INFO-F514 - Protocols, cryptanalysis and mathematical cryptology : Project Proposal

Kalala Kalonji Cedric Ramiaramanana Andriamahefa
Golhen Steven Stevens Quentin Zavaglia Mike

April 2022

1 Progress

In the proposal of our project, it was essentially divided in three parts which were in summary :

1. Implement ElGamal (EC) homomorphic with look-up tables
2. Implement and test improvements proposed in the paper in terms of space usage, speed.
3. Asses how the implementation are used in real conditions (in crypto-currencies for example).

The first part is fully implemented and working. We can use ElGamal and its homomorphic properties in encryption and decryption. We can also computes our look-up tables in advance for the decryption phase of the algorithm. Data are encoded in binary form in the table in anticipation of the second part truncation function. We keep in mind that we want something **usable in real conditions** and not just theoretically correct. Github

2 Work Distribution

Unfortunalty, Kalala Kalonji Cedric, Ramiaramanana Andriamahefa and Golhen Steven have pretty much given up on the project after the proposal was done.

Zavaglia Mike with the help of Stevens Quentin (which unfortunatly has health issues and therefor is not as involved as he wants) have done the all progress so far (as you can see on the Github repository).

Being understaffed to do the project is obviously slowing it down so we think the third part will be skipped but the second part is of course doable, just not as fast as expected.

3 Difficulties

For the first part, the main difficulty reside in the tables. We had to implement multi-processing and divided smaller sub-tables to gain both **time** and **RAM** usage. Because we use truly secure elliptic curves, the final table is expected to be around 130-150Gb (with a minimal binary representation) which is unrealistic for us to have such quantities of RAM.

Then, for speed reasons in the decryption phase, we implemented a dichotomic search in the table which means it needs to be sorted and therefor also implemented a script which merges our sub-tables in a final complete one sorted.

To do the second part, a huge work of understanding the paper had to be done. To be fair we find that important part of this paper is weird/wrong ? For example, they state that in compressed form their point takes 33 bytes to store in compressed form as our take 33 bytes in normal form. All those small misunderstandings are slowing the progression of the second part. However, now that we have a grasp on the paper, the second part should follow pretty smoothly.