IN, LCA1: Decentralized Data Sharing System based on Secure Multiparty Computation

Due on Autumn 2017

D.F, J.T-P

Max Premi

November 6, 2017

Abstract

Unlynx and Prio are two privacy-preserving data sharing systems, with each it's way to encode, decode and aggregate datas. While Unlynx uses homomorphic encryption based on Elliptic Curves and zero knowledge proofs, Prio uses Affine-aggregatable encodings and *secret-shared non-interactive proofs*(SNIP's), which perform much better in term of computation time.
However, privacy is assured in Prio if at least one client is trusted, and it then assure provides robustness and scalability, whereas Unlynx assure all this thanks to a collective authority, with several other mechanics such as Noise addition. In both case the number of servers should be significantly smaller than the number of client.
This paper presents the implementation of Prio system into unlynx, and the comparison between both system, as well with a new proof system for Unlynx based on a an efficient protocol for set membership and range proofs.

Contents

Introduction

Nowadays, tons of data are generated around of us and about us, and used to compute statistics. Even if these statistics are colleted with the goal of learning usefull aggregate informations about the users/population, it might collect and store private data from client.
The need of collecting data and sharing them in a privacy-preserving way has become crucial in this context. A lot of techniques have been develloped through the years, by major technology companies such as Google [references], but also researcher in Universities [references].

- **Put some example applications**

However, by gaining *privacy*, these protocols sacrifice *robustness* and *scalability*,

- **Explain the link between the two and why**

, leading to the use of technical agreements rather than technical solutions.
In this paper, we present two systems and the result of the merging of those.

Unlynx Actual System

- How does unlynx work (with a figure) (system of client, queries and servers)
- How Aggregation is done
- How proof is done

Prio Aggregation System

- How does Prio work (same as before)
- How does Aggregation is done
- New proof system SNIPs

Prio Proof System

- Client evaluation
- Consistency checking at the server
- Polynomial identity test
- Multiplication of shares
- Output verification

Implementation

- what is implemented in a little more detailled
- optimization apported by code from github and not aborded in details

Performance comparison

- Scaling with number of server
- Scaling with number of client
- Scaling with fairly high number of both

El Gamal range input checking

- TO be seen

Conclusion

References