Daniel Esteban Aguilera Figueroa – 202010592
Juan Esteban Arboleda Restrepo – 201921578
Juan Daniel Sepúlveda Olarte – 202113067

"Informe: Caso 3"

Realización y desarrollo del caso
Universidad de los Andes, Bogotá, Colombia
Fecha de presentación: Nov 7 de 2022

Contents

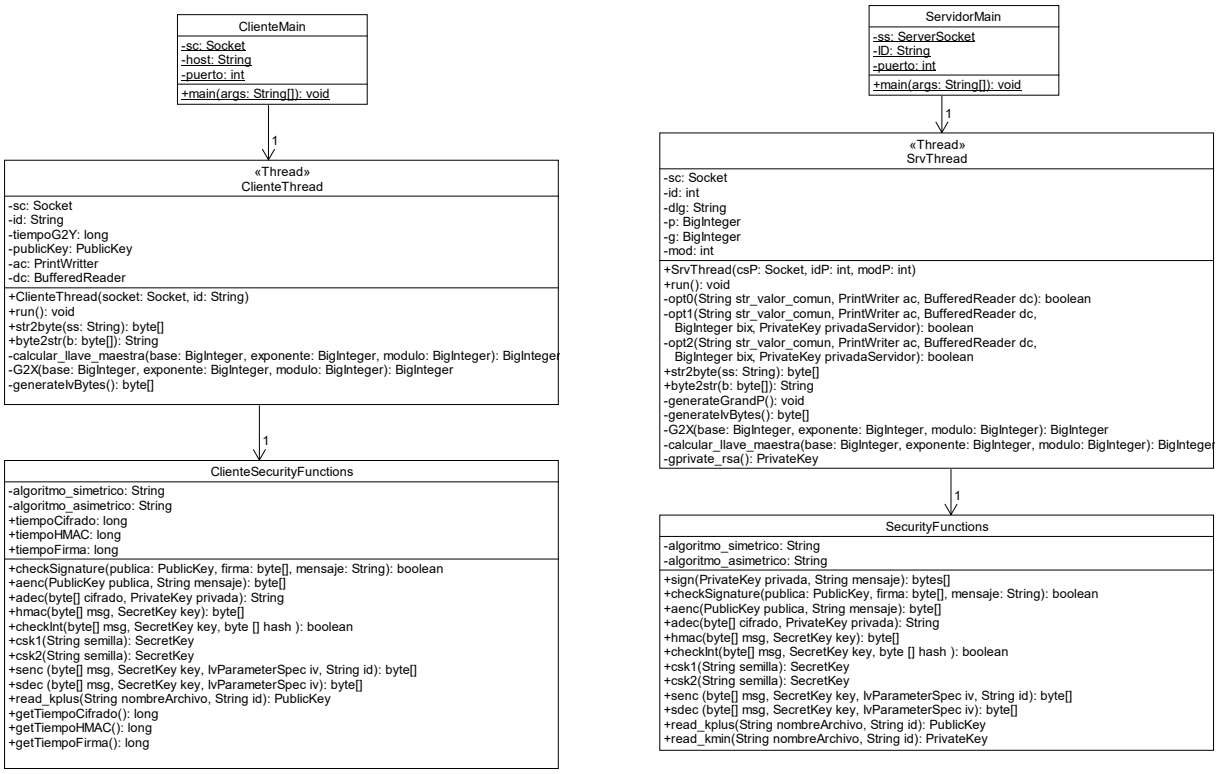
1. Introducción 1
2. UML 1
2.1. Glosario 2
3. Preguntas teóricas 2
i) En el protocolo descrito el cliente conoce la llave pública del servidor (K_w+). ¿Cuál es la manera común de enviar estas llaves para comunicaciones con servidores web? 2
ii) El protocolo Diffie-Hellman garantiza "Forward Secrecy", explique en qué consiste esta garantía. 3
4. Escenarios de análisis 3
i) Cifrar la consulta 3
ii) Generar el código de autenticación 3
iii) Verificación de la firma 4
iv) Calcular G^y 4
5. Recopilación y análisis de datos 4
6. Cálculos (procesador y sus capacidades) 5
7. Aclaraciones 7
8. Conclusión 8
9. Bibliografía 8

1. Introducción

El propósito principal de este texto es el de documentar el proceso realizado frente a la problemática de los canales seguros. Se espera demostrar correctamente la integridad y confidencialidad de la información de los clientes del problema en cuestión. Adicionalmente, se responderán dudas y preguntas respecto a la actividad.

2. UML

En este apartado del informe se analiza y modela el código fuente del caso por medio de un UML:



2.1. Glosario

Término	Descripción
ClienteMain	Clase que inicia la ejecución del proceso del cliente
ClienteThread	Clase que ejecuta las operaciones del cliente. Es un thread del proceso cliente
ClienteSecurityFunctions	Clase que ejecuta las operaciones de seguridad del cliente y lleva los tiempos de dichas operaciones
ServidorMain	Clase que inicia la ejecución del proceso del servidor
SrvThread	Clase que ejecuta las operaciones del servidor. Es un thread del proceso servidor
SecurityFunctions	Clase que ejecuta las operaciones de seguridad del servidor

3. Preguntas teóricas

i) En el protocolo descrito el cliente conoce la llave pública del servidor (K_w+). ¿Cuál es la manera común de enviar estas llaves para comunicaciones con servidores web?

- En comunicaciones para servidores web, normalmente estas llaves son enviadas por medio de entes certificadores que utilizan un esquema de distribución de llaves conocido como PKI (Public Key Infraestructure). Lo

que sucede en esta operación es que, el servidor no conoce al cliente, mismo que quiere saber si el ente con quién se va a comunicar es el servidor. Para esto, un ente certificador cifra la llave pública del servidor con su llave privada, misma que puede utilizar el cliente con la llave pública de la organización para validar que efectivamente se está comunicando con el servidor y obtener la llave pública de este.

ii) El protocolo Diffie-Hellman garantiza "Forward Secrecy", explique en qué consiste esta garantía.

- La garantía de "Forward Secrecy" consiste en el cambio de llave usada en cada sesión, es decir, una vez se mande un mensaje se usará una llave y esta llave será independiente por cada sesión que se tenga. Por ejemplo, si se manda un mensaje el día de hoy y el día de mañana un hacker decide hallar el mensaje con la llave usada previamente, será imposible para este lograr algo con esta debido a que son llaves de corto plazo para cada sesión. Adicionalmente, una característica de esta garantía es que las llaves no son guardadas y son diferentes SIEMPRE en cada sesión.

  Además, Forward Secrecy garantiza que las llaves generadas en la sesión no se vean comprometidas incluso si la llave privada del servidor se ve comprometida.

4. Escenarios de análisis

i) Cifrar la consulta

# Clientes	Tiempos obtenidos (ns)
4	52160600
16	201274400
32	445473400

ii) Generar el código de autenticación

# Clientes	Tiempos obtenidos (ns)
4	2714100
16	3323800
32	8270400

iii) Verificación de la firma

# Clientes	Tiempos obtenidos (ns)
4	5635900
16	11644300
32	30619500

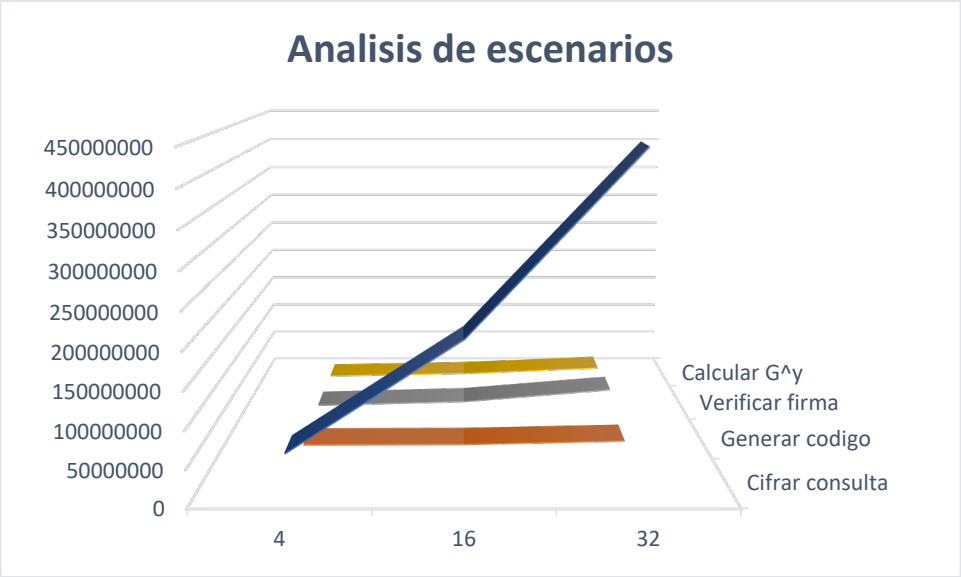
iv) Calcular G^y

# Clientes	Tiempos obtenidos (ns)
4	2457500
16	6763400
32	16361400

5. Recopilación y análisis de datos

5.1. Recopilación y grafica de datos

# Clientes	Escenarios por analizar			
	Cifrar consulta	Generar código	Verificar firma	Calcular G^y
4	52160600	2714100	5635900	2457500
16	201274400	3323800	11644300	6763400
32	445473400	8270400	30619500	16361400



5.2. Interpretación de resultados

1. La gráfica cuenta con un eje X que representan el número de clientes, un eje Y que

contiene el tipo de prueba y un eje Z con el tiempo en nanosegundos (ns).

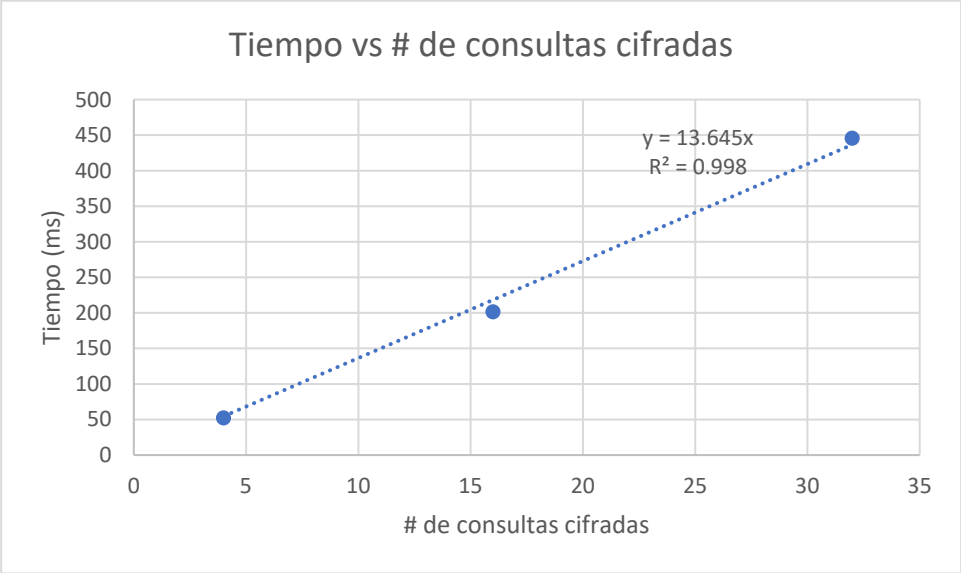
2. Se puede ver cómo el cifrado de la consulta aumenta con respecto al número de clientes. Mientras más clientes haya, más tiempo se demorará la comunicación entre el servidor y cada cliente para el cifrado de consultas. Es por esto por lo que, se puede decir que el número de clientes es proporcional al tiempo que el programa se demora en cifrar consultas.
3. El generar código y calcular G^y son casi constantes con respecto al número de clientes, por lo tanto, se puede concluir que el número de clientes no influye en la temporalidad del cálculo de estos dos datos, o al menos de una manera tan significativa como el resto de los datos.
4. El verificar firma es un caso particular, debido a que, si bien es cierto que aumenta la temporalidad del programa con respecto al número de clientes esta no aumenta de manera tan abrupta como lo es en el caso del cifrado de consulta ni tampoco es casi constante como en el caso de generar el código o calcular G^y. Por lo tanto, se puede decir que en términos de tiempo el verificar firma sí aumenta con respecto al número de clientes, pero no será tan significativo como lo puede ser con el caso de cifrar consulta.

6. Cálculos (procesador y sus capacidades)

Utilizando los cálculos recolectados y presentados en la sección "Recopilación y análisis de datos" de este informe, es posible estimar cuantas operaciones de un cierto tipo se podrían hacer por segundo, en el equipo en el que se realizaron las pruebas. A continuación, se presentan los cálculos de cuántos cifrados de consulta, códigos de autenticación, y firmas se pueden generar en un segundo.

6.1. Cifrados de consultas

Se sabe que cada cliente genera 1 solo cifrado de consulta. Por lo tanto, el número de clientes es igual al número de consultas cifradas. Así, se puede construir una regresión que permita relacionar # de consultas cifradas vs tiempo:



A partir de esta relación, se puede estimar cuantas consultas se pueden cifrar en 1000 ms:

$$T = 13{,}645x$$

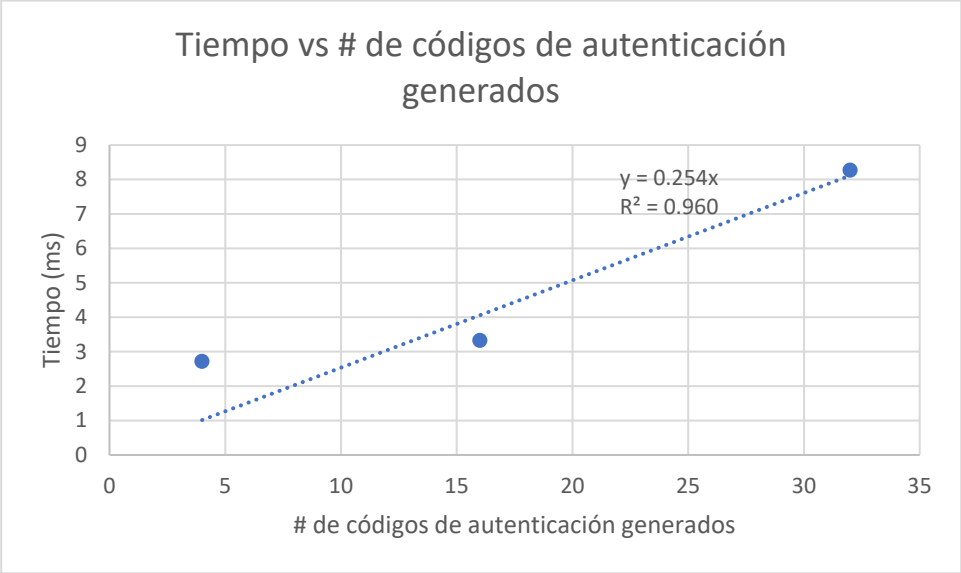
$$\frac{1000}{13{,}645} = x$$

$$x = 73{,}29$$

Así, se estima que se pueden cifrar aproximadamente 73 consultas por segundo en el equipo en el que se realizaron las pruebas.

6.2. Códigos de autenticación

Se sabe que cada cliente genera 1 código de autenticación. Por lo tanto, el número de clientes es igual al número de códigos de autenticación generados. Así, se puede construir una regresión que permita relacionar # de códigos generados vs tiempo:

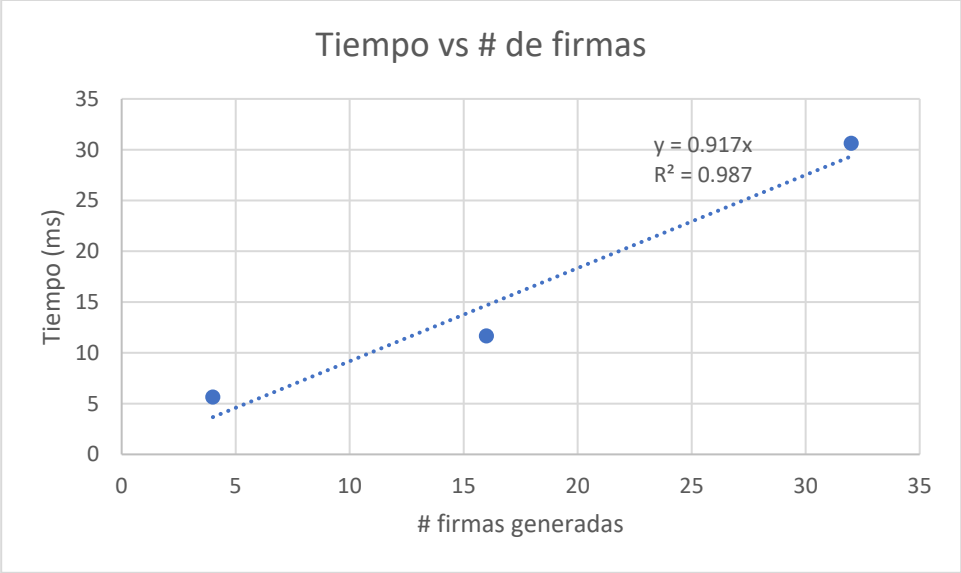


$$T = 0{,}254x$$
$$\frac{1000}{0{,}254} = x$$
$$x = 3937$$

Se obtiene entonces que se pueden generar aproximadamente 3937 códigos de autenticación en 1 segundo.

6.3 Firmas

Un cliente verifica 1 firma 1 vez. Por lo tanto, el número de clientes es igual al número de firmas verificadas. Así, se puede construir una regresión que permita relacionar # de firmas verificadas vs tiempo:



$$T = 0{,}917x = 1000$$

$$\frac{1000}{0{,}917} = x$$

$$x = 1090.5$$

Así, se tiene que se pueden verificar aproximadamente 1090 firmas cada segundo.

7. Aclaraciones

El proyecto se encuentra estructurado por medio de directorios significativos. La subdivisión esta realizada de la siguiente forma:

- En la carpeta docs se encuentra este informe
- En el directorio docs/recorded times se encuentran todos los tiempos acumulados para cada uno de los escenarios y número de clientes
- En el directorio src/seguridad20222_sevidor se encuentra el código del servidor. Este no fue modificado y es ejecutado desde el archivo .java que se encuentra en el paquete ya mencionado.
- En el directorio src/seguridad20222_cliente se encuentra el código del prototipo de cliente. Dentro de este paquete se encuentra el main del prototipo. Para ejecutar este de forma concurrente basta con indicar el número de clientes que se quieren ejecutar de esta forma. En caso de querer solamente un cliente, basta con indicar que solamente se quiere uno en la consola.
- Dentro del mismo paquete mencionado anteriormente, se encuentra el código del thread para un análisis adicional si se requiere por parte de los calificadores.

- Para la consulta a realizar por parte del cliente, esta es calculada automáticamente usando un numero aleatorio para fácil análisis a la hora de ejecutar el programa de forma concurrente.

8. Conclusión

En esta simulación sobre la comunicación entre un servidor y varios clientes se ha podido obserar qué fragmentos de la comunicación son los que tienen mayor costo temporal y cuáles resultan ser casi constantes. El número de clientes tiene una mayor influencia cuando se intenta cifrar una consulta, lo cuál tiene sentido, debido a que, el número de consultas será mayor mientras más clientes se estén comunicando con el servidor. Además, se ha logrado demostrar cómo es que el calcular el G^y y el generar códigos resultan ser casi constantes independientemente del número de clientes que la comunicación tenga. Más aún, se ha podido ver cómo la verificación de firmas resulta aumentar en costo temporal mientras más clientes haya pero no de la misma manera como lo hace el cifrado de consultas, misma que resulta ser la operación con mayor complejidad temporal en la comunicación entre clientes y servidores.

9. Bibliografía

1. https://crypto.stackexchange.com/questions/66202/what-is-perfect-forward-secrecy
2. https://www.extrahop.com/company/blog/2017/what-is-perfect-forward-secrecy/#:~:text=Perfect%20Forward%20Secrecy%20(PFS)%20is,cyber%20security%20Cone%20of%20Silence.
3. https://www.zwilnik.com/security-and-privacy/diffie-hellman-and-forward-secrecy/
4. https://bloqueneon.uniandes.edu.co//content/enforced/134478-202220_ISIS2203_02/notas-clase/lecturas-seguridad.pdf?isCourseFile=true&_&d2lSessionVal=gD3Gcj9XHcSfrDrZQsngcyRc5&ou=134478