Pontificia Universidad Católica de Chile
Departamento de Ciencias de la Computación
IIC3253 – Criptografía y Seguridad Computacional
Profesor Marcelo Arenas y Martin Ugarte
Primer Semestre del 2025

Ayudantía 3

Funciones de hash

Ayudante: Cristóbal Rojas – cristobalrojas@uc.cl

Definiciones y conceptos preliminares

Funciones Hash Criptográficas

Una función hash criptográfica es un par de algoritmos (Gen, h) donde:

- Gen es un algoritmo aleatorizado de tiempo polinomial que toma como entrada un parámetro de seguridad 1^n y genera una clave s.
- h es un algoritmo determinístico de tiempo polinomial que toma como entrada la clave s y un mensaje $m \in \{0,1\}^*$, y devuelve un hash $h^s(m) \in \{0,1\}^{\ell(n)}$, donde ℓ es una función polinomial fija.

Propiedades de seguridad

Las funciones hash criptográficas deben satisfacer las siguientes propiedades de seguridad:

1. **Resistencia a preimagen:** Dado un valor hash y, debe ser computacionalmente inviable encontrar cualquier mensaje x tal que $h^s(x) = y$.
2. **Resistencia a colisiones:** Debe ser computacionalmente inviable encontrar cualquier par de mensajes distintos x y x' tal que $h^s(x) = h^s(x')$.

Funciones despreciables

Una función $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ es despreciable si para todo polinomio $p : \mathbb{N} \to \mathbb{N}$, existe un entero n_0 tal que para todo $n > n_0$:

$$f(n) < \frac{1}{p(n)}$$

El juego Hash-Col

Para formalizar la noción de resistencia a colisiones, se define el juego Hash-Col(n):

1. El verificador genera $s = Gen(1^n)$ y se lo entrega al adversario.
2. El adversario elige mensajes m_1 y m_2 con $m_1 \neq m_2$.
3. El adversario gana el juego si $h^s(m_1) = h^s(m_2)$, y en caso contrario pierde.

Una función hash (Gen, h) se dice resistente a colisiones si para todo adversario probabilístico de tiempo polinomial $\mathcal{A}$, existe una función despreciable $negl(n)$ tal que:

$$\Pr[\mathcal{A} \text{ gana Hash-Col}(n)] \leq negl(n)$$

Problemas

Problema 1

Sean (Gen_1, h_1) y (Gen_2, h_2) dos funciones hash. Definamos (Gen, h) de manera que Gen ejecuta Gen_1 y Gen_2 para obtener las claves s_1 y s_2, respectivamente. Luego definimos $h^{s_1,s_2}(x) = h_1^{s_1}(x) \| h_2^{s_2}(x)$, donde $\|$ denota la concatenación.

Demuestre que si al menos una de (Gen_1, h_1) y (Gen_2, h_2) es resistente a colisiones, entonces (Gen, h) es resistente a colisiones.

Solución

Sea $\mathcal{A}$ un adversario que con probabilidad $\varepsilon(n)$ produce un par x, x' tal que $h^{s_1,s_2}(x) = h^{s_1,s_2}(x')$. En tal caso, por la definición de h, tenemos que $h_1^{s_1}(x) = h_1^{s_1}(x')$ y $h_2^{s_2}(x) = h_2^{s_2}(x')$. Por lo tanto, $\mathcal{A}$ encuentra una colisión tanto en h_1 como en h_2 con probabilidad ε.

Para formalizar esta idea, vamos a demostrar que si $\mathcal{A}$ es un adversario exitoso contra h en el juego Hash-Col, entonces podemos construir adversarios $\mathcal{B}_1$ y $\mathcal{B}_2$ que atacan a h_1 y h_2 respectivamente:

Construcción de $\mathcal{B}_1$ (adversario contra h_1):

1. $\mathcal{B}_1$ recibe como entrada la clave s_1 generada por $Gen_1(1^n)$.
2. $\mathcal{B}_1$ ejecuta $Gen_2(1^n)$ para obtener s_2.
3. $\mathcal{B}_1$ invoca $\mathcal{A}$ con el par de claves (s_1, s_2).
4. Si $\mathcal{A}$ produce un par (x, x') tal que $h^{s_1,s_2}(x) = h^{s_1,s_2}(x')$, entonces $\mathcal{B}_1$ devuelve este mismo par.

De manera similar, podemos construir $\mathcal{B}_2$ para atacar h_2.

Análisis: Cuando $\mathcal{A}$ encuentra una colisión en h, es decir, un par (x, x') tal que $h^{s_1,s_2}(x) = h^{s_1,s_2}(x')$, por la definición de h sabemos que:

$$h_1^{s_1}(x) \| h_2^{s_2}(x) = h_1^{s_1}(x') \| h_2^{s_2}(x')$$

Esta igualdad implica que $h_1^{s_1}(x) = h_1^{s_1}(x')$ y $h_2^{s_2}(x) = h_2^{s_2}(x')$. Por lo tanto, el mismo par (x, x') es una colisión tanto para h_1 como para h_2.

Esto significa que:

$$\Pr[\mathcal{B}_1 \text{ gana Hash-Col}(n)] = \Pr[\mathcal{A} \text{ gana Hash-Col}(n)] = \varepsilon(n)$$

$$\Pr[\mathcal{B}_2 \text{ gana Hash-Col}(n)] = \Pr[\mathcal{A} \text{ gana Hash-Col}(n)] = \varepsilon(n)$$

Si al menos una de las funciones, digamos h_1, es resistente a colisiones, entonces

$$\Pr[\mathcal{B}_1 \text{ gana Hash-Col}(n)] \leq \mathit{negl}(n)$$

para alguna función despreciable *negl*. Pero como acabamos de ver,

$$\Pr[\mathcal{B}_1 \text{ gana Hash-Col}(n)] = \varepsilon(n),$$

lo que implica que $\varepsilon(n) \leq \mathit{negl}(n)$.

Por lo tanto, $\Pr[\mathcal{A} \text{ gana Hash-Col}(n)] \leq \mathit{negl}(n)$, lo que demuestra que h es resistente a colisiones si al menos una de h_1 o h_2 lo es.

Problema 2

Sea (Gen, h) una función hash resistente a colisiones. ¿Es $(Gen, \hat{h})$ definida por

$$\hat{h}^s(x) \stackrel{\text{def}}{=} h^s(h^s(x))$$

necesariamente resistente a colisiones?

Solución

Sí, $(Gen, \hat{h})$ es necesariamente resistente a colisiones si (Gen, h) lo es. Vamos a demostrarlo por contradicción.

Supongamos que $(Gen, \hat{h})$ no es resistente a colisiones. Esto significa que existe un adversario $\mathcal{A}$ que puede encontrar colisiones para $\hat{h}$ con probabilidad no despreciable. Es decir, $\mathcal{A}$ puede encontrar un par x, x' con $x \neq x'$ tal que:

$$\hat{h}^s(x) = \hat{h}^s(x')$$

Lo que implica:

$$h^s(h^s(x)) = h^s(h^s(x'))$$

Analizamos dos casos posibles:

Caso (a): $h^s(x) = h^s(x')$.

En este caso, el par x, x' ya constituye una colisión para la función hash original h.

Esto significa que si podemos encontrar tal par con probabilidad no despreciable, entonces hemos encontrado una colisión para h con la misma probabilidad, lo que contradice nuestra suposición de que h es resistente a colisiones.

Caso (b): $h^s(x) \neq h^s(x')$.

Definamos:

$$y = h^s(x) \tag{1}$$
$$y' = h^s(x') \tag{2}$$

Por la suposición de este caso, sabemos que $y \neq y'$.
Pero, como $\hat{h}^s(x) = \hat{h}^s(x')$, tenemos que:

$$h^s(y) = h^s(y')$$

Esto significa que el par y, y' constituye una colisión para la función hash original h.

Por lo tanto, si existiera un algoritmo que pudiera encontrar colisiones para $\hat{h}$ con probabilidad no despreciable, podríamos usarlo para construir un algoritmo que encuentra colisiones para h con la misma probabilidad:

1. Utilizamos el adversario $\mathcal{A}$ para obtener una colisión x, x' en $\hat{h}$.
2. Calculamos $y = h^s(x)$ y $y' = h^s(x')$.
3. Si $y = y'$, entonces x, x' es una colisión para h.
4. Si $y \neq y'$, entonces y, y' es una colisión para h.

En ambos casos, obtenemos una colisión para h con la misma probabilidad con la que $\mathcal{A}$ encuentra colisiones para $\hat{h}$. Esto contradice la suposición de que h es resistente a colisiones.

Por lo tanto, concluimos que si h es resistente a colisiones, entonces $\hat{h}$ también debe serlo. En términos formales, si la probabilidad de encontrar colisiones en h es despreciable, entonces la probabilidad de encontrar colisiones en $\hat{h}$ también debe ser despreciable.

Problema 3

Entrega una definición formal para el juego Hash-PreImg, es decir, el juego que representa la noción de resistencia a preimagen para una función de hash (Gen, h).

Solución

Una función hash (Gen, h) es resistente a preimagen si para todo adversario probabilístico de tiempo polinomial $\mathcal{A}$, existe una función despreciable *negl* tal que la probabilidad de éxito de $\mathcal{A}$ en el siguiente experimento (denotado como Hash-PreImg) es despreciable:

Juego Hash-PreImg(n)**:**

(a) Se genera una clave s ejecutando $Gen(1^n)$, y se elige un valor aleatorio $x \in \{0,1\}^{2n}$.

(b) Al adversario $\mathcal{A}$ se le proporciona s y $h^s(x)$, y éste produce un valor x'.

(c) Decimos que $\mathcal{A}$ tiene éxito si $h^s(x') = h^s(x)$.

Formalmente, una función hash es resistente a preimagen si:

$$\Pr[\mathcal{A} \text{ gana Hash-PreImg}(n)] \leq \mathit{negl}(n)$$

para alguna función despreciable *negl*.

Observación: Esta definición captura la idea de que, dado un valor hash $h = h^s(x)$ para un valor desconocido x, debe ser computacionalmente inviable encontrar cualquier mensaje x' (no necesariamente igual a x) tal que $h^s(x') = h$.

Nótese que, a diferencia de la resistencia a colisiones, aquí el adversario no necesita encontrar dos mensajes diferentes que produzcan el mismo hash, sino simplemente encontrar un mensaje que produzca un hash específico dado. Esta propiedad es crucial para aplicaciones como el almacenamiento seguro de contraseñas, donde un atacante no debería poder determinar la contraseña original a partir de su valor hash almacenado.