PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 06/19

IoT, Authentication, Performance and Data Science – Experiments using ContextNet

Antonio Iyda Paganelli

Hélio Lopes

Markus Endler

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 06/19 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena October, 2019

IoT, Authentication, Performance and Data Science – Experiments using ContextNet

Antonio Iyda Paganelli[1], Hélio Lopes, Markus Endler[2]

apaganelli@inf.puc-rio.br, lopes@inf.puc-rio.br, endler@inf.puc-rio.br

Abstract. This work presents a series of performance experiments with ContextNet, a middleware aimed at applications for Internet of Mobile Things. It was compared several scenarios with and without authentication of messages. Additionally, it was compared the performance of application answer time using round-trip times with a centralized and a decentralized mechanism for authentication. Challenges and issues for measuring performance on this environment are presented as well as techniques for comparing data series with high variability. It was highlighted the importance of going one level deeper in order to identify conflicts in results. Moreover, the need of automating the test process and the analysis of data for better scrutiny, identification of errors, and reproducibility. Furthermore, in order to clarify results, in additional to data descriptive variables it was presented graphs with data distribution, data density distribution, and different coefficients of uncertainty. Finally, our experiments reinforced the need of designing robust methodologies and presented suggestions for implementing performance tests in this environment.

ContextNet, IoT, Authentication, Performance, data analysis, uncertainty

Resumo. Este trabalho apresenta uma série de experimentos de desempenho na plataforma ContextNet que é um middleware destinado para aplicações da Internet das Coisas Móveis. Foram comparados diversos cenários com e sem autenticação de mensagens. Adicionalmente, também foram comparados os desempenhos de um modelo centralizado e um descentralizado de autenticação. Desafios e questões para mensurar o desempenho neste ambiente são apresentados assim como técnicas para a comparação de séries de dados com alta variabilidade. Foi destacada a importância de investigar uma camada a mais para identificação de conflitos nos resultados. Além disto, para melhor escrutínio, identificação de erros e reprodutibilidade, é de suma importância a automação do processo de testes e de análise dos dados. Além da apresentação de dados descritivos foram gerados gráficos que representam a distribuição das séries, das suas densidades e diferentes coeficientes de incerteza para esclarecer os resultados. Finalmente, os experimentos reforçam a necessidade de desenhos de metodologias robustas apresentando sugestões para a implementação de testes de desempenho neste ambiente.

Palavras-chave: ContextNet, IoT, autenticação, desempenho, análise de dados, incerteza

---

* This work has been sponsored by the Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil.

In charge of publications:

PUC-Rio Departamento de Informática - Publicações
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil

Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: publicacoes@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Table of Contents

1 Introduction 1

2 Test goal 1

2.1 Methodology 2

2.2 Test environment 4

3 Implemented authentication mechanisms in ContextNet 5

3.1 Entities Authentication 5

3.2 Centralized messages Authentication mechanism 6

3.3 Decentralized Authentication mechanism 7

4. Data manipulation, analysis and visualization 8

4 Conclusion 17

References 18

1 Introduction

Techniques related to data science has been applied in many different fields. Despite data science may involve a multitude of disciplines, it fundamentally is based on statistics and computer science fields. Following Zumel and Mount (2014) data science could be defined "as managing the process that can transform hypotheses and data into actionable predictions". Based on this approach, this work explored the process usually applied into data science projects to answer some questions about using authentication of messages in an IoT middleware regarding its impact on application answer times.

One of the major challenges in IoT has been to provide security to its infrastructure [Zfar et al., 2018], mainly to communication among participants of a given application. There are many physical and logical vulnerabilities from the sensors/actuators devices up to the backend processing nodes. Sensors and actuators are generally connected through wireless networks to hubs or base stations which have more computational and communication power to integrate them to core applications, mostly hosted on the cloud. Some attacks to those infrastructures have been reported lately. Additionally, given the large number of connected devices and the huge amount of exchanged messages, adding secure mechanisms may impose reasonable overload in utilized resources [Agarwal and Wang, 2005], and it may impact negatively the applications, for example, impairing response time.

The most basic security features are identification and authentication that allow the implementation of access control policies, apply confidentiality rules, and generate signatures for assuring authenticity, and integrity of messages. Then, we analysed the introduction of two authentication mechanism strategies into an IoT platform aimed at mobile things and its impact on application round-trip times.

A methodological data science approach was applied in order to identify relevant questions, to design models and perform tests aimed at answering those questions, analyse results and automate reports. Prior approaches of manually configuring test, handling log files and importing them to spreadsheets in order to clean-up data, to generate charts and to run statistical tests, copying-and-pasting them to reports are not more acceptable. Control of tests, log file manipulation, reports and charts generation were automated achieving highly reproducible test procedures. This approaches not only allow scrutiny of applied methods but also improved our findings gradually.

2 Test goal

Defining the main goal of our project was the first hard question to be answer. Many different questions seemed to be relevant at the beginning of the project, such as were we testing the capacity of this environment? The viability of a centralized authentication mechanism? Where was the bottleneck of the system? The stability of the middleware under heavy load? The benefits of load balancing of connections or the influences of security on the middleware?

Although all those questions sounded relevant, they did not sufficiently defined our problem. It was necessary like any other research process to spend some reasonable

time defining it. We had executed few anterior unsuccessful performance tests in our environment because the objectives were not clear. Basically, it was not well-defined if we either would like to execute a stress test or a load test.

In this specific case, we had implemented a first naïve authentication mechanism that was integrated to a RADIUS Server. RADIUS is a well-established authentication, authorization and audit solution and supported a huge number of connections [RFC2865]. However, our solution centralized the authentication process on the RADIUS Server. We had failed to test the capacity of this solution when we mixed a stress test of the environment with the authentication mechanism. Firstly, we should have tested the capacity of the environment without any authentication mechanism as a baseline or benchmark and thereafter define other tests.

However, we still would like to know if this solution was useful and what its capacity would be. Yet, in the meantime, predicting that this centralized mechanism would not support larger setups, we developed a second mechanism distributing session keys to the participants and authenticating messages locally. Every session key was associated to an expiration time. Additionally, both authentication mechanisms could be loosen by two policies, by time or randomly, by probability. Time policy assures that a message must be authenticated in at most a time t, and using the probability policy a message would be verified by chance following the given probability p [0 ..1].

Our broaden test goal was to understand the impact of the authentication mechanisms on ContextNet. How would they impact the performance of applications? More specifically, what would be the delays imposed by the authentication of messages using each mechanism on round trip times?

2.1 Methodology

After defining the main question, we had to define sub-questions that support our main question, such as:

- How can I measure the impact regarding performance and capacity?
- What is the capacity of the system with and without authentication?
- Will the impact be similar with different loads on ContextNet?
- What would be the thresholds to characterize different loads?
- What does cause the major impact on performance? The number of connected mobile nodes or the frequency of messages?
- Is there a difference running on Windows Server or on Linux Server?
- How do the different authentication methods affect the system?
- Which one is the most /least efficient mechanism?

It is very difficult to perform tests in a real environment. Then, it was necessary to define a simulated test infrastructure in order to execute the tests. This infrastructure is presented in section 2.2.

Additionally, it was necessary to collect information causing the lowest impact on the system. Then, we decided to register times only on the mobile nodes host simulator avoiding recording files in the gateway and in the processing nodes. The variable that would better represent the impacts on the application level was the round-trip time of

an application messages measured at application level. On one hand, it has the advantage of considering what an application would experience. On the other hand, it has the disadvantage of being influenced by variability of many factors that are out of the scope of this paper regarding operating system scheduling, system call delays, and asynchronicity of events in sending and receiving messages in relation to when the time is clocked by the application.

Knowing that measuring performance is not a straightforward process because it requires to question whatever result is obtained. It generally needs several cycles of finding and correcting errors in applications or in the test environment before getting a stable platform for trust on outcomes. It was fundamental to build up a series of scripts that could perform the rounds of tests automatically with little manual intervention.

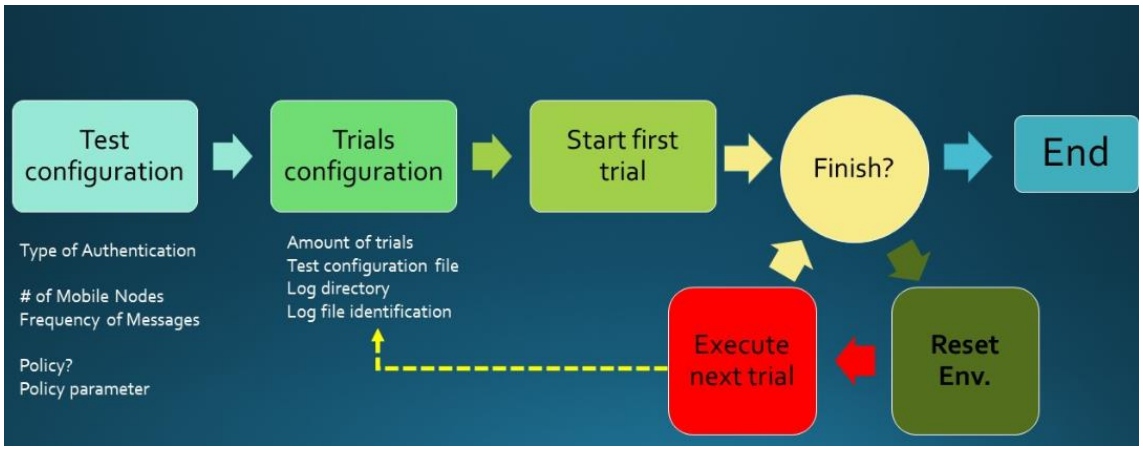

Figure 1: Automation of rounds of tests schema. It was implemented with scripts and tools of the operating system. The unique manual intervention was to start the first trial.

In order to provide reproducibility of test results it was partially automated the process of test configuration, test runs, and to perform results analysis. A set of scripts were built and put available in a local web site integrating these functionalities.

The process can be summarized as:

1. Configure the parameters for running the tests (Automated)
2. Copy the configuration parameters to the test bed environment (manually)
3. Start test runs following given configuration (manually)
4. Perform a set of configured tests (Automated)
5. Prepare data logs (Automated)
6. Transfer data logs from test environment to analysis environment (manually)
7. Data cleansing / Data manipulation (Automated)
8. Data analysis / Visualization / Reports (Automated)

Scripts were developed in bash shell for performing sets of configured tests automatically (step 4).

The remaining of scripts were developed in Python 3.6 and the web interface in Django1. The mean features of collecting, preparing, analysing, and presenting the data

1 Django is an open source framework for web applications developed in Python. http://djangoproject.com

were automated. Only, the transfer of data and start of the test runs are manually started through the operational system command line.

After defining the test process and automating it, it was necessary to establish a benchmark for our environment. Our baseline would be the capacity of our environment without any authentication mechanism enabled. The ideal approach was to perform a stress test and measure all the components of the system and find the one that reaches 100% of utilization which would avoid the improvement of system performance. Monitor all the components of a distributed middleware is a high time-demanding and difficult task. However, our goal was not in finding the limiting factors of the performance of our system, but showing the impacts of adding authentication mechanisms on the existing system.

Finding the capacity of our test environment was important to perform our tests in a lower level of this limit. Moreover, it would be parameter to define different load intensities. The stress test is explained in details at the beginning of the data analysis section.

Once defined the maximum capacity, three different scenarios with light, moderate, and heavy loads were inferred. Then, tests without authentication, with centralized and decentralized authentication of messages were performed.

2.2 Test environment

Our test environment was composed of five distinct virtual machines as follows: (1) mobile node simulator, (1) gateway, (1) processing node, (1) authentication server and (1) load balancing server. Linux Ubuntu 18.04 LTS was installed in all virtual machines. Additionally, there were two other virtual machines running Windows Server, one running as a gateway and the other one as a processing node. However, those Windows VMs were utilized only during tests for comparing differences between both operating systems as a platform for ContextNet middleware.

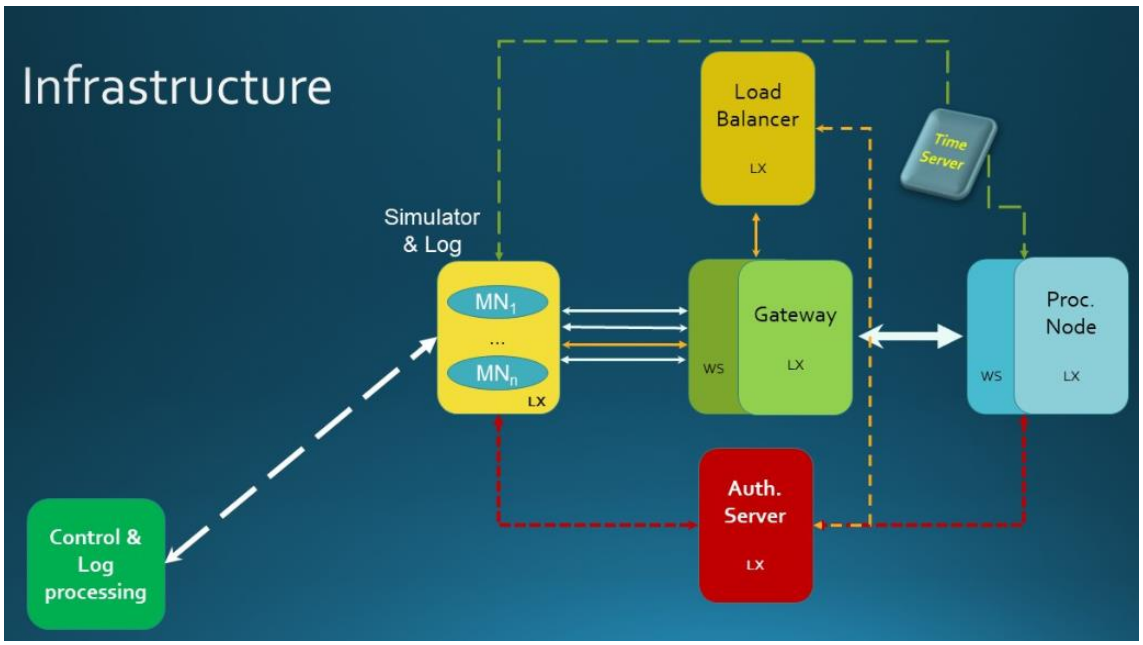

Figure 2: General description of test environment. LX (Linux) WS (Windows Server). MN – Mobile Node. Proc. Node – Processing Node.

Since we were interested in measuring the impact of authentication mechanisms in our environment, in order to reduce the number of uncontrolled variables, we decided to not introduce in our simulations variability and real latencies present in actual wireless networks as well as in traffic through cloud networks. Although, we reduce validity and generality of our findings now, we gain in clarity of effects caused by each mechanism.

3 Implemented authentication mechanisms in ContextNet

Software infrastructure for IoT applications requires specialized middleware that must be able among other features to distribute information efficiently, manage intermittent connections and balance the use of resources. Moreover, it should provide features for device management, interoperation, platform portability, context awareness, security and privacy (Bandyopadhyay et al., 2011).

The attack to DNS servers in 2016, when more than one hundred thousand devices were used for generating hostile traffic (Wei, 2016), emphasized that providing security is one of the main challenges for IoT platforms (Chaqfeh and Mohamed, 2012). Middleware may increase or reduce security and privacy of applications depending on its design characteristics (Fremantle and Scott, 2017).

A common approach to design a secure application is to analyse the risks and threats that those applications will be exposed, and then, devising countermeasures that will avoid or mitigate them. Our group proposed a security solution that classifies threats of IoT infrastructures in two levels: threats to operations of the entities of the IoMT system, and threats to the communication links (Endler, Silva, Cruz, 2017). Then, it was specified a secure architecture with focus on edge networks and smart objects given special attention to data integrity, authentication and service/device access control. It is not objective of our paper to discuss security and to evaluate benefits of our solution regarding the involved risks.

Security and privacy features affect performance of the entities and communication links because they utilize system resources. This utilization of resources may impact end-to-end delays, system throughput, packet loss, CPU loads, and memory usage and other performance indicators. Our focus will be on application messages round-trip times, end-to-end delays.

3.1 Entities Authentication

Our experiment covered part of the proposed secure architecture for ContextNet in (Endler, Silva, Cruz, 2017) and analysed the impact of including authentication procedures using a RADIUS-like approach for authenticating mobile nodes, core entities and messages sent by those elements. Currently, given the heterogeneity and the ephemeral nature of the set of smart objects and the WPAN technologies we did not extend the authentication solution to those end devices.

RADIUS protocol was chosen because it is a robust authentication, authorization, and accounting mechanism already tested in real large-scale systems (Hassel, 2002). Besides that, RADIUS allows flexible and incremental architectures being compatible to

the full implementation of ContextNet secure architecture proposed in (Endler, Silva, Cruz, 2017).

The basic idea of our authentication solutions is that every entity has a credential which is validated in a standard way using one of the available authentication protocols provided by the RADIUS server. After that, each entity receives an exclusive session-key in order to authenticate messages sent by them using Hash-based Message Authentication Code (HMAC) [RFC2104]. RADIUS server and some RADIUS packets were customized to generate and manage session-key distribution, as well as to authenticate HMACs. Figure 3 illustrates the implementation for authenticating Processing Nodes (PNs) or Mobile Nodes (MNs). A PN or MN when requesting a connection, sends an Access-Request packet to the Authentication server (RADIUS server). In our tests, we utilized the Challenge-Handshake Authentication Protocol (CHAP) [RFC1944] and SHA-256 [RFC6234] as the digest algorithm. Using CHAP, the clear-text password is never sent through the network.

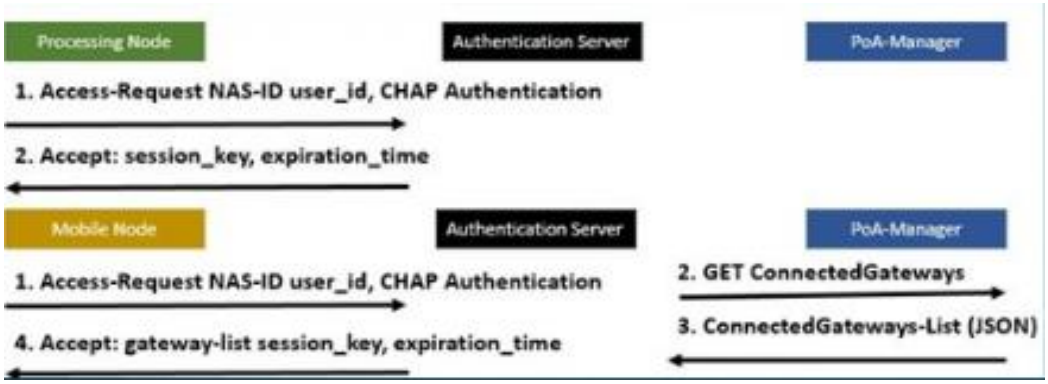

Figure 3: ContextNet entities authentication.

3.2 Centralized messages Authentication mechanism

Once MNs or PNs were authenticated, they receive a session key and a correspondent expiration time. In possession of session keys, MNs and PNs may sign messages using HMAC. For building the HMAC, each entity utilizes as input the output of a hash function over the message content in addition to its session key. Then, HMAC digest is attached to the message. It was used the SHA-256 as hash function for getting the hash of messages content and HMAC-SHA512 for HMAC function.

Receivers of messages recalculate hash of messages content. Additionally, based on senders identification, receivers consult the authentication service to validate the received HMAC. Figure 4 summarizes this process illustrating the exchange of messages between a Mobile Node and a Processing Node. The represented process describes the Centralized Authentication method. Round-trip time (RTT) was calculated based on the interval of time from the instant that a message is generated by a MN (step 1) up to the instant when it receives back the correspondent acknowledge issued by a PN (step 11). All RTT were registered at Mobile Node side.

The authentication process utilizes standard Access-Request, Access-Accept, Accounting-Request, and Accounting-Response RADIUS packets. They were customized to carry relevant information between the requesting node and the RADIUS server.

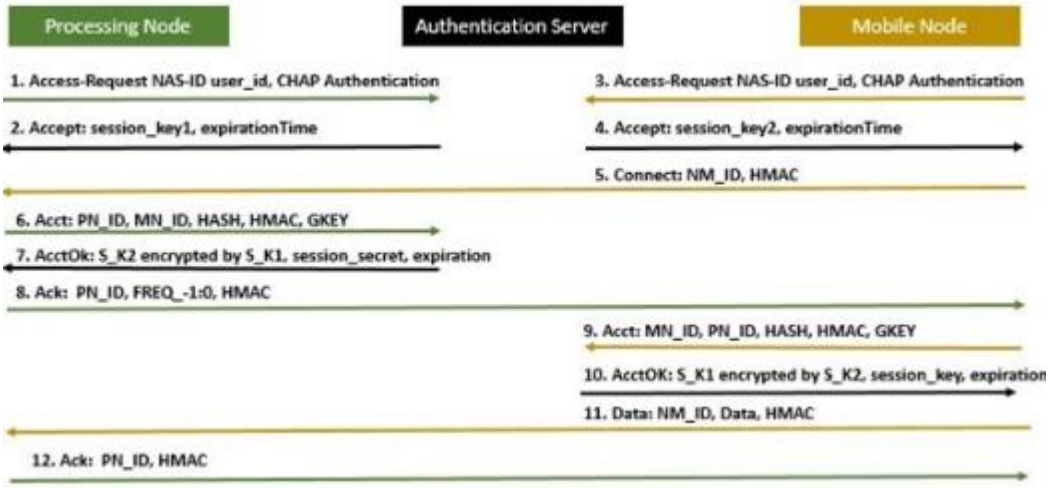

Figure 4: Centralized Authentication mechanism description.

3.3 Decentralized Authentication mechanism

A decentralized authentication method was also implemented. In this approach, when a MN or PN receives a message from an unknown sender, they consult the RADIUS server using an Accounting-Request packet to check if the sender was authenticated in the system. If it was, the RADIUS server sends the session key of that sender ciphered by the session key of the MN or PN that had done the request. In possession of the senders session key, it is possible to perform the validation of HMAC locally. Figure 5 summarizes the steps of this process.

It can be noticed that in a decentralized approach, there is an initial overload to validate the first received message remotely and to get the partners session key. After that, the validation of authenticity of messages is executed locally.

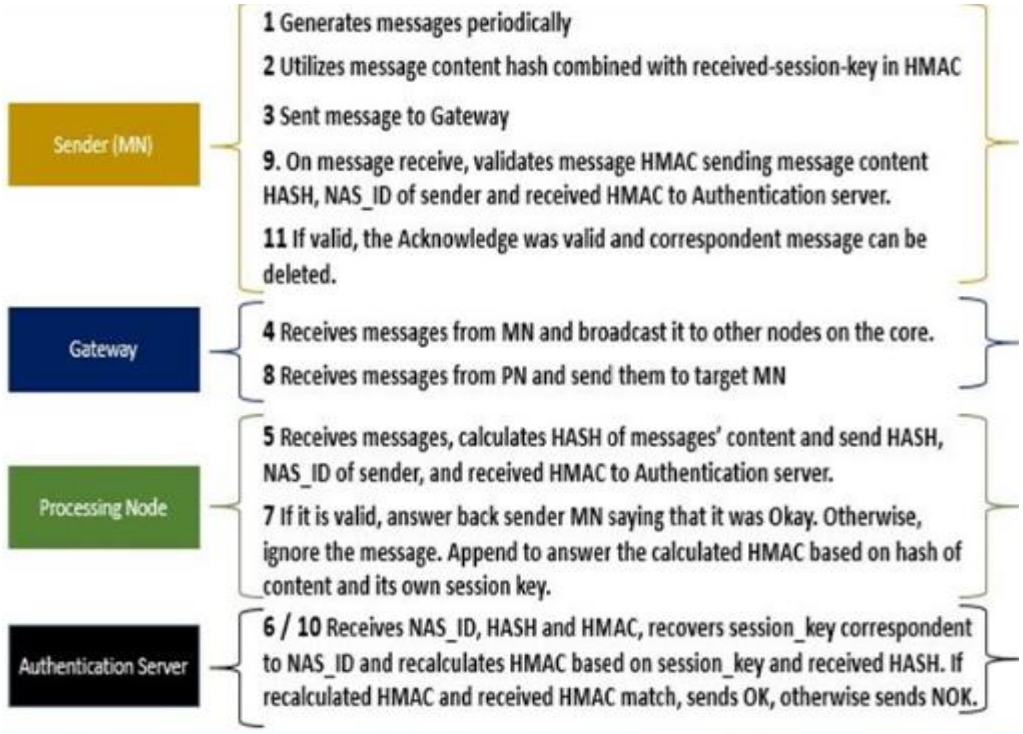

Figure 5: Description of steps taken in the decentralized Authentication mechanism.

4. Data manipulation, analysis and visualization

Our first task was to discover the capacity of our test bed environment through an incremental stress test. It was important to understand the limits of this environment. Otherwise, it would be impossible to differentiate the impact caused by authentication from other limitations in the environment. For instance, failures due to limits of the throughput capacity of our gateway or from the handling of a large number of connections in the mobile node simulator.

For the incremental stress test, a fixed number of connections was established at 4,000. It was based on previous tests using this environment. Then, increments on the frequency of messages up to crashing the system were performed.

The bootstrapping phase was separated from full-operational phase. Therefore, only after all the 4,000 mobile nodes had been connected that the simulator started sending telemetric messages. In order to perform this control, the Processing Node was responsible for counting the number of active mobile nodes until it reaches the preconfigured number of connections. When it was achieved, the PN sends a broadcast control message signalizing the desired periodicity of messages to be sent by MNs. When a mobile node receives that message, it starts sending messages periodically.

A disruption of the system was characterized by a crash in any of their components. A successful test was determined when all started mobile nodes were able to send successfully 1,000 messages each, and keeps the periodicity with delays not greater than 1% in mean. For example, using a frequency of 12 messages/minute corresponds to sending a message every 5000ms. Therefore, it was tolerate, in that case, delays, in mean for all messages up to 5050ms.

After discovering the maximum capacity of our test environment using a fixed number of connected mobile nodes, it was necessary to validate this limit. So, based on the reached maximum number of sent messages per minute by all Mobile Nodes, two new configurations were experimented. The reached estimated capacity was around 160K messages per minute regarding 4,000 Mobile Nodes sending 40 messages per minute.

Following Ousterhout (2018), an approach for validating a measure is measuring the same thing in different ways. Therefore, a corresponding configuration of 5,000 Mobile Nodes and 32 messages per minute (160K messages/minute) was successfully tested. Nonetheless, it was not possible to increase this frequency. Thereafter, another configuration with 3,200 Mobile Nodes and 50 messages per minute was also successfully tested. However, with this configuration, it was possible to increase around 4% the frequency of messages, raising to 52 messages per minute.

Remembering that our objective was not to find the best configuration to reach the maximum capacity of our system, but to find a reasonable capacity limit parameter. This capacity would drive configurations below that limit for comparing the additional burden caused by different authentication methods. Therefore, the found results were satisfactory for our objective.

Additionally, it signalized that our environment, more specifically, our gateway was more sensible to the number of connections than the frequency of messages. Results suggested that reducing the number of connections, it was possible to increase the

number of supported messages. Another evidence that the system worked better with a lower number of connections was got analysing the mean time for the simulator to send a round of messages for all Mobile Nodes.

For example, the first message should be sent by all 4,000 Mobile Nodes, so it was calculated the delta time from the first Mobile Node to send the first message up to the time of the last Mobile Node send its first message. The same was done for the second message and so on. The mean time was 3.613ms for 4K MNs / 40 msg/min, while it was 3.931ms for 5k MNs / 32 msg/min, but only 2.547ms for 3,2K MNs / 52 msg/min.

This also pointed out that our environment supported better a lower number of connections with a higher frequency than the opposite. Figure 6 shows the graphs with mean time for sending each message for the three different configurations used during the stress test.

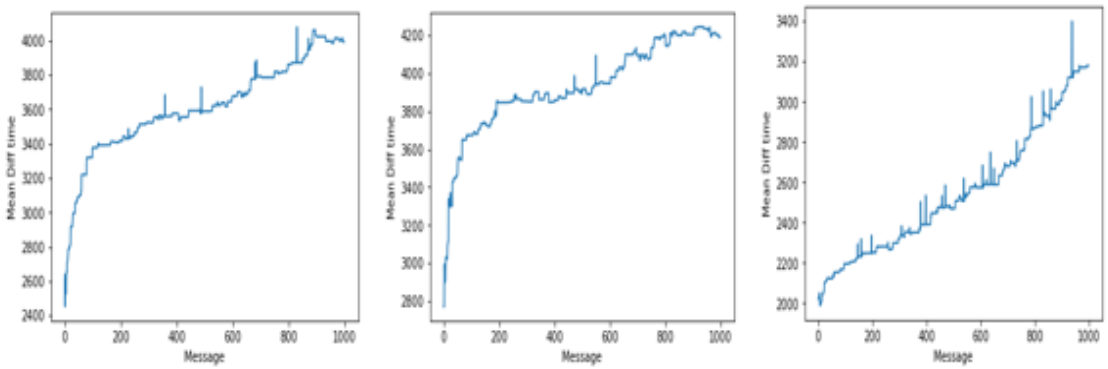

Figure 6: Mean round time to send each message in (ms) for each message. (Left) with 4,000 mobile nodes and 40 messages/minute – great mean = 3,613(ms (Centre) Setup with 5,000 mobile nodes and 32 messages/minute. Great mean = 3,931 ms. (Right) Setup with 3,200 mobile nodes and 50 messages/minute. Great mean = 2,547ms.

Once the maximum capacity of the system was established, it was possible to setup the tests in any configuration below that limit. These configurations should support some burden of adding new features, such as the authentication of messages in the system. It was decided to establish three levels of load in our test environment: light, moderate and heavy configuration.

Therefore, we performed a regression of analysis using the configuration of the stress tests with 4,000 Mobile Nodes and increasing the frequency of messages in order to have some parameters in determining those load levels.

A strong linear relationship was found between the frequency of messages and round-trip times with $r>0.8$, $p<0.001$. In mean, increasing the frequency by one, 13ms was added to RTT following the linear model as shown in Figure 7.

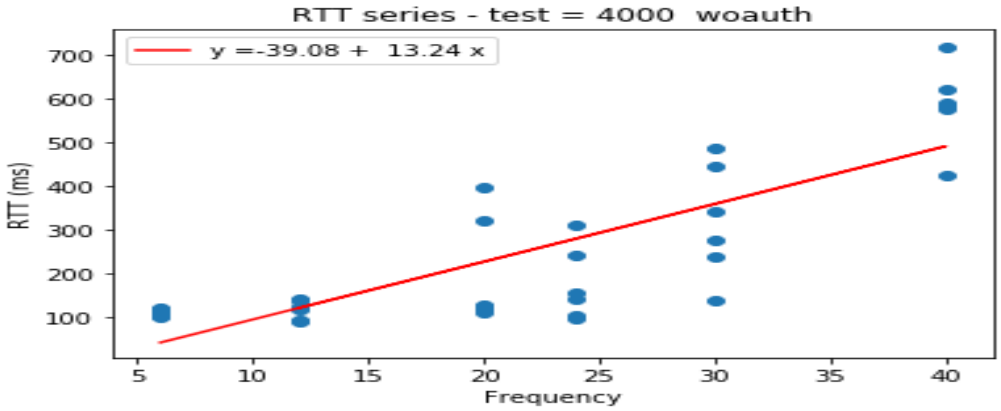

Figure 7 – Correlation analysis of tests with 4,000 mobile nodes and a crescent frequency of messages. A strong relationship was found with r=0.82, p=0.000 and a standard error=1.56 ms.

We speculated that it would be reasonable to establish a fixed periodicity at 12 messages per minute and a variable number of mobile nodes at 1,000, 2,000 and 4,000 to represent each of the desired load configuration. In order to check the established configurations, it was used the mean time for the simulator to send a round of messages. Table 1 shows those times.

Number of mobile nodes to send 12 messages per minute	Great mean time to send each round of messages (ms)
1,000	604
2,000	1,101
4,000	2,187

Table 1: Mean time to send each message by the number of mobile nodes.

It could be noticed the almost linear relationship between the number of mobile nodes and the mean time to send each round of messages by the simulator. It was also checked if the simulator was able to keep the desired frequency during the whole test. Note that the periodicity for sending 12 messages per minute is 5000 ms and the simulator stays far beyond this time for all configurations as shown in Figure 8.

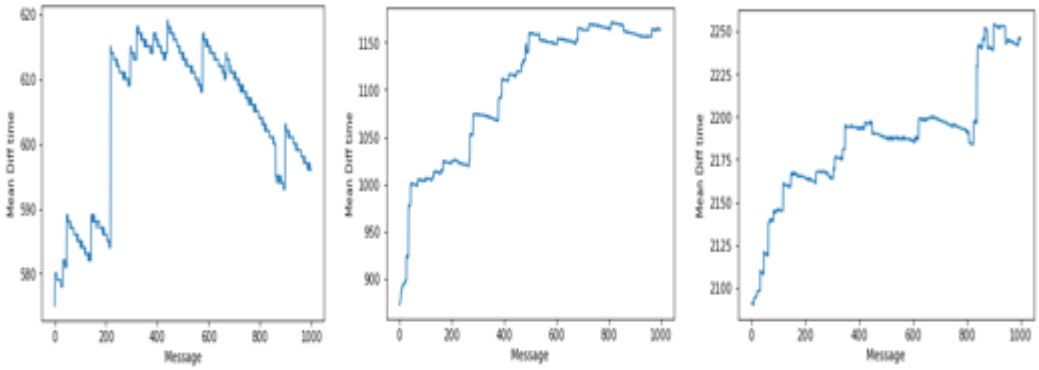

Figure 8 - Mean round time of the simulator for sending each of the 1,000 messages in (ms) at a frequency of 12 messages/minute. (Left) 1,000 Mobile Nodes. (Center) 2,000 Mobile Nodes. (Right) 4,000 Mobile Nodes.

The linearity in the simulator to process every message per round was also noticed in Round Trip Times using a progressive number of mobile nodes (500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000) and a frequency of 12 messages per minute. A linear re-

gression analysis between RTT and the number of mobile nodes demonstrated a significant moderate correlation of r=0.49 ($p<0.001$) and a standard error of 0.009 ms, as shown in Figure 9. In this chart it is possible to notice that with 4,000 Mobile Nodes, round-trip times were lower in mean than tests with a lower number of mobile nodes. Perhaps there was a not expected curvilinear relationship based on the results for 4,000 MNs. Another point to highlight was variability and the presence of outliers in our series, such as the higher dots at 2,000, and especially with 3,000 Mobile Notes.

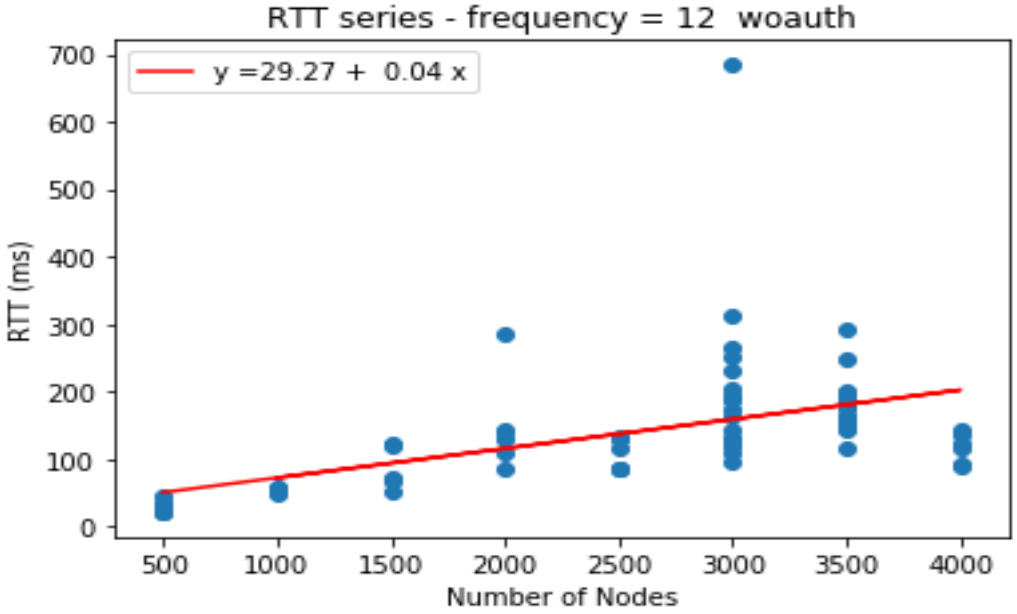

Figure 9 - Linear regression analysis of mean round trip time by a crescent number of mobile nodes.

It was also analysed the implementation of our simulation environment in the Windows Platform without authentication mechanism, as shown in Figure 10. Using Windows environment, tests with a frequency of 12 messages/minute, round-trip times got an exponential growth as the number of Mobile Nodes increased. The differences between Windows and Linux RTT were, 33ms, 43ms, 416ms and 2,551ms for 500, 1500, 2500, and 4000 mobile nodes setups, respectively. Running on Windows, the capacity of our system would be reduced to around 2,500 and 12 messages/minute.

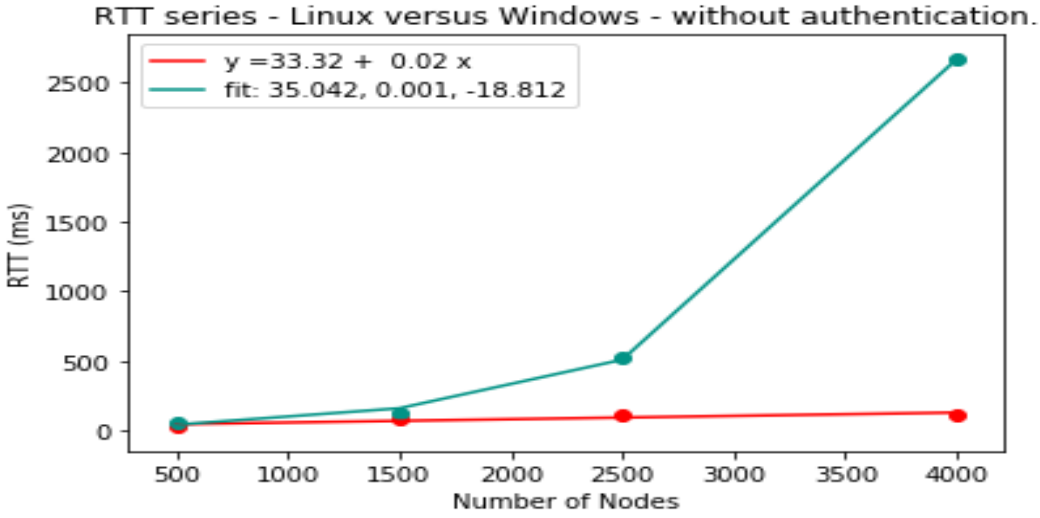

Figure 10 – Round trip times measured between environments with the gateway and processing node running on Linux (red) and in Windows Server (blue) and best fit line for a linear model (Linux) and exponential model (Windows).

Back to Linux environment, it was also noticed that the variability during the tests were very high. The scatterplot with regression model in Figure 10 for different number of Mobile Nodes indicated such variability observing the vertical distance of points for tests with the same configuration. In Figure 11, for the same tests it is presented the standard deviations of the series. It can be noticed that they were above the double of mean values for all the cases.

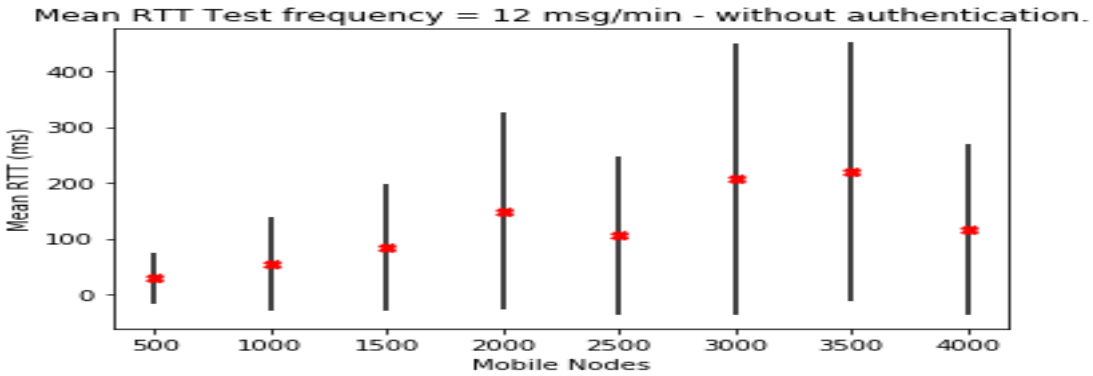

Figure 11 – Mean round-trip times (red crosses) and standard deviation (black bars) for tests with different number of mobile nodes. Standard deviations represent 156%, 153%, 136%, 118%, 134%, 118%, 105%, and 130% of mean values, respectively.

Given the huge variation during tests, it was necessary to investigate how it was presented during the tests. Looking at the mean round-trip times grouped by round of message and by time (minute) along the tests, it was not apparent a visual pattern (Figure 12). Figure 12 presents some examples of the tests. In the top left chart, apparently, mean RTT decreased from the beginning to the end of one test, but also increased and stabilized in another test (top right chart) observing the mean times for each sent message. Finally, the two charts on the bottom of Figure 12 show, when observing 6 runs of the same configuration grouped by time (minute), the variability along each test run as well as between runs.

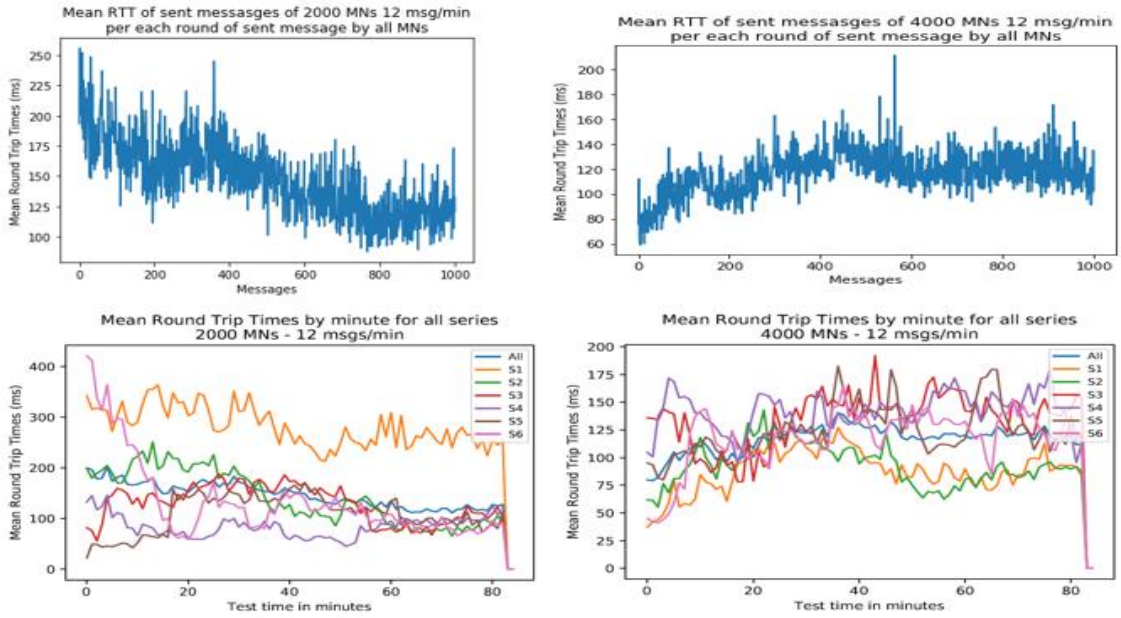

Figure 12 – Variability in mean round-trip times for configurations with 2,000 (left) and 4,000 (right) mobile nodes and frequency of 12 messages/minute grouped by message (top) and by time (minute).

In order to find the source of variability, it was investigated relationships between test variability and await time or service time for I/O requests in our simulator, but there was no clear relationship between those events as shown in Figure 13. The examples demonstrated that variability in mean round-trip time was independent of I/O request service.

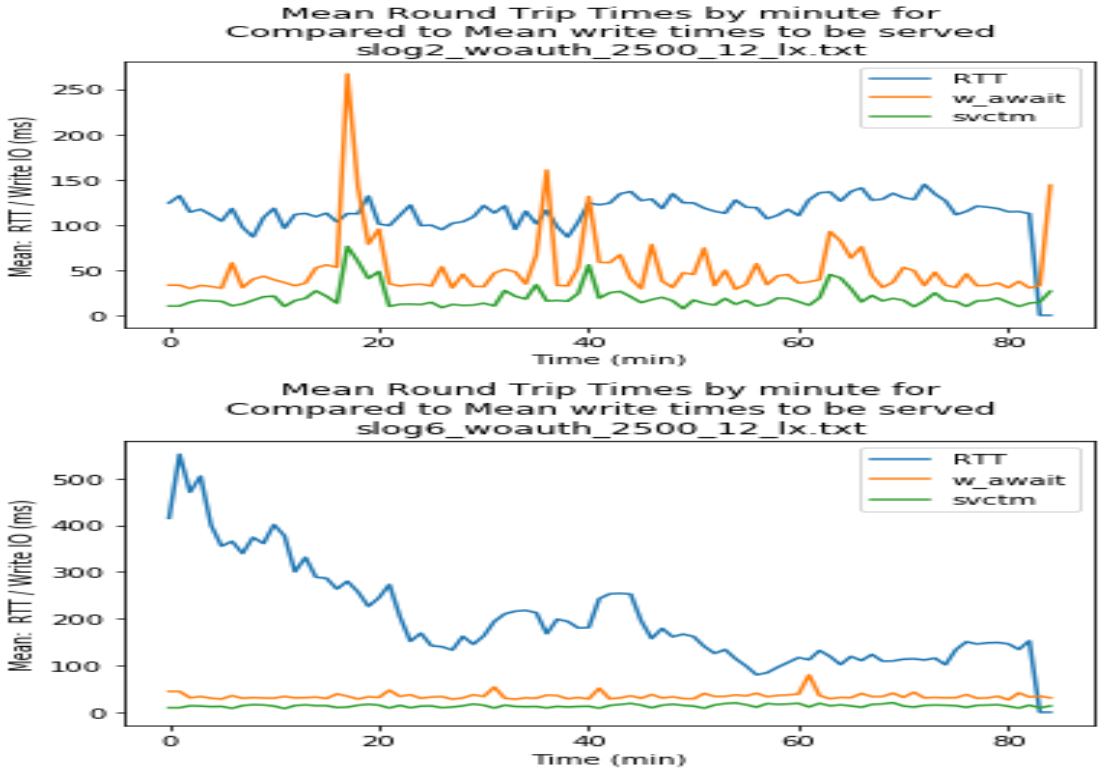

Figure 13 – Mean round trip time grouped by minute along test duration and average time for I/O requests example demonstrating that peaks in I/O request times were not related to round-trip time variability (top). At the bottom example, decreased tendency in mean RTT was totally unrelated to average I/O request times that stayed relatively stable during the whole test while mean RTT varied significantly.

High variability in network measurements with abnormal delay variability is a known fact in virtual machines (Wang and Eugene NG, 2010). Some studies explored the sources of variability (Whiteaker, Shneider, Teixeira, 2011). There are several sources of variability in the virtual environment, the extra-layer in distributing network packets or the calls in different moments for clocking applications out of kernel domain are some of possible issues. However, it was out of our scope of this work to investigate those sources.

Given that high variability was presented in our test environment, it was necessary to find a way to better represent our test behaviour for comparing the authentication scenarios. One possibility was to show the charts of the cumulative distribution function of the tests. Then, it allowed to verify the differences in each point of the frequency distribution.

Figure 14 presents the comparisons between cumulative distributions of mean round-trip times for tests of scenarios without authentication, with decentralized and centralized models of authentication and different number of mobile nodes. Clearly, centralized model introduces huge delays when compared to decentralized or without authentication models.

Surprisingly, observing the cumulative distributions the series with decentralized authentication in Figure 14, this authentication method performed better than without any authentication excepted for 4,000 Mobile Nodes and 12 messages per minute configuration. It was necessary to investigate deeper the reason of this behaviour.

Analysing the mean time to the simulator of Mobile Nodes to send each message by the 4,000 Mobile Nodes, it was noticed that the decentralized authentication method took more time in mean (3.1 seconds) than the centralized (2.7 seconds) and without authentication (2.2 seconds) methods, as shown in Figure 15. Probably, because the 4,000 Mobile Nodes had to process the authentication procedures locally what created this delay.

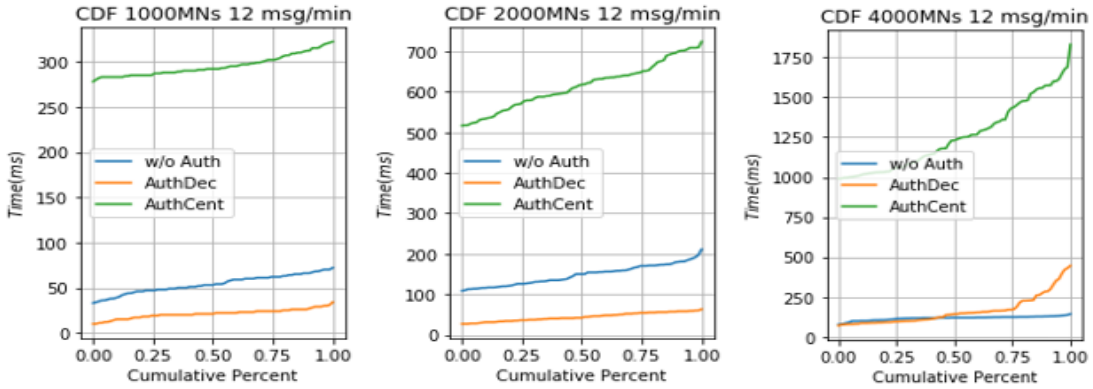

Figure 14 – Cumulative distribution function plots for tests with 1,000, 2,000 and 4,000 mobile nodes with a frequency of 12 messages / minute.

However, it was not explain possible to explain the better performance on mean round-trip times on data series with lower number of Mobile Nodes. It was hypothesized that this local processing that happens on the Mobile Node Simulator and on the Processing Node may have reduced the overhead created by context-switches among threads and the hypervisor of the virtual machines. Nonetheless, with our records it was not possible to investigate it. A deeper investigation was necessary to better understand these results.

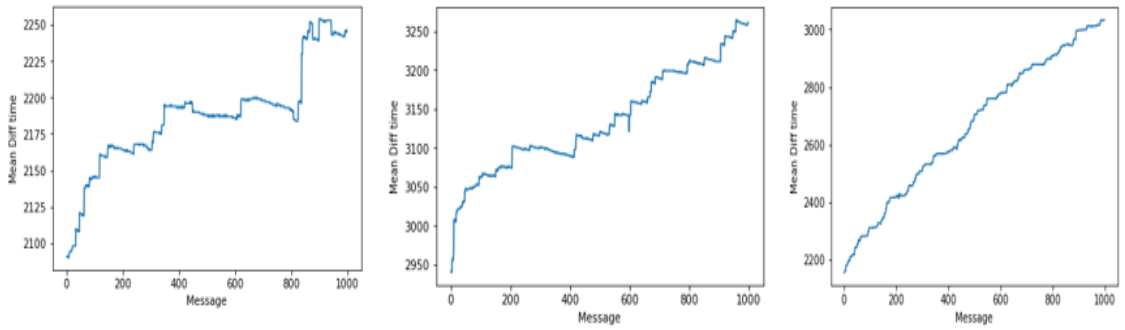

Figure 15 - Mean time to send each messages by all Mobile Nodes for 4,000 mobile node and 12 messages per minute frequency. No authentication (left) had great mean of 2,187ms, decentralized authentication of 3,136ms (center), and centralized authentication of 2,662ms.

The unique evidence supporting our theory is the data presented in Table 2. It shows that the mean time of records in our log files was the lowest in the decentralized model and higher in the model without authentication. In the centralized model, the Mobile

Nodes and the Processing Node had also to handle the messages to the authentication server that also required local CPU cycles. Our hypothesis is that reducing the context-switches from the higher rate of network requests, may have influenced positively mean round-trip times recorded at application level. However, this is just an speculation to be investigated in the future.

Log Frequency	Mean	STD	Min.	Max.	Median	P95
No Authentication	**5001.4**	**192**	**2342**	**15544**	**5000**	**5143**
Decentralized	**4999.4**	**228**	**1776**	**6921**	**5000**	**5378**
Centralized	**5000.5**	**215**	**998**	**8999**	**5001**	**5189**

Table 2: Mean frequency time for sending 12 messages/minute for the three Authentication methods with 4.000 Mobile Nodes.

Due to the high variability in our data series, it was necessary to check how trustable the information received from frequency distribution of mean round-trip times was. Another way of looking at those distributions was analysing the density of the series. Perhaps, analysing the probability density function it would be possible to infer given a random value for the round-trip time its probability to belong to one sample. From the density distributions it also would be possible to infer the uncertainty degree of those series.

In Figure 16, it was observed that charts of centralized authentication model for all configurations had higher probabilities densities. Mean round-trip times suffered less variation between each discrete value, and lower uncertainty regarding their data series values. On the other hand, densities were very low in decentralized and without authentication series with a high degree of uncertainty.

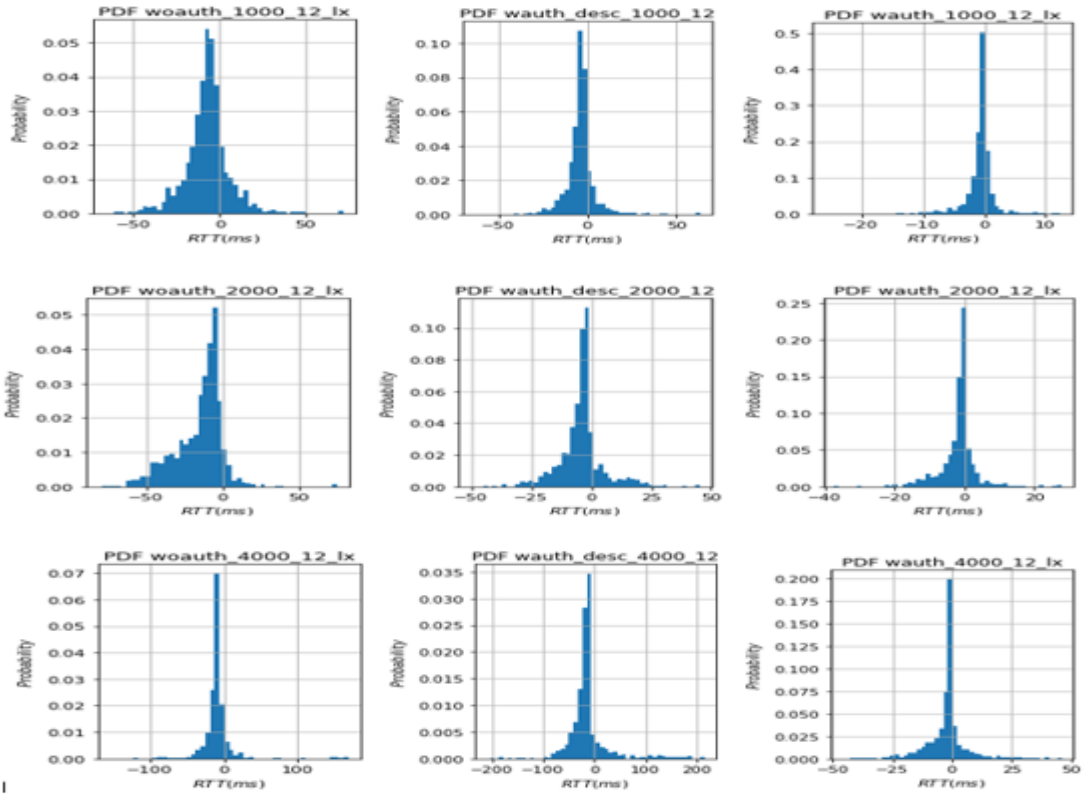

Figure 16 – Probability density function charts for configurations of 12 messages per minute and 1,000 (top), 2,000 (middle) and 4,000 mobile nodes for without Authentication (left), decentralized authentication (middle) and centralized authentication (right) models.

In order to check the entropy of the probability density series it was applied the normalized Shannon entropy formulas that confirmed the lower entropy of centralized authentication methods as showed in the Table 3.

Normalized Shannon Entropy by # of nodes	1,000	2,000	4,000
Without Authentication	0.83	0.85	0.80
Decentralized Authentication	0.75	0.77	0.87
Centralized Authentication	0.57	0.68	0.73

Table 3: Normalized Shannon Entropy coefficients for different Authentication methods and number of Mobile Nodes with a frequency of 12 messages per minute.

Considering that our primary objective was to measure the impact of adding authentication mechanisms on round trip times. Since, our results presented very high variability, it was not valid to compared directly mean RTT values. However, it still possible to compare distances in distributions using distances of their density functions. Therefore, the distances or divergence of the probability series were also evaluated using three different methods: Euclidean, Wooters and Jensen-Shannon entropy distance. The interested distance was between the series without authentication against the series with authentication, decentralized and centralized.

The Table 4 shows the distances coefficients found using different methods for calculating divergence between distribution series. Except for Wooters distance with 4,000 Mobile Nodes, all other results indicated a higher distance of Centralized method from corresponding series without authentication than Decentralized series. It confirmed that adding a centralized mechanism would cause a heavier impact on RTT than using a decentralized mechanism.

Distances without Authentication versus:	1,000	2,000	4,000
Euclidean Distance Decentralized	0.006	0.007	0.004
Euclidean Distance Centralized	0.031	0.016	0.009
Wooters Distance Decentralized	0.70	0.82	0.66
Wooters Distance Centralized	1.23	1.08	0.60
Jensen-Shannon entropy Distance Decentralized	0.21	0.19	0.16
Jensen-Shannon entropy Distance Centralized	0.30	0.23	0.23

Table 4: Distance coefficients of distributions using three different distance approaches, Euclidean, Wooters and Jensen-Shannon.

In the decentralized method, it evidenced that the local overhead caused by having few authentication operations in Mobile Nodes does not cause a greater impact in our simulator. The communication cost of transferring these operations to a central server, penalized heavily the response time in the application level even not considering all the issues related to wireless networks. The decentralized approach initial burden of exchanging session keys were not noticed in our test procedures because they happen during the bootstrapping phase after receiving the connection confirmation message. However, in the long term, any additional time of this one-time step would be diluted and insignificant in relation to every received message operations.

4 Conclusion

In this work, we utilized different data analysis techniques for testing the impact of adding authentication mechanism to our IoT middleware simulator environment. Firstly, it was analysed the capacity of our test system without any authentication mechanism and investigated the relationships between the number of connections and the frequency of messages in determining such capacity.

Thereafter, it was established three different load configurations in order to check the possible additional load of authentication mechanisms, a light, a moderate and a heavy load. Despite the linear relationship between the load of the system and the round trip times, there was a very high variability in results. Most of this variability seemed to be not related to local load of our virtual machines. It was not clear the main source of variability. However, it should be further investigated with a proper testing design for it.

Given the high variability, different analysis had to be performed in order to check the validity of our findings. Local time of our simulator to process each round of messages for all Mobile Nodes gave indicatives of different burdens using centralized and decentralized approaches, cumulative distributive function charts presented round trip times for all scenarios and indicated the high impact of the centralized system while the outperform of decentralized authentication confirmed that key distribution will have mostly a local and isolated influence more related to Mobile Node devices capacities.

Density functions also provided evidences of the increased variability of decentralized approach in relation to the centralized one. Perhaps the much higher round trip times of centralized authentication covered the inherent variability of very low times. Nonetheless, it really represented with a higher degree of certainty that round trip times were much higher given the density curves and distance entropy indices.

Entropy distances of authentication mechanisms from the baseline system were also calculated, decentralized authentication showed results closer to baseline than the centralized method. This was not a surprise given that the decentralized method have a small influence during the distribution of the keys and after that the local processing of hashes did not affected significantly the periodicity of messages.

Finally, it was not possible to fully-answer our main question about the impact on round-trip times of adding authentication of messages in our middleware due to the high variability in the different configurations and along the test runs. Nonetheless, it was possible to better understand that the impact is heavy in a centralized approach while in a decentralized mechanism it will depends more on the processing capacity of communicating participants, especially the Mobile Node devices. It may be irrelevant depending on the frequency of messages, though.

Our last, but not least learning was that performance tests should be designed with care given the high number of factors that may influence the outcomes. Automation of tests and data analysis is fundamental for finding and correcting errors, reproduce scenarios and verify the results. It is a cycle of many rounds that sheds lights in aspects not apparent in every module of a system.

References

AGARWAL, A. K. and WANG, W. Measuring performance impact of security protocols in wireless local area networks, in 2nd ITERNATIONAL CONFERENCE ON BROADBAND NETWORKS, 2005, Oct 2005, pp. 581–590 Vol. 1

BANDYOPADHYAY, S., SENGUPTA, M., MAITI, S., and DUTTA, S. A survey of middleware for internet of things, in RECENT TRENDS IN WIRELESS AND MOBILE NETWORKS, A. Ozcan, J. Zizka, and D. Nagamalai, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 288–296.

CHAQFEH, M. A., and MOHAMED, N. Challenges in middleware solutions for the internet of things, in 2012 INTERNATIONAL CONFERENCE ON COLLABORATION TECHNOLOGIES AND SYSTEMS (CTS) , May 2012, pp. 21–26.

ENDLER, M., SILVA, A., and CRUZ, R. A. M. S. An approach for secure edge computing in the internet of things, in 2017 1st CYBER SECURITY IN NETWORKING CONFERENCE (CSNet), Oct 2017, pp. 1–8.

FREMANTLE, P. and SCOTT, P. A survey of secure middleware for the internet of things. PeerJ Comput. Sci., no. 3:e114, 2017.

HANSEN, T. and 3rd, D. E. E. US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF), RFC 6234, May 2011. [Online]. Available: https://rfc-editor.org/rfc/rfc6234.txt

HASSEL, J. Radius. O'Reilly Media, 2002. [Online]. Available: https://www.amazon.com/Radius-Jonathan-Hassell/dp/0596003226?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0596003226

KRAWCZYK, D. H., BELLARE, M., and CANETTI, R. HMAC: Keyed-Hashing for Message Authentication, RFC 2104, Feb. 1997. [Online]. Available: https://rfc-editor.org/rfc/rfc2104.txt

OUSTERHOUT, J. Always measure one level deeper. Communications of the ACM, vol. 61, no. 7 pp. 74-83, 2018.

RUBENS, A., RIGNEY, C., WILLEN, S., and SIMPSON, W. A. Remote Authentication Dial In User Service (RADIUS), RFC2865, Jun. 2000. [Online]. Available: https://rfc-editor.org/rfc/rfc2865.txt

SFAR, R., NATALIZIO, E., CHALLAL, Y., and CHTOUROU, Z. A roadmap for security challenges in the internet of things, Digital Communications and Networks, vol. 4, no. 2, pp. 118-137, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352864817300214

SIMPSON, W. PPP challenge handshake authentication protocol (chap)," RFC 1944, Aug. 1996. [Online]. Available: https://www.rfc-editor.org/info/rfc1994

WANG, G., EUGENE NG, T.S. The Impact of Virtualization on Network Performance of Amazon EC2 Data Center," *2010 Proceedings IEEE INFOCOM*, San Diego, CA, 2010, pp.1-9. doi: 10.1109/INFCOM.2010.5461931

WHITEAKER, J., SCHNEIDER, F., TEIXEIRA, R. Explaining packet delays under virtualization. ACM SIGCOMM Computer Communication Review, vol. 41 (1), 2011.

WEI, J. Ddos on internet of things a big alarm for the future. Department of Computer Science, Tufts University, Tech. Rep., 2016.

ZUMEL, N., MOUNT, J. **Practical data science with R**. Manning. (2014) ISBN: 9781617291562 1617291560