Adaptive End-to-End Monitoring Framework for Heterogeneous 5G and Beyond Networks

Leandro Alfonso*
Universidad de la República
Montevideo, Uruguay
leandro.alfonso@fing.edu.uy

Nicolás Rivoir*
Universidad de la República
Montevideo, Uruguay
nicolas.rivoir@fing.edu.uy

Lucas Inglés
Universidad de la República
Montevideo, Uruguay
lucasi@fing.edu.uy

Claudina Rattaro
Universidad de la República
Montevideo, Uruguay
crattaro@fing.edu.uy

Alberto Castro†
Universidad de la República
Montevideo, Uruguay
acastro@fing.edu.uy

Abstract

5G technology has ushered in complex multi-domain environments that demand dynamic and robust monitoring solutions. Traditional network monitoring strategies often fail to address the unique challenges of 5G networks, such as managing high data volumes, diverse service requirements, and cross-domain interactions across heterogeneous technologies. In this paper, we present an innovative end-to-end monitoring framework for 5G and beyond networks designed to enhance failure detection and localization capabilities, as well as overall performance evaluation across diverse 5G deployments. Our framework leverages statistical learning techniques to efficiently and adaptively analyze network data, focusing on specific traffic sub-populations that reflect immediate monitoring needs. The proposed monitoring system is designed to be programmable and application-sensitive, allowing on-the-fly configuration changes that are essential for multi-domain operations. By integrating flow-based measurements with intelligent sampling methods, our system significantly reduces the resource footprint traditionally required for comprehensive data collection and analysis. We have implemented and validated our framework using the ns-3 5G-LENA simulator. This approach enables us to evaluate the system's performance in realistic 5G scenarios and demonstrate its effectiveness across various network conditions and configurations, addressing the challenge of limited access to commercial 5G deployments. Preliminary results from our simulations demonstrate the framework's potential to remarkably improve network reliability, performance insights, and operational efficiency across heterogeneous 5G environments. Our approach facilitates more precise and scalable network management, setting the stage for adaptive monitoring solutions as 5G and beyond networks' demands evolve.

*Both authors contributed equally to this research.
†Corresponding author.

CCS Concepts

• **Networks** → **Network monitoring**; **Network simulations**; **Network measurement**; *Network architectures*; *Mobile networks*.

Keywords

monitoring; 5G; 6G; failure localization; ns-3

ACM Reference Format:
Leandro Alfonso, Nicolás Rivoir, Lucas Inglés, Claudina Rattaro, and Alberto Castro. 2024. Adaptive End-to-End Monitoring Framework for Heterogeneous 5G and Beyond Networks. In *Proceedings of the 4th ACM Workshop on 5G and Beyond Network Measurements, Modeling, and Use Cases (5G-MeMU '24), December 9–12, 2024, Los Angeles, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3694810.3700160

1 Introduction

The evolution of mobile networks to 5G/6G introduces transformative capabilities, including significantly higher data rates, reduced latency, and the ability to connect a massive number of devices simultaneously [1, 11, 13, 21]. However, these advancements come with increased complexity in network architecture [20], particularly in monitoring. Traditional network monitoring strategies often fail to address the unique challenges of 5G networks, such as managing high data volumes, ensuring diverse service requirements, and maintaining seamless cross-domain interactions across heterogeneous technologies [16]. Effective monitoring is critical for ensuring network reliability and security and managing the Quality of Experience (QoE) for end-users [1–3, 15]. In 5G/6G networks, not only is detecting network failures crucial but localizing these failures is equally essential for swift remediation and minimizing downtime. Failure localization enables targeted interventions, reducing the scope of disruptions and facilitating more efficient resource management and repair strategies. The dynamic and heterogeneous nature of 5G/6G networks demands robust and adaptable monitoring solutions capable of handling such complexity with enhanced detection and precise localization capabilities.

Traditional network monitoring methods such as Simple Network Management Protocol (SNMP), Remote Monitoring (RMON), and packet sniffing, while foundational in legacy networks, struggle to meet the dynamic demands of 5G/6G networks [17]. Their static nature and inefficiency in processing and analyzing large volumes of data compromise their effectiveness in these more complex

environments [7, 22]. Furthermore, these traditional methods are primarily reactive, typically identifying and reporting failures after they have occurred without providing mechanisms for proactive prevention or precise localization of issues.

As networks have evolved, there has been a shift towards the adoption of software-defined networking (SDN) and network functions virtualization (NFV), which require more sophisticated monitoring tools capable of dynamically adapting to changing network conditions and supporting complex configurations [5, 14, 19]. Despite advancements in monitoring solutions, including those leveraging machine learning, significant challenges persist. These systems often struggle with scalability, particularly in multi-domain 5G/6G environments where they must process an exponential increase in data points and network interactions [8]. Moreover, many of these intelligent systems demand substantial computational resources, hindering their ability to provide the real-time analysis crucial for immediate failure localization. This limitation underscores the urgent need for a new generation of monitoring systems that combine intelligence with agility and resource efficiency. Such systems must operate effectively within the demanding and dynamic context of 5G/6G networks, providing rapid and accurate monitoring and analysis across diverse network domains and technologies.

In this paper, we present an innovative end-to-end monitoring framework for 5G and beyond networks designed to enhance failure detection and localization capabilities, as well as overall performance evaluation across diverse 5G/6G deployments. Our framework leverages statistical techniques to efficiently and adaptively analyze network data, focusing on specific traffic sub-populations that reflect immediate monitoring needs. The proposed monitoring system is designed to be programmable and application-sensitive, allowing on-the-fly configuration changes that are essential for multi-domain operations. By integrating flow-based measurements with intelligent sampling methods, our system significantly reduces the resource footprint traditionally required for comprehensive data collection and analysis. We have implemented and validated our monitoring framework using the ns-3 5G-LENA simulator [9, 12]. This approach enabled us to evaluate the system's performance in realistic 5G scenarios and demonstrate its effectiveness across various network conditions and configurations, addressing the challenge of limited access to commercial 5G deployments.

Our contributions are fourfold: *i*) Enhanced Fault Detection and Localization: Our monitoring system not only detects network anomalies but also precisely pinpoints their location. This dual capability significantly reduces service disruptions and ensures high levels of network availability; *ii*) Improved Network Management Efficiency: Our system streamlines network management processes by significantly reducing the operational overhead associated with extensive data collection and analysis; *iii*) Resource Optimization: Our approach significantly reduces the computational and storage requirements through intelligent sampling and adaptive monitoring, resulting in a cost-effective and highly scalable solution for monitoring complex future network environments; and *iv*) Open-Source Implementation and Dataset: We provide a public repository containing our fully documented source code, which extends ns-3 functionalities to implement our monitoring system ([10]). Additionally, we offer a comprehensive dataset generated from our simulations.

2 Adaptive Monitoring Framework: Core Principles and Design

Our proposed monitoring framework centers around a key component: the Monitoring and Data Analysis (MDA) system. This centralized entity is designed to collect and analyze various Quality of Service (QoS) metrics across the entire network, encompassing all end-to-end domains. The MDA system's primary innovation lies in its adaptive approach to network monitoring, which optimizes resource utilization while maintaining comprehensive oversight of network performance.

The core principles of our adaptive monitoring approach are as follows: *i*) Centralized Metric Collection: The MDA system serves as a central repository for network-wide QoS metrics, providing a holistic view of the network's performance; *ii*) Adaptive Measurement Frequency: During normal network operation, measurements are taken at regular intervals. However, when anomalous behavior is detected, the system dynamically increases the monitoring frequency, allowing for a more detailed analysis of potential issues; *iii*) Performance Thresholds: The system employs predefined performance thresholds to determine the state of the network. These thresholds are based on a combination of key QoS metrics such as delay, throughput, and jitter; and could be set dynamically according to the network's operational state; *iv*) Granular Monitoring Escalation: When performance metrics fall below the acceptable threshold, the system not only increases overall measurement frequency but also initiates node-to-node measurements for the affected flows. This granular approach allows for precise localization of performance issues (e.g., backbone network, a specific link); and *v*) Multi-domain Compatibility: The MDA system is designed to operate across various network domains, including both mobile and optical networks. This capability ensures comprehensive monitoring of complex 5G infrastructures spanning both wireless and optical technologies.

In Figure 1, we illustrated the adaptive nature of our monitoring approach. The graph depicts network performance over time, with measurement instances represented by circles. The horizontal line indicates the performance threshold. When performance drops below this threshold, the measurement intervals decrease, reflecting the system's ability to adapt its monitoring intensity based on network conditions. This adaptive strategy offers several advantages: *i*) Resource Efficiency: By adjusting monitoring frequency based on network conditions, the system minimizes unnecessary data collection and analysis during periods of normal operation; *ii*) Rapid Problem Detection: The increased monitoring frequency during anomalous periods enables quicker identification and localization of network issues; and *iii*) Scalability: The approach is inherently scalable, as it can be applied to networks of varying sizes and complexities without requiring constant high-frequency monitoring.

To implement and validate this adaptive monitoring framework, we chose the ns-3 network simulator, specifically utilizing the 5G-LENA module. This choice was motivated by ns-3's widespread use

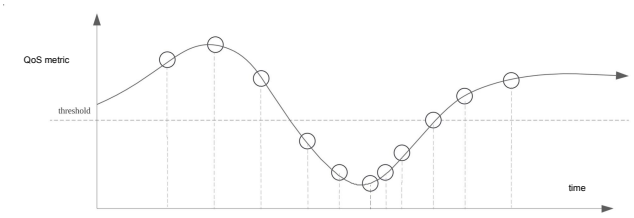

Figure 1: Adaptive monitoring concept illustration. The graph shows network performance (QoS metric) over simulation time (ms). Circles represent measurement instances, with the horizontal line indicating the acceptable performance threshold. Measurement frequency increases (intervals decrease) when performance falls below the threshold, demonstrating the system's adaptive nature.

in academia and its robust capabilities in simulating 5G network environments.

3 ns-3 Implementation Based On FlowMonitor

To develop a monitoring system capable of localizing a poorly performing link, it is necessary to identify the influence of each point-to-point link on the end-to-end metrics. Our solution is implemented based on the FlowMonitor module provided by ns-3 simulator [4].

The *FlowMonitor* framework provides a set of easily extendable classes to model monitoring behaviors tailored to specific needs. These include the *FlowMonitor* and *FlowProbe* classes, which are responsible for maintaining traces at both the end-to-end flow level and the probe or node level, respectively. Additionally, the *Ipv4FlowProbe* class, a subclass of *FlowProbe*, introduces methods for recording metrics during the transmission and reception of IPv4 packets across any of a node's interfaces.

To achieve our goal, we have developed additional classes within the simulation environment, as shown in Figure 2, which illustrates all the classes involved. One of these is a class called *BigBrotherFlowProbe*, which is derived from the *Ipv4FlowProbe* class. This new class introduces an attribute called *m_perPacketStats*, designed to store information about the delay each node introduces for each flow and associated packet. This structure allows us to accurately reconstruct the route of each packet through the network, enabling the identification of the link with the worst performance at any given time. Additionally, the class includes a polymorphic function, *AddPacketStats*, which takes an additional attribute called *packetId* and uses it to populate the *m_perPacketStats* map. Furthermore, a class named *BigBrotherFlowMonitor* was created, inheriting from *FlowMonitor*. Since the previous implementation was responsible for instantiating the *Ipv4FlowProbe* class, and to use our solution we need to instantiate "big-brother-flow-probes", a function *ReportFirstTx* was overloaded as: ReportFirstTx(Ptr<FlowProbe> probe, uint32_t flowId, uint32_t packetId, uint32_t packetSize).

Finally, an implementation of *FlowMonitorHelper* was also created, named *BigBrotherFlowMonitorHelper*. This is simply a copy of the originally provided implementation, but instead of instantiating traditional FlowMonitors, it instantiates *BigBrotherFlowMonitor*. By

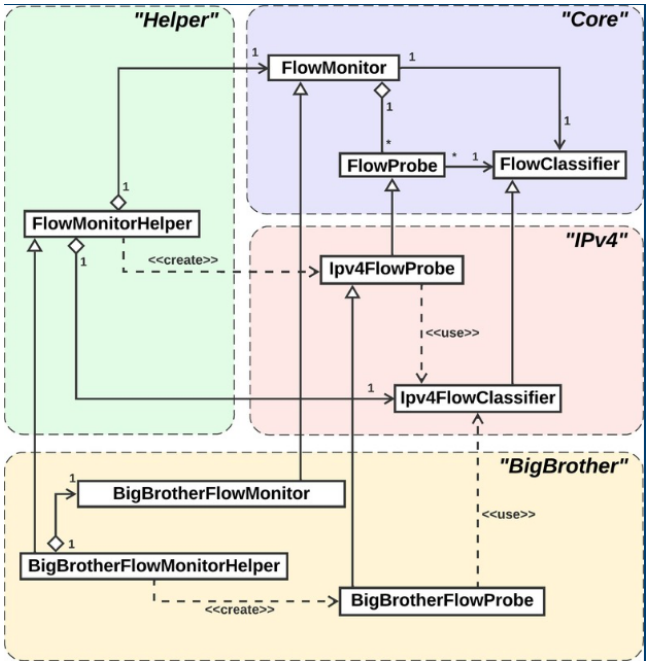

Figure 2: Architectural diagram of the monitoring system implementation. The structure consists of four main components: 'Helper' (FlowMonitorHelper), 'Core' (FlowMonitor, FlowProbe, FlowClassifier), 'IPv4' (Ipv4FlowProbe, Ipv4FlowClassifier), and 'BigBrother' (BigBrotherFlowMonitor, BigBrotherFlowMonitorHelper, BigBrotherFlowProbe). This diagram illustrates the relationships and inheritance between custom-developed classes and ns-3's existing modules.

including the "flow-monitor-helper.h" class, this solution can be utilized.

3.1 How Our Monitoring System works?

A monitoring function called *reportFlowStats* is implemented, which is responsible for measuring and analyzing the performance of a simulation (in particular our goal is to analyze flows in a 5G network simulation). This function works in conjunction with *nodeToNodeTrigger*. The operation of both functions is described in detail below.

3.1.1 reportFlowStats cycle. This function is a critical component of the developed monitoring system, designed to compute and store key QoS performance metrics. The operation of *reportFlowStats* can be detailed as follows:

(1) **Initialization and Configuration:**
- Initializes a structure called *TrackedStats* to store key performance metrics such as throughput, average latency, latency, and average jitter.
- Defines two time intervals: "NETWORK_OK_MEASURING TIME" and "NETWORK_NOT_OK_MEASURING TIME," which determine the frequency of statistics collection based on predefined thresholds. This can be scaled to other granularities in monitoring frequency.

(2) **Statistics Collection:**
- Retrieves flow statistics (*FlowStats*) from the *FlowMonitor*.
- Iterates over each flow, computing individual performance metrics and storing them in the *TrackedStats* structure.
- Prints the computed metrics to the console for debugging and writes them to a user-specified output file.

(3) **Median Latency Calculation:**
- Sorts the latency values of the flows and calculates the median latency, referred to as *FiftyTileFlowDelay*.

(4) **Threshold Comparison:**
- Compares the calculated performance metrics with predefined thresholds.
- If any metric exceeds its threshold, invokes the *nodeToNodeTrigger* function to analyze delays between network nodes and identify potential bottlenecks.

(5) **Scheduling:**
- Resets all statistics of the *FlowMonitor*.
- Schedules itself to run again after a specified time interval, determined by "MEASURING_TIME," which is adjusted based on whether the performance metrics exceeded the thresholds.

In summary, the *reportFlowStats* function is a key component in our approach, as it collects and processes flow statistics, calculates important performance metrics, identifies potential bottlenecks in the network, and schedules itself to run periodically during the simulation.

3.1.2 nodeToNodeTrigger cycle. The *nodeToNodeTrigger* function is an essential component of the developed monitoring system, leveraging the custom solution implemented with the *BigBrotherFlowProbe* and *BigBrotherFlowMonitor* classes. Its main goal is to identify the link with the worst performance in terms of *delay* between nodes in the simulated network. The functioning of *nodeToNodeTrigger* can be broken down into the following steps:

(1) **Initialization and Data Loading:**
- The function receives a pointer to a "`FlowMonitor`" and the path to an XML file where the results will be stored.
- Loads or creates an XML file to store the network measurements.

(2) **Statistics Collection:**
- Obtains the flow statistics and the *probes* from the flow monitor.
- Initializes a `nodeToNodeDelay` map to store the *delays* between pairs of nodes.

(3) **Flow Data Processing:**
- Iterates over each flow in the statistics.
- For each flow, creates a `perPacketStats` map that stores detailed information about the path of each packet.

(4) **Probe Analysis:**
- For each *probe* of type `BigBrotherFlowProbe`, extracts detailed information about the packets that passed through it.
- Collects data on the node ID and the accumulated *delay* for each packet.

(5) **Calculation of *Delays* Between Nodes:**
- Processes the collected information to calculate the *delay* between each pair of consecutive nodes in the packet path.
- Updates the `nodeToNodeDelay` map with these values.

(6) **Identification of the Worst-Performing Link:**
- Finds the pair of nodes with the highest accumulated *delay*.

(7) **XML File Update:**
- Records all *delay* measurements between nodes in the XML file.
- Adds a special entry for the worst-performing link.

(8) **Return of Results:**
- Returns a pair of integers representing the IDs of the nodes that form the worst-performing link.

This function leverages the information collected by *BigBrotherFlowProbe*, processing the packet-level data to obtain a clear view of the performance of each link in the network. By identifying the link with the highest *delay*, it provides a valuable tool for analyzing and optimizing the simulated network.

3.2 Standard Log Format

Previously, an element called "*XML File*" is mentioned. It refers to an XML file designed to store system logs in a way that they can be processed programmatically. The *tiny-xml-2* library [18] was chosen as a tool to build an *XML* document with the system results after the simulation. This XML format represents *delay* measurements and critical links in a network. The structure of the file is described below:

<**network-measurements**> This is the root element containing all network measurement information.
<**worst-links**> This element contains a list of the slowest or most critical links in the network.
  <**worst-link**> Represents a specific critical link.
    <**delay-value**> The *delay* value measured on this critical link (in nanoseconds).
    <**timestamp**> The timestamp at which this critical link measurement was recorded (in nanoseconds).
    <**node-pair**> The pair of nodes that defines the critical link.
      <**node-id**> The identifier of each node in the pair.
<**delays**> This element contains a list of *delays* measured between specific pairs of nodes.
  <**delay**> Represents the *delay* measurements between a pair of nodes.
    <**node-pair**> The pair of nodes for which *delay* measurements are recorded.
      <**node-id**> The identifier of each node in the pair.
    <**measurements**> Contains a list of individual *delay* measurements for this pair of nodes.
      <**measurement**> A specific *delay* measurement.
        <**delay-value**> The *delay* value measured (in nanoseconds).
        <**timestamp**> The timestamp at which this measurement was recorded (in nanoseconds).

In summary, the XML file contains information about the slowest links in the network (`<worst-links>`), as well as detailed *delay* measurements between pairs of nodes (`<delays>`). This can be useful for identifying bottlenecks and performance issues in the

network. It represents a more intuitive and compact way to analyze ns-3 traces, providing only the useful information needed for decision-making based on monitoring.

4 Experimentation

A progressive validation strategy was adopted to validate the monitoring system's correct functioning. It began with simple simulations to ensure certain aspects of the system worked correctly and gradually increased in complexity to create scenarios that resemble real 5G-NR networks. In our tests, the QoS metric evaluated was the delay (it could have been throughput, jitter, or packet loss probability).

4.1 Validation of Node-to-Node Metrics Extraction

The first test validates whether the classes implemented within the *FlowMonitor* module correctly extract node-to-node metrics for a specific flow. This functionality is essential for the system, as *FlowMonitor* cannot, by default, measure node-to-node link performance; it can only record metrics at the end-to-end or node level without being able to determine packet origins.

To demonstrate this, we designed several scenarios, ranging from simple point-to-point networks to more complex topologies. For this test, we used a simple topology consisting of four nodes (`n0`, `n1`, `n2`, `n3`) connected by point-to-point links. Nodes `n0` and `n1` are connected to `n2` by 5 Mb/s links with 4 ms delays. Node `n3` is connected to `n2` by a 1.5 Mb/s link, with an initial delay of 10 ms, later modified to 2 ms and 11 seconds. There are two constant UDP flows (CBR) between `n0` to `n3` and `n3` to `n1`. The UDP packet size is 210 bytes with a transfer rate of 448 Kbps. DropTail (FIFO) queues are used, and all packet queues and receptions are recorded in a trace file.

The flows are disjoint at nodes `n0` and `n1`. Thus, when using the monitoring solution, the system should correctly register traces for nodes `n0`, `n2`, and `n3` in Flow 1, but not for `n1`. Similarly, it should register traces for nodes `n2`, `n3`, and `n1` in Flow 2, but not for `n0`. The system's ability to generate low-granularity traces only for nodes involved in the measured flow is crucial for operating in more complex simulations with hundreds of nodes without creating excessive overhead or inefficient traces.

When the monitoring system is installed in the scenario using the *nodeToNodeTrigger* function (explained in Section 3.1.2), it successfully obtains the system trace (see Figure 3). This demonstrates the system's capability to perform active measurements only on the relevant nodes. In Figure 3, we show each section linked by an arrow to the point-to-point link they measure. The following points highlight why this experiment is successful:

- The links `(n0,n2)` and `(n1,n2)` are each measured once. This is correct as they are used by only one flow each, flows 1 and 2, respectively.
- The link `(n2,n3)` is measured twice. This is correct as it is accessed by both Flow 1 and Flow 2 and is the only link present in both flows. Its two `<measurements>` entries indicate success.

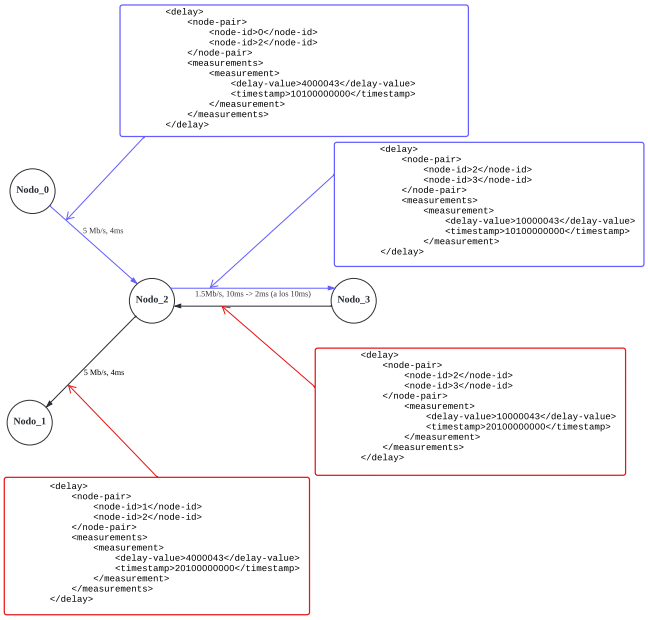

Figure 3: Validation scenario topology with associated XML measurement outputs. Four-node network (Nodo_0 to Nodo_3) with point-to-point links. XML snippets demonstrate the system's ability to correctly measure and record delays for each link, with blue representing flow 1 (Nodo_0 to Nodo_3) and red representing flow 2 (Nodo_3 to Nodo_1).

- The measured delays correspond to those configured in the scenario described, with `(n0,n2)` and `(n1,n2)` having a delay of 4 ms or 4×10^6 ns, and `(n2,n3)` having a delay of 10 ms.

4.2 Controlled Anomaly Detection

In this second group of experiments, we simulate a 5G network and generate anomalies at certain points during the simulation.

Indicators: Several key indicators were considered to evaluate the performance and accuracy of the network monitoring system:

- True Positive (TP): Correct activation of the system
- False Positive (FP): Incorrect activation of the system
- False Negative (FN): Incorrect non-activation of the system
- True Negative (TN): Correct non-activation of the system
- Precision (PR): $\frac{TP}{TP+FP}$
- Recall (RE): $\frac{TP}{TP+FN}$
- Accuracy (CA): $\frac{TP+TN}{TP+TN+FN+FP}$

Evaluation: To simulate a network and experiment by generating anomalies, the scenario shown in Figure 4 was used. The 5G operator's backbone network was emulated with several point-to-point links where malfunctions were generated.

By evaluating these indicators, network administrators can determine how well the monitoring system performs in detecting anomalies, ensuring that the network remains robust, secure, and efficient.

Table 1: Performance Metrics.

UEs	gNBs	TP	FP	FN	TN	PR	RE	CA	F1
30	3	21	1	0	17	0.954545	1	0.678571	0.971429
60	3	19	1	1	17	0.95	0.95	0.666667	0.944444
90	3	19	1	1	17	0.95	0.95	0.666667	0.944444
120	3	19	15	1	10	0.558824	0.95	0.537037	0.555556
210	21	13	5	4	15	0.722222	0.764706	0.583333	0.769231

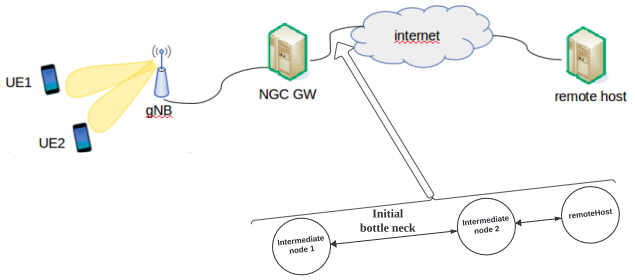

Figure 4: Enhanced 5G end-to-end architecture for simulation. The diagram illustrates the integration of intermediate nodes (intermediate_node_1, intermediate_node_2) and remoteHost within a representative 5G network topology. This configuration enables fine-grained control over link performance and facilitates the simulation of backbone network bottlenecks for more comprehensive monitoring system evaluation. (Adapted from [6])

In Table 1, the results of the experimentation conducted are presented. For this evaluation, we have modeled a 5G network with only NGMN_VIDEO DL traffic and UDP packets. The number of *ues* and *gNBs* was incrementally increased in order to study the monitoring system's behavior in increasingly complex scenarios. The conclusions from this experiment were that the system performed satisfactorily, and the extracted metrics reflect this. It is understood that the experiment's success is partly due to the experimental design favoring the monitoring system's operation. This refers to the fact that as the links where the *bottleneck* is installed are progressively affected by introducing more *delay*, it is logical that the system starts taking more frequent measurements because they occurred at a time when the network was proactively deteriorated. Each of these will be classified as TP, thus achieving a high rate of TP values and a low rate of FP, which substantially improves the system's performance. Although in this experiment we have focused on the backbone network depicted in Figure 4, it is important to emphasize that the system we have implemented not only facilitates the monitoring of optical links (backbone links) but also provides the capability to detect anomalies in the radio links.

5 Conclusions

We have introduced a novel multi-domain monitoring framework for 5G and beyond networks, addressing the critical challenges of failure detection and localization in complex, heterogeneous network environments. This monitoring framework is designed to be integrated into a comprehensive system supporting slice provisioning, intelligent routing, and Quality of Experience (QoE) management, while enhancing network security through improved anomaly detection capabilities.

Preliminary results from our simulations demonstrate the framework's potential to remarkably improve network reliability, performance insights, and operational efficiency across heterogeneous 5G environments. In particular, we showed several key advancements: *i*) Dynamic Identification of Performance Bottlenecks: The system can pinpoint the worst-performing node-to-node links in real-time, enabling rapid localization and resolution of network issues. This capability is crucial for maintaining the high reliability and performance standards required in 5G networks; *ii*) Adaptive Anomaly Detection: By dynamically adjusting measurement frequency based on network conditions, the system efficiently detects performance degradations across end-to-end data flows. This adaptive approach optimizes resource utilization while ensuring timely detection of potential issues; *iii*) Scalable and Flexible Architecture: The modular design allows seamless application across various network architectures and traffic profiles, from simple test scenarios to complex, realistic 5G deployments. This flexibility ensures the system's relevance across diverse network implementations; *iv*) Standardized Logging Format: The implementation of a structured XML log format enhances data analysis capabilities and facilitates integration with existing network management tools. This standardization improves interoperability and simplifies the integration of our monitoring solution into existing network management ecosystems; and *v*) Comprehensive Dataset Generation: Through extensive simulations, we have created a valuable dataset of delay, throughput, and jitter metrics. This dataset will be made publicly available and contribute to the broader research community's efforts in 5G network optimization. In line with principles of open science and to foster collaborative advancement in 5G network monitoring, we have made our entire project codebase publicly available in a GitLab repository [10]. This includes the monitoring architecture implementation, simulation scripts, and analysis tools.

Future work will focus on extending our framework to incorporate machine learning techniques for predictive failure detection, exploring integration with emerging network slicing technologies, and conducting large-scale trials in operational 5G networks.

Acknowledgments

This work was partially supported by the Sectoral Scientific Research Commission (CSIC) under the Research and Development program ("5/6G Optical Network Convergence: a holistic view") and the CAP-UdelaR PhD scholarship program.

References

[1] Mamta Agiwal, Abhishek Roy, and Navrati Saxena. 2016. Next Generation 5G Wireless Networks: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials* 18, 3 (2016), 1617–1655. https://doi.org/10.1109/COMST.2016.2532458

[2] Sima Barzegar, Shaoxuan Wang, Marc Ruiz, Luis Velasco, Matias Richart, and Alberto Castro. 2023. Coordination of Radio Access and Optical Transport. In *2023 International Conference on Optical Network Design and Modeling (ONDM)*. 1–3.

[3] Alvaro Bernal, Matias Richart, Marc Ruiz, Alberto Castro, and Luis Velasco. 2020. Near real-time estimation of end-to-end performance in converged fixed-mobile networks. *Computer Communications* 150 (2020), 393–404. https://doi.org/10.1016/j.comcom.2019.11.052

[4] Gustavo Carneiro, Pedro Fortuna, and Manuel Ricardo. 2010. FlowMonitor - a network monitoring framework for the Network Simulator 3 (NS-3). In *2nd International ICST International Workshop on Network Simulation Tools (NSTOOLS)*. https://doi.org/10.4108/ICST.VALUETOOLS2009.7493

[5] Tao Chen, Marja Matinmikko, Xianfu Chen, Xuan Zhou, and Petri Ahokangas. 2015. Software defined mobile networks: concept, survey, and research directions. *IEEE Communications Magazine* 53, 11 (2015), 126–133.

[6] CTTC OpenSim Research Unit. 2024. *ns-3 LENA 5G: Network Simulator 3 LENA 5G Documentation.* Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Open Source. https://cttc-lena.gitlab.io/nr/nrmodule.pdf Accessed: 2024-07-03.

[7] Habtegebreil Haile, Karl-Johan Grinnemo, Simone Ferlin, Per Hurtig, and Anna Brunstrom. 2021. End-to-end congestion control approaches for high throughput and low delay in 4G/5G cellular networks. *Computer Networks* 186 (2021), 107692.

[8] Kostas Katsalis, Navid Nikaein, and Andy Edmonds. 2016. Multi-domain orchestration for nfv: Challenges and research directions. In *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*. IEEE, 189–195.

[9] K. Koutlia, B. Bojovic, Z. Ali, and S. Lagen. 2022. Calibration of the 5G-LENA System Level Simulator in 3GPP reference scenarios. *Simulation Modelling Practice and Theory (SIMPAT)* 119 (September 2022), 102580.

[10] Leandro Alfonso and Nicolás Rivoir. 2024. Monitoring Topologies Repository. https://gitlab.fing.edu.uy/mobile-optical-networks/monitoring_topologies. Accessed: 2024-07-03.

[11] Hongbo Lu, Gengchen Liu, Roberto Proietti, Vincent Squitieri, Kaiqi Zhang, Alberto Castro, Q. Jane Gu, Zhi Ding, and S. J. Ben Yoo. 2018. mmWave Beamforming using Photonic Signal Processing for Future 5G Mobile Systems, In Optical Fiber Communication Conference. *Optical Fiber Communication Conference*, M4J.3. https://doi.org/10.1364/OFC.2018.M4J.3

[12] N. Patriciello, S. Lagen, B. Bojovic, and L. Giupponi. 2019. An E2E Simulator for 5G NR Networks. *Simulation Modelling Practice and Theory (SIMPAT)* 96 (Nov. 2019), 101933.

[13] Nguyen Huu Phuoc Dai, Lourdes Ruiz, and Rajnai Zoltan. 2021. 5G revolution: Challenges and opportunities. In *2021 IEEE 21st International Symposium on Computational Intelligence and Informatics (CINTI)*. 00211–00216. https://doi.org/10.1109/CINTI53070.2021.9668550

[14] R. Proietti, H. Lu, G. Liu, A. Castro, M. Shamsabardeh, and S. J. B. Yoo. 2017. Experimental Demonstration of Elastic RF-Optical Networking (ERON) for 5G mm-wave Systems. In *2017 European Conference on Optical Communication (ECOC)*. 1–3. https://doi.org/10.1109/ECOC.2017.8346014

[15] M. Ruiz, M. Richart, A. Castro, and L. Velasco. 2020. End-To-End KPI Analysis in Converged Fixed-Mobile Networks. In *2020 22nd International Conference on Transparent Optical Networks (ICTON)*. 1–4. https://doi.org/10.1109/ICTON51198.2020.9203114

[16] F. Salahdine, T. Han, and N. Zhang. 2023. 5G, 6G, and Beyond: Recent advances and future challenges. *Annals of Telecommunications* 78 (2023), 525–549. https://doi.org/10.1007/s12243-022-00938-3

[17] William Stallings. 1998. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2.* Addison-Wesley Longman Publishing Co., Inc.

[18] Lee Thomason. 2024. TinyXML2: Simple, small, efficient, C++ XML parser. https://github.com/leethomason/tinyxml2. Accessed: 2024-07-12.

[19] Slavica Tomovic, Milica Pejanovic-Djurisic, and Igor Radusinovic. 2014. SDN based mobile networks: Concepts and benefits. *Wireless Personal Communications* 78 (2014), 1629–1644.

[20] Marcos Toscano, Federico Grunwald, Matías Richart, Javier Baliosian, Eduardo Grampín, and Alberto Castro. 2019. Machine Learning Aided Network Slicing. In *2019 21st International Conference on Transparent Optical Networks (ICTON)*. 1–4. https://doi.org/10.1109/ICTON.2019.8840141

[21] L. Velasco, A. Castro, A. Asensio, M. Ruiz, G. Liu, C. Qin, R. Proietti, and S. J. B. Yoo. 2017. Meeting the Requirements to Deploy Cloud RAN Over Optical Networks. *J. Opt. Commun. Netw.* 9, 3 (Mar 2017), B22–B32. https://doi.org/10.1364/JOCN.9.000B22

[22] Menglei Zhang, Marco Mezzavilla, Russell Ford, Sundeep Rangan, Shivendra Panwar, Evangelos Mellios, Di Kong, Andrew Nix, and Michele Zorzi. 2016. Transport layer performance in 5G mmWave cellular. In *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 730–735.