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 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.

   Corrección. En esta pregunta se obtiene un punto base por que su programa tenga el setup correcto. Esto significa que si su programa es ejecutado en un ambiente con las variables correctas, intenta obtener el mac para un mensaje interactuando correctamente con el servicio. Si es necesario modificar su programa para que se ejecute correctamente entonces usted no obtendrá ese punto base. A lo anterior se le suma:

1 punto. El programa hace algunos requests al servicio de logs pero evidentemente no está explotando una vulnerabilidad. Por ejemplo, el programa está intentando obtener el MAC por fuerza bruta o tratando de calcular el mac sacando la llave secreta de alguna variable de entorno.

2.5 puntos. El programa hace requests al servicio de logs tratando de explotar una vulnerabilidad pero, por errores en su lógica interna, está lejos de obtener un MAC válido.

4 puntos. El programa explota una vulnerabilidad, pero

- por algún error menor, no obtiene un tag correcto (por ejemplo el último byte no es correcto); o
- obtiene el tag correcto en al menos dos de tres intentos, pero demorando más de 10 minutos en encontrarlo; o
- obtiene el tag correcto en uno de tres intentos demorando menos de 10 minutos.

5 puntos. El programa explota correctamente una vulnerabilidad, ya sea un timing attack o un ataque enviando tags de largo incorrecto. Obtiene el tag correcto en al menos 2 de 3 intentos.

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:

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

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 $\mathcal{M} = \{0, 1\}^n$.

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

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

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

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

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

(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 $i \in \{1, \ldots, n\}$. 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 $r \cdot s \equiv 1 \mod q$ (sabemos que este número existe puesto que $MCD(q, r) = 1$). Además, sean s_1, ..., s_n tales que $s_i \in \{0, 1\}$ para cada $i \in \{1, \ldots, n\}$ y:

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

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 (†), y puede ser implementado en tiempo polinomial.

(c) Sea $T = \{t_1, ..., t_n\}$ definido como en el paso (3) del protocolo. Demuestre que $t_i \neq t_j$ para $i, j \in \{1, \ldots, n\}$ 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 $Dec_{S_A}(Enc_{P_A}(m)) = m$.

Corrección. La asignación de puntajes para cada pregunta de este problema es la siguiente.

(a) El puntaje máximo en esta pregunta es 1.5 puntos. Se obtiene 0.7 puntos si se entrega una versión de la demostración con algunos errores menores, y se obtiene el puntaje completo si la demostración no tiene errores.

(b) El puntaje máximo en esta pregunta es 1.5 puntos. Se obtiene 0.5 puntos si se entrega un algoritmo que funciona en tiempo polinomial pero no se explica por qué es correcto, se obtiene 1 punto si se entrega un algoritmo que funciona en tiempo polinomial y se explica por qué es correcto, pero no se demuestra que se cumple esto, y se obtiene 1.5 puntos si se entrega un algoritmo que funciona en tiempo polinomial y se demuestra que es correcto.

(c) El puntaje máximo en esta pregunta es 1.5 puntos. Se obtiene 0.7 puntos si se entrega una versión de la demostración con algunos errores menores, y se obtiene el puntaje completo si la demostración no tiene errores.

(d) El puntaje máximo en esta pregunta es 1.5 puntos. Se obtiene 0.7 puntos si se da una idea de por qué es cierto que $Dec_{S_A}(Enc_{P_A}(m)) = m$ pero no se entrega una demostración completa de esta propiedad, y se obtiene el puntaje completo si se demuestra correctamente que esta propiedad es cierta.