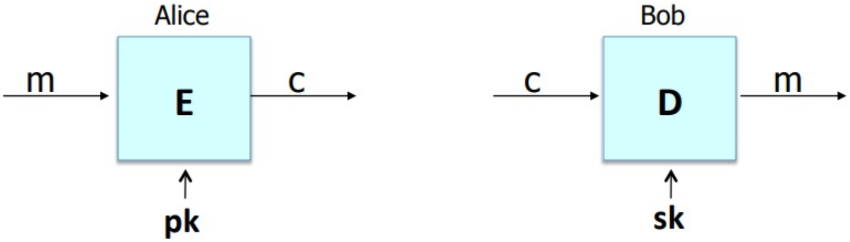
Criptografía de Clave Pública

Alan Reyes-Figueroa

Teoría de Números (Aula 24b) 30.septiembre.2023

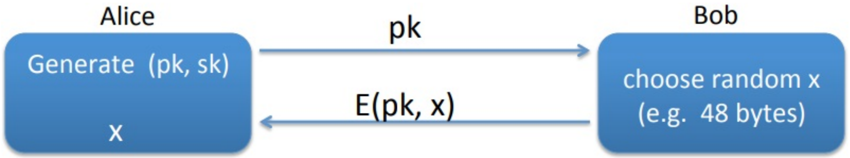
Clave Pública

Idea básica: Bob genera un par (**pk**, **sk**), y le entrega a Alice **pk**.



Esquema básico de la criptografía de clave pública.

Aplicaciones: Configuración de sesión (por ahora, sólo se hablará de seguridad de escucha).



Esquema para inicio de sesión.

Otras aplicaciones:

- Aplicaciones no interactivas: (*e.g.*, correo electrónico).
- Bob envía un correo electrónico a Alice encriptado usando $\mathbf{pk}_{Alice}$
- Bob necesita $\mathbf{pk}_{Alice}$ (administración de claves públicas).

Definición

*Un sistema de **cifrado de clave pública** es una tripla de algoritmos* $\mathcal{E} = (G, E, D)$*, donde*

- *G: es un algoritmo aleatorizado. Emite un par de claves* $(\mathbf{pk}, \mathbf{sk})$.
- $E(\mathbf{pk}, \mathbf{m})$ *es un algoritmo aleatorizado que toma* $\mathbf{m} \in \mathcal{M}$ *y produce* $\mathbf{c} \in \mathcal{C}$
- $D(\mathbf{sk}, \mathbf{c})$ *es un algoritmo determinista que toma* $\mathbf{c} \in \mathcal{C}$ *y produce* $\mathbf{m} \in \mathcal{M}$.

el cual satisface la condición de consistencia:

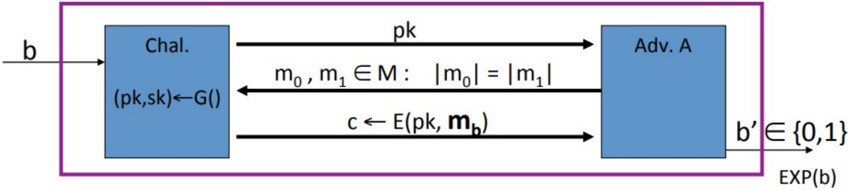
$$\forall\, \mathbf{m} \in \mathcal{M},\ \forall (\mathbf{pk}, \mathbf{sk}) \text{ producido por } G \quad \Longrightarrow \quad D(\mathbf{sk}, E(\mathbf{pk}, \mathbf{m})) = \mathbf{m}.$$

Ejemplos: Veremos dos de los métodos más populares de clave pública

- RSA,
- ElGamal.

Mostramos cómo funciona la seguridad en contra de escuchas en clave pública.

Para $b = 0, 1$, definamos experimentos $EXP(0)$ y $EXP(1)$ de la siguiente forma:



Definición

*Diremos que el sistema de cifrado $\mathcal{E} = (G, E, D)$ es **semánticamente seguro** (a.k.a IND-CPA), si para todo algoritmos eficiente $\mathcal{A}$, se tiene que*

$$\text{Adv}_{SS}(\mathcal{A}, \mathcal{E}) = \big|\mathbb{P}[EXP(0) = 1] - \mathbb{P}[EXP(1) = 1]\big| < \varepsilon,$$

con ε negligible.

Seguridad de Clave Pública

Recordemos que en los cifrados simétricos: para los cifrados simétricos teníamos dos nocions de seguridad:

- Seguridad *one-time* y seguridad *many-time* (CPA).
- Demostramos que la seguridad *one-time* $\Longrightarrow$ seguridad *many-time*.

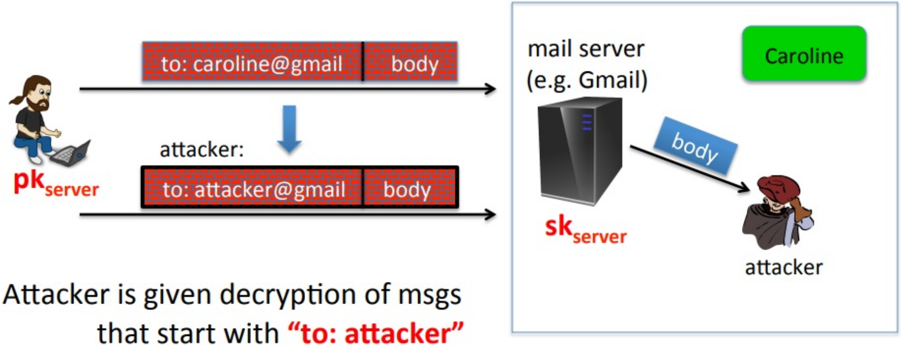
Para los cifrados de clave pública:

- Seguridad *one-time* $\Longrightarrow$ seguridad *many-time* (CPA).
  (se deduce del hecho de que hacker puede cifrar por sí mismo).
- El cifrado de clave pública **debe ser aleatorio**.

  Esto es importante porque en muchos libros y blogs se describen los métodos de clave pública sin una parte aleatoria (imagino, por razones de simplicidad). Sin embargo, sin la parte aleatoria, los métodos no son seguros.

Seguridad de Clave Pública

Seguridad contra ataques activos: ¿Qué ocurre si un atacante quiere alterar o modificar el texto cifrado?

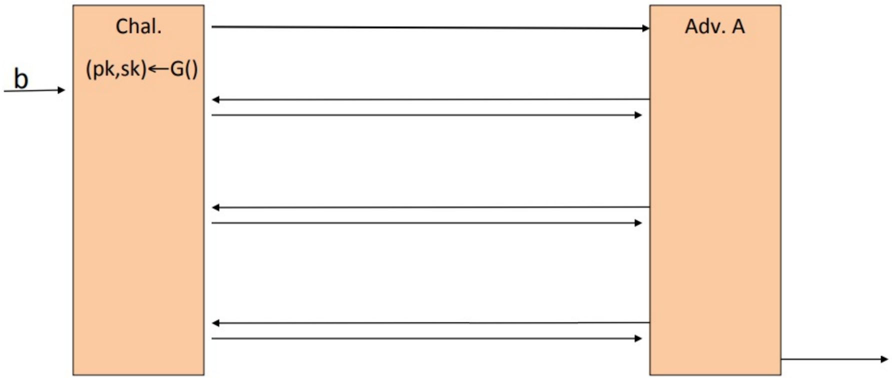


El atacante recibe la descripción de varios mensajes que comienzan con "to: attacker".

Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

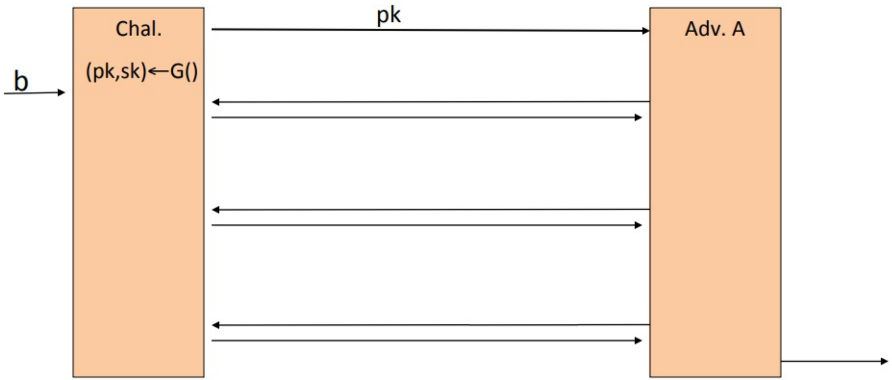
$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:



Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:

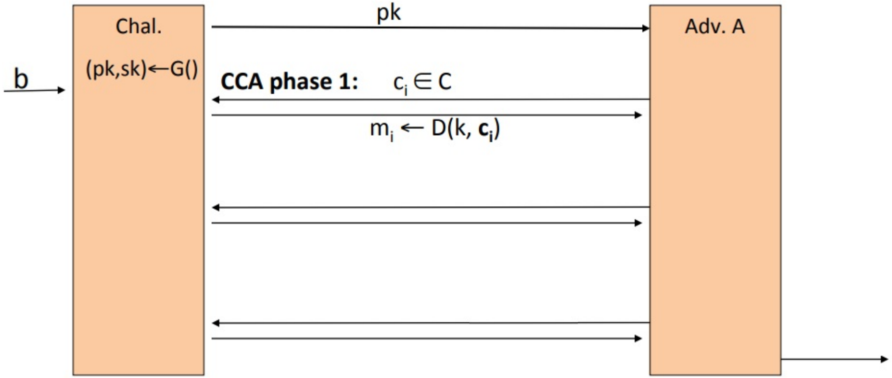


UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

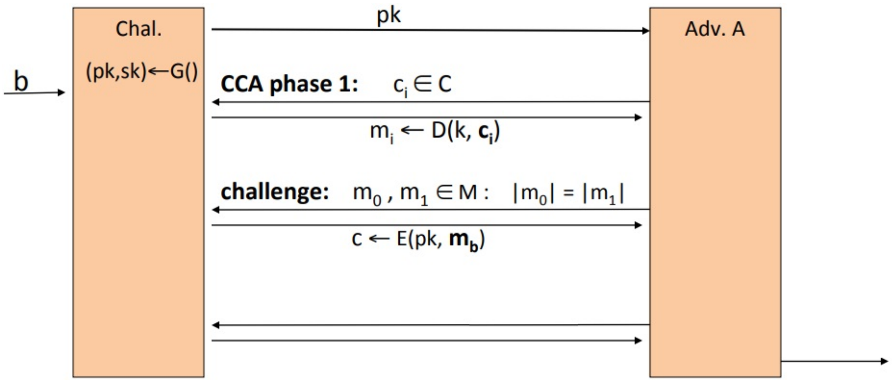
$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:



Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

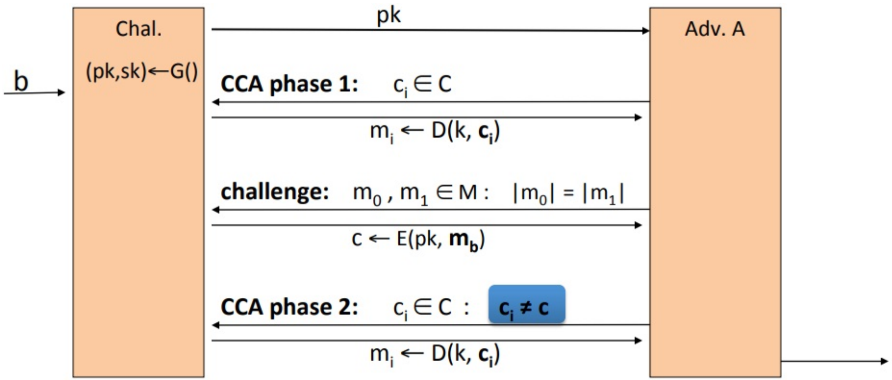
$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:



Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

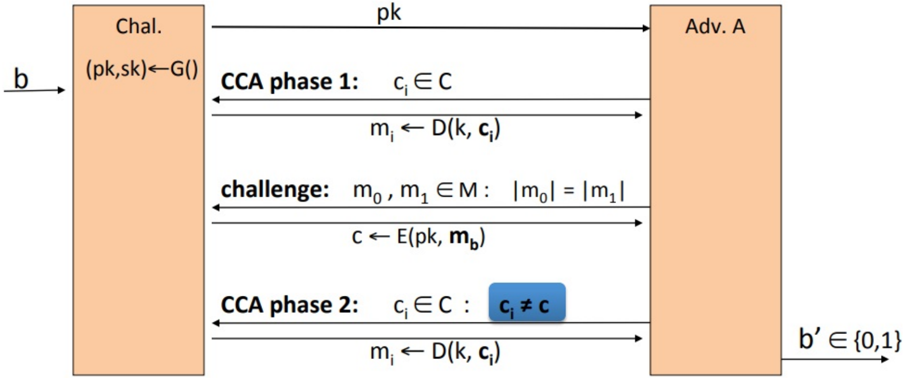
$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:



Seguridad de Clave Pública

Definición de seguridad de cifrado por elección (*chosen ciphertext security*).

$\mathcal{E} = (G, E, D)$ cifrado de clave pública sobre $(\mathcal{M}, \mathcal{C})$. Para $b = 0, 1$, se define $EXP(b)$:



Definición

El cifrado $\mathcal{E} = (G, E, D)$ es ***CAA seguro*** *(a.k.a IND-CCA) si para todo algoritmo eficiente $\mathcal{A}$ se tiene*

$$\mathrm{Adv}_{CCA}(\mathcal{A}, \mathcal{E}) = \Big|\mathbb{P}[EXP(0) = 1] - \mathbb{P}[EXP(1) = 1]\Big| \text{ es negligible.}$$

Ejemplo: Suponga que un atacante cambia el destinatario de Alice a David.

Seguridad de Clave Pública

De nuevo: en los cifrados simétricos seguros, éstos proporcionan mecanismoa de encripción autenticada, esto es, seguridad del texto plano e integridad del texto cifrado a elección (de cambios).

- Hablando en términos generales: un atacante no puede crear nuevos textos cifrados.
- Esto implica seguridad contra ataques de texto cifrado elegidos.

En contraste, en los esquemas de clave pública, tenemos:

- Un atacante puede crear nuevos textos cifrados usando **pk**.
- Entonces, en su defecto, requerimos directamente el concepto de seguridad de texto cifrado por elección.

Veremos a continuación un primer ejemplo de cómo construir un sistema CCA seguros de clave pública.

Funciones Trapdoor

Definición

*Una **función de trampilla** (trapdoor function $f : X \to Y$ es un tripla de algoritmos eficientes (G, F, F^{-1}), donde*

- *G es un algoritmo aleatorizado. Emite un par de claves $(\mathbf{pk}, \mathbf{sk})$.*
- *$F(\mathbf{pk}, \cdot) : X \to Y$ es un algoritmo determinista que define una función $X \to Y$.*
- *$F^{-1}(\mathbf{sk}, \cdot) : Y \to X$ define una función $Y \to X$ que invierte $F(\mathbf{pk}, \cdot)$, esto es*

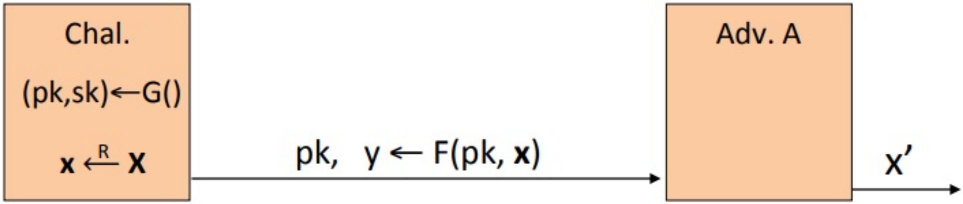
Más precisamente, para todo par de claves $(\mathbf{pk}, \mathbf{sk})$ emitido por G, se tiene que

$$F^{-1}\big(\mathbf{sk}, F(\mathbf{pk}), \mathbf{x})\big) = \mathbf{x}, \quad \forall \mathbf{x} \in X.$$

No todas las funciones trapdoor son seguras. Intuitivamente, para que una función trapdoor (G, F, F^{-1}) sea segura, debe ser una función de *una vía*:

Se puede evaluar F, pero no se puede invertir, sin conocer **sk**.

Funciones Trapdoor



Esquema de una función trapdoor segura.

Definición

*La función trapdoor $f = (G, F, F^{-1})$ es **TDF segura**, si para todo algoritmo eficiente $\mathcal{A}$, vale*

$$\text{Adv}_{OW}(\mathcal{A}, F) = \mathbb{P}(\mathbf{x} = \mathbf{x}') < \varepsilon,$$

con ε es negligible.

Funciones Trapdoor

Es posible construir mecanismos de cifrado de clave pública seguros, a partir de TDF seguras, de la siguiente forma

- $f = (G, F, F^{-1})$ es una TDF segura $f : X \to Y$,
- (Es, Ds) es un cifrado de autenticación simétrica definido sobre $(\mathcal{K}, \mathcal{M}, \mathcal{C})$,
- $H : X \to \mathcal{K}$ una función hash.

Entonces, construimos un sistema de escripción de clave pública $\mathcal{E} = (G, E, D)$:

- Generación de claves para G: igual que para la TDF.

E(pk, m) :

$x \xleftarrow{R} X, \quad y \leftarrow F(pk, x)$

$k \leftarrow H(x), \quad c \leftarrow E_s(k, m)$

output (y, c)

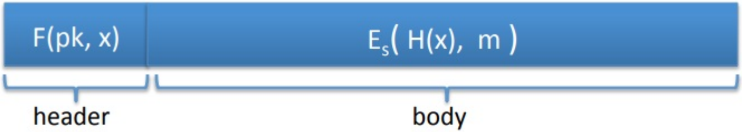
D(sk, (y,c)) :

$x \leftarrow F^{-1}(sk, y),$

$k \leftarrow H(x), \quad m \leftarrow D_s(k, c)$

output m

En figuras:



Teorema (Teorema de Seguridad)

Si $f = (G, F, F^{-1})$ es una TDF segura, (E_s, D_s) es un mecanismo de autenticación encriptada, y $H : X \to K$ es un "oráculo aleatorio" entonces $\mathcal{E} = (G, E, D)$ es CCA seguro.

Obs! Uso incorrecto de una TDF. Nunca cifrar aplicando F directamente al texto plano:

E(pk, m) :
output $\quad c \longleftarrow F(pk, m)$

D(sk, c) :
output $\quad F^{-1}(sk, c)$

Problemas: No es semánticamente seguro! Existen muchos ataques.

El método RSA: (Rivest, Shamir, Adleman)
Publicado en *Scientific American*, agosto de 1977.
Muy utilizado:

- SSL / TLS: certificados e intercambio de claves,
- correo electrónico seguro y sistemas de archivos,
- muchas otras aplicaciones.

La idea base es la siguiente:
Tomamons $N = pq$, donde p, q son primos distintos. Recodremos que la función φ de Euler de N, cuenta el número de elementos distintos en $U(N)$:

$$\varphi(N) = \varphi(pq) = \varphi(p)\varphi(q) = (p-1)(q-1) = pq - p - q + 1 = N - p - q + 1.$$

Si $\mathbf{x} \in \mathbb{Z}$ es un elemento invertible módulo N (esto es $\mathbf{x} \in U(N)$, el teorema de Euler-Fermat dice que $\mathbf{x}^{\varphi(N)} \equiv 1 \pmod{N}$.

¿Cómo funciona el RSA?

1. $G()$: Elegir primos aleatorios p, q, distintos, de aproximadamente 1024 bits (o más).
   Establecer $N = pq$.
   Elegir enteros $e, d \in \mathbb{Z}$, tales que $ed \equiv 1 \pmod{\varphi(N)}$.

   Con ello construimos: $\mathbf{pk} = (N, e)$, y $\mathbf{sk} = (N, d)$.

2. Definimos las funciones

$$F(\mathbf{pk}, \mathbf{x}) : U(N) \to U(N), \qquad RSA(\mathbf{x}) = \mathbf{x}^e \pmod{N}.$$

3. Definimos también

$$F^{-1}(\mathbf{sk}, \mathbf{y}) = \mathbf{y}^d, \qquad \mathbf{y}^d = RSA(\mathbf{x})^d = \mathbf{x}^{ed} = \mathbf{x}^{k\varphi(N)+1} = \left(\mathbf{x}^{\varphi(N)}\right)^k \cdot \mathbf{x} \equiv 1 \cdot \mathbf{x} \equiv \mathbf{x} \pmod{N}.$$

El método RSA se basa en el siguiente supuesto:

Lema (Supuesto básico de RSA)

$RSA : U(N) \to U(N)$ es una permutación de una vía.

Esto es, RSA no es invertible, a menos que se conozca la clave secreta **sk**.

En particular, vale que para todo algoritmos eficiente $\mathcal{A}$,

$$\mathbb{P}(\mathcal{A}(N, e, y) = y^{1/e}] < \varepsilon,$$

con ε negligible, donde p, q son primos de n-bits, distintos, $N = pq$, y **y** es un valor aleatorio en $U(N)$.

RSA

Algoritmo: (RSA Escripción de clave pública, ISO estándar).

Inputs: (E_s, D_s): esquema de encripción simétrica, con autenticación (e.g. AES).
$H : \mathbb{Z}/N\mathbb{Z} \to \mathcal{K}$, función hash, donde $\mathcal{K}$ es el espacio de claves de (E_s, D_s)

- $G()$: generar los parámetros RSA: $\mathbf{pk} = (N, e)$, $\mathbf{sk} = (N, d)$.
- $E(\mathbf{pk}, \mathbf{m})$:
  1. Elegir aleatoriamente $\mathbf{x} \in \mathbb{Z}/N\mathbb{Z}$,
  2. Definir $\mathbf{y} = RSA(\mathbf{x}) = \mathbf{x}^e$, $\mathbf{k} = H(\mathbf{x})$,
  3. Return $(\mathbf{y}, E_s(\mathbf{k}, \mathbf{m}))$.
- $D(\mathbf{sk}, (\mathbf{y}, \mathbf{c}))$: Return $D_s(H(RSA^{-1}(\mathbf{y})), \mathbf{c})$.