Wireless Crowdsourced Performance Monitoring and Verification

WiFi Performance Measurement Using End-User Mobile Device Feedback

Vasileios Kokkinos, Kostas Stamos, Nikolaos Kanakis
Computer Engineering and Informatics
University of Patras, GRNET
Patras, Greece
kokkinos@cti.gr, stamos@cti.gr, kanakisn@ceid.upatras.gr

Kurt Baumann
Peta Solutions
SWITCH
Zurich, Switzerland
kurt.baumann@switch.ch

Anna Wilson
Network Development
HEAnet
Dublin, Ireland
anna.wilson@heanet.ie

James Healy
Network Management
Dublin City University (DCU)
Dublin, Ireland
james.healy@dcu.ie

***Abstract*—The use of crowdsourced-based network performance measurement services and technologies is set to increase continually among the National Research and Education Networks (NRENs) in the near future. This requires an understanding of the behavior of network performance issues, and their localization and verification on wireless campus networks. The approach presented in this paper is based on the end-user mobile device measurement feedback and allows the visualization of network performance in real time.**

Keywords—WiFi; crowd sensing systems; crowdsourced network performance; performance monitoring; performance verification

I. INTRODUCTION

Eduroam [1] and campus-specific designed wireless networks enable academic users' (students, researchers and staff) broadband access to the Internet via mobile devices and laptops. A consequence of all of this improved access is a growing user population and a nearly exponential increase of data. Such growth results in the need for even more quality monitoring tools, and Quality Assurance (QA), coupled to Service and Operational Level Agreements (SLAs/OLAs) on monitoring, measuring, verifying and visualization of wireless network performance.

Measuring and verifying the quality (performance) of a Wireless Network (WiFi) is particularly challenging, as there is no single tool that covers all aspects of performance monitoring and verification. What we did observe was a particular weakness in existing tools in determining how individual end users (mobile clients) experienced WiFi at a given place on the network, at a given time. With this in mind, we are looking for a network quality statement determined by end-user behavior. Therefore, it is important to verify performance issues, to localize them in real time, and to have a visualization of reference data [2], which shows the network performance, or lack thereof, with (automated) advice to the network provider solving these network issues.

At present, information for performance measurement and verification can be reported in three ways:

- *Mobile End-User Device:* In the past, these devices have been highly neglected in terms of their unique WiFi characteristics. Laptops behave differently to smartphones and have a very different set of RF sensor radios (interfaces) that can provide relevant data for performance behavior on the campus WiFi network
- *Wireless Access Points (WAP)/WiFi-Controller: W*AP management interfaces show specific details on the radio(s) and traffic. They generally do not show the performance of actual applications or the performance of mobile devices without a specific vendor software extension or solution.
- *Network Management Systems (NMS):* These systems have visibility of aggregate and individual client, RF signal quality levels, usage, and spectrum interference levels. This allows basic troubleshooting on clients and does indicate potential problem areas, but doesn't necessarily tell the whole story from the mobile client device perspective.

Wireless Access Points/WiFi Controllers and NMS provide a needed indication of the overall campus wireless network performance (e.g. if the network is up or down). What we are investigating is how to provide network-relevant data (bandwidth, latency, etc.) based on end-users' behavior on the campus wireless network.

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

Our motivation for this work is to research and verify the following thesis:

Is it possible to gather data from multiple sources, including browser-based measurements in addition to traditional monitoring, and extract meaningful information on the performance of a WiFi from that data?

With this thesis, we do not propose to replace traditional hardware-probe based performance measurements. Instead, these are supplemented with "non-invasive" performance measurements from the end users' devices, to provide a hybrid solution that combines static infrastructure performance information (objective measurements) with dynamic performance behavior, thanks to the end-users (mobile clients).

The rest of this paper is structured as follows: Section II shows the architectural concept of our approach, the building blocks and their functionality. Section III presents a Walk-Through of the process that results in the end-users to consuming the reference data. Section IV describes the WiFiMon software, and the collection of data and their correlation using pseudo-code examples with a Web-UI discussion. In Section V we report on implementation forms with main focus on the Dublin City University (DCU) expertise. In Section VI we conclude our work with an outlook on future steps.

II. WiFiMon - Architecture Building Blocks

The key indicator for providing a wireless crowd source performance monitoring and verification schema is the end-user, which means that all mobile devices (clients) can be treated as traffic generators. The architecture can be divided into four blocks: the data source block, the analytics, the aggregation and the visualization block. This approach follows the process steps outlined below:

- *COLLECTING* data on the WiFi interface of eduroam-enabled infrastructures, collectors defined for Syslog, RADIUS Accounting, L2/L3 address binding IPv4/6, DHCP logs and websites, and mobile apps with embedded tests.
- *STORING* temporary, persistent data in (non-) SQL DBs formats.
- *ANALYSING* data, where raw data will be analyzed in real time, the end-user-row data must be set up to reference data regarding security and privacy requirements, and
- *CONSUMING* data, which includes the aggregation of raw-data (source data) from the analyze engine for visualizing in real-time reference data. Open APIs allows customizing queries.

These steps reflect the architecture building blocks/topology, as depicted in Fig. 1.

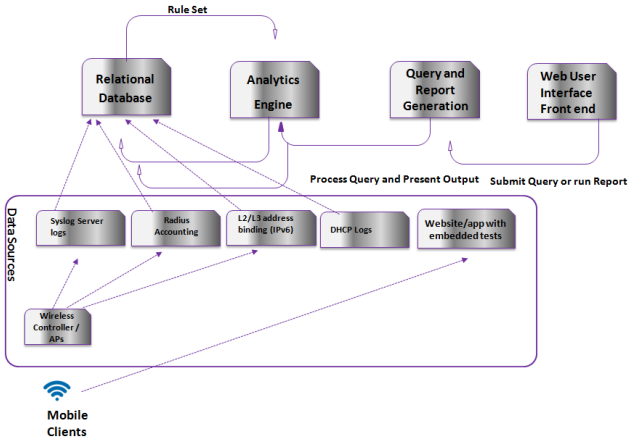

Fig. 1. Architecture Building Blocks

A. Data Source

The data source layer (1) generates information through websites with embedded test procedures, namely JavaScript code embedded in the web source that enables users to run process tests without intervention, and (2) is exporting raw data from data source collectors.

The Open Source tools, Boomerang [3] and NetTest [4] are embedded into websites in order to extract network-related information within a Web browser. Such information includes performance data, such as throughput on downloads and uploads of images with various sizes, and round-trip time (RTT) through ping, as experienced by the end-user.

B. Relational Database

The raw data, such as Syslog Server logs, DHCP logs, and RADIUS Accounting [5], are automatically collected in the Relational Database (RDB). This means that the raw data generated from the data source block are automatically imported in the RDB. The RDB will use SQL as the language for querying and maintenance. The database will be based on Open Source technologies, such as PostgreSQL for the relational database, and InfluxDB, for a time-series database used for visualization.

C. Analytic Engine

The Analytic Engine (AE) is the architecture block used to examine/analyze the RDB's raw data and preparation of reference data, namely, the data objects relevant to transactions comprising value sets, statuses or classification schema, such as raw data, in visualization transactions. Thus, the main functional of the AE is to sort the raw data collected, analyze it, and provide visualizations using tools that offer the greatest insight on the wireless network performance. In the current approach Grafana [6] is used to create the measurement and monitoring dashboards.

D. Query and Report Generation

The Query and Report Generator (QaRG) is used to obtain specific information from the RDB and the AE. Its main purpose is (1) to search for usable information from these two

architecture blocks and (2) to post this information in the form of reports or visualization options to the Web-user Interface (Web-UI), the front end.

E. Web User Interface Front End

The available data from the RDB and the AE is accessible through a network admin Web-UI, which allows data querying. Web-UI allows investigation of collected performance reference data, and in turn, status checks of the wireless network. This block of the architecture shows collected data and allows real-time visualization options, such as collected data from a specific time period, collected data from a specific WAP, min-max-mean values of download/upload/latency measurements, non-normal measurements and top-five performing locations.

III. WiFiMon – Walk-Trough

The end user connects a mobile device to the eduroam SSID and an association is made with the wireless access point. Authentication is completed thorough IEEE 802.1x EAP TTLS [7] and entries are created in the RADIUS accounting log and the DHCP log, along with coordinated timestamps, for the successfully authenticated mobile device which now has an IP address and can communicate (Fig. 2). This data is extracted and populated into the database (this process is achieved in a number of ways, but initially was a manual process). The end user is then required to visit a web page, and in the context, this webpage may be a frequently visited page such as a university e-learning portal, a conference website, or a webmail application. The website has the JavaScript installed (see IV) and this is executed within the browser of the mobile device.

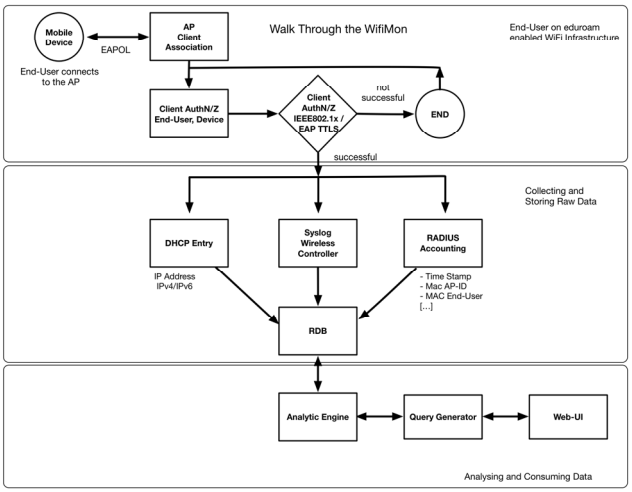

Fig. 2. WiFiMon overview

This initiates a series of download and upload file requests and performance metrics, which are stored on the NetTest server. This data is then extracted and populates the RDB where it is parsed along with the data from the RADIUS and DHCP logs (and perhaps syslog), to correlate the access point identifier (location) with the mobile device and its performance on the wireless network.

IV. WiFiMon Software

This section describes the software that is used in WiFiMon, and how it takes browser-based measurements (through JavaScript) and correlates these measurements with the information extracted from the RADIUS servers. Additional information regarding the Web-UI is also provided.

A. Measurements and Correlation Software

The WiFiMon concept provides wireless, crowdsourced performance monitoring and verification on highly frequented web sources. In the current version of WiFiMon, the download/upload throughput and RTT are calculated using NetTest. NetTest is an open source browser-based network measurement library, licensed under the MIT License, capable of determining throughput, latency, and other network parameters, using JavaScript and/or Flash.

To enable measurements, some sample images (with a predefined size in bytes) have been hosted in an Apache2 server. In order to estimate the download throughput, NetTest calculates the duration of the download process and estimates the download throughput between the client and the web server hosting the sample image based on the sample image size. A similar process is used in order to estimate the upload throughput and RTT.

The above network-related information is then stored in the RDB, together with some user-related information (user IP, location, user agent) and the InfluxDB [8] for time-series visualization. The pseudo code for the above procedure can be found below:

Pseudo code for performing/storing measurements

```
1:  SET registered subnets   //allow measurements only from WiFi subnet
2:  CHECK if cookie is set for the user //avoid repeated measurements and
3:                                      //network overloading
4:  IF user_IP inside registered_subnets
5:      IF cookie is not set
6:          GET timestamp
7:          CALCULATE download_throughput, upload_throughput, RTT
8:          GET user_IP, user_agent
9:          GET user_location          // with Google API loader
10:         POST timestamp, download_throughput, upload_throughput,
11:             RTT, user_IP, user_agent, user_location to Postgres and
12:             InfuxDB databases
13:     SET cookie
14:     ENDIF
15: ENDIF
```

Storing the user-related information and the measurements, however, is only half of the process. In university campuses (where the WiFi network consists of a number of WAPs), this information has to be correlated with the WAP that the measurement was taken from, so that the administrator can have a clear view of the network performance. Fortunately, the missing information can be derived from the RADIUS accounting, the authN/Z log files. In detail, RADIUS servers

store information such as timestamp of authorization, MAC/IP of client device, MAC/IP of WAP, Connection info (e.g. IEEE 802.11b), Username associated with device, etc.

To correlate the two parts, the client IP and the timestamp of authorization of the RADIUS logs can be compared with the user IP and the measurement timestamp as described in the pseudo code below:

Pseudo code for correlating measurements with Radius logs

```
1:  CHECK user_IP, timestamp              //from measurements
2:  CHECK client_IP, auth_timestamp       // from Radius logs
3:  WHILE auth_timestamp < timestamp      // in descending order to
4:                                        // select the most recent entry
5:      IF user_IP == client_IP
6:          INNER JOIN measurement and Radius_entry ON IP
7:          BREAK
8:      ENDIF
9:  ENDWHILE
```

B. *WiFiMon Web-UI*

The WiFimon Web-UI consists of a Spring Boot [9] web application that is using Hibernate [10] for data query. It includes the functionalities that are necessary to enable the measurements (i.e. register subnets), project the correlated data (Fig. 3) and allow real-time visualization options, such as:

- Measurements for a specific time period (Fig. 4).
- Measurement results from a specific WAP.
- Measurement results from specific IPs or users.
- Measurement results for selected wireless technologies (e.g. IEEE 802.11b, etc.)
- Measurement results for user-related parameters (i.e. per operating system, browser used), see Fig. 5.
- Min-max-mean values of download/upload/latency measurements.

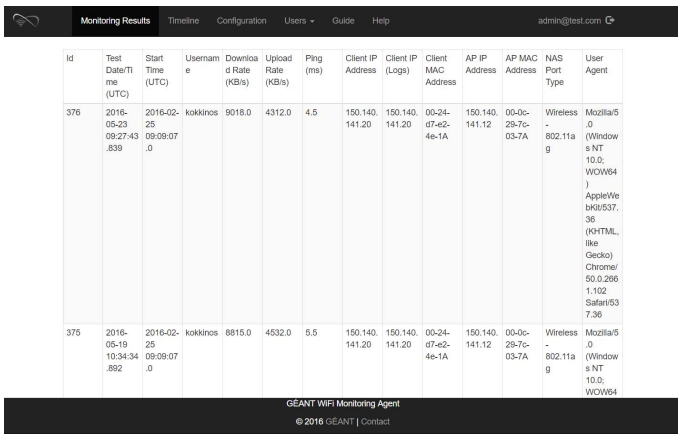

Fig. 3. Measurements' table after correlation with RADIUS logs

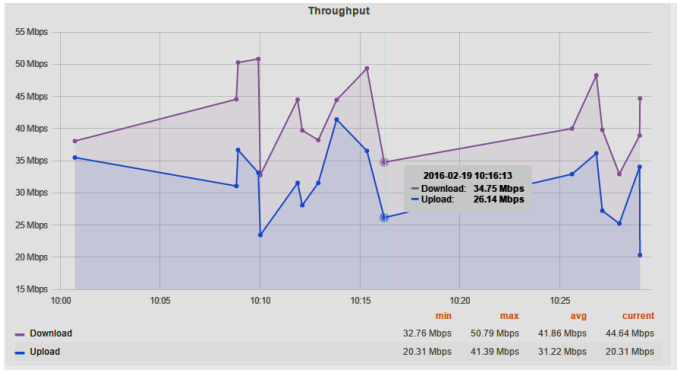

Fig. 4. Download and upload throughput

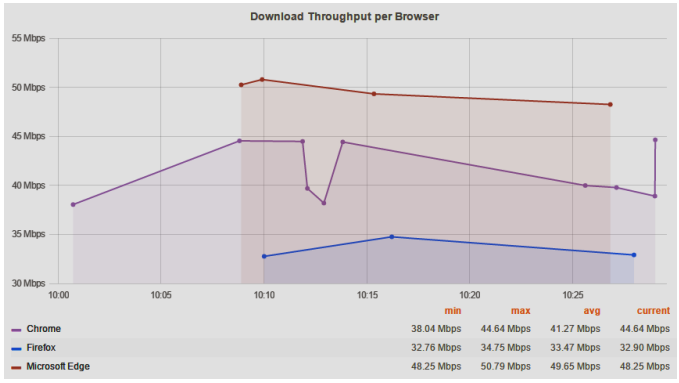

Fig. 5. Download Throughput per Browser

Finally, the WiFiMon UI includes the Grafana tool, which loads the measurement data via InfluxDB and allows the user to display and organize them in multiple ways.

V. WiFiMon Expertise

WiFiMon expertise allows various implementations at various places, based on feasibility studies e.g. at the Dublin City University (DCU) to wireless performance monitoring and their verification demonstrated at the TERENA Networking Conference 2015 (TNC2015).

The implementation discussed here is the feasibility study at the campus of Dublin City University. This is our initial testbed to ascertain the feasibility of the research under consideration.

A. *Dublin City University (DCU) campus*

The WiFi infrastructure at DCU comprises a pair of Motorola (now Zebra Technologies) RFS7000 wireless controllers, with a range of Motorola-dependent and lightweight access points. DCU uses FreeRADIUS [11] running on a Linux platform, with an ISC DHCP server on Linux. There are over 800 wireless access points, across multiple campuses. The solution is implemented to use non-tunneled bridge mode, where client devices are placed on a local switch VLAN after authentication based on a RADIUS attribute, which distinguishes categories of users. This allows authenticated staff to be dropped into staff VLANs, students into student VLANs and external (visiting) eduroam users into

external user VLANs. WiFi authentication for all access on the Dublin City University campus is through the eduroam configured service, for local users and for eduroam visitors. RADIUS logs for end user mobile device associations contain an WAP identifier, which can be matched to a room/corridor description on the controller. This allows location based performance monitoring and verification.

The demonstration at DCU allowed the performance of pilot tests to determine whether it was possible to obtain useful metrics such as download and upload rates and RTTs , on the wireless network, using JavaScript executing on the end user mobile device. The hypothesis was that these measurements could be correlated with the information contained in the RADIUS and DHCP logs equating to WAPs and location. At the same time, there was the challenge of distributed locations, and how to enroll the measurement schema over multiple locations. The parsing of the data allowed for the clustering of WAPs in the different locations to give an aggregate picture of performance at that specific locations with multiple WAPs. For example it is possible to obtain an aggregate performance measure for a large auditorium with multiple WAPs and to visualize the overall performance at the auditorium historically over time.

B. Procedure

While roaming, a number of clients executed the NetTest on a number of occasions over a period of time from different locations across the campus. When NetTest was executed a query was triggered in order to automatically populate the data (authN timestamp, download-upload rate of defined images on NetTest, RTTs, latitude, longitude, client IP) from the individual measurements to the PostgreSQL database (RDB of the architecture, see Fig 1).

C. Results

A total of 154 performance tests were recorded where it was possible to associate with access points across the DCU campus. These measurements were only triggered by GN4-1 SA3 Task 3 members (three devices in total) after they visited a test page that had the NetTest JavaScripts embedded. End-Users roaming in the DCU campus executed the test measurements on a number of occasions, over a period of time, from different locations across the campus.

Fig. 6 and Fig. 7 present a sampling of the results after the correlation of the download and upload rate with the Client MAC. Generally, the download and upload rates show great variation, ranging from 16 KB/s to 9300 KB/s in the case of download and 16 KB/s to 2070 KB/s for the upload. At last, these variations may be due to the wireless technology (e.g. 300Mbps IEEE 802.11an, 650Mbps IEEE 802.11ac, 130Mbps IEEE 802.11bgn) and the user's distance from the WAP during the measurement.

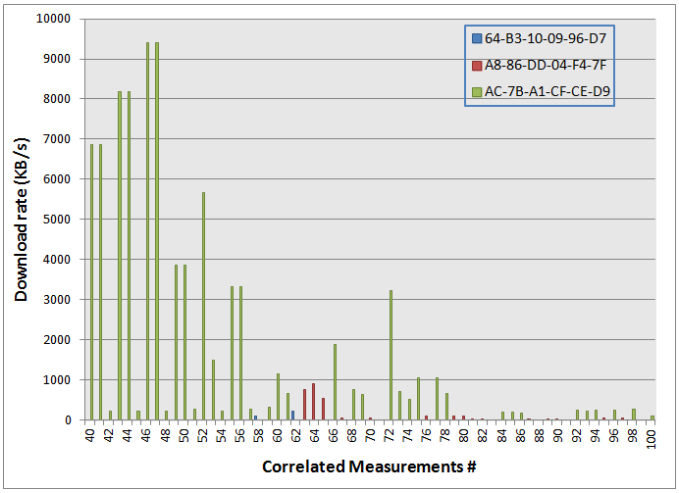

Fig. 6. Correlation of Download rate with Client MAC

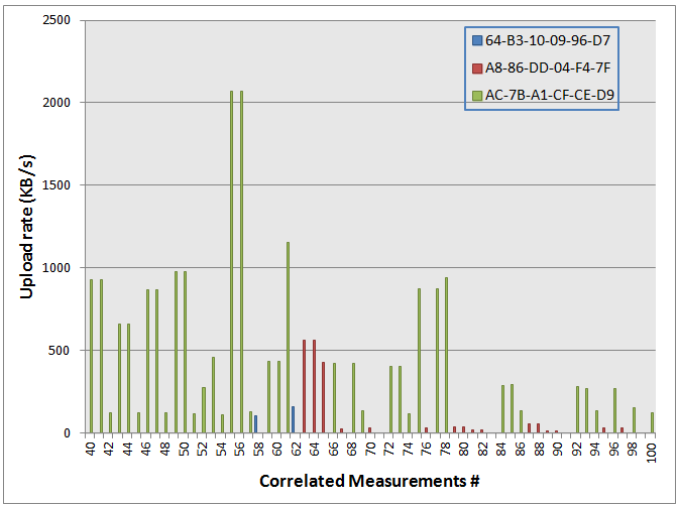

Fig. 7. Correlation of Upload rate with Client MAC

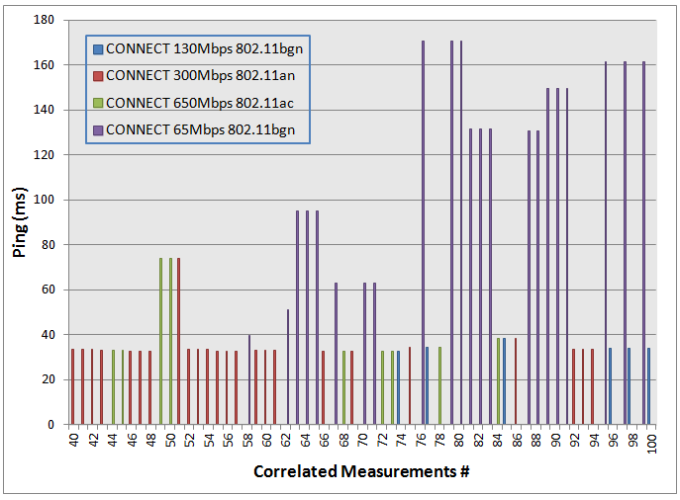

Fig. 8. Correlation of RTT with Connect Info

To better present the effect of the wireless technology, in Fig. 8 we have included the results after the correlation of the RTT with "Connect Info", i.e. the information provided by the RADIUS logs regarding the wireless technology used. Fig. 8 shows that the RTT ranges between 31.5 ms and 170.5 ms, where the highest values are observed for measurements where both the download and the upload rates are relatively low. In addition, this figure reveals that the technology of the wireless network has a direct impact on the RTT performance. Indeed, the majority of the high RTT values were observed when the user was connected to a low speed WAP, i.e. "65Mbit/s IEEE 802.11bgn".

VI. Conclusion and Future Work

The expertise gained so far shows that it is possible to measure specific parameters of a wireless network through JavaScript, and to correlate these measured raw data from various log files. Several steps should be made in order to translate the knowledge base into a complete, sustainable and automated service for monitoring and validating the performance of WiFi on campus. In order to achieve this, the recommended steps should address the following topics:

A. Measurements and Verification

It is necessary to verify whether the measurements obtained through JavaScripts embedded in frequently visited pages are accurate enough. This could be accomplished by installing HW monitoring probes in different rooms of a campus/conference and at different distances from the WAPs (location based, objective measurements). In this hybrid approach, the hard-ware monitoring probes could measure the quality of the wireless network in parallel with the measurements that take place automatically through JavaScript and shows the dynamic behavior of the wireless network performance on the campus.

B. Mobile app deployment

On the DCU campus and at various conferences we observed end-users satisfy their communication needs through the use of smart devices/phones. Most of this communication was not carried out using browser-based functions, but rather using conference applications. While JavaScript in the browser is an easy way to reach users, the amount of time users spend in mobile apps is growing [12].

At last, however analysis such as this requires the largest number of data points possible, so while a standalone performance testing app may be a useful proof of concept, a next step would be to provide an embeddable library so that performance tests can be run (with the user's consent, but without interrupting their work) from an existing app, such as a University's own app.

C. Privacy

While the performance data itself, and the mapping to access points, might be reasonably considered non-sensitive information, we deal with data (such as RADIUS logs), which certainly does contain sensitive information about users. This must be handled with the utmost care, and in accordance with campus policies and legal obligations. As this method is deployed, it is necessary to work not just on the technical ways to handle the data in a variety of heterogeneous networks, but also on how to handle appropriate privacy requirements in every campus where it is deployed in (e.g. collecting, analyze data, access log files etc.).

References

[1] Eduroam, eduoram.org, "How to deploy eduroam on-site or on campus". Available from: https://wiki.geant.org/display/H2eduroam/How+to+deploy+eduroam+on-site+or+on+campus, [09 June 2016]

[2] WhatIS TechTarget, "Definiton of Reference Data". Available from: http://whatis.techtarget.com/definition/reference-data. [09. June 2016]

[3] Boomerang, "SOASTA/boomerang". Available from: https://github.com/SOASTA/boomerang. [09 June 2016]

[4] NetTest, "Google Code Archive". Available from: https://code.google.com/archive/p/nettest/. [09 June 2016]

[5] C. Ringney, S. Willens Livingston, A. Rubens Merit, W. Simpson Daydreamer, RADIUS, "Radius Authentication Dial In User Service (RADIUS)", IETF, RFC2138, June 2000. Available from https://tools.ietf.org/html/rfc2865. [09 June 2016]

[6] Grafana, grafana.org, "Beautiful metric & analytic dashboards". Available form: http://grafana.org/. [09 June 2016]

[7] P. Funk, Funk Software Inc., Simon Blake-Wilson, Basic Commerce & Industries Inc., EAP-TTLS, "EAP Tunneled TLS Authentication Protocol", IETF, Internet Draft, July 2004. Available from: https://tools.ietf.org/html/draft-ietf-pppext-eap-ttls-05. [09 June 2016]

[8] Influxdata, "influxDB". Available from: https://influxdata.com/time-series-platform/influxdb/. [09 June 2016]

[9] Spring, "Building an Application with Spring Boot". Available from: https://spring.io/guides/gs/spring-boot/. [09 June 2016]

[10] HIBERNATE, "Hibernate. Everything data.". Available from: http://hibernate.org/. [09 June 2016]

[11] D. Nelson, Elbrys Networks Inc, A. DeKok, FreeRADIUS, "Common Remote Authentication Dial In User Service (RADIUS) Implementation Issues and Suggested Fixes", December 2007. Available from: https://tools.ietf.org/html/rfc5080. [09 June 2016]

[12] Nielsen, "SO MANY APPS, SO MUCH MORE TIME FOR ENTERTAINMENT". Available form: http://www.nielsen.com/us/en/insights/news/2015/so-many-apps-so-much-more-time-for-entertainment.html. [09 June 2016]