PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

IIC3253 - Criptografía y Seguridad Computacional
1er semestre del 2021
Vicente Merino

Tarea 3

Pregunta 1

Parte (a)

El principal problema es la **auditabilidad**, es decir poder asegurar desde el lado cliente que el tercero a quien mande las credenciales, esté manejando correctamente dichas credenciales de forma segura. Algunos casos hipotéticos podría ser un servicio, que tenga un servidor oculto con una base de datos en donde guarde mi usuario y contraseña en texto plano (sin hash, salt, ni nada de eso), con lo cual después podrían por ejemplo venderle la base de datos a un mal agente, o bien, ellos mismos comportarse como un mal agente y vulnerar mi seguridad. No tengo forma de saber si el servicio está incurriendo en esta práctica o no. Otro caso hipotético es que el servicio diga que yo realicé cierta acción, modificando los datos, siendo que esto no fue hecho por el usuario (como por ejemplo una transacción fraudulente en un banco o servicio financiero) y no hay forma de corroborar si esto fue o no así.

Parte (b)

La arquitectura del protocolo que seguiremos está basado en firmas digitales desde el lado cliente:

1. El usuario deberá seguir definiendo un nombre de usuario y contraseña (por simplicidad para él).
2. Genero un par llave pública/privada que me permita firmar (como ElGamal por ejemplo) las cuales guardo localmente.
3. Cuando me intente registrar, solo enviaré mi nombre de usuario, mi llave pública y mi nombre de usuario firmado con mi llave privada.
4. El servicio deberá comprobar que la firma sea correcta, en cuyo caso guardará el par nombre de usuario y llave pública.
5. Cada vez que me **autentifique**, debo firmar mi nombre de usuario con mi llave privada y el servicio debe comprobar que la firma coincida usando los datos que tiene guardados. De ser el caso, entonces devuelve un Token.
6. Cada vez que el usuario haga una request, envió también el token y el token firmado con mi llave privada (esto da una capa extra de seguridad). Si la firma es correcta y además el token es el que me envió el servidor, entonces el servicio **autoriza** la request y me devuelve los datos correspondientes.

Esto soluciona los problemas de auditabilidad, ya que ya no me importa como el servidor esté guardando mis datos, si los está guardando de forma segura o no, ya que lo que está guardando es mi llave pública, la cual cualquiera la puede conocer y me daría lo mismo (ya que es pública y por el funcionamiento de los protocolos de criptografía simétrica). Además que ahora si el servicio me engaña y agrega una acción que yo digo que no hice, ya no es solo una cuestión de palabra cliente v/s palabra servicio, sino que basta mostrar la firma de la acción, para comprobar quien tiene la razón.

Una posible vulneración a este protocolo, es ¿qué pasaría si un tercero tiene acceso físico a mi ordenador y puede saber mi llave privada? Con ello podría fácilmente acceder al servicio haciéndose pasar por mí, ya que podría firmar los mensajes y las requests, presentándose un posible problema de seguridad. Una solución a este problema es tener un password maestro *password*, que seguramente es simple a nivel de usuario y a partir de este password se deriva por PBDKF2 una llave $k = \text{PBDKF2}(password)$. Luego mi llave privada la guardo encriptada usando un protocolo de criptografía asimétrica, como DES o AES, usando este k como la llave para encriptar. Esto lo puedo guardar tanto en mi computador o un sevidor, lo cual da lo mismo, ya que está encriptada, y si después necesito mi llave privada original para firmar por ejemplo, obtengo mi llave k de la misma forma, a partir de mi clave *password* maestra y la uso con la función de decriptación del protocolo para obtener la llave privada. Esto solucionaría el problema, ya que ya no estaría guardada en ninguna parte como texto plano la llave privada. Seguiría suponiendo un problema de seguridad (un poco menos grave), ya que en el caso de DES puede ser vulnerado en $O(2^{56})$ con fuerza bruta. Una alternativa es usar 3-DES, que tomaría $O(2^{112})$.