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

Caso de Estudio 2 – Canales Seguros
Logística y Seguridad Aeroportuaria

Objetivos

- Identificar los requerimientos de seguridad de los canales usados para transmisión de la información en el sistema de Logística y Seguridad Aeroportuaria.
- 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 enunciado del caso, el sistema cuenta con varias aplicaciones: Novasoft financiero en línea y fuera de línea, OpenERP, Sistema Time & Attendance, correo electrónico y página web. En este contexto, surgen diversos 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, considerando solo aspectos de seguridad, las tareas relacionadas con la aplicación Novasoft financiero en línea.

Tareas:

Suponga que el sistema tiene tres servidores, uno soporta la aplicación financiera, el segundo soporta el sistema Time & Attendance y correo electrónico y el tercero soporta la página web.

- Todos los servidores implementan control de acceso a nivel del sistema operativo y ejecutan transacciones solo para usuarios autenticados, de acuerdo con los permisos asignados.
- Las aplicaciones también manejan usuarios, cada una maneja su propio archivo de configuración de usuarios y soporta operaciones de cifrado con una librería que implementa algoritmos propios.

A. [20%] Análisis y Entendimiento del Problema.

En el sistema descrito en el párrafo anterior:

1. Identifique los datos que deben ser protegidos por la aplicación Novasoft financiero en línea. Justifique su respuesta (para cada dato responda la pregunta ¿ Si un actor no autorizado consigue acceso al dato mencionado, cómo podría afectar la empresa?
2. Identifique cuatro vulnerabilidades del sistema, teniendo en cuenta únicamente aspectos técnicos (no organizacionales o de procesos). Identifique vulnerabilidades no solo en lo relacionado con la comunicación sino también con el almacenamiento. Explique su respuesta en cada caso.

Nota: *Sus explicaciones DEBEN estar relacionadas con el contexto del problema planteado (las justificaciones indicarán cómo). NO se aceptarán respuestas genéricas*.

B. [10%] Propuesta de Soluciones.

Para cada una de las vulnerabilidades que usted identificó en el punto anterior, proponga mecanismos de resolución/mitigación. Justifique brevemente por qué el mecanismo propuesto resuelve la vulnerabilidad.

En sus justificaciones tenga en cuenta aspectos relacionados con eficacia, costo, eficiencia, flexibilidad, aspectos de implementación, y otros aspectos técnicos que considere convenientes.

C. [70%] Implementación del Prototipo.

En esta parte del proyecto nos centraremos únicamente en la aplicación Novasoft financiero en línea.

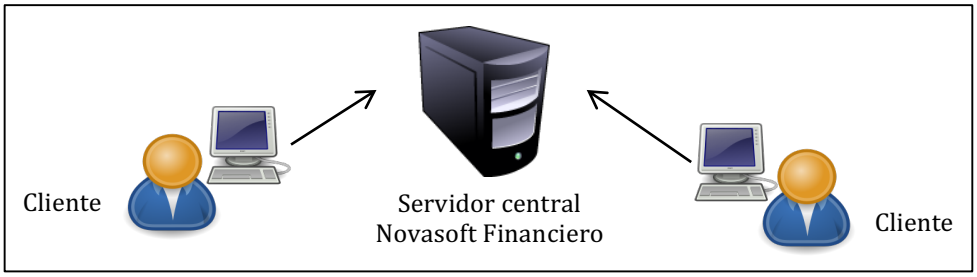

Figura 1. Esquema del Servidor Novasoft Financiero en Línea.

Su tarea consiste en construir un cliente que se comunique con el servidor para hacer una consulta en línea. El cliente y el servidor seguirán el protocolo descrito a continuación para su comunicación (Figura 2):

1. El cliente se comunica con el servidor para iniciar una sesión de consulta.
2. El servidor responde con un mensaje de confirmación.
3. El cliente envía la lista de algoritmos de cifrado que usará durante la sesión y espera un mensaje del servidor confirmando que soporta los algoritmos seleccionados (si no, el servidor envía un mensaje de terminación).
4. El servidor responde con un mensaje de confirmación. O con un mensaje de error si no soporta alguno de los algoritmos, en este caso ambos terminan la comunicación.
5. El cliente y el servidor intercambian certificados digitales (CD). Los dos certificados deben seguir el estándar X509.
6. El servidor genera una llave simétrica (LS) y la envía al cliente. Para proteger la llave, el servidor usa la llave pública del cliente (Kc+).
7. El cliente recibe la llave simétrica cifrada y la extrae. Después, el cliente responde enviando la misma llave simétrica, cifrada con la llave pública del servidor (Ks+).
8. A continuación, el cliente usa la llave simétrica para cifrar la consulta (un código de identificación de cuenta) y el código de autenticación HMAC correspondiente.
9. El servidor recibe la información, descifra y chequea el código. El servidor responde la consulta con la cadena OK:DEBE, OK:PAZYSALVO. Si encuentra problemas responde con la cadena ERROR.
10. Los dos terminan la comunicación.

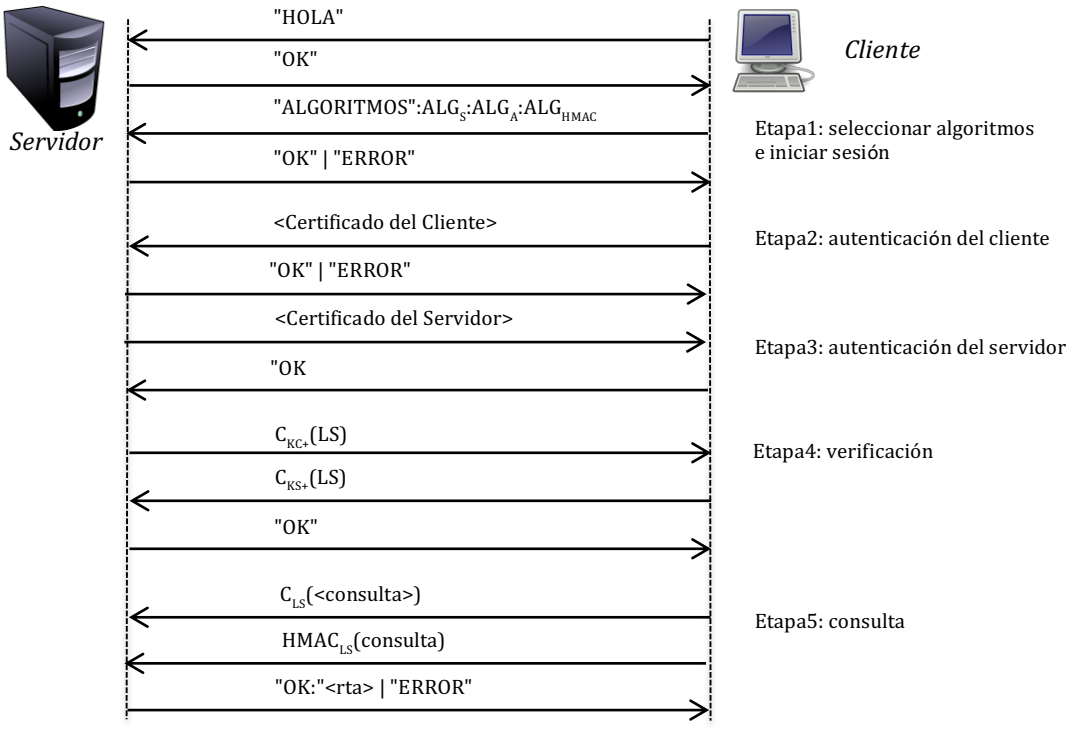

Figura 2. Protocolo de comunicación entre cliente y servidor (con seguridad).

TENGA EN CUENTA:

- El protocolo de comunicación maneja la siguiente convención:
  - Cadenas de Control: `"HOLA","ALGORITMOS", "OK", "ERROR"`, etc.
  - Separador Principal: " : "
- A continuación, se presentan los algoritmos disponibles en el servidor para manejo de las tareas de cifrado. Es decir, los algoritmos que deben reemplazar las cadenas ALG_S, ALG_A y ALG_D en el protocolo. Para implementar el cliente usted debe seleccionar un algoritmo en cada caso.
  - Simétricos (ALG_S):
    - AES. Modo ECB, esquema de relleno PKCS5, llave de 128 bits.
    - Blowfish. Cifrado por bloques, llave de 128 bits.
  - Asimétricos (ALG_A):
    - RSA. Cifrado por bloques, llave de 1024 bits.
  - HMAC (ALG_D):
    - HmacMD5
    - HmacSHA1
    - HmacSHA256

  Las cadenas que identifican cada uno de los algoritmos son: `"AES", "Blowfish", "RSA", "HMACMD5", "HMACSHA1", "HMACSHA256"`.
- Utilizaremos la versión 3 del estándar X509 para el CD. La idea es que tanto el cliente como el servidor pueden comprobar la identidad del otro a partir de un CD (en un caso real este debería ser expedido por una entidad certificadora pero aquí se va a generar localmente).
- Usaremos la librería Bouncycastle para las operaciones de cifrado y manejo de certificado.
- La comunicación se realiza a través de sockets de acuerdo con el protocolo de comunicación definido.
- Dado que existen problemas en la transmisión de los bytes cifrados, manejaremos encapsulamiento con cadenas hexadecimales para transmisión de datos de este tipo. Es necesario codificar en hexadecimal la información cifrada, para luego enviarla. El servidor hará lo mismo. Use DatatypeConverter para construir los métodos apropiados:

```
DatatypeConverter.printHexBinary(byteArray);
DatatypeConverter.parseHexBinary(cadena);
```

- Se entregarán dos archivos .jar: uno del servidor con seguridad y uno del servidor sin seguridad. La figura 3 ilustra el protocolo sin seguridad.

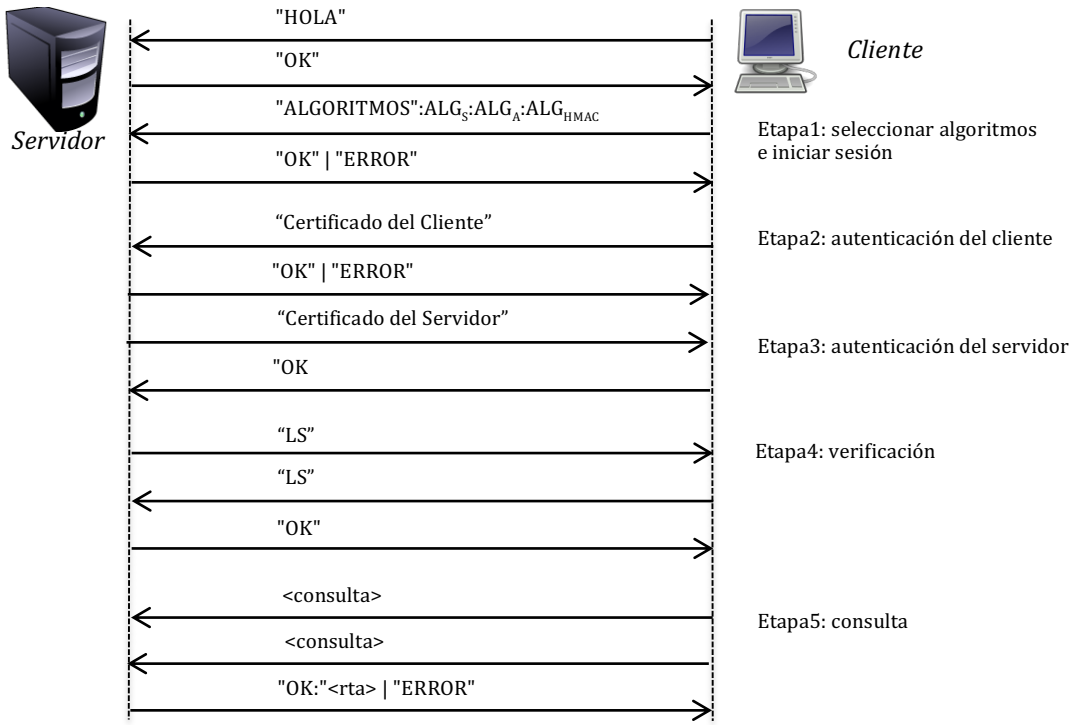

Figura 3. Protocolo de comunicación sin seguridad.

- A continuación se muestra un ejemplo del código para enviar el certificado. Primero se encapsula y luego se envía.

```
java.security.cert.X509Certificate certificado = generarCertificado(llaves);
byte[] certificadoEnBytes = certificado.getEncoded( );
String certificadoEnString = printHexBinary(certificadoEnBytes);
socketParaComunicacion.println(certificadoEnString);
```

Entrega:

Cada grupo debe entregar un zip de un proyecto Java con:

- La implementación correspondiente al cliente (descrito en la parte C),
- En el subdirectorio docs un archivo que incluya el informe (con las respuestas a las tareas A y B). El informe DEBE incluir los nombres de los integrantes del grupo.

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