Demo Abstract: Talos a Platform for Processing Encrypted IoT Data

Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi
Department of Computer Science
ETH Zurich, Switzerland
{shafagh, lubu, hithnawi}@inf.ethz.ch

ABSTRACT

Internet of Things (IoT) applications today often utilize the cloud to provide storage and ubiquitous access to collected data. Leaving such granular, sensitive, and personal data unprotected on the cloud and vulnerable to system breaches or curious administrators is critical. In this extended abstract, we present Talos [3], a framework that embraces the computational resources available in the cloud and exploits them for encrypted data processing (e.g., using partially homomorphic encryption schemes). We have developed Talos further to enable encrypted data sharing, such that users can securely share their homomorphically encrypted data with add-on services and other users. We demonstrate a prototype implementation of a real-world application utilizing Talos. Talos enables applications to efficiently store and interact with encrypted sensory data in the cloud. With our prototype application, we show that with Talos in place the user experience remains unchanged (i.e., response time below 1 s) although all operations over data in the cloud incorporate encrypted data processing. Moreover, we demonstrate how Talos can be smoothly integrated into IoT apps (currently supporting Contiki and Android). Hence, we hope to encourage the development of secure IoT applications on top of the Talos framework.

1. INTRODUCTION

The confluence of networked embedded devices, wearables, and sensing technologies, commonly referred to as the Internet of Things (IoT), has expedited the emergence of a whole spectrum of powerful applications that is radically changing the way we perceive and interact with the physical world. One of the prominent and promising offerings in the IoT domain is the emerging field of fitness and health tracking which has led to impactful innovations.

Today, we have access to a plethora of fitness and health monitoring and tracking applications and devices. These track a broad range of data from sensory values (e.g., body temperature), activity meta-data (e.g., duration, type), to health-related symptoms (e.g., migraine headaches). The collected data can be used to infer privacy-sensitive information, such as heart-related diseases, personal well-being, and fertility-related data (e.g., based on body temperature). While the quantification of such health-related data has many benefits, the storage of it on third-party clouds is accompanied with privacy risks. For instance, unauthorized data disclosures can be for personal advertising [4], for trading with insurances [1], or due to data breaches.

SenSys '16 November 14-16, 2016, Stanford, CA, USA
© 2016 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4263-6/16/11.
DOI: http://dx.doi.org/10.1145/2994551.2996536

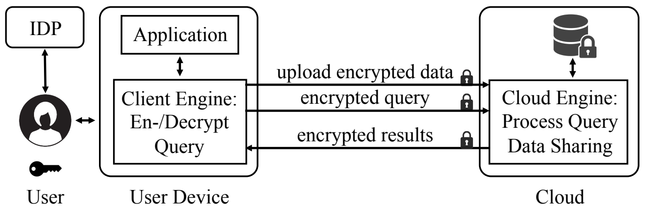

Figure 1: Overview of Talos.

To preserve the users' privacy, it is crucial to protect their data while stored in the cloud. The intuitive solution to utilize an efficient symmetric encryption scheme, such AES, to store only encrypted data on the cloud renders them, however, unsearchable. Therefore, it requires the download of large data volumes to a trusted client prior to processing. This requires high bandwidth and computational resources on the client and would cause undesirable application delays.

Approach. Our framework Talos [3] addresses this issue leveraging encrypted data processing. While fully homomorphic encryption schemes [2] allow arbitrary computations on encrypted data, there are presently too expensive for real-world systems. Hence, we leverage practical partially homomorphic encryption schemes [5] which offer a good trade-off between efficiency, expressiveness, and security. Talos establishes a secure communication channel to the backend cloud and stores the data in encrypted form. Moreover, we introduce an encrypted data sharing mode to support secure sharing of sensitive data with individuals or third party add-on services, which is an important feature for IoT applications. We allow users to form groups for sharing their encrypted data with peers and answer community-related questions, without disclosing any secret keys to the cloud. Our revocation mechanism allows users to terminate their data sharing instantly. To protect data from unauthorized access, Talos performs all encryption and decryption operations solely on users personal devices and only stores encrypted data in the cloud.

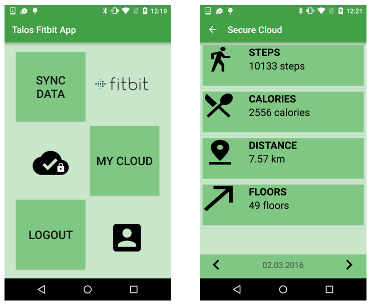

Figure 2: Screenshot of our Fitbit prototype application in Android built on top of the Talos framework. Live Demo: https://youtu.be/Wn5J3fnjpQM

2. SYSTEM OVERVIEW

Protecting data while enabling encrypted computing requires expertise that is not readily available to most application developers. We have designed Talos to be compatible with the existing IoT application ecosystem, such that application developers can integrate Talos seamlessly into their systems. To this end, the API of Talos abstracts away the mathematical complexity of the underlying cryptosystems. We realize Talos within three main components, as depicted in Figure 1.

The **client engine**'s API handles the application communication with the cloud. The client engine accommodates the logic for interaction with encrypted data, which can be omitted and reduced to only encryption in case of very constrained IoT devices. The homomorphic encryption can be either in standard mode or data sharing mode. In [3], we discuss in more details our Contiki based client engine for constrained IoT platforms (i.e., ARM-Cortex M3 microcontroller) and the benchmarks for partially homomorphic encryptions.

The **cloud engine** is designed to efficiently perform encrypted queries and operations on the cloud. Given the corresponding cryptographic authorization, the cloud engine performs the data sharing in situ. The cloud is application-agnostic and provides the basic database interface for interaction with data. In the current design of Talos, we consider structured databases. This allows defining the encrypted data computing algorithms in forms of User Defined Functions (UDF). The UDFs are added to the database and replace the default routines, without the need of recompiling the database.

The **Identity Provider** (IDP), which is a standard requirement in multi-user systems, verifies the identity-to-key pairs of users. It can be a known and trustworthy external entity or an internal unit.

3. DEMO

To show the feasibility of Talos, we integrate it into a real-world application for activity tracking. Our Android-based app (Figure 2) operates over data collected by FitBit wristbands. We use Amazon's Web Services to host the cloud components of Talos for our application demo (i.e., MySQL database, REST engine, order-preserving server logic). Fitbit does not allow third party applications to fetch data directly from their devices. To bypass this issue, our app first fetches our latest data from Fitbit cloud and then stores them on our cloud instance (i.e., AWS) through Talos.

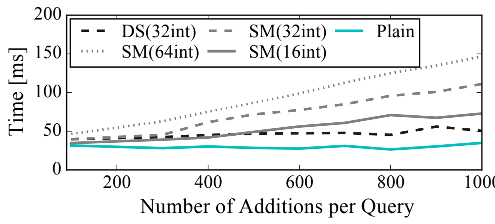

Figure 3: End-to-End latency, including an average of 30 ms network delay, neglecting the application-specific rendering. Homomorphic additions are performed on the cloud engine on Amazon for the Standard Mode (SM) and Data Sharing (DS) mode. Decryptions are performed on a Nexus 5 device.

Our application does not cache any data which allows us to study the worst-case performance while interacting with remote data. The cloud and the client communicate over https and data is encoded in a compact data representation format (i.e., JSON). For authentication, we rely on the OpenID Connect, where we currently support Google accounts as a proof-of-concept.

Figure 3 depicts the E2E latency for varying sum queries, which have the highest computational overhead compared to other queries, such as insert, delete, or range queries. For lower range sum queries, the average performance of data sharing and standard mode are close to queries over plaintext values (i.e., 0.1% and 18% higher). For larger ranges, the average latency increases by a factor of 1.5 and 3, respectively. In order to guarantee a smooth user interaction with encrypted data, the latency should be well below 1 s, which is the case even for larger ranges (i.e., 150 ms). Utilizing parallelization for the cryptographic operations on the cloud would allow a considerable performance gain, as the underlying operations are highly parallelizable.

4. ONGOING WORK

In our ongoing efforts, we are further optimizing the overheads due to data sharing and partially homomorphic encryption. Important design factors for us are efficient computations and reasonable ciphertext expansion to adhere to constrained resources of IoT platforms in terms of bandwidth, CPU, and memory. Further details on Talos and the open-source code (i.e., Contiki and Android-based client engines, Fitbit example app, and cloud engine) are available on our website: https://talos-crypto.github.io

5. REFERENCES

[1] S. Dredge. Yes, those Free Health Apps are Sharing your Data with other Companies. Guardian: theguardian.com/technology/appsblog/2013/sep/03/fitness-health-apps-sharing-data-insurance, 2013.

[2] C. Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In *ACM Symp. on Theory of Comp.*, 2009.

[3] H. Shafagh et al. Talos: Encrypted Query Processing for the Internet of Things. In *ACM SenSys*, 2015.

[4] P. Olson. For Google Fit, Your Health Data Could Be Lucrative. Forbes: forbes.com/sites/parmyolson/2014/06/26/google-fit-health-data-lucrative/, 2014.

[5] H. Shafagh. Toward Computing Over Encrypted Data in IoT Systems. In *ACM XRDS Crossroads(22:2)*, 2015.