IIC 3253 - Tarea 3

Matías Correa I.
15634183

Pregunta 1

a) El primer problema surge al considerar que nuestro (user, pass) se envía en texto plano al servidor: nuestra información personal queda a merced del trato que el servidor le dé.

Pensemos en que un hacker roba la base de datos del servidor S. Si S almacenó mi info tal cual, el problema es evidente. Si S hasheó mi contraseña, se puede crackear usando rainbow tables (conjuntos de hashes precomputados), o bien se puede determinar si es que repetí mi contraseña en otro sitio del cual el hacker pudo robar info.

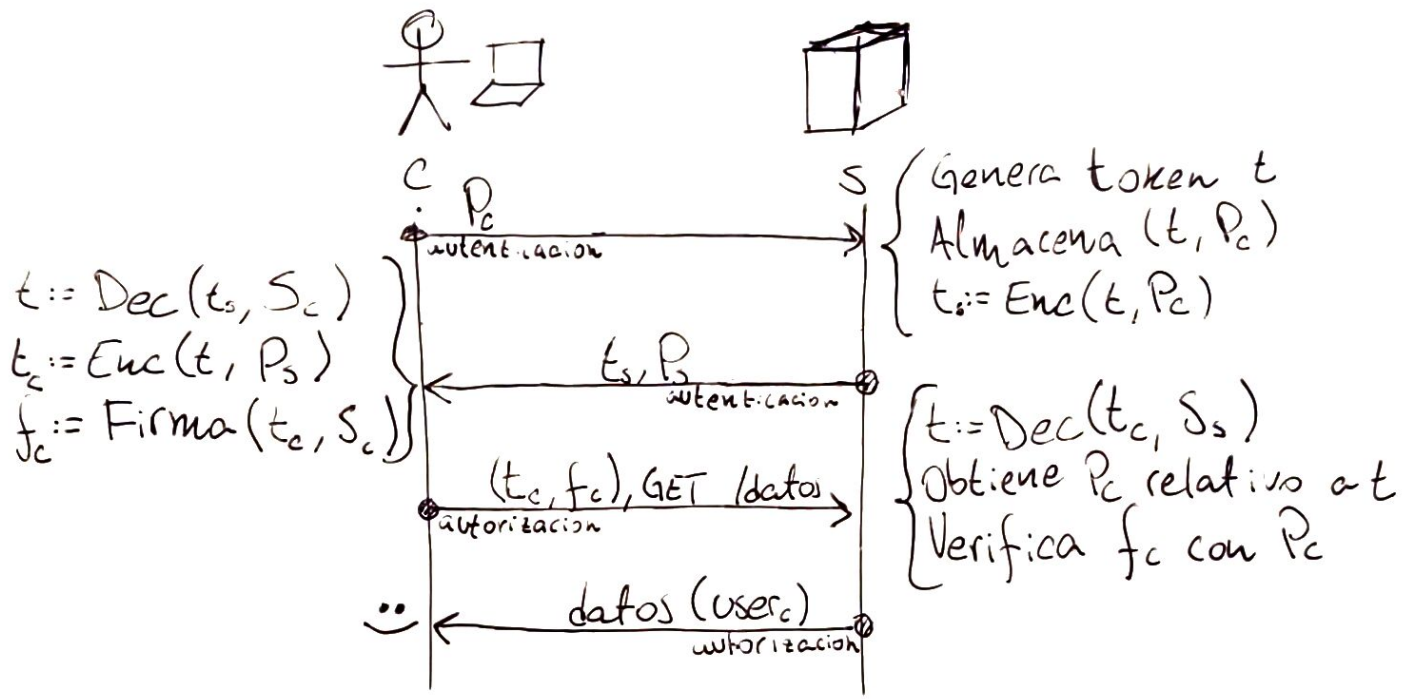
Otro gran problema es la Auditabilidad: efectivamente comprobar que el autor de una acción es quien dice ser. En el esquema existente a base de (user, pass, token) no está el concepto de firmar mis requests: son anónimas.

b) PROPUESTA

Se ideó un protocolo que prescinde del uso de contraseñas para autenticarse en un sitio.

El protocolo funciona a base de pares de claves públicas y privadas para autenticar, autorizar y mantener auditabilidad de los clientes

Sean: C: Cliente ; (P_c, S_c) : su par de claves
S: Server ; (P_s, S_s) : "



Obviando el proceso de registro, donde S enlaza una cuenta de usuario con una llave pública, el diagrama presentado ilustra la creación de una sesion y su persistencia. Una sesión existe en términos de un token t.

Sin embargo, t siempre se envía encriptado con la clave pública de la contraparte, de manera que para mantener comunicación siempre es necesaria la clave secreta relativa. Más aún, aprovechando que el servidor cuenta con la clave pública del cliente, se aprovecha de firmar el mensaje para:

1) Mayor Seguridad: si un atacante roba la

base de tokens, nunca podrá recibir autorización si no cuenta con la clave secreta correspondiente a la clave pública enlazada a cada token.

2) Auditabilidad: la firma digital es una manera de certificar la autoría de un mensaje.

Ahora, pensando en que P_c debe ser enviado en texto plano para ser usada por S, también quedamos a merced de cómo manejen ellos esta data. Si un atacante obtiene P_c no es algo malo en sí, es es una clave pública. Un problema sería si además C usa P_c para autenticarse en varios sitios: si el atacante logra llegar a S_c, gana acceso instantáneo a varias cuentas de C.

Pensando en ese problema, este protocolo debiese existir acompañado de soluciones hardware/software que generen y administren pares (P, S) diferentes para cada servicio en el cual C se registre.

Cabe destacar que no se espera que C_i al iniciar sesión en un servicio i, ingrese $P_{c,i}$ a mano. La idea es que estas soluciones se encarguen de gestionar de manera imperceptible al usuario todo lo relativo a sus claves $(P_{c,i}, S_{c,i})$.