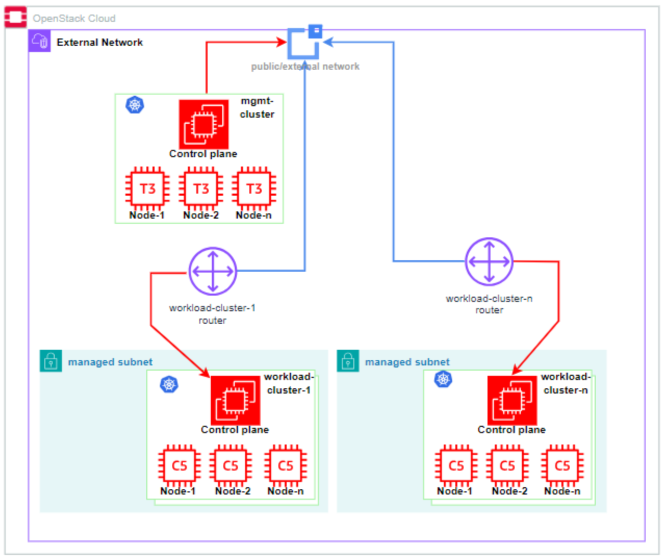

Fig. 2. Nephio OpenStack Subnets View

clusters within the private 5G network topology. OpenStack offers a dashboard for administrators to maintain control while enabling users to provision resources via a web interface [13]. In the default Nephio configuration, Containerlab typically employs a flat networking model, where all containers share a single network namespace or bridge. In our proposed architecture, depicted in Fig. 1, we adhere to the original Nephio network topology, maintaining the two cluster types: the management cluster and workload cluster (the core cluster, the regional cluster and the edge cluster). However, instead of running the clusters on a KIND-containerlab infrastructure, they are now deployed as real clusters within an OpenStack environment. Each workload cluster operates in its own isolated subnet, as shown in Figure 2. All workload clusters are created through the Cluster API through Nephio webui in the management cluster, which resides in the public subnet. The management cluster is set up separately using Ansible scripts. The management cluster is connected to the public network for external access directly for the git server especially if there would be other workload clusters deployed in a different cloud, while workload clusters remain isolated in managed subnets but can still access the external network through routers. The experimental network setup works because all clusters share the same underlying cluster network. When configurations for each cluster are deployed to their respective Git repositories, the ArgoCD instance running in each cluster pulls these configurations. The 5G OAI network functions are static, pre-configured KRM files stored in an upstream Git repository. Upon deployment, these network functions connect seamlessly through specified named interfaces such as NG, Xn, E2, and F1. For example, the RAN component in the regional cluster connects to the Core and Edge clusters via these interfaces, while the Edge cluster connects to the Core cluster. These connections adhere to ORAN standards. The placement of Core and RAN network functions, as well as the naming conventions for the clouds, is based on the O-RAN.WG6.CADS-v08.00 [14]

IV. Preliminary Experiment and Results

This section highlights the initial multi-cluster setup on OpenStack and a basic comparison with the KIND setup, based on the original proof of concept from the Nephio project community.

A. Cluster Setup on OpenStack

The management cluster with three nodes(one control plane node and two worker nodes) is connected to the public network, allowing interaction with external resources, while workload clusters are isolated within managed subnets but still have access to the external network through routers. In other words, our setup reflects a typical deployment model for a cloud environment that considers scalability, network segmentation, and dynamic IP management—ideal for complex systems like 5G networks. On all the clusters, we replaced ConfigSync with ArgoCD as the deployment tool to use for gitops operations. The git repository for each cluster still stored over gitea git server running in the management cluster. New Nephio packages are introduced to match the OpenStack infrastructure.

B. Experiment and Preliminary Results

In our preliminary results, we conducted a series of experiments to evaluate our proposal. OpenStack cloud consists of three nodes each having Intel Xeon Gold 6230 with 20 cores each and 188 GB of memory. Both the management cluster on KIND and on OpenStack were configured with the same instance specifications as outlined in the original Nephio requirements. The guide and configuration instructions of our experiment can be found here [15]. To test the connectivity between workload clusters, we utilized the iPerf program [16], which allowed us to a ping test across the workload clusters - assumed to be in different regions or availability zones. On the other hand, we used cilium Container Network Interface(CNI) cluster mesh [17] for multi-cluster inter-connectivity.

From the experiment, a comparison of the average deployment time required to bootstrap the management cluster on a

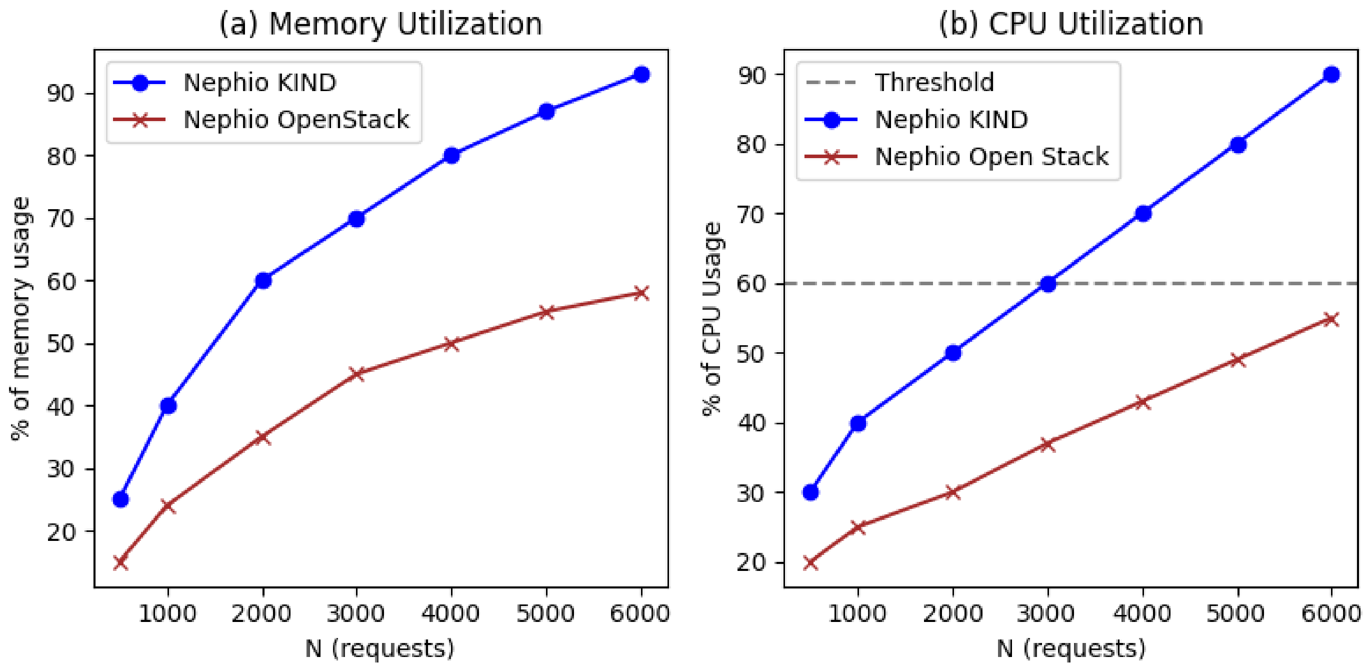

Fig. 3. Resource Utilization

KIND setup versus our proposed approach was conducted. The results clearly indicate that our approach is twice as efficient in terms of deployment time. Specifically, our proposed method completed the bootstrap process 40 minutes faster than the Nephio KIND approach, which required up to 1 hour and 21 minutes to bootstrap the management cluster. However, when comparing the deployment time for workload clusters, there were no significant differences. Additionally, even though setting up multi-cluster connectivity is challenging [18], we conducted inter-cluster connectivity tests using the iPerf program. Although these tests were performed manually at this stage, we successfully established inter-cluster connectivity. We further tested the CPU and memory utilization of our approach in comparison with the original Nephio KIND to evaluate its feasibility and performance in a production environment. Specifically, we investigated how an increasing number of requests to the system would affect resource utilization. Firstly, in the memory usage scenario (as shown in Figure 3a), we observed that in both cases—Nephio KIND and Nephio OpenStack—memory usage increased as the number of requests increase. Similarly, we examined CPU usage in relation to the number of requests, setting a threshold at 60%. The results, as shown in Figure 3b, indicate that CPU usage increases linearly in both Nephio KiND and Nephio OpenStack. However, Nephio KIND was shown to use more resources than Nephio OpenStack because all Nephio components (management cluster and workload cluster) are installed on a single server, whereas in Nephio OpenStack, the components are distributed across multiple nodes in clusters. Nephio KIND reaches the 60% threshold at 3,000 requests, again due to the fact that all system components are installed on one server. Nephio OpenStack has shown to be feasible because it fell below the threshold at 55% with 6,000 requests.

V. Conclusion and Future Directions

In this study, we presented the design of a 5G network topology automation setup based on the Nephio project, which currently supports KIND cluster deployments. We showcased our preliminary network setup and experiments to demonstrate the efficiency and potential of enhancing and transition from the initial Nephio KIND proof of concept. Our preliminary experiment and results show that our proposal is feasible for a production use cases. Future work will focus on expanding the setup to other cloud providers, fully automating the network configuration, particularly cluster inter-connectivity, and further reducing deployment time during the bootstrap.

Acknowledgment

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (RS-2022-II221015, Development of Candidate Element Technology for Intelligent 6G Mobile Core Network and RS-2024-00398379, Development of High Available and High Performance 6G Cross Cloud Infrastructure Technology)

References

[1] Á. Vázquez-Rodríguez, C. Giraldo-Rodríguez, and D. Chaves-Diéguez, "A cloud-native platform for 5g experimentation," in *2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. IEEE, 2022, pp. 60–64.

[2] Á. Leiter, A. Hegyi, I. Kispál, P. Böõsy, N. Galambosi, and G. Z. Tar, "Gitops and kubernetes operator-based network function configuration," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–5.

[3] P. Wörndle, S. Terrill, and T. Dinsing, "Automating telecom software deployment with gitops," *Ericsson Technology Review*, vol. 2023, no. 2, pp. 2–10, 2023.

[4] T. A. Limoncelli, "Gitops: a path to more self-service it," *Communications of the ACM*, vol. 61, no. 9, pp. 38–42, 2018.

[5] A. Kapadia, "What is the nephio project?" Available at https://www.aarnanetworks.com/post/what-is-the-linux-foundation-nephio-project (2022/06/29).

[6] Nephio, "Nephio: Cloud native network automation," Available at hhttps://nephio.org/ (2024/07/14).

[7] G. Madapparambath, "Exploring kubernetes 1.29 with kind," Available at https://medium.com/techbeatly/exploring-kubernetes-1-29-with-kind-a2704e1c729d (2024/01/13).

[8] S. Rajhi, "Containerlab: A great tool for orchestrating and managing container-based networking labs," Available at https://medium.com/p/7d05d50deec7 (2023/10/26).

[9] V. Q. Rodriguez, F. Guillemin, and A. Boubendir, "Automating the deployment of 5g network slices using onap," in *2019 10th International Conference on Networks of the Future (NoF)*. IEEE, 2019, pp. 32–39.

[10] Q. Zhao, S. Ranganath, S. Feng, G. Li, S. J. Li, Z. Shi, B. Ding, and J. Gao, "Customizable cloud-native infrastructure for private 5g," in *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. IEEE, 2023, pp. 50–57.

[11] K. Du, X. Wen, L. Wang, and T.-T. Nguyen, "A cloud-native based access and mobility management function implementation in 5g core," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. IEEE, 2020, pp. 1251–1256.

[12] Á. Leiter, I. Kispál, A. Hegyi, P. Fazekas, N. Galambosi, P. Hegyi, P. Kulics, and J. Bíró, "Intent-based 5g upf configuration via kubernetes operators in the edge," in *2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2022, pp. 186–189.

[13] OpenStack, "Welcome to openstack documentation," Available at https://docs.openstack.org/2024.1/?_ga=2.29009081.480386473.1725245166-399538587.1722342014 (2024/04/13).

[14] A. ORAN, "O-ran working group 6 cloud architecture and deployment scenarios for o-ran virtualized ran," Available at https://specifications.o-ran.org/specifications (2024/10/29).

[15] M. Vitumbiko, "Nephio over openstack," Available at https://github.com/VituMafeni/nephio-test-infra-openstack (2024/03/04).

[16] E. . L. B. N. Laboratory, "What is iperf / iperf3 ?" Available at https://iperf.fr/ (2024/02/20).

[17] C. Authors, "Setting up cluster mesh," Available at https://docs.cilium.io/en/stable/network/clustermesh/clustermesh/#gs-clustermesh (2024/08/14).

[18] T.-N. Nguyen, J. Lee, M. Vitumbiko, and Y. Kim, "A design and development of operator for logical kubernetes cluster over distributed clouds," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, 2024, pp. 1–6.