Project statement: Part 1

LINFO2241

September 19, 2023

1 Introduction

This is the project statement for the LINFO2241 *Architecture and performance of computer systems* 2023-2024 course. In this project, you'll have to build a server application and learn how to measure its performance efficiently. This project is split into 5 different parts. In this first part, you only have to set up the server application based on a code we prepared for you. The next parts will tackle how to measure its performance and how to improve it.

2 Situation

You just got your master's degree from UCLouvain and you're now working for the micro-blogging social network company called *F*. On your first day, your boss Alan Mask reads on the internet that a new ciphering algorithm would change the world. He's dreaming of an encryption application running on a server, that receives plain text files and returns them encrypted. Alan Mask thinks he could charge people to use this application, and wants you to implement it.

3 Task description

3.1 Encryption algorithm

The algorithm is constituted of 2 different operations applied one after the other a fixed number of times. Once the 5 elements are extracted from the request, the file is read as a $K \times K$ squared matrix, and the algorithm encryption is performed by repeating *NbRounds* times the following steps :

- **Matrix product**: Computing the matrix product between the file and a constant matrix called *multiplication_matrix*. We called the result of this operation *product*. (See figure 1).
- **Ciphering**: We'll XOR each character of the product obtained in the previous step with the sum of each element of the key. But after summing the key, you must also apply a XOR on each element of the key with the product of current coordinates (See figure 2). For example, to encrypt $product_{ij}$, you must follow these steps :
  1. Sum all elements of the key
  2. Encryption $product_{ij}$ can be performed as $product_{ij} \bigoplus_{XOR} Sum$
  3. All elements of the key must be XORed with $i \times j$. **As the key changes at each character encryption, you must recompute its sum for each character.**

The value of K (the dimension of the file matrix), is defined as $floor(\sqrt{FileSize})$. Because of the use of the floor function (i.e. we round floats to the inferior integer), we might lose some information. It's normal, don't worry about it.

To guide you through the realisation of this algorithm, exercises are available on the INGInious platform: https://moodle.uclouvain.be/mod/lti/view.php?id=389538

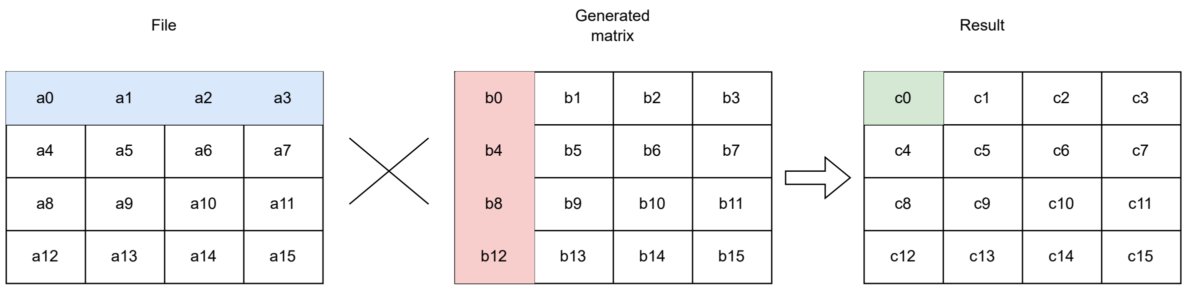
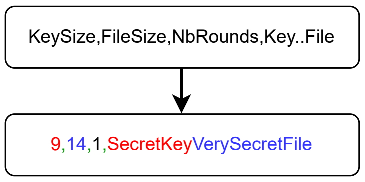


Figure 1: Line x column matrix product

3.2 Request format

You will use the HTTP protocol to communicate between the client and the server. It is a text-based protocol. In your case, the client will send a POST request to the server containing all the necessary information in the body of the request. The format of the body is the following :


KeySize,FileSize,NbRounds,Key..File
9,14,1,SecretKeyVerySecretFile

As you can see, the different components of the request are separated by a comma, except the file and the key. This is intended to allow commas in the file and the key itself.

3.3 Practicalities

The skeleton is available at `https://forge.uclouvain.be/LINFO2241/project-2023-student`. It already contains a README file that explains how to run the project. Once you're ready, you can start implementing the application by modifying the *server_implementation/main.c* file. For this project, you're only expected to implement the server application **and not the client**. The README details how to send requests manually.

4 Evaluation

In this first project, we expect a functional but not yet optimized implementation of your server. We will provide a task on INGInious which will only test the correctness of your server, not its performance. A Makefile is provided in the skeleton to make a suitable archive for INGInious. From the root of the project, you can type "make INGInious" and submit the create project.tar.gz archive.

4.1 Expected output

You should hand in your archive on the INGInious platform (accessible through Moodle) and a video report on Moodle. The deadline for both is Sunday, October 1st, 23:59. The video should contain a working scenario where you start the server, make a request and show the response on your own computer or on the machines of the Intel room in maximum one minute. Please use screen capture software, don't film with your smartphone. If you don't know how to record your screen, we recommend using OBS: `https://obsproject.com/fr`.

4.2 Evaluation criteria

- Correctness of your code verified on INGInious (3 points). We might add supplementary post-deadline tests, ensure the correctness by yourself.

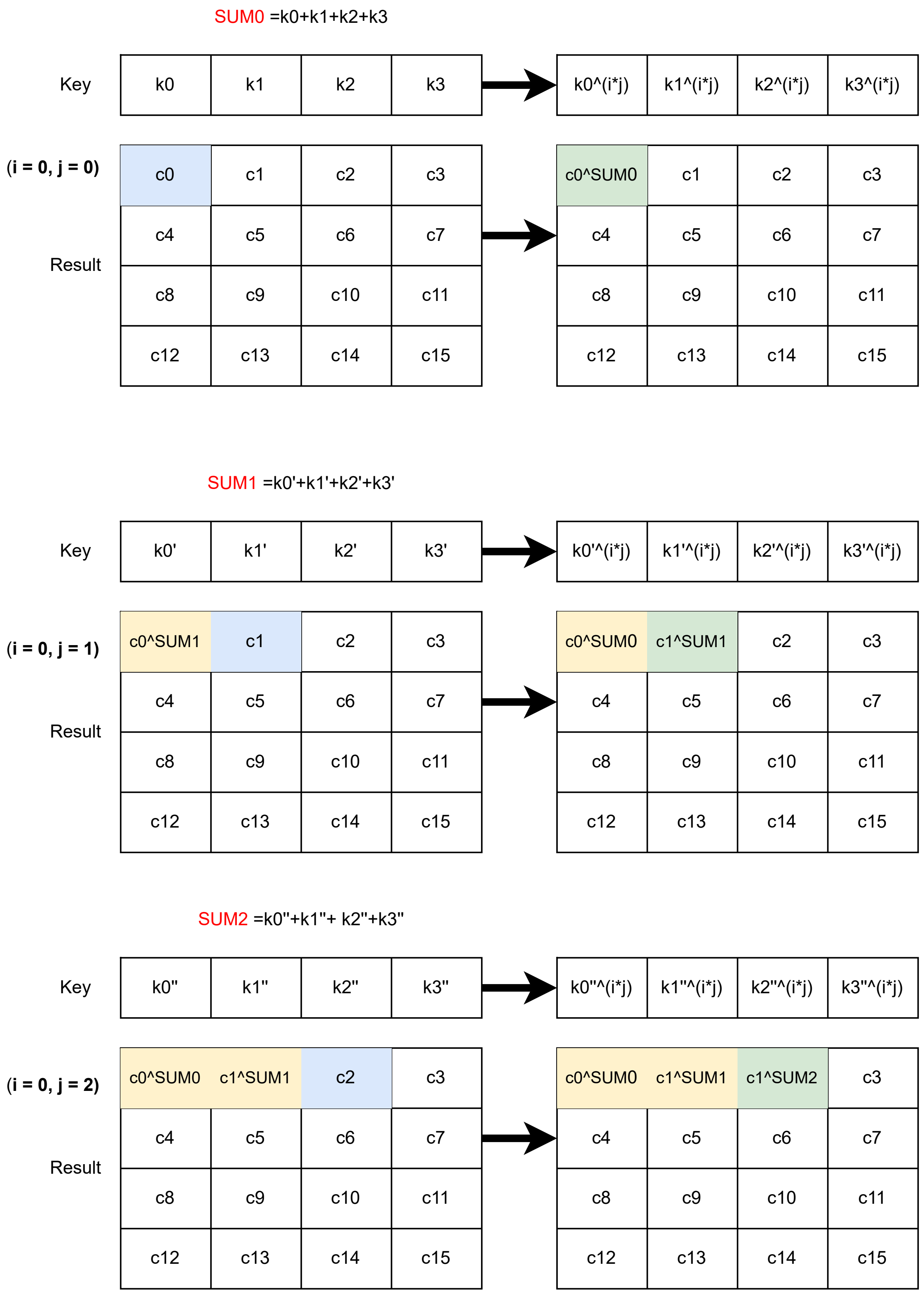

SUM0 =k0+k1+k2+k3
Key
k0
k1
k2
k3
k0^(i*j)
k1^(i*j)
k2^(i*j)
k3^(i*j)
(i = 0, j = 0)
Result
c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15
c0^SUM0 c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15
SUM1 =k0'+k1'+k2'+k3'
Key
k0' k1' k2' k3'
k0'^(i*j) k1'^(i*j) k2'^(i*j) k3'^(i*j)
(i = 0, j = 1)
Result
c0^SUM1 c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15
c0^SUM0 c1^SUM1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15
SUM2 =k0"+k1"+ k2"+k3"
Key
k0" k1" k2" k3"
k0"^(i*j) k1"^(i*j) k2"^(i*j) k3"^(i*j)
(i = 0, j = 2)
Result
c0^SUM0 c1^SUM1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15
c0^SUM0 c1^SUM1 c1^SUM2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15

Figure 2: A few iterations of the ciphering phase

- Quality of the video: does it show a working example of the server starting, the request and the response (1 point), is the video clear (1 point)?