Universidad de los Andes
Ingeniería de Sistemas y Computación
ISIS 2203 Infraestructura Computacional

Caso de Estudio 2 – Canales Seguros
Sistema de Apoyo a las Aplicaciones Misionales de una Entidad Oficial

Objetivos

- Identificar los requerimientos de seguridad en el sistema de apoyo a las aplicaciones de la entidad administradora de pensiones.
- Construir un prototipo a escala del sistema que permita satisfacer algunos de los requerimientos de seguridad identificados, entendiendo las garantías de seguridad y las limitaciones de la implementación propuesta.

Problemática:

Como se indicó en el documento que describe el contexto del caso, las principales tareas del sistema son Afiliación, Recaudo, AFE, Historia laboral, Nómina de pensionados, tutelas y portal web.
En este contexto, surgen diferentes problemas de seguridad para algunas de las transacciones que el sistema soporta, tanto a nivel de transmisión, como en procesamiento y almacenaje de datos. Como consecuencia, es necesario evaluar riesgos y determinar medidas para mitigar los problemas detectados. Su tarea en este caso es actuar como consultor de seguridad y analizar la seguridad de las tareas relacionadas con el portal web.

Tareas:

Suponga que el portal web permite a los afiliados consultar su información en línea y solicitar reportes de su estado de cuenta. El portal corre una aplicación en un servidor en la oficina principal, que cuenta con un backup de procesamiento y almacenamiento. Los afiliados de la entidad usan su navegador para conectarse con la aplicación principal y deben autenticarse para tener acceso. Las comunicaciones se dan vía internet.

Los computadores de escritorio de los empleados que trabajan en la oficina principal de la entidad están conectados a una subred interna separada de la subred en la que corre el servidor que corre el portal web. El servidor del portal maneja dos interfaces de red, una para conexiones con los clientes externos, entre los que se cuentan los afiliados, y una para conexiones con los computadores de la red interna. La oficina principal cuenta además con un firewall configurado apropiadamente para proteger la subred interna y la subred del servidor web. Suponga que la arquitectura del sistema incluye un servidor diferente para el manejo administrativo y operacional de la entidad.

A. [15%] Análisis y Entendimiento del Problema

Considerando el sistema descrito en el párrafo anterior:

1. Identifique los datos que maneja el portal web y que deben ser protegidos. Explique su respuesta en cada caso y responda la pregunta Si un actor no autorizado consigue acceso al dato mencionado, ya sea en modo lectura o escritura, ¿cómo podría afectar a la entidad?
2. Identifique cuatro vulnerabilidades de este sistema, teniendo en cuenta únicamente aspectos técnicos o de procesos (no organizacionales). Identifique vulnerabilidades no solo en lo relacionado con la comunicación sino también con el almacenamiento y procesamiento de los datos. Explique su respuesta en cada caso.

(*) *Sus explicaciones DEBEN corresponder al contexto planteado. NO escriba respuestas genéricas.*

B. [85%] Implementación del Prototipo

En esta parte del proyecto nos centraremos únicamente en el sistema de intercambio de información con el portal web. Un afiliado enviará su número de cédula y clave, y el servidor responderá con el monto del ahorro de ese afiliado. Usted solo debe construir la aplicación cliente; este cliente debe comunicarse con el servidor que se le entregará.

Como queremos concentrarnos en el protocolo de comunicaciones y sus requerimientos de seguridad, construiremos una aplicación cliente/servidor simplificada en Java, sin usar el protocolo https, que es común para asegurar las comunicaciones. El cliente y el servidor se comunicarán siguiendo el protocolo descrito en la Figura 1.

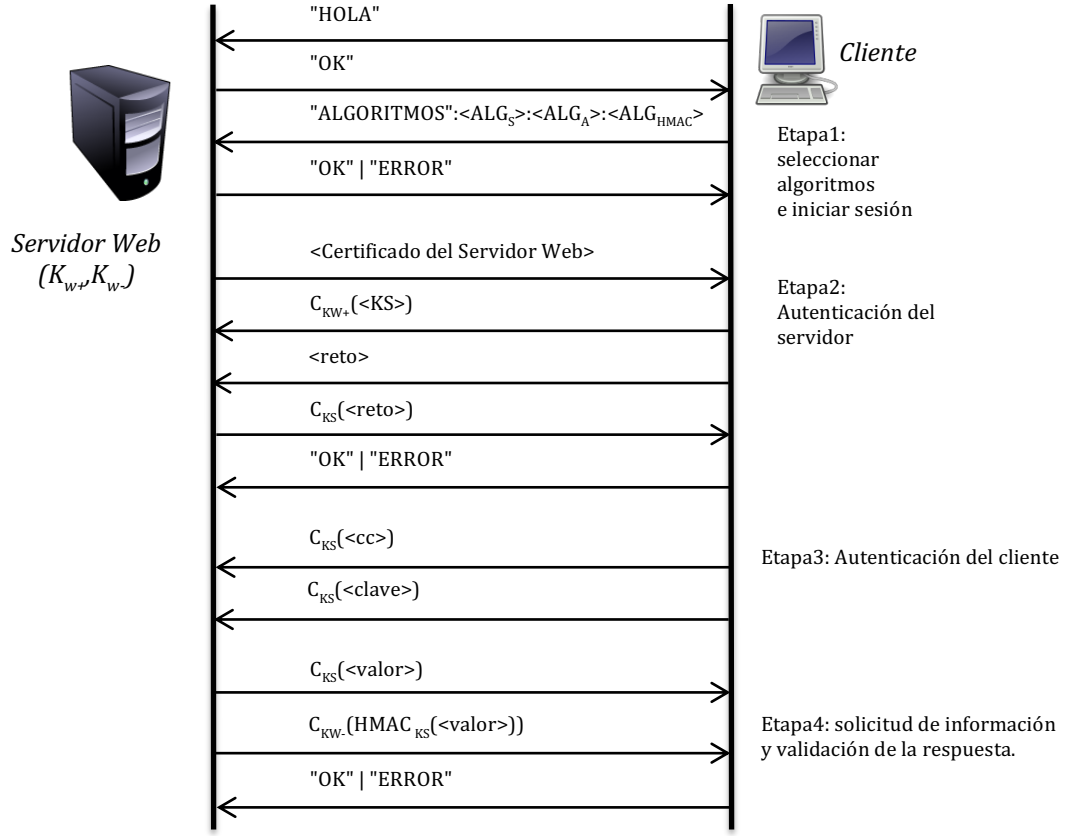

Figura 1. Protocolo de comunicación entre cliente y servidor.

TENGA EN CUENTA:

- El protocolo de comunicación maneja:
  - Cadenas de Control: "HOLA", "ALGORITMOS", "OK", "ERROR".
  - Separador Principal: " : "
- A continuación, se presentan los algoritmos disponibles en el servidor para manejo de integridad y confidencialidad. Es decir, usted debe seleccionar un algoritmo para reemplazar cada una de las cadenas: <ALGS>, <*ALGA*> y <*ALGHMAC*> en el protocolo. Los algoritmos disponibles son:
  - Simétricos (ALGS):
    - AES (Configuración por defecto: Modo ECB, esquema de relleno PKCS5, llave de 128 bits).
    - Blowfish (Configuración por defecto: Cifrado por bloques, llave de 128 bits).
  - Asimétricos (ALGA):
    - RSA. (Configuración por defecto: Llave de 1024 bits.)
  - HMAC (ALGHMAC):
    - HmacSHA1
    - HmacSHA256
    - HmacSHA384
    - HmacSHA512

  Las cadenas que identifican cada uno de los algoritmos son: "AES", "BLOWFISH", "RSA", "HMACSHA1", "HMACSHA256", "HMACSHA384", "HMACSHA512".

- El servidor usa el estándar X509 para su certificado digital (CD). La idea es que el cliente compruebe la identidad del servidor a partir de un CD (en un caso real este debería ser expedido por una entidad certificadora pero aquí se va a generar localmente). El CD debe contener, entre otros, la llave pública para usarla en el proceso de comunicación.
- El servidor usa la librería BouncyCastle para el manejo de certificados.
- La comunicación se realiza a través de sockets de acuerdo con el protocolo de comunicación definido en la Figura 1.
- Las cadenas entre los caracteres "<" y ">" debe reemplazarse por los valores correspondientes (cc, clave, etc).
- Para traducir de String a byte[] y viceversa el servidor maneja codificación en base 64. Use DatatypeConverter para construir los métodos apropiados:
```
DatatypeConverter.printBase64Binary(byteArray);
DatatypeConverter.parseBase64Binary(cadena);
```
- Para usar el método `parseBase64Binary(cadena)`, la longitud de la cadena debe ser múltiplo de 4. En particular, los valores que el cliente envía al servidor (reto, cc y clave) deben tener una longitud múltiplo de 4. Puede completar la longitud con 0 a la izquierda o a la derecha.
- El servidor siempre envía los datos en formato String y los recibe en el mismo formato:
```
socketParaEnviar.println(cadenaConInformacion);
cadenaConInformacion = socketParaRecibir.readLine();
```
- A continuación, se ilustra el procedimiento que usa el servidor para enviar el certificado.
```
java.security.cert.X509Certificate certificado = generarCertificado(llaves);
byte[] certificadoEnBytes = certificado.getEncoded( );
String certificadoEnString = printBase64Binary(certificadoEnBytes);
socketParaComunicacion.println(certificadoEnString);
```
- El servidor recibe cc y clave, pero esta información no es verificada en una base de datos, es usada para generar el valor asociado. (Para verificar, el protocolo tendría que implementar una fase para registrar un usuario).
- El cliente debe validar la información que recibe del servidor en las etapas 2 y 4.
- Usted solo debe desarrollar el cliente. La librería BouncyCastle y el .jar del servidor estarán disponibles en SICUA+.

Entrega:

- Cada grupo debe entregar un zip de un proyecto Java con: La implementación correspondiente al cliente (descrito en la parte B). En el subdirectorio docs debe haber un archivo que incluya el informe con las respuestas a la parte A. **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.
- El trabajo se realiza en grupos de **2** personas. No debe haber consultas entre grupos.
- 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.
- El proyecto debe ser entregado por Sicua+ por uno solo de los integrantes del grupo.
- **La fecha límite de entrega es el 24 de octubre, 2019 a las 23:55 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