Informe Caso 3

Infraestructura computacional

Universidad de los Andes

Juan Andrés Reyes - 202210295

Juan Diego Sánchez - 202214625

Santiago Tinjacá - 202215991

La entrega realizada para dar respuesta al caso 3 contiene las siguientes clases:

CifradoAsimetrico: Se encarga de cifrar y descifrar con modo de encriptación o decodificación asimétrica.

CifradoSimetrico: Se encarga de cifrar y descifrar con modo de encriptación o decodificación simétrica. También se miden los tiempos en las instrucciones que medimos.

Cliente: Tiene el main() del lado de clientes. Se encarga de enviar la cantidad de clientes dada por consola en forma de threads hacia el servidor. Le da un socket a cada cliente e inicia el protocolo de cliente.

DigestCalculator: Se encarga de calcular dos tipos de Hash. En primer lugar, se hashea un arreglo de Bytes con SHA-512. El otro algoritmo hashea un texto dada una llave con HMACSHA-256.

ProtocoloCliente: Ejecuta los pasos comúnicandose con el servidor. Su atributo outputLine es aquello que está enviando al servidor e inputLine es la información que recibe del servidor.

ProtocoloServidor: El protocolo se encarga de manejar las peticiones, responder y realizar las verificaciones necesarias para la comunicación con el cliente.

Servidor: Tiene el main() de lado del servidor, se inicializa con una llave privada y una pública. Se encarga de recibir las peticiones de cada cliente y dar respuesta a cada una de ellas. Se agrega un socket por cada cliente que llega.

Instrucciones para ejecutar el servidor y el cliente:

> Se debe ejecutar Servidor.java y Cliente.java ya sea por consola o por medio del IDE. Al iniciar Cliente.java, se le preguntará por el número de clientes. La respuesta se escribe en consola e inicia la comunicación entre el Servidor y los Clientes. Se ponen los clientes de cada cliente en doc/dataCliente y para guardar los datos del servidor, se debe terminar ese proceso (ej. Ctrl+C en terminal o detener proceso).

Preguntas:

(i) En el protocolo descrito el cliente conoce la llave pública del servidor (K_w+). ¿Cuál es el método comúnmente usado para obtener estas llaves públicas para comunicarse con servidores web?

R/ Usualmente se utiliza una entidad encargada de generar certificados para validar que las llaves pertenezcan a la entidad que dice ser la dueña. Estos se encargan de tener un repositorio de certificados que contienen información que permite verificar la identidad de la entidad. Este certificado incluye información de la organización, su llave pública, el nombre de la entidad emisora de certificados y la firma digital del emisor del certificado.

(ii) ¿Por qué es necesario cifrar G y P con la llave privada?

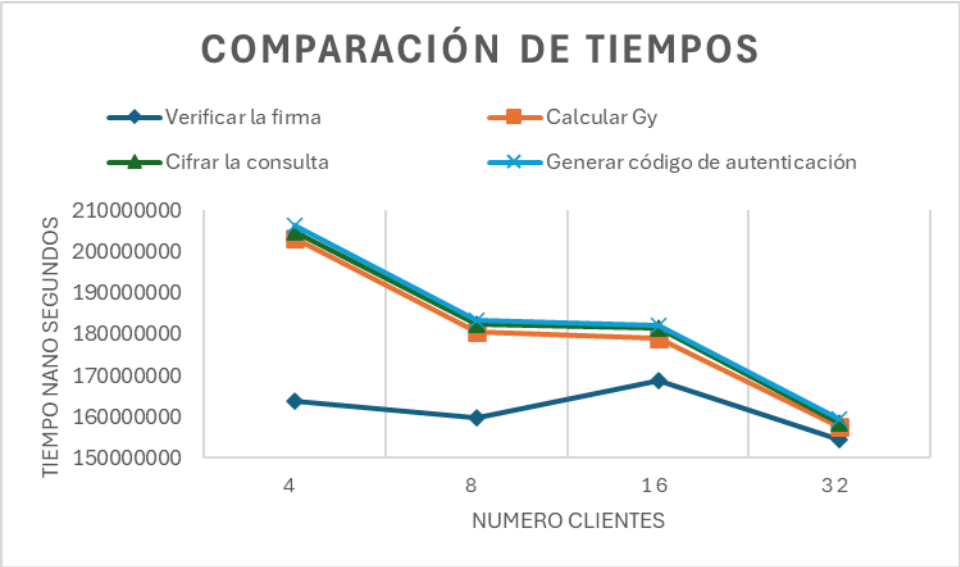
R/ Es necesario cifrar Gy P con la llave privada ya que así el cliente puede descifrarla con la llave pública. Adicionalmente, de esta forma se aseguran la integridad de los datos y se protegen de posibles cambios que se puedan realizar a los datos por personas malintencionadas. Ya que el cliente se asegura de que los datos recibidos no sufren de ninguna alteración por parte de un tercero. Al cuál le quedaría muy difícil modificar el cifrado de forma que concuerde con los datos enviados sin cifrar.

(iii) El protocolo Diffie-Hellman garantiza "Forward Secrecy", presente un caso en el contexto del sistema Banner de la Universidad donde sería útil tener esta garantía, justifique su respuesta (por qué es útil en ese caso).

R/ El protocolo Diffie-Hellman es útil para el caso del sistema Banner de la Universidad debido a que garantiza que, aunque una llave anterior se vea comprometida, el resto de claves no sean descubiertas. Esto se debe a que "Forward Secrecy" crea nuevas llaves que varían de la conexión realizada. De esta forma, en el caso de que la universidad se logre descubrir la llave en una conexión pasada, esta sea inútil para nuevas conexiones, lo que impide la suplantación. Más en un caso de información sensible como lo es Banner.

Datos Clientes

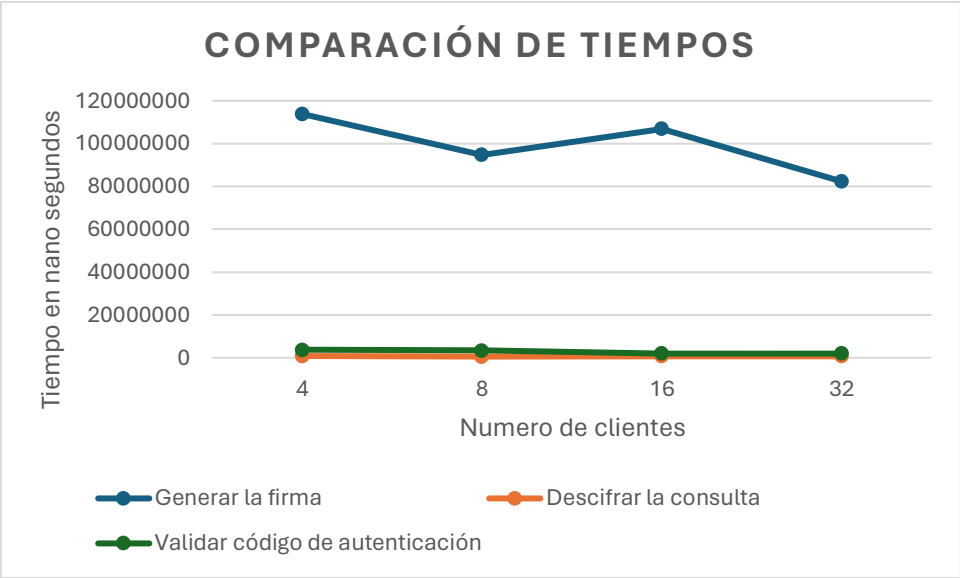
# clientes \ tiempo(ns)	Verificar la firma	Calcular Gy	Cifrar la consulta	Generar código de autenticación
4	163654100	39587450	1617775	1498650
8	159773525	20899925	1699862,5	1114175
16	168685619	10461200	2225893,8	874856,25
32	154491534	3139875	856146,88	832412,5



En la gráfica se puede ver que el número de clientes simultáneos en el servidor no son suficientes para generar un gran cambio en los tiempos de ejecución. Sin embargo, se pueden ver cómo hay tareas que demoran más que otras. Verificar la firma del servidor es lo que menos tiempo toma en todos los casos, mientras que Calcular Gy y cifrar consultas son los procesos más costosos.

Datos Servidor

# clientes \ tiempo(ns)	Generar la firma	Descifrar la consulta	Validar código de autenticación
4	113857150	807325	3752175
8	94795000	543425	3387612,5
16	106957056	758587,5	1932137,5
32	82368690,6	697418,75	1991981,3



Nuevamente se pude evidenciar cómo el número de clientes no impacta de forma significativa el estado del sistema. Sin embargo, se puede ver como la tarea más costosa en tiempo para el servidor es generar la firma que consiste en cifrar con su llave privada el reto para conseguir autenticación.

6) Identifique la velocidad de su procesador, y estime cuántas consultas puede cifrar su máquina, cuántos códigos de autenticación puede calcular y cuántas verificaciones de firma, por segundo. Escriba todos sus cálculos.

Velocidad Base: 2.8 GHz

2225893,8

Referencias

Claves privadas, claves públicas y certificados digitales

https://www.ibm.com/docs/es/sia?topic=osdc-private-keys-public-keys-digital-certificates-7

Perfect forward secrecy

https://www.ibm.com/docs/en/sss/3.1.1?topic=reference-perfect-forward-secrecy