

Protecting Private Data in Third-Party Compute Clouds

Krishna P. N. Puttaswamy, Christopher Kruegel, and Ben Y. Zhao
Computer Science Department, U. C. Santa Barbara

Introduction.¹ Third-party compute clouds such as Amazon AWS have significantly reduced the barrier to create and deploy new applications on the Internet by taking away many of the management and maintenance tasks from application developers. While it is easier to deploy applications, the applications today need to trust the clouds with all their data, including the sensitive and private data of the users. Take for example a location-based service to keep track of whereabouts of friends. In this application, users continuously upload their location information to the server hosted on the cloud. The server takes in the location of all the users and computes the distance between a user and all her friends, and sends back the results to the user. In this application, if the location history stored on the application server is leaked, the location privacy of a large number of users is compromised. With widespread use of cloud services, this threat to user privacy will only increase.

Elasticity of resources available on the cloud is a key attraction to the companies hosting their services on the cloud. This elasticity allows companies to scale up resources when the user count spikes and scale down resources when the user count goes down. But since the third-party clouds are untrusted, these companies give up on privacy. Alternatively, companies can run the service on their in-house trusted hardware. But it is expensive for the companies (referred to as *startups* here onwards) to provision the in-house hardware resources for peak load. In this paper, we develop an approach such that the startups using the clouds can benefit from both elasticity and privacy, and enable both existing and novel startups incorporate privacy into their services operating on the clouds.

System Overview. We observe that many of the applications using compute clouds combine data from a back-end database and serve the combined data to the users. Majority of the services involve only a limited amount of computation on the cloud. Our solution is ideal for such applications. The main idea behind our solution is to always store the sensitive application data (location coordinates, for instance) in encrypted form on the untrusted server, and process this data only on the trusted client devices. The key to decrypt the sensitive data is known only to the user and the startup. Since the third-party server does not have the key, it cannot leak the sensitive data. The server can, however, operate on the non-sensitive data and on the encrypted sensitive data. Any computation on the sensitive data happens on the client devices, after they are decrypted. By combining the processing on the server and the client, the users obtain the services offered by the startup, as depicted in the Figure 1. This approach is inspired by our recent work [1], but differs in that the applications need not be completely re-designed to incorporate the techniques in this paper.

Goals. We aim to develop mechanisms, and implement all of them in a publicly-available tool that the startups can use to migrate their application to our privacy-preserving model. Our tool aims to provide strong privacy guarantees to sensitive data and yet achieve the following specific goals. (i) Retain most of the application functionalities on the server, and move only a limited amount of computation to the clients. (ii) Impose minimal effort on the startup in order to migrate their application to our model.

Key Challenges. There are two major challenges we need to

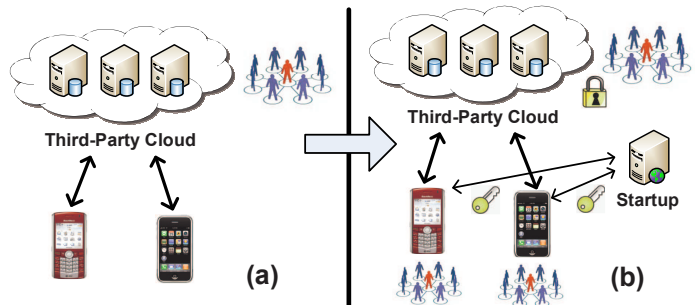


Fig. 1. (a) Currently, data is stored in plain text at the untrusted server. (b) In our approach, the application functionality is split between the server and the client. The sensitive data on the server is encrypted, but the decryption keys are known only to the startup and the clients.

address in realizing these goals. First is the key management. Our tool should identify the different types of sensitive data that the application uses, and identify the right kind of key to use for different data. For instance, data that is accessed only by a user should be encrypted with user-specific keys, while the data accessed by a group of users need to be encrypted with group keys.

Second challenge is code partitioning. Our tool should identify the part of the startup's code that is operating on the sensitive data (encrypted data), and hence needs to be moved out of the server and into the client devices. The tool also needs to provide sufficient information for the developers to effectively partition the code. For instance, it should inform which sensitive data is responsible for moving a major portion of the code to the clients, or which sensitive data leads to significant data transfer from the server to the client.

Our Approach. We briefly sketch our approach to address the challenges above. We address the key management challenge by using a combination of static and dynamic analysis of the startup's code. We analyze the data access pattern in the SQL statements to classify the data into user-specific data, group data, and global data. This analysis tells us the type of the key that different data objects in the software should be encrypted with. The developers can then tweak their service to encrypt the data with the right key.

We address the code partitioning challenge using taint tracking. We taint the sensitive data and run our tool to track all the parts of the code that uses sensitive data for computation. This process identifies the amount of code that is touching each piece of sensitive data, and then prints out statistics to describe the amount of code changes necessary to protect the privacy of each sensitive data item.

Conclusions. In this paper, we propose a novel set of techniques to incorporate privacy guarantees into applications running on third-party cloud providers. Our techniques can be applied not only to newly developed applications but also to the deployed ones, with little programmer effort. Given that this work is still in early stages, we hope to benefit from the feedback for our poster.

REFERENCES

- [1] PUTTASWAMY, K. P. N., AND ZHAO, B. Y. Preserving privacy in location-based mobile social applications. In *Hotmobile* (2010).

¹©Copyright 2010 by ACM, Inc.