Toward Computing over Encrypted Data in IoT Systems

The multitude of IoT devices contributes to the enormous amount of data stored on corporate clouds. Yet the level of computing power has outpaced advances in privacy protection. Could encrypted search preserve the privacy of data, while utilizing the computing power of the cloud?

By Hossein Shafagh

DOI: 10.1145/2845157

Internet of Things (IoT) application scenarios are finding their ways into various aspects of our lives: health- and activity monitoring, home-automation, elderly care, and connected cars, to name a few. Ultimately, IoT applications aim to create unique forms of interaction with our environment and provide novel insights based on rich data collected from the physical world. A major concern hereby is how to preserve the privacy of user data and still be able to provide insightful information. Methods based on encrypted data computing appear to be promising approaches. However, they bring new challenges with respect to functionality and computing resources, which must first be solved before they can be integrated into IoT systems. In this article, we discuss encrypted data computing approaches, their use-cases, and the challenges yet to be addressed, especially with regard to the IoT.

INTRODUCTION

IoT applications contribute to our increasingly cloud-centric and data-driven world by digitizing elements of the physical world that were previously non-digital. The potential of such increasingly larger amounts of data is yet to be fully explored. One important consequence of this increase in data collection is the growing intrusion into sensitive user data. These intrusions stem not only from security attacks, but also from normal, more innocent data collection mechanisms. Sophisticated machine learning tools and inter-correlation techniques allow for the inference of private information even from seemingly innocuous data. For instance, consider common health monitoring applications such as FitBit. The data FitBit collects—such as heart rate, location, step-counts, and quality of sleep—can be used to infer private information such as chronic diseases, social interactions, and mental health conditions, to name only a few examples. From the user's point of view, inferring such information might be of value. However, the user might not be interested in sharing such information with a corporate service provider. In other words, we should avoid unauthorized sharing of data, as well as disclosure of sensitive information, which can be learned from our data. Incidents from the past (e.g., online disclosure of sexual activity of health tracking users) show that user awareness and corporate inclination to address these issues are still missing.

An intuitive and simple approach to preserve the privacy of stored data is to only send and store encrypted data to the cloud, as already practiced by zero knowledge cloud storage vendors (e.g., Tresorit). This technique relies on efficient symmetric key encryption schemes, such as Advanced Encryption

Standard (AES) block ciphers. This way the cloud is unable to learn anything about the stored data. However, the downside of this approach is twofold: (i) the computing power of the cloud cannot be utilized to process the data; and (ii) in order to analyze the data, the user needs to first download, decrypt, and finally process the data. This requires high bandwidth and computing power on the user-side, and it results in undesired application delays. Moreover, users lose the power to search their data, which is in fact the most important functionality considering the ever-increasing amount of data.

Encrypted search schemes, however, allow an untrusted party to perform search operations over encrypted data without the need for decryption. Consider the case of structured data stored in database systems, the underlying search operations are performed through queries. These queries are structured based on simple mathematical operations, such as addition, comparison, and equality checks. Hence, an encrypted query processing, in its essence, should be capable of performing such mathematical operations in the ciphertext domain.

Previous *XRDS* articles have shed light on encrypted search and fully homomorphic cryptosystems [1, 2]. These approaches lay the foundation for encrypted query processing. Here we focus on practical solutions for interacting with encrypted data stored in databases. Practical solutions have to strike a good balance between security, efficiency, and functionality. Moreover, the engineering efforts to build such systems provide invaluable insights into how to further optimize cryptographic schemes, which may remain undiscovered in the theory.

In the following, we first elaborate on challenges specific to IoT systems, continue with an overview of encrypted query processing techniques, and conclude with open research questions.

CHALLENGES OF IOT SYSTEMS

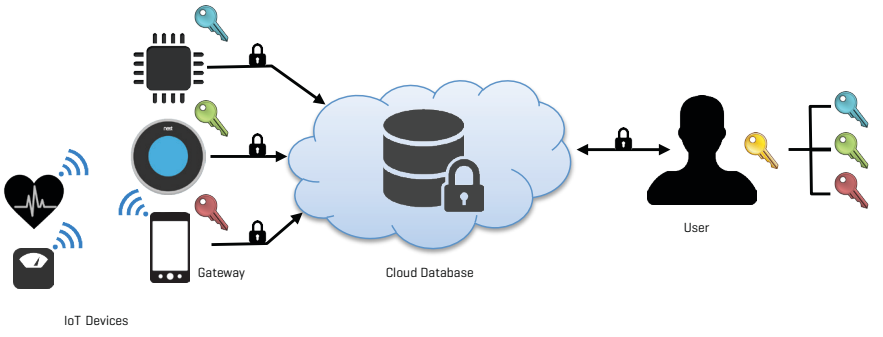
IoT devices are inherently resource limited with regard to energy. Energy constraints mandate the use of low-power components, thus limiting the computational capabilities, transmission range, and communication bandwidth of IoT devices. Furthermore, new emerging application scenarios differ from the currently dominant web or mobile application traffic in the following aspects.

Connectivity. Wireless, embedded devices can be connected to the Internet by means of a gateway. This connection, however, can either be long-term (as is the case in home access points) or opportunistic (as is the case with smartphones for wearables). The connectivity characteristics impact the protocol design in order to guarantee tolerable communication delays.

Machine-to-machine. Even if the user is interested in the data, he or she might not necessarily be involved in the communication loop of IoT devices. Devices collect data, which they store in the cloud; other devices can then be triggered by pre-defined events. This means there is no human being involved who could be intercepted while entering credentials, such as username or passwords.

Content. The communicated information in IoT scenarios is composed primarily of sensor readings and metadata. By design, IoT devices can only store data for a limited amount of time. Hence, the need for cloud storage in the IoT is unavoidable, not only to store data,

Figure 1. The encrypted search pipeline.



but also to make this data accessible to other devices and services.

Resource asymmetry. The components involved in an IoT system—such as the IoT mote, the gateway, and the back end—exhibit a strong asymmetry in available resources. For instance the clock frequency reflecting computing power spans from a few MHz for the IoT device (e.g., 32 MHz ARM Cortex M3 microcontrollers), to a few GHz at the gateway (e.g., 1.4 GHz dual-core iPhones), and to potentially several GHz at the back end. The same asymmetry applies for both memory and bandwidth.

Energy for IoT devices is provided by batteries or is harvested, whereas a gateway might be connected to the mains (access point) or equipped with a significantly stronger battery (smartphone). The back end, on the other hand, is considered to have continuous access to sufficient energy. Consequently, the energy issue is one of the most important design factors for IoT systems. When designing secure systems for the IoT, one has to bear in mind these particular properties and limitations.

ENCRYPTED QUERY PROCESSING

One common way to store IoT data is in structured databases, such as SQL databases. In an encrypted query processing system, a plaintext SQL query is transformed to an encrypted query, such that the cloud cannot learn about the values in the query. The query is then executed over encrypted data and the encrypted result is sent back to the user. For example, consider a health data system. A simple query to get all entries with heart rate larger than 100, for instance, is translated into an encrypted query, where 100 is replaced with its order-preserving encryption (described later in this section) and the heart rate with the corresponding encrypted column name:

```
SELECT * FROM tab WHERE
heart rate > 100
SELECT * FROM tab WHERE
column-043 > 0x19
```

The fact that database queries are based on mathematical operations has inspired researchers to build such encrypted database (EDB) systems, which can operate over encrypted data. CryptDB is one of the early systems, which employed property-preserving encryption schemes and partial homomorphic encryption to build an efficient EDB [3]. To this end, such EDBs rely on the insight that to support common SQL-like queries, it is necessary to be capable of performing equality checks and have knowledge about the order of encrypted values. However, enabling computation over encrypted data also means leaking information. That is, any order-preserving encryption scheme will, by definition, reveal order relations. The encryption schemes are selected per column and account for the intended query type (e.g., min, order by, etc.). Hence, data items that are not involved in the processing of queries should be encrypted with the strongest cryptographic scheme (i.e., probabilistic encryption). For the encrypted query processing, the following encryption schemes can be utilized.

IoT applications contribute to our increasingly cloud-centric and data-driven world by digitizing elements of the physical world that were previously non-digital.

Random (RND). Probabilistic or random encryption is the strongest security scheme, allowing no operation over encrypted data. This scheme is the conventional scheme, widely used in secure communication and storage. It has the property that the encryption of the same plaintext m results in two different ciphers c1 and c2 such that c1 and c2 are by no means related (i.e., semantically secure under chosen plaintext attack or CPA). AES in cipher block chaining (CBC) mode has such properties, and efficient hardware implementations of it are already integrated in most IoT devices. This allows the computation of AES-CBC on a typical IoT device in a few microseconds.

Homomorphic encryption (HOM). Research on fully homomorphic cryptosystems has made significant advancements in the recent years and been able to show that arbitrary computations on encrypted values can be implemented [4]. However, the computations involved are presently prohibitively expensive even for full-fledged devices and highly infeasible for resource-limited devices. In order to support sum and average operations over encrypted data, it is however sufficient to utilize additive homomorphic encryption schemes, such that:

$$\text{decrypt}(c_1 \square c_2) = \text{decrypt}(c_1) + \text{decrypt}(c_2)$$

The Paillier cryptosystem is one of the most well-known, additive homomorphic schemes [5]. One challenge of Paillier, with regard to the limited bandwidth in the IoT domain, is its ciphertext expansion. In the Paillier cryptosystem the plaintext size is between 1 to n bytes, where n is the key length. Regardless of the plaintext size, the ciphertext has a size of $2n$. This would mean with a 1024 bit key, the encryption of a 32-bit integer value requires 256-byte space. Moreover, the encryption appears to be a costly operation for low-power devices. For instance, the encryption of a 32-bit integer requires 1.6 seconds on the ARM Cortex-M3 microcontrollers, which is several times higher than the 9 milliseconds required on a desktop machine.

Deterministic (DET). Deterministic encryption allows for equality checks. The encryption of the plaintext m results always into the same cipher c. AES in electronic codebook (ECB) mode is a block-cipher encryption with such a property. Due to this deterministic property it is in general advised not to use ECB for encryption of large packets, as an attacker can: change the order of the blocks or replace a block in an indistinguishable manner (i.e., substitution attack), or learn information about the plaintext with a histogram of repeated blocks. Therefore, for maximum security in DET, AES-ECB should be only used for plaintexts smaller than or equal to 16 bytes. For large plaintexts, AES in CMC mode can be utilized [6]. AES-CMC is a tweaked combination of AES-CBC with a zero initialization vector, where AES-CBC is applied twice on the input. The second CBC round is applied in the reverse order, i.e., from the last block to

the first block. This way, the first blocks become deterministically random and do not leak equality within a data item. The performance of DET is comparable to RAND.

Order-preserving encryption (OPE). The order relationship between the plaintext inputs m_1, m_2, and m_3 is preserved after encryption, i.e.,

if $m_1 \leq m_2 \leq m_3$, then $c_1 \leq c_2 \leq c_3$

This way, the order information among the encrypted data items c_i is revealed, but the data itself is not. Order comparison is a common operation in SQL-like databases, such as for sorting, range checks, ranking, etc. One of the first provably secure OPE schemes is the approach introduced by Boldyreva et al. [7]. This OPE scheme is, however, as computationally intensive as Paillier encryption. The interactive OPE approach by Popa et al. solely relies on symmetric cryptography and trades computation overhead for latency (i.e., it involves more communication) [8]. This lightweight OPE scheme is referred to as mutable order-preserving encoding (mOPE), as the order encodings are mutable. Popa et al. prove that mOPE fulfills the ideal security (IND-OCPA), i.e., no additional information than the order is revealed. mOPE is more secure than any other OPE approach and, yet, one to two orders of magnitude less computationally intensive than traditional OPE schemes. Although mOPE appears to be more suitable for the IoT, the status of opportunistic connectivity might have an impact on the performance of this protocol.

PRACTICAL SECURE SYSTEMS

While designing a secure encrypted query processing system for the IoT domain, the challenge becomes centered upon which entity should take the role of performing the encryption or decryption operations and thus has access to encryption keys. Pushing this role to the origin of the data would allow protection of the data at the source, but poses the challenge of minimizing the impact of computational overheads on the battery-life of IoT devices. In the ecosystem of the IoT, the gateway could be utilized for heavy computations. Here the challenge is how to establish trust with the gateway and execute private functions over encrypted data. Once the data is stored in the correct form on the back end, the challenge becomes about how to allow other low-power IoT devices to securely retrieve information from the EDB.

The need for cloud storage in the IoT is unavoidable, not only to store data, but also to make this data accessible to other devices and services.

It is important to stress that encrypted query processing is possible due to utilization of weaker encryption schemes, such as DET and OPE, which leak information. Hence, the decision about which data types to encrypt with the property-preserving encryption schemes becomes very important. Recently, Naveed et al. showed how simple attack techniques can be used to disclose encrypted medical data in the OPE and DET schemes [10]. The attack schemes require access to auxiliary information, such as range of data and distribution of it, which is seemingly simple to retrieve for certain application scenarios. Given the protected plaintext has low entropy, techniques such as frequency attack can be used to learn about encrypted data. With respect to IoT data, the question is: How applicable are such attacks on sensor readings?

The resource asymmetry in IoT systems makes research efforts on homomorphic evaluation of the AES circuit appear promising [9]. Efficient and low-power AES encryption is readily available in most radio front ends of constrained devices. The possibility of transforming AES ciphertexts into homomorphic ones would allow for the minimization of the encryption overhead on IoT devices, while allowing systems to exploit the full potential of fully homomorphic encryption in the powerful cloud. In more practical terms, this means the current computational overhead of four minutes for a single AES-128 encryption operation must be significantly reduced.

CONCLUDING REMARKS

Applied cryptography is gaining popularity among cryptographers. System and networking researchers can benefit from this, in that they can capture more insights about novel cryptographic schemes and explore their feasibility and benefits in practical systems. The challenges due to the ecosystem of IoT applications require a more careful system design with regard to resource constraints. The effort of realizing theoretical cryptographic approaches under real-world conditions sheds light on undiscovered weaknesses and creates opportunities for enhancements in cryptosystems. It remains an important open research problem to design and prove secure and practical encrypted data processing schemes for the IoT domain.

References

[1] Kamara, S. Encrypted search. *XRDS* 21, 3 (2015).

[2] Wu, D. Fully homomorphic encryption: Cryptography's holy grail. *XRDS* 21, 3 (2015).

[3] Popa, R. A., Redfield, C. M. S., Zeldovich, N., and Balakrishnan, H. CryptDB: Protecting confidentiality with encrypted query processing. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles* (SOSP '11). ACM, New York, 2011, 85-100.

[4] Gentry, C. A fully homomorphic encryption scheme. Ph.D. thesis. Stanford University: AAI3382729, Advisor: Dan Boneh. 2009.

[5] Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques* (EUROCRYPT '99). ACM, New York, 1999, 223-238.

[6] Halevi, S. and Rogaway, P. A tweakable enciphering mode. In *Advances in Cryptology* (CRYPTO '03). ACM, New York, 2003.

[7] Boldyreva, A., Chenette, N., Lee, Y., and O'Neill, A. Order-preserving symmetric encryption. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques* (EUROCRYPT '09). ACM, New York, 2009, 224-231.

[8] Popa, R. A., Li, F. H., and Zeldovich, N. An ideal-security protocol for order-preserving encoding. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, Washington D.C.,2013, 463-477.

[9] Gentry, C., Halevi, S., and Smart, N. P. Homomorphic evaluation of the AES circuit. In *Proceedings of Advances in Cryptology* (CRYPTO '12). ACM, New York, 2012.

[10] Naveed, M., Kamara, S., and Wright, C. V. Inference attacks on property-preserving encrypted databases. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (CCS '15). ACM, New York, 2015.

Biography

Hossein Shafagh is currently a second-year Ph.D. student at the computer science Department of ETH Zurich, Switzerland. His primary research interests are in the design of secure communication systems for low-power wireless devices.

1528-4972/15/12 $15.00