Architecture for Privacy in Cloud of Things

Luis Pacheco, Eduardo Alchieri and Priscila Solis
Department of Computer Science, University of Brasília, Brazil

Keywords: Privacy, Security, Internet of Things, Cloud Computing, Cloud of Things.

Abstract: A large number of devices are connected to the internet through the Internet of Things (IoT) paradigm, resulting in a huge amount of produced data. Cloud computing is a computing paradigm currently adopted to process, store and provide access control to these data. This integration is called *Cloud of Things - CoT* and is useful in personal networks, like residential automation and health care, since it facilitates the access to the information. Although this integration brings benefits to the users, it introduces many security challenges since the information leaves the user control and is stored at the cloud providers. Particularly interesting, in order for these technologies to be adopted, it is important to provide protocols and mechanisms to preserve the users privacy when storing their data in the cloud. In this context, this paper proposes an architecture for privacy in Cloud of Things, which allows the users to fully control the access to the data generated by the devices of their IoT networks and stored in the cloud. The proposed architecture enables a fine grained control over data, since the privacy protocols and controls are executed at the IoT devices instead of at the network border by a gateway, which also could represent a single point of failure or a component that could impair the security properties of the system once it is compromised by a successful attack.

1 INTRODUCTION

The recent advances in the miniaturization of electronic components and in the wireless communications enabled the advent of the Internet of Things (IoT). In the IoT paradigm, the most diverse things (devices) of our everyday lives are connected to the Internet, bringing a lot of benefits to the population (Chui et al., 2010). It is foreseen that the connection of billions of things to the Internet that will provide the most diverse types of information to the users (Sundmaeker et al., 2010).

Wireless Sensor Networks (WSN) are composed by several small devices that have a processing unit, a sensor that enables the interaction with the physical world and an antenna for wireless communication (Akyildiz and Vuran, 2010). WSNs are seen, among other technologies, to enable the Internet of Things. Mostly WSN devices have constrained size and energy source, therefore the network stack must have a low processing cost. Several network protocols, covering all network stack layers, were proposed for WSN. The IEEE 802.15.4 Standard (Group, 2006) covers the Physical (PHY) and Medium Access Control (MAC) layers, it was released in 2006 and updated several times until now. This standard is currently used by most WSNs. However, protocols for the upper layers are very fragmented, with different proposes for different application areas, many times requiring translation in order to communicate with the Internet, which usually is achieved by the use of a gateway (Zhu et al., 2010; Nadeem et al., 2013).

However, the Internet of Things can be greatly benefited by the direct communication among WSNs or any IoT device with the Internet (without the use of a gateway), because the user can access the information coming from the devices directly (Granjal et al., 2015). Currently, there are several studies whose objective is to adapt Internet protocols to WSNs, such as 6LoWPAN (Kushalnagar et al., 2007), a standard that compresses and encapsulates IPv6 (Deering, 1998) headers to fit in IEEE 802.15.4 frames, and Constrained Application Protocol (CoAP) (Shelby et al., 2014), which is a message exchange protocol for constrained devices and networks and, since it is very similar to HTTP, it is very simple to translate messages from one standard to another.

The IoT limitations (e.g., storage and processing) can be mitigated by the use of Cloud Computing paradigm. These technologies complement each other, since IoT devices generate an huge amount of data and the cloud is able to provide resources to store and process it. This combination, called Cloud of Things (CoT), is currently being widely

Pacheco, L., Alchieri, E. and Solis, P.
Architecture for Privacy in Cloud of Things.
DOI: 10.5220/0006357504870494
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 2*, pages 487-494
ISBN: 978-989-758-248-6

studied (Aazam et al., 2014; Botta et al., 2016).

The Cloud Computing paradigm can complement the IoT paradigm in several ways, since the cloud is able to store, process and present the data generated by the IoT devices. Applications can utilize the virtually unbounded cloud resources to process and present information acquired from the IoT devices and stored in the cloud. Other cloud computing features are also beneficial for the IoT, such as the scalability inherent to the cloud and the increased security once the access control is performed in the cloud, which has appropriate computational power to execute this task.

Although this integration brings huge benefits, it is important to provide protocols and mechanisms to ensure the security properties and, particularly interesting in the case of personal IoT networks, to preserve the users privacy when storing their data in the cloud. In this context, this paper proposes an architecture for privacy in Cloud of Things, which allows the users to fully control the access to the data generated by the devices of their IoT networks and stored in the cloud. The proposed architecture enables a fine grained control over data, since privacy protocols and controls are executed at the IoT devices instead of at the network border by a gateway, which also could represent a single point of failure or a component that could impair the security properties of the system once it is compromised by a successful attack.

The remaining of this paper is organized as follows. Section 2 discusses the main related works. Section 3 details the proposed architecture for privacy in Cloud of Things, specifying the mechanisms and protocols necessary to allow the users access control over the data generated by their IoT devices. A detailed discussion about the proposed architecture is presented in Section 4. Finally, this work is concluded and future works are presented in Section 5.

2 RELATED WORK

The cloud computing service based approach can be extended to enable the integration with Internet of Things, in a way the features from the IoT can be offered as services in the cloud. As an example, a multitude of weather sensors can upload its data to the cloud, where a cloud service provides a easy and secure way of access to this data by the most varying user types. Several studies proposed the integration of IoT and Cloud Computing to provide services like this one in an infrastructure called Cloud of Things. The IoTCloud (Fox et al., 2012) is an open source platform that utilizes the cloud to facilitate the management of IoT devices, while Nimbits (Sautner, 2016) offers a solution where IoT data is collected and processed in the cloud. Since the Cloud of Things involves many new elements in the process, it also increases the complexity of the security solutions that must be adopted (Roman et al., 2013). In personal IoT networks, the storage of user data in the cloud brings a privacy problem since now the data is outside the user control sphere and therefore could be used for matters that the user does not approve. In order to preserve the user privacy, the data must by handled only by entities previously authorized by the user. In this sense, the SensorCloud project (Eggert et al., 2014) facilitates the implementation and utilization of privacy features in the Cloud of Things architecture. SensorCloud implements a cloud based platform which integrates sensor networks to the Internet. A layer based architecture is proposed where WSNs connect to the cloud through trusted points (gateways) that are responsible for both *(i)* communication with the cloud and *(ii)* security enforcement. The architecture does not specify the sensor network inner functioning, such as its protocol stack, it only defines the entities and protocols starting from the gateway.

The User-driven Privacy Enforcement for Cloud-based Services in the IoT (UPECSI) (Henze et al., 2016) extends the SensorCloud platform, it implements a privacy wise user-developer integrated solution. A Privacy Development Language (PDL) was conceived to facilitate the development of privacy wise cloud services, with the PDL the developer can provide detailed information about how the data is handled by its application. This feature is then used by the user to allow/deny the features of the application based on its privacy wishes. The security between IoT networks and the cloud is achieved by a multi-layered architecture where Privacy Enforcement Points (PEP), that are gateways at the border of the IoT, are responsible for security and privacy enforcement. This architecture enables the users to dynamically change the Privacy Policy (PP) according to their wishes, which enables a great level of control around their data stored on the cloud.

The Privacy Enforcement Point is the last entity on the user control sphere, since it is not a constrained device (as the IoT devices), it can apply robust security mechanisms, such as the use of Transport Layer Security (TLS) (Dierks, 2008) in the transport layer. According to the privacy policy set by the user, the PEP determines the data that can leave the user control sphere, and to which services the data can be sent. The data sent to the cloud is encrypted with a symmetric algorithm and the key is periodically changed, enabling the revocation of access to a given cloud ser-

vice when desired. Public Key Infrastructure (PKI) is used to encrypt the symmetric key for each authorized cloud service. Finally, the encrypted data is sent to the cloud platform and the encrypted keys are sent to their respective cloud services. This scheme enables the protection of the users data stored in the cloud, giving access only to previously authorized cloud services, not even the cloud platform can access the data.

That way that the cloud platform and the cloud services handle the user data are also privacy requirements. For example, the definition of the duration and the location where the data can be stored (Henze et al., 2013a). The UPECSI architecture uses Data Handling Annotations (Henze et al., 2013b) to enforce such requirements, these annotations are sent along with the data to the cloud platform indicating how the data is supposed to be handled. The previously mentioned PDL also generates data to enable auditing and monitoring of the cloud services by the cloud platform.

3 ARCHITECTURE FOR PRIVACY IN CLOUD OF THINGS

This work extends the UPECSI architecture (Henze et al., 2016) by specifying the mechanisms and protocols necessary to enable the direct communication of IoT devices with the cloud. By enabling the IoT devices with a network stack capable of direct communication with the Internet, it is possible to transfer the security and privacy enforcement responsibilities from the Privacy Enforcement Points (gateways) to IoT devices. Therefore this work also proposes the PEP removal, although a gateway could still be used for protocol translation or other tasks that do not require privileges over the data.

The proposed approach brings at least the following advantages: *(1)* it improves the fault-tolerance of the system since it removes the gateway, which is a single point of failure, from the architecture; *(2)* it improves the security of the system since it removes a component that is responsible for all security tasks and, consequently, could impair the security properties of the system once it is compromised by a successful attack; and *(3)* by executing the security and privacy enforcement at the IoT devices, it is possible to implement a fine grained control over the data since each device could adopt a different security policy.

The UPECSI architecture covers a wide range of aspects of the Cloud of Things, from the development of a cloud service to the communication protocols between the border of the IoT network and the cloud platform. Since this work focus on the IoT devices communication schemes, what does not involve this area is not mentioned in the remaining of this section, such as the PDL mentioned in Section 2. A given user can have several IoT networks under his or hers control. The user binds the IoT devices to the cloud platform, consequently the data generated by them is stored in the cloud. The authorized cloud services can then process the stored data and present it to the user. A Trusted Third Party (TTP) audits and monitors the cloud services, provides standard privacy policies (intended to naive users) and executes some other security tasks discussed later on this section.

The IoT devices store data they generate in the cloud according to the Privacy Policy. To respect this policy, it is possible that some data *(1)* can not be sent to the cloud, *(2)* can be sent encrypted, or even *(3)* can be sent without any encryption processing. Afterward, privacy requirements are held by the cloud services since they have the keys necessary to access only the data they are allowed to access. The remaining of this section describes the communication schemes defined in the proposed architecture.

3.1 Binding of IoT Devices to the Cloud Platform

Users must first register an account in order to use the cloud service. After this, users can then bind their IoT devices to store the generated data. The binding process is achieved by using the OAuth 2.0 protocol (Hardt, 2012), which is an open source and secure authorization protocol. After the binding process, the IoT devices can upload its data without the possessing of the users credentials.

The OAuth 2.0 protocol requires a secure communication in order to provide real secure authorization. In the Internet this is achieved by the use of the TLS protocol, which is used together with TCP. Since TCP cannot be used in IoT networks another solution must be employed. This area is currently under active research and there are already many proposals for security in IoT transport layer. Particularly interesting, an adaptation of the Datagram Transport Layer Security (DTLS) (McGrew and Rescorla, 2010) is being studied by the DTLS In Constrained Environments (DICE) working group (Tschofenig and Fossati, 2016). This work does not define a protocol for secure communication, but requires the usage of one.

The user can bind one device at time or multiple devices at once. The process of binding multiple devices at once is only achieved when using a gateway, in this way all IoT devices will share the same identification, and thus the user will not be able to

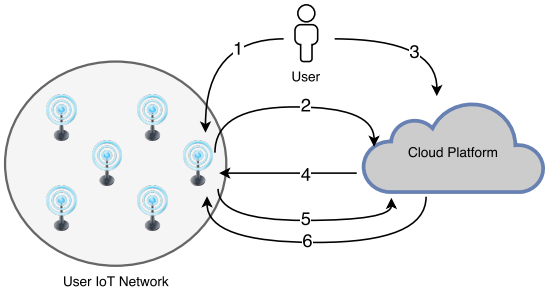

Figure 1: Binding Scheme.

uniquely configure each device. OAuth's Authorization Code Grant permission is used, this permission type requires user intervention only at the setup phase. After authorized, devices can upload data to the cloud without caring the user credentials.

Figure 1 presents the binding process for one device at time, that is executed as following: *1* the user starts the process as the Resource Owner by entering device's binding web page; *2* the device requests an Authorization Code for the cloud platform; *3* in order to receive the Authorization Code, the user is redirected to a login page; *4* after successfully receiving the user credentials, the cloud platform sends the Authorization Code to the device; *5* in possession of the Authorization Code, the device requests an Access Token to the cloud; *6* the cloud platform delivers the Access Token.

At the end of the binding process the device has an Access Token that must be attached in every data sent to the cloud, and the cloud platform has the device's identification. The process is the same when binding multiple devices at once, the only difference is that at the end the gateway sends the received Access Token to all devices, that is why, by using this method, the user cannot identify which device generated a specific data. After binding all devices, the user could create logical networks at the cloud platform in order to easily manage them.

3.2 Privacy Policy Enforcement

A Privacy Policy (PP) defines whether some data can be stored in the cloud, the way it is stored (encrypted or not) and what is done with the user data by a cloud service. Different from common Privacy Policies, where the user can only accept or deny all of it, in our architecture the user can change it over the time, in this way the user can assume real control over the data it sends to the cloud. A cloud service provides an interface for users to see what data is used and for what purpose, the users can enable or disable specific features in order to restrict the access to their data. As example, users could enable or disable their location monitoring by a given cloud service, according to their privacy desires.

The Privacy Policy is enforced every time an IoT device sends data to the cloud. The enforcement is done by the IoT device itself. The update process of a Privacy Policy is as follows: *1* the user updates the PP trough the cloud service interface; *2* the cloud platform send the updated PP to the Trusted Third Party (TTP), which audits the information and generates a Privacy Configuration (CP); *3* the TTP sends the new CP to all devices that provide data to the cloud service and needs to update their policies.

3.3 Data Access Control

After binding devices to a cloud platform and setting specific Privacy Policies for the authorized services, it is necessary to ensure that IoT data is stored and accessed only by authorized entities.

Data access control is achieved by a cryptography scheme. As in the binding process, the communication between IoT devices and cloud must be secure, which is achieved by a proper transport layer protocol (this subject is further discussed in Section 4). The IoT sensitive data is encrypted before being stored in the cloud (no-sensitive data could be stored without encryption), preventing access even by the cloud platform. The keys necessary to its decryption are stored in a key depot also in the cloud platform.

Before sending data to the cloud, an IoT device must filter it according to the Privacy Configuration, deciding if data must leave the user control sphere. The data then is encrypted with a symmetric algorithm, the key used by the algorithm is, encrypted with the public key of the authorized cloud service. In case there are more them one cloud service authorized to access the same data, the symmetric key must be encrypted once for each service. The data is stored only once in the cloud platform.

The symmetric key is updated regularly, preventing new services to access data stored previously to its authorization or revoked services to continue to access data. Past keys remain in the cloud, enabling access to data previously stored when desired.

Cryptography operations can significantly decrease the IoT network lifetime. To mitigate this scenario we propose two different approaches to ensure data access control: in the first one the IoT devices manage cryptography keys (Section 3.3.1) and in the second one these keys are managed by a Trusted Third Party (Section 3.3.2). In the following these approaches are presented and a discussion about the costs and benefits of each of them are further pre-

sented in Section 4.

3.3.1 Keys Management by IoT Devices

Figure 2 presents the approach where the device itself is responsible for generating symmetric keys, encrypt them with the cloud services public keys and send them to the key depot in the cloud platform.

This approach is intended for IoT devices that have enough computational and power capabilities to execute the following tasks (Figure 2) without significantly decrease their lifetimes: *1* receive public keys from cloud platform; *2* periodically generate symmetric keys; *3* encrypt each new symmetric key once for each authorized cloud service; *4* send the encrypted symmetric key to the key depot; *5* send encrypted data to the cloud platform.

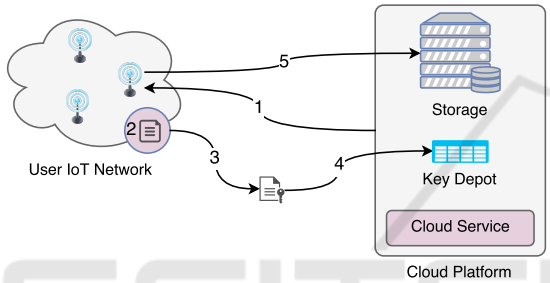

Figure 2: IoT Device key Management.

3.3.2 Keys Management by a Trusted Third Party

Figure 3 presents the second approach, where a Trusted Third Party (TTP) is responsible for some tasks related to data access control, relieving the constrained devices from some costly tasks.

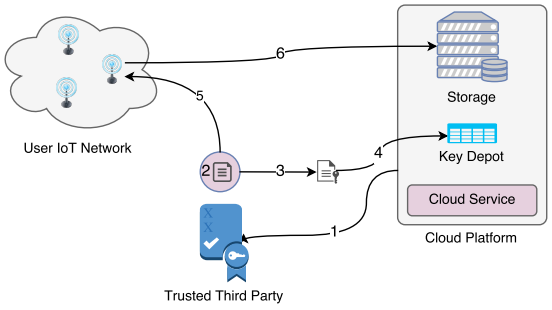

Figure 3: TTP key Management.

This approach is useful for devices with constrained capabilities, where the tasks shifted to the TTP would significantly reduce the communication delay and increase their lifetimes. This approach has the following steps (Figure 3): *1* **TTP**: Receive services public keys from cloud platform; *2* **TTP**: Periodically generate symmetric keys; *3* **TTP**: Encrypt each new symmetric key once for each authorized cloud service; *4* **TTP**: Send the encrypted symmetric key to the key depot; *5* **TTP/IoT Device**: Securely send the symmetric keys to the IoT devices; *6* **IoT Device**: Send encrypted data to the cloud platform.

3.4 Flexible Privacy Policies

The data control scheme provided by the proposed architecture enables only previously authorized entities to handle the user data in the cloud. However, in some exceptional situations, it is convenient to decrease privacy requirements in order to allow new entities access to data. As an example, when a health monitoring IoT network detects an emergency, it could be more beneficial to the user if any medical personal could have access to his data, increasing the chances of getting assistance.

In order to attend the scenario above described, this work proposes the use of Flexible Privacy Policies (FPP), which are secondary PPs only enabled when a given event is detected. The user must previously set the thresholds that will trigger the FPP, which can be based on IoT network generated data or even in external events. FPPs are created following the same scheme as common PPs (Section 3.2).

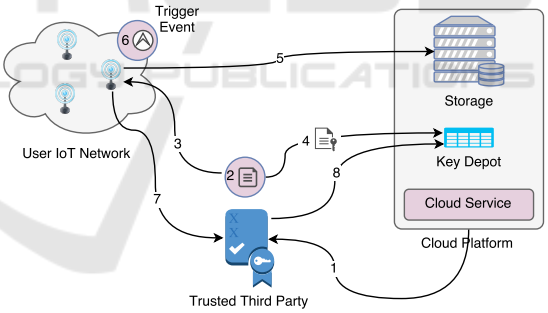

Figure 4: Flexible Privacy Policy Activation.

Figure 4 shows how FPPs are activated. The scheme is similar to regular access control for common data, but in this case the public keys used to encrypt symmetric keys are generated by the TTP and cloud services can access user data only after the FPP activation. The FPP processing has the following steps: *1* TTP receives services public keys from cloud; *2* TTP generates symmetric keys (used by IoT devices to encrypt their generated data) and public/private pair keys (used by the TTP itself to manage the data access), as follows: Symmetric keys, Public/private key pair for each cloud service that will be grant access; *3* TTP securely sends (using a transport layer security protocol as discussed at Section 3.1)

the symmetric keys to IoT devices; *4* TTP sends the symmetric keys encrypted with the public key (generated at previous step 2) to the key depot; *5* IoT device sends encrypted data to the cloud; *6* IoT device detects an event that activates the FPP; *7* IoT device signals the TTP; *8* TTP sends, through the key depot, the private key to the cloud service that must gain access to data, this private key is encrypted with the service public key (received at step 1).

In order to prevent access to data generated after FPPs are deactivated, the public/private key pair must be recreated and new symmetric keys must be encrypted with new public key. The FPP scheme heavily relies in TTP, since IoT devices have a intrinsic unreliable nature.

4 DISCUSSIONS

This section presents a discussion about aspects related to the proposed architecture. Firstly, the direct communication among IoT devices and the Internet is depicted, mainly the end-to-end security related to it is discussed. Afterward, a discussion regarding the schemes developed by this study is presented, indicating its limitations, advantages and disadvantages.

4.1 Direct Internet Communication in IoT Devices

An important aspect of the proposed architecture is the utilization of equivalent Internet protocols in IoT devices, providing the ability of direct communication between devices and cloud. Based on UPECSI (Henze et al., 2016), this work transfer the Privacy Enforcement Point features to the IoT devices, adapting those features according to IoT devices constrained nature. Although devices communicate directly with the cloud, a gateway could still be used for translating protocols. For example, the IEEE 802.15.4 packet size is 127 bytes but many Internet protocols have headers and messages that would use much of this size, leaving very little room for application data (e.g., IPv6 header has a minimum of 40 bytes).

UPECSI defines communication schemes only between the gateway and the cloud, this work proposes a secure architecture between IoT device and cloud, regarding only the application layer. However, here we discuss security in transport layer, since it is a topic currently being widely studied. Security in Link and Network layers in IoT, although still with much room for improvement, can be considered more mature, since the IEEE 802.15.4, 6LoWPAN and RPL (Winter, 2012) are already consolidated standards.

The overhead introduced in the processing and message exchange caused by security protocols, specially in cryptography operations, is the main barrier for its implementation in Internet of Things. Internet security protocols use Public Key Infrastructure (Salomaa, 2013), that uses two keys for each entity. Creating and exchanging these keys are expensive, thus many IoT security protocols admits they are preallocated in devices beforehand. This work assumes that communications take place in a secure channel. Although there are already many proposals (Sethi et al., 2012; Hummen et al., 2014; Tschofenig and Fossati, 2016), the IETF is still working to define a standard for constrained device networks.

4.2 Proposed Architecture

This section discusses the schemes proposed in this work, first a discussion about the binding protocol is presented, indicating its benefits and possible difficulties of its implementation in IoT devices. Afterward, we discuss how the Privacy Policy is enforced by IoT devices, taking account of its constrained characteristics. The data access control scheme is also debated since it is the architecture protocol that most adds load to devices. Finally, the Flexible Privacy Policy scheme is discussed.

4.2.1 Binding Scheme

The load that the binding scheme adds to the network can be neglected, since it is performed only once by each network device. As stated, this scheme could also be executed by a gateway, as in UPECSI, but in this case all devices would share the same Access Token. This feature speeds up the binding procedure, but decreases the data access control granularity, since all devices will be bound to the same cloud services, i.e., these services will have access to all data from the devices.

The OAuth 2.0 assumes a secure channel is used for communication, since it does not provide reliability for exchanged messages. In order to provide device specific binding, the user must access an interface provided by each device (step 1 at Figure 1). This kind of service should have an optimized implementation, in order to prevent too much load to the constrained devices.

4.2.2 Privacy Policy Enforcement

The Privacy Policy Enforcement is performed as in the UPECSI architecture, but instead of being executed by the PEP (gateway) it is executed by each IoT devices. Updating a Privacy Policy is achieved

through the respective cloud service, usually done by a web interface or smartphone application. This scheme only introduces the reception and enforcement of Privacy Policies by the devices. The reception is unlikely to occur very often and the enforcement is implemented by simple conditional statements. Consequently, these steps does not significantly increase the devices load.

4.2.3 Data Access Control

User privacy requirements are guaranteed by using cryptography. Private data is stored encrypted in cloud platforms, and only authorized cloud services have the key to decrypt it. This method provides data access only to allowed entities. As already discussed, cryptography operations can be expensive for IoT devices. Although IoT devices must encrypt data going to the cloud, some operations could be relieved by shifting them to the Trusted Third Party. In this approach the key management is performed by the TTP, leaving the IoT devices only with the tasks of receiving keys and encrypting data. This scheme is recommended for IoT networks where devices have constrained resources.

IoT devices can be responsible for managing symmetric keys, excluding the TTP from the process. This method is more expensive to devices, which must execute tasks with high computational costs: symmetric key creation, asymmetric encryption of symmetric keys and sending encrypted keys to services. The encryption of symmetric keys is periodic and performed once for each service authorized to access the respective data, which means that these operations can be quite expensive if there are many services. This method must be used with cautious, if the number of services and the periodicity of symmetric key generation is high it can potentially compromise the IoT network lifetime and responsiveness. The advantage of using this method is that the TTP will not have the keys necessary to access data, therefore decreasing the surface of attack for the network.

In both schemes data will be encrypted twice, once in the transport layer, by a protocol that ensures a secure channel, and once in the application layer, by the architecture proposed in this work. The encryption performed at the application layer is always necessary, since that is how data is stored in the cloud platform. Consequently, to prevent data from being encrypted twice, it is necessary to remove the encryption at the transport layer. The messages sent by the devices contain the application encrypted data and the Access Token, which now must be protected at the application layer. This can be achieved simply by securely sharing a key between cloud platform and IoT device before it needs to send data. This process could be done at the end of the binding process, which already uses a secure channel. In this way, IoT devices can encrypt the Access Token with this shared key and send both data protected, the Access token and the devices generated data.

4.2.4 Flexible Privacy Policy

Flexible Privacy Policies were proposed to increase the architectures adaptability. This scheme enables the user to define boundaries, where some cloud services can begin or stop to have access to private data. This is specially useful in exceptional situations, where it is more beneficial to give up privacy then keep it.

Since the main purpose of FPPs are exceptional situations, it may be the case where the IoT device itself ceases functioning, in this case it is only possible to activate the FPP by an external agent. To ensure the proper behavior in this situation the TTP is responsible for enforcing the FPP, managing keys and granting access to services when necessary. This method adds much more reliability to the scheme.

The load introduced by this feature is only related to triggering the FPP by signaling the TTP when a FPP must be activated, thus requiring very lower resources for a very useful feature.

5 CONCLUSIONS

The Internet of Things and Cloud Computing integration certainly brings many benefits for applications, since while the first generates a huge amount of data, the second has the power to store, process and present it, including solutions for big data. Security is an extremely relevant aspect in this integration. In personal networks the user privacy must be specially dealt with, since private data leaves the user control sphere when stored in the cloud.

In order to prevent or at least mitigate this problem, this work proposed an architecture where users have more control over access to their data when uploaded to the cloud. This architecture enables a fine grained control over data, since the privacy protocols and controls are executed at the IoT device, not at the network border by a gateway.

The proposed approach introduces more load at devices, which could impact the network performance and lifetime in scenarios where the devices have limited energy source such as WSNs. However, many IoT devices have unlimited energy and, consequently, this is not a constraint. As future work we intend to

define scenarios and evaluate the performance of the proposed solutions.

ACKNOWLEDGEMENTS

Luis Pacheco is supported by CAPES (Brazil). This work is partially supported by CNPq (Brazil) and RNP/CTIC (Brazil) through projects FREESTORE (457272/2014-7) and EUBrasilCloudForum, respectively.

REFERENCES

Aazam, M., Khan, I., Alsaffar, A. A., and Huh, E.-N. (2014). Cloud of things: Integrating internet of things and cloud computing and the issues involved. In *Proceedings of 11th International Bhurban Conference on Applied Sciences & Technology*, pages 414–419.

Akyildiz, I. F. and Vuran, M. C. (2010). *Wireless sensor networks*, volume 4. John Wiley & Sons.

Botta, A., de Donato, W., Persico, V., and Pescapé, A. (2016). Integration of cloud computing and internet of things: a survey. *Future Generation Computer Systems*, 56:684–700.

Chui, M., Löffler, M., and Roberts, R. (2010). The internet of things. *McKinsey Quarterly*, 2(2010):1–9.

Deering, S. E. (1998). Internet protocol, version 6 (ipv6) specification. RFC 2460.

Dierks, T. (2008). The transport layer security (tls) protocol version 1.2. RFC 5246.

Eggert, M., Häußling, R., Henze, M., Hermerschmidt, L., Hummen, R., Kerpen, D., Pérez, A. N., Rumpe, B., Thißen, D., and Wehrle, K. (2014). Sensorcloud: Towards the interdisciplinary development of a trustworthy platform for globally interconnected sensors and actuators. In *Trusted Cloud Computing*.

Fox, G. C., Kamburugamuve, S., and Hartman, R. D. (2012). Architecture and measured characteristics of a cloud based internet of things. In *International Conference on Collaboration Technologies and Systems*.

Granjal, J., Monteiro, E., and Silva, J. S. (2015). Security in the integration of low-power wireless sensor networks with the internet: A survey. *Ad Hoc Networks*, 24:264–287.

Group, W. W. (2006). *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*. IEEE Standard for Information Technology.

Hardt, D. (2012). The oauth 2.0 authorization framework. RFC 6749.

Henze, M., Großfengels, M., Koprowski, M., and Wehrle, K. (2013a). Towards data handling requirements-aware cloud computing. In *IEEE International Conference on Cloud Computing Technology and Science*.

Henze, M., Hermerschmidt, L., Kerpen, D., Häußling, R., Rumpe, B., and Wehrle, K. (2016). A comprehensive approach to privacy in the cloud-based internet of things. *Future Generation Computer Systems*, 56:701–718.

Henze, M., Hummen, R., and Wehrle, K. (2013b). The cloud needs cross-layer data handling annotations. In *IEEE Security and Privacy Workshops (SPW)*.

Hummen, R., Shafagh, H., Raza, S., Voig, T., and Wehrle, K. (2014). Delegation-based authentication and authorization for the ip-based internet of things. In *Annual IEEE International Conference on Sensing, Communication, and Networking*.

Kushalnagar, N., Montenegro, G., and Schumacher, C. (2007). Ipv6 over low-power wireless personal area networks (6lowpans): overview, assumptions, problem statement, and goals. Technical report.

McGrew, D. and Rescorla, E. (2010). Datagram transport layer security (dtls) extension to establish keys for secure real-time transport protocol (srtp).

Nadeem, Q., Rasheed, M. B., Javaid, N., Khan, Z., Maqsood, Y., and Din, A. (2013). M-gear: gateway-based energy-aware multi-hop routing protocol for wsns. In *Broadband and Wireless Computing, Communication and Applications (BWCCA), 2013 Eighth International Conference on*, pages 164–169. IEEE.

Roman, R., Zhou, J., and Lopez, J. (2013). On the features and challenges of security and privacy in distributed internet of things. *Computer Networks*, 57(10):2266–2279.

Salomaa, A. (2013). *Public-key cryptography*. Springer Science & Business Media.

Sautner, B. (2016). Nimbits. `http://bsautner.github.io/com.nimbits/`. Accessed at 2017-01-19.

Sethi, M., Arkko, J., and Keränen, A. (2012). End-to-end security for sleepy smart object networks. In *IEEE Conference on Local Computer Networks Workshops*.

Shelby, Z., Hartke, K., and Bormann, C. (2014). The constrained application protocol (coap). RFC 7275.

Sundmaeker, H., Guillemin, P., Friess, P., and Woelfflé, S. (2010). Vision and challenges for realising the internet of things. *Cluster of European Research Projects on the Internet of Things, European Commision*.

Tschofenig, H. and Fossati, T. (2016). Transport layer security (tls)/datagram transport layer security (dtls) profiles for the internet of things. RFC 7925.

Winter, T. (2012). Rpl: Ipv6 routing protocol for low-power and lossy networks. RFC 6550.

Zhu, Q., Wang, R., Chen, Q., Liu, Y., and Qin, W. (2010). Iot gateway: Bridgingwireless sensor networks into internet of things. In *IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing*.