Theoretical and Applied Informatics
Vol. 28 (9), no. 3, pp. 17–35
DOI: 10.20904/283017

Secure gateway for Internet of Things with internal AAA mechanism

Dominik Samociuk[1]* Błażej Adamczyk[1]

[1]Institute of Informatics
Silesian University of Technology
Akademicka 16, 44-100 Gliwice, Poland

Abstract In this paper, we describe secure gateway for Internet of Things (IoT) devices with internal AAA mechanism, implemented to connect IoT sensors with Internet users. Secure gateway described in this paper allows to (1) authenticate each connected device, (2) authorise connection or reconfiguration performed by the device and (3) account each action. The same applies to Internet users who want to connect, download data from or upload data to an IoT device. Secure Gateway with internal AAA mechanism could be used in Smart Cities environments and in other IoT deployments where security is a critical concern. The mechanism presented in this paper is a new concept and has been practically validated in Polish national research network PL-LAB2020.

Keywords Internet of Things; security; AAA mechanism

Received 15 MAR 2016 **Revised** 03 MAR 2017 **Accepted** 17 MAR 2017

1 Introduction

The ability to connect, communicate and remotely manage different devices has led to rapid development of the Internet of Things (IoT) technology. There were fears that this development is happening too fast, without proper handling of safety issues and regulatory changes that may be necessary. According to surveys conducted by the Business Insider in the last quarter of 2014 [1], 39% of respondents believed that the privacy and security aspects of IoT are the biggest barrier to investing and implementing technology in their companies. Further, there were doubts about the Return on Investment (ROI) - 27% and the lack of real function in a corporation - 16%. In particular, while maintaining the current speed of development and the spread of technology, cyber attacks may become more and more physical (not just virtual) threat. Joseph

*E-mail: dominik.samociuk@polsl.pl

Steinberg's article [2] listed a number of devices that can now spy on people in their homes, including televisions, kitchen appliances, cameras, thermostats or baby monitors. House obtain intelligence that can be understood in two ways:

- as the ability to acquire and apply knowledge and skills,
- as the collection of information of military or political value.

Similarly, mechanisms in automobiles such as brakes, engine, the driver-aid systems (for example ABS) and systems providing comfort during a ride (for example air conditioning) more and more often can also be controlled remotely. There are some documented cases, where these systems were hacked and exploited remotely:

- hijacking system of Chrysler's cars [3], in which researchers completely took control of the car and were able to activate the brakes remotely.
- remote unlocking of BMW brand cars [4], in which attackers spoofed the BMW server communicating with the car (in order to draw latched keys) what among other allows to unlock the car.
- extraction of data from internal systems of the vehicle based on the USB dongle hack inserted into the radio [5]. Originally it was prepared for monitoring driving performance for insurance companies but due to lack of authentication and authorization mechanism opens the possibility to hack the car's internal systems.

As part of the Horizon 2020 program, the European Commission released a Work Programme for years 2016-2017 ([6] - Commission communicate dated 13.10.2015). Among the key priorities the programme specifies: "A Connected Digital Single Market" which includes the following: the Internet and digital technologies are changing the modern world. The activities, which are to contribute to the development of innovative digital solutions are as follows:

- the area of Internet of Things (139 million euros) will complement the technological developments of large-scale pilot projects carried out in areas of major social challenges.
- Digital Security area (118 million euros) will respond to the opportunities and risks associated with the processes of digitisation and computerization.

Among already completed project one worth noticing is the SmartSantander project [7]. It proposes a unique worldwide experimental research centre for typical applications and services in a smart city. This unique experimental facility is large enough, open and flexible to allow the creation of a federation with other experimental objects and stimulate the development of new algorithms and platforms by users of various types, including advanced technology research, the Internet of Things and a realistic assessment of user acceptability tests. The facility consists of more than 20,000 sensors and is based on realistic distribution facilities in the city. The heart of the facility is located in Santander, the capital of Cantabria region, situated on the northern coast of Spain. SmartSantander allows the future Internet of Things to become a reality [8].

In this paper, we describe a secure gateway for Internet of Things devices with internal Authentication, Authorisation and Accounting (AAA) mechanism, implemented to connect the devices with the national research network PL-LAB2020 [9] (a large testbed and experimental network being built in Poland right now). Secure gateway described in this paper allows to (1) authenticate each connected device, (2) authorise connection or reconfiguration performed by the device and (3) account each action. The same applies to Internet users who want to connect, download data from or upload data to an IoT device.

This paper is organised as follows. In Section 2, we discuss related works. Section 3 presents an overview of the proposed system architecture. It additionally defines and explains the AAA mechanism and describes its components. In Section 4, we present technical details of the proposed solution, including a secure connection of the IoT device to Internet and communication process between Internet users and the IoT device. In Section 5, a brief validation of proposed solution is carried out considering performance, security and energy consumption aspects. Section 6 concludes the paper and presents our planned future work on the subject, including the deployment process on PL-LAB2020 and testing solution in wide area network topology.

2 Related work

Research topics that contribute to the development of IoT are carried out by many communities and are launched from different perspectives. They usually share common or similar problems what induces the cooperation between these communities in order to solve them effectively. In this chapter, we will highlight and discuss seven key research topics and how they relate to the two-way secure IoT communication described in this paper.

1. Architecture

   Billions of things connected to the Internet need principles that describe control, communication and applications. Researchers try to define and develop a good IoT architecture e.g. [10], and how the devices will connect with and between platforms e.g. [11]. The communication scheme and routing protocols are well studied in the literature, and there were many articles regarding the comparison of different protocols and their efficiency e.g. [12, **?**]. In IoT, similarly as in other architectures (e.g. [13]), security is an essential element. Thus, the secure communication between the Internet (users, applications) and things (devices) is crucial for further development.

2. Robustness

   IoT application will be based on sensing, actuation and communication. A crucial aspect of proper deployment and usage is synchronisation or in other words solving the problem of deterioration of clock synchronisation, see [14]. Another interesting aspect is the required entropy [15] and running time assurances by authorities to certify that the system is secure and working as expected [16].

3. Big Data

   In IoT the amount of collected data will be enormous. There are techniques to collect [17], convert and send this data stream to the proper server application. One of the areas where this technique will be applicable is the medicine. However, security and privacy maintenance should be mission-critical when validating solutions before production deployment.

4. Scaling

   There are estimates that IoT will consist at least 50 billion objects by 2020 [18]. With the current pace of evolution, several questions must be answered. How will massive amount of data be collected, transferred and stored? Will addressing protocols be enough? How to provide reliability? These topics are discussed for example in [19] and [20]. Of course aspects regarding authenticating access, protection, authorization and accounting can not be neglected when considering scaling.

5. Open source

   Currently, most of the sensor-based platforms are closed systems, without direct access to the code. Their security is sometimes assessed by a third party, only by means of black-box testing - [16]. Due to the fact of rapid development of new platforms, it is better to focus on open platforms that allow co-operating easily between different vendor devices. From the security perspective open platform allows for better assessment by the community.

6. Security

   As mentioned in the Introduction, the fundamental problems in IoT systems are security, privacy and dealing with security breaches [21], [22]. As mentioned in [21], unique solutions (as the one presented in this paper) will be necessary for providing authentication, authorization and accounting.

7. Human interaction

   As Human-Device interaction in IoT is one of the crucial aspects, developing a new mechanism to track data received from humans (e.g. [23]) is another important research topic. Secure communication and applying security policies to such interaction should be mission-critical for developers and architects.

There are numerous papers dealing with IoT Gateways, built on embedded devices such as Raspberry PI [24] or standalone servers (as in the proposed solution) [25]. However, to the best of the authors' knowledge, there is no work presented in creating Secure AAA solution for Internet of Things devices. Especially, we are not aware of any paper presenting AAA Gateway connecting several IoT devices (for example built with Raspberry Pis) together and providing the two-way security (AAA for both - Internet users and the devices). It is important to assume that the objective of this work, was to create a secure gateway, not secure nodes (devices), so security mechanisms are implemented on a gateway and not on the nodes (however nodes are also secured as much as possible).

3 System components

In the following section, we present the proposed system components. First, we describe AAA mechanism, how authentication, authorization and accounting systems work. How it can be deployed in the production environment and which architecture we used in our work. The second part is a list of devices and software we used to build and validate our solution, with short description and explanation of each component.

3.1 AAA mechanism

Authentication, Authorisation and Accounting (AAA) is a mechanism for (1) identifying the other end, (2) defining policies to control access to resources, (3) auditing and reporting on actions performed in the system. The combined processes are considered important for effective and safe network management. Securing network services according to the AAA mechanism provides a basic framework through which you can configure access control on an input device - in our case IoT Gateway. The characteristics of AAA components is described in the following subsections.

3.1.1 Authentication.

In the first process, authentication provides a method for identifying a user (or device), usually by entering the correct username/login and the correct password before access is granted. The authentication process is based on the possession of unique user's credentials. AAA server compares them with user's credentials stored in the database. If they match, the user gets access to the network. Otherwise, authentication fails, and network access is denied.

3.1.2 Authorisation.

After authentication, the user must obtain a permit to perform certain tasks. The authenticated user sends a request to the server for a certain action (in the case of this article, for example, it could be downloading IoT device data). The authorization process determines whether the user has permission to request this action. Namely, authorization is a process to enforce security policy: determine what kind of action (e.g. which resources or services) the user can perform. The granularity of authorization depends on application but usually systems distinguish several levels of access rights, such as: direct rights - e.g. “User ABC is allowed to download data gathered on device XYZ”, group/role based rights - e.g. “Users having ABC role are allowed to download data gathered on device XYZ”.

3.1.3 Accounting.

The final element in the framework of the AAA is the accounting, which logs all user/device actions performed during access. It could be the amount of time on the system, the amount of data that user sent or received during a session, or even the actual actions that user performed during it. Accounting is carried out by recording the session statistics and information and is used to control the authorization, billing, trend analysis, resource utilisation and planning activities. Accounting keeps track of how network resources are used, e.g. ”User ABC accessed device XYZ for 15 minutes and requested access to following data REQUEST-LIST.”

3.1.4 Possible architectures.

There are three main ways to implement AAA mechanism:

1. Local AAA database

   AAA service is locally-contained on the gateway itself. This type of architecture is also known as local authentication and will be used in the solution described in this paper.

2. Access Control Server

   AAA service running on gateway connects to some external Access Control Server (ACS) where all authentication and authorization data is stored.

3. Identity Service Engine

   AAA service connects to Identity Service Engine (ISE) to define and enforce security polices.

The ACS and ISE architectures need additional, dedicated software and/or hardware. Also, there is a more elaborate scheme of communication; namely, there is no direct communication between client and server during authentication/authorisation process. In this paper, we present the simplest mechanism, created on the same hardware as actual IoT Gateway minimising needed hardware and topology.

Another aspect related to AAA are two different types of authentication:

- Character mode

  The user that wants to establish the connection and authenticate, sends login/password when prompted by a gateway. In our solution used when authenticating the user connecting to IoT device.

- Packet mode

  The user sends to gateway a packet with proper data and certificates to get access to the network. In our solution used when IoT device registers and wants the connection to Internet.

3.2 Solution components

Architecture for implementing and validating proposed solution contains server acting as a proxy between IoT devices and Internet users with AAA mechanism, IoT Device sensing environment and presenting data using web-interface and communication scheme based on the user-proxy-device path and both-sided AAA mechanism.

3.2.1 Gateway with AAA mechanism.

As a gateway between Internet users and IoT device, we use a server with Intel Core i7-3610QM processor, Kali Linux operating system, 32GB DDR3 1600Mhz memory, HDD storage, 1Gbps network card with Ethernet RJ45 connector and b/g/n wireless card. From the software perspective, the main part is AAA mechanism. In Linux environment, we install Remote Authentication Dial In User Service (RADIUS) Server and update local database with authentication, authorization and accounting rules. Also, Public Key Infrastructure for secure communication with IoT device has been deployed on the gateway.

RADIUS is a remote authentication service for users who want to connect to the system (in the case of the proposed solution these will be users from the Internet accessing IoT Device web-interface). It is currently a very popular protocol for authentication and authorization of users. It is also used in wireless networks. In response to the attempt to log into the network, network access server (gateway) generates a request for user information, including the user ID/login and password. After retrieving the answer from the user, the identifier along with the encoded password are sent to a RADIUS daemon. After checking the user data, their confrontation with the contents of a local database, RADIUS server can answer in one of the following messages:

- ACCEPT - means the success of authentication,
- REJECT - the user is not properly authenticated, access to network/IoT device resources is prohibited.
- CHALLENGE - prompt to enter additional credentials.

After passing the first phase successfully, the RADIUS server checks the database, what services are available to the user (requests to web interface). The RADIUS server for authorization phase also checks whether the actions of the user on the network should not be subject to restrictions that result from the access lists deployment. The last phase would be to log proper data/request-s/actions as accounting phase until connectivity between user and device has been terminated.

3.2.2 IoT Device.

As an IoT device, we assumed and used the Raspberry Pi 2 Model B, which is the second generation Raspberry Pi (replacing Raspberry Pi 1 Model B+). This model features a more powerful processor quad-core ARM Cortex-A7 900 MHz and more memory - 1 GB of RAM. The kit features peripherals, which include four USB slots, an additional 40 GPIO connectors, a microSD card, an Ethernet port and four mounting holes. We retrofitted device with USB WiFi Adapter (to connect to IoT Gateway) and Raspberry Pi Camera Board Module (to act as a sensor), collecting data - accessed from web-interface by Internet users.

3.2.3 Communication scheme.

In our solution, we chose the proxied approach over direct access as shown in Figure 1.

From IoT device perspective, when connecting it to the Internet through the gateway, it utilises registration and authentication scheme using mechanisms described below in Algorithm section. After successful connection, it sets up classic Web or WebService interface (secured with HTTPS protocol) where users can download data obtained using installed and configured sensors and/or reconfigure device remotely.

When Internet user needs to access IoT device, it sends a direct request, however, intercepted by a gateway. The gateway stops forwarding the request to IoT device until AAA mechanism authenticates and gives the user proper access level to the device. After successful check against RADIUS server, the request is forwarded through a secure channel to IoT device. Specifics are described in Algorithm section.

4 Algorithm

For the purpose of this paper, we assume that all the IoT Devices are managed by a Linux-based controller (for example a Raspberry PI or other similar device). Every device presents its output

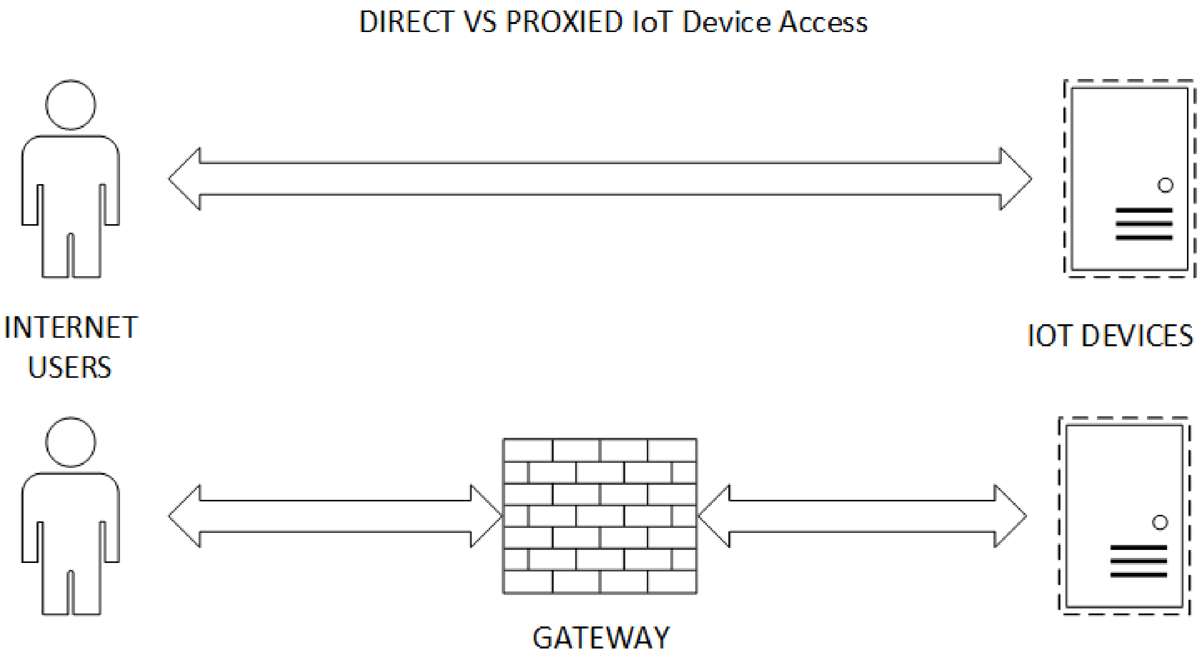

Figure 1 IoT device deployment options.

through a simple HTTP server. The idea presented in this paper is to introduce a Gateway device which provides the AAA. The Gateway works like an HTTP proxy server forwarding the HTTP request further to the IoT Devices and bringing the responses back to the client. The flow of information is depicted in Fig. 2.

The Gateway needs to secure both communication channels: client and IoT side. These security mechanisms are described in the following subsections.

4.1 Users accessing IoT device

The user initiates the process with generating http (https) request directly to IoT device. The request is intercepted by a proxy gateway, and the user starts the authentication process. Gateway prompts for username and password. User replies. On the gateway, RADIUS client sends username and an encrypted password to the RADIUS server. RADIUS server responds with Accept, Reject, or Challenge. The RADIUS client acts upon services and services parameters bundled with Accept or Reject. The gateway passes additional access list entries down to the network interface configuration to allow the users through after authentication. Request to IoT device is forwarded, rest of the communication is performed until TCP session between the user and IoT device has been terminated or reaches timeout. All actions are accounted on the gateway with requests and corresponding timestamps. The complete process is shown in Figure 3.

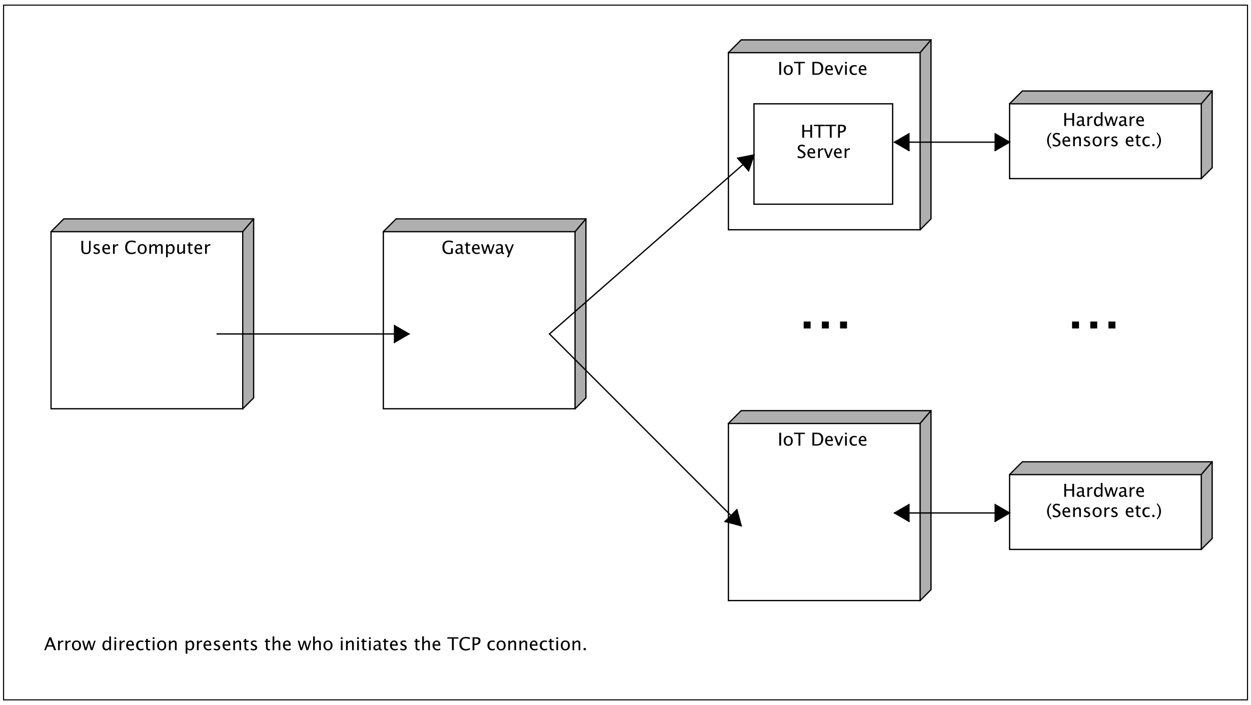

Figure 2 Flow of information.

4.2 Secure IoT Device Communication

The internal IoT Device network in some applications may be considered as secure. But quite often this is not the case - especially when IoT networks are becoming gigantic, and the devices are distributed across a large area (e.g. *Smart City*). Sometimes the configuration may require using an untrusted network or even public Internet instead of a dedicated internal one. In such cases, it is necessary to additionally secure the communication between the Gateway and the IoT Devices.

The proposed solution is to provide a two-way authentication, authorization and communication encryption by using the existing SSL/TLS Client Authentication [26] mechanism. In order to do this properly, it is necessary to create a Public Key Infrastructure (*PKI*)[27] and generate certificates for all the IoT Devices as well as the Gateway.

Certificate Authority (*CA*) is the main component responsible for issuing all certificates used in the system. By design it is stored on the Gateway and at the beginning contains only the *root* self-signed certificate and private key. Using this key, the CA signs the certificates for Gateway and all IoT Devices. Such architecture provides the following:

1. allows the Gateway to authenticate all IoT Devices,
2. allows the IoT Devices to authenticate the Gateway,
3. prevents communication eavesdropping between Gateway and IoT Devices, and

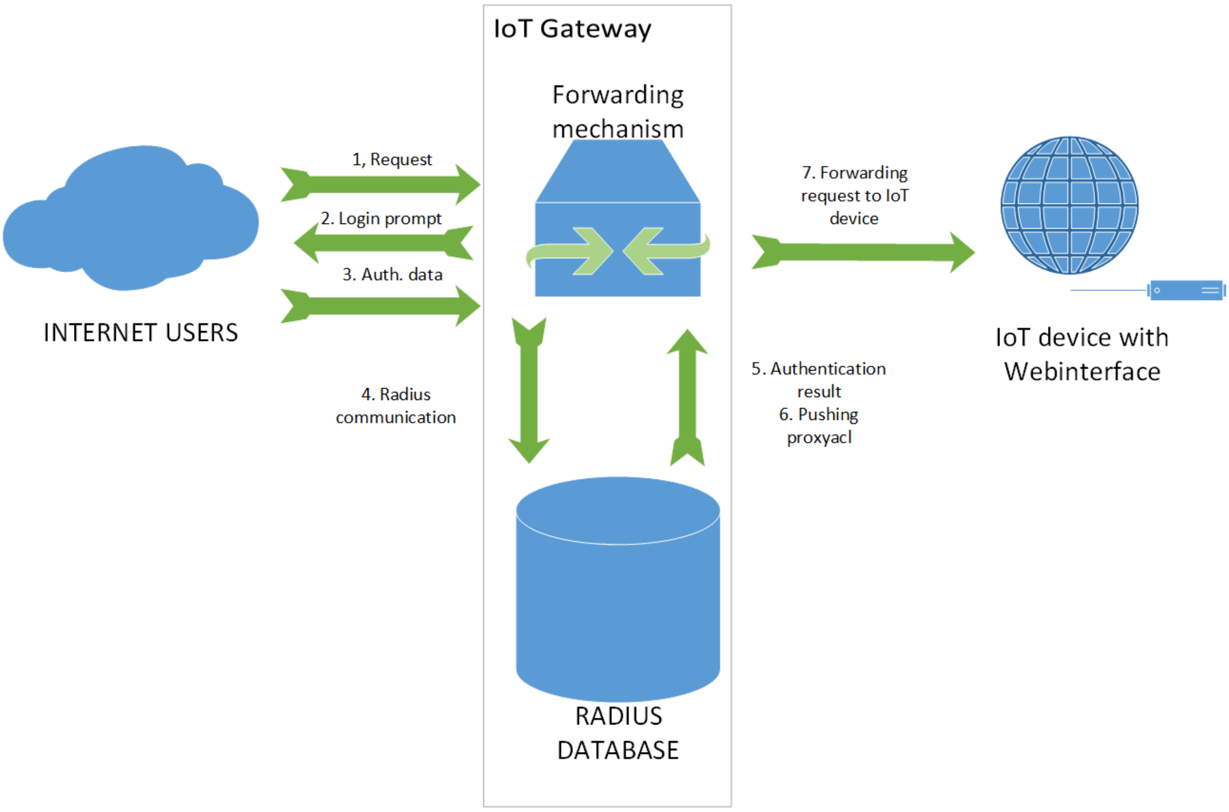

Figure 3 Users accessing IoT device through gateway with internal AAA mechanism.

4. allows using existing HTTP servers (the HTTPS client authentication is supported by the majority of HTTP servers).

Before the Gateway can provide the device's data to users, it needs to register all the IoT Devices. It is a preliminary phase and is done only once for each device. After the registration, the Gateway is ready to forward client requests further to the devices.

4.2.1 Registration

Every device before being published through the Gateway needs to be approved by an administrator and registered by signing its Certificate Signing Request (*CSR*). This process is initiated from IoT device. The whole registration process is presented in Fig. 4.

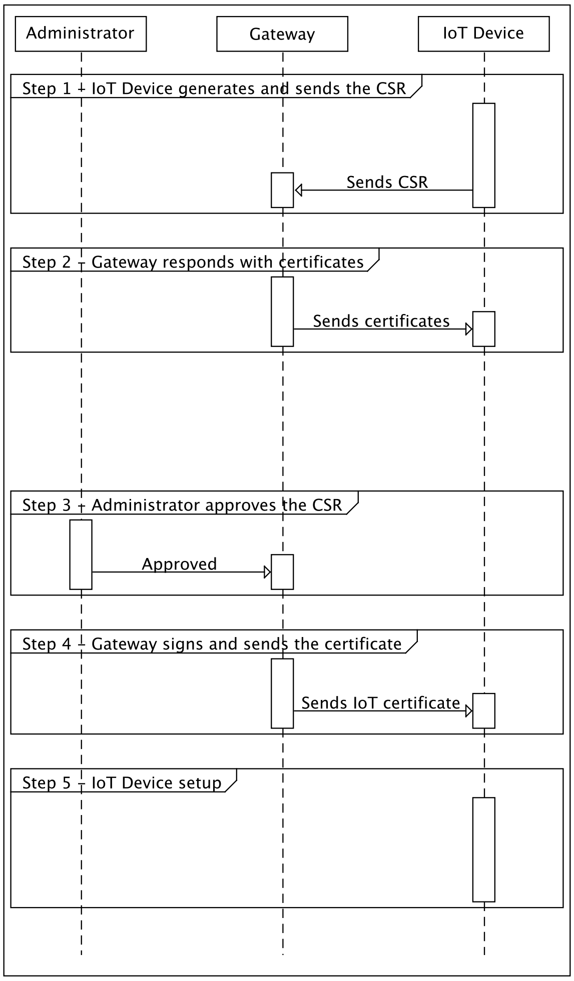
Administrator
Gateway
IoT Device
Step 1 - IoT Device generates and sends the CSR
Sends CSR
Step 2 - Gateway responds with certificates
Sends certificates
Step 3 - Administrator approves the CSR
Approved
Step 4 - Gateway signs and sends the certificate
Sends IoT certificate
Step 5 - IoT Device setup

Figure 4 IoT Device registration process.

The following steps (presented in Fig. 4) are necessary to perform device registration:

1. IoT Device:
   - generates its private key and the CSR,
   - authenticates using a password at the Gateway Registering Web Service,
   - sends the CSR to the Gateway Registering Web Service.
2. Registering Web Service:
   - stores the CSR for administrator approval,
   - returns the Gateway and root certificates to the IoT Device.
3. Administrator:
   - logs into the Gateway Portal,
   - verifies the CSR fingerprint and approves the CSR.
4. Gateway:
   - signs the CSR using CA root certificate,
   - sends the signed device certificate to the IoT Device.
5. IoT Device:
   - verifies if the certificate is signed by the root CA,
   - configures its HTTP server to use HTTPS using its signed certificate and private key,
   - configures its HTTP server to require client certificate equal to the Gateway certificate,

4.2.2 Authentication, Authorisation and Encryption

When the registering process is finished, the IoT Device is fully configured and can properly handle the HTTPS requests coming from the Gateway proxy service. When a user requests the data from a given IoT Device (the client side authentication and authorisation were described in section 4.1) the Gateway proxy service forwards the request to the IoT device using and verifying the certificates retrieved in the registration phase. The whole process is depicted in Fig. 5.

The process is initiated by the user (step 1 in Fig.5). The user requests a concrete IoT Device data and thus the Gateway proxy has to redirect the request to the IoT HTTP Server. First, an SSL/TLS handshake is performed (step 2 in Fig. 5). It has been fully described in [26]. In short words - the IoT HTTP server is configured to authorise only those clients who use HTTPS client authentication and identify them with Gateway certificate and key. The Gateway, on the other hand, using the TLS/SSL protocol, accepts only a server certificate and key which was signed by the root CA. Finally, when both sides are verified, the handshake ends when a symmetric encryption key is negotiated by the client and the server. After the successful handshake, the communication is fully encrypted.

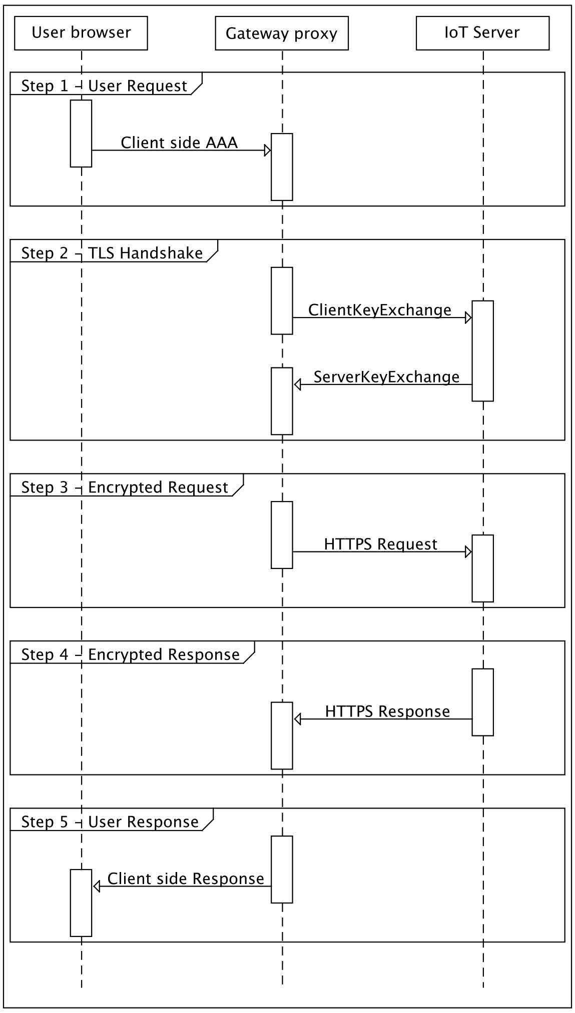

Figure 5 Standard communication process between Gateway and IoT Device.

Further, the Gateway (SSL/TLS client in this case) can send the request using the HTTP protocol over an encrypted and secure SSL/TLS channel (step 3 in Fig. 5). Further, the IoT Device verifies if the SSL/TLS client certificate matches the Gateway certificate (step 4). If the requester is indeed the Gateway, the IoT Device can finally respond with data. In the end, the Gateway proxy re-sends the response to the user (step 5).

5 Solution validation

After successful implementation, authors performed several tests to validate proposed solution. Namely three categories was questioned:

- Security

  Top ten vulnerabilities committed by developers and architects were listed and checked if proposed solution mitigates them.

- Performance

  Latency when using a standard mechanism (without security), secured using https without authentication, authorization and accounting, and secured using both-sided https with AAA proxy, was measured and compared in a few scenarios.

- Energy consumption

  The last category was energy consumption of IoT device with and without security functions presented in this paper.

5.1 Security

The OWASP Internet of Things (IoT) Top 10 [28] is a project *designed to help manufacturers, developers, and consumers better understand the security issues associated with the Internet of Things, and to enable users in any context to make better security decisions when building, deploying, or assessing IoT technologies.* The project defines the top ten security vulnerabilities areas significant to IoT world and provides information on attack vectors, vulnerabilities, impacts and recommendations associated with each. The list of vulnerabilities with proposed mitigation acquired by IoT gateway with internal AAA mechanism is presented in Table 1.

5.2 Performance

Authors compared latency when sending requests to IoT device with three variants: using a standard mechanism (without security), secured using https without authentication, authorization and accounting, and secured using both-sided https with AAA proxy. In addition, several sizes of responses were examined and presented on Figure 6 and Table 2. Obtained results for standard IoT web-interface requests(500kB) were: 201,7(+/-4,4)ms, 211,8(+/-5)ms, 222,6(+/-5,1)ms respectively for HTTP, HTTPS and HTTPS with client side authentication. In the second test, latency grows with response size. However, when requesting bigger files (f.e., images) percentage difference between HTTP and HTTPS/HTTPS with client authentication, drops from 11% and 22% for 233kB response to 1% and 3% for the 2MB response.

5.3 Energy consumption

In the last test, authors compare the energy consumption of IoT device when no cryptography is involved (sensing and hosting web-interface), with cryptography module is enabled (hosting web-interface with https protocol) and with implemented secure solution. The results are presented on Figure 7. Obtained results are 5,18mW, 5,2mW and 5,21mW respectively for HTTP, HTTPS and HTTPS with client side authentication. When converting this results to percentage differences HTTPS has power consumption higher by only 0,3% and proposed fully secure environment 0,5% more than insecure architecture.

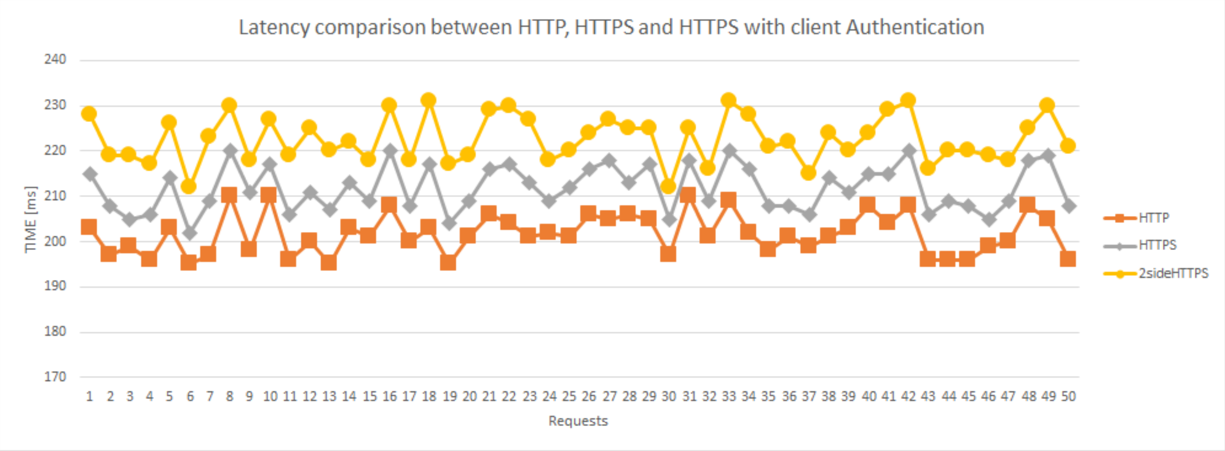

Figure 6 Latency comparison between HTTP, HTTPS and HTTPS with client authentication.

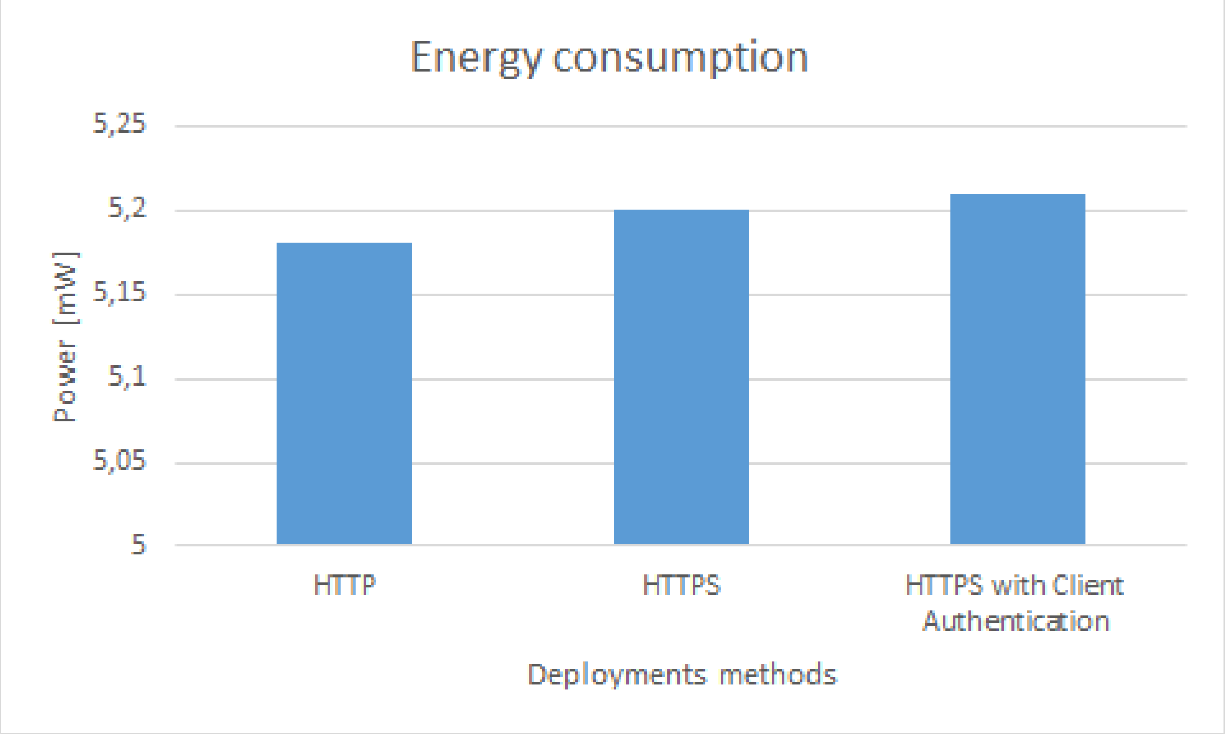

Figure 7 Energy consumption when different level of security is obtained.

Table 1 The OWASP IoT Top 10 with solutions

Vulnerability	**Solution with IoT Gateway with internal AAA mechanism**
Insecure Web Interface	Web Interface can be accessed after successful authentication
Insufficient Authentication/Authorisation	Full Authentication, Authorisation and Accounting
Insecure Network Services	ProxyACL blocking direct access to IoT device
Lack of Transport Encryption	Usage of both-sided HTTPS protocol
Privacy Concerns	By using encryption and AAA mechanism privacy should not be concern any more
Insecure Cloud Interface	No direct access to IoT device mitigates insecure cloud
Insecure Mobile Interface	No direct access to IoT device mitigates insecure mobiles
Insufficient Security Configurability	Cauterisation mechanism allow different roles to configure (or not) IoT device, also with security functionality
Insecure Software/Firmware	No direct access to IoT device and the accounting mechanism minimise the surface area of IoT device vulnerabilities
Poor Physical Security	In authors' opinion, in smart cities environment physical security is a deployment aspect and cannot be mitigated otherwise then good security practices

6 Conclusion and future work

In this paper, possible solutions to the lack of security, access control and privacy in IoT area were investigated. As argued, implementing direct IoT device access to the Internet does not address communication security or is impossible to implement due to CPU/memory restrictions. With the core idea of increased security of the IoT topology, a new utilisation of IoT Gateway with internal AAA mechanism was proposed.

By implementing proxied access between Internet users and IoT devices and then, validating proposed solution against performance, security and energy consumption it was demonstrated that with small overhead on performance and insignificant increase in energy consumption we obtain a secured and private topology that can be implemented in smart cities and other similar projects. The obtained results may vary between different proxy servers and IoT devices, but overall effect should be the same.

As for the future work, we are planning to utilise the PL-LAB2020 laboratory to validate solution against other IoT devices and analyse its security and performance using network traffic

Table 2 Latency comparison with several responses sizes.

Response size	**Response time for HTTP** [ms]	**Response time for HTTPS** [ms]	**Response time for HTTPS with client Auth** [ms]
233 kB	94	104,1	114,9
500 kB	201,7	211,8	222,6
2 MB	826,2	836,3	847,1

generators and analysers. Also, we want to deploy IoT Gateway in each of geographically dispersed PL-LAB2020 nodes.

Acknowledgemnts This work has been supported by the National Centre for Research and Development under the European Regional Development Fund, Grant No. POIG.02.03.01-00-104/13, PLLAB2020 project.

References

[1] C. G. Weissman. Survey: We asked executives about the Internet Of Things and their answers reveal that security remains a huge concern. `http://www.businessinsider.in/We-Asked-Executives-About-The-Internet-Of-Things-And-Their-Answers-Reveal-That-Security articleshow/45959921.cms`, Accessed: 20.10.2015.

[2] J. Steinberg. These devices may be spying on you (even in your own home). `http://www.forbes.com/sites/josephsteinberg/2014/01/27/these-devices-may-be-spying-on-you-even-in-your-own-home`, Accessed: 20.10.2015.

[3] A. Greenberg. Hackers remotely kill a jeep on the highway—with me in it. `http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/`, Accessed: 20.10.2015.

[4] A. Charlton. BMW ConnectedDrive hack sees 2.2 million cars exposed to remote unlocking. `http://www.ibtimes.co.uk/bmw-connecteddrive-hack-sees-2-2-million-cars-exposed-remote-unlocking-1486215`, Accessed: 20.10.2015.

[5] I. Foster, A. Prudhomme, K. Koscher, and S. Savage. Fast and vulnerable: A story of telematic failures. In *Proceedings of the 9th USENIX Conference on Offensive Technologies*, pages 15–15. USENIX Association, 2015.

[6] Horizon 2020. Work Programme 2016-17. European Commission Decision C (2015)6776 of 13 October 2015.

[7] SmartSantander Projetct. `http://www.smartsantander.eu/`, Accessed 20.10.2015.

[8] SmartSantander Infrastructure overview. Accessed 20.10.2015.

[9] PL-LAB2020 – infrastructure for research,. `http://www.pllab.pl/` [accessed 20.04.2017].

[10] A. Brachman. *RPL Objective Function Impact on LLNs Topology and Performance*, pages 340–351. Springer Berlin Heidelberg, 2013. DOI: 10.1007/978-3-642-40316-3_30.

[11] Y. Yu, L. J. Rittle, V. Bhandari, and J. B. LeBrun. Supporting concurrent applications in wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 139–152. ACM, 2006. 10.1145/1182807.1182822.

[12] A. Brachman. *Simulation Comparison of LEACH-Based Routing Protocols for Wireless Sensor Networks*, pages 105–113. Springer Berlin Heidelberg, 2013. 10.1007/978-3-642-38865-1_12.

[13] D. Samociuk. Secure communication between OpenFlow switches and controllers. In *Proceedings of the 4th International Conference on Advances in Future Internet*, pages 32–37. IARIA XPS Press, 2015.

[14] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi. The flooding time synchronization protocol. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 39–49. ACM, 2004. 10.1145/1031495.1031501.

[15] B. Chen, G. Peterson, G. Mainland, and M. Welsh. *LiveNet: Using Passive Monitoring to Reconstruct Sensor Network Dynamics*, pages 79–98. Springer Berlin Heidelberg, 2008. 10.1007/978-3-540-69170-9_6.

[16] J. A. Stankovic. Research directions for the Internet of Things. *IEEE Internet of Things Journal*, 1(1):3–9, 2014. 10.1109/JIOT.2014.2312291.

[17] M. Kozielski, M. Sikora, and Ł. Wróbel. DISESOR - decision support system for mining industry. In *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 67–74, 2015. 10.15439/2015F168.

[18] D. Evans. The Internet of Things how the next evolution of the internet is changing everything (april 2011). *White Paper by Cisco Internet Business Solutions Group (IBSG)*, 2012.

[19] T. R. Abdelzaher, S. Prabh, and R. Kiran. On real-time capacity limits of multihop wireless sensor networks. In *25th IEEE International Real-Time Systems Symposium*, pages 359–370, 2004. 10.1109/REAL.2004.37.

[20] T. He, J. A. Stankovic, T. F. Abdelzaher, and C. Lu. A spatiotemporal communication protocol for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 16(10):995–1006, 2005. 10.1109/TPDS.2005.116.

[21] S. Ravi, A. Raghunathan, and S. Chakradhar. Tamper resistance mechanisms for secure embedded systems. In *17th International Conference on VLSI Design. Proceedings.*, pages 605–611, 2004. 10.1109/ICVD.2004.1260985.

[22] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 46–57. ACM, 2005. 10.1145/1062689.1062697.

[23] A. Gruca, K. Harezlak, and P. Kasprowski. *Application of Dimensionality Reduction Methods for Eye Movement Data Classification*, pages 291–303. Springer International Publishing, 2016. 10.1007/978-3-319-23437-3_25.

[24] Z. W. Kong, B. Y. Ooi, and C. S. Wong. Storage Performance Evaluation for IoT Gateway Implementation Using Raspberry Pi 2. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 8(4):7–10, 2016.

[25] N. Saxena, A. Roy, B. J. R. Sahu, and H. Kim. Efficient IoT Gateway over 5G Wireless: A New Design with Prototype and Implementation Results. *IEEE Communications Magazine*, 55(2):97–105, 2017. 10.1109/MCOM.2017.1600437CM.

[26] T. Dierks and S. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2, 2008.

[27] C. Adams and S. Farrell. Internet X. 509 Public key Infrastructure. IETF RFC2510, 1999.

[28] OWASP Internet of Things (IoT) Project. `https://www.owasp.org/index.php/OWASP_Internet_of_Things_Top_Ten_Project`, Accessed: 20.10.2015.