Universidad de los Andes
Ingeniería de Sistemas y Computación
ISIS 2203 Infraestructura Computacional
Semestre 2022-20

Caso de Estudio 3 – Canales Seguros

Objetivos

- Comprender ventajas y limitaciones de los algoritmos para proteger datos (confidencialidad e integridad).
- Construir un prototipo a escala de una herramienta para soportar confidencialidad e integridad.

Problemática:

Supondremos que una compañía ofrece a sus clientes la posibilidad de hacer consultas en línea, garantizando confidencialidad e integridad de la información. Su tarea consiste en construir un programa cliente que envía solicitudes al servidor de la compañía y verifica la integridad de los resultados recibidos de acuerdo con el protocolo de comunicación establecido. A continuación, se presenta información adicional del servidor y el cliente.

- El servidor es un proceso que recibe y responde las consultas de los clientes. Para simplificar esta parte del problema y concentrarnos en los aspectos de seguridad, la consulta solo será un número y el servidor debe responder con número+1. El código fuente del servidor será entregado por los profesores del curso.
- El cliente es un proceso que envía una consulta al servidor, espera la respuesta y la presenta al usuario.

Tanto servidor como cliente deben cumplir con las siguientes condiciones:

- Las comunicaciones entre cliente-servidor deben usar sockets y deben estar cifradas de acuerdo con el protocolo definido.
- Desarrolle en Java. No use librerías especiales, solo las librerías estándar (java.security, javax.crypto).
- Generaremos por adelantado las llaves pública y privada del servidor. Tenga en cuenta que en una instalación real la llave pública puede ser leída por cualquiera, pero la llave privada solo debería estar al alcance del propietario.

PARA TENER EN CUENTA:

- Como algoritmos usaremos:
  - Cifrar - C(..) en la figura 1: AES. Modo CBC, esquema de relleno PKCS5Padding, llave de 256 bits.
  - Firma - F(..) en la figura 1. SHA256withRSA
  - Código de autenticación - HMAC(..) en la figura 1: HMACSHA256

- Para generar las llaves necesarias:
  - Primero calcule una llave maestra por medio de Diffie-Hellman
  - Necesitará usar la clase BigInteger de Java y los números primos deben tener 1024 bits de longitud
  - Luego use la llave maestra para calcular un digest con SHA-512
  - Parta el digest en dos mitades para generar la llave para cifrado y la llave para código de autenticación: usar los primeros 256 bits se deben usar para construir la llave para cifrar, y los últimos 256 bits para construir la llave para generar el código HMAC

El cliente y el servidor deben comunicarse siguiendo el protocolo que se describe en la figura 1:

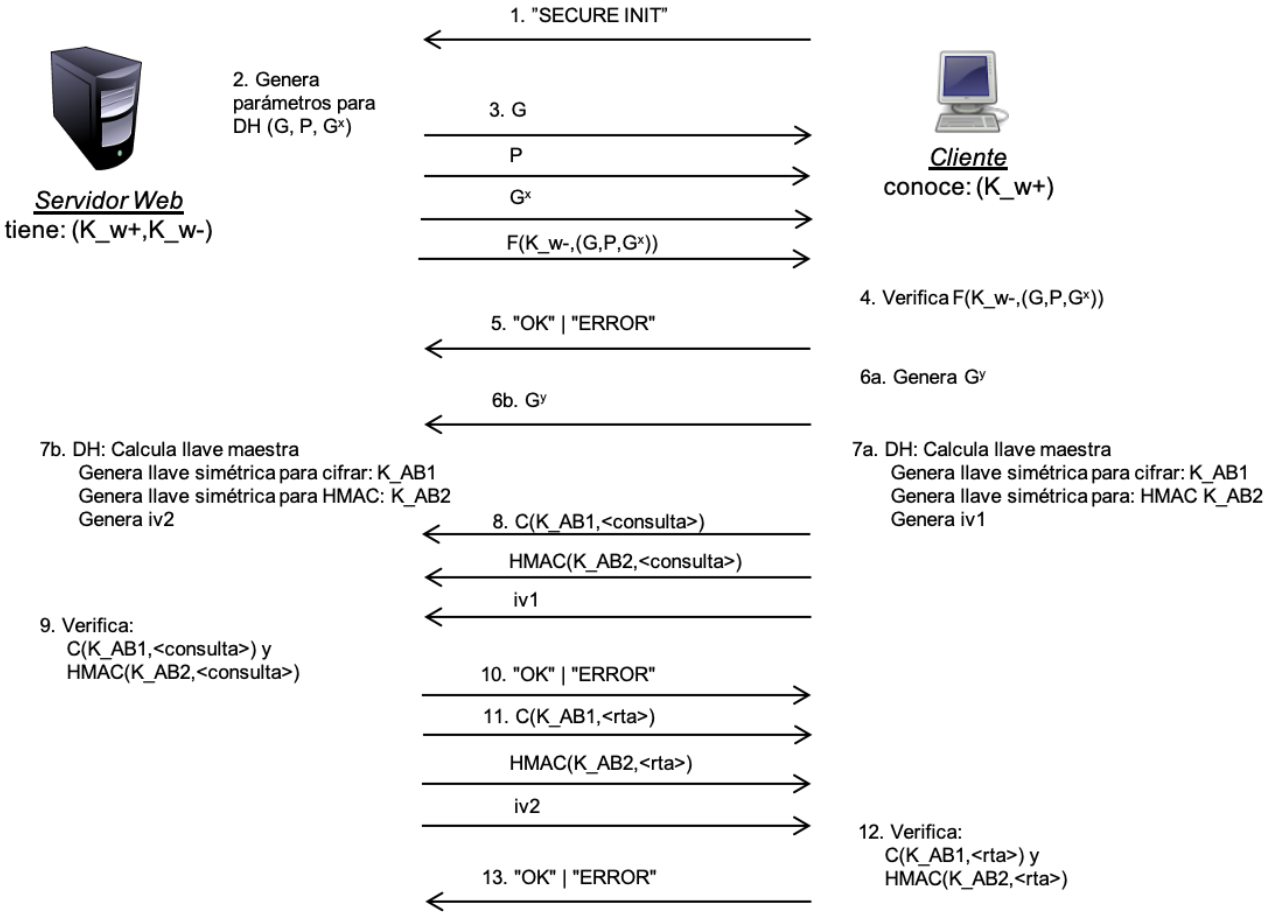

Figura 1. Protocolo de comunicación entre el servidor y el cliente.

Si el cliente encuentra un error en la validación debe enviar el mensaje "ERROR" al servidor y terminar (ver pasos 5 y 13 en la figura). Así mismo, si el servidor encuentra un error en la validación enviará el mensaje "ERROR" al cliente y terminará la conexión (paso 10).

Adicionalmente, resuelva lo siguiente:

1. Responda las siguientes preguntas:
   (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?
   (ii) El protocolo Diffie-Hellman garantiza "Forward Secrecy", explique en qué consiste esta garantía.
2. Corra su programa en diferentes escenarios y mida los tiempos que el cliente demora para:
   (i) Cifrar la consulta
   (ii) Genere el código de autenticación
   (iii) Verificación de la firma
   (iv) Calcular G^y

   Los escenarios de interés son: el servidor y el cliente deben implementar delegados concurrentes. El número de clientes delegados debe variar entre 4, 16 y 32.
3. Construya una tabla con los datos recopilados (tenga en cuenta que necesitará correr cada escenario en más de una ocasión para validar los resultados) y luego construya una gráfica con los datos de la tabla.
4. Escriba sus comentarios sobre los resultados.
5. 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.

Entrega:

- Esta entrega vale 85% de la calificación del caso (el otro 15% corresponde a la parte 1)
- Cada grupo debe entregar un zip con:
    - archivos con la implementación del prototipo del cliente.
    - un subdirectorio docs con un informe que incluya: (i) la descripción de la organización de los archivos en el zip, (ii) las instrucciones para correr el prototipo del cliente, incluyendo cómo correr los clientes de forma concurrente, (iii) las respuestas a las tareas 1 a 5.
    - **Al comienzo del informe, deben estar los nombres y carnés de los integrantes del grupo**. Si un integrante no aparece en el documento entregado, el grupo podrá informarlo posteriormente. Sin embargo, habrá una penalización: la calificación asignada será distribuida (dividida de forma equitativa) entre los integrantes del grupo.
- Recuerde incluir en su informe **todas las referencias** que use para resolver este proyecto.
- El trabajo se realiza en grupos de **3** personas. No debe haber consultas entre grupos; con una excepción: los grupos emparejados de UCSC y de los Andes pueden reunirse para resolver dudas, pero cada uno debe desarrollar su propio código cliente.
- El grupo responde solidariamente por el contenido de todo el trabajo, y lo elabora conjuntamente (no es trabajo en grupo repartirse puntos o trabajos diferentes). Se puede solicitar una sustentación a cualquier miembro del grupo sobre cualquier parte del trabajo. Dicha sustentación será parte de la calificación de todos los miembros. Tenga en cuenta que desarrollar sus habilidades para trabajo en grupo le puede ayudar en su futuro profesional para integrarse más fácilmente a un grupo de trabajo y trabajar de forma productiva.
- Habrá una coevaluación del trabajo en grupo. Cada miembro del grupo evaluará a sus dos compañeros y las notas recibidas por un estudiante ponderarán la nota final de caso para ese estudiante. Si un grupo se disuelve, la calificación del caso se recalculará así:
    - Si la disolución se reporta al profesor durante los últimos tres días hábiles antes de la fecha límite de entrega, entonces la calificación de la entrega se multiplicará por 0,5 (50%)
    - Si la disolución se reporta, se justifica de forma apropiada, y el profesor acepta la disolución con anterioridad a los últimos tres días hábiles antes de la fecha límite de entrega, entonces la calificación de la entrega se multiplicará por 0,75 (75%)
- En el parcial se incluirá una pregunta sobre el desarrollo de alguna de las funcionalidades del caso
- El proyecto debe ser entregado en bloqueneon.
- **La fecha límite de entrega es noviembre 7, 2022 a las 23:50 p.m.**

Referencias:

- *Cryptography and network security*, W. Stallings, Ed. Prentice Hall, 2003.
- *Computer Networks*. Andrew S. Tanenbaum. Cuarta edición. Prentice Hall 2003, Caps 7, 8.
- *Blowfish*. Página oficial es: http://www.schneier.com/blowfish.html
- *RSA.* Puede encontrar más información en: http://www.rsa.com/rsalabs/node.asp?id=2125
- *CD X509*. Puede encontrar la especificación en: http://tools.ietf.org/rfc/rfc5280.txt
- *MD5*. Puede encontrar la especificación en : http://www.ietf.org/rfc/rfc1321.txt