ORWELL: Monitorization Platform for a 5G Testbed

Alexandre Serras (97505), Gonçalo Leal (98008), Pedro Duarte (97673), Vasco Regal (97636)
Projeto em Informática 2021/2022
DETI
Universidade de Aveiro
Aveiro, Portugal
Email: alexandreserras@ua.pt, goncalolealsilva@ua.pt, pedro.dld@ua.pt, vascoregal24@ua.pt

***Abstract*—5G networks are fiercely developing and so are the vertical applications of this technology. To develop new 5G applications, researchers need to test them. This necessity led to the creation of 5G testbeds with state-of-the-art technology. These tests require a controlled environment, thus all the infrastructure has to be monitored. This work discusses an approach that complies with 5GASP's approach and centralizes all the information on a unique database. Moreover, the proposed methodologies were implemented in a proof of concept product which demonstrates the potential of the proposed approach.**

***Index Terms*—5G, Monitoring, Testbed, Prometheus, Grafana, OSM, VNF**

I. Introduction

Before embarking upon a full-scale project, researchers conduct pilot studies that evaluate feasibility, computational costs and potential problems regarding their Network Applications (NetApps). These tests are held in a controlled environment over state-of-the-art infrastructure called testbed. 5G testbeds offer complete 5G system functionalities such as a 5G Core and 5G RAN. Besides the network infrastructure and since 5G applications are deployed on virtual machines or containerized environments based on the cloud, 5G testbeds provide sets of servers that host several Virtual Network Functions. Allied with all this infrastructure, a fully functional testbed also grants a trustworthy CI/CD pipeline for NetApps deployment on the test environment

The testing phase requires continuous monitoring of both, the VNFs being tested and the network infrastructure. In order to achieve quality results in the end of this phase, the testbed must remain as a controlled environment maintaining the same performance throughout the testing process. Assuming that the testbed's performance levels are stable, one can assume that every performance variation detected during a NetApp test was caused by malfunction on the VNF being tested. This leads to the implementation of monitoring tools in the testbed and in all the machines under test. However, most monitoring solutions require direct access to the VNFs which may cause some privacy issues and can bring some future issues regarding the testing results, since it was accessed by someone other than its developer.

II. Proposed Monitoring System

This work presents new mechanisms and a toolset to monitor a 5G testbed. The toolkit relies mostly on open-source tools capable of monitoring VNFs, network's performance and security, system's security liabilities and 5G infrastructure. When compared to other state of the art projects such as NetGraf [1], Orwell presents an alternative architecture, which brings an uniform access interface and data storage format. This system considers VNFs as blackboxes, so all the metrics are collected through a non-intrusive process.

A. Exporting Metrics

Because both VNFs and the network are monitored, multiple approaches for different metrics are required.

Regarding VNF monitorization, as we can't ever access them, in the worst case scenario we have to collect metrics through the Gnocchi API, which collects metrics via OpenStack (hypervisor). However, the quantity of metrics measured through this tecnique is quite limited. On the other hand, different OS image were created with pre-configured exporters, which allows a VNF to launch with a proper metric exporter such as Telegraf or Prometheus' Node Exporter running, which allow the gathering of much more detailed metrics.

When it comes to network metrics, they will be collected for both the network and the 5G core, for all network devices, called slots, and network interfaces. PerfSonar, the tool used to obtain network metrics was configured in two different VMs, one running PerfSonar Toolkit, considered the main node, and the other a PerfSonar Testpoint, which is a lighter version. This setup is currently running inside of ITAv's network, but the objective is having nodes in different testbeds, enabling to collect metrics about the connectivity between two physically distant places. In the context of obtaining 5G core metrics, requests to Huawei's eSight API are made. However, this API is not yet stable inside of ITAv and we were only able to collect a small amount of metrics compared to our initial objective.

B. Middleware

This is the most crucial piece of our solution which is responsible for linking the data providers, metric exporters, with the data storage, Prometheus. More than that, through its API, it contains the required service discovery endpoints, directly linked with the OSM, a PerfSonar endpoint for configuring tests dinamically with new nodes and a couple informative endpoints for checking running VNF's details, such as the available exporters. It is also responsible for orchestrating Grafana, creating and updating user accounts, organizations and dashboards in response to events.

C. Data Visualization

To see the metrics collected, Grafana queries Prometheus, and presents the returned information in four separate dashboards: VNFs, perfsonar, eSight slots, and eSight interfaces, and thus the various metrics gathered can be observed in the corresponding dashboards without a mix of information.

D. Security Mechanisms

Suricata and Infection Monkey, two existing security technologies, were configured to assure the security of everything that happens inside the testbed.

Suricata is a Network Security Monitoring application that examines and processes network traffic using sets of community-created and user-defined signatures.

Infection Monkey is a security tool to simulate network exploits and report vulnerabilities.

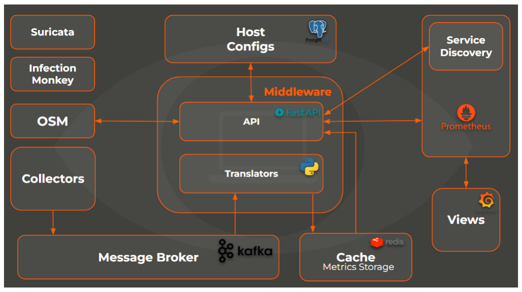

Fig. 1. System Architecture

III. Workflow

The workflow of our final solution, currently implemented inside of ITAv's network, starts with the deployment of a VNF. This action is done through the OSM service, where an admin is able to configure the new machine with the intended hardware resources and a base OS image which, as already mentioned, should have a running metrics exporter by default. As of now, we provided one image with a Telegraf instance and another with a Prometheus' Node Exporter but launching a different image, even if it has no exporters running, will not exclude the new VNF from our system's flow, as we are also able to gather metrics from Gnocchi, at the hypervisor level.

When the machine is running it is a matter of time until it is detected by our Middleware's service discovery, which is linked with Prometheus, making every machine available from of the OSM's API appear in Prometheus' targets. The information that reaches this last tool is associated with a url where it is expected that the VNF's metrics will appear, already translated, and always points to our Middleware's /metrics/<host> endpoint. Here we also introduce the concept of virtual hosts, which are associated with static endpoints, reserved for the metrics of specific services not related with a unique host, namely network and 5G core metrics. Also at this step, when the service discovery finds a new machine it creates Grafana credentials for the registered responsible for the VNF and generates a dashboard for tracking metrics.

The multiple exporters now running push their metrics regularly to a Kafka topic reserved for the specific service associated and soon a translator should read that message, transform it to comply with Prometheus' format and store it in a Redis cache, inside of a list with the name of the data origin. A request to the /metrics/<host> endpoint referred earlier has the effect of retrieving all the metrics stored in a specific Redis' list and deleting it. The requests are done frequently by Prometheus, which will read and store all the metrics returned.

Finally, a user is now able to login into Grafana with his credentials, access the created dashboard and visualize the results. Users will also be able to receive alerts through Slack when certain metrics meet certain condition(s).

IV. Conclusion

We were able to achieve most of our goals, monitoring VNFs with different or no exporters installed without directly accessing them, monitoring network metrics and monitoring the 5G Core but, even more important, we were able to create a fully modular system and develop strategies to facilitate the easy integration of new metrics. To assure the testbed's security we configured security tools, Suricata and Infection Money, and even created a Selenium-based API for Infection Monkey, which was lacking and is fundamental for proper automation of tasks. We did all this depending mostly on open-source tools and created a few new tools which will be made available for the community.

In this work we studied a large number of tools, technologies and concepts that were not familiar to any member of the group and subsequently ran into many challenges which were only surpassed due to the good dynamic and dedication of every team member, including the advisors. As a group, we are also excited about our contribution for the 5GASP European Project, which aims to boost the development process of 5G applications.

Future work would include dealing with horizontal scalibility mechanisms, configuring Kafka and Redis as scalable clusters and adding the ability to lauch new translators as required, depending on the amount of messages that are left to translate, improve our application's initialization procedure and developing a graphic interface for admins to analyze logs and statistics of the system as a whole and interact with specific components such as Grafana or Infection Monkey.

References

[1] D. Kaur, B. Mohammed and M. Kiran, 2022. NetGraf: A Collaborative Network Monitoring Stack for Network Experimental Testbeds. [online] Arxiv.org. Available at: https://arxiv.org/pdf/2105.10326.pdf