PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACION

Criptografía y Seguridad Computacional - IIC3253
Rúbrica Tarea 3

Preguntas

1. En clases se mencionaron ciertas consecuencias negativas que conlleva el autentificar y autorizar a usuarios en la Web en base a sus nombres de usuario (o correos electrónicos) y contraseñas.

   (a) Explique en detalle cuáles son estas consecuencias, y describa casos hipotéticos que evidencien que son negativas.

   (b) Diseñe una alternativa para autentificar y autorizar usuarios en la web que **no** traiga consigo las consecuencias negativas mencionadas. Explique cómo funcionaría su sistema de autentificación/autorización, y qué problemas prácticos podría traer consigo. Finalmente, explique qué medidas tomaría para prevenir dichos problemas.

Corrección. La corrección de cada ítem se llevará a cabo de acuerdo a lo especificado en las tablas que se presentan a continuación.

(a)

0 puntos	La respuesta no menciona que hay un problema en compartir la clave con el servicio, ni menciona el problema de la auditabilidad de los sistemas que basan su autentificación y autorización en una contraseña y un nombre de usuario o correo.
1 punto	Aunque la respuesta menciona ciertos problemas, ya sea relacionados a la auditabilidad o a compartir la contraseña, no describe casos hipotéticos que ejemplifiquen por qué se presentan estos problema en los sistemas que basan su autentificación y autorización en una contraseña y un nombre de usuario o correo.
2 puntos	La respuesta menciona que hay problemas de auditabilidad, ya sea por compartir la contraseña con el servicio o porque las acciones tomadas en dicho servicio no son auditables. Sin embargo presenta casos hipotéticos insuficientes, que no evidencian completamente estos problemas.
3 puntos	Menciona que el autentificar y autorizar usuarios en base a una contraseña y un nombre de usuario o correo trae consigo problemas tanto porque se debe compartir la contraseña y porque las acciones tomadas en los sistemas no son auditables, y presenta al menos dos situaciones hipotéticas en las que dichos problemas quedan expuestos claramente.

(b)

0 puntos	O bien no se presenta una alternativa, o la alternativa presentada es trivial y no resuelve ninguno de los problemas relacionados a la auditabilidad vistos en clases.
0.5 puntos	Presenta una alternativa que resuelve ciertos problemas prácticos del lado del usuario, tales como utilizar contraseñas fuertes o contraseñas distintas para distintos servicios, pero dicha alternativa no resuelve los problemas de auditabilidad vistos en clases.
1 punto	Presenta una alternativa que permite al usuario no compartir su secreto o bien auditar las acciones tomadas en el sistema, pero no discute los problemas prácticos asociados..
2 puntos	Presenta una alternativa que permite al usuario no compartir su clave secreta y auditar las acciones tomadas en el sistema, y discute el problema práctico del manejo de llaves secretas pero su solución a dicho problema no es suficiente para que el sistema sea considerado seguro. Por ejemplo, no menciona que la o las llave(s) secreta(s) se deben guardar encriptadas.
3 puntos	Presenta una alternativa que permite al usuario no compartir su clave secreta y auditar las acciones tomadas en el sistema, y discute el problema práctico del manejo de llaves secretas. Propone una solución al manejo de llaves secretas que permite a un usuario estar seguro de que sus llaves secretas no pueden ser generadas/sustraídas sin tener acceso a un secreto que sólo conoce el usuario.

Descuentos extras:

- No explicar en detalle el método de comunicación bajo el nuevo sistema. (-0.5)
- Confundir conceptos básicos vistos en clases (-1).

En esta pregunta usted deberá implementar en Python el protocolo de firma digital de Schnorr. Suponga como dados un número primo p y un par de elementos $g, q \in \{1, \ldots, p-1\}$ tal que q es el orden del sub-grupo generado por g en el grupo $\mathbb{Z}_p^* = (\{1, \ldots, p-1\}, \cdot)$. Vale decir, se deben cumplir las siguientes condiciones:

$$\begin{aligned} g^i &\not\equiv g^j \mod p \qquad \text{para cada } i, j \in \{1, \ldots, q\} \text{ con } i \neq j, \\ g^q &\equiv 1 \mod p. \end{aligned}$$

Además, suponga que el espacio M de posibles mensajes corresponde con los strings de Python (tipo `str`), y que $h : M \to H$ es una función de hash con H un subconjunto finito de $\mathbb{N}$. En el protocolo criptográfico ElGamal, la clave privada de un usuario A es un número $x \in \{1, \ldots, q-1\}$ generado al azar con distribución uniforme, y la clave pública de A es $y = g^x \mod p$. En el protocolo de firma digital de Schnorr, A utiliza su clave privada x para firmar un mensaje $m \in M$ de la siguiente forma. A genera $k \in \{1, \ldots, q-1\}$ al azar con distribución uniforme, y luego calcula:

$$\begin{aligned} r &= g^k \mod p, \\ e &= h(r \| m), \\ s &= k - x \cdot e, \end{aligned}$$

donde $r \| m$ es la concatenación del número r visto como un string con m (por ejemplo, se tiene que `3341`$\|$`hola` = `3341hola`). Entonces la firma de Schnorr de m es definida como el par (e, s).

Un usuario B puede verificar que una firma (e, s) para un mensaje m fue generada por A de la siguiente forma. Utilizando la clave pública y de A, el usuario B calcula $r' = (g^s \cdot y^e) \mod p$, y luego verifica si $h(r' \| m) = e$.

Para implementar el protocolo de firma digital de Schnorr, primero deberá crear un archivo `grupo.txt` para almacenar los valores de p, g y q que definen el grupo a ser utilizado. En particular, estos valores debe ser tomados desde aquí: `https://datatracker.ietf.org/doc/html/rfc5114#section-2.3`. El archivo `grupo.txt` deberá ser incluido en su repositorio, ya que será utilizado para corregir su tarea. Después deberá implementar las siguientes funciones, sólo suponiendo como dadas las funciones aritméticas básicas para números enteros (suma, resta, multiplicación, división y resto).

2.
   - Una función que permite crear claves públicas y privadas de acuerdo al protocolo criptográfico ElGamal.

```python
def generar_clave_ElGamal():
    # Retorna :
    #   Genera una clave privada y una clave publica segun el protocolo
    #   critografico ElGamal, para el grupo almacenado en grupo.txt.
    #   Almacena la clave privada en private_key.txt, y la clave publica
    #   en public_key.txt.
```

   - Una función que retorna una firma de un mensaje utilizando el protocolo de Schnorr.

```python
def firmar_Schnorr(m: str) -> (int,int):
    # Argumentos :
    #   m: str - mensaje
```

```
    # Retorna :
    #   (int,int) - firma de Schnorr (e,s) del mensaje m segun la clave
    #   privada almacenada en private_key.txt, para el grupo almacenado
    #   en grupo.txt
```

- Una función que verifica si un par (e, s) es un firma de un mensaje m de acuerdo al protocolo de Schnorr.

```
def verificar_firma_Schnorr(m: str, firma: (int,int)) -> bool:
    # Argumentos :
    #   m: str - mensaje
    #   firma: (int,int) - firma de Schnorr (e,s) para m
    # Retorna :
    #   bool - retorna True si para el usuario con clave publica
    #   almacenada en public_key.txt, el par (e,s) es una firma de
    #   Schnorr correcta para el mensaje m, en el grupo almacenado en
    #   grupo.txt. En caso contrario retorna False.
```

- Una función que implementa MD5 interpretando la salida como un número natural.

```
def md5(m: str) -> int:
    # Argumentos :
    #   m: str - mensaje
    # Retorna :
    #   int - valor de la funcion de hash MD5 aplicada sobre m
```

Corrección. Para corregir esta pregunta se realizarán 12 tests sobre firmas de mensajes utilizando el protocolo de Schnorr. Por cada test correcto usted recibirá 0.5 puntos en esta pregunta.