



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

## Criptografía y Seguridad Computacional - IIC3253

### Tarea 3

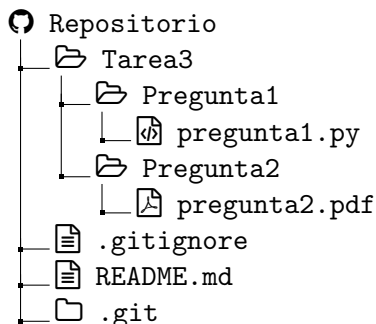
Plazo de entrega: martes 10 de junio

## Instrucciones

Cualquier duda sobre la tarea se deberá hacer en los *issues* del repositorio del curso. Los issues son el canal de comunicación oficial para todas las tareas.

**Configuración inicial.** Para esta tarea utilizaremos *github classroom*. Para acceder a su repositorio privado debe ingresar al siguiente link, seleccionar su nombre y aceptar la invitación. El repositorio se creará automáticamente una vez que haga esto y lo podrá encontrar junto a los repositorios del curso. Para la corrección se utilizará Python 3.11.

**Entrega.** Al entregar esta tarea, su repositorio se deberá ver exactamente de la siguiente forma:



Para cada problema cuya solución se deba entregar como un documento (en este caso la pregunta 2), deberá entregar un archivo **.pdf** que, o bien fue construido utilizando **L<sup>A</sup>T<sub>E</sub>X**, o bien es el resultado de digitalizar un documento escrito a mano. En caso de optar por esta última opción, queda bajo su responsabilidad la legibilidad del documento. Respuestas que no puedan interpretar de forma razonable los ayudantes y profesores, ya sea por la caligrafía o la calidad de la digitalización, serán evaluadas con la nota mínima.

## Preguntas

1. En esta pregunta usted debe escribir un programa para atacar a un servicio de almacenamiento de logs. Dicho servicio ha sido protegido utilizando HMAC con MD5. En particular, cada vez que se envía el mensaje `log`, se debe enviar también `HMAC-MD5(k, log)`, donde `k` es una llave que usted desconoce. La implementación de este servicio tiene una vulnerabilidad que usted deberá explotar.

Deberá entregar un archivo llamado `pregunta1.py` que contenga una función `get_tag(log: string) -> string` que retorne el tag correspondiente a `log`, esto es, `HMAC-MD5(k, log)`. Vale decir, debe explotar la vulnerabilidad en la implementación de manera tal de poder generar un tag válido para cualquier mensaje `log` dado como parámetro a la función `get_tag`. Su programa debe:

- Obtener la url y puerto del servicio desde las variables de entorno `LOG_SERVICE_URL` y `LOG_SERVICE_PORT`, respectivamente.
- Demorar, en cada ejecución de `get_tag`, no más de 10 minutos en el computador de un ayudante (esto es más que suficiente).

Para encontrar la vulnerabilidad y programar su función, debe usar un programa que se encuentra en el repositorio del curso junto a este enunciado. Al ejecutar dicho programa, el servicio de logs quedará disponible en la url `http://localhost` bajo el puerto 8080. Este servicio recibe logs en el path `send_log` con `{"log": string, "tag": string}` en el body de un POST. El servicio usa como llave para calcular el MAC la variable de entorno `KEY` en caso de estar presente. De lo contrario usará un valor por defecto.

2. En esta pregunta estudiaremos un sistema criptográfico de clave pública que es definido a partir de un problema NP-completo. Para esto considere el problema SUBSET-SUM definido de la siguiente forma. La entrada de SUBSET-SUM es un conjunto  $S$  de números naturales y un número natural  $b$ , y la pregunta responder es si existe  $I \subseteq S$  tal que:

$$\sum_{a \in I} a = b.$$

Por ejemplo, si  $S_1 = \{1, 2, 3, 4, 5\}$  y  $b_1 = 7$ , entonces tanto  $\{2, 5\}$  como  $\{1, 2, 4\}$  son soluciones del problema. Por el contrario, si  $S_2 = \{2, 4, 6, 8\}$  y  $b_2 = 5$ , entonces el problema no tiene solución. Es conocido que SUBSET-SUM es un problema NP-completo, por lo que se cree que no existe un algoritmo de tiempo polinomial que lo solucione.

En el sistema criptográfico basado en SUBSET-SUM, un usuario establece sus claves pública y secreta a través del siguiente protocolo, suponiendo que el espacio de mensaje es  $\mathcal{M} = \{0, 1\}^n$ .

- (1) Se escoge un conjunto  $S = \{w_1, \dots, w_n\}$  de números naturales tal que  $S$  es súper creciente, vale decir, para cada  $k \in \{2, \dots, n\}$  se tiene que:

$$\sum_{i=1}^{k-1} w_i < w_k.$$

Por ejemplo,  $S_1 = \{1, 2, 4, 8\}$  es súper creciente, mientras que  $S_2 = \{1, 2, 3, 4\}$  no es súper creciente.

(2) Se escogen números naturales  $q, r$  tales que  $MCD(q, r) = 1$  y

$$\sum_{i=1}^n w_i < q.$$

(3) Se construye el conjunto  $T = \{t_1, \dots, t_n\}$  de números naturales tales que  $t_i = (w_i \cdot r) \bmod q$  para cada  $i \in \{1, \dots, n\}$ .

(4) Se define la clave secreta como  $(S, q, r)$  y la clave pública como  $T$ .

Las funciones de cifrado y descifrado son definidas de la siguiente forma, considerando que la clave pública de un usuario  $A$  es  $P_A = T$  y la clave secreta de  $A$  es  $S_A = (S, q, r)$ . Suponga que  $m = b_1 \cdots b_n$  es un mensaje, donde  $b_i \in \{0, 1\}$  para cada  $i \in \{1, \dots, n\}$ . Entonces

$$Enc_{P_A}(m) = \sum_{i=1}^n b_i \cdot t_i.$$

Observe que el texto cifrado de un mensaje es un número natural.

Por otro lado, la función de descifrado  $Dec_{S_A}$  se define de la siguiente forma. Dado un texto cifrado  $c \in \mathbb{N}$ , sea  $d = (s \cdot c) \bmod q$ , donde  $s$  es un número natural tal que  $r \cdot s \equiv 1 \bmod q$  (sabemos que este número existe puesto que  $MCD(q, r) = 1$ ). Además, sean  $s_1, \dots, s_n$  tales que  $s_i \in \{0, 1\}$  para cada  $i \in \{1, \dots, n\}$  y:

$$\sum_{i=1}^n s_i \cdot w_i = d. \quad (\dagger)$$

Entonces  $Dec_{S_A}(c) = s_1 \cdots s_n$ .

A continuación usted debe demostrar que el protocolo anterior es correcto y puede ser implementado en tiempo polinomial, para lo cual debe responder las siguientes preguntas.

(a) Demuestre que, dado un conjunto  $S$  de números naturales que es súper creciente y un número  $b$ , existe a lo más un conjunto  $I \subseteq S$  tal que:

$$\sum_{a \in I} a = b.$$

(b) Construya un algoritmo polinomial tal que, dado un conjunto  $S$  de números naturales que es súper creciente y un número  $b$ , verifica si existe  $I \subseteq S$  tal que:

$$\sum_{a \in I} a = b.$$

Este algoritmo debe además construir  $I$  si tal conjunto existe. Note que de (a) y (b) se concluye que  $Dec_{S_A}$  está bien definido, si existe una solución para la ecuación  $(\dagger)$ , y puede ser implementado en tiempo polinomial.

(c) Sea  $T = \{t_1, \dots, t_n\}$  definido como en el paso (3) del protocolo. Demuestre que  $t_i \neq t_j$  para  $i, j \in \{1, \dots, n\}$  tal que  $i \neq j$ . Note que de este se deduce que  $T$  contiene  $n$  elementos distintos y  $Enc_{P_A}$  está bien definido.

(d) Demuestre que para todo mensaje  $m \in \{0, 1\}^n$ , se tiene que  $Dec_{S_A}(Enc_{P_A}(m)) = m$ .