Securing Data Storage System for Internet of Things Using Key Aggregate Cryptosystem

A D Ambade, J R Pansare

Abstract— In the most recent couple of decade, the thought of connecting existing computing devices has brought forth a new concept called "connecting things". Due to the advances in data collection technology in sensors, such as embedded devices, ubiquitous and RFID technology has led large number of gadgets connected in network which are consistently transmitting their data over the time. This data obtained is very precious to many enterprises, so there is need of secure mass storage system for this data. To make the transmission and storage of data more secure, data is encrypted before storing. In the proposed system the data obtained from the IoT network is divided into blocks based on the default threshold, so that it is more reliable and secure. The blocks are then encrypted using the Key Aggregate Cryptosystem (KAC). Using the KAC scheme the data is encrypted with a public key and a ciphertext class after it is divided into blocks. An aggregate key is generated for set of blocks which is used in decryption.

Index Terms— Internet of Things(IoT); Security; Key Aggrgate Cryptosystem(KAC)

1 Introduction

MODERN business creates an increasing demand for sharing, querying and mining information over autom-ous enterprises while maintaining the privacy. The explosive progress in networking, storage, and processor technologies is resulting in an unprecedented amount of digitization of information. The recent development in the networking the concept called Internet of Things, which connects various computing devices and interoperates with each other to share information. Each device in a network has a processing power which led to the generation of huge data.

The term "Big Data" is used for data with three V's, Volume, Velocity and Variety. Today, with improvement in technology, the data is also increasing exponentially. From the RFID tags and industrial equipment to jet engines and electronic devices, the world around us is generating ever increasing amount of data. This huge and growing data is important to many business enterprises to perform data mining and analytics activities. Many companies use this data to improve products, identify defects and enhance security. So, there is need for mass storage system [1] to store this ever increasing data.

The four things to consider while designing a storage system are availability, reliability, flexibility and security. Availability ensures that data is available throughout. For this data is replicated to storage device at different location. So that data available at the time of system failure. Reliability ensures that data integrity is maintained. Data should be same as it was stored and not altered. Flexibility is that devices can be added and removed as and when required. The major concern today is security which is ensures that data stored is secure. Securing stored data involves preventing unauthorized access along with preventing destruction of data (accidental or intentional), corruption or infection of information.

The main objective is to provide security to storage system using KAC and to implement encryption and decryption algorithms based on KAC. To make the transmission and storage of data more secure, data is encrypted before storing. It is not feasible and secure to encrypt the whole data obtained via IoT network. In the proposed system the data obtained by the IoT network is divided into blocks based on the default threshold, so that it is more reliable and secure. The blocks are then encrypted using the Key Aggregate Cryptosystem (KAC).

Key Aggregate Cryptosystem is a special type of scheme designed to share data securely over cloud. Using KAC, the client will encrypt each data block not only under a public-key, but also under an identifier of ciphertext called class. That means the ciphertexts are further categorized into different classes. The key owner, that is, holds a master-secret called master secret key, which can be used to extract secret keys for different classes. The extracted key can be an aggregate key which is as compact as a secret key for a single class, but aggregates the power of many such keys, that is, the decryption power for any subset of ciphertext classes.

2 Related Works

Internet of Things virtually is a network of real world systems with real-time interactions [2]. Various security and privacy threats are discussed in [3][4]. Security issues are categorized as front-end, back-end and network threats, and privacy concerns in IoT are categorized as privacy in device, privacy during communication, privacy in storage and privacy at processing. Security should be provided not only for data in transit but also for data at rest i.e., dynamic and static data respectively. There are many schemes designed to protect the data, cryptographic or non-cryptographic. Hai Jiang et al[5], has designed a secure storage system to store aggregate IoT data obtained from various devices. They used Shamir's secret sharing scheme to protect data. Shamir's secret sharing algorithm is used with added internal padding. Before storing the data it is split into shares and this data is stored to different storage devices.

POTSHARDS (Protection Over Time, Securely Harbouring And Reliably Distributing Stuff) is a long term secure storage system without using encryption [6]. Encryption is unsuited in many situations for data storage which requires indefinitely long periods of time. Since key management is difficult cryptosystems should be updated to provide secrecy through encryption over periods of decades. The goal of the system is provide security to relatively static data with indefinite long lifetime.

Shield is a stackable secure storage system for sharing file in public storage[7]. Here a proxy server is in charge of authentication and access control. A new variant of Merkel Hash Tree was proposed which support file content update and efficient integrity checking. Also a hierarchical key organization was designed for convenient key management and efficient permission revocation. Cryptonite[8] is secure storage repository for sharing scientific datasets in public cloud. Architecture for a secure distributed repository is proposed by Haupt et al[9],for Grid environments supporting secure sharing of confidential data by members of ad-hoc created groups. This repository is designed to make it easier for ad-hoc collaborations which require sharing of restricted-access data between members of dynamically created groups.

HASS[10] is a Highly Available, Scalable and Secure distributed data storage systems. To achieve scalability in terms of performance and key management, file systems such as Object based Storage Devices (OSD) and Identity Based Encryption (IBE) are integrated. Su Chen et al[11], has proposed a revised Blakley's secret sharing scheme is applied to the DFS to support security and reliability without sacrificing scalability .The distributed system is deployed with Graphics Processing Unit (GPU). There are two phases of secret sharing scheme: share generation and data restoration. To improve the security level the large data is split into shares and a random number is added.

Companies that handle critical data are using cloud storage services to store their data due to its increasing popularity. The critical data includes medical record databases, power system historical information and financial data. DepSky[12] is a system that provides encryption, encoding and replication of the data on diverse clouds that form a cloud-of-clouds to improve the availability, integrity and confidentiality of information stored in the cloud.

3 Proposed System

The aim is to design a secure storage system for the aggregate data generated from the IoT network. The data generated from IoT network is huge, which require secure mass data storage systems. In the proposed system Key Aggregate Cryptography scheme is used to secure the data. KAC is a public key encryption scheme in which any set of cipher text is decryptable by a constant size decryption key[13]. This key is known as an aggregate key.

The data collected by client from IoT network is divided into blocks of fixed threshold. For each data block a key pair is generated using KeyGen, such as public and master key. Then each data block is encrypted using encrypt algorithm of KAC scheme. Here each data block is encrypted with a public key

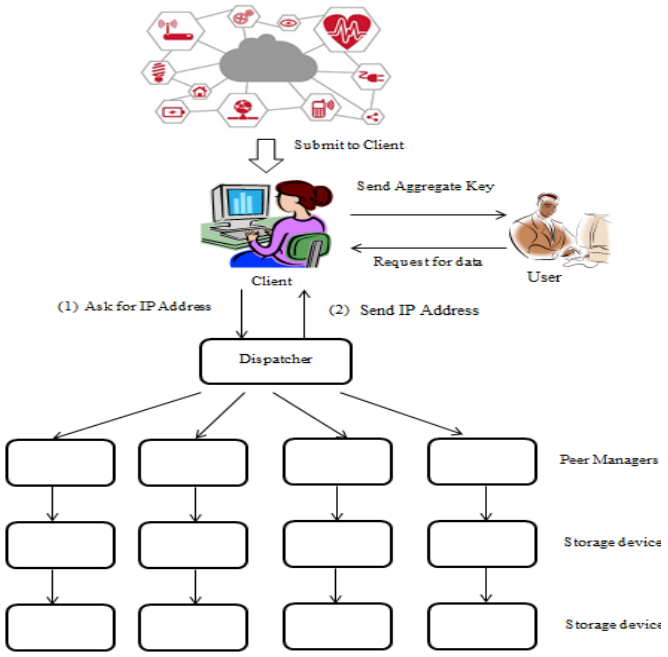

Fig.1: Syatem Architecture

and a ciphertext identifier. The output of this phase is the encrypted data. Then the encrypted data is stored to the storage devices. To access that data client has a master key. But if any other user wants to access any set of data, the user needs to send a request to the client. Then client will send an aggregate key to the user for the requested set of data. With the help of aggregate key the user can access the requested set of data blocks.

3.1 System Architecture

There are mainly five components: client, dispatcher, peer managers, regular storage devices and user. Fig.1 shows the system architecture. Client is the data owner. To save the data to the system, client will divide the data into blocks based on the fixed threshold. According to the KAC scheme the client will be configured to the system by a general registration step. The client will divide the data into blocks of fixed threshold. Then a key pair is generated, namely master key and a public key. Master key is kept as a secret with the client while public key is made public. Each data block is encrypted with the public key and a ciphertext class. The encrypted data is now ready to be uploaded to the storage devices. To store the data client node contacts the dispatcher about where to save. In this file management system, the dispatcher is the only centralized node. However, it only maintains the IP addresses of peer managers and will not involve in file operations. The dispatcher serves as a proxy and selects peer groups/managers according to the work load situation or in round robin manner. Based on the clients' request, a number of peer groups are selected and their peer managers IP addresses are returned. The client node keeps these addresses locally for the future use. Then each encrypted data block is distributed to corres-

ponding peer managers. Once peer managers receive their shares, replication phase starts for high availability.

To retrieve a file, the client node does not need to contact the dispatcher with the locally cached addresses. This reduces the possibility of turning it into a bottleneck for storage operations. With locally maintained manager lists, the client node can contact related peer managers directly. Now if a trusted user other than client wants to access a set of stored data blocks, he has to send a request to the client for the required set of blocks. The client will then with the help of a secure channel like an email will send an aggregate key for a set of requested data blocks to the user. With the help of the aggregate key the client will decrypt the data blocks.

3.2 Algorithms

The key aggregate encryption scheme consists of five algorithms. The data owner i.e., client establishes the public system parameter via Setup algorithm and generates a public/master-secret key pair via KeyGen algorithm. The data blocks can be encrypted via Encrypt algorithm. The data owner can use the master-secret to generate an aggregate decryption key for a set of encrypted data blocks via Extract algorithm. The generated keys can be passed to other users securely (through secure e-mails or secure devices) Finally, any user with an aggregate key can decrypt any set of data blocks provided that the ciphertexts identifier is contained in the aggregate key via Decrypt algorithm.

- Setup(1^{λ}, n): Takes number of ciphertext classes n and the group order parameter λ as input. The client executes setup phase by creating an account with the system. In this phase the data collected by client is divided into blocks based on the fixed threshold
- KeyGen( ): executed by the data owner to randomly generate a public/master-secret key pair (*pk, msk*). In this phase a pair of keys i.e., public key and a master key is generated for each block.
- Encrypt(*pk, i, Bi*): On input a public-key pk, an index i denoting the ciphertext class, and a data blocks Bi, it outputs a ciphertext *C*. Each data block is encrypted with a public key and a ciphertext class in this phase.
- Extract(*msk, S*): On input the master-secret key msk and a set *S* of indices corresponding to different classes, it outputs the aggregate key for set *S* denoted by K_S. This phase is executed by data owner for delegating the decrypting power for a certain set of data blocks and it outputs the aggregate key for set data blocks.
- Decrypt(K_S, *S, i, C*): On input K_S, the set *S*, an index i denoting the ciphertext class the ciphertext C belongs to, and *C*, it outputs the decrypted result m if $i \in S$. This phase is executed by a delegate who received, an aggregate key generated by Extract. On input of an aggregate key, set of data blocks, the output is the original data block.

3.3 Implementation Details

The aim of proposed system is to protect the data in transit and data at static. The in transit is directed from client to storage devices, while static data is the data stored in the peers. It is really difficult to conduct the experiment on real storage devices as it requires huge amount of resources and cost. The system will be working in five major olephases:

- Setup phase
- Key generation phase
- Encryption phase
- Extract Phase
- Decryption phase

In the setup phase the client will register to the system through portal. Next step is to divide the collected data into blocks of fixed threshold, as it is not feasible to store and encrypt the entire collected data. After dividing the data into blocks a key pair of public key and master key is generated. In the encryption phase each data block is encrypted with a public key along with a ciphertext identifier. The master key is used by the client to decrypt the data blocks. If any other user wants to access any set of data blocks, he has request for the same. Client will send an aggregate key to the user of constant size, which will be used to decrypt the requested set of data blocks. Aggregate key generation is done in the extract phase.

4 Conclusion

In the last couple of decades, with the advancement in integrated circuit and networking technologies, computers, devices and networks have become highly pervasive with the introduction, development and deployment of the Intetnet of Things. These tiny identifying devices and wearables process, generate and store data at an exponential rate with the increase in their use worldwide. This data is precious to many enterprises, so there is requirement of secure mass storage systems.

The proposed system is a secure storage system for IoT data based on the key aggregate cryptosystem scheme. As it is not feasible to encrypt the whole data, here data is divided into blocks and then each block is encrypted. More security is provided as each block is encrypted with a public key and a block identifier. To decrypt any set of data a constant size aggregate key is generated.

References

[1] C. Aggarwal, N. Ashish, A. Sheth, "The Internet of Things: a survey from the datacentric perspective, in: Managing and Mining Sensor Data", Springer, 2013.

[2] Atzori, Luigi, Antonio Iera, and Giacomo Morabito. "The internet of things: A survey." Computer networks 54.15 (2010): 2787-2805.

[3] Rolf H. Weber, "Internet of Things – New security and privacy challenges", in computer law & security review, volume-26(2010)

[4] J. Sathish Kumar, Dhiren R. Patel, "A survey on Internet of Things: security and privacy issues", Int. J. Comput. Appl. 90 (11) (2014).

[5] [5] Hai Jiang, Feng Shen, Su Chen, Kuan-Ching Li, Young-Sik Jeong, "A secure and scalable storage system for aggregate data in IoT", In Future Generation Computer Systems, Elsevier(2015)

[6] Mark W. Storer, Kevin M. Greenan, Ethan L. Miller, Kaladhar Voruganti, "POTSHARDS: secure long-term storage without encryption",

in: Proceedings of USENIX Annual Technical Conference, 2007.

[7] Jiwu Shu, Zhirong Shen, Wei Xue, "Shield: a stackable secure storage system for file sharing in public storage", *J. Parallel Distrib. Comput.* 74 (9) (2014).

[8] Alok Kumbhare, Yogesh Simmhan, Viktor Prasanna, "Designing a secure storage repository for sharing scientific datasets using public clouds", *in: Proceedings of the Second International Workshop on Data Intensive Computing in the Clouds*, 2011.

[9] Tomasz Haupt, Anand Kalyanasundaram, Igor Zhuk, "Architecture for a secure distributed repository", *in: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing*, 2006.

[10] Zhiqian Xu, Hai Jiang, "HASS: highly available, scalable and secure distributed data storage systems", *in: Proceedings of the 2009 IEEE/IFIP International Symposium on Trusted Computing and Communications*, 2009.

[11] Su Chen, Yi Chen, Hai Jiang, Laurence T. Yang, Kuan-Ching Li, "A secure distributed file system based on revised Blakley's secret sharing scheme", *in: Proceedings of the 11th IEEE International Conference on Trust*, Security and Privacy in Computing and Communications, Liverpool, UK, 2012.

[12] Bessani, Alysson, et al. "DepSky: dependable and secure storage in a cloud-of-clouds." *ACM Transactions on Storage (TOS)* 9.4 (2013): 12.

[13] Cheng-Kang Chu, Sherman S.M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng, "Key-aggregate cryptosystem for scalabledata sharing in cloud storage", *IEEE Trans. Parallel and Distributed System*, vol.25, pp. 468-477, 2014