Informe caso 2

Prototipo de sistema de rastreo de paquetes en una compañía transportadora

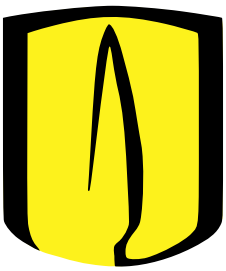
Primer semestre 2022

Autores:

Juan Andrés Méndez

201815808

Profesor: Harold Enrique Castro



Departamento de ingeniería de Sistemas & Computación

Universidad de los Andes

Bogotá - Colombia

12 de mayo de 2022

Índice

1. Descripción de la organización de los archivos en el zip **1**

2. Instrucciones para correr el prototipo incluyendo cómo correr el servidor y como correr los clientes de forma concurrente. **1**

3. Descripción del esquema que siguieron para la generación de las llaves y nombres de los archivos que las almacenan. **2**

4. Tareas **2**

4.1. Corra su programa en diferentes escenarios y mida el tiempo que el servidor requiere para cifrar el reto con cifrado simétrico y con cifrado asimétrico. Los escenarios son: 2

4.2. Análisis de resultados. 3

4.3. Identifique la velocidad de su procesador, y estime cuántos retos puede cifrar su máquina por segundo, en el caso simétrico y en el caso asimétrico. Escriba los cálculos con los que llego a ese resultado. 3

5. Conclusiones **3**

1. Descripción de la organización de los archivos en el zip

En el zip se encuentran dos partes, esta la parte auxiliar que esta hecha en python, y la parte principal que esta en java.

La de java tiene 3 componentes importantes que son: La clase Client y la clase server, el folder de resources y las clases auxiliares. Para la clase client implemente la interfaz runnable para que se comporte con una thread, esta se encarga de generar un cliente con un nombre y un id de paquete que le permite hacer una solicitud al servidor. Esta genera la llave simétrica para la comunicación.

Para la clase Server esta es la que se encarga de tener el serverSocket del servidor, tiene un arreglo con una clase paquete que guarda el id del paquete, el nombre del dueño y el estado del paquete. También implementa runnable y se puede manejar como una thread.

La clase Util, Symmetric y Asymmetric tienen los metodos necesarios para hacer el cifrado de los mensajes.

El folder de resources guarda la llave public a del servidor que fue generada al momento de correrlo, guarda un formato csv de la tabla de paquetes. Los logs de cifrado simétrico y asimétrico y las gráficas.

la parte de python simplemente es un script para general las gráficas de los tiempos del cifrado simétrico versus el asimétrico.

2. Instrucciones para correr el prototipo incluyendo cómo correr el servidor y como correr los clientes de forma concurrente.

Las instrucciones se pueden encontrar en el readme.md del archivo .zip

3. Descripción del esquema que siguieron para la generación de las llaves y nombres de los archivos que las almacenan.

4. Tareas

4.1. Corra su programa en diferentes escenarios y mida el tiempo que el servidor requiere para cifrar el reto con cifrado simétrico y con cifrado asimétrico. Los escenarios son:

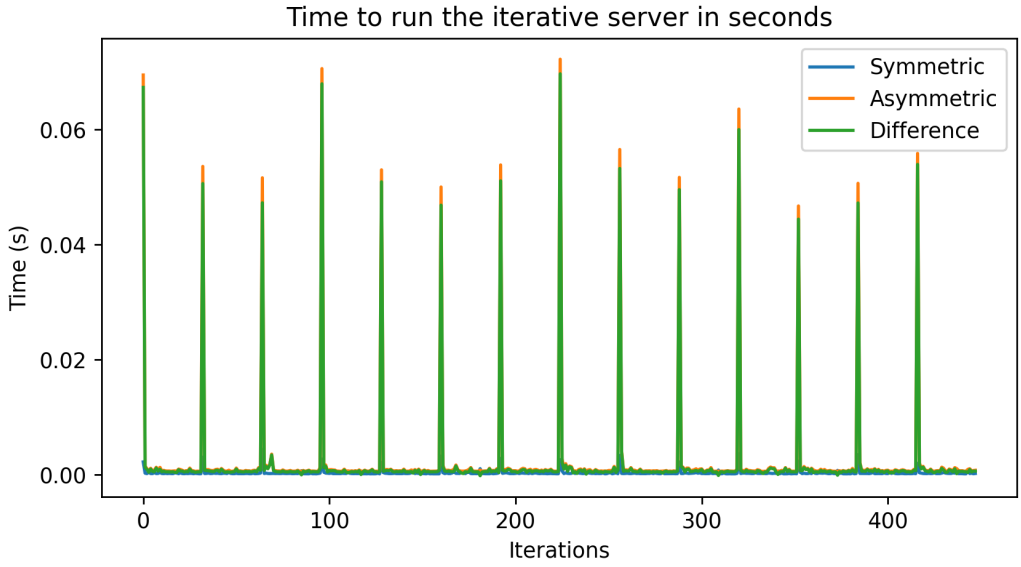


Figura 1: Tiempos de cifrado simétrico vs asimétrico

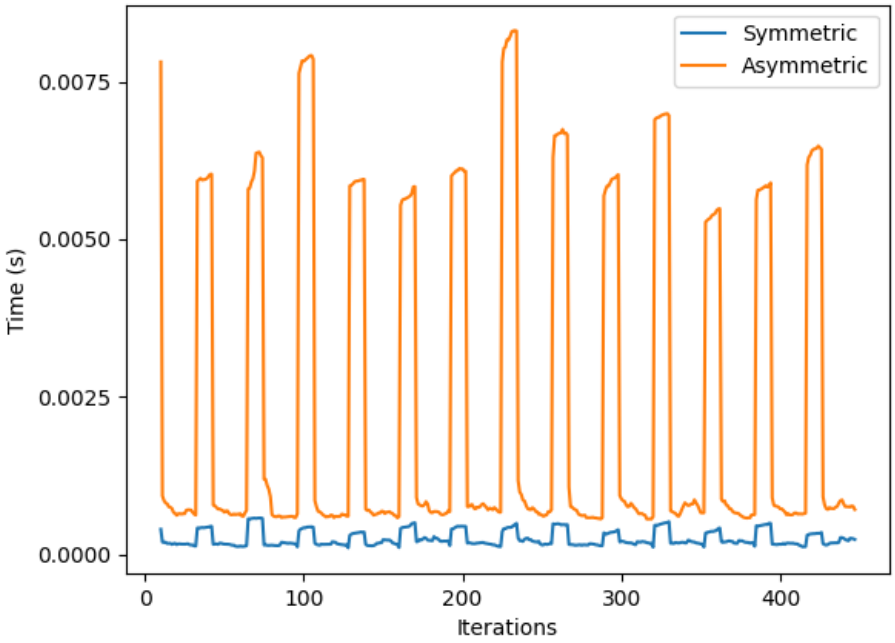


Figura 2: Tiempos de cifrado simétrico vs asimétrico filtrado

4.2. Análisis de resultados.

Se puede evidenciar que el caso asimétrico se demora mucho mas que el simétrico en todos los casos. Adicionalmente se puede ver que cada vez que comiensa una nueva instancia del servidor iterativo se genera una subida en el tiempo de cifrado de ambos, pero la que mas se ve afectada es el cifrado asimetrico.

A pesar de que no se realizo el caso con delegados se espera que los resultados no cambien ya que la operación en si es igual y no le importa si es con threads o no.

4.3. Identifique la velocidad de su procesador, y estime cuántos retos puede cifrar su máquina por segundo, en el caso simétrico y en el caso asimétrico. Escriba los cálculos con los que llego a ese resultado.

5. Conclusiones

1. Se pudo de manera exitosa seguir el protocolo y garantizar los requerimientos de transferencia para el caso de servidor iterativo.

2. se pudo ver el impacto que tiene el cifrado asimétrico en la velocidad de cifrado