Criptografía y Seguridad Computacional **2016-01**

Clase 8: 20/04/2016

Profesor: Fernando Krell *Notas: Francisco Maturana*

1. Supuestos criptográficos no genericos

1.1. Factorizar es difícil

Sea:

$$\mathsf{GenMod}(1^\lambda) \to N, P, Q \text{ tal que } P, Q \text{ primos y } N = PQ$$

Se define el experimento 1, $\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}$. Luego, el supuesto es que existe GenMod tal que para todo PPT $\mathcal{A}$ se cumple $\Pr[\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$.

$\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(\lambda)$

1. $N, P, Q \leftarrow \mathsf{GenMod}(1^\lambda)$
2. $P', Q' \leftarrow \mathcal{A}(N)$
3. output $1 \Leftrightarrow P'Q' = N$

Figura 1: Experimento $\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}$

1.2. El supuesto RSA

Sea:

$$\begin{aligned}\mathsf{GenRSA}(1^\lambda) \to N, e, d \text{ tal que } N = PQ, \text{ donde } P, Q \text{ son primos}\\ \text{y } e \in \mathbb{Z}^*_{\Phi(N)},\ d = e^{-1} \mod \Phi(N),\\ \Phi(N) = (P-1)(Q-1)\end{aligned}$$

Se define el experimento 2, $\mathsf{ExpRSA}_{\mathcal{A},\mathsf{GenRSA}}$. Luego, el supuesto es que existe GenRSA tal que para todo PPT $\mathcal{A}$ se cumple $\Pr[\mathsf{ExpRSA}_{\mathcal{A},\mathsf{GenRSA}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$.

El algoritmo GenRSA puede ser construido a partir del algoritmo GenMod como se detalla en el algoritmo 1.

$\mathsf{ExpRSA}_{\mathcal{A},\mathsf{GenRSA}}(\lambda)$

1. $N, e, d \leftarrow \mathsf{GenRSA}(1^\lambda)$
2. $y \leftarrow \mathbb{Z}_N^*$
3. $x \leftarrow \mathcal{A}(N, e)$
4. output $1 \Leftrightarrow x^e = y \mod N$

Figura 2: Experimento $\mathsf{ExpRSA}_{\mathcal{A},\mathsf{GenRSA}}$

Algoritmo 1 $\mathsf{GenRSA}(1^\lambda)$

$N, P, Q \leftarrow \mathsf{GenMod}(1^\lambda)$
$\Phi(N) = (P-1)(Q-1)$
$e \leftarrow \mathbb{Z}^*_{\Phi(N)}$
$d \leftarrow e^{-1} \mod \Phi(N)$
output N, e, d

1.3. Supuestos en grupos cíclicos

Sea $\mathbb{G}$ un grupo y $g \in \mathbb{G}$. Definimos $\langle g \rangle = g^0, g^1, g^2, \ldots$. Diremos que el orden de g es el mínimo $i > 0$ tal que $g^i = 1$. En otras palabras, el número de elementos distintos en $\langle g \rangle$.

Proposición 1. *Sea el grupo $\mathbb{G}$ y $g \in \mathbb{G}$. Si $g^x = g^y$ entonces $x = y \mod i$, donde i es el orden de g.*

Demostración.

$$g^x = g^y \Leftrightarrow g^{x-y} = 1 = g^{x-y} \mod i$$

Sabemos que $x - y \mod i < i$ por definición de reducción modular, pero i es el mínimo tal que $g^i = 1$. Por lo tanto, $x - y \mod i = 0$. □

Definición 2. *Sea q el orden de $\mathbb{G}$. Si $|\langle g \rangle| = q$ entonces diremos que g es un generador de $\mathbb{G}$. Es decir, para todo $h \in \mathbb{G}$ tendremos que existe un $x \in \{0, \ldots, q-1\}$ tal que $g^x = h$.*

Proposición 3. *Sea el grupo $\mathbb{G}$ de orden q. Si $|\langle g \rangle| = i$, entonces $i|q$.*

Demostración.

$$g^q = 1 = g^0 \Rightarrow q = 0 \mod i$$

□

Corolario 4. *Si $\mathbb{G}$ es de orden primo p, entonces $\mathbb{G}$ is cíclico y todo $g \in \mathbb{G} \setminus \{1\}$ es generador.*

1.4. Logaritmo Discreto

Dado un grupo $\mathbb{G}$ de orden q, g el generador de $\mathbb{G}$ y g^x, el problema del logaritmo discreto es encontrar x. El supuesto del logaritmo discreto es que existe un grupo $\mathbb{G}$ (o más bien que existe un un algoritmo $\mathcal{G}$ que nos entrega $\mathbb{G}$ y un generador g) tal que el problema del logaritmo discreto es difícil. Para formalizar esto, definimos el experimento 3.

$\mathsf{DLog}_{\mathcal{A},\mathcal{G}}(\lambda)$

1. $\mathbb{G}, q, g \leftarrow \mathcal{G}1^\lambda$
2. $x \xleftarrow{\$} \mathbb{Z}_{q-1}$
3. $x' \leftarrow \mathcal{A}(\mathbb{G}, q, g, q^x)$
4. output $1 \Leftrightarrow g^{x'} = g^x$

Figura 3: Experimento $\mathsf{DLog}_{\mathcal{A},\mathcal{G}}$

Luego, consideraremos el supuesto de que existe $\mathcal{G}$ tal que $\Pr[\mathsf{DLog}_{\mathcal{A},\mathcal{G}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$.

1.5. Problemas de Diffie-Hellman

1.5.1. Diffie Hellman Computacional (CDH)

Dado un grupo $\mathbb{G}$ de orden q, $g \in \mathbb{G}$, g^x, g^y, el problema de Diffie-Hellman computacional es computar computar $g^{x \cdot y}$. El supuesto CDH, es que existe $\mathbb{G}$ tal que computar $g^{x \cdot y}$ es difícil (i.e. probabilidad negligible en rtiempo polinomial para cualquier algoritmo)

1.5.2. Diffie Hellman Desicional (DDH)

Dado un grupo $\mathbb{G}$ de orden q, $g \in \mathbb{G}$, g^x, g^y, T, en donde x, y son uniformes en $\mathbb{Z}_q$, el problema de Diffie-Hellman desicional es distinguir si T es uniforme en $\mathbb{G}$ o $T = g^{x \cdot y}$. El supuesto es que distinguir en tiempo polinomial es difícil para ciertos grupos, y por lo tanto, el elemento $g^{x \cdot y}$ es pseudo-aleatorio.

Claramente si DLog es fácil, tambien lo es CDH, y si CDH es fácil, también lo es DDH. Decimos entonces que DLog es un supuesto mas débil que CDH, y a la vez, CDH es mas débil que DDH.

1.5.3. Funciones de hash resistentes a colisiones

Una aplicación del supuesto DLog es la creación de funciones de hash resistentes a colisiones, como muestra el algoritmo 2.

Algoritmo 2 Función de hash resistente a colisiones

$\mathsf{Gen}(1^\lambda)$:
  $\mathbb{G}, q, g \leftarrow \mathcal{G}(1^\lambda)$
  $h \xleftarrow{\$} \mathbb{G}$
  output $(\mathbb{G}, q, g, h)$

$H^s(x_1||x_2)$:
  $x_1, x_2 \in \mathbb{Z}_q$
  output $g^{x_1} h^{x_2}$

Para ver que esta función es resistente a colisiones, reduciremos al problema del logaritmo discreto. Sea el adversario $\mathcal{A}$ que encuentra Y_1, Y_2 tal que $H^s(Y_1) = H^s(Y_2)$. Tenemos que:

$$Y_1 \to Y_{11}||Y_{12} \qquad Y_2 \to Y_{21}||Y_{22}$$

$$\begin{aligned} H^s(Y_1) &= H^s(Y_2) \\ &\Leftrightarrow \\ g^{Y_{11}}h^{Y_{12}} &= g^{Y_{21}}h^{Y_{22}} \\ &\Leftrightarrow \\ g^{(Y_{11}-Y_{21})} &= h^{(Y_{22}-Y_{12})} \\ &\Leftrightarrow \\ g^{(Y_{11}-Y_{21})(Y_{22}-Y_{12})^{-1}} &= h = g^x \\ &\Leftrightarrow \\ x &= (Y_{11}-Y_{21})(Y_{22}-Y_{12})^{-1} \mod q \end{aligned}$$

Por lo tanto, si el problema de logaritmo discreto es difícil, entonces esta función será resistente a colisiones.

2. Grupos de orden primo

El problema del logaritmos discreto y los problemas de Diffie-Hellman son más difíciles en grupos de orden primo. Además, encontrar un generador en tales grupos es trivial.¿Cómo generamos estos grupos?

Supongamos que tenemos un grupo $\mathbb{Z}_p^*$. Sin embargo, $|\mathbb{Z}_p^*| = p-1$ no es primo. Vamos ahora a encontrar un subgrupo de $\mathbb{Z}_p^*$ de orden primo cuando es de cierta forma especial.

Teorema 5. *Sea $p = rq+1$ con p y q primos y $r \in \mathbb{N}$. El grupo $\mathbb{G} = \{h^r \mod p \mid h \in \mathbb{Z}_p^*\}$ es un subgrupo de $\mathbb{Z}_p^*$ de orden q.*

Demostración. Recordemos que si $a \in \mathbb{G}$ y $b \in \mathbb{G}$, entonces $ab \in \mathbb{G} \Rightarrow \mathbb{G}$ es un grupo.

Sea $a, b \in \mathbb{G}$, entones exiten $h_a, h_b \in \mathbb{Z}_p^*$ tales que

$$a = h_a^r \mod p, \qquad b = h_b^r \mod p$$

Por lo tanto,

$$ab = (h_a h_b)^r \mod p \quad \Rightarrow \quad ab \in \mathbb{G}$$

Ahora, demostraremos que el tamaño de $\mathbb{G}$ es q. Consideremos, la función f_r : $f_r(g) = g^r \mod p$. Notemos que para un generador g, el grupo $\mathbb{Z}_p^*$ se puede describir

como $\{g^0, g^1, \ldots, g^{p-2}\}$. Sean $g^i \neq g^j \neq 1$ elementos de $\mathbb{Z}_p$.

$$\begin{aligned}(g^i)^r = (g^j)^r \text{ si } ir = jr \mod p-1 \\ \Leftrightarrow (p-1) \mid (ir - jr) \\ \Leftrightarrow q \mid (i-j)\end{aligned}$$

Fijemos $j \in \{0, 1, \ldots, p-2\}$. ¿Cuáles son los i tal que $(g^i)^r = (g^j)^r$ o, más bien, $q \mid (i-j)$? Esos son $q+j, 2q+j, \ldots, rq+j$. Por lo tanto, tenemos r elementos para cada j. Esto implica que la función f_r mapea r elementos distintos de $\mathbb{Z}_p^*$ en un único elemento en $\mathbb{G}$, y la imagen de $g^0 = 1$ es 1. Por lo tanto, el orden de $\mathbb{G}$ es $q = (p-1)/r$ □

3. Cifradores de llave pública

$$\begin{aligned}\mathsf{Gen}(1^\lambda) &\to (pk, sk) \\ \mathsf{Enc}_{pk}(m) &\to c \\ \mathsf{Dec}_{sk}(c) &= m \text{ o } \perp \text{ con baja probabilidad.}\end{aligned}$$

Definimos el experimento 4 $\mathsf{Pub}^{\mathsf{ind}}_{\mathcal{A},\Pi}$ de llave pública.

$\mathsf{Pub}^{\mathsf{ind}}_{\mathcal{A},\Pi}(\lambda)$

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$
2. $b \xleftarrow{\$} \{0,1\}$
3. $m_0, m_1 \leftarrow \mathcal{A}(pk)$
4. $b' \leftarrow \mathcal{A}(\mathsf{Enc}_{pk}(m_b))$
5. output 1 si $b' = b$

Figura 4: Experimento $\mathsf{Pub}^{\mathsf{ind}}_{\mathcal{A},\Pi}$

Diremos que Π tiene cifrados indistinguibles si para todo PPT $\mathcal{A}$ tenemos que:

$$\Pr[\mathsf{Pub}^{\mathsf{ind}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}\lambda$$

Teorema 6. Π *tiene cifrados indistinguibles si y solo si* Π *es cpa-seguro.*

Demostración. Trivial, si $\mathcal{A}$ tiene pk, entonces tiene acceso al oráculo. □

También definimos el experimento 5 $\mathsf{Pub}^{\mathsf{DI-CPA}}_{\mathcal{A},\Pi}$ para múltiples mensajes. Es posible demostrar que si Π es cpa-seguro, entonces es seguro frente a múltiples mensajes.

$\mathsf{Pub}^{\mathsf{DI-CPA}}_{\mathcal{A},\Pi}(\lambda)$

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$
2. $b \xleftarrow{\$} \{0,1\}$
3. $b' \leftarrow \mathcal{A}^{\mathsf{Enc}_{pk,b}(\cdot,\cdot)}(pk)$
4. output $1 \Leftrightarrow b = b'$

donde $\mathsf{Enc}_{pk,b}(m_0, m_1)$ retorna $\mathsf{Enc}_{pk}(m_b)$.

Figura 5: Experimento $\mathsf{Pub}^{\mathsf{DI-CPA}}_{\mathcal{A},\Pi}$

4. Cifrador de El Gamal

El cifrador de El Gamal sigue el algoritmo 3.
¿Es este esquema seguro?

Proposición 7. *Si DDH se cumple en* $\mathbb{G}$*, entonces El Gamal es cpa-seguro.*

Demostración. Sea $\mathcal{A}$ un adversario exitoso para El Gamal en el sentido CPA (es decir, gana en el experimento con probabilidad $1/2 + 1/\text{poly}(\lambda)$. Definimos el siguiente distinguidor para $\mathbb{G}, q, g, g^x, g^y, T$. Sea:

$$pk = \mathbb{G}, q, g, g^x$$
$$\mathcal{A}(pk) \to m_0, m_1$$
$$b \xleftarrow{\$} \{0,1\}$$
$$b' \leftarrow \mathcal{A}(g^y, m_b T)$$
$$\text{output } 1 \Leftrightarrow b = b'$$

Algoritmo 3 Cifrador de El Gamal

$\mathsf{Gen}(1^\lambda)$:
$(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^\lambda)$
$x \leftarrow \mathbb{Z}_q$
$h = g^x$
$pk = (\mathbb{G}, q, g, h)$
$sk = (\mathbb{G}, q, g, x)$

$\mathsf{Enc}(pk, m)$:
$y \leftarrow \mathbb{Z}_q$
output (g^y, mh^y)

$\mathsf{Dec}_{sk}(c = \langle c_1, c_2 \rangle)$:
$h^y = (g^x)^y = g^{xy}$
output $c_2(c_1^x)^{-1}$

Si T es uniforme, entonces $\Pr[b = b'] = \frac{1}{2}$. Si T es g^{xy} entonces $\Pr[b = b'] = \Pr[\mathsf{Pub}^{\mathsf{ind}}_{\mathcal{A},\Pi}(\lambda) = 1] = \frac{1}{2} + \frac{1}{\mathrm{poly}(\lambda)}$. Concluimos entonces que El Gamal es CPA seguro. □

5. Cifrador RSA plano

El cifrador RSA sigue el algoritmo 4.

Algoritmo 4 Cifrador RSA

$\mathsf{Gen}(1^\lambda)$:
$N, e, d \leftarrow \mathsf{GenRSA}(1^\lambda)$
$pk = N, e$
$sk = N, d$

$\mathsf{Enc}_{pk}(m)$:
output $m^e \mod N$

$\mathsf{Dec}_{sk}(c)$:
output $c^d \mod N$

¿Es este algoritmo seguro frente a múltiples mensajes? No, ya que es determinista.