Intercambio de claves. Método de Diffie-Hellman.

Alan Reyes-Figueroa

Teoría de Números (Aula 24a) 30.septiembre.2024

Logaritmo Discreto

Dado un primo $p > 2$, tomemos $\mathbf{g} \in (\mathbb{Z}/p\mathbb{Z})^*$, un elemento de orden q, esto es

$$\mathbf{g}^q \equiv 1 \pmod{p}, \qquad \text{pero } \mathbf{g}^j \not\equiv 1 \pmod{p}, \text{ para } 1 \leq j < q.$$

Ya vimos que calcular potencias módulo p es una tarea relativamente simple. Veamos ahora el problema inverso.

Dada la función $\mathbf{x} \longrightarrow \mathbf{g}^{\mathbf{x}} = \mathbf{y} \pmod{p}$, consideramos la función inversa

$$\log_{\mathbf{g}}(\mathbf{y}) = \log_{\mathbf{g}}(\mathbf{y}) = \mathbf{x}, \;\; \text{donde } \mathbf{x} \in \{0, 1, \ldots, q-1\}.$$

Esta función se llama el **logaritmo discreto** con base **g** de **y**, módulo p.

Ejemplo: Para $p = 11$

a	1	2	3	4	5	6	7	8	9	10
$\log_2 a$	0	1	8	2	4	9	7	3	6	5

Definición

*Un **grupo** G es un conjunto no vacío de elementos, junto con una operación binaria $\cdot : G \times G \to G$ que satisface*

1. *(asociatividad)* $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
2. *(elemento neutro) existe* $e \in G$ *tal que* $(a \cdot e) = e \cdot a = a$*, para todo* $a \in G$.
3. *(elementos inversos) para todo* $a \in G$*, existe* $b \in G$ *tal que* $a \cdot b = b \cdot a = e$.

Si adicionalmente G cumple con

4. *(conmutatividad)* $a \cdot b = b \cdot a$*, para todo* $a, b \in G$

*entonces G se llama un **grupo conmutativo**.*

Ejemplos: $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$, $\mathbb{Z}/p\mathbb{Z}$
$\mathbb{R}^n$, $\mathbb{R}^{m \times n}$, $F(a, b)$, $C(a, b)$,
S_n es el grupo de permutaciones de n elementos. se **g** de **y**, módulo p.

Dado un grupo **cíclico** y un generador $\mathbf{g}$ para el grupo G:

$$G = \{1, \mathbf{g}, \mathbf{g}^2, \mathbf{g}^3, \mathbf{g}^4, \ldots, \mathbf{g}^{q-2}, \mathbf{g}^{q-1}\}.$$

(recordemos que q es el orden de $\mathbf{g}$).

Definición

Decimos que el problema del logaritmo discreto $\log_{\mathbf{g}} \mathbf{y} = \log_{\mathbf{g}}(\mathbf{g}^{\mathbf{x}}) = \mathbf{x}$ *es* ***difícil*** *en G si para todo algoritmo eficiente $\mathcal{A}$, se cumple*

$$\mathbb{P}_{\mathbf{g} \leftarrow G,\ \mathbf{x} \leftarrow \mathbb{Z}/q\mathbb{Z}}[\mathcal{A}(G, q, \mathbf{g}, \mathbf{y}) = \mathbf{x}] < \varepsilon$$

es negligible.

Ejemplos:

- $U(p)$ grupos de unidades módulo p, para p muy grande.
- Grupos de curvas elípticas módulo p.

Intercambio de Claves

En esta aula veremos protocolos para el intercambio de claves **k** de modo que dos entes puedan iniciar un esquema de intercambio de información segura.

Observaciones:

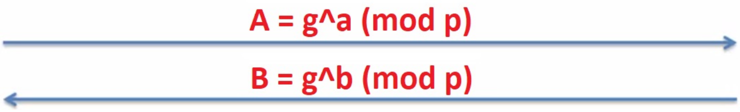
- En este caso nos centramos únicamente en el problema de intercambiar una clave y evita que sea leída por terceros. No se analiza el problema de autenticación, esto es, verificar que no ha ocurido modificación de información.
- Existen protocolos de intercambio de claves basados en funciones hash (*e.g.* SHA-256), que se conocen tomo *Merkle puzzles*. Sin embargo, éstos se consideran demasiando inseguros desde el punto de vista práctico.

Método de Diffie-Hellman

Desarrollado por WHITFIELD DIFFIE y MARTIN HELLMAN en 1975, publicado en “New Directions in Cryptography”, IEEE Transactions on Information Theory. 22 (6): 644–654.

- Fijamos un primo p grande (600 o más dígitos, alrededor de 2000 bits).
- Fijamos un entero $g \in \{1, 2, \ldots, p-1\}$. Así, (p, g) se vuelven parámetros del método de Diffie-Hellman. Una vez elegidos, no cambian.

- Alice elige un entero aleatorio $\mathbf{a} \in \{1, 2, \ldots, p-1\}$.
- Bob elige un entero aleatorio $\mathbf{b} \in \{1, 2, \ldots, p-1\}$.
- Alice envía $A = g^{\mathbf{a}} \pmod{p}$, y Bob envía $B = g^{\mathbf{b}} \pmod{p}$.



Ahora, ambos comparten la clave secreta $\mathbf{k} = g^{\mathbf{ab}} \pmod{p}$.

Método de Diffie-Hellman

Basta con que Alice calcule (Alice conoce cuánto vale **a** y B):

$$B^{\mathbf{a}} \equiv (g^{\mathbf{b}})^{\mathbf{a}} \equiv g^{\mathbf{ab}} \pmod{p}.$$

Similarmente, basta que Bob calcule (Bob conoce cuánto vale **b** y A):

$$A^{\mathbf{b}} \equiv (g^{\mathbf{a}})^{\mathbf{b}} \equiv g^{\mathbf{ab}} \pmod{p}.$$

De estos elementos, sólo **a** y **b** son secretos:

- p = primo (público), , conocido por Alice, Bob, y cualquier otro.
- g = generador base (público), conocido por Alice, Bob, y cualquier otro.
- **a** = llave privada de Alice.
- **b** = llave privada de Bob.
- A = llave pública de Alice, conocida por Alice, Bob, y cualquier otro.
- B = llave pública de Bob's, conocida por Alice, Bob, y cualquier otro.

Seguridad

¿Por qué este método es seguro?

Cualquier persona conoce $p, g, A = g^{\mathbf{a}} \pmod{p}$ y $B = g^{\mathbf{b}} \pmod{p}$.

¿Qué tan fácil es calcular $\mathbf{k} = g^{\mathbf{ab}}$ a partir de estos datos?

Más generalmente, definimos la función $DH_g(g^{\mathbf{a}}, g^{\mathbf{b}}) = g^{\mathbf{ab}} \pmod{p}$. Qué tan compleja de la función DH?

Supongamos que el primo p es de longitud n bits. Actualmente los mejores algoritmos (GNFS, *General number field sieve*) para calcular g^{ab} corren en tiempo $\exp(\widetilde{O}(\sqrt[3]{n}))$.

Tamaño de la clave	Tamaño del módulo
80 bits	1024 bits
128 bits	3072 bits
256 bits (AES)	15360 bits