



**5TH GENERATION END-TO-END NETWORK, EXPERIMENTATION,  
SYSTEM INTEGRATION, AND SHOWCASING**

[H2020 - Grant Agreement No. 815178]

Deliverable D3.6

# Monitoring and Analytics (Release B)

**Editor** Ö. Alay, G. Caso (SRL)

**Contributors** SRL ( Simula Metropolitan Center For Digital Engineering), KAU (Karlstads Universitet), NCSRD ( National Center For Scientific Research “DEMOKRITOS”), LMI (L.M. Ericsson Limited), FON (FON Technology SL), ATOS (Atos Spain SA), UMA (Universidad De Malaga), HU (Humboldt University), INF (INFOLYSiS P.C.), FhG ( Fraunhofer Gesellschaft Zur Foerderung Der Angewandten Forschung E.V.), SHC (Space Hellas (Cyprus) Ltd), INT (Intel)

**Version** 1.0

**Date** April 05 2021

**Distribution** PUBLIC (PU)



## List of Authors

<b>SRL</b>	<b>SIMULA METROPOLITAN CENTER FOR DIGITAL ENGINEERING</b>
Ö. Alay, G. Caso	
<b>KAU</b>	<b>KARLSTADS UNIVERSITET</b>
A. Brunstrom, A. Rabitsch, M. Rajiullah, K.-J. Grinnemo, J. Karlsson	
<b>NCSRD</b>	<b>NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”</b>
T. Anagnostopoulos, G. Xilouris, H. Koumaras	
<b>LMI</b>	<b>L.M. ERICSSON LIMITED</b>
E. Aumayr, A-M Bosneag, J. McNamara	
<b>FON</b>	<b>FON TECHNOLOGY SL</b>
I. Pretel, I. Etxebarria	
<b>ATOS</b>	<b>ATOS SPAIN SA</b>
E. Jimeno	
<b>UMA</b>	<b>UNIVERSIDAD DE MALAGA</b>
A. Diaz-Zayas, B. Garcia	
<b>HU</b>	<b>HUMBOLDT UNIVERSITY</b>
L. Reichert	
<b>INF</b>	<b>INFOLYSIS P.C.</b>
C. Sakkas, A. Papaioannou, V. Koumaras	
<b>FhG</b>	<b>FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V.</b>
S. K. Rajaguru, A. Prakash, F. Eichhorn, M. Emmelmann, O. Keil	
<b>SHC</b>	<b>SPACE HELLAS (CYPRUS) LTD</b>
D. Lioprasitis, G. Gardikis	
<b>INT</b>	<b>INTEL</b>
V. Frasca	

## Disclaimer

---

The information, documentation and figures available in this deliverable are written by the 5GENESIS Consortium partners under EC co-financing (project H2020-ICT-815178) and do not necessarily reflect the view of the European Commission.

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The reader uses the information at his/her sole risk and liability.

---

## Copyright

---

Copyright © 2021 the 5GENESIS Consortium. All rights reserved.

The 5GENESIS Consortium consists of:

NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”	Greece
AIRBUS DS SLC	France
ATHONET SRL	Italy
ATOS SPAIN SA	Spain
AVANTI HYLAS 2 CYPRUS LIMITED	Cyprus
AYUNTAMIENTO DE MALAGA	Spain
COSMOTE KINITES TILEPIKOINONIES AE	Greece
EURECOM	France
FOGUS INNOVATIONS & SERVICES P.C.	Greece
FON TECHNOLOGY SL	Spain
FRAUNHOFER GESELLSCHAFT ZUR FOERDERUNG DER ANGEWANDTEN FORSCHUNG E.V.	Germany
IHP GMBH – INNOVATIONS FOR HIGH PERFORMANCE MICROELECTRONICS/LEIBNIZ-INSTITUT FUER INNOVATIVE MIKROELEKTRONIK	Germany
INFOLYSIS P.C.	Greece
INSTITUTO DE TELECOMUNICACOES	Portugal
INTEL DEUTSCHLAND GMBH	Germany
KARLSTADS UNIVERSITET	Sweden
L.M. ERICSSON LIMITED	Ireland
MARAN (UK) LIMITED	UK
MUNICIPALITY OF EGALEO	Greece
NEMERGENT SOLUTIONS S.L.	Spain
ONEACCESS	France
PRIMETEL PLC	Cyprus
RUNEL NGMT LTD	Israel
SIMULA RESEARCH LABORATORY AS	Norway
SPACE HELLAS (CYPRUS) LTD	Cyprus
TELEFONICA INVESTIGACION Y DESARROLLO SA	Spain
UNIVERSIDAD DE MALAGA	Spain
UNIVERSITAT POLITECNICA DE VALENCIA	Spain
UNIVERSITY OF SURREY	UK

This document may not be copied, reproduced or modified in whole or in part for any purpose without written permission from the 5GENESIS Consortium. In addition to such written permission to copy, reproduce or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

## Version History

---

Rev. N	Description	Author	Date
1.0	Release of D3.6	G. Caso, Ö. Alay (SRL)	31/3/2021

## LIST OF ACRONYMS

---

Acronym	Meaning
(R)MSE	(ROOT) MEAN SQUARE ERROR
AI	ARTIFICIAL INTELLIGENCE
AP	ACCESS POINT
API	APPLICATION PROGRAMMING INTERFACE
BE	BACKWARD ELIMINATION
BLER	BLOCK ERROR RATE
CA	CONSORTIUM AGREEMENT
CIB	CHARACTERISTICSC INFORMATION BASE
CPE	CUSTOMER PREMISES EQUIPMENT
CPU	CENTRAL PROCESSING UNIT
CQI	CHANNEL QUALITY INDICATOR
DC	DATA CENTER
e/gNB	EVOLVED / NEXT-GENERATION NODE B
E2E	END-TO-END
ELCM	EXPERIMENTAL LIFE CYCLE MANAGER
FTP	FILE TRANSFER PROTOCOL
GA	GRANT AGREEMENT
GUI	GRAPHICAL USER INTERFACE
HCI	HYBRID CLOUD INFRASTRUCTURE
HTTP(S)	HYPertext TRansfer PROTOCOL (SECURE)
IM	INFRASTRUCTURE MONITORING
IoT	INTERNET OF THINGS
IP	INTERNET PROTOCOL
JSON	JAVASCRIPT OBJECT NOTATION
KPI	KEY PERFORMANCE INDICATOR
LSTM	LONG SHORT-TERM MEMORY
M&A	MONITORING AND ANALYTICS
MAC	MEDIUM ACCESS CONTROL
MAD	MEDIAN ABSOLUTE DEVIATION
MAE	MEAN ABSOLUTE ERROR
MANO	MANAGEMENT AND ORCHESTRATION

MBB	MOBILE BROAD BAND
MCS	MODULATION AND CODING SCHEME
ML	MACHINE LEARNING
MONROE VN	MONROE VIRTUAL NODE
NFV(O)	NETWORK FUNCTION VIRTUALIZATION (ORCHESTRATOR)
NSI	NETWORK SLICE INSTANCE
O5GC	OPEN5GCORE
OLS	ORDINARY LEAST SQUARE
OSM	OPEN SOURCE MANO
PDCP	PACKET DATA CONVERGENCE PROTOCOL
PDSCH	PHYSICAL DOWNLINK SHARED CHANNEL
PIB	POLICY INFORMATION BASE
PM	PERFORMANCE MONITORING
QoS / QoE	QUALITY OF SERVICE / EXPERIENCE
RAM	RANDOM ACCESS MEMORY
RAN	RADIO ACCESS NETWORK
ReLu	RECTIFIED LINEAR UNIT
REST	REPRESENTATIONAL STATE TRANSFER
RFE	RECURSIVE FEATURE ELIMINATION
RLC	RADIO LINK CONTROL
RSRP / RSRQ	REFERENCE SIGNAL RECEIVED POWER / QUALITY
S-NSSAI	SINGLE-NETWORK SLICE SELECTION ASSISTANCE INFORMATION
SDN	SOFTWARE DEFINED NETWORK
SGD	STOCHASTIC GRADIENT DESCENT
SLA	SERVICE LEVEL AGREEMENT
SNMP	SIMPLE NETWORK MANAGEMENT PROTOCOL
SNR	SIGNAL TO NOISE RATIO
SVM	SUPPORT VECTOR MACHINE
TAP	TEST AUTOMATION PLATFORM
UE	USER EQUIPMENT
VIM	VIRTUAL INFRASTRUCTURE MANAGER
VM	VIRTUAL MACHINE
VNF	VIRTUAL NETWORK FUNCTION
WAN	WIDE AREA NETWORK

WIM	WAN INFRASTRUCTURE MANAGER
Xvfb	X VIRTUAL FRAME-BUFFER



## Executive Summary

---

This document describes the design and implementation of the 5GENESIS Monitoring & Analytics (M&A) framework in its Release B, developed within Task T3.3 of the project work plan. M&A Release B leverages and extends M&A Release A, which has been documented in the previous Deliverable D3.5 [1]. In particular, we present new features and enhancements introduced in this new Release compared to the Release A. We also report some examples of usage of the M&A framework, in order to showcase its integrated in the 5GENESIS Reference Architecture.

The instantiation of a M&A framework is crucial for 5G. In particular, this is due to the fact that the services provided by 5G systems have to comply with Service Level Agreements (SLAs), which state the end-to-end (E2E) performance that have to be guaranteed to end-users and verticals, leading to the need for careful management and monitoring of the instantiated resources. Therefore, a 5G M&A framework should consider both end-users' and operators' perspectives, aiming at satisfying and improving user's Quality of Service and Experience (QoS/QoE) and operator's management and operational costs [2][3].

The 5GENESIS M&A framework includes advanced Monitoring tools and both statistical and Machine Learning (ML)-based Analytics, and it is devoted to the collection and analysis of the heterogeneous data produced during the usage of the 5GENESIS Facility. Its main goal, within the project scope, is to verify the status of the infrastructure components during the execution of experiments for the validation of 5G Key Performance Indicators (KPIs), including the ones related to vertical use cases.

Following the design choices of Release A, the 5GENESIS M&A Release B is also designed and implemented in three main interoperable functional blocks:

- *Infrastructure Monitoring (IM)*, which focuses on the collection of data that synthesize the status of architectural components, e.g., end-user devices, radio access and networking systems, core network, and cloud and edge units;
- *Performance Monitoring (PM)*, which enables the active measure of E2E QoS/QoE KPIs;
- *Storage and Analytics*, which enables the management of large sets of heterogeneous data, and drives the discovery of hidden values, correlation, and causality among them.

The parallel use of IM/PM tools and Analytics functionalities enables a reliable assessment of KPIs, while also pinpointing possible issues that lead to performance losses, and ultimately triggering the use of improved policies and configurations in next experiment executions.

Both the design and implementation of the M&A framework have been carried out by considering the 5GENESIS Reference Architecture [4][5], as well as commonalities and peculiarities of the 5GENESIS platforms.

# Table of Contents

---

<b>LIST OF ACRONYMS .....</b>	<b>6</b>
<b>1. INTRODUCTION .....</b>	<b>13</b>
1.1. Purpose of the Document.....	13
1.2. Document Dependencies .....	13
1.3. Structure of the document .....	13
1.4. Target audience .....	14
<b>2. RELEASE A SUMMARY AND RELEASE B INTRODUCTION .....</b>	<b>15</b>
2.1. The 5GENESIS M&A Framework .....	15
2.2. From Release A to Release B .....	17
<b>3. MONITORING .....</b>	<b>20</b>
3.1. Prometheus for Infrastructure Monitoring.....	20
3.1.1. Main configuration and setup .....	20
3.1.2. Exporters for RAN and Core Network Monitoring .....	21
3.1.2.1. Amarisoft RAN Exporter .....	21
3.1.2.2. Open5GCore Exporter .....	22
3.1.3. Slice Monitoring.....	23
3.2. MONROE VN enhancements for Performance Monitoring .....	24
3.2.1. Browsertime5g .....	25
3.2.2. 360dash .....	26
3.3. Other Probes for Performance Monitoring .....	27
3.3.1. Remote Agents .....	27
3.3.2. PM Agents for Android devices .....	27
3.3.3. IxChariot.....	28
<b>4. STORAGE AND ANALYTICS .....</b>	<b>29</b>
4.1. InfluxDB.....	29
4.2. Analytics.....	29
4.2.1. Visualization Service .....	30
4.2.2. Data Handler Service .....	34
4.2.2.1. Time Series Synchronization .....	34
4.2.2.2. Anomaly Detection .....	35
4.2.3. Statistical Analysis Service .....	35
4.2.4. Linear Correlation Analysis Service .....	36

4.2.5. Feature Selection Service .....	36
4.2.6. KPI Prediction Service .....	37
4.3. APIs for result retrieving .....	37
4.3.1. Analytics service APIs .....	38
4.3.1.1. Data Handler Service .....	38
4.3.1.2. Correlation Service .....	39
4.3.1.3. Prediction Service .....	40
4.3.1.4. Statistical Analysis Service .....	41
4.3.1.5. Feature Selection Service .....	44
<b>5. M&amp;A INTEGRATION WITH OTHER COMPONENTS .....</b>	<b>45</b>
5.1. Integration with 5GENESIS Portal .....	45
5.2. Integration with 5GENESIS Slice Manager and Policy Engines.....	45
5.2.1. NEAT .....	45
5.2.2. APEX.....	47
5.3. Integration with 5GENESIS Security Analytics.....	47
5.3.1. Edge Cloud.....	48
5.3.1.1. Prometheus server, Node Exporter, Amarisoft Exporter .....	48
5.3.2. Core DC.....	48
5.3.2.1. InfluxDB .....	48
5.3.2.2. Grafana .....	49
5.3.2.3. Autoencoder .....	50
<b>6. TESTING AND VALIDATION.....</b>	<b>53</b>
6.1. Testing and Validation of the Analytics Services.....	53
6.1.1. Time Series Overview .....	53
6.1.2. Outlier Removal .....	53
6.1.3. Statistical Analysis.....	54
6.1.4. Correlation .....	55
6.1.5. Feature Selection .....	55
6.1.6. KPI Prediction.....	56
<b>7. CONCLUSION .....</b>	<b>58</b>
<b>REFERENCES.....</b>	<b>59</b>
<b>ANNEX 1 – ADDITIONAL INFRASTRUCTURE MONITORING TOOLS.....</b>	<b>61</b>
<b>ANNEX 2 – WIFI MANAGEMENT AND MONITORING .....</b>	<b>63</b>
<b>ANNEX 3 – IoT MANAGEMENT AND MONITORING .....</b>	<b>70</b>

ANNEX 4 – DATA ANONYMIZATION ..... 76

ANNEX 5 – MONROE VN ..... 85

# 1. INTRODUCTION

## 1.1. Purpose of the Document

In this document, we summarize the main activities carried out within Task T3.3 towards the design and implementation of the 5GENESIS Monitoring & Analytics (M&A) framework in its Release B. We aim at presenting new features and enhancements introduced in this new Release, compared to the Release A described in Deliverable D3.5 [1]. We further report some examples of usage of the framework, in order to showcase how it is integrated in the 5GENESIS Reference Architecture and used across the 5GENESIS platforms.

## 1.2. Document Dependencies

The table below summarizes the relevance of other 5GENESIS deliverables against the content of the present document.

id	Document title	Relevance
D3.5 [1]	5GENESIS Monitoring and Analytics (Release A)	The document describes M&A Release A.
D2.1 [4]	Requirements of the Facility	The document defines the requirements related to the features supported by the 5GENESIS Facility. M&A main features are also discussed.
D2.4 [5]	Final report on facility design and experimentation planning	The document defines the 5GENESIS Reference Architecture and lists its functional components. The integration of the M&A framework in the Reference Architecture is made explicit and discussed.
D6.1 [6]	Trials and Experimentations (Cycle 1)	The document reports the results of the 1 <sup>st</sup> Experimentation Cycle. The procedure for the KPI statistical validation, implemented as part of the Analytics functionalities, is defined and described in detail.

## 1.3. Structure of the document

Section 2 provides a summary on M&A Release A and introduces M&A Release B on a high level. Sections 3 and 4 describe the advances of M&A Release B in terms of Infrastructure and

Performance Monitoring functionalities, respectively. Storage and Analytics components are described in Section 5, while Section 6 describes M&A integration with other components of the 5GENESIS Reference Architecture. Section 7 reports M&A testing and validation examples, as well as some use cases of the framework. Conclusions are drawn in Section 8, while further details on specific M&A components and functionalities are given in the Annexes.

## 1.4. Target audience

The document targets the 5GENESIS Consortium, in order to provide a common overview of the M&A Release B architecture, components, and functionalities.

The document is also turned to external users and experimenters of the 5GENESIS Facility, in order to give them an overview on how to use the M&A framework and which functionalities can be expected within the 5GENESIS Facility.

## 2. RELEASE A SUMMARY AND RELEASE B INTRODUCTION

---

### 2.1. The 5GENESIS M&A Framework

5GENESIS targets the realization of a full-chain M&A framework for a reliable validation of 5G KPIs [1]. The framework enables the analysis of experimental data collected by dedicated monitoring probes during the usage of the 5GENESIS Facility. This in turn allows to pinpoint the interdependencies between network configurations, scenarios and environmental conditions, and QoS and QoE KPIs, ultimately leading to the derivation of optimized management policies for further improvement of users', verticals', and operators' performance.

The 5GENESIS M&A framework includes several Monitoring tools and both statistical and ML-based Analytics. It is formed by three main blocks:

- *Infrastructure Monitoring (IM)*, which focuses on the collection of data on the status of infrastructure components, e.g., User Equipment (UE), radio access and core networks, Software Defined Network / Network Function Virtualization (SDN/NFV) environments, and distributed edge units;
- *Performance Monitoring (PM)*, which is devoted to the active measure of E2E QoS/QoE KPIs. These include traditional indicators, such as throughput and latency, but also other indicators tailored on specific use cases and applications (e.g., mission critical services and massive communications);
- *Storage and Analytics*, which enables the efficient management of large amounts of heterogeneous data, and drives the discovery of hidden values, correlation, and causality among them.

Among others, the 5GENESIS M&A framework aims at providing the following Analytics functionalities:

- 1) *KPI Validation*, i.e., the execution of the KPI statistical analysis defined in 5GENESIS for validating a KPI [6];
- 2) *Time series management*, which allows to coherently merge the data coming from different probes, in order to perform further analyses. In a M&A system, this task is needed for several reasons. First, different sampling rates might be used by different probes. For example, QoS/QoE KPIs might be collected at higher sampling rates as compared to infrastructure data. Second, the probes might be not perfectly synchronized. Hence, time synchronization can be applied when the time series collected from different probes present similar sampling rates, while interpolation better suits situations where the probes use different sampling rates;
- 3) *Outlier detection*, in order to eliminate data obtained under incorrect functioning of the probes, which may negatively affect the analyses;
- 4) *Feature selection*, which allows to simplify the analyses by eliminating some of the collected parameters. As a matter of fact, 5G networks include a huge number of components. Hence, using ML and Artificial Intelligence (AI) approaches for network management and optimization could be challenged by the large amount of data that

can be potentially collected; some of these data might be not useful and could negatively affect the analyses. Hence, feature selection algorithms can be used to remove redundant features, making the next analyses computationally simpler and faster. In general, feature selection allows to train ML algorithms faster, reduce model complexity and overfitting, and improve model accuracy;

- 5) *Correlation analysis*, which allows to highlight how system configurations and network conditions, collected via IM probes, are correlated and affect QoS/QoE KPIs, collected via PM tools. Revealing the correlation between IM and PM parameters allows to improve network management and derive better configuration policies for assuring SLAs. Lack of correlation between parameters which are known to have dependencies is also a key indicator for pinpointing system malfunctioning and can help trigger needed fixes;
- 6) *KPI prediction*, which allows to build a model and estimate QoS/QoE KPIs by looking at other parameters, collected under different circumstances and scenarios. Being able to accurately predict a KPI would enable better network planning and management.

The M&A framework spans across all layers of the 5GENESIS Reference Architecture, from Infrastructure to Coordination, via Management and Orchestration (MANO). In particular, IM and PM probes mainly lie at the Infrastructure layer, in order to fulfill the requirement of tracking the status of components and functions, thus collecting large amounts of heterogeneous parameters. Then, a management instance of the Monitoring is placed at the MANO layer, so that the parameters scraped from the infrastructure components (i.e., physical and virtual hosts) are redirected to a centralized collector, e.g., a Prometheus<sup>1</sup> server. The Coordination layer hosts the storage utilities and the Analytics functionalities. The Analytics results are shown in a dedicated visualization utility.

As anticipated above and depicted in Figure 1, the M&A framework comprises of IM, PM, Storage, and Analytics. The main connection point with the 5GENESIS Reference Architecture is the Experimental Life Cycle Manager (ELCM), developed in Task T3.8, whose main functionalities are the scheduling, composition, and supervision of experiments in the platforms, as detailed in Deliverable D3.15 [7].

In each 5GENESIS platform, the ELCM activates on-demand IM/PM probes via the *Activation Plugins*, in order to start monitoring the components involved in a specific experiment, e.g., the components of a network slice, while actively running the experiment. The ELCM also automatizes both formatting and long-term storage of the data collected during the experiment execution, via so-called *Results Collectors*. Within 5GENESIS, the Keysight Test Automation Platform (TAP) software<sup>2</sup> deals with most of the ELCM operations. Therefore, the 5GENESIS *Activation Plugins* are *TAP Plugins*, while the *Result Collectors* are *TAP Result Listeners*.

---

<sup>1</sup> <https://prometheus.io>, Accessed on: March 2021.

<sup>2</sup> <https://www.keysight.com/en/pc-2873415/test-automation-platform-tap>, Accessed on: March 2021.



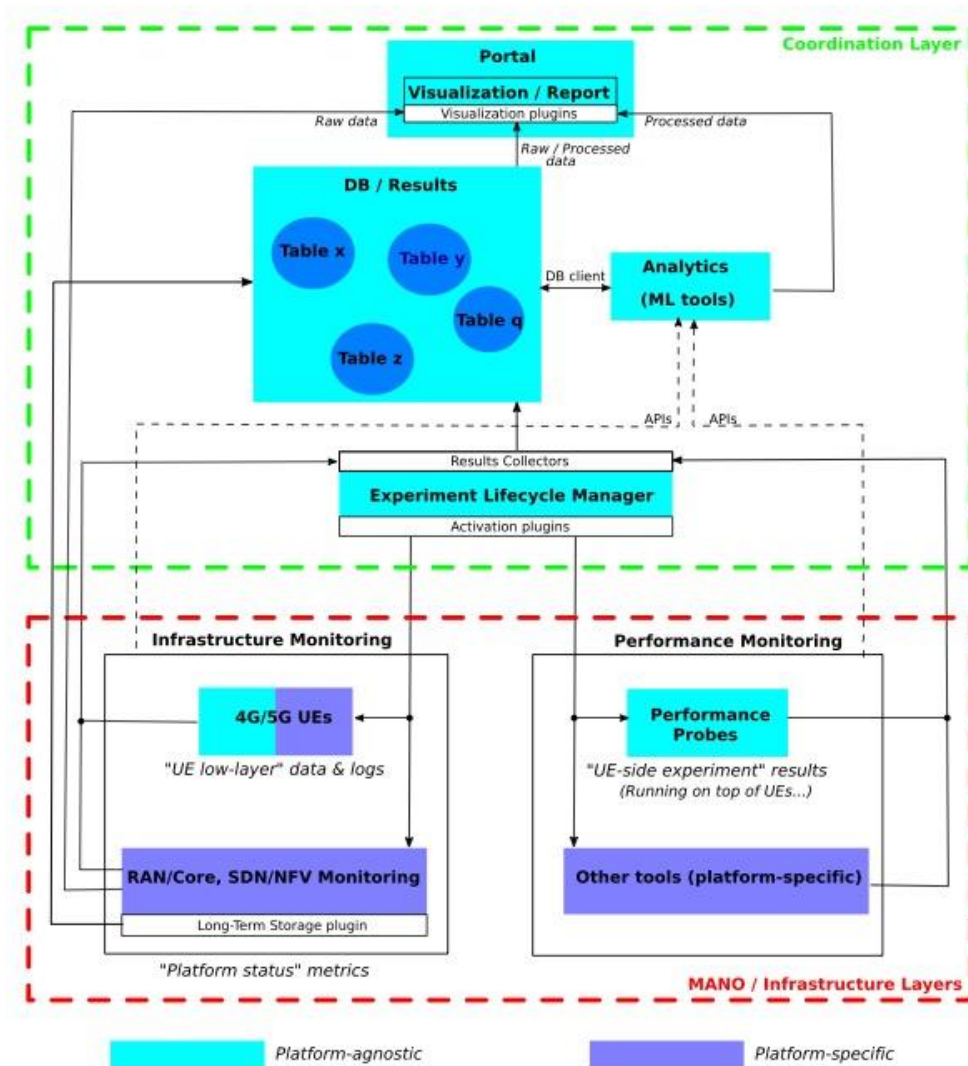


Figure 1. 5GENESIS M&amp;A Framework.

## 2.2. From Release A to Release B

This Section summarizes the main components developed and integrated in M&A Release A, and introduces the add-ons and upgrades included in M&A Release B. These latter will be described in detail in the following dedicated sections.

**IM.** With regards to the IM functionality, the 5GENESIS platforms use Prometheus as the main IM tool, adopting the configurations already described in [1] and also summarized in Section 3.1.1. Prometheus covers the monitoring of physical hosts, physical/virtual network functions, SDN/NFV instances, as well as RAN, Core network, and edge units.

In Release A, the *Node Exporter* has been used in order to collect low-layer metrics on the status of physical units involved in the experiments, e.g., in terms of memory load, Central Processing Unit (CPU) and disk utilization, and traffic sent/received on the available network interfaces, among others. In M&A Release B, dedicated *Simple Network Management Protocol (SNMP)*

*Exporters* have been also developed and flanked to the Node Exporter, in order to monitor the status of vendor-specific components used in the 5GENESIS Facility, e.g., the Amarisoft RAN and Open5GCore<sup>3</sup> (O5GC) components used in some of the platforms (see Section 3.1.2). Moreover, Prometheus is also used for enabling the so-called *Slice Monitoring*, as detailed in Section 3.1.3.

Considering the monitoring of UEs, specific tools are required for the collection of radio parameters experienced upon connection to the e/gNBs, such as, Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), Signal to Noise Ratio (SNR), and Channel Quality Indicator (CQI). The same tools can be also used for collecting information on transmission settings, such as, the adopted Modulation and Coding Scheme (MCS). The monitoring of Android-based 4G and 5G UEs used in the 5GENESIS Facility is executed via a dedicated application, referred to as Android Resource Agent, which has been developed during the first Project cycle for 4G UEs (Release A), and being upgraded in the second Project cycle for 5G UEs (Release B).

**PM.** With regards to the PM probes, several platform-agnostic instruments have been designed and developed in both Release A and Release B. These are the MONROE Virtual Node (VN) and the so-called Remote Agents and Android Agents. In their Release A, these probes have been developed aiming at their use in experiments for the validation of latency and throughput KPIs, as also reported in Deliverables D6.1 [6] and D6.2 [8]. In M&A Release B, MONROE VN has been enhanced, improving its integration in the M&A framework but also upgrading its functionalities aiming at enabling the measurement of application-layer KPIs, e.g., web browsing and video streaming performance (Section 3.2). The Remote and Android Agents have maintained their primary functionalities but are configured more efficiently (Section 3.3).

**Storage and Analytics.** For both M&A Release A and Release B, the 5GENESIS Consortium has agreed on the use of InfluxDB<sup>4</sup> as the common tool for the creation of platform-specific instances of a long-term storage utility. InfluxDB is the open-source storage engine provided within the InfluxData framework, and handles in particular time series data (Section 4.1).

With regards to the Analytics component, it went through a significant enhancement from Release A to Release B. In the first Release, the Analytics functionalities have been provided as interconnected Python scripts, covering all the analysis steps, i.e., a) data retrieval from an InfluxDB database (executed via the *InfluxDB-Python client* [9]), b) data preprocessing and management (e.g., time series synchronization and outlier detection/removal), and c) specific statistical and ML functions (e.g., KPI statistical validation and prediction).

In Release B, the Analytics component has been reconfigured to simplify both its integration in the 5GENESIS platforms and its usage by the experimenters. In particular, Analytics is now provided as a collection of micro-services developed as Docker<sup>5</sup> containers. As detailed in Section 4.2, each container is dedicated to a specific functionality (e.g., data retrieval and management, KPI statistical validation, correlation analysis, and so on), and they can be used

---

<sup>3</sup> <https://www.open5gcore.org>, Accessed on: March 2021.

<sup>4</sup> <https://www.influxdata.com/products/influxdb-overview/>, Accessed on: March 2021.

<sup>5</sup> <https://www.docker.com/>, Accessed on: March 2021.

standalone via Representational State Transfer (REST) Application Programming Interfaces (APIs) or by means of a visualization container that gives parallel access to all the other containers through a Graphical User Interface (GUI). On the one hand, the REST APIs have been developed via Flask<sup>6</sup>, so that the code running inside each container is a web application easily accessed from remote (e.g., via a web browser). On the other hand, the GUI is built by using the DASH<sup>7</sup> visualization library. Finally, as for Release A, Pandas<sup>8</sup>, NumPy<sup>9</sup>, and SciKit-Learn<sup>10</sup> libraries are used for data processing and ML functionalities. The source code of implemented Analytics algorithms is part of the Open5GENESIS Suite.<sup>11</sup>

Besides the above developments, in this document we also provide a description of some of the most relevant integrations of M&A with other components of the 5GENESIS Reference Architecture, which target both platform-agnostic and platform-specific use cases. In particular, we document the activities towards:

- The access to Analytics via the 5GENESIS Portal, described in Section 5.1;
- The development of M&A solutions for automatic network configuration and optimization, i.e., *prescriptive analytics* methods. These developments aim at showcasing possible solutions for ML/AI-driven configuration and optimization of 5G systems. Under investigation in Athens and Surrey platforms, such solutions leverage the combination of M&A and Slice Manager [10] functionalities with so-called *policy engines*, i.e., NEAT and APEX. More details are provided in Section 5.2;
- The integration of M&A with the Security Analytics framework, realized in the Limassol platform and enabling deep learning-based anomaly detection (Section 5.3);
- The instantiation of M&A components for the collection of network logs, and for management and monitoring of non-3GPP access components, such as WiFi Access Points (APs). These activities have been performed in Berlin and Surrey platforms. More details are provided in Annexes 1 and 2;
- The instantiation of M&A solutions for management and monitoring of Internet of Things (IoT) access components and devices. This activity is related to the Surrey platform, and aims at demonstrating 5G massive connectivity scenarios. More details are provided in Annex 3;
- The introduction of methodologies for data anonymization. Although the M&A framework aims at specifically supporting dedicated experimentation for KPIs measurement, analysis, and validation, its usage can be also envisioned in experiments involving real users, e.g., the showcasing of 5G functionalities during large-scale events. In this case, data anonymization may be needed in order to comply with European and national privacy regulations. Therefore, data anonymization techniques have been studied and tested, towards their adoption in 5GENESIS large-scale demonstrations, e.g., the Festival of Lights 2021 targeted by the Berlin platform. More details are provided in Annex 4.

---

<sup>6</sup> <https://flask.palletsprojects.com/en/1.1.x/>, Accessed on: March 2021.

<sup>7</sup> <https://plotly.com/dash/>, Accessed on: March 2021.

<sup>8</sup> <https://pandas.pydata.org/>, Accessed on: March 2021.

<sup>9</sup> <https://numpy.org/>, Accessed on: March 2021.

<sup>10</sup> <https://scikit-learn.org/>, Accessed on: March 2021.

<sup>11</sup> <https://github.com/5genesis>, Accessed on: March 2021.

## 3. MONITORING

---

### 3.1. Prometheus for Infrastructure Monitoring

#### 3.1.1. Main configuration and setup

The main IM functionalities in the 5GENESIS platforms are based on Prometheus, and include four components: a) a main server, where both monitoring system and time series database are deployed; b) the alert manager, for creating and managing monitoring alarms; c) the Node Exporter<sup>12</sup>, for retrieving low-layer metrics from the monitored network components (also, *targets*); d) a Grafana<sup>13</sup> server, which makes it possible to visualize the monitored data at runtime. Apart from these main components, several extensions such as libraries, language client, and the Pushgateway can be used, leveraging the developments in the Prometheus repository.<sup>14</sup> Software extensions supporting the implementation of more agents that monitor heterogeneous network components and resources using available Prometheus exporters is also available online.<sup>15</sup>

With regards to the monitoring sever, a global yaml<sup>16</sup> configuration file allows to modify the default configuration context and enables the monitoring of multiple targets via the exporters deployed in the infrastructure. The alert manager is configured during runtime when the target endpoints are registered. General alarm rules can be set up during the configuration phase and/or reloaded at runtime.

The Prometheus server provides an endpoint to interact and configure the monitoring of several parameters and the management of targets. After the server has been deployed and is running in the infrastructure, several configurations can be executed:

- *Add monitoring targets*: multiple exporters can be added to the monitoring system by configuring the targets in the Prometheus server. The exposed metrics will be labelled with the parameters configured in the registration process, for being later able to filter metrics and operations, based on those parameters;
- *Delete monitoring targets*: after the finalization of the monitoring process, a target can be deleted from the system. When a target is deleted, the system stops providing the metrics related to that target to the server.
- *Add defined rules*: the system is able to create alarms based on predefined rules. Such rules are configured based on the monitored metrics and associated to the parameters defined in the targets. A rule can include an internal logic, based on one or more elements that at a given point will trigger a message of the active rule by the alert manager instance.

---

<sup>12</sup> [https://github.com/prometheus/node\\_exporter](https://github.com/prometheus/node_exporter), Accessed on: March 2021.

<sup>13</sup> <https://grafana.com>, Accessed on: March 2021.

<sup>14</sup> <https://github.com/prometheus/>, Accessed on: March 2021.

<sup>15</sup> <https://prometheus.io/docs/instrumenting/exporters/>, Accessed on: March 2021.

<sup>16</sup> <https://yaml.org>, Accessed on: March 2021.

Once installation and configuration are performed, the system answers to the requests made by registered targets and defined alerts. Within the 5GENESIS Reference Architecture, the ELCM supports possible on-demand activation of exporters on specific targets, and redirects the collected data towards the InfluxDB database.

### 3.1.2. Exporters for RAN and Core Network Monitoring

Besides the usage of the general-purpose Node Exporter, dedicated Prometheus exporters have been also developed by 5GENESIS for monitoring specific components used across 5GENESIS platforms. In the following, we document the development and usage of the Amarisoft Exporter, dedicated to the monitoring of Amarisoft RAN nodes, and of the Open5GCore (O5GC) Exporter, that collects metrics from the O5GC used in the Berlin platform.

#### 3.1.2.1. Amarisoft RAN Exporter

The Amarisoft Exporter is a Prometheus exporter for Amarisoft RAN metrics written in Go. It gathers RAN metrics using the Amarisoft API and makes them accessible under a dedicated Hypertext Transfer Protocol (HTTP) endpoint. As depicted in Figure 2, the Prometheus server scrapes metrics from the Amarisoft Exporter through the exposed endpoint.

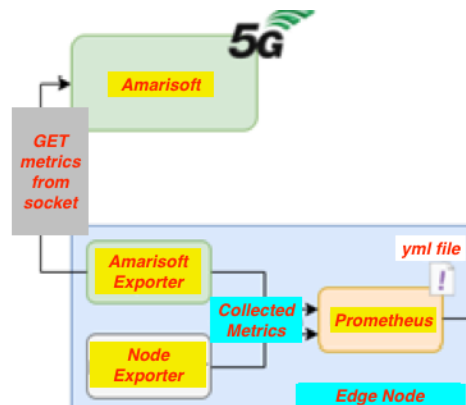


Figure 2. Integration and usage of the Amarisoft Exporter (in parallel to the Node Exporter).

The Amarisoft Exporter functions by opening a socket connection to the Amarisoft API. Then, it sends a “stats” message, parses the replies into metric fields, and exposes them under a “/metrics” endpoint in Prometheus format. The exported metrics are reported in Table 1.

Table 1. Metrics collected by the Amarisoft Exporter.

Metric name	Description
dl_bitrate	Downlink bitrate in bits per second
ul_bitrate	Uplink bitrate in bits per second
dl_txok	Number of successfully transmitted transport blocks
ul_rxok	Number of received uplink transport blocks without CRC error
rx_sample_rate	CPU consumption for reception chain (RX) in Million samples per sec
tx_sample_rate	CPU consumption for transmission chain (TX) in Million samples per sec

rx_cpu_time	CPU consumption (percentage) for RX
tx_cpu_time	CPU consumption (percentage) for TX
rxtx_delay_min	TX-RX delay min *
rxtx_delay_max	TX-RX delay max *
rxtx_delay_avg	TX-RX delay average *

\* RX/TX delay is the remaining time before data are processed and the time they must be sent in downlink.

### 3.1.2.2. Open5GCore (O5GC) Exporter

The Fraunhofer FOKUS's O5GC implements a monitoring mechanism for its Virtual Network Functions (VNFs) to expose relevant metrics. In addition to the Zabbix<sup>17</sup> support presented in Deliverable D3.5, O5GC monitoring component was also extended to support the Prometheus system, in order to align the Berlin platform with the other 5GENESIS platforms. It supports the scraping of gathered metrics via HTTP in a compatible format. As is common practice, the metrics can be found on the given host on a preconfigured port at the “/metrics” url.

Currently, the O5GC Exporter exposes metrics regarding the memory usage of the VNFs, e.g., allocated bytes and memory pool usage, but more metrics are planned and can be easily added. Figure 3 shows an example of Grafana dashboard with some of the memory-related metrics monitored across O5GC's VNFs.



<sup>17</sup> <https://www.zabbix.com>, Accessed on: March 2021.





Figure 3. Excerpts of Grafana dashboard showing O5GC metrics.

### 3.1.3. Slice Monitoring

The Release B of the 5GENESIS Slice Manager includes a monitoring, visualization, and alerting toolkit, which is responsible for tracking the status of components and services that are part of the instantiated slices. In order to achieve this, the 5GENESIS Slice Manager is capitalizing on Prometheus and Grafana for creating the slice monitoring module, which is packaged and delivered as part of the micro-service architecture.

Prometheus is used for scraping, storing, and organizing metrics from both physical and virtual components and services of the underlying infrastructure that have been instantiated and/or configured to be part of a deployed slice. Prometheus collects data from the following sources:

- The Node Exporter, which collects health-related metrics of the VNFs that have been created as part of a slice. In addition, it tracks generic slice information, such as the number and the status of the slices;
- Monitoring tools or exporters that are implemented from various components of the underlying infrastructure. In the current implementation, the Slice Manager collects and exposes metrics from the Virtual Infrastructure Manager (VIM), the Network Function Virtualization Orchestrator (NFVO) and the Wide Area Network (WAN) Infrastructure Manager (WIM), such as the network traffic, bandwidth utilization, VNF health and performance, CPU, memory and disk usage, etc.

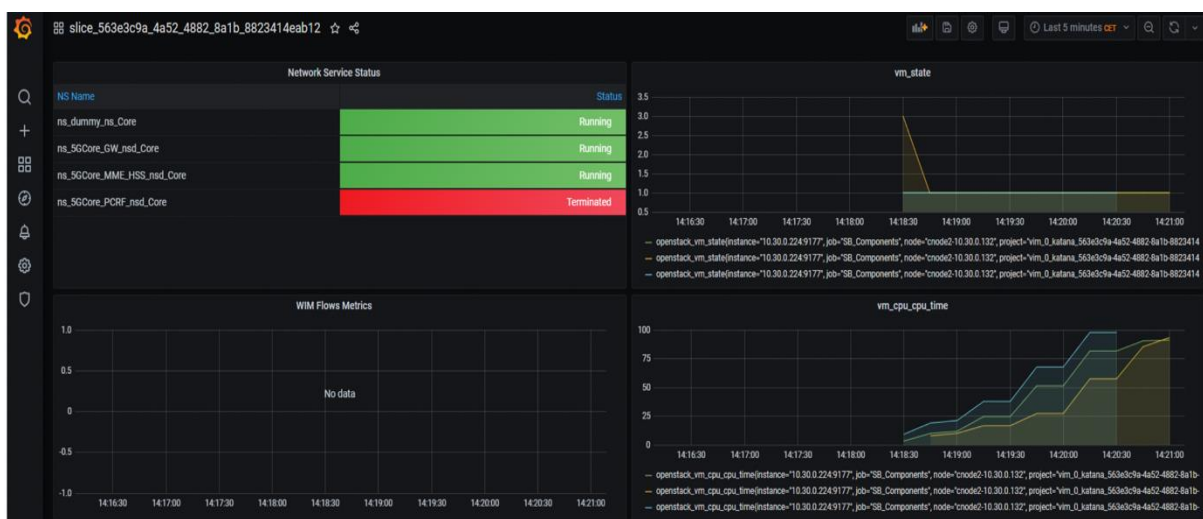




Figure 4. Excerpts of the Grafana dashboard showing an example of Slice Monitoring.

Figure 4 provides a snapshot of the Grafana Dashboard that visualizes metrics collected for a deployed slice. The Slice Manager exploits the Prometheus functionalities in order to expose the collected metrics through a specific API. This information is available to be collected by any other monitoring tool, including another higher-level Prometheus server in a hierarchical federation. This allows the slice monitoring module to be integrated with the IM tools of each 5GENESIS platform. Full description of this functionality and its implementation are included in Deliverable D3.4, which focuses on Slice Management (Release B).

### 3.2. MONROE Virtual Node (VN) enhancements for Performance Monitoring

To carry out the experiments defined in 5GENESIS, we have designed a number of MONROE probes. The ping and throughput containers have already been described in D3.5. These two containers first came out as a part of Release A. In the Release A version of the throughput container, the output contains nested JavaScript Object Notation (json<sup>18</sup>), which is problematic to automatically store at the InfluxDB in the ELCM backend. Therefore, in support of the generalized support for MONROE VN in 5GENESIS Release B, we have updated the throughput container such that it flattens the json at the output. An example of the updated throughput container output is included in Annex 5, where the integration and usage of MONROE VN in the Berlin platform is also reported as a reference example.

Furthermore, two new containers called browsertime5g<sup>19</sup> and 360dash<sup>20</sup> have been designed for measuring web browsing and 360-degree video streaming performance from 5GENESIS platforms. These new containers are further described below.

<sup>18</sup> <https://www.json.org/>, Accessed on: March 2021.

<sup>19</sup> <https://github.com/5genesis/monroe-experiments/tree/ReleaseB/browsertime5g>, Accessed on: March 2021.

<sup>20</sup> <https://github.com/5genesis/monroe-experiments/tree/ReleaseB/360dash>, Accessed on: March 2021.



### 3.2.1. Browsertime5g

Browsertime5g is based on browsertime<sup>21</sup> but specifically engineered to run from a Monroe probe. We configured Browsertime5g to mimic a mobile device browser (by setting both the screen resolution and the user-agent accordingly) to retrieve the mobile versions of the visited pages. With it, we direct the browser to load a page and, at the end of page rendering, it executes a custom JavaScript script to collect a large number of metrics. X virtual frame-buffer<sup>22</sup> (Xvfb) emulates a display to let the browsers actually render the web pages.

Browsertime5g provides a configurable experiment template to enable web measurements. We can configure each measurement by controlling (i) the network to test (Ethernet, or a Mobile Broad Band (MBB) interface), (ii) the browser (Firefox 83 or Chrome 85.0.4183.83), and (iii) the web protocol (HTTPS [11], HTTP/2.0 [12], or HTTP3 [13]). A combination of these parameters builds an experiment setup. Browsertime5g can be configured to visit any number of pages. Given a network to test, Browsertime5g shuffles the order of pages to visit. It visits each page with every browser and protocol combination, in a random order. Every visit is independent.

For each page visit Browsertime5g uses Selenium<sup>23</sup> to start a browser (Firefox/Chrome). It also starts FFMPEG<sup>24</sup> to record a video session to record the page download. Once the page is loaded in the browser, Browsertime5g executes a custom java script to collect a large number of metrics such as pageloadtime [14], Rum SpeedIndex<sup>25</sup>, first paint etc. It also collects a HTTP Archive<sup>26</sup> file that saves all request/response on that page. Next, the FFMpeg is stopped and the video is analyzed to determine several visual metrics, such as Speed Index<sup>27</sup>.

The default input values are:

```
{
  "guid": "no.guid.in.config.file", # Should be overridden by scheduler
  "url": "http://193.10.227.25/test/1000M.zip",
  "size": 3*1024, # The maximum size in Kbytes to download
  "time": 3600, # The maximum time in seconds for a download
  "zmqport": "tcp://172.17.0.1:5556",
  "modem_metadata_topic": "MONROE.META.DEVICE.MODEM",
  "dataversion": 1,
  "dataid": "5GENESIS.EXP.HEADLESS.BROWSERTIME",
  "nodeid": "fake.nodeid",
  "meta_grace": 120, # Grace period to wait for interface metadata
  "exp_grace": 120, # Grace period before killing experiment
}
```

<sup>21</sup> <https://github.com/sitespeedio/browsertime>, Accessed on: March 2021.

<sup>22</sup> <https://www.x.org/releases/X11R7.6/doc/man/man1/Xvfb.1.xhtml>, Accessed on: March 2021.

<sup>23</sup> <https://www.selenium.dev/selenium/docs/api/javascript/index.html>, Accessed on: March 2021.

<sup>24</sup> <https://www.ffmpeg.org>, Accessed on: March 2021.

<sup>25</sup> <https://github.com/WPO-Foundation/RUM-SpeedIndex>, Accessed on: March 2021.

<sup>26</sup> <http://www.softwareishard.com/blog/har-12-spec/>, Accessed on: March 2021.

<sup>27</sup> <https://www.keycdn.com/blog/speed-index>, Accessed on: March 2021.

```

    "ifup_interval_check": 6, # Interval to check if interface is up
    "time_between_experiments": 5,
    "verbosity": 2, # 0 = "Mute", 1=error, 2=Information, 3=verbose
    "resultdir": "/monroe/results/",
    "modeminterfacename": "InternalInterface",
    "urls": ['www.instagram.com'],
    "http_protocols":["h1s","h2","http3"],
    "browsers":["chrome","firefox"],
    "iterations": 1,
    "allowed_interfaces": ["eth0"], # Interface to run the experiment on
    "interfaces_without_metadata": ["eth0"] # Manual metadata on these IF
  }

```

Using the above inputs (default), the `browstime5g` will download the Instagram page over the selected source interface with different combinations of three different http protocols each time. `Browstime5g` outputs a single json file combining all the metrics and a http archive information. All fields in the output are flattened to be compatible with InfluxDB. An example of produced output that corresponds to visiting Instagram using google-chrome with HTTP3 is provided in Annex 5.

### 3.2.2. 360dash

360dash<sup>28</sup> is based on dashc and the work done in [15], but specifically engineered to run from a Monroe probe. We configured 360dash to stream a dash video/url for a specified amount of time (or until the video is finished).

360dash provides a configurable experiment template to enable dash video measurements. We can configure each measurement by controlling (i) the network to test (Ethernet, or an MBB interface), (ii) the video stream to use and optionally (iii) the duration of the experiment. A combination of these parameters builds an experiment setup.

While streaming, 360dash records multiple metrics related to the video stream such as current buffer level, number of stalls etc.

The default input values are:

```

{
    "zmqport": "tcp://172.17.0.1:5556",
    "guid": "fake.guid", # Need to be overridden
    "nodeid": "virtual",
    "metadata_topic": "MONROE.META",
    "dataid": "5GENESIS.EXP.DASHC",
    "dataversion": 1,

```

<sup>28</sup> <https://github.com/5genesis/monroe-experiments/tree/ReleaseB/360dash>, Accessed on: March 2021.

```

"verbosity": 2, # 0 = "Mute", 1=error, 2=Information, 3=verbose
"resultdir": "/monroe/results/",
"flatten_delimiter": '.',
"allowed_interfaces": ["ens160", "ens192", "eth0"],
"url": "http://panoplay.duckdns.org/abs/stream/out.mpd",
"duration": 900 # If 0 or less wait until video is done
}

```

Using the above inputs (default), the 360dash will download an example 360 video over each allowed interface in sequence (if the interface is available). All fields in the output are flattened to be compatible with InfluxDB. An example of produced output with duration set to 10 seconds, all other values set to the default, is provided in Annex 5.

## 3.3. Other Probes for Performance Monitoring

### 3.3.1. Remote Agents

The iPerf and Ping remote agents presented in Deliverable D3.5 are also available as part of Release B. Their functionalities have not changed; however, several bugs have been fixed and, in the case of the Ping agent, the exposed REST API has been modified in order to improve usability. Table 2 shows the updated endpoints on the Ping Agents.

**Table 2. Ping Agent endpoints.**

Endpoint	Method	Description
/Ping/<address>	GET	Starts a Ping instance using the address specified. Accepts the following URL parameters: interval, size, ttl.
/Close	GET	Stops the running Ping instance.
/LastJsonResult	GET	Retrieves the results from the previous execution.
/StartDateTime	GET	Returns the time and date when the previous instance of Ping was started.
/IsRunning	GET	Returns a message indicating if there is an active Ping instance.

### 3.3.2. PM Agents for Android devices

As for the Remote Agents, the Android applications presented in Deliverable D3.5 have received continued support, fixing bugs detected during the previous experimentation cycle and further refining their functionalities.

In order to support testing outside of the lab environment, all agents are now able to write the recorded measurements to text files that are saved in the device. These files can later be

analyzed offline in the 5GENESIS facilities, uploading these results in the main database and making them available to the Analytics module.

### 3.3.3. IxChariot

IxChariot is a traffic generator tool that focuses on network performance assessment and application simulation.

To run the networking tests, IxChariot uses software agents called endpoints, that can be installed in various platforms (Linux, Windows, Android and iOS), to allow for the evaluation of the network between them. The platform offers a variety of traffic patterns ranging from basic TCP or UDP flow groups, to complex application mixes derived from real world traffic scenarios such as video streaming services (YouTube, Netflix) or voice call services (Skype).

Each testing session aims to benchmark network performance by measuring specific KPIs such as RTT, One Way Delay, Jitter, Packet Loss and Throughput based on the traffic patterns defined between the user groups. Additional options include the use of QoS policies per endpoint, along with the simulation of multiple user groups. Testing sessions are defined, configured and ran by the web application that runs as part of the IxChariot server, where the experimenter can register the endpoints, test options and traffic patterns, as shown below on Figure 5.

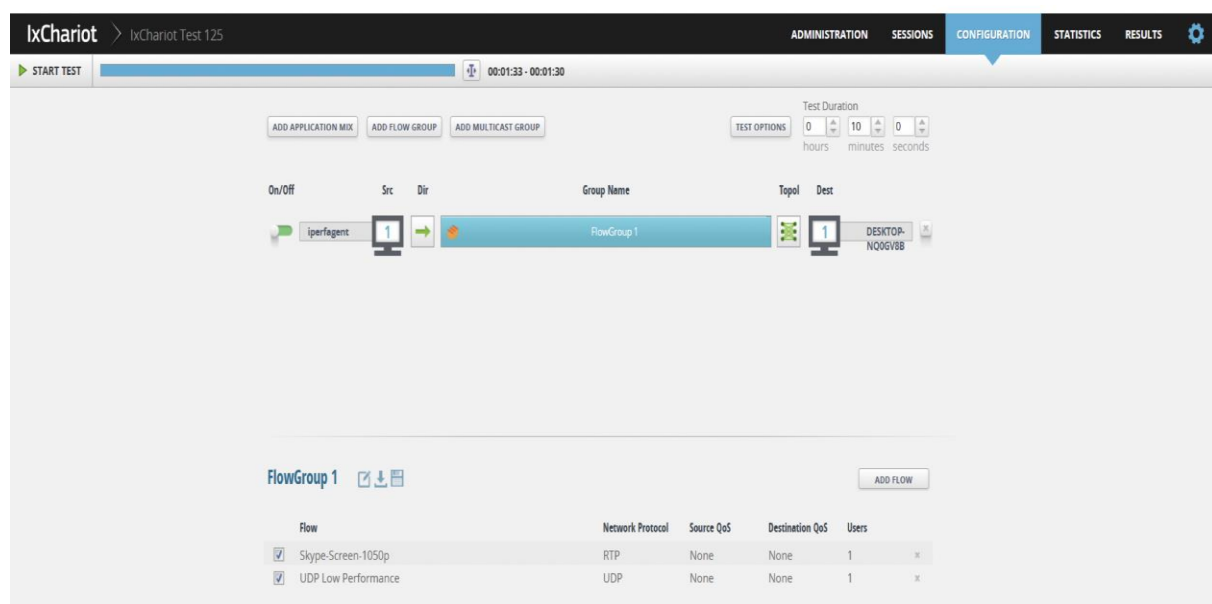


Figure 5. IxChariot interface for experiment composition and execution.

Results can be exported either on csv format or as part of a detailed report with custom graphs that visualise target KPIs for each test execution.

For automating testing processes, IxChariot server offers an API endpoint.

## 4. STORAGE AND ANALYTICS

### 4.1. InfluxDB

In M&A Release B, we have maintained InfluxDB for long-term storage of experimental data. The InfluxDB Result Listener allows the ELCM to retrieve IM/PM metrics in the form of time series, and automatically redirect them towards dedicated InfluxDB measurement tables within a platform-specific database instance.

Analytics can query and read the data needed for the analyses via the Data Handler service described in Section 4.2.2, which adopts the APIs provided by the open-source InfluxDB-Python client for accessing a platform-specific InfluxDB instance.

### 4.2. Analytics

In order to tackle the analysis of health and performance of the 5GENESIS platforms during the experiment executions, we have implemented a variety of Analytics methods. In M&A Release B, such methods have been developed as micro-services, using Docker containers for the encapsulation of functionalities into services, and Flask for REST inter-service communication. Figure 6 gives an overview of the implemented containers and their functions. These include anomaly detection and correlation analysis for health monitoring purposes (i.e., is the experiment running as expected?), and KPI prediction for performance analysis (e.g., to support potential performance improvement of certain network elements).

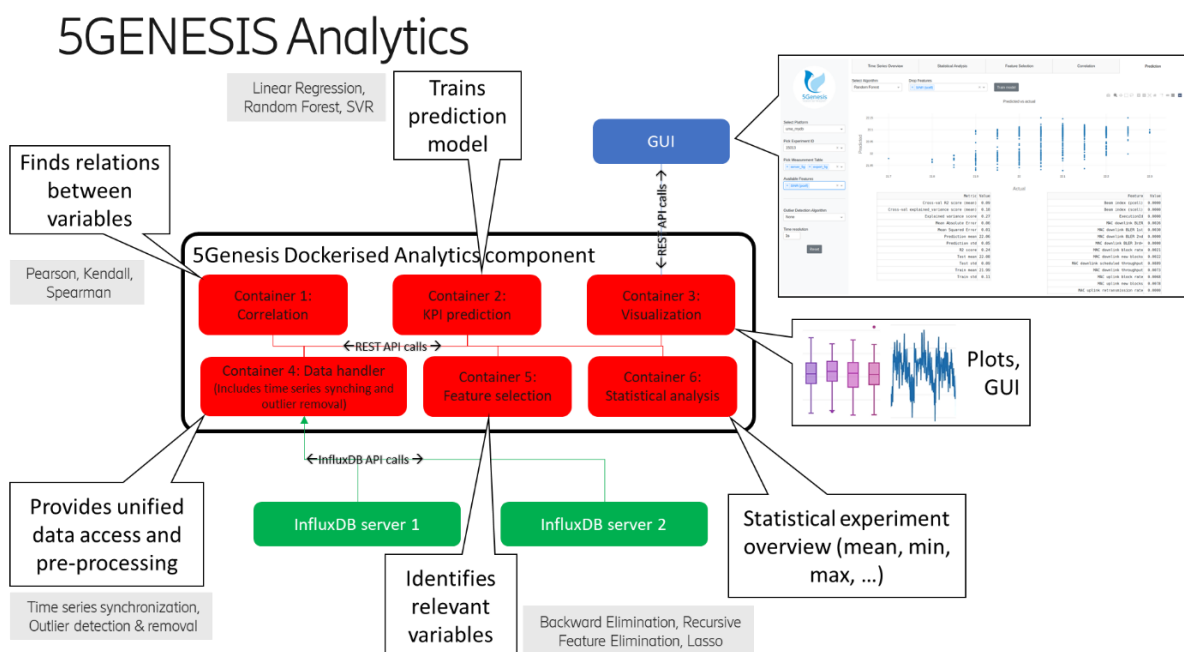


Figure 6. Overview of the containerised analytics micro-services and their functions. A GUI provides easy access to the different functionalities (see next figures for GUI examples).

The following sections describe the algorithms and services in detail.

### 4.2.1. Visualization Service

The visualization service allows for visual representation of the results from the other services in an interactive fashion. The user can access this visualization through a GUI that runs in the browser. The interactive analytics dashboard is implemented using Plotly DASH.<sup>29</sup>

After an experiment is run and its data is stored in the InfluxDB database, the user can browse and analyze the experiment results through the GUI. The user can select which experiment results to display on the left-hand side of the GUI, using the parameters of the experiment query. The main part of the interface is a tabbed environment, where the user can switch between the different Analytics services. From left to right, the tabs contain (1) a time series overview window, (2) a statistical analysis window, (3) a correlation window, (4) a feature selection window and (5) a KPI prediction window as described below.



Figure 7. Time series overview window.

After selecting the desired experiment and KPI(s), the time series overview window (Figure 7) displays a graph with the selected KPI(s). When multiple KPIs are selected for comparison, they are added to the plot with different colours and y-axis scales, as exemplified in Figure 7. Any number of KPIs can be added to this plot and each one will have its own colour and y-axis, however, for best visual comparison, not more than four or five KPIs should be plotted together.

<sup>29</sup> <https://plotly.com/dash/>, Accessed on: March 2021.

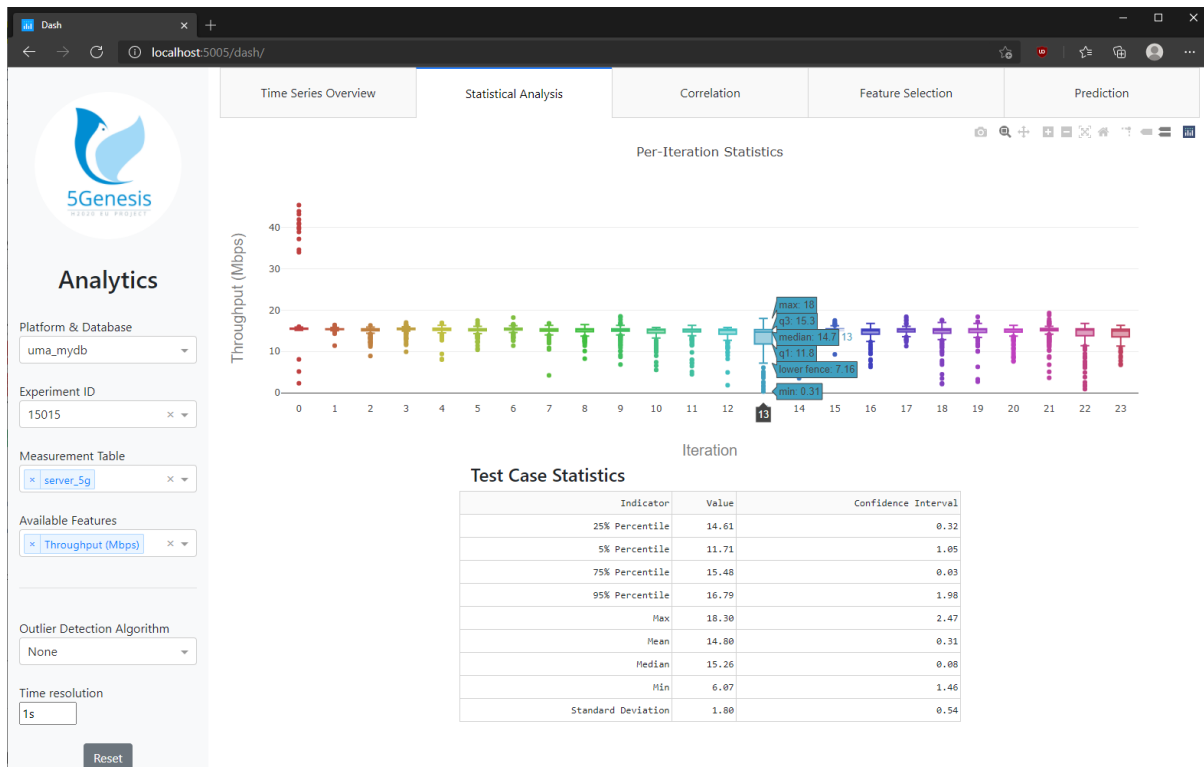


Figure 8. Statistical analysis window.

The statistical analysis window (Figure 8) displays the results of the KPI statistical validation procedure, as defined by 5GENESIS, i.e., per iteration statistics and test case statistics (obtained by averaging over iterations) with a confidence interval [6]. On mouse-over, the graph will show details for each box plot representing each iteration.

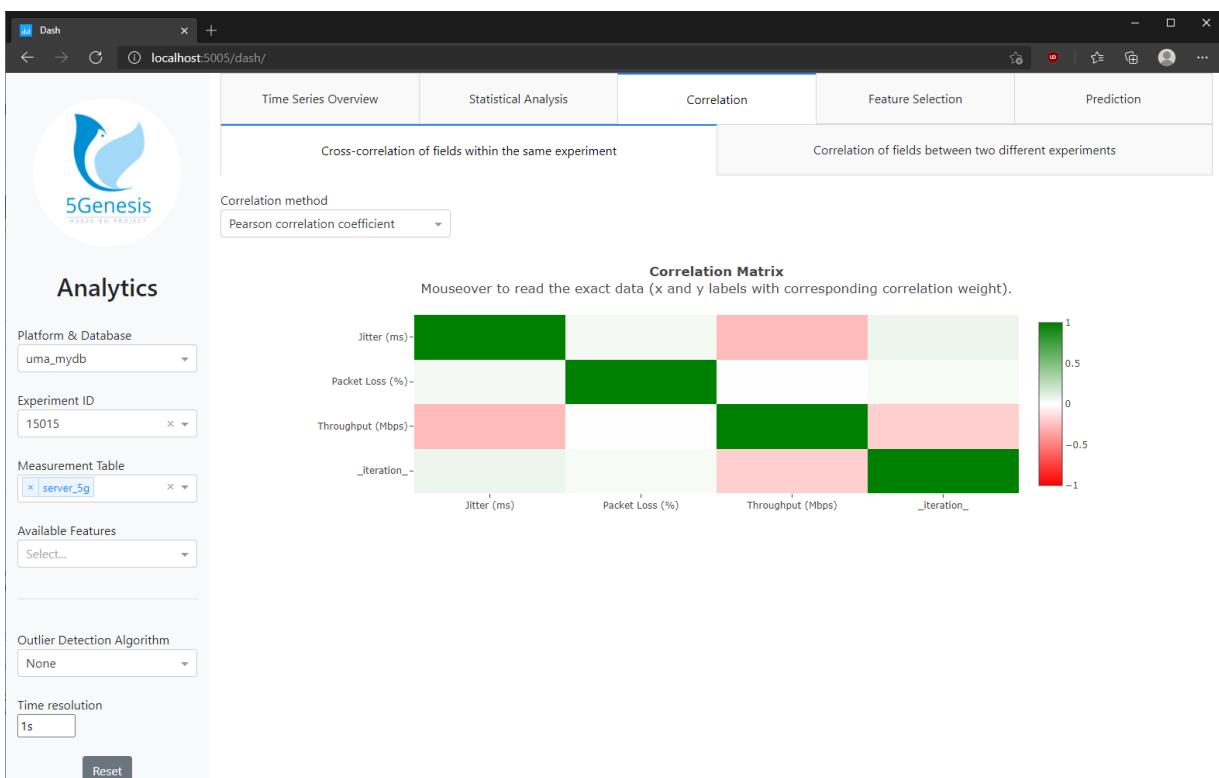


Figure 9. Correlation window showing the cross-correlation of fields.

In the correlation window, the user can select between two types of correlation. The *intra-experiment* cross-correlation (Figure 9) offers a correlation matrix of all numerical fields in the selected experiment.

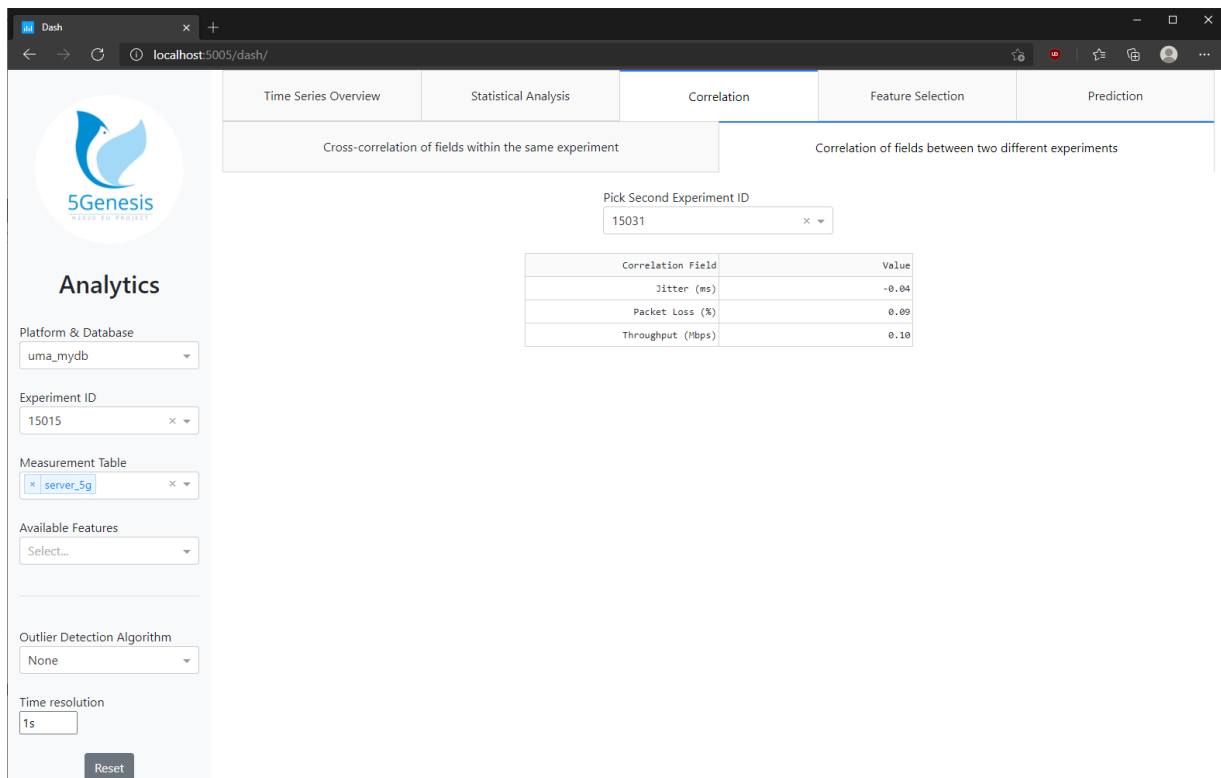


Figure 10. Correlation window showing the inter-experiment field correlation.

The *inter-experiment* field correlation (Figure 10) lets the user select a second experiment from a dropdown menu. The correlation between experiments, for all numerical fields available in both experiments (originally selected one and added one), are then displayed in a table format.



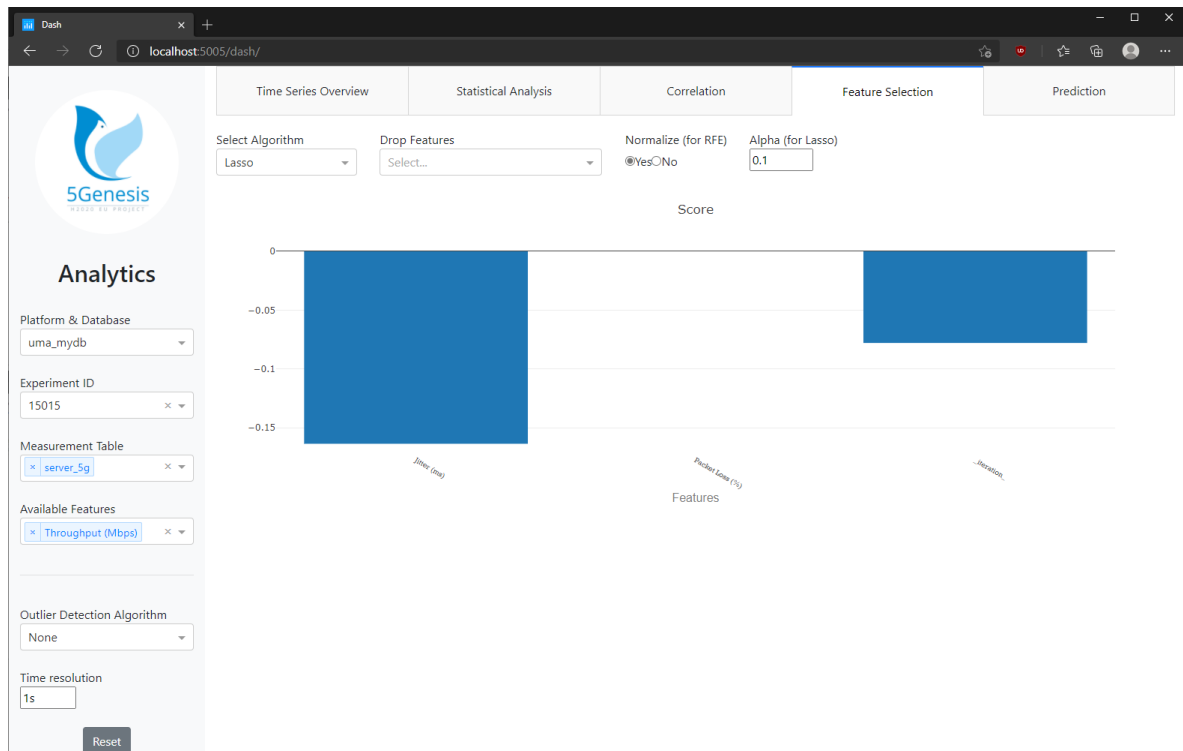


Figure 11. Feature selection window.

The feature selection window (Figure 11) shows the result of the chosen feature selection algorithm, for which additional parameters can be specified and some features may be excluded from the feature selection. The feature selection allows the experimenter to filter out unnecessary parameters for the prediction of a selected target KPI for the current experiment.

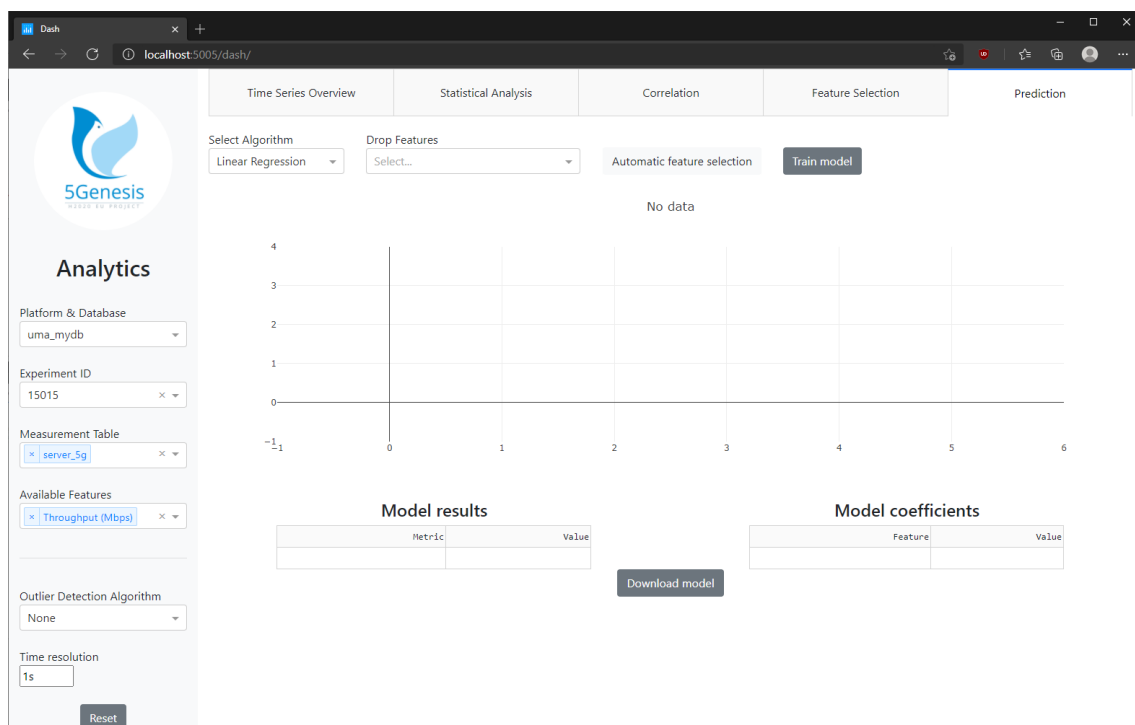


Figure 12. KPI prediction window.

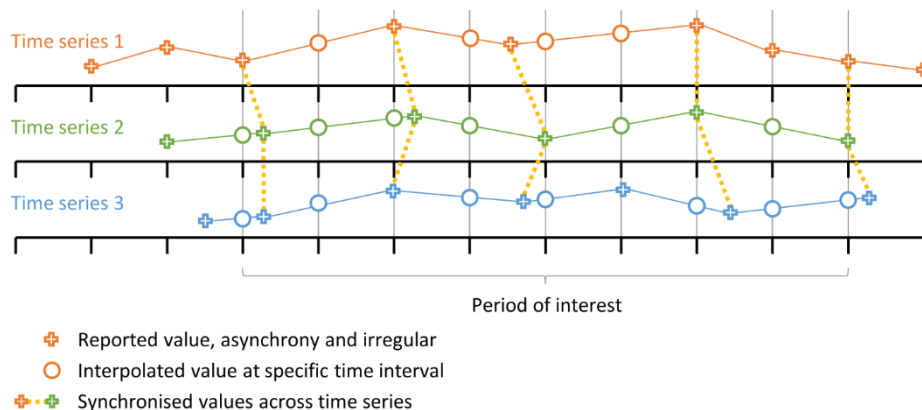
The last window displays the KPI model training for the selected target KPI in the current experiment (Figure 12). The user can select the ML algorithm to be used for the training and exclude features from the training. A button allows for automatic feature selection by calling the Feature Selection service in the background. After the “Train model” button is clicked and the training concluded, the training results are displayed. The results include a table with several error rate metrics, a scatter plot that contrasts actual vs predicted values, and a table that shows the model coefficients (in the case of the linear regression) or the feature importance values (in the case of the Random Forest or SVR algorithms).

#### 4.2.2. Data Handler Service

The data handler service provides a unified access to data that is stored in various InfluxDB instances. It also provides data preprocessing functionality, including time series synchronization, as well as outlier detection and removal as outlined below. The data handler converts InfluxDB data into Pandas dataframes that can be directly used by the various analytics algorithms. For transferring data between services, a Pandas dataframe is converted to json format.

##### 4.2.2.1. Time Series Synchronization

Before running advanced analyses on the data collected during the execution of experiments, the data coming from different monitoring probes needs to be merged. The measurements may be recorded at slightly different times, often with a few milliseconds difference, or with a different granularity, as exemplified in Figure 13.



**Figure 13. Time series synchronisation: Individual measurement points (pluses) can be linked by interpolation (circles) or synchronisation approaches (yellow lines).**

As an example in the scope of the 5GENESIS M&A framework, IM and PM probes may be nearly but not perfectly synchronized, since they are activated by the ELCM in successive steps. Moreover, they may collect data with different sampling periods, e.g., targeting the framework scalability, IM probes collecting the CPU consumption of computing units may work at reduced rates when compared with PM counterparts, which are instead tracking a QoS/QoE KPI, e.g., the user throughput. Within the current version of the Analytics component, we handle the alignment of measurements from different monitoring sources by means of time series synchronization.

Synchronization is a way to align the data points that are closest together in time, irrespective of the exact time when they were recorded. In Figure 13, the synchronization is demonstrated with the yellow dotted lines that connect real data points from the red, green and blue measurements. The synchronization approach is feasible because the time difference between data points across measurements is usually much smaller (often in the order of milliseconds) than the time difference to the neighboring data points in the same measurement (often in the order of seconds), which allows for accurate matching of data points across measurements.

In the current implementation, a time granularity can be specified, on which the data points from all measurements will be synchronized. For example, if the granularity of one second is specified, the timestamp of each data point will be truncated to the full second. If there are multiple data points per second in one measurement, the average is calculated.

#### 4.2.2.2. Anomaly Detection

Another challenge that must be addressed before diving into analytics is the occurrence of drastic anomalies that may skew analytical results. Anomalies are values that were recorded but do not lie in the typical range of a variable, for example a 16-digit Kbps figure where the normal range is in the 3-digit area. A recorded number that is that far off the expected values suggests an error in the measurement or reporting, which is relevant for the health monitoring of the experiment components, such as measurement probes. Two anomaly detection methods are currently integrated in the Analytics component:

- Z-Score uses the standard deviation of the given distribution of recorded data points (per variable) to determine whether a data point does not belong to the expected value range. Any value that is at least three standard deviations away from the mean is considered an outlier and will be removed before proceeding to apply analytical methods;
- Median Absolute Deviation (MAD), also known as Robust Z-Score, is a method that uses the so-called MAD score to detect outliers. The score is defined, for each population sample  $x_i$ , as follows:

$$M_i = \frac{0.6745(x_i - \tilde{x})}{MAD},$$

where  $MAD = \text{median}\{|x_i - \tilde{x}|\}$ ,  $\tilde{x}$  represents the median of the sampled population, and the value of 0.6745 comes out under the assumption of normally distributed data, being in fact the 75% percentile of the standard normal distribution. The samples for which  $|M_i| > 3.5$  are considered outliers.

Both methods are implemented using Pandas and NumPy vectorization methods with a focus on performance and near real time application, and present comparable computational costs.

#### 4.2.3. Statistical Analysis Service

The statistical analysis service provides a statistical overview of the recorded data, following the 5GENESIS guidelines on how a KPI should be validated [6]. The provided statistics include per iteration average, minimum, maximum and quartiles. Box plots also show the quartiles (25%, median, 75%) and outliers to get a quick overview of how stable the experiment has

performed over all iterations. These indicators are then averaged over iterations to produce a test case overall indicator, that is also accompanied by a confidence interval.

#### 4.2.4. Linear Correlation Analysis Service

The correlation service provides state of the art correlation algorithms to find linear relationships between variables.

The experimenter may be interested in the similarity of temporal behavior between recorded variables. A correlation approach is a fast and computationally efficient way to gain insight into the similarity between variables or KPIs. In the current implementation, the experimenter can specify the type of correlation one wants to perform, choosing from state-of-the-art correlations, such as Pearson, Spearman and Kendall. These algorithms are provided by the Pandas package for the programming language Python.

The correlation module currently provides two types of correlation use cases:

- Cross-correlation of fields in the same experiment;
- Correlation of fields across different experiments.

The first correlation analysis use case allows the experimenter to compare variables in the same experiment. For example, the experimenter can investigate the correlation between several recorded throughput and power consumption variables for an Urban Pedestrian iPerf experiment. The second correlation use case offers correlation of targeted measurement fields across different experiments. For example, the experimenter may wish to compare the average throughput measurement of an Urban Pedestrian iPerf experiment with that of an Ideal iPerf experiment.

The correlation service utilizes the `corr()` function that is provided by Pandas and can be applied directly on dataframes.

#### 4.2.5. Feature Selection Service

The feature selection service uses algorithms, such as Backward Elimination (BE), Recursive Feature Elimination (RFE) and Lasso to identify the most relevant variables for a target KPI.

Feature selection is an important step in data analysis, and hence this functionality has been also included in the 5GENESIS Analytics component. Given a dataset related to a particular experiment, the feature selection utility can be primarily applied in quest of dimensionality reduction, e.g., to discard parameters collected from the monitoring probes that are not directly correlated or too much correlated with the KPI under test and hence superfluous. The presence of these parameters may in fact hinder following analyses, e.g., the prediction, as they may negatively impact the fitted model. The algorithms for numeric feature selection are traditionally divided in three main categories, that is, *filter*, *wrapper*, and *embedded* methods. In the second and last version of Analytics, we focus on numeric feature selection, that is, all categorical variables are not considered. Moreover, as discussed in the following, we implement two wrapper methods, namely BE and RFE, and one embedded method based on Lasso-regularized regression.

As part of the wrapper category, BE adopts an ML algorithm to fit a specific model using the available features. It starts with the entire set, and iteratively removes features based on the model accuracy. More precisely, the same model is built at each iteration using the remaining features, and the p-value for each of them is evaluated. The features resulting in a p-value larger than 0.05 are removed. In its current implementation, an Ordinary Least Squares (OLS) model is adopted, being this latter largely used to perform linear regression.

Similar to BE, RFE works by recursively removing features while building a model. It initially fits a model, e.g., linear regression, based on all features. Then, at each iteration, it evaluates feature coefficients and importance, ranks them on the basis of the linear regression accuracy, and finally removes low ranking features.

Across embedded methods, the ones based on regularization are quite popular. Lasso regularization lies in the  $L_1$  function space and deals with possible linear regression overfitting by penalizing unimportant features, assigning them coefficients up to zero. It then follows that this approach can be directly used for feature selection tasks.

The feature selection functionalities included in our Analytics components build on top of Pandas, NumPy, and SciKit-Learn libraries.

#### 4.2.6. KPI Prediction Service

The KPI prediction service provides classic state-of-the-art ML algorithms to train prediction models on a given target KPI for predictive analysis. The trained models can be used further down the line for live predictions.

One of the performance analysis use cases we investigate is KPI prediction. With that the experimenter can attempt to identify how the various network elements impact the targeted measurement KPIs in order to understand how the network can be manipulated to achieve a desired increase or decrease of a KPI. A typical application is to predict the resource requirements that are needed to achieve a desired throughput.

We implement a number of prediction algorithms towards that goal. In the current release of the analytics module, we focus on Linear Regression, Random Forest and Support Vector Machine (SVM)-based regression algorithms, utilizing freely available SciKit-Learn modules. The selection does not include deep learning algorithms at this point as we focus on faster trainable models that are served to the visualization service in near real-time (linear regression) or with an acceptable waiting time (random forest and SVM), for small and medium-sized data. For these experiments, we obtained good results for prediction with the current methods.

### 4.3. APIs for result retrieving

Once the execution of the experiment is finalized, the experimenter retrieves the result information to get insights of the observed performance. The Analytics containers have all the information related to the experiment, and can thus provide the evaluated KPIs that have been defined during the experiment setup, but also information on other metrics collected by the probes in different infrastructure components.

The experimenter can request the desired KPIs through the Dispatcher, by means of command line interface and the 5GENESIS Open APIs, or by using the 5GENESIS Portal. Before the

requested information is provided, the Dispatcher verifies and validate that the user is the owner of the experiment, securing the platform and making available the results of the owner executed experiment.

To offer this information two main endpoints have been provided to the dispatcher environment

- Raw data info, to get better insight of specific metrics in the infrastructure;
- Statistical analysis of a given experiment, with the required measurements and KPI parameters.

### 4.3.1. Analytics service APIs

In addition to the GUI described in Section 4.2.1, each of the Analytics services can be consumed through a REST API. The descriptions in this section refer to a locally deployed analytics platform which can be accessed on <http://localhost>. For remote connections, the “localhost” must be replaced with the remote IP address. Note that the term “measurement” used throughout this section refers to one of the measurement tables composing an InfluxDB database. For example, a measurement may comprise parameters and KPIs related to throughput performance, while another measurement may contain parameters related to network configurations and other information (e.g., location information if available).

#### 4.3.1.1. Data Handler Service

The Data Handler container is the access point for the analytics algorithm containers to retrieve the data from different sources.

The Data Handler can be accessed at <http://localhost:5000>.

A brief API description is available at <http://localhost:5000/api> (or alternatively <http://localhost:5000/help>) and includes the following commands:

- List all available datasources: [http://localhost:5000/get\\_datasources](http://localhost:5000/get_datasources)
- List all available experiments: [http://localhost:5000/get\\_all\\_experimentIds/datasource](http://localhost:5000/get_all_experimentIds/datasource)
- List available experiments for given measurement:  
[http://localhost:5000/get\\_experimentIds\\_for\\_measurement/datasource/measurementId](http://localhost:5000/get_experimentIds_for_measurement/datasource/measurementId)
- List available measurements for a given experiment:  
[http://localhost:5000/get\\_measurements\\_for\\_experimentId/datasource/experimentId](http://localhost:5000/get_measurements_for_experimentId/datasource/experimentId)
- Retrieve data from a given experiment:  
[http://localhost:5000/get\\_data/datasource/experimentId](http://localhost:5000/get_data/datasource/experimentId)
  - Parameters:
    - measurement: e.g. Throughput\_Measures (default all available measurements)
    - remove\_outliers: zscore or mad (default none)
    - match\_series: to synchronize data from multiple measurements (default false)
    - max\_lag: time lag for synchronisation (default 1s)

- limit: any integer to indicate a limit on the returned rows (default none)
- offset: any integer to indicate the offset for the row limit (default none)
- additional\_clause: any InfluxDB clause (default none)
- chunked: whether the results are retrieved from the server in chunks (default false)
- chunk\_size: any integer to define the chunk size (default 10000 if chunked is enabled)
- Retrieve data from two experiments (e.g. for correlation):  
[http://localhost:5000/get\\_data/datasource/experimentId1/experimentId2](http://localhost:5000/get_data/datasource/experimentId1/experimentId2)
  - Parameters same as above
- Clear the data handler's cache: [http://localhost:5000/purge\\_cache](http://localhost:5000/purge_cache)

Example output for /get\_data/:

```
{
  "ADB_Resource_Agent": {
    "Available RAM (MB)": {
      "0": 845,
      "1": 803,
      "2": 808,
      ...
    },
    ...
  },
  "Throughput_Measures": {
    "DL - AverageThput (Kbps)": {
      "0": 3617.0005,
      "1": 4436.538269,
      "2": 3900.982222,
      ...
    },
    ...
  }
}
```

#### 4.3.1.2. Correlation Service

The Correlation container enables the user to perform:

- a cross-correlation on the fields within the same experiment (resulting in a correlation matrix)
- a correlation of the fields across different experiments (resulting in a list of correlation values)

Access the Correlation module at <http://localhost:5001>.

An API description is available at <http://localhost:5001/api> and includes the following commands:

- Cross-correlation of fields within the same experiment:  
<http://localhost:5001/correlate/fields/datasource/experimentId>
  - Parameters:
    - measurement: e.g. Throughput\_Measures (default all available measurements)
    - method: any Pandas correlation method (default pearson)
    - remove\_outliers: zscore or mad (default none)
    - field: filter for fields, e.g. Throughput (default none, all fields are included)
- Correlation of fields between different experiments:  
<http://localhost:5001/correlate/experiments/datasource/experimentId1/experimentId2>
  - Parameters same as above

Example output for /correlate/fields/:

```
{
  "correlation_matrix": {
    "DL - AverageThput (Kbps)": {
      "SNR": 0.21373483,
      "RSSI": 0.1842762049,
      ...
    },
    ...
  }
}
```

Example output for /correlate/experiments/:

```
{
  "correlation_list": {
    "Available RAM (MB)": 0.190006483,
    "DL - AverageThput (Kbps)": -0.0301038513,
    ...
  }
}
```

#### 4.3.1.3. Prediction Service

The Prediction container enables the user to train and download a KPI prediction model on a specified target KPI.

Access the Prediction module at <http://localhost:5002>.

An API description is available at <http://localhost:5002/api> and includes the following commands:

- KPI Prediction using different algorithms:  
<http://localhost:5002/train/datasource/algorithm/target>
  - Algorithm (mandatory selection):



- Linear Regression: linreg
- Random Forest: rf
- Support Vector Regression (experimental): svr, linear\_svr, nu\_svr
- Target (mandatory selection): Target variable for prediction, e.g. "DL - AverageThput (Kbps)"
- Parameters:
  - drop\_feature: any feature to be ignored for training (default None)
  - experimentid: e.g. 499 (at least one experiment ID is mandatory)
  - measurement: e.g. Throughput\_Measures (default all available measurements)
  - normalize: not relevant for some algorithms, e.g. Random Forest or SVR (default False)
  - remove\_outliers: zscore or mad (default none)
- Download the trained model: <http://localhost:5002/model>

Example output for /train/:

```
{
  "coefficients": {
    "DL - StatDTX Ratio": 0.6477855028,
    "DL - PDCCH Ratio": 0.1479913101,
    ...
  },
  "real_predicted_values": {
    "y_pred": {
      "0": 3814.6712453507,
      "1": 4633.0326365561,
      ...
    },
    "y_test": {
      "0": 3741.438577,
      "1": 4772.295,
      ...
    }
  },
  "results": {
    "Cross-val R2 score (mean)": 0.7576147901,
    "Cross-val explained_variance score (mean)": 0.7694582032,
    ...
  }
}
```

#### 4.3.1.4. Statistical Analysis Service

The Statistical Analysis container enables the user to perform KPI statistical validation.

Access the Statistical Analysis module at <http://localhost:5003>.

An API description is available at <http://localhost:5003/api> and includes the following commands:

- KPIs statistical validation across experiments and measurement tables stored in InfluxDB: [http://localhost:5003/statistical\\_analysis/datasource](http://localhost:5003/statistical_analysis/datasource)

- Parameters:
  - experimentid: e.g. 499 (at least one experiment ID is mandatory)
  - measurement: e.g. Throughput\_Measures (at least one measurement is mandatory)
  - it: iteration identifier (default= 'iteration')
  - kpi: e.g. 'Delay (ms)' (at least one kpi is mandatory)
  - unit: e.g. ms (default= None)
  - exptype': [0,1] 0: if the experiments contain several kpi samples per iteration, 1: if the experiments contain one kpi sample per iteration (e.g. Service Creation Time) (default= 0)

Example output for /statistical\_analysis/:

```
{
  "experimentid": {
    "520": {
      "Delay (ms)": {
        "Iteration Statistics": {
          "0": {
            "25% Percentile":54.800000000000004,
            "5% Percentile":24.259999999999998,
            ...
          },
          "1": {
            "25% Percentile":54.550000000000004,
            "5% Percentile":25.61,
            ...
          },
          ...
        },
        "Test Case Statistics": {
          "25% Percentile": {
            "Confidence Interval":1.0124404204175417,
            "Value":57.058
          },
          "5% Percentile": {
            "Confidence Interval":1.1079808517791605,
            "Value":24.4246
          },
          ...
        }
      }
    }
  }
}
```

If units are inserted, they will be associated to the KPIs in the same order of appearance. In the hereunder example only one unit is inserted, thus it will be associated to the first KPI, whilst the latter will be printed with no unit.

Example output:

```
{
  "experimentid": {
    "520": {
      "Delay (ms)": {
        "Iteration Statistics": {
          "0": {
            "25% Percentile":54.800000000000004,
            "5% Percentile":24.259999999999998,
            ...
          },
          "1": {
            "25% Percentile":54.550000000000004,
            "5% Percentile":25.61,
            ...
          },
          ...
        },
        "Test Case Statistics": {
          "25% Percentile": {
            "Confidence Interval":1.0124404204175417,
            "Value":57.058
          },
          "5% Percentile": {
            "Confidence Interval":1.1079808517791605,
            "Value":24.4246
          },
          ...
        }
      },
      "SNR - (dB)": {
        "Iteration Statistics": {
          "0": {
            "25% Percentile":30,
            "5% Percentile":30,
            ...
          },
          "1": {
            "25% Percentile":30,
            "5% Percentile":30,
            ...
          },
          ...
        },
        "Test Case Statistics": {
          "25% Percentile": {
            "Confidence Interval":0,
            "Value":30
          },
          "5% Percentile": {
            "Confidence Interval":0,
```

#### 4.3.1.5. Feature Selection Service

The Feature Selection container enables the user to select the most relevant features according to a specified target KPI.

Access the Feature Selection module at <http://localhost:5004>.

An API description is available at <http://localhost:5004/api> and includes the following commands:

- Feature Selection using different algorithms:
  - <http://localhost:5004/selection/datasource/algorithm/target>
    - Algorithm (mandatory selection):
      - Backward Elimination: backward, backward\_elimination
      - Recursive Feature Elimination: RFE, rfe
      - Lasso Regression: LASSO, Lasso, lasso
    - Target (mandatory selection): Target variable, e.g. "DL - AverageThput (Kbps)"
    - Parameters:
      - drop\_feature: any feature to be ignored for training (default None)
      - experimentid: e.g. 499 (at least one experiment ID is mandatory)
      - measurement: e.g. Throughput\_Measures (default all available measurements)
      - normalize: relevant for RFE (default False)
      - alpha: relevant for Lasso (default 0.1)
      - remove\_outliers: zscore or mad (default none)

Example output for /selection/:

```
{
  "Features - Original": {
    "0": "DL - ACK Count",
    "1": "DL - ACK Ratio",
    ...
  },
  "Features - Selected": {
    "0": "DL - ACK Ratio",
    "1": "DL - Byte Count",
    ...
  },
  "Score": {
    "DL - ACK Ratio": 1428937721.858489,
    "DL - Byte Count": 5003183391.821027,
    ...
  }
}
```

## 5. M&A INTEGRATION WITH OTHER COMPONENTS

### 5.1. Integration with the 5GENESIS Portal

During the development phase, it was decided to keep the 5GENESIS Portal and Analytics as two separate entities with a loose integration. The reason for this was to allow each component to progress separately, while providing the best possible interface for two different use cases:

- The 5GENESIS Portal provides an interface for experiment definition and execution, as well as access to logs and fast visualization of a limited set of raw data;
- Analytics gives experimenters in-depth access to all the generated experiment data, as well as advanced tools for statistical and ML analyses.

Nevertheless, the 5GENESIS Portal provides quick access to the Analytics Module capabilities in the form of a custom link that is generated for each executed experiment.


Execution ID	Status	Start Time	End Time	Action
46	Finished	02 December 2020, 12:49:05	02 December 2020, 12:50:55	

Figure 14. Experiment actions on the 5GENESIS Portal. The third button identifies the access to the Analytics module.

Figure 14 shows the actions available in the Portal for each experiment execution, including the access to the Analytics dashboard of the experiment (third button). After being authenticated in the Portal via the Dispatcher, an experimenter can make use of this button, and a new browser tab containing the Analytics dashboard of that particular experiment opens.

Since this functionality makes use of standard web addresses, the access to Analytics uses an encrypted reference to the Execution ID instead of its numeric value. This is done in order to avoid that malicious actors may be able to manually create links to Analytics and thus gain access to private experiment data. For this reason, the Analytics module disables access to the full list of experiments saved in the database when accessed through the Portal.

### 5.2. Integration with the 5GENESIS Slice Manager and Policy Engines

#### 5.2.1. NEAT

In line with the plans outlined in Deliverable D3.5, the NEAT policy framework has been integrated with the 5GENESIS Slice Manager during the second Project cycle, enabling policy management at the UE to take advantage of slice monitoring information. By using NEAT as a front-end to the Slice Manager, the UE can leverage available network slices in a way that better meets the needs of its running applications. As the key component in the management of network slices, the Slice Manager has a complete view of existing network-slice instances.

An overview of the integration is illustrated in Figure 15.

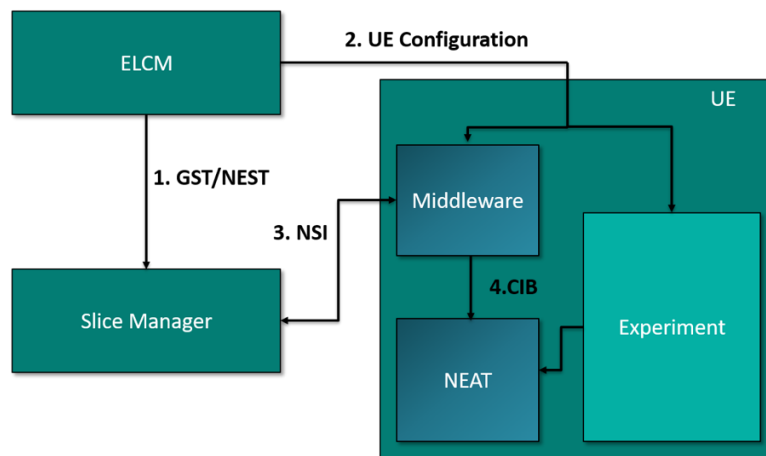


Figure 15. NEAT-Slice Manager integration.

NEAT runs on the UE and decouples the application from the used transport protocol, allowing the protocol and its configuration to be dynamically selected at run-time. The NEAT policy components comprise a Characteristics Information Base (CIB), a repository which stores information on network characteristics, available interfaces, and cached information from previous connections; a Policy Information Base (PIB), a repository for those policies that govern the selection of transport services; and a Policy Manager, the component of NEAT that combines application requirements with information from the CIB and the PIB repositories to guide transport selection.

Normally, an application must use the NEAT User API in order to take advantage of the system's capabilities. As it is not always feasible to rewrite existing applications to take advantage of the NEAT User API, we use a NEAT proxy [16] deployed on the UE to provide applications with transparent access to NEAT. Linux TProxy is used to route traffic through the proxy, which terminates outgoing connections in order to establish a new connection to the original destination using the NEAT system. The UE's applications can provide the proxy with their requirements by interfacing with the NEAT Policy Manager via a REST API.

A middleware component handles the communication between the Slice Manager and NEAT.

When an experiment is started, the UE obtains the details required for contacting the Slice Manager from the ELCM, and also retains a unique identifier (such as the Single-Network Slice Selection Assistance Information (S-NSSAI), or a similar identifier) to each slice it belongs to. The ELCM is responsible for initiating the creation of a network slice by interacting directly with the Slice Manager when setting up the experiment. Upon creation, the Slice Manager returns an identifier for the created slice to the ELCM that can be passed on to the UE. The middleware component then contacts the Slice Manager to register the NEAT instance as a policy system through the Slice Manager's southbound interface. The middleware component is also responsible for deregistering the policy system when an experiment is completed.

Once registered at the Slice Manager, the middleware component begins to periodically poll for Network Slice Instance (NSI) information on the slices identified by the slice identifiers. The NSI information is then prepared by the middleware component for use in the NEAT system, which is added to the NEAT Policy Manager CIB repository via the Policy Manager's REST API.

The information in the CIB repository, together with local policies can then assist the NEAT proxy in tailoring transport services to its applications.

### 5.2.2. APEX

The APEX policy engine, described in Deliverable D3.1 [17], is used for flexible adaptation of slices to dynamic situations in the network. This is an optimization strand for the Slice Manager, with APEX being integrated into the Slice Manager functionality. In order to be able to adapt the slicing mechanisms to dynamic situations, APEX will rely on input from the slice monitoring module (e.g., alarms from Prometheus can be used for triggering the APEX events). Full description of this functionality and its implementation are included in Deliverable D3.4 on Slice Management (Release B).

## 5.3. Integration with 5GENESIS Security Analytics

Anomaly Detection is the process of identifying outliers in data, and Deep Learning models can detect anomalies in a set of data, if they have trained using normal data.

The 5GENESIS Security Analytics component, described in Deliverables D3.13 (Release A) [18] and D3.14 (for its Release B) implements a functionality to detect anomalies on an Edge Node. For performing the detection, metrics from RAN and the Edge Node, collected under the M&A framework, are given as input to the detection algorithm. Finally, detected anomalies are presented in a custom Grafana GUI.

This section describes the components involved in this functionality, ultimately showcasing how M&A and Security Analytics are integrated to provide accurate anomaly detection at the edge.

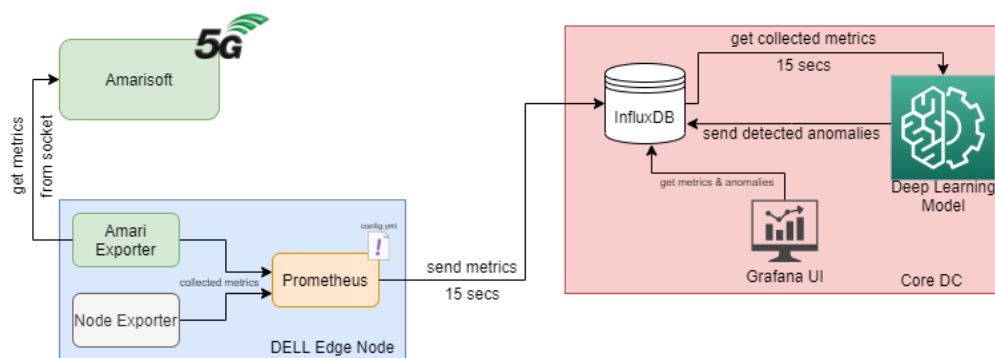


Figure 16. M&A and Security Analytics integration for anomaly detection.

As shown in Figure 16, there are three main components realizing the integration between M&A and Security Analytics, i.e., Amarisoft RAN, Edge Node, and Core Data Center (DC). Amarisoft RAN provides an API for collecting 5G metrics. In Edge Node, a Prometheus server and two exporters are running as part of the M&A framework. An InfluxDB for the Security Analytics is deployed in Core DC (it could be a dedicated instance as well as the same instance used for collecting experimental results), where the anomaly detection software and a dedicated Grafana GUI are also running.

In the Edge Node, the Amarisoft Exporter described in Section 3.1.2.1 collects metrics from the Amarisoft RAN, while the Node Exporter collects metrics from the Edge Node. Then, a Prometheus server scrapes both exporters and publishes the collected metrics to InfluxDB every 15 seconds.

The anomaly detection model fetches the ingested data from InfluxDB and decides if a record is an anomaly compared to normal records or not. Detected anomalies are also saved to InfluxDB, while the Grafana GUI depicts metrics gathered from Edge Node and Amarisoft RAN, and also keeps a table with all detected anomalies. For each anomaly a feature allows to show more info about collected and monitored metrics.

### 5.3.1. Edge Cloud

#### 5.3.1.1. Prometheus server, Node Exporter, Amarisoft Exporter

As also anticipated in Section 3.1.1, Prometheus can be installed either standalone or using Docker image, and it is configured using a yml file, where the url of the InfluxDB instance used for storing the collected metrics is indicated along with other monitored targets. An example of yml file used for the anomaly detection functionality is given below:

```
# Remote write configuration for InfluxDB
remote_write:
  - url: "http://{influx_host}:{port}/api/v1/prom/write?db={db}&u={user}&p={pass}"
scrape_configs:
  - job_name: 'amari exporter'
    scrape_interval: 5s
    static_configs:
      - targets: ["{job_ip}:{job_port}"]
  - job_name: 'node_exporter_dellEdge'
    scrape_interval: 15s
    static_configs:
      - targets: ["{job_ip}:{job_port}"]
```

In this case, Prometheus scrapes two targets, i.e., Node Exporter and the Amarisoft Exporter, collecting metrics from the Edge Node and RAN-related metrics from Amarisoft RAN.

### 5.3.2. Core DC

#### 5.3.2.1. InfluxDB

An InfluxDB instance must have a specific organization in measurement tables, where different metrics are stored as time series. Figure 17 shows the organization of the InfluxDB instance



used for the anomaly detection functionality, where columns with stars (\*) are explicitly required and used by the detection algorithm.

tx_cpu_time	
time-series	time (*)
tag	__name__
tag	instance
tag	job
field	value (*)

rx_cpu_time	
time-series	time (*)
tag	__name__
tag	instance
tag	job
field	value (*)

dl_bitrate	
time-series	time (*)
tag	__name__
tag	cellId (*)
tag	instance
tag	job
field	value (*)

ul_bitrate	
time-series	time (*)
tag	__name__
tag	cellId (*)
tag	instance
tag	job
field	value (*)

node_cpu_seconds_total	
time-series	time (*)
tag	__name__
tag	cpu (*)
tag	instance
tag	job
tag	mode (*)
field	value (*)

node_memory_MemFree_bytes	
time-series	time (*)
tag	__name__
tag	instance
tag	job
field	value (*)

node_network_receive_bytes_total	
time-series	time (*)
tag	__name__
tag	device (*)
tag	instance
tag	job
field	value (*)

node_network_transmit_bytes_total	
time-series	time (*)
tag	__name__
tag	device (*)
tag	instance
tag	job
field	value (*)

Figure 17. InfluxDB organization in measurement tables. Among all the metrics, the anomaly detection algorithm uses the ones identified by a star (\*).

In addition, certain columns in some measurement tables require specific values:

- **node\_cpu\_seconds\_total:**
  - cpu: 0 or 1, represent the two cpu cores. Seconds are recorded per core.
- **node\_network\_receive\_bytes\_total / node\_network\_transmit\_bytes\_total:**
  - device: enp1s0, enp0s20u1 or ppp0 are used for collecting metrics and aggregating received and transmitted bytes and their rate.
- **ul\_bitrate / dl\_bitrate:**
  - cellId: 1 for 4G connections or 2 for 5G connections. The algorithms collect metrics only from cellId 2.

### 5.3.2.2. Grafana

Similarly to Prometheus, Grafana can also be installed either locally or within a Docker container. After installation, the Grafana dashboard dedicated to anomaly detection require two data sources to be set, i.e., a Prometheus data source and an InfluxDB data source. This is a custom build dashboard and can be imported from a json file (ui/threat-detection-ui.json).

The Grafana dashboard including metrics and detected anomalies is shown in Figure 18. The first two panels in top row report the average CPU and memory load. The next two panels in the top row report the sum of all received and transmitted bytes across all network interfaces. In the second row there are two panels. The left one shows the percentage of used CPU and Random Access Memory (RAM) and the second one the network bytes that received and transmitted from three specific interfaces. Finally, in the bottom row there is the table with all detected anomalies. The table contains the time the anomaly was detected and values for some features at the time the anomaly occurred, like CPU and RAM usage, network metrics and 5G metrics. Finally, if the cause of anomaly can be detected, a message informs about it.

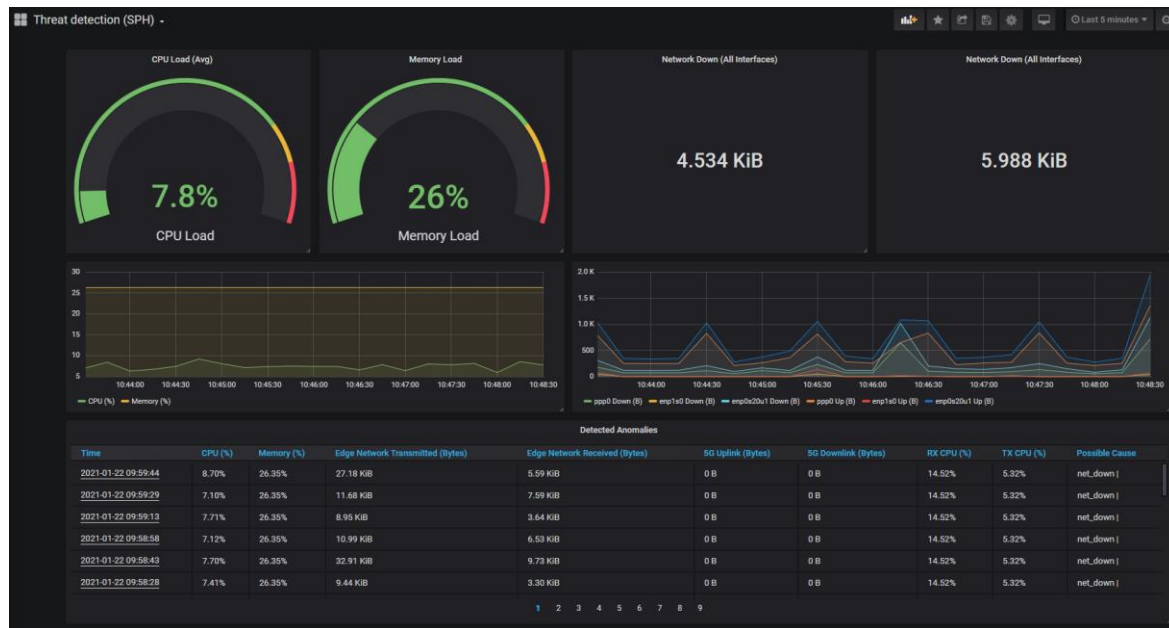


Figure 18. An example of Grafana dashboard for anomaly detection.

### 5.3.2.3. Autoencoder

One way of detecting anomalies is using Autoencoders. Autoencoders are formed by two main parts, the Encoder and the Decoder. The Encoder takes the input and compresses it in a smaller representation. The Decoder takes the compressed input and decompresses it, trying to reconstruct the original input. Between the actual and the decompressed input there is an error. Given that the model has been trained in normal data, this error must be very small. If this error is above a threshold then the given input can be considered as an anomaly.

In order to be used, the Autoencoder for anomaly detection requires Docker, Python 3.7, and further Python packages.<sup>30</sup> It functions in the following phases:

**Preprocessing.** Training data have been collected from both Node Exporter and Amarisoft Exporter. Because of these two different systems, there is a small difference in timestamps for these metrics. For this reason, the training dataset is resampled every 15 seconds, to synchronize records from these two exporters. For training, the following features have been selected: CPU percentage rate, RAM percentage rate, RX/TX CPU percentage rate, rate of transmitted/received bytes and rate of bytes downloaded/uploaded from 5G interface. These features are normalized using Min-Max normalization. The min and max values that will be used for normalization will be saved in a json file. After resampling and normalize, the dataset is split in sequences, with four steps per sequence, where the first three records will be used to predict the fourth.

**Training.** For our model we have used the following Autoencoder architecture. Our Encoder has 5 Bidirectional Long Short-Term Memory (LSTM) layers, which take the given data and compress them to a smaller representation. Our Decoder also has 4 Bidirectional LSTM layers, which try to decompress the compressed input and recreate the original given data. We have

<sup>30</sup> <https://github.com/5genesis/Security-Framework/tree/main/anomaly%20detection>, Accessed on: March 2021.

decided to use LSTMs since our data are a batch of time series records, which are suitable for LSTMs. Also, we have selected to use Bidirectional LSTMs, because they can utilize features from both previous and next records. Finally, at the end we have a Dense layer, which maps the output of the Autoencoder in the features we want to predict (Figure 19).

```
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
bidirectional_1 (Bidirection	(None, 4, 128)	37376
bidirectional_2 (Bidirection	(None, 4, 64)	41216
bidirectional_3 (Bidirection	(None, 4, 32)	10368
bidirectional_4 (Bidirection	(None, 4, 16)	2624
bidirectional_5 (Bidirection	(None, 4, 8)	672
bidirectional_6 (Bidirection	(None, 4, 16)	1088
bidirectional_7 (Bidirection	(None, 4, 32)	4224
bidirectional_8 (Bidirection	(None, 4, 64)	16640
bidirectional_9 (Bidirection	(None, 128)	66048
dense_1 (Dense)	(None, 8)	1032
Total params: 181,288		
Trainable params: 181,288		
Non-trainable params: 0		

```
INFO:root:None
```

Figure 19. LSTM layers of the Autoencoder used for anomaly detection.

A Rectified Linear Unit (ReLU) activation function is used. Stochastic Gradient Descent (SGD) with Nesterov momentum has been selected as optimizer, with learning rate equal to 0.01. As validation dataset 10% of the trainset has been used. The model is trained for 50 epochs.

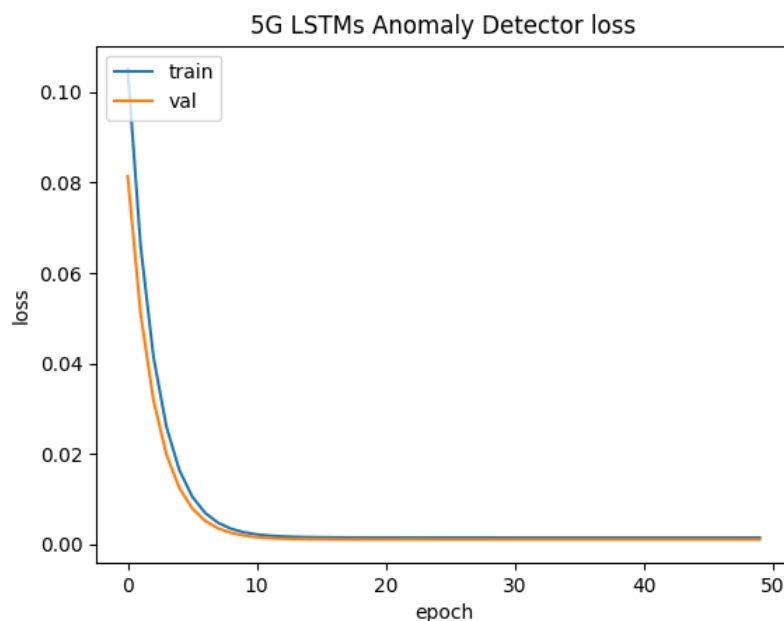


Figure 20. Anomaly detector loss.

Our model is trained in normal traffic in order to predict the next value in data time-series, given that its input data is part of normal traffic. Anomaly Detector's loss, as shown in Figure 20, describes the Mean Absolute Error (MAE) between the predicted values of time-series points and their actual values, during training epochs.

**Evaluation.** If evaluation is enabled, the trained model will be loaded and used for testing in three different datasets. The first dataset has been collected during a CPU stress test attack. The second dataset has been collected during an iPerf stress test attack. The third dataset is the training set. All these datasets will be normalized with saved normalization values during training. Using these datasets, the Root Mean Square Error (RMSE) between the actual and predicted values will be calculated. Network features' thresholds will not be affected from CPU stress dataset and CPU features' thresholds will not be affected from iPerf stress test dataset. The 99<sup>th</sup> percentile of each feature's RMSE will be considered as threshold from the anomaly detection algorithm. The 99<sup>th</sup> percentile has been selected compared with the max value in order to avoid outliers, that may exist in these datasets. User-defined thresholds will not be overridden from calculated RMSEs.

**Testing.** Trained model will be used in order to predict the next features' values in the time series. Because it is trained in normal traffic it will predict these values considering that the incoming traffic is normal. If the RMSE between the predicted and actual value is above a set threshold, this record is considered as anomaly. Algorithm fetches the last received records from InfluxDB every 15 seconds. For predicting the next value, a sliding window with size equal to 30, keeping only last records is used. Fetched records are preprocessed and normalized before they are used by the model for predicting. After prediction the RMSEs between the actual and predicted values are calculated and compared with set thresholds, to identify if the record is considered as an anomaly or not. A possible cause of the anomaly is also saved in the database with the detected anomalies or an 'unknown cause' will be set if the cause cannot be identified. A proposed thresholds json file for testing is the following:

```
{
  "cpu_threshold": 0.05, "mem_threshold": 0.1, "cpu_tx_threshold": 0.1,
  "cpu_rx_threshold": 0.1, "net_up_threshold": 0.501,
  "net_down_threshold": 0.501, "net_5g_up_threshold": 0.501,
  "net_5g_down_threshold": 0.501, "overall_threshold": 0.15
}
```

## 6. TESTING AND VALIDATION

### 6.1. Testing and Validation of the Analytics Services

This section validates the Analytics services using a throughput experiment executed in the Malaga platform, with one static UE connected to one 5G gNB, in a non-line-of-sight scenario, for the validation. The sections below show the functionality of the time series overview, outlier removal, statistical analysis, correlation, feature selection and KPI prediction.

#### 6.1.1. Time Series Overview

The goal of the time series overview is to give the user a first glance at the behavior of the selected KPIs of interest. The following example (Figure 21) shows the Throughput KPI collected during the experiment (note that, as also clarified in Section 7.1.3, the experiment is repeatedly executed over 24 iterations).

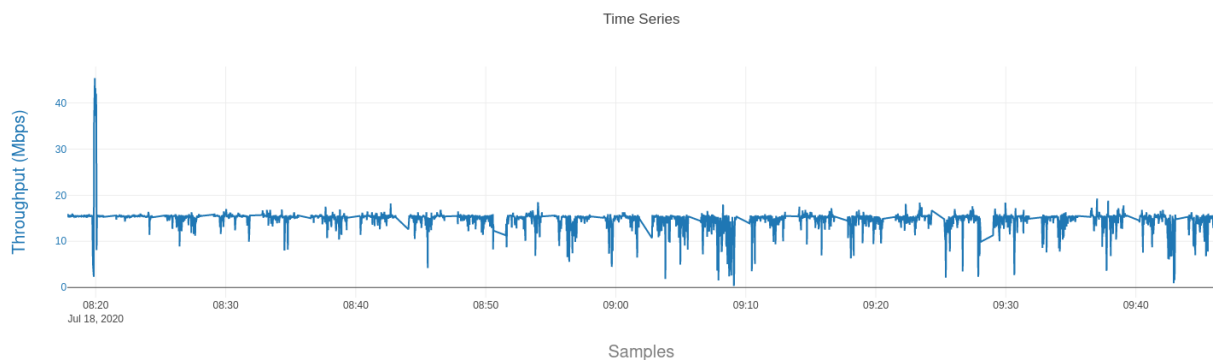


Figure 21. Time series overview of the Throughput KPI.

The user learns at least three pieces of information about the Throughput KPI from this chart:

- It lies between 0 and 50 Mbps;
- It hovers around 15 Mbps with frequent drops to 10 Mbps and sometimes 0 – 5 Mbps;
- It spikes to around 50 Mbps once during the early phase of the experiment, which is an obvious outlier.

#### 6.1.2. Outlier Removal

The outliers that the time series overview shows can easily be caught and removed with the outlier detection and removal function that is part of the Data Handler service.

Figure 22 shows visually how outliers are removed from the time series data. Note that in the following results, the outliers have been removed with the Z-score outlier removal algorithm described in Section 4.2.2.2.

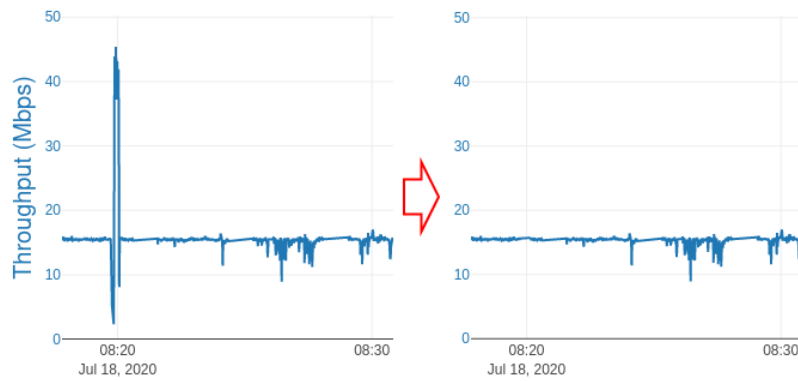


Figure 22. Outlier removal in the Throughput KPI.

### 6.1.3. Statistical Analysis

The statistical analysis gives an overview of the min, max, quartiles and outliers – if not removed in the previous step – of each of the 24 experiment iterations (Figure 23). This allows the user to inspect and compare individual experiment iterations.

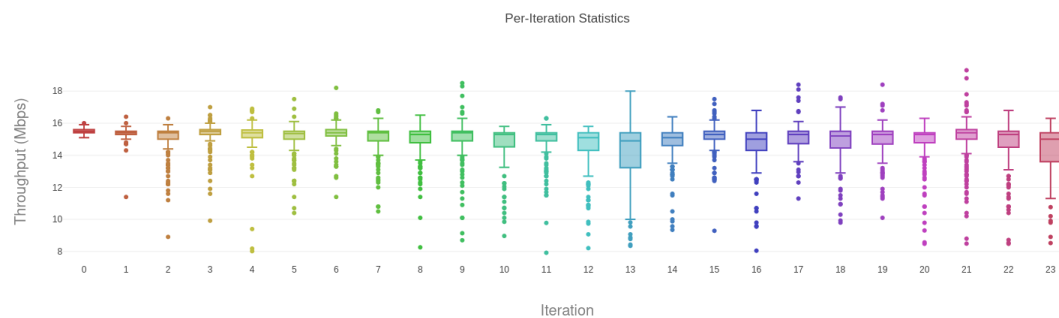


Figure 23. Statistical analysis of 24 experiment iterations.

Both the time series overview and the statistical analysis show that the experiment ran relatively stable through the 24 iterations. In addition to the statistics about individual experiment iterations, a summary table is also provided. As shown in Figure 24, the summary shows percentile, min, max and other values averaged over the whole experiment run, together with a confidence interval calculated over all iterations. The units of the reported figures correspond to the units of the investigated KPI.

#### Test Case Statistics

Indicator	Value	Confidence Interval
25% Percentile	14.61	0.32
5% Percentile	11.71	1.05
75% Percentile	15.48	0.03
95% Percentile	16.79	1.98
Max	18.30	2.47
Mean	14.80	0.31
Median	15.26	0.08
Min	6.07	1.46
Standard Deviation	1.80	0.54

Figure 24. Test case statistics table showing the summary over all experiment iterations with 95% confidence interval.

### 6.1.4. Correlation

Figure 25 shows the complete correlation matrix of all numerical KPIs recorded for this experiment, where green values represent positive correlation and red values negative correlation. At a glance, the user can verify if the KPIs correlate as expected.

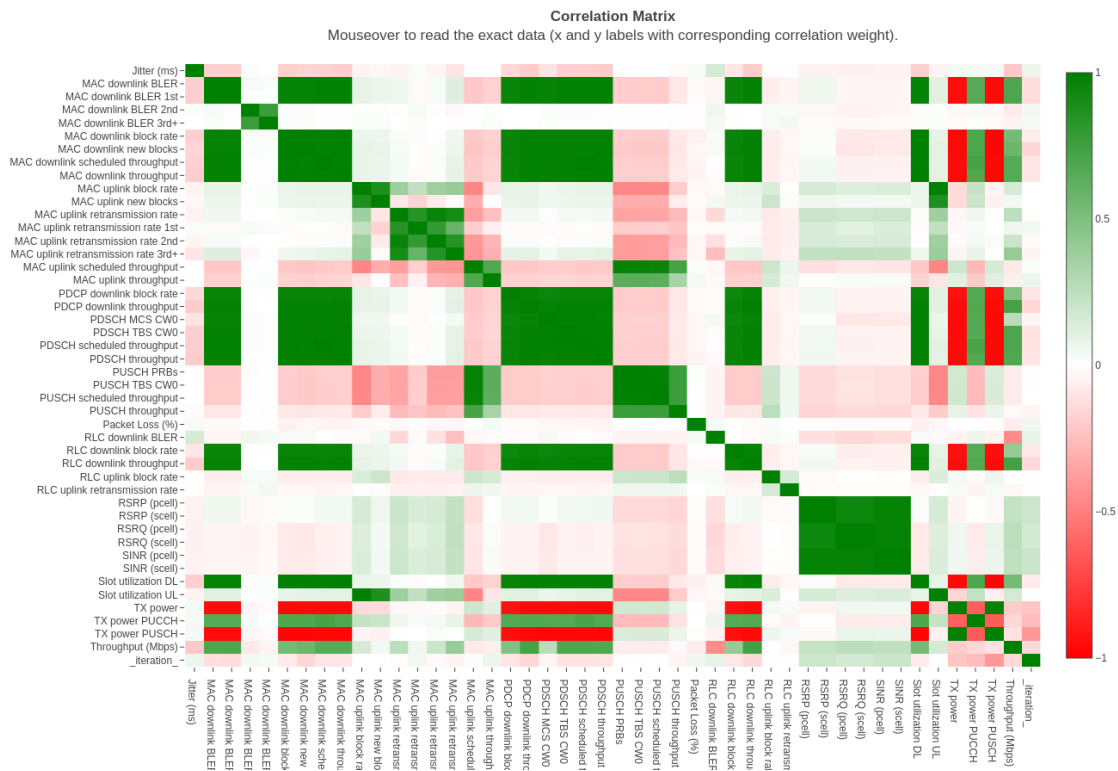
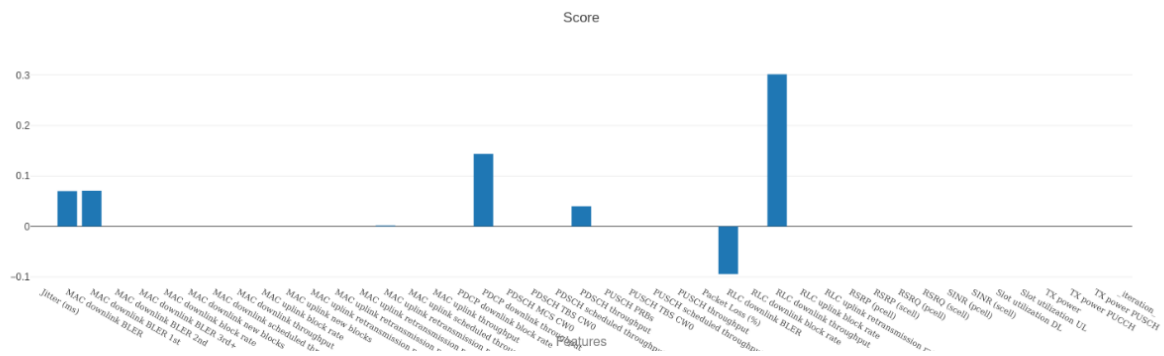


Figure 25. Correlation matrix of all numerical KPIs.

The correlation matrix can be further filtered by KPI to reduce the size of the correlation matrix and avoid information overload. The user can also choose between one of three correlation methods: Pearson correlation coefficient, Spearman rank correlation coefficient, and Kendall rank correlation coefficient.

### 6.1.5. Feature Selection

The feature selection function helps to identify the most relevant KPIs for a given target KPI, for example towards a prediction task. This can be achieved through analyzing the correlation matrix to some extent, but many feature selection algorithms are superior to the simple linear relations between variables that can be discovered through correlation.



**Figure 26. Feature selection output, with 6 most relevant variables remaining as most relevant for predicting the Throughput KPI.**

Figure 26 shows the result of the feature selection task, where from over 40 features only 6 remain that the Lasso feature selection algorithm deemed most impactful towards predicting the Throughput KPI.

### 6.1.6. KPI Prediction

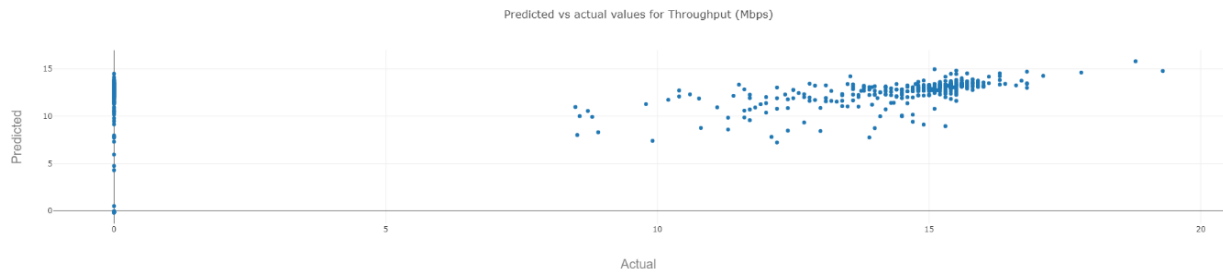
In the KPI prediction task, we perform predictive analysis by training a prediction model on the experiment data. The trained model gives a deeper insight into the relationship between a target KPI and other parameters monitored during an experiment, and can also be downloaded for future predictions. In this case, we trained a linear regression model on the Throughput KPI after unnecessary features through the Feature Selection service were excluded. The model coefficients of the linear regression model are listed in Table 3.

**Table 3. Model coefficients of the trained linear regression model.**

Feature	Value
Medium Access Control (MAC) downlink Block Error Rate (BLER)	1.8745079925
MAC downlink BLER 1st	-2.2167396141
MAC uplink retransmission rate 3rd+	-0.0116183685
Packet Data Convergence Protocol (PDCP) downlink throughput	0.5055404854
Physical Downlink Shared Channel (PDSCH) throughput	0.1948166665
Radio Link Control (RLC) downlink BLER	-0.5718531061
RLC downlink throughput	0.2813455215
intercept	-0.0374639244

In order to gauge how effective the prediction model is, we inspect the model training results. The Analytics module displays several results for error rates, for testing the trained model on separate test data. The model can be tested in a single data split (e.g. 80/20 train/test split) and in a 10-fold cross validation, where 10 separate training/test splits are performed and the results averaged over all 10 folds to get a more unbiased result over the whole data. The MAE of 4.68 and Mean Square Error (MSE) of 46.74 reveal relatively high error rates, when taking into account that the range of the data is [0,50].





**Figure 27.** Plot of the actual vs predicted data points.

For a better picture of the model performance across the whole data range, the user can inspect the actual vs predicted data plot (Figure 27). An ideal plot would display a diagonal line of predicted vs actual data points. Here, it shows a scattered cloud of data points along the diagonal where the model performs ok but also a collection of points on the y-axis, where the model clearly failed to predict the correct value. The conclusion from the Throughput KPI prediction task is that the trained model performs better for data points above 5 Mbps but fails to predict data points at 0 Mbps (i.e., connectivity failures). Further investigation of the experiment data may be needed to identify the source of the unpredictable data points.

## 7. CONCLUSION

---

The instantiation of a M&A framework is key for 5G systems and testbeds, in order to derive data-driven insights on the functioning of the heterogeneous components composing such complex architectures, toward enabling ML/AI-driven management and optimization.

The 5GENESIS M&A framework positions itself as a clear enabler toward the above goals, and it is a key functional block in the 5GENESIS Reference Architecture and experimental pipeline, aiming at 5G showcasing and KPI validation in heterogenous scenarios and use cases.

In this deliverable we have reported the main activities performed in Task 3.3. for the development of the Release B of the 5GENESIS M&A framework in its three main components, i.e., Infrastructure Monitoring, Performance Monitoring, and Analytics.

As also highlighted by the integration activities with other architectural components (e.g., Slice Manager, policy engines, and Security Analytics), the 5GENESIS M&A framework results in an extremely flexible component that can be easily maintained and extended, toward supporting current functionalities while being used in a wide range of applications.

## REFERENCES

---

- [1] 5GENESIS, Deliverable D3.5 “Monitoring and Analytics (Release A)” [Online], [https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS\\_D3.5\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.5_v1.0.pdf), Accessed on: March 2021.
- [2] M. G. Kibria et al., “Big Data Analytics, Machine Learning, and Artificial Intelligence in Next-Generation Wireless Networks”, IEEE Access, vol. 6, 2018, pp. 32328–32338.
- [3] E. Pateromichelakis et al., “End-to-End Data Analytics Framework for 5G Architecture”, IEEE Access, vol. 7, 2019, pp. 40295–40312.
- [4] 5GENESIS, Deliverable D2.1 “Requirements of the Facility” [Online], [https://5genesis.eu/wp-content/uploads/2018/11/5GENESIS\\_D2.1\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2018/11/5GENESIS_D2.1_v1.0.pdf), Accessed on: March 2021.
- [5] 5GENESIS, Deliverable D2.4 “Final report on facility design and experimentation planning” [Online], [https://5genesis.eu/wp-content/uploads/2020/07/5GENESIS\\_D2.4\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2020/07/5GENESIS_D2.4_v1.0.pdf), Accessed on: March 2021.
- [6] 5GENESIS, Deliverable D6.1 “Trials and Experimentations (Cycle 1)” [Online], [https://5genesis.eu/wp-content/uploads/2019/08/5GENESIS\\_D6.1\\_v1.00.pdf](https://5genesis.eu/wp-content/uploads/2019/08/5GENESIS_D6.1_v1.00.pdf), Accessed on: March 2021.
- [7] 5GENESIS, Deliverable D3.15 “Experiment and Lifecycle Manager (Release A)” [Online], [https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS\\_D3.15\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.15_v1.0.pdf), Accessed on March 2021.
- [8] 5GENESIS, Deliverable D6.2 “Trials and Experimentations (Cycle 2)” [Online], [https://5genesis.eu/wp-content/uploads/2020/08/5GENESIS\\_D6.2\\_v1.0\\_FINAL.pdf](https://5genesis.eu/wp-content/uploads/2020/08/5GENESIS_D6.2_v1.0_FINAL.pdf), Accessed on: March 2021.
- [9] “Python client for InfluxDB” [Online], <https://github.com/influxdata/influxdb-python>, documentation at <https://influxdb-python.readthedocs.io/en/latest/>, Accessed on: March 2021.
- [10] 5GENESIS, Deliverable D3.3 “Slice Management (Release A)” [Online], [https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS\\_D3.3\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.3_v1.0.pdf), Accessed on: March 2021.
- [11] E. Rescorla, “HTTP Over TLS. RFC 2818 (Informational)” [Online], <http://www.ietf.org/rfc/rfc2818.txt> Updated by RFCs 5785, 7230, 2000, Accessed on: March 2021.
- [12] M. Belshe, R. Peon, and M. Thomson, “Hypertext Transfer Protocol Version 2 (HTTP/2)”, RFC 7540 [Online], <https://tools.ietf.org/html/rfc7540>, 2015, Accessed on: March 2021.
- [13] Hypertext Transfer Protocol Version 3 (HTTP/3), IETF Draft [Online], <https://quicwg.org/base-drafts/draft-ietf-quic-http.html>, Accessed on: March 2021.
- [14] M. Rajiullah et al., “Web Experience in Mobile Networks: Lessons from Two Million Page Visits”, in The World Wide Web Conference (WWW '19), ACM, New York, NY, USA, 2019, pp. 1532–1543.
- [15] M. Klausen, “Livestreaming Panoramic Video in Dense User Scenario,” Master Thesis, University of Oslo, 2020.
- [16] K.-J. Grinnemo et al., “NEAT Deliverable D3.3 – Extended Transport System and Transparent Support of Non-NEAT Applications,” Tech. Rep., 2017 [Online], <http://kau.diva-portal.org/smash/record.jsf?pid=diva2%3A1273851>, Accessed on: March 2021.

- [17] 5GENESIS, Deliverable D3.1 “Management and Orchestration (Release A)” [Online], [https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS\\_D3.1\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.1_v1.0.pdf), Accessed on: March 2021.
- [18] 5GENESIS, Deliverable D3.13 “Security Framework (Release A)” [Online], [https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS\\_D3.13\\_v1.0.pdf](https://5genesis.eu/wp-content/uploads/2019/10/5GENESIS_D3.13_v1.0.pdf), Accessed on: March 2021.

# ANNEX 1 – ADDITIONAL INFRASTRUCTURE MONITORING TOOLS

---

## Telegraf and Logstash integration in Berlin platform

When the Berlin platform incorporated a NetApp Hybrid Cloud Infrastructure (HCI) system, the need for a VMWare VSphere compatible monitoring solution arose. *“NetApp HCI is a hybrid cloud infrastructure that consists of a mix of storage nodes and compute nodes.”*<sup>31</sup> It combines NetApp and VMWare storage and compute virtualization technologies into a highly converged system. The Berlin platform team deployed Telegraf<sup>32</sup> to monitor various metrics from the HCI.

Telegraf is a metrics gathering software developed by Influxdata, the company which also maintains InfluxDB. It can be used to collect monitoring data and send it to an InfluxDB instance. It offers a plugin system, for which numerous plugins, integrating many different infrastructure components, already exist, including the VMWare VSphere plugin<sup>33</sup>. As part of the 5GENESIS infrastructure, an InfluxDB instance was already deployed in the Berlin platform, which eased the integration of Telegraf. A complete list of the metrics available for monitoring can be found online.<sup>34</sup> They include CPU, memory and network usage metrics at cluster, host, and virtual machine levels.

For the purpose of fault analysis, a centralized collection of logs can also play an essential role. Due to their heterogeneous nature, application logs require more specialized monitoring, compared to other infrastructure and performance metrics. To address this, the Berlin platform deployed logstash in conjunction with filebeat<sup>35</sup> to collect the logs from O5GC VNFs during phase three trials. *“Logstash is a server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to your favorite “stash.”*”<sup>36</sup> Filebeat allows the forwarding of log files from distributed systems to logstash, elasticsearch<sup>37</sup>, and others. Logstash allows the definition of pipelines for filtering of log data, using plugins. As an industry standard, logstash supports the syslog<sup>38</sup> protocol. Furthermore, it provides various output plugins, for forwarding the data to analysis and monitoring systems. Systems like elasticsearch and Apache Solr<sup>39</sup>, which can receive, store, and index such data, or monitoring systems like Zabbix and Nagios<sup>40</sup>. Another storage system supported via plugin, interesting in the 5GENESIS context, is InfluxDB.

---

<sup>31</sup> [https://docs.netapp.com/us-en/hci-solutions/hcvdivds\\_netapp\\_hci\\_overview.html](https://docs.netapp.com/us-en/hci-solutions/hcvdivds_netapp_hci_overview.html), Accessed on: March 2021.

<sup>32</sup> <https://www.influxdata.com/time-series-platform/telegraf/>, Accessed on: March 2021.

<sup>33</sup> <https://www.influxdata.com/integration/vmware-vsphere/>, Accessed on: March 2021.

<sup>34</sup> <https://github.com/influxdata/telegraf/blob/master/plugins/inputs/vsphere/METRICS.md>, Accessed on: March 2021.

<sup>35</sup> <https://github.com/elastic/beats>, Accessed on: March 2021.

<sup>36</sup> <https://github.com/elastic/logstash>, Accessed on: March 2021.

<sup>37</sup> <https://github.com/elastic/elasticsearch>, Accessed on: March 2021.

<sup>38</sup> <https://en.wikipedia.org/wiki/Syslog>, Accessed on: March 2021.

<sup>39</sup> <https://lucene.apache.org/solr/>, Accessed on: March 2021.

<sup>40</sup> <https://www.nagios.org/> Accessed on: March 2021.

The accumulated information can be leveraged by the M&A framework, which already operates on InfluxDB. Further integration with elasticsearch, Apache Solr, InfluxDB or RSyslog<sup>41</sup> is under investigation.

As described in Deliverable D3.5, MONROE VN and Remote Agents are the platform-agnostic PM probes that were developed for the entire 5GENESIS Facility during the first Project cycle, and thus included in M&A Release A. Furthermore, Android Agents for iPerf<sup>42</sup>, Ping, and File Transfer Protocol (FTP) performance monitoring were also developed and integrated. Below we describe the updates for these components during the second Project cycle.

---

<sup>41</sup> <https://www.rsyslog.com/>, Accessed on: March 2021.

<sup>42</sup> <https://iperf.fr>, Accessed on: March 2021.

## ANNEX 2 – WIFI MANAGEMENT AND MONITORING

The WiFi Slice Manager, as a whole component, is the main element which contains the business logic to monitor and manage the behavior of WiFi. In particular, it is composed by two different components to delegate these two main duties.

The WiFi Monitoring task is performed by the WiFi Slice Analytic and Monitoring (WSAM) component. This component is integrated with the WiFi Slice Controller (WSC) and monitors not only the network behavior, but also the status of each WiFi Slice. Figure A2-1 depicts the component ecosystem which monitors and manages the WiFi RAT.

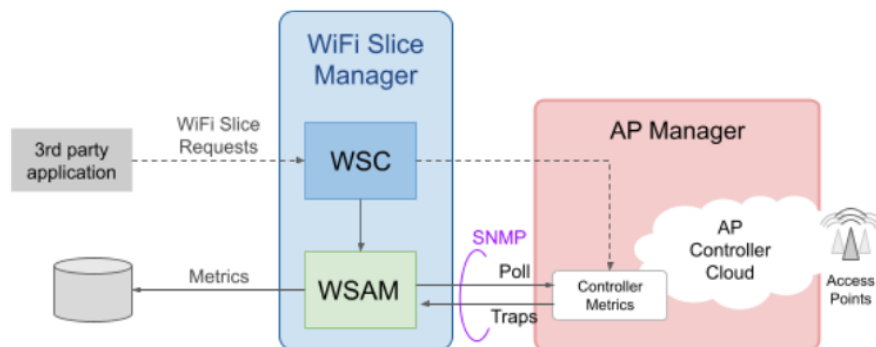


Figure A2-1. Main components of WiFi Monitoring and Management.

WiFi Management is performed by WSC. It receives requests from the 3rd party applications (i.e., the 5G Core of the Surrey platform) to create, deploy, shutdown and delete WiFi Slices. The WSAM component monitors both the WSC and the AP Manager using SNMP traps and requests. The collected information (metrics) is forwarded to a time series storage service, i.e., InfluxDB.

### WiFi Management

When the WSC receives a request to create or destroy a WiFi Slice, the WSC forwards the request to the AP Manager and then, accesses the WSAM API to perform the corresponding notification, including the request type (creation or destroy of a WiFi Slice), the WiFi Slice information (SSID, and internal UUID), and the timestamps of request and task execution.

As depicted in Figure A2-2, the WiFi Slice life-cycle is composed of three main states which can be adopted during the creation (transitions 1 to 3) and the destruction (transitions 4 to 5) of it:

- **Nonexistent:** The WiFi Slice does not exist. If the creation of a particular slice is requested, several operations are performed inside this state in order to instantiate the WiFi Slice to create;
- **Inactive:** The WiFi Slice exists but is not deployed. After being created, several operations to be deployed are performed. Once all the corresponding APs have the slice deployed, WiFi Slice status changes to available. If the slice has been undeployed, several operations to destroy the instance of the slice are being performed. Once all destruction operations end, the new status of the slice is nonexistent;

- **Available:** The WiFi Slice is running and deployed. Consequently, all the corresponding APs are projecting the signal of the slice. If the destruction of the slice is requested, inside this status several operations are performed in order to stop projecting the signal of the slice. Once all the corresponding APs have the slice undeployed the available status change to inactive;

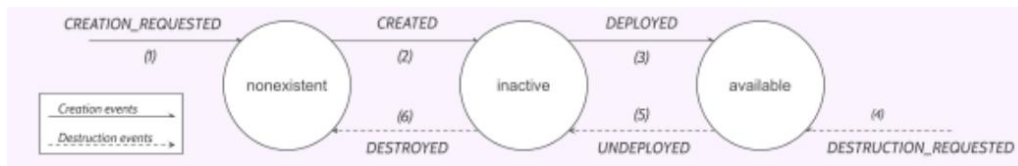


Figure A2-2. WiFi Slice life-cycle.

The transitions from one state to another are status events. These events are defined and listed chronologically:

- (1) CREATION\_REQUESTED: A creation request has been received in WSC to create and deploy a particular WiFi Slice. It triggers the creation of a new instance of a slice and enters in a nonexistent state. Thus, it is not created but it is being instantiated;
- (2) CREATED: All the instantiation operations have been performed. Consequently, the requested WiFi Slice has been created but is inactive and thus, its state changes from nonexistent to inactive;
- (3) DEPLOYED: The requested WiFi Slice has been deployed and is available, the corresponding APs are projecting the slice and stations can be connected to it. Consequently, the slice state changes from inactive to available;
- (4) DESTRUCTION\_REQUESTED: A destruction request has been received in WSC to undeploy and destroy a particular WiFi Slice. The undeployment starts and all the APs which are projecting the slice stop projecting it. It starts the destruction of a WiFi Slice when its current state is available;
- (5) UNDEPLOYED: The WiFi Slice has been undeployed and is not available. It still exists but is not projected in any AP. The slice state changes from available to inactive;
- (6) DESTROYED: The WiFi Slice state changes from inactive to nonexistent because of its destruction;

## WiFi Monitoring

The main aim of WSAM is to gather the information which describes the performance of networks deployed by WSC. In particular, the behaviour of each AP, connected station and WiFi Slice that compounds the networks managed by WSAM. In terms of performance, WSAM is focused on five baseline metrics:

- **Density of users (DoU):** Applied to the WiFi Radio Access Technologies (RAT), this baseline metric identifies the maximum number of user devices that can be connected throughout a WiFi Slice at one or many APs within an observed zone;



- **Delay (Del):** Current implementation of WSAM does not collect information from user devices and destination endpoints, just the metrics obtained in the APs, hereby this metric cannot be measured. However, it can be considered for future developments;
- **Service Creation Time (SCT):** The time required for the provision, deployment, configuration and activation of a full E2E communication service over a network slice, including all the physical and virtual components that are entailed in the Communication Service descriptors. Applied to the WiFi segment, the SCT is the elapsed time between a request to establish a new WiFi Slice is received, and the instant the service is completely up and running;
- **Throughput (Thr):** The taken metrics are collected at the deployed Access Points (APs). The measurement domain is from the connected User Equipment (UE) to the destination endpoint in both uplink and downlink traffic;
- **Reliability (Rel):** This baseline metric can be evaluated from general network performance metrics including APs availability, packet loss and retransmissions. As well as Throughput, the measurement domain is from the connected UE to the destination endpoint in both uplink and downlink traffic;

Please note that both Reliability and Throughput have a high dependency on the distance of user devices from the APs and the concrete network usage at the time of measurement. Consequently, the modelling of the deployed WiFi RAT under several combinations of these factors is pretty relevant to perform a better interpretation of the results.

### Measurement Perspectives

In order to calculate the baseline metrics, several measurements are gathered by monitoring not only WSC, but also the deployed APs and WiFi Slices. All this collected information is merged and processed to have a detailed overview of the network. The managed network can be seen by four main perspectives, as also shown in Figure A2-3:

- **WiFi Slice:** It is a perspective focused on the whole wireless network architecture that enables an independent logical network on the same physical network infrastructure (i.e. several access points inside a particular WiFi Slice which is managed by WSC);
- **AP:** This perspective refers to a particular access point which creates one or several Wireless Local Area Networks (WLANs) and projects them to its coverage area;
- **WLAN:** It refers to a whole Wireless Local Area Network. It aggregates all the information of a particular WLAN which is projected through all the corresponding APs;
- **WLAN-AP:** This perspective focuses on the disaggregation of the information of the whole Wireless Local Area Network. Thus, it covers the information of a particular AP which projects the signal of that particular WLAN;

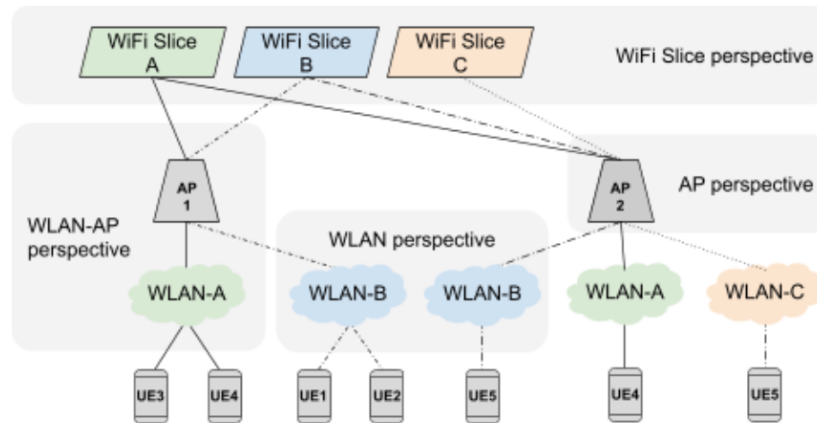


Figure A2-3. WiFi Slice Deployment Perspective.

All the measurements which compose the mentioned perspectives are taken following a snapshot approach. This approach refers to a copy of the values of several variables at a specific moment in time.

#### Data sources

In order to gather the networks' performance information, the following components are monitored to feed analytics.

- **WiFi Slice Controller:** The WSC collects the WiFi Slice event requests (e.g., create, deploy, destroy) and the respective occurrence timestamps, as well as the time references at which those requests have been fulfilled. All those metrics are then sent to WSAM;
- **Access Points:** The APs of the platform collect its own performance metrics (without distinction of the deployed WLANs within), including the number of connected stations, the transferred and received bytes, transmission fails, and other related metrics. All this information is then handled from the AP Manager to the WSAM component;
- **WLANs:** AP Manager collects, analogously to each AP as a whole, the relevant network performance metrics of each deployed WLAN indistinctly to the APs that offer its signal. Those metrics include the number of connected stations, the transferred and received bytes, information on transmission fails, and other related metrics. Again, all this information is then sent to WSAM periodically;

#### Monitoring techniques

The WSAM component provides monitoring of Simple Network Management Protocol (SNMP). SNMP is an 'agentless' method of monitoring network devices and servers, and is often preferable to installing dedicated agents on target machines. Thousands of different network devices and operating systems from different vendors support SNMP for delivering critical information on health and usage metrics, service state, and more. Furthermore, the WSAM component exposes a private channel (WSC-WSAM channel) to receive from WSC the events triggered by the status changes of WiFi Slice instances among their lifecycles.

Using SNMP and the WSC-WSAM channel, the WSAM component gathers all the analytics information applying two main techniques:

- **Passive Monitoring:** Registers changes in the APs' availability, WLAN creation, deployment and finalization for a defined zone. It receives data not only from the AP Manager, but also from WSC via WSC-WSAM channel. Using the explained technique, all the information related to the lifecycle of a WiFi Slice (i.e. creation, deletion, requests...) has the timestamp of WSC. Consequently, in order to not lose the time consistency and merge all the collected information both clocks (WSC and WSAM ones) must be perfectly synchronized;
- **Active Monitoring:** Periodical polling of available metrics obtained from the AP Manager. It periodically performs several SNMP queries to monitor the APs' measurements (i.e. availability, status and performance metrics), number of stations attached, and WLANs' measurements (i.e. performance metrics). These collected measurements are dynamically stored (i.e. RAM). Every 6 minutes, all the stored information is validated and then persisted to the configured database. The validation procedure consists of contrasting the actively collected metrics with the passive monitoring metrics (i.e. status change notifications in either APs or WLANs) within that 6-minute batch. For example, if any actively collected metric is outside the valid range limited by the passive metrics, then it is checked as 'non-valid'.

Regarding the frequencies of the polled measurements of the active monitoring techniques, two main phases can be depicted:

- **Dynamic storage phase:** As it was previously mentioned, the snapshot approach saves the current values of all polled metrics to the program memory each minute, as most of the polled metrics are updated at the AP Manager every minute;
- **Database persistence phase:** The previously saved snapshots are persisted every 6 minutes. Due to validation requirements, WSAM requires a minimum of 6 minutes to check if any alarm that invalidates the consistency of the saved snapshot has been received.

### Captured metrics

In order to feed the baseline metrics, several measurements were defined and grouped in the four exposed perspectives. Further details are defined for each metric in order to understand the taken measurements.

On the one hand, metrics taken by the passive method are stored inside the 'WiFi Slice events (WSEvents)' table (see Table A2-1). On the other hand, metrics taken by the active method are stored inside three main tables distinguished by the perspective of the snapshots which store: 'Access Point Snapshot (APSnapshot)', 'WLAN Snapshot (WLANSnapshot)' and 'WLAN per AP Snapshot (WLANPerAPSnapshot)' (Tables A2-2, A2-3, and A2-4).

The measurements focused on WiFi Slice mainly describes the events produced to change the current situation inside the life cycle for each managed WiFi Slice.

**Table A2-1: WiFi Slice events (WSEvents).**

Name	Description
wlanId	The unique Id of the WLAN
timestamp	The UNIX timestamp of the moment when the screenshot was taken.
statusEvent	The event occurred to change the state of the WLAN during the WiFi Slice lifecycle.

The metrics focused on a particular WiFi AP describes not only the main characteristics of it (e.g., uptime), but also the ones related to the generated traffic and the connected devices.

**Table A2-2: Access Point Snapshot (APSnapshot).**

Name	Description
apId	The unique Id of the AP.
timestamp	The UNIX timestamp of the moment when the screenshot was taken.
valid	Indicates whether the snapshot information has been checked as valid.
apIsConnected	Metric which represents whether the AP is connected.
apUptime	The number of milliseconds since the AP was connected
numConnectedStations	The number of devices connected to the AP.
apIpsecRxPkts	The number of packages received from the IPsec tunnel.
apIpsecTxPkts	The number of packages transmitted to the IPsec tunnel
apIpsecRXPktsDropped	The number of dropped packages received from the IPsec tunnel
apIpsecTXPktsDropped	The number of dropped transmitted sent to the IPsec tunnel

The metrics focused on a particular WLAN describes the aggregated measurements of all the transported traffic and the connected devices to all the APs which are projecting the WLAN signal.

**Table A2-3: WLAN Snapshot (WLANSnapshot).**

Name	Description
apId	The unique Id of the AP.
timestamp	The UNIX timestamp of the moment when the screenshot was taken.

ssid	The string of the projected SSID to deploy the WLAN.
valid	Indicates whether the snapshot information has been checked as valid.
wlanLastHourRxBytes	The number of received bytes in the last full hour.
wlanRawTxBytes	The number of transmitted bytes in the last full hour.
wlanSnapIncrRxBytes	Increment of received bytes since last snapshot.
wlanSnapIncrTxBytes	Increment of transmitted bytes since last snapshot.
wlanNumStations	The number of devices connected to the WLAN.
wlanUptime	The number of milliseconds since the WLAN was available.
wlanTotalTxFail	The number of packages transmitted which have been failed.

The metrics focused on a particular WLAN projected by a particular AP describes the disaggregated information of the generated traffic processed by an Access Point inside a WLAN.

**Table A2-4: WLAN per AP Snapshot (WLANPerAPSnapshot).**

Name	Description
apId	The unique Id of the AP.
wlanId	The unique Id of the WLAN.
timestamp	The UNIX timestamp of the moment when the screenshot was taken.
ssid	The string of the projected SSID to deploy the WLAN.
valid	Indicates whether the snapshot information has been checked as valid.
numConnectedStations	The number of devices connected to the WLAN in this AP.
rxBytes	The increment of received bytes for all devices connected to the WLAN in this AP, since the last saving batch (6 minutes).
txBytes	The increment of transmitted bytes for all devices connected to the WLAN in this AP, since the last saving batch (6 minutes).

## ANNEX 3 – IoT MANAGEMENT AND MONITORING

The IoT interoperable data are part of the Surrey platform and they are stored in a database in the OpenStack cloud platform. These data are essentially raw sensor data that can be later moved to InfluxDB and processed by the M&A framework. In order to make the IoT data available, a REST API was developed. To ensure the safety and security of the IoT data, the IoT API is only available through the Surrey VPN and it is also protected by HTTP basic authentication scheme requesting username and password for each and every request.

The IoT API can support many concurrent requests and it can serve all the IoT data that are stored in the database. It makes the process transparent to the data consumer and it can be used by every programming language thus making it very flexible. It supports five different API calls that will be explained in detail below. All API calls will be described and showcased with the Postman tool that is being used to develop and test APIs.

### Entities API Call

The Entities API Call (depicted in Figure A3-1) is a GET request that returns all the unique entities producing IoT data. In the example below (Figure A3-2), it currently returns five different results that are the IDs of the pysense sensors producing IoT data in the Surrey platform.

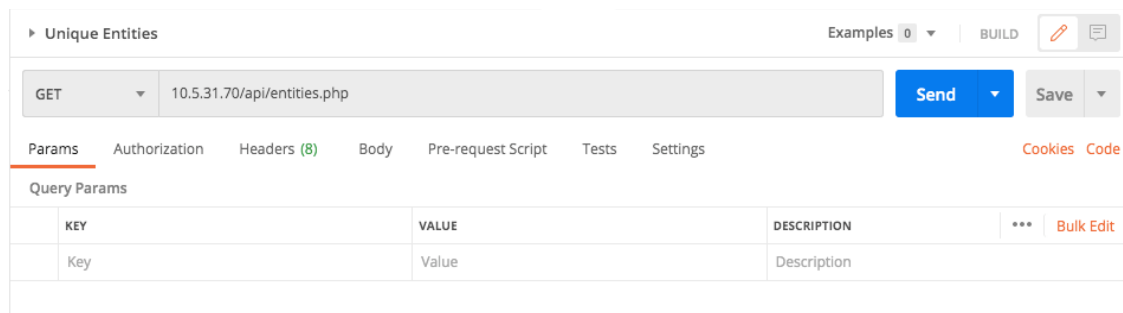


Figure A3-1. Entities IoT API Call.

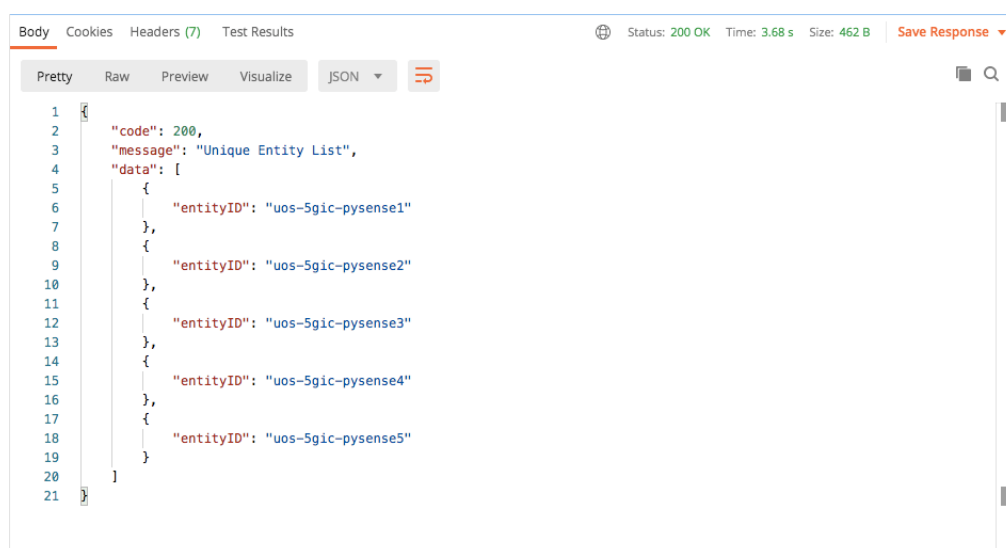


Figure A3-2. Results of Entities IoT API Call.

## Protocols API Call

The Protocols API Call (depicted in Figure A3-3) is a GET request that returns all the unique application level protocols that are used to transfer the IoT data from the sensors to the IoT vGateway. In the example of Figure A3-4, it returns HTTP, COAP, and MQTT.

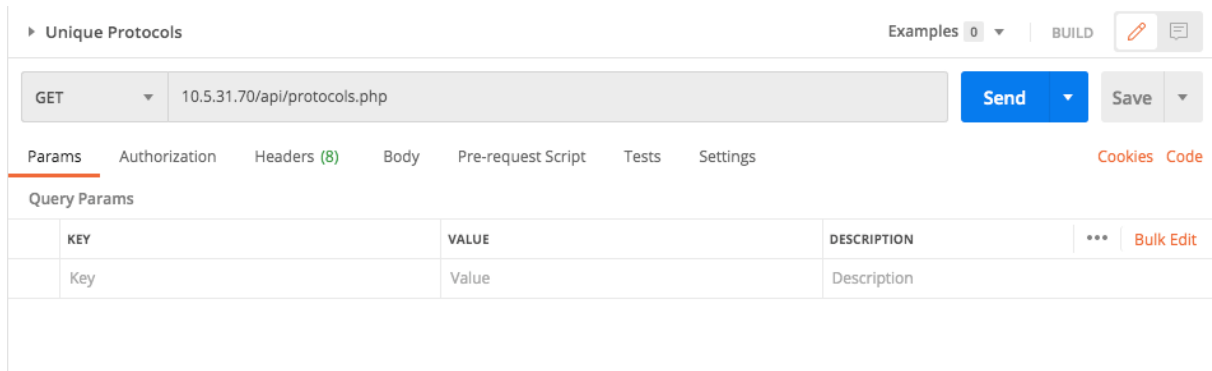


Figure A3-3. Protocols IoT API Call.

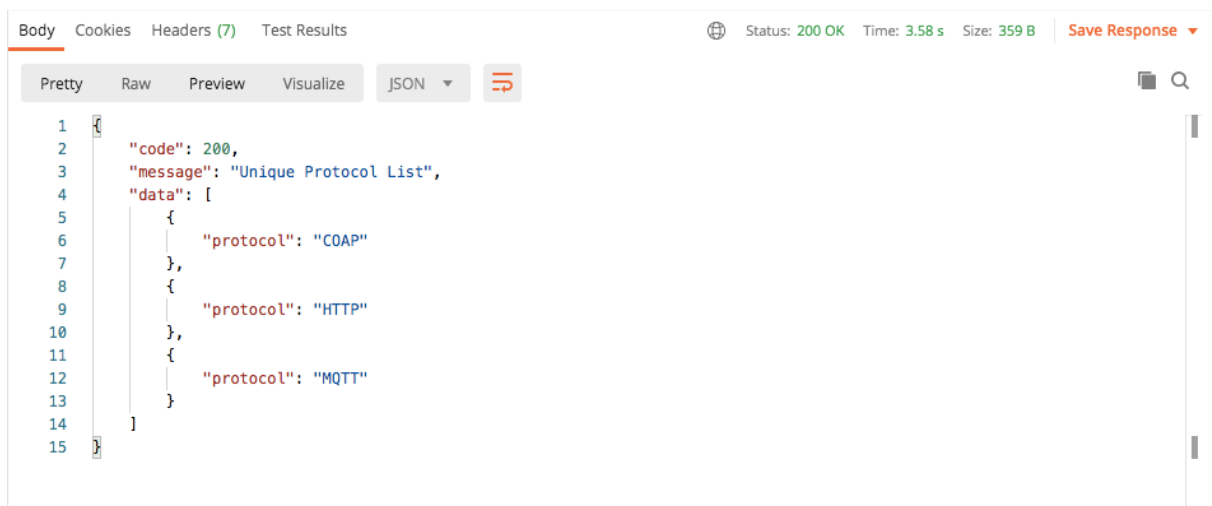


Figure A3-4. Results of Protocols IoT API Call.

## Datasources API Call

The Datasources API Call (depicted in Figure A3-5) is a GET request that returns all the wireless technologies that are used to transfer the IoT data from the sensors to the IoT vGateway. In the example below (Figure A3-6) it returns WiFi and LORA.

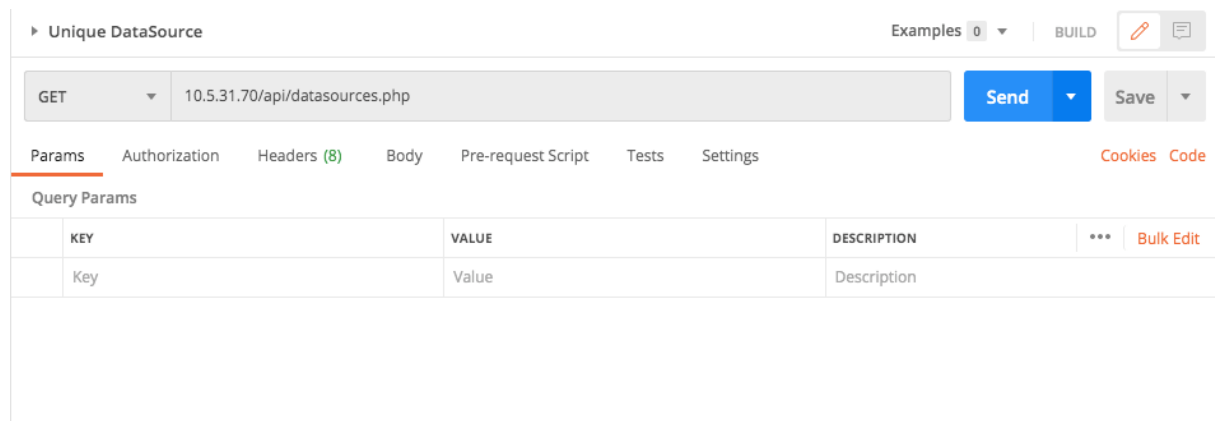


Figure A3-5. Datasources IoT API Call.

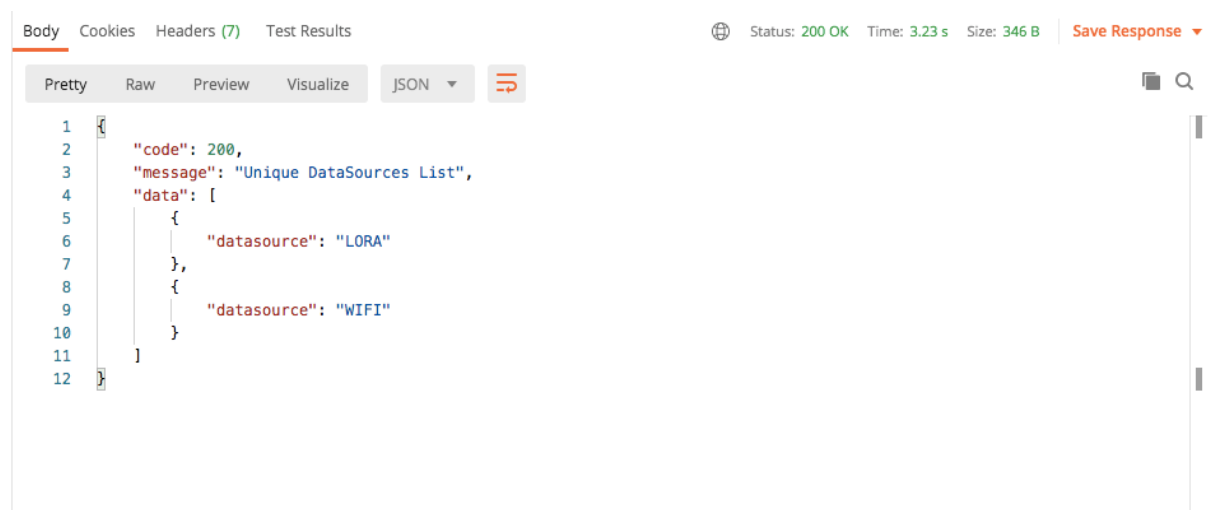


Figure A3-6. Results of Datasources IoT API Call.

## Latest Data API Call

The Latest Data API Call (depicted in Figure A3-7) is a GET request that returns the latest IoT data measurements. Several parameters can be defined in order to filter the results. These parameters are:

- entityID: ID of the sensor platform;
- protocol: Application protocol that the data were originating from before interoperability. Can be HTTP/COAP/MQTT;
- entityType: Type of sensor platform. For now, it is always pysenseBoard;
- entityOwner: Owner of the IoT sensor platform, currently UoS (University of Surrey);
- datatype: This field is for now always sensorData;
- limit: Number of records to be returned. If empty defaults to 10;
- dataSource: Technology that was used to transfer the IoT data to INFOLYSiS system. Can be WIFI/LORA/CELL.



Latest Data Examples 0 BUILD Send Save

GET 10.5.31.70/api/latestdata.php

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Code

KEY	VALUE	DESCRIPTION
entityID	uos-5gic-pysense1	ID of the sensor platform
entityType	pysenseBoard	Type of sensor platform. For now it is always pysens
entityOwner	UoS	Owner of the IoT sensor platform, currently UoS
dataType	sensorData	This field is for now always sensorData
limit	10	Number of records to be returned. If empty defaults
dataSource	WIFI	Technology that was used to transfer the IoT data to
protocol	COAP	

Figure A3-7. Latest Data IoT API Call.

Body Cookies Headers (7) Test Results Status: 200 OK Time: 354 ms Size: 6.91 KB Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "code": 200,
3   "message": "Latest Data List",
4   "data": [
5     {
6       "id": "3261833",
7       "entityID": "uos-5gic-pysense1",
8       "entityType": "pysenseBoard",
9       "entityOwner": "UoS",
10      "dataType": "sensorData",
11      "pressure": "102208.3",
12      "temperature": "33.27721",
13      "light": "35",
14      "humidity": "15.86203",
15      "proximity": "",
16      "gravity": [
17        "",
18        "",
19        ""
20      ],
21      "accelerometer": [
22        "-1.188596",
23        ""
24      ]
25    }
26  ]
27 }

```

Figure A3-8. Results of Latest Data IoT API Call.

In Figure A3-8, results are depicted from an API call. Each API call returns a json array of IoT data. Each IoT data has the following structure:

```

{
  "id": "3261833",
  "entityID": "uos-5gic-pysense1",
  "entityType": "pysenseBoard",
  "entityOwner": "UoS",
  "dataType": "sensorData",
  "pressure": "102208.3",
  "temperature": "33.27721",
  "light": "35",
  "humidity": "15.86203",
  "proximity": "",

```

```

    "gravity": [
        "",
        "",
        ""
    ],
    "accelerometer": [
        "-1.188596",
        "",
        "-0.735518"
    ],
    "pitch": "",
    "roll": "",
    "compass": [
        "",
        "",
        ""
    ],
    ],
    "timestamp": "2021-02-14 11:03:18",
    "latitude": "",
    "longitude": "",
    "protocol": "HTTP",
    "dataSource": "WIFI",
    "metadata": {
        "data_rate": "",
        "modulation": "",
        "airtime": "",
        "coding_rate": "",
        "frequency": "",
        "gtw_id": "",
        "gtw_timestamp": "",
        "gtw_altitude": "",
        "gtw_longitude": "",
        "gtw_rf_chain": "",
        "gtw_snr": "",
        "gtw_time": "",
        "gtw_latitude": "",
        "gtw_rssi": "",
        "gtw_channel": ""
    }
}

```

## Operate Data API Call

The Operate Data API Call (depicted in Figure A3-9) is a GET request that returns the IoT data measurements based on advanced operations such as various comparisons of different sensor measurements. The parameters that need to be defined for a successful API call are the following:

- limit: Number of records to be returned. If empty defaults to 100;

- measurement: Measurement type from IoT data. Can be pressure / temperature / light / humidity / accelerometer;
- operator: Operator to be used between measurement and value. Can be g (greater) / e (equals) / l (less);
- value: Threshold value of measurement.

In this example the results (Figure A3-10) contain the most recent 100 IoT data measurements of the temperature sensor that have a value greater than 33.08.

Operate Data

Examples 0 BUILD

GET 10.5.31.70/api/operatedata.php?limit=100&measurement=temperature&operator=g&value=33.08 Send Save

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> limit	100	Number of records to be returned. If empty defaults		
<input checked="" type="checkbox"/> measurement	temperature	Measurement type from IoT data. Can be pressure/temperature/light/humidity/accelerometer		
<input checked="" type="checkbox"/> operator	g	Operator to be used between measurement and value. Can be g (greater) / e (equals) / l (less)		
<input checked="" type="checkbox"/> value	33.08	Threshold value of measurement		
Key	Value	Description		

Figure A3-9. Operate Data IoT API Call.

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 302 ms Size: 39.93 KB Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "code": 200,
3   "message": "IoT Data List",
4   "data": [
5     {
6       "id": "3261873",
7       "entityID": "uos-5gic-pysense1",
8       "entityType": "pysenseBoard",
9       "entityOwner": "UoS",
10      "dataType": "sensorData",
11      "pressure": "102204.0",
12      "temperature": "33.28793",
13      "light": "38",
14      "humidity": "15.8544",
15      "proximity": "",
16      "gravity": [
17        "",
18        "",
19        ""
20      ],
21      "accelerometer": [
22        "-1.131553",
23        ""
24      ]
25    }
26  ]
27 }
```

Figure A3-2. Results of Operate Data IoT API Call.

## ANNEX 4 – DATA ANONYMIZATION

---

### Anonymizing Data for Publishing

#### Background

One of the main objectives of the Berlin Platform is to set up a multi-technology, multi-site E2E 5G environment for the evaluation of enhanced Mobile Broad Band (eMBB) and massive Machine Type Communication (mMTC) services, in a city testbed in the context of the yearly event “Festival of Lights”. During this event, tourists and bypassers can participate in a measurement campaign by logging into a 5GENESIS test network. To evaluate data collected without revealing private information about the data collectors, the use of differential privacy (DP) was promoted in Deliverable D3.5. DP is a technique that estimates the impact of single users to a specific database query and adds noise to the result to mitigate the identification of single users. This interference is calculated using a privacy budget  $\epsilon$ . Each interaction with the real data uses up some amount of  $\epsilon$ . A smaller  $\epsilon$  ensures that users’ data remains hidden, leading to more inaccurate results for the query.

#### Queries

For the Festival of Lights, a set of relevant queries have been identified in cooperation with the partners from the Berlin platform, i.e., Mean connection time, Mean distance to base station, Mean throughput over time, Mean packet loss rate over time, Mean power signal level over time, Categorize users based on velocity (pedestrian, bike, inside a car), and Coordinates of user over time/ density of users over time.

#### Mean

To answer the first five queries relevant in the context of the Festival of Lights, it is necessary to calculate a mean over a given statistical database without leaking private information.

#### Calculating DP Mean

To calculate a DP mean, a common algorithm requires a DP sum and a DP count (Ninghui Li, 2016). The sensitivity describes the impact of a single user. In case of a counting query, it is set to one, as the maximal contribution of a single user is one if they only appear once in the data set. To derive the sum, it is necessary to know which range of values are covered by the data. This allows to estimate the maximal impact a single user can have. For example, when calculating the mean age of a group of people, 0 and 120 are reasonable values for upper and lower bound. When no such domain knowledge is accessible, bounds can also be estimated using DP. One way to estimate bounds is to create non-overlapping buckets covering the whole domain of the real data. Then, DP is used to count the number of elements falling into each bucket. The resulting histogram illustrates the frequency of elements in buckets, which can be used to estimate the occurrences of specific values. To hide real minimum and maximum, buckets with a low count are ignored. This means the returned minimum corresponds to the lowest end of the bucket for which the DP count surpasses a certain threshold.

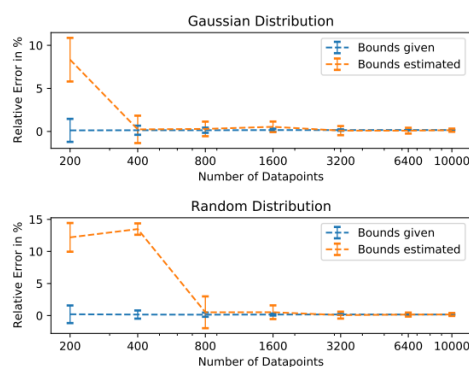
When calculating a mean, it is generally also useful to determine the confidence. To calculate confidence, parts of the privacy budget have to be used. This means doing so results in a noisier mean, since less budget is available for this step. The privacy budget can be split up in multiple

ways. It is for example possible to assign 90% of the budget to calculating the mean, while the rest can be used to compute the error bar.

### Implementation and Evaluation

To evaluate feasibility and utility of algorithms for DP with given bounds and estimated bounds, the query focusing on mean connection times of users was taken as example (see Query 1). As no data from real measurements was available, synthetic data was generated. With a Python simulator, two data sets with 10.000 data points were created. Both data sets have a different underlying distribution, one following a Gaussian distribution, the other being truly random. For an underlying Gaussian distribution, the true mean was set to be 5 min with a sigma of 2 min. We assume that users need to connect at least 10s so the connection is logged, and are automatically disconnected after 10 min.

For calculating the mean and estimating the bounds, the DP library of Google (Google, 2019) was used. For  $\epsilon$ , the default value of the library was used ( $\epsilon_{Default}=1.09861$ ). In some cases, the algorithm for estimating bounds failed because too little data was available. One solution to this problem was to decrease the success probability of the algorithm, thereby decreasing the quality of the estimated bounds. This was only necessary for measurements with random distribution containing less than 400 data points. In this case the success probability was set to  $1 \cdot 10^{-8}$ . For evaluating the utility, the relative error is used. Given the true answer  $v$ , the DP answer  $v_{DP}$  and a threshold  $\Theta$  the relative error is defined as:  $E_R = |v - v_{DP}| / \max(v, \Theta)$ .

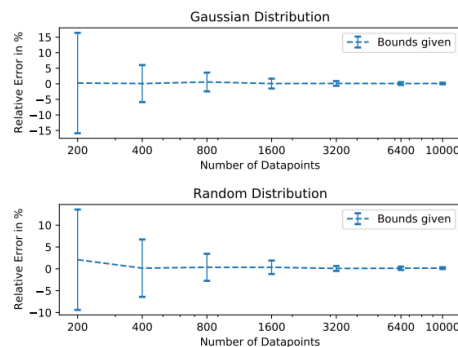


**Figure A4-1. Evaluation of utility regarding the mean connection time for two different underlying distributions.**

Figure A4-1 shows that, especially for a DP mean with given bounds (here 10s and 600s), the relative error is very small. If bounds have to be estimated, a part of the available privacy budget is used up for this operation. But for more than 400 data points in case of a Gaussian distribution and for more than 800 data points in case of a random distribution, the algorithm where a bound has to be estimated first performs as well as the algorithm where the bounds are given.

In case of the Festival of Lights, a single user does not only contribute a single measurement to the database but a set of measurements. The contribution of a single user to be greater than one would normally be expect. To account for this, it would be possible to calculate one mean per user and then only use these means to calculate the differentially private mean for the whole data set. Then, each user's impact is capped as it only contributes a single aggregated measurement. But this solution might not be applicable to all cases and deteriorates the result

as some users with little measurements might have the same impact as users with a large amount of measurements.



**Figure A4-2. The relative error for calculating a differentially private mean under the assumption that a single user can contribute  $k=10$  data points.**

We therefore use group theory, as described by Dwork and Roth (Cynthia Dwork, 2014), to counter the fact that one user corresponds to more than one data point. Group theory states that any DP mechanism protects a group of size  $k$  by scaling noise to  $\epsilon * k$ . For our evaluation, we set  $k$  to 10. This means, we assume that a single user contributes a maximum of 10 data points to the data set. This assumption can also be enforced by removing data points for users that exceed the threshold. In Figure A4-1 it is shown that, for the example, an algorithm with pre-set bounds performs better than one with estimated bounds. Figure A4-2 displays the results for a both distributions with bounds set in advance and  $\epsilon = \epsilon_{Default}/k$ . It shows that to achieve good utility for user data consisting of multiple sequential measurements, a large number of users is required to balance out the effect.

## Classification

Not all queries relevant for the context of the Festival of Lights require the calculation of a mean. Query 6, which asks for counts of users based on velocity, needs to be answered using histograms. Histograms require a set of counting queries, which are also necessary to derive a DP mean to find out how many data points fall into each category.

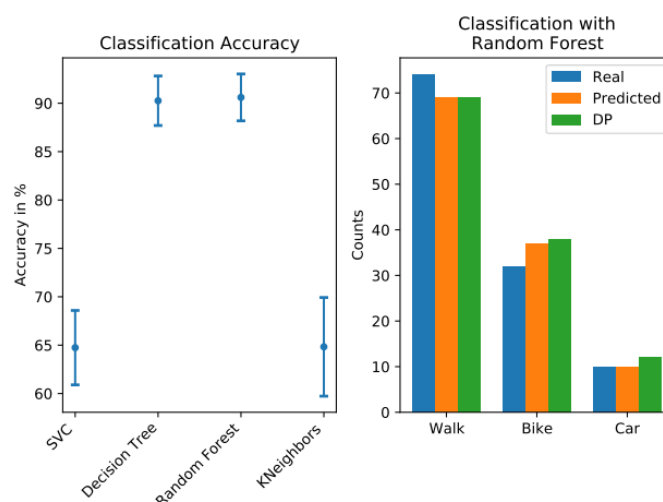
## Approach

While it is possible to automatically detect the types of relevant categories for a histogram by using parts of the privacy budget for this purpose, it is not relevant in the context of Query 6. We assume that a user only uses a method of transport that falls into one of these three categories: by foot, by bicycle or by car. Means of transport can be distinguished by analyzing various attributes of a GPS trajectory. One approach is based on machine learning (Xue Yang, 2018). Here, techniques such as support vector machines (SVM), decision trees, random forests and k-neighbor classifiers can be used. Attributes like velocity, acceleration, heading, movement patterns and tortuosity can be used in various combinations to differentiate. Users can change their means of transport so it is important to correctly detect such changes. This can be accomplished by splitting the trajectories and classifying individual parts separately (Xue Yang, 2018). To release the result of the classification, a histogram can be used. This requires one DP counting query per category. In the case of Query 6, the category itself does not leak information as they are preset.

## Implementation and Evaluation

A data set commonly used for evaluation in the context of transport mode detection is Microsoft's GeoLife data set (Microsoft, 2012). It contains GPS trajectories by 182 users which were collected over a period of three years and are partially labeled. Algorithms for machine learning are available in the Sklearn library in Python. For evaluation, the code from (DABIRI, 2019) was used as basis.

Before the data was usable, it had to be sanitized. First, outliers and errors were removed. Only trajectories with labels could be utilized to determine the accuracy of the classification algorithm, which reduced the set. Next, trajectories that contain less than 20 locations, are shorter than 60s or travel less than 50 meters were excluded. Trajectories consisting of more than 100 data points were split into multiple ones. Before they can be used, all trajectories had to be padded to the same length. The data set contains the transport modes walk, bike, bus, car, subway, train, airplane, boat, run and motorcycle. Of these, only the modes walk, bike and car were relevant. This left 233 trajectories, of which 143 were labeled as walk, 66 had the label bike and 24 were collected inside a car. For each trajectory the following features were extracted: traveled distance, average velocity, expected velocity, variance of velocity, maximal velocity, maximal acceleration, heading change rate, stop rate as well as the number of changes in velocity. These tuples were then used as input to the algorithms. For training, a fraction of indices was drawn from the set of inputs. For the remaining ones, the label was predicted.



**Figure A4-3.** Left: a comparison of various ML alternatives for transport mode detection on the GeoLife data set. Right: an example histogram is created comparing real, predicted and DP results.

The accuracy of the tested Sklean algorithms is shown on the left of Figure A4-3. To derive the standard deviation, 10 independent runs were conducted per algorithm. Also evaluated were support vector machines and multi-layer perceptrons, but these two are not shown since prediction rates were less than 60%. Both the decision tree and the random forest perform well so they can be used for classification. An example for how results predicted by a random forest classifier and the corresponding DP counts is presented on the right side of Figure A4-3. DP counts were derived using the Diffprivlib with a privacy budget of  $\epsilon_{Default}=1.09861$ .

This evaluation shows that both random forest and decision tree have an accuracy of ~90%. However, it has been argued that such techniques require well set bounds that can be difficult to establish and methods which learn the bound between different classes automatically can

have a better performance (Sina Dabiri, 2019). Various types of neural networks such as deep learning with auto-encoders or convolutional neural networks fall into this category. Another method to increase accuracy is combining GPS data with information about the public transit network or locations where people change between means of transports (Xue Yang, 2018).

## Histograms of Location Data

The distribution of locations of users is important to identify clusters that might have influenced availability of the 5G network at certain locations. Query 7, which focuses on this aspect, has different privacy requirements compared to the others as the data to be presented is two-dimensional and consists of sequences of locations which vary for each user.

### Approaches

One approach for visualizing user density are heat-maps. To compute a heat map, the covered space is split into areas. For each area a DP count of users is calculated. Since we are interested in the distribution of users, the counts have to be normalized. This way, a 2D histogram is created. There are different ways to determine how these areas for the histogram are composed. Setting a fixed size of each area does certainly not leak any data. But it might also not be optimal for sparse data, as few points fall into the same area which in return would cause a large signal-to-noise ration. In Su et al. (Dong Su, 2019) and in the book by Li et al. (Ninghui Li, 2016), different kinds of adaptive grids are discussed depending both on the number of data points and  $\epsilon$ . The M-estimator algorithm splits a given two-dimensional space

into  $M$  equal-sized rectangles. The number of rectangles is calculated using  $M = \frac{N}{\sqrt{\log(N)}}$ ,

where  $N$  represents the number of real data points. The Uniform Grid (UG) partitioning scheme works similarly as it also produces equal sized areas. Here, the number of areas is given by

$M = \frac{N\epsilon}{10}$ . Unlike the M-estimator, the UG algorithm takes the selected  $\epsilon$  value into account.

To visualize the development of user distribution over time, multiple 2D-histogramms can be combined. Using DP stops a potential attacker from using changes of counts of a normal histogram to estimate the movement of single users. To account for movement of users in groups the sensitivity can be increased.

### Implementation and Evaluation

To create two-dimensional histograms, the Diffprivlib by IBM (IBM, 2019, Naoise Holohan): was used. If no bounds of the data in each dimension are given, they can be estimated. For this purpose, calls were made to the ApproxBounds algorithm of the Google DP library. As data set the NSCU Mobility Models (Injong Rhee, 2009) was used. This data set consists of GPS Traces from five different sites collected by 4-8 volunteers between 2006 and 2008. For the evaluation, we chose the NC State Fair traces (see Figure A4-4), as it was recorded during an outdoor event similar to the Festival of Lights and covers a confined area. Since these traces were collected by only a small number of volunteers who walked around the premises, we ignored attribution of traces to individuals and timestamps.



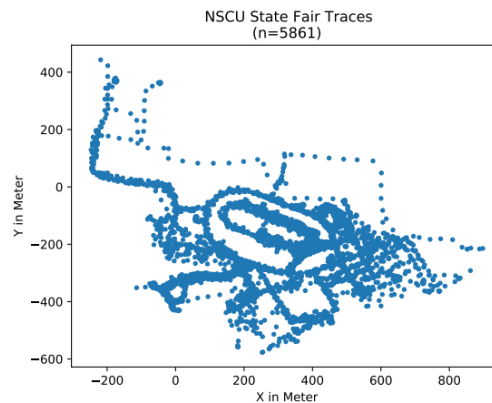


Figure A4-4. GPS Data of the NSCU State Fair data set.

First, let's assume that data points are completely unconnected. The first two plots of Figure A4-5 display DP histograms of the two described partitioning schemes. To derive the general area to plot, the minimal and maximal coordinates were estimated using DP. For the privacy budget  $\epsilon$  was set to the default value used in Section "Implementation and Evaluation". Of this budget 50% were used to estimate the bounds. To evaluate the utility of both histograms, the relative error when compared to non-DP histograms with same partitioning is presented in the second row of Figure A4-5. As some areas of the non-DP histogram have a true count of zero, the threshold becomes important. We set this threshold to  $\Theta=1/n$ .

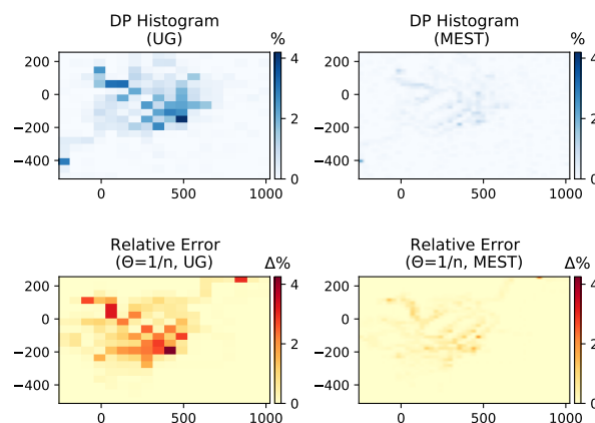


Figure A4-5. Comparing the M-Estimator ( $M=45^2$ ) and the Universal Grid ( $M=18^2$ ) partitioning schemes on DP histograms.

As we can see, both types of partitioning have a small relative error and represent the general distribution of the real data. The M-estimator partitioning scheme allows for a more fine-grained resolution. Using the symmetric Kullback–Leibler divergence, which is a measure for the similarity of two distributions, we see that the Uniform Grid algorithm performs better in the order of one magnitude ( $KL_{Mest}=0.11$  compared to  $KL_{UG}=0.015$ ).

Next, we need to account for the fact that a user might contribute multiple data points to the histogram. Again, we assumed that a user collected a maximum of  $k=10$  data points and set  $\epsilon=\epsilon_{Default}/k$ . To minimize error and make better use of the available privacy budget, the UG partitioning was used. Less areas result in less counting queries which leaves more privacy budget for each query. In Figure A4-6 we clearly see that having a lower  $\epsilon$  greatly impacts the quality of the estimated bound and deteriorates utility as the relative error becomes larger.

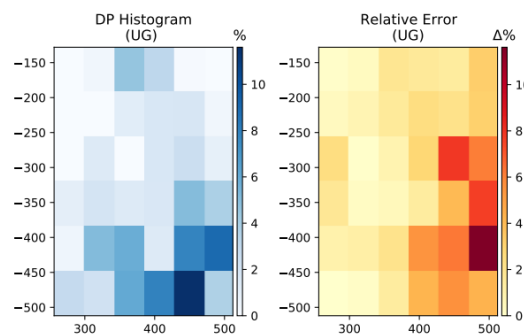


Figure A4-6. DP histogram and relative error with UG ( $M=6^2$ ) and setting  $k=10$ .

Query 7 mostly aims at determining the areas with the largest user density at a given time to evaluate the network load. For this purpose, simply returning centers of clusters would suffice. A common approach for calculating such clusters is K-Means. This algorithm starts with  $k$  centroids. At every iteration, all data points are assigned to their closest centroids. The location of these centroids is then corrected to minimize the overall distance to their assigned points. This process is repeated until it converges. Various versions of DP K-Means are discussed by Su et al. (Dong Su, 2019). Best suited for this scenario is the non-interactive Extended Uniform Grid k-Means (EUGkM) algorithm or simply called UGkM when dealing with two-dimensional data. Here, the area is split into equally sized cells. For each cell the contained points are counted using DP. Then, the K-Means algorithm is applied using the sanitized data. Important for the outcome of K-Means is the selection of the initial centers. Since the data used by the algorithm is differentially private, it is possible to simply restart clustering with different centroids without leaking information. Due to the workflow of the EUGkM algorithm it is similar to the heat-maps mentioned earlier. Both methods could be combined to enhance the results.

## Discussion

There is active research on the topic of anonymization and many possible approaches to problems discussed in the sections above have not yet found their way into the big libraries for DP. In this section we discuss some of these approaches.

### Hierarchical Approach to Histograms

Instead of splitting a 2D space into a grid and using one counting query for each area, it is also possible to use a hierarchic approach for deriving histograms. Here, at different levels of granularity individual counting queries are executed. The result is a tree structure with  $h$  levels. The DP private responses at different levels of the tree can be used to improve accuracy for example by weighted averaging of counts from leaves to root. Another solution which works in the opposite direction is enforcing the consistency of the mean. But it has been found that for a lower number of areas, this approach does not perform better than a simple histogram. To be more precise, it is better to use a simple approach to histograms if less than 45 bins are used in the one-dimensional case or less than 4096 bins in the two-dimensional case (Ninghui Li, 2016). In case of location histograms with  $M$ -estimator, only 2025 bins were used. The data utilized in the evaluation was collected on an area larger than what will be covered by the Festival of Lights. It is therefore unlikely that using a hierarchical approach to histograms will perform better.

## Computing DP Trajectories

The goal of Query 7 is to graphically represent mass movements and animate the resulting plot to highlight temporal changes. One tool for achieving a DP for spatial data is geo-indistinguishability proposed by Andrés et al. (Miguel E. Andrés, 2013). The exact position  $x$  of a user is obscured by creating a circular area with radius  $r$  around  $x$ . A value  $z$ , which is located inside the circle, is returned instead of  $x$ . Knowing  $z$  improves the attacker estimate for  $x$  at most by  $e^l$ . Here,  $l$  is called the level of discrepancy and is similar to the privacy budget. This mechanism is slightly different from general differential privacy as user contribution is not completely hidden but approximate location data is revealed. Geo-indistinguishability has sometimes been criticized to have bad utility, since the distance between reported location and actual location can be rather large (Jian Wang, 2019).

Xiao and Xiong (Xiao & Xiong, 2015) present a mechanism to compute a DP alternative for a current location which takes the history of data points into account. Here, a Markov model is used to compute a number of probable locations for a user at a given time. One of these locations is reported and then used by the Markov model to adapt the probabilities for the next step. However, methods focused on obscuring the location or trajectory of a single user like those mentioned above suffer a common weakness. By design these mechanisms leak the number of participants. A simple solution for this problem is to use only a fraction of the available data. Both approaches above focus on perturbing a single user's collected data and are designed for the use case of location-based services. But the scope of the Festival of Lights is to analyze the development of user distribution, meaning the single user is not relevant. One solution of this problem is to create new trajectories from the given data set which are inherently private. Chen et al. (Rui Chen, 2012) presented an algorithm which applies DP to sequential data by using an  $n$ -gram model. Here, variable length  $n$ -grams of trajectories are stored in an exploration tree. Transitions from parent location to leave location are modeled using DP counts. If the parent's real count is small, so will be the count of the child. Using this approach, the most common location sequences can be derived. It has been criticized that this mechanism is not applicable to realistic databases, as it does not scale well to larger geographical areas. The approach by He et al. (Xi He, 2015) also synthesizes mobility data by using a prefix tree and Markov model. The space is partitioned multiple times with different resolutions. For each resolution one prefix tree is created which represents movement and transition probabilities at a specific speed. Transitions can only occur to neighboring areas. To ensure privacy, noise is applied to the nodes of the prefix trees. New trajectories are created by drawing the next possible location from the forest consisting of prefix trees. First, a reference system is selected and then a possible location is sampled by taking the last few locations on the artificial trajectory into account. A trajectory ends when a stopping symbol is sampled. In their evaluations He et al. compared their work to the one of Mohammed et al. (Mohammed, Chen, Fung, & Yu, 2011) and found that it outperforms regarding the quality of generated trajectories, but report a similar score for detecting frequent patterns.

## References

- (Sarwate, 2016): "DP-Stats" [Online], <https://gitlab.com/dp-stats/dp-stats>, Accessed on Feb. 2021
- (Ninghui Li, 2016): N. Li et al., "Differential Privacy: From Theory to Practice", Morgan & Claypool Publishers, 2016

(Google, 2019): "Differential Privacy" [Online], <https://github.com/google/differential-privacy>, Accessed on Feb. 2021

(Cynthia Dwork, 2014): C. Dwork, A. Roth, "The Algorithmic Foundations of Differential Privacy", Foundations and Trends in Theoretical Computer Science, 9/3-4, 08/2014, pp. 211--407

(Xue Yang, 2018): X. Yang et al., "A review of GPS trajectories classification based on transportation mode", Sensors, 18/11, 2018/11, pp. 3741

(Microsoft, 2012): "The Geolife Dataset" [Online], <https://www.microsoft.com/en-us/download/details.aspx?id=52367>, Accessed on Feb. 2021

(DABIRI,2019): "Deep-Semi-Supervised-GPS-Transport-Mode" [Online], <https://github.com/sinadabiri/Deep-Semi-Supervised-GPS-Transport-Mode>, Accessed on Feb. 2021

(Sina Dabiri, 2019): D. Dabiri et al., "Semi-supervised deep learning approach for transportation mode identification using GPS trajectory data", IEEE Transactions on Knowledge and Data Engineering, 32/5, 02/2019, pp. 1010-1023

(Dong Su, 2019): D. Su et al., "Differentially private k-means clustering", Proceedings of the sixth ACM conference on data and application security and privacy, New Orleans Louisiana, USA, 03/2016, pp. 26-37

(IBM, 2019): "Diffprivlib" [Online], <https://github.com/IBM/differential-privacy-library>, Accessed on Feb. 2021

(Naoise Holohan): N. Holohan et al., "Diffprivlib: The IBM Differential Privacy Library", CoRR, abs/1907.02444, 07/2019

(Injong Rhee, 2009): "CRAWDAD dataset ncsu/mobilitymodels (v. 2009-07-23), traceset: GPS" [Online], <https://crawdad.org/ncsu/mobilitymodels/20090723/GPS>, Accessed on Feb. 2021

(Miguel E. Andrés, 2013): M.E. Andrés et al., "Geo-indistinguishability: Differential privacy for location-based systems", Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, Berlin, Germany, 11/2013, pp. 901-914

(Jian Wang, 2019): J. Wang et al., "Location protection method for mobile crowd sensing based on local differential privacy preferenc", Peer-to-Peer Networking and Applications, 12/5, 07/2019, pp. 1097-1109

(Xiao & Xiong, 2015): Y. Xiao, L. Xiong, "Protecting locations with differential privacy under temporal correlations", Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver Colorado, USA,10/2015, pp. 1298-1309

(Rui Chen, 2012): R. Chen et al., "Differentially private sequential data publication via variable-length n-grams", Proceedings of the 2012 ACM conference on Computer and communications security, Raleigh North Carolina, USA, 10/2012, pp. 638-649

(Xi He, 2015): X. He et al., "DPT: differentially private trajectory synthesis using hierarchical reference systems", Proceedings of the VLDB Endowment, 8/11, 06/2015 pp. 1154-1165

(Mohammed, Chen, Fung, & Yu, 2011): N. Mohammed et al. "Differentially private data release for data mining", Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, San Diego California, USA, 08/2011, pp.493-501

## ANNEX 5 – MONROE VN

### Example of MONROE VN integration and usage in Berlin platform

As described in Section 3, a number of probes and enhancements were performed during the second project cycle for the purpose of performance monitoring in the 5GENESIS platforms. In this annex, we describe the updates and proposed experimental setup that would be used for conducting the performance monitoring experiments between MONROE VN and O5GC, with MONROE VN used for the traffic generation and subsequent collection of the results and analysis. In the first project cycle, Open Baton<sup>43</sup> was used as the NFV MANO solution for the Berlin platform. However, with Open Baton Release 6, it was not possible to retrieve the Internet Protocol (IP) addresses of the installed VNFs. This inhibited the use of MONROE VN as a PM probe. During the second project cycle, the Berlin platform migrated to the Open Source MANO<sup>44</sup> (OSM) solution. OSM Release 7 provides the necessary API to retrieve and update the IP addresses of the instantiated VNFs. Thus, two PM E2E experiments were proposed during the second project cycle. The experiments illustrated in Figure A5-1 and Figure A5-2 are as follows:

- **E2E Experiment 1** – MONROE VN probes stream the performance monitoring traffic through an emulated UE that is connected to an emulated 5G Base Station (gNB) of the O5GC. The emulated UE acts as the gateway for the MONROE VN PM experiments.

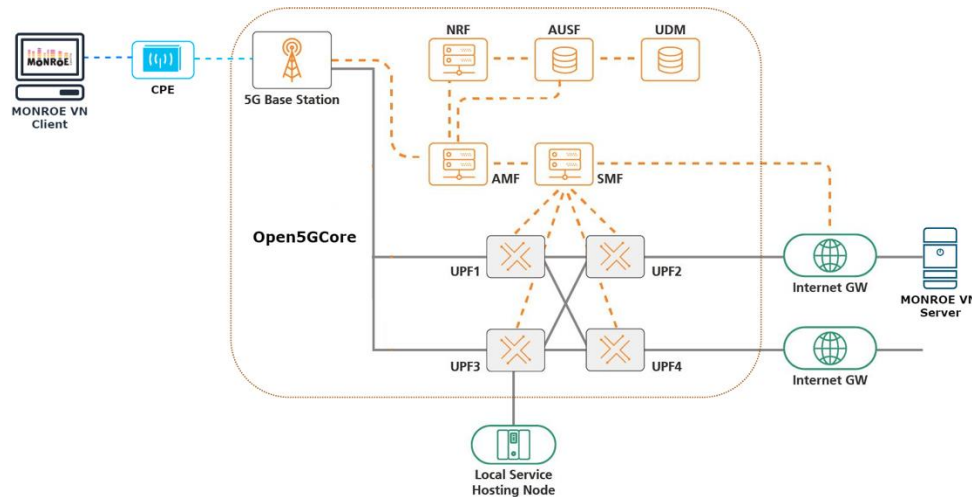


Figure A5-1. E2E Experiment 1 – MONROE performance monitoring traffic through Emulated UE.

- **E2E Experiment 2** – In this case, MONROE VN streams traffic through an actual Customer Premises Equipment (CPE) connected to a gNB. The CPE acts as the gateway for the MONROE VN probes.

<sup>43</sup> <http://openbaton.github.io>, Accessed: March 2021.

<sup>44</sup> <https://osm.etsi.org>, Accessed: March 2021.

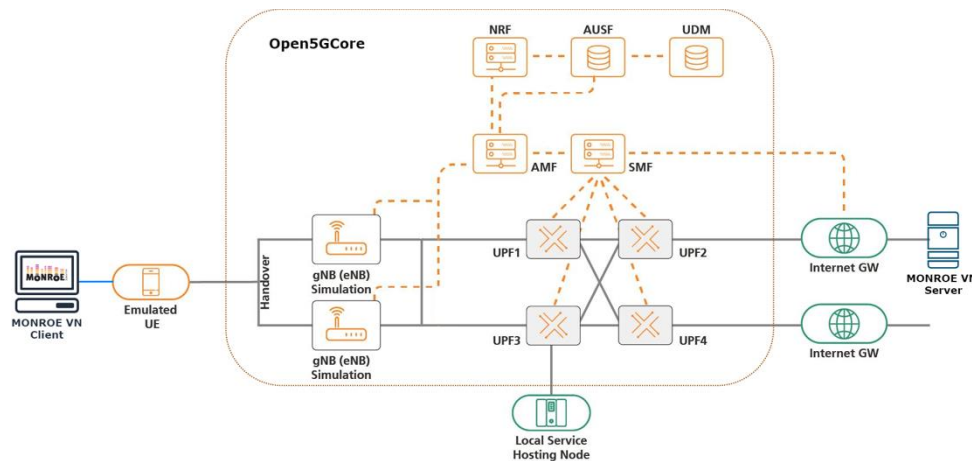


Figure A5-2. E2E Experiment 2 – MONROE performance monitoring traffic through CPE.

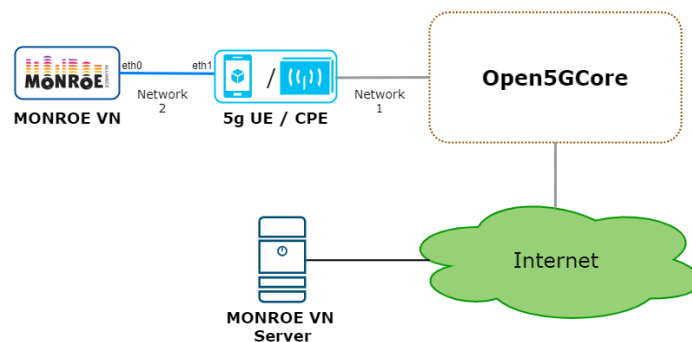


Figure A5-3. MONROE 5G Emulated UE / CPE Gateway setup with Open5GCore.

In both E2E experiments, the MONROE VN relies on either the emulated 5G UE or the CPE to act as its gateway, as illustrated in Figure A5-3. Thus, additional steps are followed in order to achieve this. In the case of the 5G emulated UE acting as the gateway, the steps are as follows:

- Install the emulated 5g UE and the MONROE VN in different Virtual Machines (VMs);
- Configure the network (Network 1) between the emulated UE and the O5GC;
- Configure the network (Network 2) on the emulated UE (eth1) with a static IP to act as the gateway for all MONROE VN on the network (eth0);
- Initialize the emulated UE and update its configuration file (command module) to listen on the gateway IP address for receiving commands and then start the emulated UE service;
- Configure the emulated UE as the gateway to allow forwarding of traffic from Network 2 to the O5GC (Network 1). To achieve this, we will need to masquerade the traffic received on eth1 with emulated UE's IP address received from the O5GC. The following system command is used to achieve this result:

```
# iptables -t nat -A POSTROUTING -o pdu1 -j MASQUERADE
```

- To trigger registration and deregistration of the emulated UE from MONROE VN, the following commands can be used:  
For registration, MONROE VN would use:

```
curl -v -X POST -H "Content-Type: application/json-rpc" -d '{"jsonrpc": "2.0", "method": "ue5g.register", "params":{"access_type":1}, "id": 1}' http://<gateway_ip>:10010/jsonrpc
```

For deregistration, MONROE VN would use:

```
curl -v -X POST -H "Content-Type: application/json-rpc" -d '{"jsonrpc":  
"2.0", "method": "ue5g.deregister", "id": 1}'  
http://<gateway_ip>:10010/jsonrpc
```

These commands would facilitate MONROE VN to execute its PM experiments through the emulated 5G UE acting as the gateway.

The steps required to configure the CPE to act as the gateway for MONROE VN depends on the functionalities provided by the CPE.

## Example of outputs from the Throughput container

### "iperf2"

```
{  
  "DataId": "5GENESIS.EXP.IPERF",  
  "DataVersion": 2,  
  "Guid": "sha256:a4b55ff5a8893c2e267394fd6481a7908e0a7dd9a48d6a29458104b411712ff9.test-iperf2.7.1",  
  "Interface": "eth0",  
  "NodeId": "7",  
  "Protocol": "tcp",  
  "Results.bits_per_second": "809884422",  
  "Results.destination_address": "192.168.100.13",  
  "Results.destination_port": "5001",  
  "Results.interval": "0.0-10.0",  
  "Results.source_address": "172.18.3.2",  
  "Results.source_port": "52977",  
  "Results.timestamp": "20190301131852.883",  
  "Results.transferID": "3",  
  "Results.transferred_bytes": "1012662272",  
  "Timestamp": 1551446332.883225  
}
```

### "iperf3"

```
{  
  "DataId": "5GENESIS.EXP.IPERF",  
  "DataVersion": 3,  
  "Guid": "sha256:a4b55ff5a8893c2e267394fd6481a7908e0a7dd9a48d6a29458104b411712ff9.test-iperf3.7.1",  
  "Interface": "eth0",  
  "NodeId": "7",  
  "Protocol": "tcp",  
  "Results.end.cpu_utilization_percent.host_system": 1.269514,  
  "Results.end.cpu_utilization_percent.host_total": 1.376223,  
  "Results.end.cpu_utilization_percent.host_user": 0.119017,  
  "Results.end.cpu_utilization_percent.remote_system": 0.000918,  
  "Results.end.cpu_utilization_percent.remote_total": 0.001014,  
  "Results.end.cpu_utilization_percent.remote_user": 9.5e-05,  
  "Results.end.streams.0.receiver.bits_per_second": 810489767.650944,  
  "Results.end.streams.0.receiver.bytes": 1013134794,  
}
```



```
"Results.end.streams.0.receiver.end": 10.000223,
"Results.end.streams.0.receiver.seconds": 10.000223,
"Results.end.streams.0.receiver.socket": 4,
"Results.end.streams.0.receiver.start": 0,
"Results.end.streams.0.sender.bits_per_second": 812731463.278758,
"Results.end.streams.0.sender.bytes": 1015936976,
"Results.end.streams.0.sender.end": 10.000223,
"Results.end.streams.0.sender.max_rtt": 11932,
"Results.end.streams.0.sender.max_snd_cwnd": 1234434,
"Results.end.streams.0.sender.mean_rtt": 8559,
"Results.end.streams.0.sender.min_rtt": 3960,
"Results.end.streams.0.sender.retransmits": 5,
"Results.end.streams.0.sender.seconds": 10.000223,
"Results.end.streams.0.sender.socket": 4,
"Results.end.streams.0.sender.start": 0,
"Results.end.sum_received.bits_per_second": 810489767.650944,
"Results.end.sum_received.bytes": 1013134794,
"Results.end.sum_received.end": 10.000223,
"Results.end.sum_received.seconds": 10.000223,
"Results.end.sum_received.start": 0,
"Results.end.sum_sent.bits_per_second": 812731463.278758,
"Results.end.sum_sent.bytes": 1015936976,
"Results.end.sum_sent.end": 10.000223,
"Results.end.sum_sent.retransmits": 5,
"Results.end.sum_sent.seconds": 10.000223,
"Results.end.sum_sent.start": 0,
"Results.intervals.0.streams.0.bits_per_second": 817688301.083527,
"Results.intervals.0.streams.0.bytes": 102221760,
"Results.intervals.0.streams.0.end": 1.000105,
"Results.intervals.0.streams.0.omitted": false,
"Results.intervals.0.streams.0.retransmits": 2,
"Results.intervals.0.streams.0.rtt": 3960,
"Results.intervals.0.streams.0.seconds": 1.000105,
"Results.intervals.0.streams.0.snd_cwnd": 426390,
"Results.intervals.0.streams.0.socket": 4,
"Results.intervals.0.streams.0.start": 0,
"Results.intervals.0.sum.bits_per_second": 817688301.083527,
"Results.intervals.0.sum.bytes": 102221760,
"Results.intervals.0.sum.end": 1.000105,
"Results.intervals.0.sum.omitted": false,
"Results.intervals.0.sum.retransmits": 2,
"Results.intervals.0.sum.seconds": 1.000105,
"Results.intervals.0.sum.start": 0,
"Results.intervals.1.streams.0.bits_per_second": 819540175.03198,
"Results.intervals.1.streams.0.bytes": 102442644,
"Results.intervals.1.streams.0.end": 2.000106,
"Results.intervals.1.streams.0.omitted": false,
"Results.intervals.1.streams.0.retransmits": 0,
"Results.intervals.1.streams.0.rtt": 5492,
"Results.intervals.1.streams.0.seconds": 1.000001,
"Results.intervals.1.streams.0.snd_cwnd": 575976,
"Results.intervals.1.streams.0.socket": 4,
"Results.intervals.1.streams.0.start": 1.000105,
"Results.intervals.1.sum.bits_per_second": 819540175.03198,
"Results.intervals.1.sum.bytes": 102442644,
"Results.intervals.1.sum.end": 2.000106,
```



```
"Results.intervals.1.sum.omitted": false,
"Results.intervals.1.sum.retransmits": 0,
"Results.intervals.1.sum.seconds": 1.000001,
"Results.intervals.1.sum.start": 1.000105,
"Results.intervals.2.streams.0.bits_per_second": 816434902.675058,
"Results.intervals.2.streams.0.bytes": 102056796,
"Results.intervals.2.streams.0.end": 3.00013,
"Results.intervals.2.streams.0.omitted": false,
"Results.intervals.2.streams.0.retransmits": 2,
"Results.intervals.2.streams.0.rtt": 6494,
"Results.intervals.2.streams.0.seconds": 1.000024,
"Results.intervals.2.streams.0.snd_cwnd": 687816,
"Results.intervals.2.streams.0.socket": 4,
"Results.intervals.2.streams.0.start": 2.000106,
"Results.intervals.2.sum.bits_per_second": 816434902.675058,
"Results.intervals.2.sum.bytes": 102056796,
"Results.intervals.2.sum.end": 3.00013,
"Results.intervals.2.sum.omitted": false,
"Results.intervals.2.sum.retransmits": 2,
"Results.intervals.2.sum.seconds": 1.000024,
"Results.intervals.2.sum.start": 2.000106,
"Results.intervals.3.streams.0.bits_per_second": 807718879.060123,
"Results.intervals.3.streams.0.bytes": 100963560,
"Results.intervals.3.streams.0.end": 4.000117,
"Results.intervals.3.streams.0.omitted": false,
"Results.intervals.3.streams.0.retransmits": 1,
"Results.intervals.3.streams.0.rtt": 7389,
"Results.intervals.3.streams.0.seconds": 0.999987,
"Results.intervals.3.streams.0.snd_cwnd": 787074,
"Results.intervals.3.streams.0.socket": 4,
"Results.intervals.3.streams.0.start": 3.00013,
"Results.intervals.3.sum.bits_per_second": 807718879.060123,
"Results.intervals.3.sum.bytes": 100963560,
"Results.intervals.3.sum.end": 4.000117,
"Results.intervals.3.sum.omitted": false,
"Results.intervals.3.sum.retransmits": 1,
"Results.intervals.3.sum.seconds": 0.999987,
"Results.intervals.3.sum.start": 3.00013,
"Results.intervals.4.streams.0.bits_per_second": 807663227.948988,
"Results.intervals.4.streams.0.bytes": 100963560,
"Results.intervals.4.streams.0.end": 5.000173,
"Results.intervals.4.streams.0.omitted": false,
"Results.intervals.4.streams.0.retransmits": 0,
"Results.intervals.4.streams.0.rtt": 8568,
"Results.intervals.4.streams.0.seconds": 1.000056,
"Results.intervals.4.streams.0.snd_cwnd": 877944,
"Results.intervals.4.streams.0.socket": 4,
"Results.intervals.4.streams.0.start": 4.000117,
"Results.intervals.4.sum.bits_per_second": 807663227.948988,
"Results.intervals.4.sum.bytes": 100963560,
"Results.intervals.4.sum.end": 5.000173,
"Results.intervals.4.sum.omitted": false,
"Results.intervals.4.sum.retransmits": 0,
"Results.intervals.4.sum.seconds": 1.000056,
"Results.intervals.4.sum.start": 4.000117,
"Results.intervals.5.streams.0.bits_per_second": 808756111.780852,
```

```
"Results.intervals.5.streams.0.bytes": 101092176,
"Results.intervals.5.streams.0.end": 6.00015,
"Results.intervals.5.streams.0.omitted": false,
"Results.intervals.5.streams.0.retransmits": 0,
"Results.intervals.5.streams.0.rtt": 9689,
"Results.intervals.5.streams.0.seconds": 0.999977,
"Results.intervals.5.streams.0.snd_cwnd": 959028,
"Results.intervals.5.streams.0.socket": 4,
"Results.intervals.5.streams.0.start": 5.000173,
"Results.intervals.5.sum.bits_per_second": 808756111.780852,
"Results.intervals.5.sum.bytes": 101092176,
"Results.intervals.5.sum.end": 6.00015,
"Results.intervals.5.sum.omitted": false,
"Results.intervals.5.sum.retransmits": 0,
"Results.intervals.5.sum.seconds": 0.999977,
"Results.intervals.5.sum.start": 5.000173,
"Results.intervals.6.streams.0.bits_per_second": 812324711.522149,
"Results.intervals.6.streams.0.bytes": 101542332,
"Results.intervals.6.streams.0.end": 7.000167,
"Results.intervals.6.streams.0.omitted": false,
"Results.intervals.6.streams.0.retransmits": 0,
"Results.intervals.6.streams.0.rtt": 10018,
"Results.intervals.6.streams.0.seconds": 1.000017,
"Results.intervals.6.streams.0.snd_cwnd": 1034520,
"Results.intervals.6.streams.0.socket": 4,
"Results.intervals.6.streams.0.start": 6.00015,
"Results.intervals.6.sum.bits_per_second": 812324711.522149,
"Results.intervals.6.sum.bytes": 101542332,
"Results.intervals.6.sum.end": 7.000167,
"Results.intervals.6.sum.omitted": false,
"Results.intervals.6.sum.retransmits": 0,
"Results.intervals.6.sum.seconds": 1.000017,
"Results.intervals.6.sum.start": 6.00015,
"Results.intervals.7.streams.0.bits_per_second": 809288339.432058,
"Results.intervals.7.streams.0.bytes": 101156484,
"Results.intervals.7.streams.0.end": 8.000122,
"Results.intervals.7.streams.0.omitted": false,
"Results.intervals.7.streams.0.retransmits": 0,
"Results.intervals.7.streams.0.rtt": 10836,
"Results.intervals.7.streams.0.seconds": 0.999955,
"Results.intervals.7.streams.0.snd_cwnd": 1105818,
"Results.intervals.7.streams.0.socket": 4,
"Results.intervals.7.streams.0.start": 7.000167,
"Results.intervals.7.sum.bits_per_second": 809288339.432058,
"Results.intervals.7.sum.bytes": 101156484,
"Results.intervals.7.sum.end": 8.000122,
"Results.intervals.7.sum.omitted": false,
"Results.intervals.7.sum.retransmits": 0,
"Results.intervals.7.sum.seconds": 0.999955,
"Results.intervals.7.sum.start": 7.000167,
"Results.intervals.8.streams.0.bits_per_second": 811791676.23538,
"Results.intervals.8.streams.0.bytes": 101478024,
"Results.intervals.8.streams.0.end": 9.000162,
"Results.intervals.8.streams.0.omitted": false,
"Results.intervals.8.streams.0.retransmits": 0,
"Results.intervals.8.streams.0.rtt": 11213,
```

```
"Results.intervals.8.streams.0.seconds": 1.00004,
"Results.intervals.8.streams.0.snd_cwnd": 1171524,
"Results.intervals.8.streams.0.socket": 4,
"Results.intervals.8.streams.0.start": 8.000122,
"Results.intervals.8.sum.bits_per_second": 811791676.23538,
"Results.intervals.8.sum.bytes": 101478024,
"Results.intervals.8.sum.end": 9.000162,
"Results.intervals.8.sum.omitted": false,
"Results.intervals.8.sum.retransmits": 0,
"Results.intervals.8.sum.seconds": 1.00004,
"Results.intervals.8.sum.start": 8.000122,
"Results.intervals.9.streams.0.bits_per_second": 816107503.326209,
"Results.intervals.9.streams.0.bytes": 102019640,
"Results.intervals.9.streams.0.end": 10.000223,
"Results.intervals.9.streams.0.omitted": false,
"Results.intervals.9.streams.0.retransmits": 0,
"Results.intervals.9.streams.0.rtt": 11932,
"Results.intervals.9.streams.0.seconds": 1.000061,
"Results.intervals.9.streams.0.snd_cwnd": 1234434,
"Results.intervals.9.streams.0.socket": 4,
"Results.intervals.9.streams.0.start": 9.000162,
"Results.intervals.9.sum.bits_per_second": 816107503.326209,
"Results.intervals.9.sum.bytes": 102019640,
"Results.intervals.9.sum.end": 10.000223,
"Results.intervals.9.sum.omitted": false,
"Results.intervals.9.sum.retransmits": 0,
"Results.intervals.9.sum.seconds": 1.000061,
"Results.intervals.9.sum.start": 9.000162,
"Results.start.connected.0.local_host": "172.18.3.2",
"Results.start.connected.0.local_port": 36717,
"Results.start.connected.0.remote_host": "192.168.100.13",
"Results.start.connected.0.remote_port": 5201,
"Results.start.connected.0.socket": 4,
"Results.start.connecting_to.host": "192.168.100.13",
"Results.start.connecting_to.port": 5201,
"Results.start.cookie": "90aa79fc96fb.1551446043.597668.7500f",
"Results.start.system_info": "Linux 90aa79fc96fb 4.9.0-8-amd64 #1 SMP Debian 4.9.110-3+deb9u6 (2018-10-08) x86_64",
"Results.start.tcp_mss_default": 1398,
"Results.start.test_start.blksize": 131072,
"Results.start.test_start.blocks": 0,
"Results.start.test_start.bytes": 0,
"Results.start.test_start.duration": 10,
"Results.start.test_start.num_streams": 1,
"Results.start.test_start.omit": 0,
"Results.start.test_start.protocol": "TCP",
"Results.start.test_start.reverse": 0,
"Results.start.timestamp.time": "Fri, 01 Mar 2019 13:14:03 GMT",
"Results.start.timestamp.timesecs": 1551446043,
"Results.start.version": "iperf 3.1.3",
"Timestamp": 1551446053.787472
}
```

## Example of output from the browsertime5g container

```
{
  "browsertime-har.log.entries.0.request.headers.1.name": ":method",
  "browsertime-har.log.entries.8.response.headers.5.name": "date",
  "browsertime-har.log.entries.11._initiator_type": "parser",
  "browsertime-har.log.entries.20.request.headers.8.name": "accept-language",
  "browsertime-har.log.entries.1.response.headers.9.name": "x-fb-trip-id",
  "browsertime-har.log.entries.0.timings._queued": 11.357,
  "browsertime-har.log.entries.1.connection": "23",
  "browsertime-har.log.entries.17.timings.blocked": 1.117,
  "browsertime-har.log.entries.14.request.httpVersion": "h2",
  "browsertime-har.log.entries.6.request.url":
"https://www.instagram.com/static/bundles/es6/en_US.js/96f78e625432.js",
  "browsertime-har.log.entries.2.timings.receive": 14.41,
  "browsertime-json.0.browserScripts.0.pageinfo.layoutShift": 0.015138028407647146,
  "browsertime-har.log.entries.22.response.headers.7.name": "etag",
  "browsertime-json.0.android.power": [],
  "browsertime-json.0.visualMetrics.0.VisualProgress.11.timestamp": 2069,
  "browsertime-har.log.entries.17.response.headers.6.name": "edge-control",
  "browsertime-har.log.entries.8.request.cookies": [],
  "browsertime-har.log.entries.18.response.content.size": 199514,
  "browsertime-har.log.entries.10.response.bodySize": 320795,
  "browsertime-json.0.visualMetrics.0.VisualProgress.9.timestamp": 1869,
  "browsertime-har.log.entries.17.time": 70.152,
  "browsertime-har.log.entries.18.response.headers.12.name": "expires",
  "browsertime-har.log.entries.20.timings.send": 0.546,
  "browsertime-har.log.entries.2.response.headers.0.name": "access-control-allow-origin",
  "browsertime-har.log.entries.21.request.headers.8.value": "en-US,en;q=0.9",
  "browsertime-har.log.entries.7.request.headers.1.name": "Referer",
  "browsertime-json.0.browserScripts.0.timings.navigationTiming.startTime": 0,
  "browsertime-har.log.entries.22.timings.receive": 3.504,
  "browsertime-har.log.entries.6.request.headers.10.value": "https://www.instagram.com",
  "browsertime-har.log.entries.2.request.headers.6.value": "https",
  "browsertime-har.log.entries.3.timings._queued": 4.717,
  "browsertime-har.log.entries.16.request.bodySize": 0,
  "browsertime-har.log.entries.24.request.method": "POST",
  "browsertime-har.log.entries.0.connection": "23",
  "browsertime-har.log.entries.15._initialPriority": "Low",
  "browsertime-har.log.entries.19.timings._queued": 1.229,
  "browsertime-har.log.entries.12.response.headers.6.value": "max-age=1209600, no-transform",
  "browsertime-har.log.entries.15.response.headers.5.name": "date",
  "browsertime-json.0.browserScripts.0.pageinfo.longTask.1.startTime": 1147.1050000400282,
  "browsertime-json.0.browserScripts.0.pageinfo.longTask.1.attribution.0.containerType": "window",
  "browsertime-json.0.browserScripts.0.pageinfo.documentWidth": 1365,
  "browsertime-har.log.entries.3.timings.dns": -1,
  "browsertime-har.log.entries.16.response.headers.0.value": "*",
  "browsertime-har.log.entries.10.response.headers.4.name": "content-type",
  "browsertime-har.log.entries.10.timings.blocked": 30.832,
  "browsertime-har.log.entries.1.request.headers.12.value": "style",
  "browsertime-har.log.version": "1.2",
  "browsertime-har.log.entries.6.request.bodySize": 0,
  "browsertime-har.log.entries.12.response.bodySize": 25880,
  "browsertime-har.log.entries.0.response.content.compression": 33711,
  "browsertime-har.log.entries.9.request.headers.1.value": "https://www.instagram.com/",
  "browsertime-har.log.entries.21.timings.connect": -1,
```

```

"browsertime-har.log.entries.13.response.headers.5.name": "date",
"browsertime-har.log.entries.5.request.headers.6.name": ":scheme",
"browsertime-json.0.cdp.performance.0.DevToolsCommandDuration": 10.083,
"browsertime-har.log.entries.4.request.url":
"https://www.instagram.com/static/bundles/es6/LandingPage.css/8f3e856ac244.css",
"browsertime-har.log.entries.9.response.headersSize": -1,
"browsertime-har.log.entries.10.request.headers.6.value": "https",
"browsertime-json.0.browserScripts.0.timings.navigationTiming.redirectEnd": 0,
"browsertime-har.log.entries.11.request.headers.9.name": "accept-language",
"browsertime-har.log.entries.7.request.headers.2.value": "Mozilla/5.0 (Linux; Android 10; SM-G950F
Build/R16NW) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.83 Mobile Safari/537.36",
"browsertime-har.log.entries.6.response.headers.5.value": "Thu, 17 Dec 2020 22:04:46 GMT",
"browsertime-har.log.pages.0.pageTimings.onContentLoaded": 766.646,
"browsertime-har.log.entries.20.response.headers.12.name": "origin-trial",
"browsertime-har.log.entries.5.response.headers.5.name": "date",
"browsertime-har.log.entries.3.pageref": "page_1",
"browsertime-har.log.entries.15._initiator_function_name": "",
"browsertime-har.log.entries.2.request.headers.9.name": "accept-language",
"browsertime-json.0.browserScripts.0.timings.largestContentfulPaint.id": "",
"browsertime-har.log.entries.13.connection": "0",
"browsertime-har.log.entries.22.response.headers.6.name": "edge-control",
"browsertime-har.log.entries.20.request.headers.4.name": ":path",
"browsertime-har.log.entries.5.cache": {},
"browsertime-har.log.entries.22.response.headersSize": -1,
"browsertime-har.log.entries.3.response.headers.0.value": "*",
"browsertime-har.log.entries.1.response.headers.9.value": "1679558926",
"browsertime-har.log.entries.7.cache": {},
"browsertime-har.log.entries.5.response.headersSize": -1,
"browsertime-har.log.entries.12.timings._queued": 21.992,
"browsertime-har.log.entries.14.response.headers.20.name": "x-frame-options",
"browsertime-har.log.entries.2.request.headers.1.name": "Referer",
"browsertime-har.log.entries.18.response.redirectURL": "",
"browsertime-har.log.entries.3.request.url":
"https://www.instagram.com/static/bundles/es6/Consumer.css/cea689ffd2fe.css",
"browsertime-har.log.entries.18.response.headers.16.value": "Accept-Encoding",
"browsertime-har.log.entries.13.response.headers.3.name": "content-length",
"browsertime-har.log.entries.17.response.headers.3.value": "142252",
"browsertime-har.log.entries.0.response.headers.13.name": "report-to",
"browsertime-har.log.entries.13.request.headers.0.value": "https://www.instagram.com",
"browsertime-har.log.entries.24.response.headers.11.value": "Fri, 18 Dec 2020 10:34:17 GMT",
"browsertime-har.log.entries.1.response.headers.5.value": "Wed, 16 Dec 2020 19:58:21 GMT",
"browsertime-har.log.entries.24.response.headers.4.value": "private, no-cache, no-store, must-revalidate",
"browsertime-har.log.entries.1.response.content.size": 125118,
"browsertime-har.log.entries.14.timings.ssl": 11.734,
"browsertime-har.log.entries.20.response.headers.18.value": "nosniff",
"browsertime-har.log.entries.1.request.headers.0.name": "Origin",
"browsertime-har.log.entries.20.response.headers.6.value": "report-uri
https://www.instagram.com/security/csp_report/; default-src 'self' https://www.instagram.com; img-src data:
blob: https://*.fbcdn.net https://*.instagram.com https://*.cdninstagram.com https://*.facebook.com; font-src
data: https://*.fbcdn.net https://*.instagram.com https://*.cdninstagram.com; media-src 'self' blob:
https://www.instagram.com https://*.cdninstagram.com https://*.fbcdn.net; manifest-src 'self'
https://www.instagram.com; script-src 'self' https://instagram.com https://www.instagram.com
https://*.www.instagram.com https://*.cdninstagram.com wss://www.instagram.com https://*.facebook.com
https://*.fbcdn.net https://*.facebook.net 'unsafe-inline' 'unsafe-eval' blob:; style-src 'self'
https://*.www.instagram.com https://www.instagram.com 'unsafe-inline'; connect-src 'self'
https://instagram.com https://www.instagram.com https://*.www.instagram.com https://graph.instagram.com

```

```
https://*.graph.instagram.com https://*.cdninstagram.com https://api.instagram.com https://i.instagram.com
wss://www.instagram.com wss://edge-chat.instagram.com https://*.facebook.com https://*.fbcdn.net
https://*.facebook.net chrome-extension://boadgeojelhgnadghljhdicfkmllpafd blob;; worker-src 'self' blob:
https://www.instagram.com; frame-src 'self' https://instagram.com https://www.instagram.com
https://*.instagram.com https://staticxx.facebook.com https://www.facebook.com https://web.facebook.com
https://connect.facebook.net https://m.facebook.com; object-src 'none'; upgrade-insecure-requests",
"browsertime-json.0.visualMetrics.0.VisualProgress.7.percent": 98,
"browsertime-har.log.entries.12.request.headers.7.name": "accept",
"browsertime-har.log.entries.6.request.headers.0.value": "https://www.instagram.com",
"browsertime-har.log.pages.0.pageTimings._firstContentfulPaint": 363,
"browsertime-har.log.entries.4.request.headers.14.name": "sec-fetch-site",
"browsertime-har.log.entries.8.request.headersSize": -1,
"browsertime-har.log.entries.12.request.headers.0.name": "Origin",
"browsertime-har.log.entries.6.response.headers.6.name": "edge-control",
"browsertime-har.log.entries.14.response.headers.15.name": "x-content-type-options",
"browsertime-har.log.entries.14.response.statusText": "",
"browsertime-har.log.entries.23.response.headers.2.value": "br",
"browsertime-har.log.entries.2.response.headers.4.name": "content-type",
"browsertime-json.0.cpu.0.longTasks.totalDuration": 911.9499999214895,
"browsertime-har.log.entries.10.request.headers.5.name": ":path",
"browsertime-har.log.entries.9.timings.dns": -1,
"browsertime-har.log.entries.19.request.bodySize": 0,
"browsertime-har.log.entries.18.pagerref": "page_1",
"browsertime-har.log.entries.12.request.headers.14.value": "cors",
"browsertime-har.log.entries.24.response.headers.23.name": "x-robots-tag",
"browsertime-har.log.entries.7._initiator": "https://www.instagram.com/",
"browsertime-har.log.entries.15.request.headersSize": -1,
"browsertime-json.0.browserScripts.0.timings.navigationTiming.workerStart": 0,
"browsertime-har.log.entries.5.request.httpVersion": "h3-29",
"browsertime-har.log.entries.18.request.headers.3.value": "connect.facebook.net",
"browsertime-har.log.entries.3.response.content.mimeType": "text/css",
"browsertime-har.log.entries.0.pagerref": "page_1",
"browsertime-har.log.entries.3.response.headers.6.name": "edge-control",
"browsertime-har.log.entries.4._priority": "VeryHigh",
"browsertime-har.log.entries.5.request.headers.14.value": "same-origin",
"browsertime-har.log.entries.24.request.queryString": [],
"browsertime-har.log.entries.5.response.headers.2.value": "br",
"browsertime-har.log.entries.9.request.headers.12.name": "sec-fetch-dest",
"browsertime-har.log.entries.20.timings.receive": 2.985,
"browsertime-har.log.entries.4.request.headers.9.name": "accept-language",
"browsertime-har.log.entries.3._requestTime": 505322.781433,
"browsertime-har.log.entries.8.response.headers.4.value": "text/javascript",
"browsertime-har.log.entries.0.request.headers.0.value": "www.instagram.com",
"browsertime-json.0.browserScripts.0.pageinfo.longTask.1.duration": 431.27999996067956,
"browsertime-har.log.entries.15.connection": "0",
"browsertime-har.log.entries.15.timings.dns": -1,
"browsertime-har.log.entries.6.response.headers.8.value": "Accept-Encoding",
"browsertime-har.log.entries.15.response.status": 200,
"browsertime-json.0.cdp.performance.0.FirstMeaningfulPaint": 1680.7109999936074,
"browsertime-har.log.entries.9.response.redirectURL": "",
"browsertime-har.log.entries.13.request.headers.16.value": "Mozilla/5.0 (Linux; Android 10; SM-G950F
Build/R16NW) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.83 Mobile Safari/537.36",
"browsertime-har.log.entries.4.request.headers.13.name": "sec-fetch-mode",
"browsertime-har.log.entries.2.request.method": "GET",
"browsertime-har.log.entries.13.response.headers.4.name": "content-type",
"browsertime-har.log.entries.11.response.headers.0.value": "**",
```



```

"browsertime-har.log.entries.14.request.queryString": [],
"browsertime-har.log.entries.5.request.headers.9.value": "en-US,en;q=0.9",
"browsertime-har.log.entries.7.response.headers.5.name": "date",
"browsertime-har.log.entries.24.response.headers.10.value": "same-origin-allow-popups;report-to=\"coop\"",
"browsertime-har.log.pages.0.pageTimings._largestContentfulPaint": 1816,
"browsertime-har.log.entries.14.response.headers.3.value": "public,max-age=1200,stale-while-revalidate=3600",
"browsertime-har.log.entries.9.request.queryString": [],
"browsertime-har.log.entries.12.request.headers.15.value": "same-origin",
"browsertime-har.log.pages.0._visualMetrics.VisualProgress.467": 1,
"browsertime-har.log.entries.13.request.headers.13.value": "empty",
"browsertime-har.log.entries.20.response.headers.5.name": "content-length",
"browsertime-har.log.entries.5.response.status": 200,
"browsertime-har.log.entries.20.timings.connect": 11.795,
"browsertime-json.0.browserScripts.0.timings.navigationTiming.loadEventEnd": 1761,
"browsertime-har.log.entries.13.response.content.mimeType": "text/css",
"browsertime-har.log.entries.21.request.headers.12.value": "same-origin",
"browsertime-json.0.browserScripts.0.pageinfo.documentTitle": "Instagram",
"browsertime-json.0.cdp.performance.0.Frames": 1,
"browsertime-har.log.entries.15._requestTime": 505323.625008,
"browsertime-har.log.entries.20.request.method": "GET",
"browsertime-har.log.entries.21.response.headers.0.value": "",
"browsertime-har.log.entries.20.response.headers.4.value": "en",
"browsertime-har.log.entries.3.response.httpVersion": "h3-29",
"browsertime-har.log.entries.13.timings.blocked": 30.198,
"browsertime-har.log.entries.10.request.headers.3.value": "www.instagram.com",
"browsertime-har.log.entries.5.response.headers.3.value": "67702",
"browsertime-har.log.entries.6._priority": "High",
"browsertime-har.log.entries.1.request.headers.4.name": ":method",
"browsertime-har.log.entries.19._initiator_function_name": "h",
"browsertime-har.log.entries.18.request.url":
...
21 url requests removed for brevity
...
"browsertime-har.log.entries.2.request.url":
"https://www.instagram.com/static/bundles/es6/ConsumerAsyncCommons.css/0608bd6190e0.css",
"browsertime-har.log.entries.24.response.headers.7.value": "report-uri
https://www.instagram.com/security/csp_report/; default-src 'self' https://www.instagram.com; img-src data:
blob: https://*.fbcdn.net https://*.instagram.com https://*.cdninstagram.com https://*.facebook.com; font-src
data: https://*.fbcdn.net https://*.instagram.com https://*.cdninstagram.com; media-src 'self' blob:
https://www.instagram.com https://*.cdninstagram.com https://*.fbcdn.net; manifest-src 'self'
https://www.instagram.com; script-src 'self' https://instagram.com https://www.instagram.com
https://*.www.instagram.com https://*.cdninstagram.com wss://www.instagram.com https://*.facebook.com
https://*.fbcdn.net https://*.facebook.net 'unsafe-inline' 'unsafe-eval' blob:; style-src 'self'
https://*.www.instagram.com https://www.instagram.com 'unsafe-inline'; connect-src 'self'
https://instagram.com https://www.instagram.com https://*.www.instagram.com https://graph.instagram.com
https://*.graph.instagram.com https://*.cdninstagram.com https://api.instagram.com https://i.instagram.com
wss://www.instagram.com wss://edge-chat.instagram.com https://*.facebook.com https://*.fbcdn.net
https://*.facebook.net chrome-extension://boadgeojelhgndaghljhdicfkmllpafd blob:; worker-src 'self' blob:
https://www.instagram.com; frame-src 'self' https://instagram.com https://www.instagram.com
https://*.instagram.com https://staticxx.facebook.com https://www.facebook.com https://web.facebook.com
https://connect.facebook.net https://m.facebook.com; object-src 'none'; upgrade-insecure-requests",
"browsertime-har.log.entries.6.request.headers.8.value": "gzip, deflate, br",
"browsertime-har.log.entries.22._requestTime": 505324.415616,
"browsertime-json.0.visualMetrics.0.VisualProgress.5.timestamp": 1735,
"browsertime-json.0.browserScripts.0.pageinfo.longTask.1.name": "self",

```

```
"browsertime-har.log.entries.14.response.headers.4.value": "gzip",
"browsertime-har.log.entries.16.response.statusText": "",
"browsertime-har.log.entries.4.request.headers.8.name": "accept-encoding",
"browsertime-har.log.entries.24.connection": "0",
"browsertime-har.log.entries.15.startedDateTime": "2020-12-18T10:34:15.950Z",
"browsertime-har.log.entries.23.request.headers.9.value": "https://www.instagram.com/",
"browsertime-har.log.entries.21.request.headers.10.value": "image",
"browsertime-har.log.entries.5.request.headers.1.value": "https://www.instagram.com/",
"browsertime-har.log.entries.13._initiator": "https://www.instagram.com/",
"browsertime-har.log.entries.0._initiator_column": 30,
"browsertime-har.log.entries.19.response.headers.11.name": "pragma",
"browsertime-har.log.entries.3.response.headers.5.name": "date",
"browsertime-har.log.entries.19.response.headers.9.value": "Sat, 01 Jan 2000 00:00:00 GMT",
"browsertime-har.log.entries.3.response.cookies": [],
"browsertime-json.0.browserScripts.0.timings.navigationTiming.loadEventStart": 1757,
"browsertime-har.log.entries.16.response.headers.1.value": "public,max-age=31536000,immutable",
"browsertime-har.log.entries.6.response.redirectURL": "",
"browsertime-har.log.entries.19.timings.send": 0.999,
"browsertime-har.log.entries.11.response.content.size": 22977,
"browsertime-har.log.entries.12.request.headers.9.value": "en-US,en;q=0.9",
"browsertime-har.log.entries.0.timings.wait": 173.472,
"browsertime-har.log.entries.3.request.bodySize": 0,
"browsertime-har.log.entries.4._initialPriority": "VeryHigh",
"browsertime-har.log.entries.17.request.method": "GET",
"browsertime-har.log.entries.20.request.headers.2.value": "www.instagram.com",
"browsertime-har.log.entries.2.request.cookies": [],
"browsertime-har.log.entries.18.response.headers.4.name": "content-encoding",
"browsertime-har.log.entries.5.request.headers.13.value": "cors",
"browsertime-har.log.entries.20.response.content.mimeType": "application/json",
"browsertime-har.log.entries.1.request.headers.9.value": "en-US,en;q=0.9",
"browsertime-har.log.entries.10._initiator_line": 28,
"browsertime-har.log.entries.8.request.headers.1.value": "https://www.instagram.com/",
"browsertime-har.log.entries.17.response.headers.1.name": "cache-control",
"browsertime-har.log.entries.5.response.headers.1.name": "cache-control",
"browsertime-har.log.entries.15._initiator_column": 614,
"browsertime-json.0.visualMetrics.0.VisualProgress.3.percent": 96,
"browsertime-har.log.entries.7.request.queryString": [],
"browsertime-har.log.entries.16.response.redirectURL": "",
"browsertime-har.log.entries.4.response.content.size": 5780,
"browsertime-har.log.entries.13.timings.receive": 217.742,
"browsertime-har.log.entries.18.request.headers.5.name": ":path",
"browsertime-har.log.entries.12.response.headers.7.value": "\\ec3a97963ca1\\",
"browsertime-har.log.entries.22.timings.blocked": 1.339,
"browsertime-har.log.entries.10.response.headers.2.value": "br",
"browsertime-har.log.entries.19._initiator":
"https://connect.facebook.net/en_US/sdk.js?hash=57ffa29eb315d486bf6190dd85bb34e5&ua=modern_es6",
"browsertime-har.log.entries.14.timings._queued": 1.143,
"browsertime-har.log.entries.6.request.headers.1.name": "Referer",
"browsertime-har.log.entries.19._requestId": "426.55",
"browsertime-har.log.entries.12.request.headers.6.value": "https",
"browsertime-har.log.entries.18.response.headers.18.name": "x-fb-content-md5"
}
```



## Example of output from the 360dash container

```
{
  "DataId": "5GENESIS.EXP.DASHC",
  "DataVersion": 1,
  "FramesonRepLevel.2038": 19,
  "Guid": "fake.guid",
  "Interface": "eth0",
  "NodeId": "virtual",
  "NRFrames": 19,
  "NRRepLevelSwitches": 0,
  "NRStalls": 0,
  "Results.0.Act_Rate": 2250,
  "Results.0.Arr_time": 925,
  "Results.0.Buff_Leveltimestamp": 2.0,
  "Results.0.Byte_Size": 562644,
  "Results.0.Del_Rate": 4861,
  "Results.0.Del_Time": 925,
  "Results.0.Rep_Level": 2038,
  "Results.0.Seg_#": 1,
  "Results.0.Stall_Dur": 0,
  "Results.1.Act_Rate": 2054,
  "Results.1.Arr_time": 1476,
  "Results.1.Buff_Leveltimestamp": 4.0,
  "Results.1.Byte_Size": 513736,
  "Results.1.Del_Rate": 7464,
  "Results.1.Del_Time": 550,
  "Results.1.Rep_Level": 2038,
  "Results.1.Seg_#": 2,
  "Results.1.Stall_Dur": 0,
  "Results.10.Act_Rate": 2218,
  "Results.10.Arr_time": 5553,
  "Results.10.Buff_Leveltimestamp": 17.925,
  "Results.10.Byte_Size": 554516,
  "Results.10.Del_Rate": 9143,
  "Results.10.Del_Time": 485,
  "Results.10.Rep_Level": 2038,
  "Results.10.Seg_#": 11,
  "Results.10.Stall_Dur": 0,
  "Results.11.Act_Rate": 2122,
  "Results.11.Arr_time": 6021,
  "Results.11.Buff_Leveltimestamp": 19.456,
  "Results.11.Byte_Size": 530516,
  "Results.11.Del_Rate": 9066,
  "Results.11.Del_Time": 468,
  "Results.11.Rep_Level": 2038,
  "Results.11.Seg_#": 12,
  "Results.11.Stall_Dur": 0,
  "Results.12.Act_Rate": 2201,
  "Results.12.Arr_time": 6504,
  "Results.12.Buff_Leveltimestamp": 20.973,
  "Results.12.Byte_Size": 550450,
  "Results.12.Del_Rate": 9112,
  "Results.12.Del_Time": 483,
  "Results.12.Rep_Level": 2038,
  "Results.12.Seg_#": 13,
```

```
"Results.12.Stall_Dur": 0,
"Results.13.Act_Rate": 2088,
"Results.13.Arr_time": 6963,
"Results.13.Buff_Leveltimestamp": 22.514,
"Results.13.Byte_Size": 522246,
"Results.13.Del_Rate": 9107,
"Results.13.Del_Time": 458,
"Results.13.Rep_Level": 2038,
"Results.13.Seg_#": 14,
"Results.13.Stall_Dur": 0,
"Results.14.Act_Rate": 2064,
"Results.14.Arr_time": 7417,
"Results.14.Buff_Leveltimestamp": 24.061,
"Results.14.Byte_Size": 516183,
"Results.14.Del_Rate": 9107,
"Results.14.Del_Time": 453,
"Results.14.Rep_Level": 2038,
"Results.14.Seg_#": 15,
"Results.14.Stall_Dur": 0,
"Results.15.Act_Rate": 2189,
"Results.15.Arr_time": 7896,
"Results.15.Buff_Leveltimestamp": 25.581,
"Results.15.Byte_Size": 547310,
"Results.15.Del_Rate": 9130,
"Results.15.Del_Time": 479,
"Results.15.Rep_Level": 2038,
"Results.15.Seg_#": 16,
"Results.15.Stall_Dur": 0,
"Results.16.Act_Rate": 2475,
"Results.16.Arr_time": 8442,
"Results.16.Buff_Leveltimestamp": 27.036,
"Results.16.Byte_Size": 618768,
"Results.16.Del_Rate": 9078,
"Results.16.Del_Time": 545,
"Results.16.Rep_Level": 2038,
"Results.16.Seg_#": 17,
"Results.16.Stall_Dur": 0,
"Results.17.Act_Rate": 1174,
"Results.17.Arr_time": 8700,
"Results.17.Buff_Leveltimestamp": 28.778,
"Results.17.Byte_Size": 293543,
"Results.17.Del_Rate": 9090,
"Results.17.Del_Time": 258,
"Results.17.Rep_Level": 2038,
"Results.17.Seg_#": 18,
"Results.17.Stall_Dur": 0,
"Results.18.Act_Rate": 2054,
"Results.18.Arr_time": 9148,
"Results.18.Buff_Leveltimestamp": 30.331,
"Results.18.Byte_Size": 513736,
"Results.18.Del_Rate": 9191,
"Results.18.Del_Time": 447,
"Results.18.Rep_Level": 2038,
"Results.18.Seg_#": 19,
"Results.18.Stall_Dur": 0,
"Results.2.Act_Rate": 1990,
```

```
"Results.2.Arr_time": 1910,  
"Results.2.Buff_Leveltimestamp": 5.566,  
"Results.2.Byte_Size": 497737,  
"Results.2.Del_Rate": 9185,  
"Results.2.Del_Time": 433,  
"Results.2.Rep_Level": 2038,  
"Results.2.Seg_#": 3,  
"Results.2.Stall_Dur": 0,  
"Results.3.Act_Rate": 1970,  
"Results.3.Arr_time": 2345,  
"Results.3.Buff_Leveltimestamp": 7.132,  
"Results.3.Byte_Size": 492558,  
"Results.3.Del_Rate": 9063,  
"Results.3.Del_Time": 434,  
"Results.3.Rep_Level": 2038,  
"Results.3.Seg_#": 4,  
"Results.3.Stall_Dur": 0,  
"Results.4.Act_Rate": 1965,  
"Results.4.Arr_time": 2774,  
"Results.4.Buff_Leveltimestamp": 8.703,  
"Results.4.Byte_Size": 491391,  
"Results.4.Del_Rate": 9160,  
"Results.4.Del_Time": 429,  
"Results.4.Rep_Level": 2038,  
"Results.4.Seg_#": 5,  
"Results.4.Stall_Dur": 0,  
"Results.5.Act_Rate": 2144,  
"Results.5.Arr_time": 3244,  
"Results.5.Buff_Leveltimestamp": 10.232,  
"Results.5.Byte_Size": 536147,  
"Results.5.Del_Rate": 9118,  
"Results.5.Del_Time": 470,  
"Results.5.Rep_Level": 2038,  
"Results.5.Seg_#": 6,  
"Results.5.Stall_Dur": 0,  
"Results.6.Act_Rate": 2105,  
"Results.6.Arr_time": 3707,  
"Results.6.Buff_Leveltimestamp": 11.77,  
"Results.6.Byte_Size": 526443,  
"Results.6.Del_Rate": 9113,  
"Results.6.Del_Time": 462,  
"Results.6.Rep_Level": 2038,  
"Results.6.Seg_#": 7,  
"Results.6.Stall_Dur": 0,  
"Results.7.Act_Rate": 1997,  
"Results.7.Arr_time": 4149,  
"Results.7.Buff_Leveltimestamp": 13.328,  
"Results.7.Byte_Size": 499318,  
"Results.7.Del_Rate": 9038,  
"Results.7.Del_Time": 441,  
"Results.7.Rep_Level": 2038,  
"Results.7.Seg_#": 8,  
"Results.7.Stall_Dur": 0,  
"Results.8.Act_Rate": 1995,  
"Results.8.Arr_time": 4584,  
"Results.8.Buff_Leveltimestamp": 14.893,
```

```
"Results.8.Byte_Size": 498868,  
"Results.8.Del_Rate": 9165,  
"Results.8.Del_Time": 435,  
"Results.8.Rep_Level": 2038,  
"Results.8.Seg_#": 9,  
"Results.8.Stall_Dur": 0,  
"Results.9.Act_Rate": 2197,  
"Results.9.Arr_time": 5067,  
"Results.9.Buff_Leveltimestamp": 16.41,  
"Results.9.Byte_Size": 549462,  
"Results.9.Del_Rate": 9100,  
"Results.9.Del_Time": 483,  
"Results.9.Rep_Level": 2038,  
"Results.9.Seg_#": 10,  
"Results.9.Stall_Dur": 0,  
"Timestamp": 1615450344.472974  
}
```