Privacy Preserving Architecture for IoT Cloud Services

Ayush Kumar
Department of Electrical and Computer Engineering
National University of Singapore
ayush.kumar@u.nus.edu

Nguyen Quoc Binh
School of Computing
National University of Singapore
nqbinh@comp.nus.edu.sg

ABSTRACT

With the growing number of network-connected devices owned by users, the Internet of Things (IoT) is set to become a indispensable part of our lives. To manage such a large number of devices, a central management solution is required. Cloud based solutions are a popular candidate but they give rise to many privacy concerns for the user. In this project, we study the cloud based management of IoT devices where the cloud service is not trustworthy and identify related privacy concerns. Further, we design an alternative Cloud Based Service (CBS) that addresses those privacy concerns. The proposed CBS uses three approaches to achieve privacy, namely distribution of trust over multiple parties, policy based solutions and technical solutions (for e.g. cryptography). Finally, we derive important insights from our proposed model and discuss them in detail.

Keywords

Internet of Things, Cloud Computing, IFTTT, Privacy, Security.

1. INTRODUCTION

The Internet of things (IoT) refers to the network of sensing devices capable of connectivity, i.e., using technologies such as RFID, Zigbee, WiFi, Bluetooth, 3G/4G etc., these devices are able to send and receive data to other devices. It has been predicted that we will have almost 50 billion of IoT devices by 2020 [1]. A large number of these IoT devices will form a part of our homes and each person will possess many smart networking capable devices such as lighting systems, fitness trackers, wearables, washing machines etc.

As the number of such devices grows, it becomes tenuous to manage them and hence we need some sort of central management solutions to control these devices. With the advent of cloud computing and its associated advantages (high computing power and performance, low upfront infrastructure costs, scalability and availability), cloud based management of IoT devices is emerging as a strong candidate. Launched in 2013 by Linden Tibbets, Jesse Tane, Scott Tong and Alexander Tibbets in San Francisco, IFTTT is the first popular cloud service for IoT devices across vendors, according to our knowledge. It is a free web-based service that allows users to create simple conditional statements to realize useful tasks in *If 'This' Then 'That*' form, which are called *recipes*.

To have an idea about the privacy aspects of such a cloud service, we looked at IFTTT's privacy statement [2]. It mentions that the service automatically collects information such as the number and frequency of visitors to the site, the user's IP address, the URL that one just came from, the URL one goes to subsequently, and the user's computer browser information. Besides, the service stores other information that users provide, including contacts and camera-photos on their mobile devices. It also collects users' location and calendar information. With so much personal information at its disposal including that collected without users' explicit consent, IFTTT is a gold mine of private user data. If a hacker gains access to IFTTT database, he would have access to all this data at once place. Further, IFTTT allows permitted employees to access users' personal information. It's likely that a rogue employee may store that information and sell it to third parties. Not only this, the service accepts that there is a possibility that adversaries may intercept the private communications between IFTTT and users. IFTTT may also share users' personal data with third party service providers, partners or other businesses in case of asset sellout. These entities use the personal data subject to their own privacy regulations which may be weaker. The users would have no control over their privacy in such a case since they would have signed the privacy agreement with IFTTT only. The only thing IFTTT mentions about technical security is that it uses SSL (Secure Socket Layer) to encrypt and secure users' personal information during transmission. This would prove to be insufficient if IFTTT itself decides to share private data with third parties or if IFTTT is compromised by an attacker. There are no other details available about the security policies and architecture implemented by IFTTT.

In the rest of this report, we go through past works on privacy concerns in IoT and IoT cloud services. We then provide the motivation for our work and go on to define the problem statement for our project. Subsequently, we propose a privacy preserving architecture for IoT cloud services. We derive some important insights from our proposed model. Finally, we identify few challenges to be addressed in our future work.

2. RELATED WORKS

The concept of privacy in the context of IoT needs to be defined first. Ziegeldorf et. al [3] propose a privacy definition for the Internet of Things, which highlights the idea of informational self-determination by enabling the subject to access his personal privacy risk, to take appropriate action to protect his privacy, and to be ensured that it is enforced beyond his immediate control sphere.

Privacy related concerns for IoT users have been studied in detail in existing literature. Porambage et. al [4] show that using eavesdropping, attackers can analyze the timings of data transmissions and conclude when the residents are at home or away. Further, using wireless signal based localization, an adversary can know about the internal arrangement of the home. Besides, the fact that users depend too much on a particular IoT CSP (Cloud Service Provider) can lead to user lock-in scenarios. Abomhara et. al [5] also mention about attacks on privacy, such as eavesdropping and passive monitoring, traffic analysis, and data mining. They point out the challenges of security and privacy in the IoT, including user privacy and data protection, authentication and identity management, trust management and policy integration, authorization and access control, end-to-end security, and coming up with attack resistant security solution. Roman et. al [6] focus on the distributed IoT and show that the IoT design principles should provide user-centric support for security and

privacy that guarantees user anonymity and surrounding information and transparent policies to prevent silent control of IoT devices. They also show that there can still be another privacy threat arising from the existence of entities that profile and track users without their consent.

The following works have tried to identify properties of a privacy protecting system. Porambage et. al [4] give a list of privacy framework characteristics. It includes openness, transparency and a specified purpose, identity privacy, temporal and location privacy, query privacy, access control, interoperability, data minimization, accountability, and security. They also give seven fundamental principles for the Privacy by Design (PbD) approach which is a security requirement engineering methodology that considers privacy requirements as organizational goals in business and identification processes. In their paper, Abomhara et. al [5] present a list of security and privacy framework requirements in dealing with IoT, including lightweight and symmetric solutions, lightweight key management systems, cryptographic techniques, techniques to support PbD concepts, keeping information as local as possible, and prevention of location privacy and personal information inference.

In this project, our focus is on cloud solutions offered to users for managing their IoT devices. One such popular cloud service as mentioned in the previous section is IFTTT (If-This-Then-That). In [7], the authors present the high level overview of IFTTT. They also provide a dataset of publicly shared recipes on IFTTT and then proceed to characterize and analyze the dataset. Reference architectures for IFTTT and similar TASs (Task Automation Services) have been presented and discussed in detail in [8] and [9]. Our work also takes inspiration from [10], which presents AKI (Accountable Key Infrastructure), a novel public-key validation infrastructure. AKI reduces trust in a single entity (Certificate Authority) by distributing trust over multiple entities. Finally, we use the concept of pseudonyms which hide the real identity of users. Pseudonyms are quite popular in Bitcoin networks [11]. Various papers in literature have talked about implementations of pseudonyms [12-14].

3. MOTIVATION

We assume the model shown in Fig.1. A group of IoT devices is connected to an IoT gateway which is in turn connected to the cloud management service. The IoT devices gather data and send to the IoT gateway. The gateway accumulates this data and processes it before forwarding to the cloud. The cloud service further processes the data and presents it to user in desired form or takes some action on basis of the data. For instance, IFTTT has more than 224,950 recipes created by 100,000 different users as of September 2015. Each user is associated with a user profile which stores the user email address, IP address, location information and other private data mentioned in Section 1. Recipes are basically configurations that combine triggers and actions (e.g. "If *there is motion detected by camera*, Then *turn on the Philips Hue lights*"). These recipes store a lot of private information such as the users who created or added them, the IoT devices used by those users and user actions. In addition, IFTTT obtains permissions to read data from the IoT devices on behalf of the users. In case IFTTT is compromised by an adversary with sufficient computational capabilities, it is reasonable to claim that there will be a huge leakage of private data belonging to the users. Further, since IFTTT currently provides free service to users and doesn't guarantee privacy, it may share user data with third parties. This data divulges a whole lot of information about user behavior and the third parties may use it for various purposes including targeted advertising.

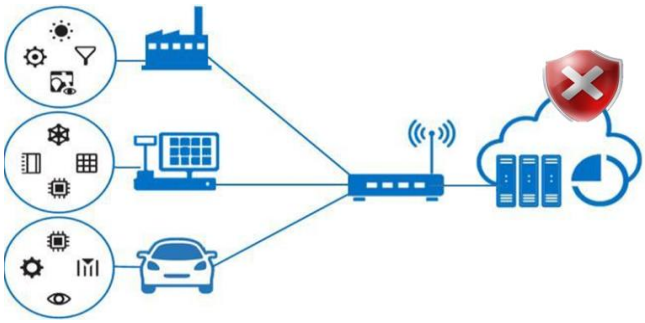

Fig.1. Model of a cloud-managed IoT system with untrusted CSP

(Source: Intel)

4. PROBLEM DEFINITION

In this section, we define the privacy related problem considered in our project. Before that, we present the high level working of a cloud based solution (CBS) to manage IoT devices. In the *first* step, users create profiles on the CBS platform. Next, the users connect to apps (e.g. Fitbit, Nest etc.) by logging in and giving permission to the CBS to collect data from corresponding IoT devices on their behalf. Then the users create rules by combining above apps using trigger-action statements. They may also add rules created by other users to their profiles. Subsequently, the users may choose to activate the rules selectively or leave them deactivated. Only the active rules are executed. Finally, the CBS acts on the rules created by users and the data read from IoT devices by sending appropriate actions to the actuating devices. The goal of our project is to design a CBS possessing the following desired privacy related properties:

- It shouldn't be able to link any rule to the user who has created/added that rule. The full list of rules is assumed to the public though.
- It shouldn't be able to link the data received from IoT devices to the corresponding users. In spite of this restriction, the rules should work without any issues.

5. PROPOSED SOLUTION

5.1 Overview

To satisfy the desired privacy related properties mentioned in previous section, we propose a privacy preserving CBS architecture. In the IFTTT model, user profiles, recipes, channels, recipes and IoT device data are all stored in a single database. As mentioned in Section 1, if a hacker compromises IFTTT database, he would gain access to all this information at a single place. We therefore propose to distribute trust over multiple servers, each of which would store a separate piece of information. These servers have been named according to the information that they hold. The Profile Server (PS), which directly communicates with the end user machines, stores and processes users' profiles. Another server called the Rule Server (RS) stores and processes apps and the rules created or added by users. Lastly, the Device Server (DS) communicates with the API services. API services are provided by the IoT device manufacturers to read and write data from the devices. The complete architecture of the proposed CBS is shown in Fig.2.

We assume that these servers are managed by independent organizations that are unlikely to collude under normal circumstances. This policy based solution for privacy can be implemented, for e.g., by allowing the DS to be managed by a service like IFTTT, the RS be operated by large internet organizations such as Google and the PS be controlled by some government organization since it stores sensitive user profile information.

We also introduce some technical solutions for privacy. Inter-server communication is assumed to be encrypted and signed. Rules are retrieved by associated users through public-key cryptographic mechanism. Besides, pseudonyms (PNs) used to hide real user identities, are updated periodically.

5.2 Communication flows

In the setup phase, the user sends a profile creation request to PS. This may happen when the user visits a web page rendered by the PS, fills up the profile information required and presses the link for creating a profile. The PS then creates a PN corresponding to this user and sends the PN and public key of this user (Kpub) to RS. The PN is required here because we want the real user identity to be known only to the PS. The remaining servers use PNs to link their information to any particular user. The PN corresponding to a user is assumed to be unique and the PN mapping function is assumed to be a one-way function so that an adversary can't retrieve user identity from PN. The RS sends only the PN to DS. In the next step, the user can send an app connection request to RS through PS. The RS then forwards this request to the DS which in turn forwards it to the API Service. The API service sends a login request to the user. In effect, the user is redirected to the login page of the IoT device manufacturer. After the user logs in correctly, the API Service provides an access token to DS. The access token needs to be refreshed periodically. Finally, the user can create rules and chose to activate them or leave them deactivated. This is done by sending rule creation and activation requests to the RS through PS.

Post setup phase, the system enters a loop phase which runs indefinitely for every T seconds unless terminated (value of T can be selected). The DS retrieves rules corresponding to a user from RS using the PN. Subsequently, the DS makes an API call to the API service and receives trigger IoT device data. The DS then evaluates the rules retrieved in previous step based on this data. If the trigger condition is satisfied, the DS makes another API call to send the corresponding action.

When the user wants to retrieve his profile information and the rules he has created, he sends a profile retrieval request to PS while a rule retrieval request is forwarded to RS by the PS which inserts PN and Kpub in the request. The RS encrypts the rule using Kpub and sends it to PS which in turn forwards it to the user. The user decrypts the rule using his private key. The main purpose behind encrypting the rule is that we don't want the PS to know about rules.

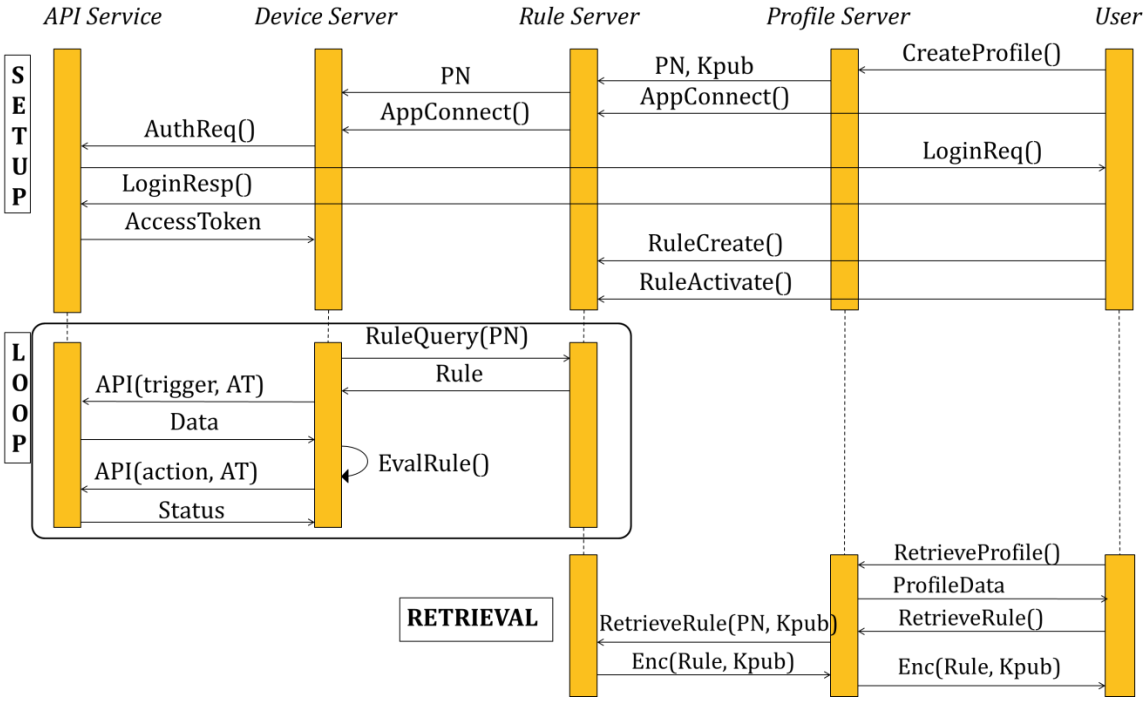

Fig.2. Architecture and communication flows for the proposed CBS

6. KEY INSIGHTS

In this section, we present some of the important insights we can derive from our proposed solution. Firstly, our proposed CBS can work with complex rules for e.g. If (cond1 AND cond2 OR cond3) then (action1 AND action2 AND action3), where *cond1* is a trigger and *action1* is an action. This is possible because we store the complete rule on *Rule Server* and can be fetched in its entirety when required. If the rules were split among multiple servers, it would be hard to fetch all the parts of a complex rule and assemble them together. Secondly, if collusion among servers is somehow identified, for e.g., if the profile and rule information are simultaneously leaked and we can indirectly detect it, the

colluding servers can be identified for our model. This is because in our model, different pieces of information (profiles, rules/apps, IoT device data) are stored on different servers. Thirdly, we define a metric called *No Collusion Probability (NCP)*. It is defined as *(1- probability of collusion of servers in a model)* for the same amount of information leakage. Since we have four pieces of private information, i.e., user profiles, rules, apps and IoT device data, we can split the original IFTTT model where there is just once entity storing and processing all the information, into 2, 3, or 4 servers. It can be mathematically proved that *NCP(2 servers)<NCP(3 servers)<NCP(4 servers)*. This can be intuitively explained as well. The likelihood of 2 servers colluding is more than the likelihood of all 3 servers colluding which in turn is more than the likelihood of all 4 servers colluding. Moreover, there is a tradeoff between *NCP* and resources such as inter-server communication delays and the amount of hardware required in terms of servers. Our goal is to therefore improve the *NCP* while reducing the amount of resources required. Keeping this goal in mind, we have chosen the number of servers in our proposed CBS to be 3. In simple terms, though the 2 server model requires the least resources, even a single collusion event reveals the complete information. On the other hand, the 4 server model makes it most difficult to reveal the complete information by collusion, it requires much more resources. Hence, we settle for the middle solution, i.e., the 3 server model.

Cloud Service	Trust distribution over multiple servers	Policy based solution for privacy	Technical solution for privacy
IFTTT	✗	✗	✗
Proposed CBS	√	√	√

Fig.3. Comparison of IFTTT and our proposed CBS

7. CHALLENGES

In this section, we mention some aspects of our work that need further investigation and may be addressed in our future work. For starters, the optimal solution for privacy in IoT cloud services is that the users perform most of the processing (for e.g. app connection, rule creation and storage, IoT device data storage) themselves and only communicate with the API services to read data from IoT devices and send actions to actuating devices. We need to compare the performance of our proposed CBS with this privacy optimal solution. Secondly, though we use *NCP* to measure the likelihood of collusion not taking place among servers, a probabilistic guarantee of no collusion is weak. It needs to be modified and made stronger, at par with cryptographic probabilistic guarantees if possible. Thirdly, in case that collusion has already happened between two or more servers, detecting such collusion is an open problem and needs to be investigated. There are some works in digital forensics that can be used towards this purpose. Finally, we need to specify the particular implementation of PN that is used in our proposed CBS. Moreover, though pseudonym is a well-known technique for achieving user anonymity, it may not guarantee complete anonymity. For e.g., some works in the area of Bitcoin [11] show that the user identity can still be revealed through transaction history analysis.

8. CONCLUSION

In this report, we investigated the privacy concerns of a cloud service for IoT device management. Further, we proposed an alternative cloud based service with privacy preserving architecture. The proposed solution implemented trust distribution over multiple servers (profile server, rule server and device server), policy based solution for privacy (server management by independent organizations) and technical solutions for privacy (encrypted inter server communication and pseudonyms). Subsequently, we analyzed our proposed model in detail and presented some significant insights. In the end, few challenges were identified which we intend to address in extensions to this work in future.

9. REFERENCES

[1] Dave Evans. April 2011. The Internet of Things: How the Next Evolution of the Internet Is Changing Everything (PDF). *Cisco*.

[2] https://ifttt.com/privacy

[3] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle. 2014. Privacy in the Internet of Things: threats and challenges. *Security and Communication Networks*, *7*(12), 2728-2742. DOI= http://dx.doi.org/10.1002/sec.795

[4] P. Porambage et al. 2016. The Quest for Privacy in the Internet of Things. *IEEE Cloud Computing*, *3*(2), 36-45. DOI= http://dx.doi.org/10.1109/MCC.2016.28

[5] M. Abomhara, & G.M. Køien. 2014. Security and privacy in the Internet of Things: Current status and open issues. In *Privacy and Security in Mobile Systems (PRISMS), 2014 International Conference on* (pp. 1-8). IEEE. DOI= http://dx.doi.org/10.1109/PRISMS.2014.6970594

[6] R. Roman, J. Zhou, & J. Lopez. 2013. On the features and challenges of security and privacy in distributed internet of things. *Computer Networks*,*57*(10), 2266-2279. DOI= http://dx.doi.org/10.1016/j.comnet.2012.12.018

[7] Blase Ur et al. 2016. Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (CHI '16). ACM, New York, NY, USA, 3227-3231. DOI: http://dx.doi.org/10.1145/2858036.2858556

[8] S. Vorapojpisut. 2015. A Lightweight Framework of Home Automation Systems Based on the IFTTT Model. *Journal of Software,10(12)*, 1343-1350. DOI= http://dx.doi.org/10.17706/jsw.10.12.1343-1350

[9] Miguel Coronado and Carlos A. Iglesias. 2016. Task Automation Services: Automation for the Masses. *IEEE Internet Computing* 20, 1 (January 2016), 52-58. DOI= http://dx.doi.org/10.1109/MIC.2015.73

[10] Tiffany Hyun-Jin Kim, Lin-Shung Huang, Adrian Perring, Collin Jackson, and Virgil Gligor. 2013. Accountable key infrastructure (AKI): a proposal for a public-key validation infrastructure. In *Proceedings of the 22nd International conference on World Wide Web* (WWW '13). ACM, New York, NY, USA, 679-690. DOI: http://dx.doi.org/10.1145/2488388.2488448

[11] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. 2014. Deanonymisation of Clients in Bitcoin P2P Network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (CCS '14). ACM, New York, NY, USA, 15-29. DOI: http://dx.doi.org/10.1145/2660267.2660379

[12] Seungyeop Han, Vincent Liu, Qifan Pu, Simon Peter, Thomas Anderson, Arvind Krishnamurthy, and David Wetherall. 2013. Expressive privacy control with pseudonyms. *SIGCOMM Comput. Commun. Rev.* 43, 4

(August 2013), 291-302. DOI: http://dx.doi.org/10.1145/2534169.2486032

[13] Katrin Borcea-Pfitzmann, Elke Franz, and Andreas Pfitzmann. 2005. Usable presentation of secure pseudonyms. In *Proceedings of the 2005 workshop on Digital identity management* (DIM '05). ACM, New York, NY, USA, 70-76. DOI=http://dx.doi.org/10.1145/1102486.1102498

[14] Janaka Seneviratne, Udaya Parampalli, and Lars Kulik. 2014. An authorised pseudonym system for privacy preserving location proof architectures. In *Proceedings of the Twelfth Australasian Information Security Conference - Volume 149* (AISC '14), Udaya Parampalli and Ian Welch (Eds.), Vol. 149. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 47-56.