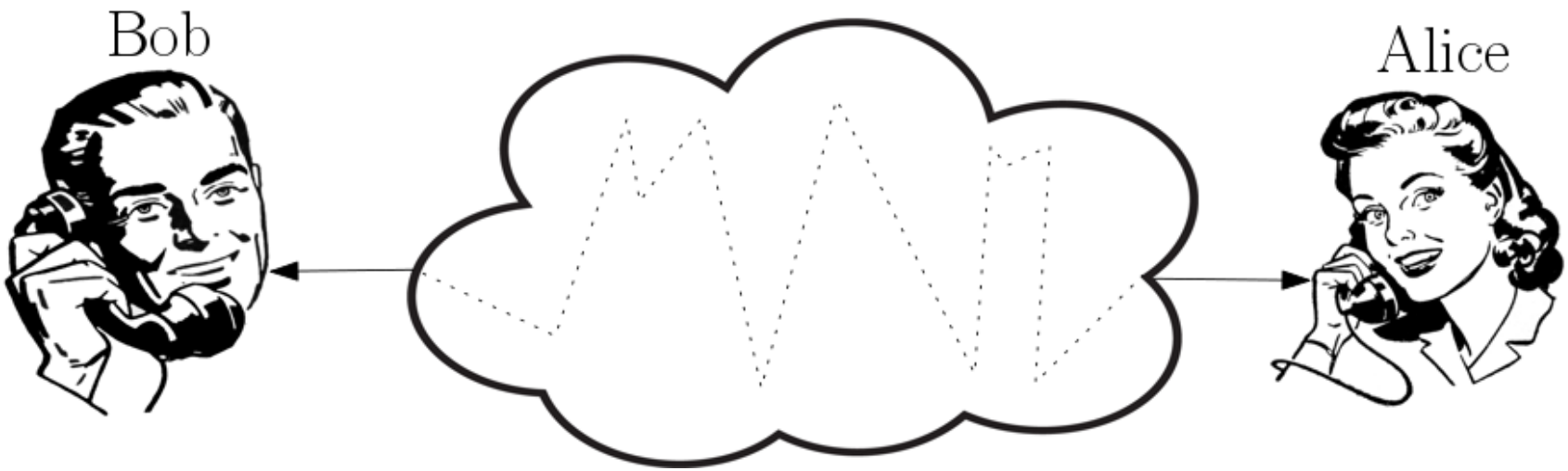
Clase 2: Resumen del Curso

Fernando Krell
fekrell@uc.cl

Pontificia Universidad Católica de Chile

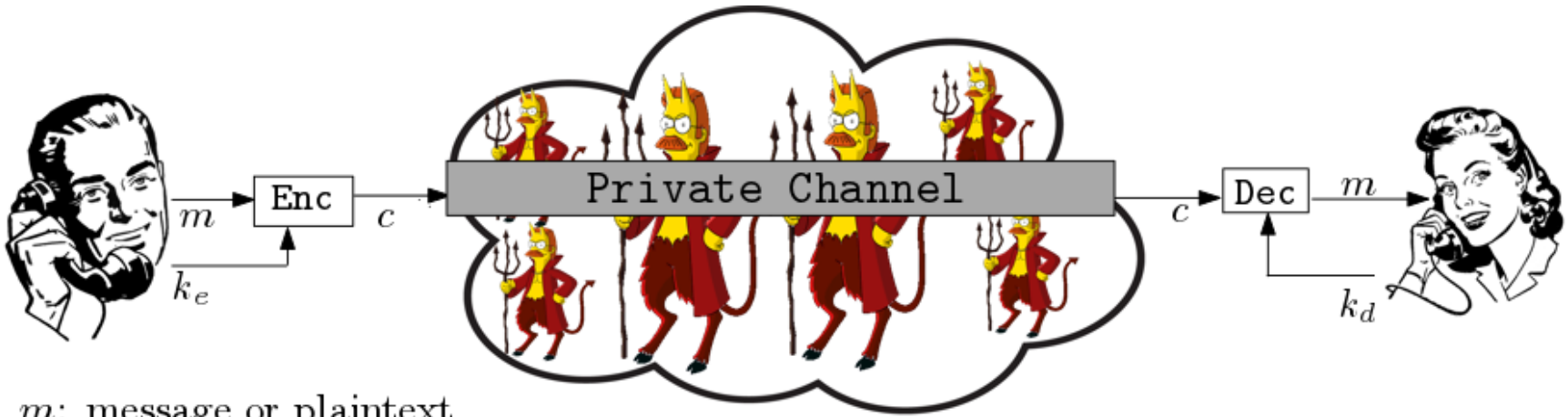
8 de Marzo 2017

Caso Típico



Caso Típico

Caso Típico



m: message or plaintext

k_e: Encryption key

Enc: Encryption Algorithm

c: ciphertext

k_d: Decryption key

Dec: Decryption Algorithm

El Adversario

- Modela comportamiento adversarial.
- Inteligente, pero no es superman.
- Astuto: No sabemos cuál será su estrategia.
- Pasivo: Sólo escucha la comunicación.
- Activo: Modifica, inyecta, bota mensajes.
  - Encriptación no provee integridad .
  - Integridad: Códigos de autenticación de mensajes(MACs) o firmas digitales.
  - No podemos hacer nada contra negación de servicio.

1. Cómo no hacerlo:

¿Qué dice aquí?

bsuejqysgenu qs bsuejqoqyu zf jni esgbjzbi gnl fjmlu qa jibnnzhmif aqs fibmsi bqddmnzbgjzqn zn jni esifinbi qa jnzsl egsjzif bgooil glpisfgszif. dqsi yinisgoou, bsuejqysgenu zf gtqmj bqnfjsmbjzny gnl gngoukzny esqjqbqof jngj esipinj jnzsl egsjzif qs jni emtozb asqd siglzny eszpgji diffgyif; pgszqmf gfeibjf zn znaqsdgjzqn fibmszju fmbn gf lgjg bqnazlinjzgozju, lgjg znjiyszju, gmjninjzbgjzqn, gnl nqn-siemlzgjzqn gsi binjsgo jq dqlisn bsuejqysgenu. dqlisn bsuejqysgenu irzfjf gj jni znjisfibjzqn qa jni lzfbzeoznif qa dgjnidgjzbf, bqdemjis fbzinbi, gnl ioibjszbgo inyzniiszny. geeozbgjzqnf qa bsuejqysgenu znbomli gjd bgslf, bqdemjis egffvqslf, gnl ioibjsqnzb bqddisbi.

Análisis de Frequencia

Ciphertext	%	English	%
i	12.06%	E	12.70%
n	11.20%	T	9.09%
j	8.87%	A	8.10%
s	8.40%	O	7.50%
q	8.07%	I	6.96%
z	7.50%	N	6.70%
g	7.30%	S	6.30%
b	5.51%	H	6.09%
f	5.08%	R	5.98%
l	4.20%	D	4.20%
e	4.13%	L	4.00%
u	2.9%	C	2.70%

Texto descifrado

Texto plano:

Cryptography or cryptology is the practice and study of techniques for secure communication in the presence of third parties called adversaries. More generally, cryptography is about constructing and analyzing protocols that prevent third parties or the public from reading private messages; various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation are central to modern cryptography. Modern cryptography exists at the intersection of the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

- ¿Qué cifrador fue usado? Uno muy simple.
- Llamado cifrador de sustitución: Alfabeto permutado.

Cifradores antiguos débiles

- Cifrador César o de desplazamiento.
- Cifrador de sustitución.
- Cifrador de Vignère.
  - Elegir palabra pequeña como llave, e.g. `HACKER`, y guardamos su valor numérico (`7,0,2,10,4,17`).
  - Repetir llave hasta largo del mensaje.
  - Applicamos desplazamiento letra por letra.

msg	t	h	i	s	c	i	p	h	e	r	i	s	n	o	t	g	o	o	d
key	7	0	2	10	4	17	7	0	2	10	4	17	7	0	2	10	4	17	7
ctext	a	h	k	c	g	z	w	h	g	b	m	j	u	o	v	q	s	f	k

Cómo lo quebramos:

1. Encontrar el largo de la llave.
2. Análisis de Frequencia.

Un mal cifrador mucho más complejo

Enigma

Seguridad de cifradores antiguos

- Todos han sido quebrados.
- Basados en heurísticas.
- Usualmente mantenidos en forma privada.
- No hay análsis formal.
- Por lo tanto, Seguridad NO garantizada!

Un enfoque formal al estudio de la seguridad:

- Damos una *definición de seguridad*.
- Proponemos un *sistema*.
- Describimos supuestos, ojalá lo más débiles posible.
- *Demostramos* que, bajo los supuestos, el sistema propuesto satisface la definición de seguridad.

Intuición:

- ¿Qué es lo que queremos? Texto cifrado no debe revelar información alguna sobre el mensaje.
- ¿Cómo deberíamos hacerlo?

Construyendo un Cifrador Simétrico

Intuición:

- ¿Qué es lo que queremos? Texto cifrado no debe revelar información alguna sobre el mensaje.
- ¿Cómo deberíamos hacerlo? Texto cifrado se vea aleatorio, no importando cuál sea el mensaje. Ej. Sumamos un número aleatorio para cifrar, y lo restamos para descifrar.
- ¿Es seguro?

Construyendo un Cifrador Simétrico

Intuición:

- ¿Qué es lo que queremos? Texto cifrado no debe revelar información alguna sobre el mensaje.
- ¿Cómo deberíamos hacerlo? Texto cifrado se vea aleatorio, no importando cuál sea el mensaje. Ej. Sumamos un número aleatorio para cifrar, y lo restamos para descifrar.
- ¿Es seguro? Como el texto cifrado es aleatorio sin saber la llave, entonces no podemos deducir nada sobre el mensaje.

Construyendo un Cifrador Simétrico

Intuición:

- ¿Qué es lo que queremos? Texto cifrado no debe revelar información alguna sobre el mensaje.
- ¿Cómo deberíamos hacerlo? Texto cifrado se vea aleatorio, no importando cuál sea el mensaje. Ej. Sumamos un número aleatorio para cifrar, y lo restamos para descifrar.
- ¿Es seguro? Como el texto cifrado es aleatorio sin saber la llave, entonces no podemos deducir nada sobre el mensaje.

Seguridad Perfecta

Texto cifrado no debe revelar información alguna sobre el mensaje.

Definition

$\langle \texttt{Gen}, \texttt{Enc}, \texttt{Dec} \rangle$ es *perfectamente secreto* si $\forall m, c$

$$\Pr[M = m | Enc(M) = c] = \Pr[M = m]$$

(o equivalentemente $\Pr[Enc(M) = c | M = m] = \Pr[Enc(M) = c]$)

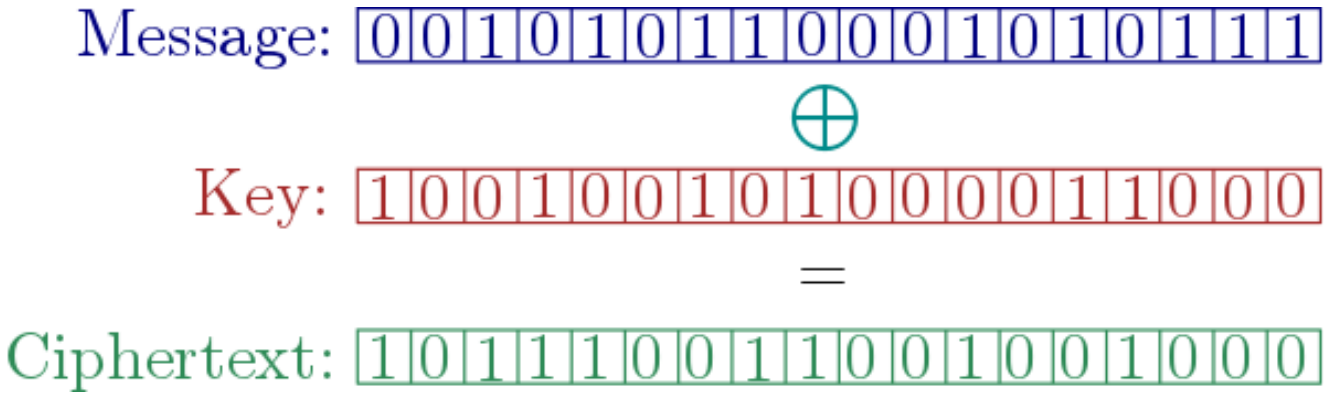
La probabilidad de que el mensaje sea un valor específico m se no ve afectada conociendo su respectivo texto cifrado

Seguridad Perfecta

Texto cifrado es aleatorio, sin importar cuál sea el mensaje original

El equema One-Time Pad (OTP):

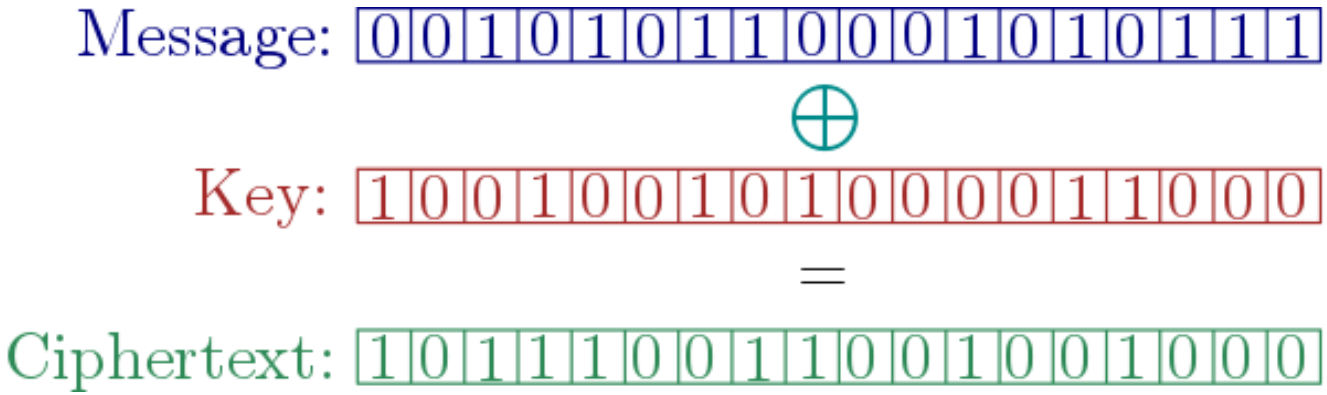
$\oplus$	0	1
0	0	1
1	1	0



Seguridad Perfecta

El equema One-Time Pad (OTP):

$\oplus$	0	1
0	0	1
1	1	0



One-Time Pad

$m \in \{0,1\}^n$

$\texttt{Gen}$: $k \xleftarrow{\$} \{0,1\}^n$, output k

$\texttt{Enc}_k(m) : c = k \oplus m$, output c

$\texttt{Dec}_k(c) : m = c \oplus k$ $(= m \oplus k \oplus k = m \oplus 0^n = m)$, output m

Seguridad del OTP

Como el texto cifrado es aleatorio sin saver la llave, entonces no podemos deducir nada sobre el mensaje.

Theorem

El OTP es un esquema de cifrados perfectamente secreto. (Sin supuestos!)

Proof.

1. La llave k es uniforme en $\{0,1\}^n$, por lo tanto $\forall x \ \Pr[K = x] = 2^{-n}$.
2. $\Pr[\mathtt{Enc}_K(M) = c | M = m] = \Pr[M \oplus K = c | M = m] = \Pr[K = c \oplus m] = 2^{-n}$.
3. $\Pr[\mathtt{Enc}_K(M) = c] = \Pr[M \oplus K = c] = \Pr[K = c \oplus M] = 2^{-n}$.

□

Seguridad del OTP

Como el texto cifrado es aleatorio sin saver la llave, entonces no podemos deducir nada sobre el mensaje.

Theorem

El OTP es un esquema de cifrados perfectamente secreto. (Sin supuestos!)

Proof.

1. La llave k es uniforme en $\{0,1\}^n$, por lo tanto $\forall x \ \Pr[K = x] = 2^{-n}$.
2. $\Pr[\mathtt{Enc}_K(M) = c | M = m] = \Pr[M \oplus K = c | M = m] = \Pr[K = c \oplus m] = 2^{-n}$.
3. $\Pr[\mathtt{Enc}_K(M) = c] = \Pr[M \oplus K = c] = \Pr[K = c \oplus M] = 2^{-n}$.

□

TODOS LOS ESQUEMAS FUNCIONAN, MÁS O MENOS, DE LA MISMA MANERA

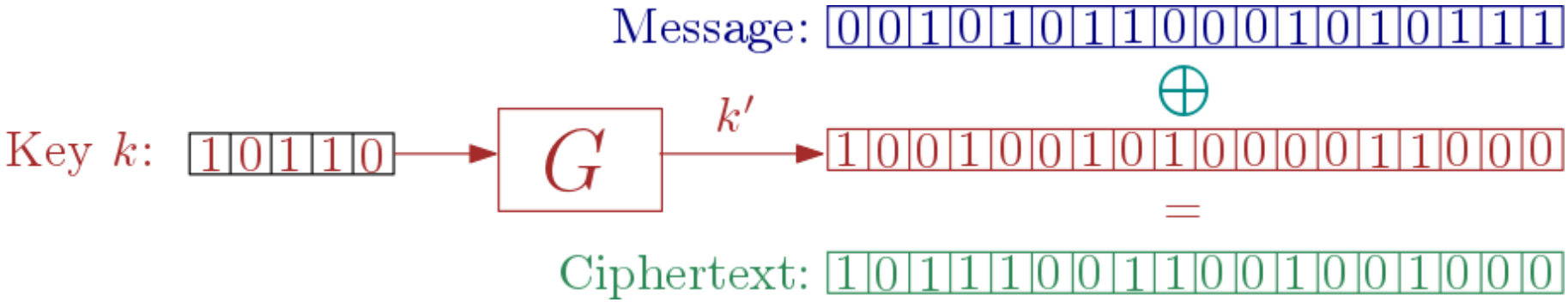
¿Problema resuelto?

TODOS LOS ESQUEMAS FUNCIONAN, MÁS O MENOS, DE LA MISMA MANERA

¿Problema resuelto? NO!

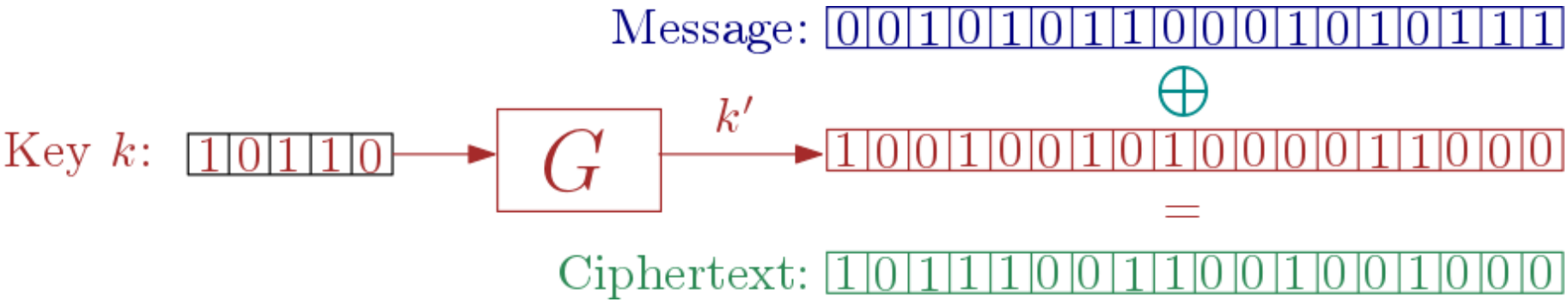
1. Llave es tan larga como el mensaje.
2. OTP puede ser utilizado una sola vez.
3. ¿Cómo fue la llave compartida originalmente?

Problema 1: Alargando la llave



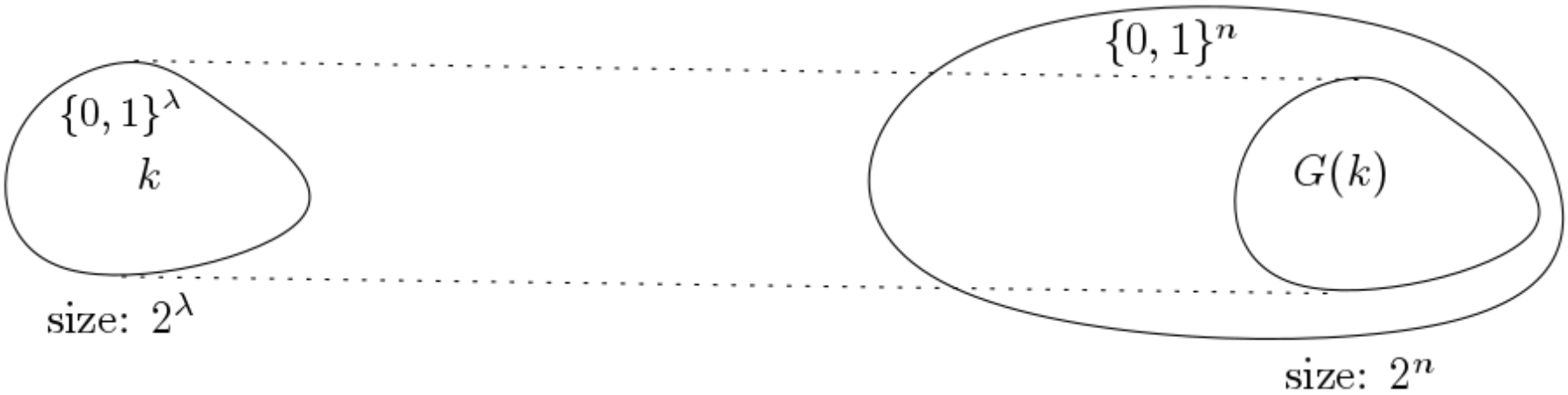
$G : \{0,1\}^{\lambda} \to \{0,1\}^{n}$ es un generador Pseudo-Aleatorio

Problema 1: Alargando la llave

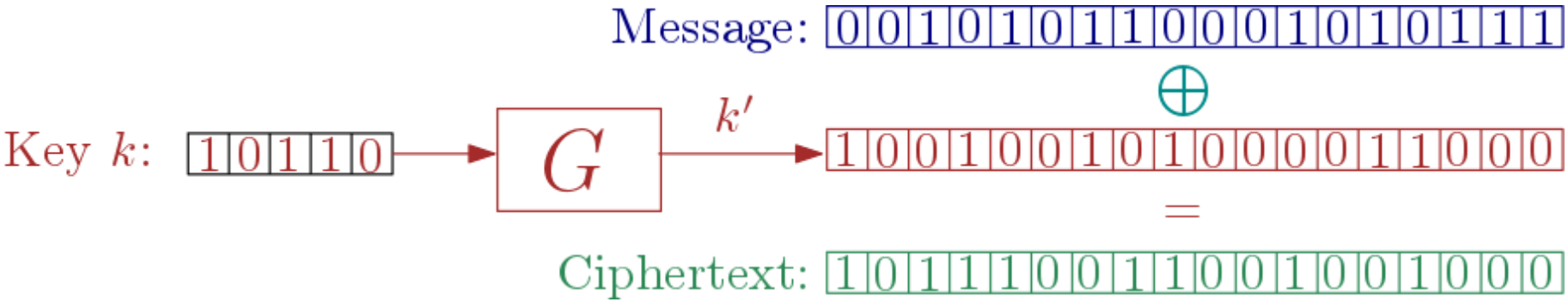


$G : \{0,1\}^\lambda \to \{0,1\}^n$ es un generador Pseudo-Aleatorio

Problema: $k' = G(k)$ no puede ser uniformemente aleatorio. Seguridad perfecta es imposible de obtener.

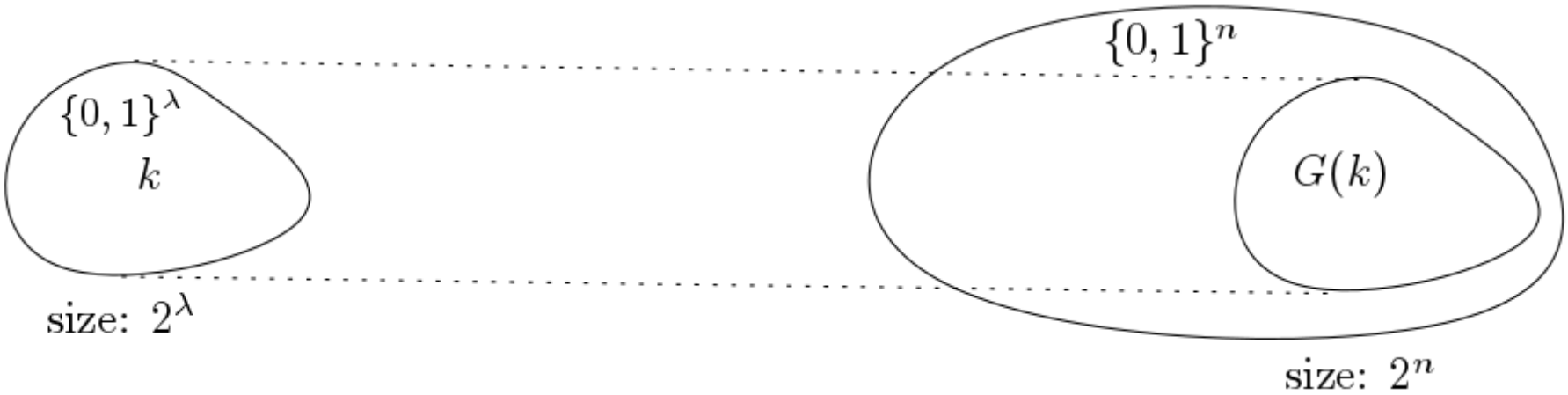


Problema 1: Alargando la llave



$G : \{0,1\}^\lambda \to \{0,1\}^n$ es un generador Pseudo-Aleatorio

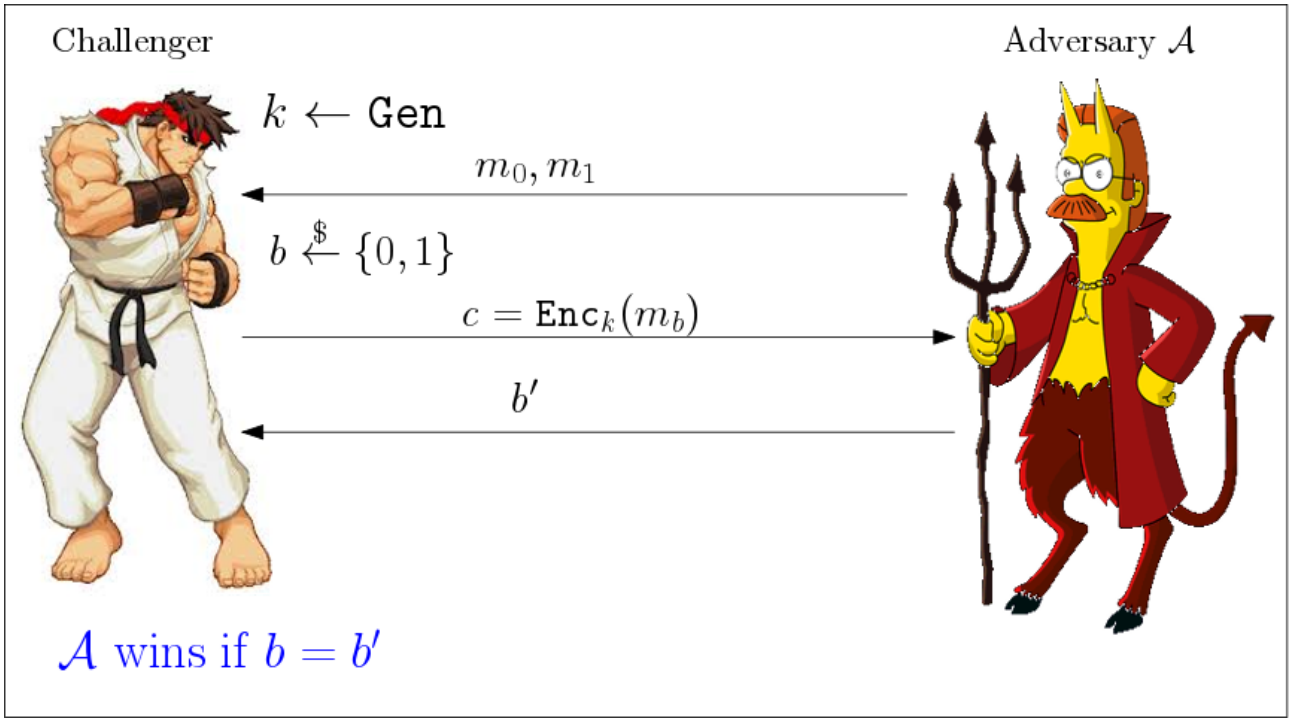
Problema: $k' = G(k)$ no puede ser uniformemente aleatorio. Seguridad perfecta es imposible de obtener.



CAMBIAMOS LA DEFINICIÓN

Nuevo paradigma: Juegos de indistinguibilidad

Experimento de Indistinguibilidad:

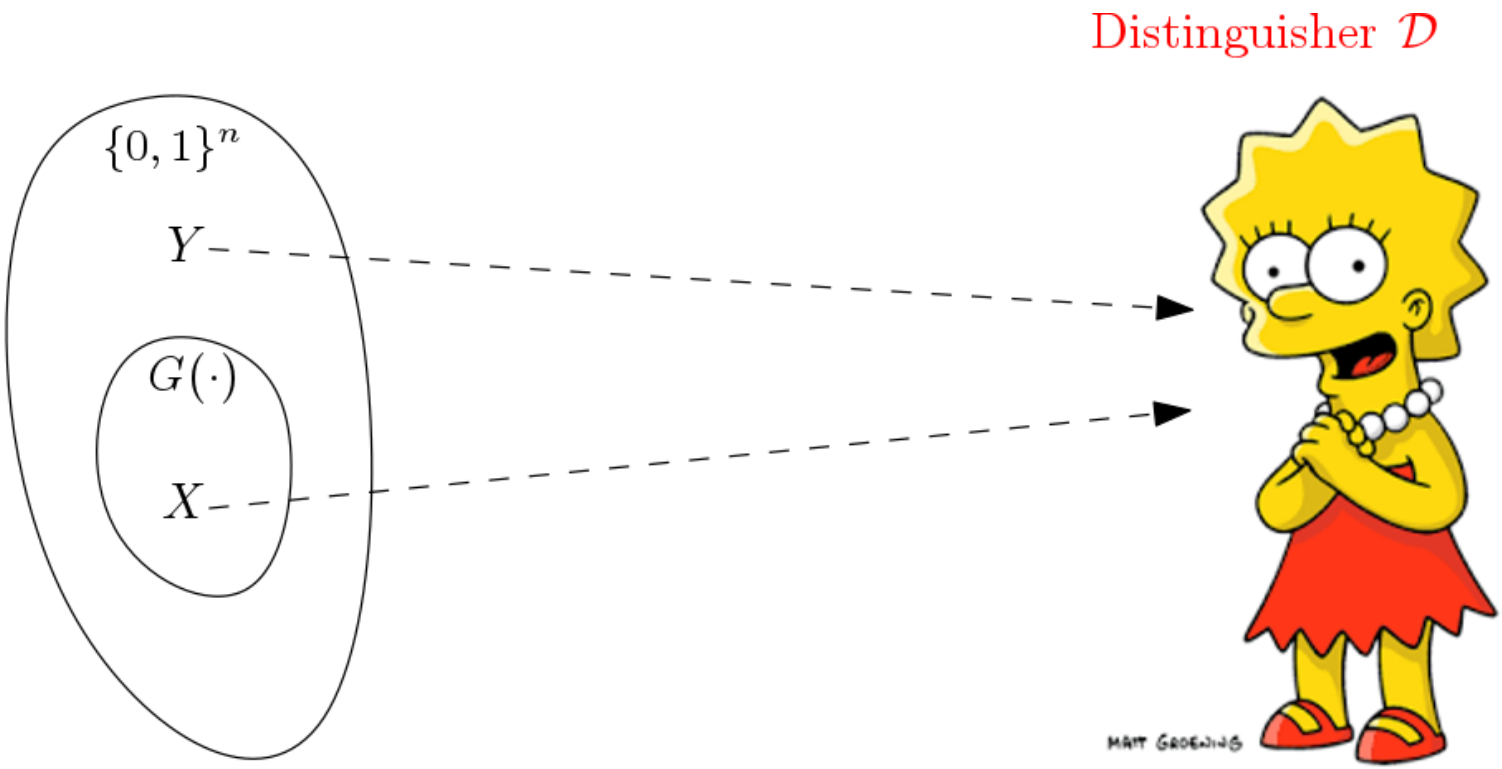


Definition

$\langle \mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec} \rangle$ tiene *cifrados indistinguisbles si* $\forall \mathcal{A}$

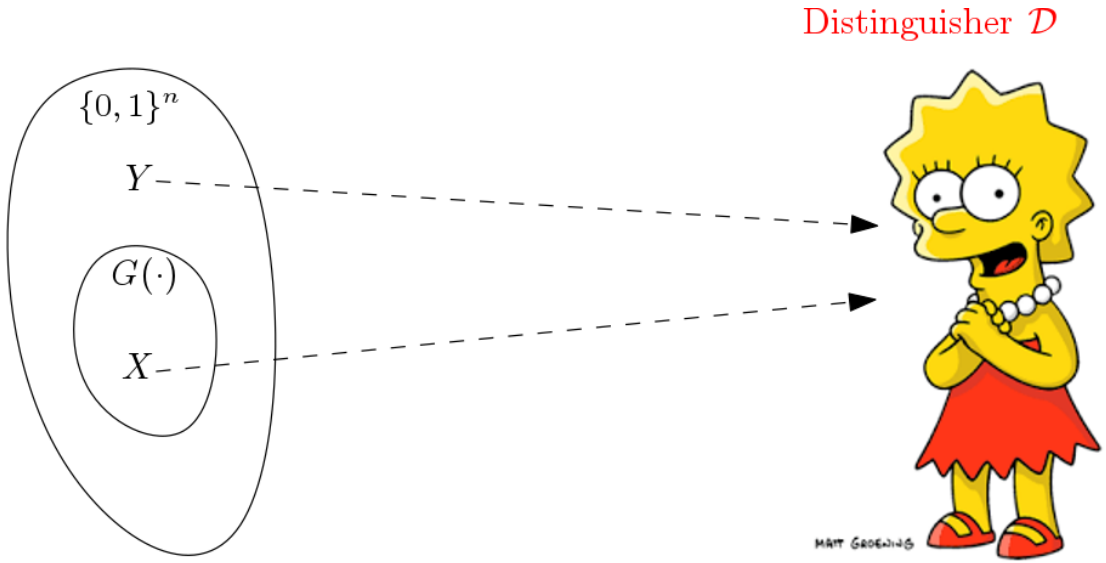
$$\Pr[\mathcal{A} \text{ wins}] <= 1/2 + \text{ pequeño}$$

Elementos Pseudo-Aleatorios



$\mathcal{D}$ no puede diferenciar entre Y aleatorio y X aleatorio

Elementos Pseudo-Aleatorios



$\mathcal{D}$ no puede diferenciar entre Y aleatorio y X aleatorio

Definition (PRG)

Una function $G : \{0,1\}^\lambda \to \{0,1\}^n$ es un *generador pseudo-aleatorio* si

1. $\lambda < n$ (stretching)
2. Ningún $\mathcal{D}$ puede distinguir entre $X = G(K)$, para K uniforme en $\{0,1\}^\lambda$, e Y uniforme en $\{0,1\}^n$

Esquema

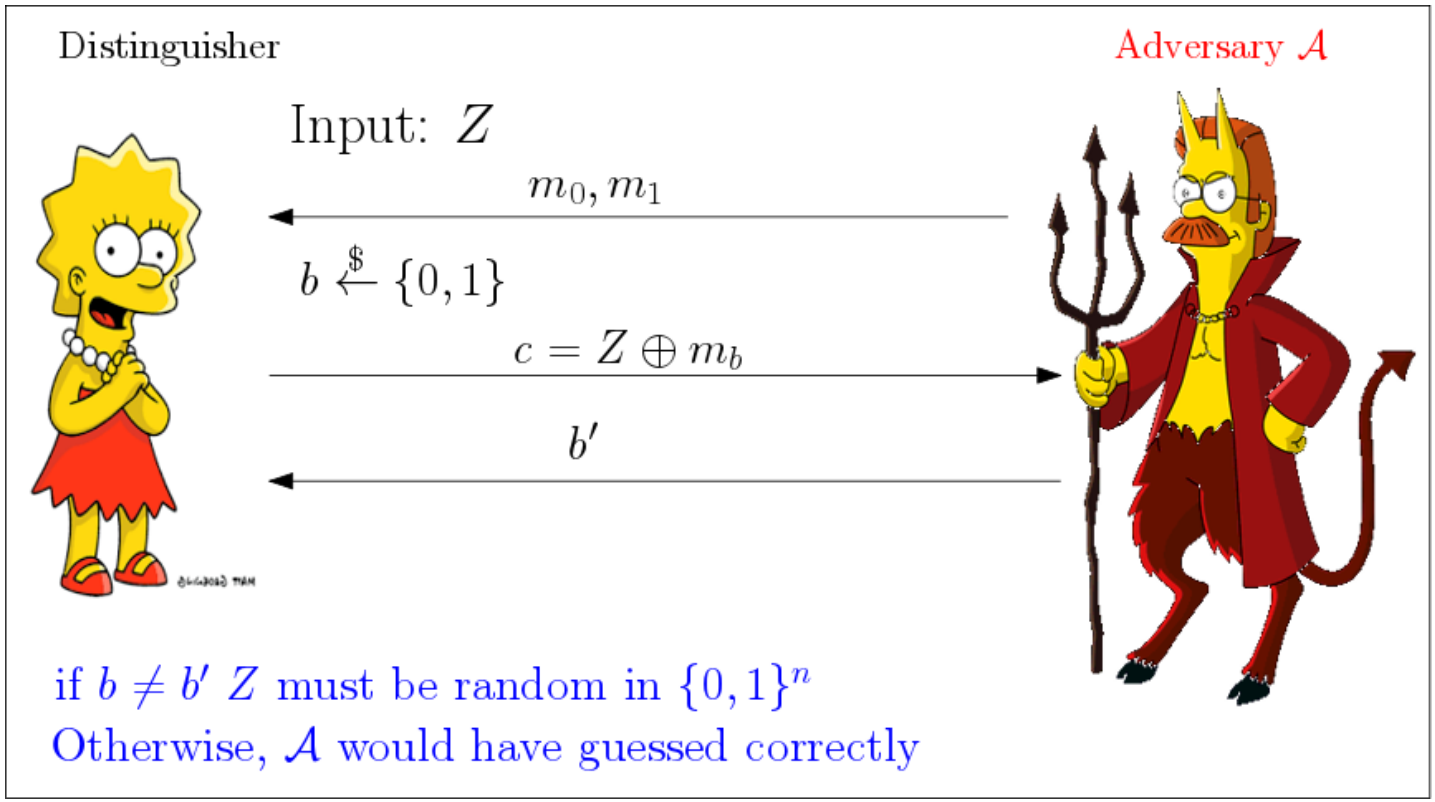
- $\texttt{Gen}(1^\lambda)$: Elegimos una llave uniformemente aleatorio en $\{0,1\}^\lambda$.
- $\texttt{Enc}_k(m)$: Computamos $k' = G(k)$, output $k' \oplus m$.
- $\texttt{Dec}_k(c)$: Computamos $k' = G(k)$, output $k' \oplus c$.

Theorem

Si G es un generador pseudo-aleatorio, entonces el esquema anterior tiene cifrados indistinguibles. (Ahora tenemos un supuesto!)

Demostración de Seguridad

Asumimos que el esquema no es seguro, y probamos que G no puede ser un generador pseudo-aleatorio.
Construimos un distinguidor $\mathcal{D}$ que use un adversario $\mathcal{A}$ que quiebra el esquema de cifrado.

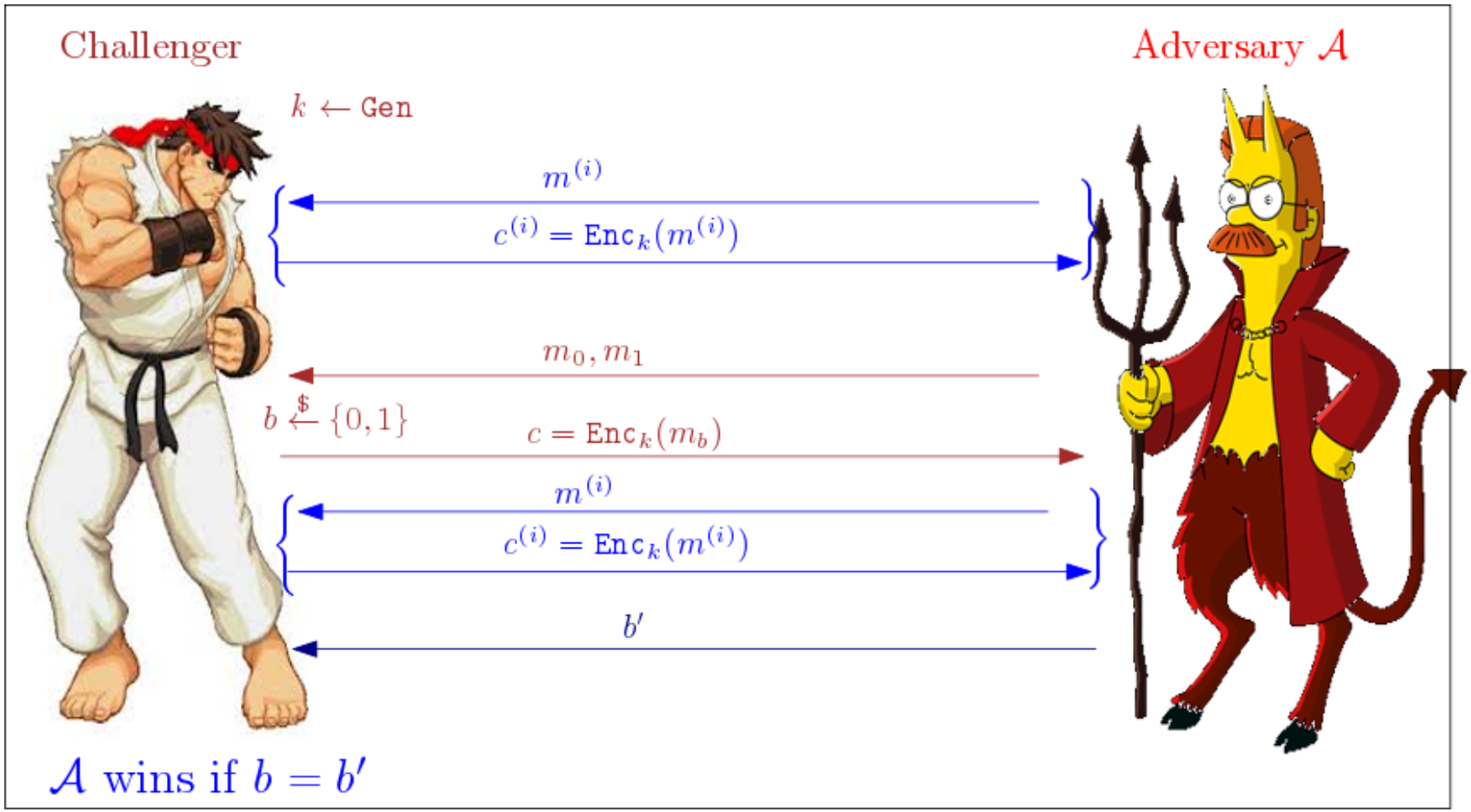


Problema 2: ¿Cómo reusamos la llave? Seguridad CPA

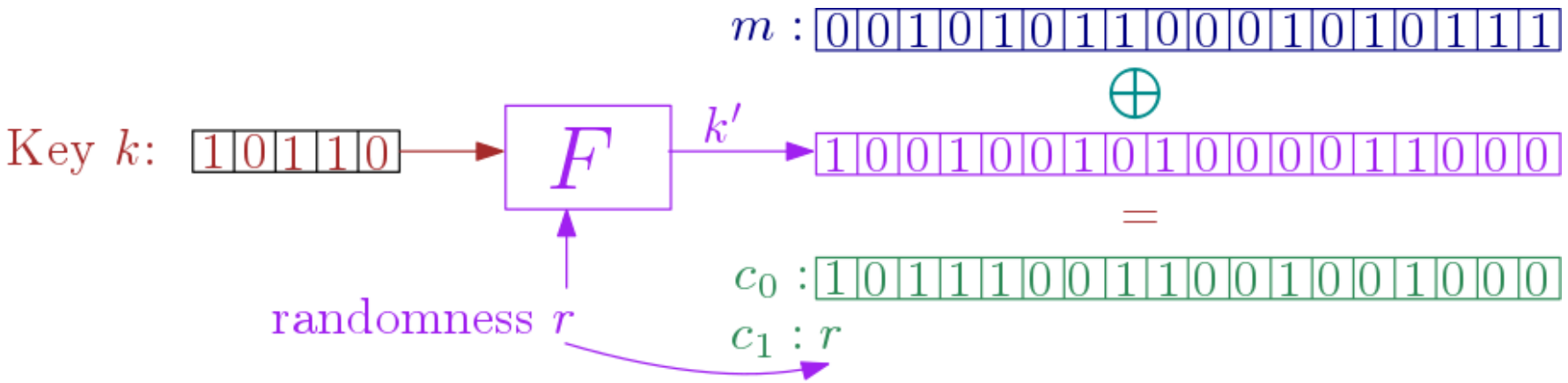
Resolvimos el problema del largo de la llave, pero aún no podemos reutilizarla

Nueva definición de seguridad

Experimento Ataque de textos planos (CPA):

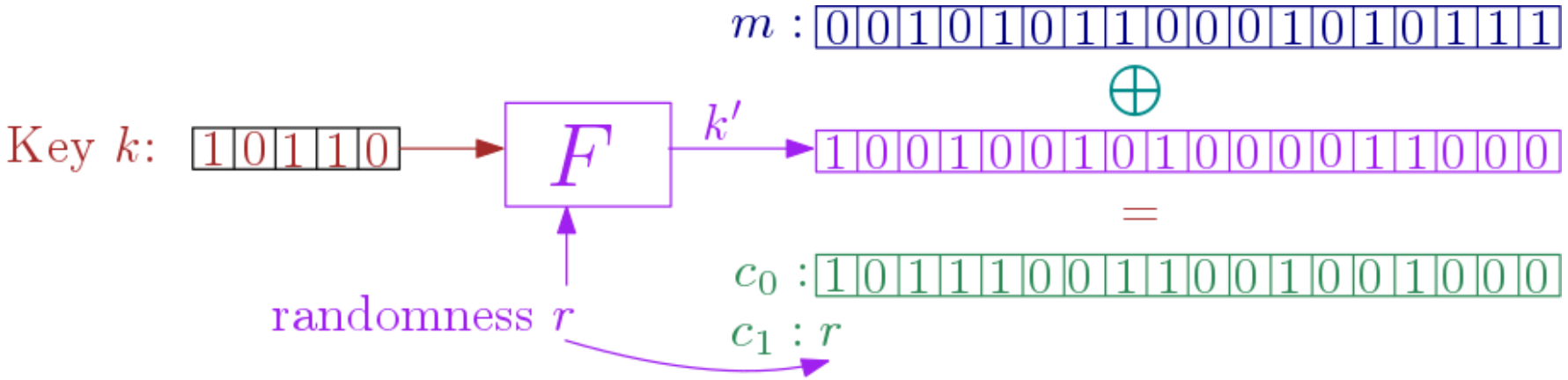


Nueva Herramienta: Funciones Pseudo-aleatorias



$F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^n$ se llama Función Pseudo-Aleatoria

Nueva Herramienta: Funciones Pseudo-aleatorias



$F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^n$ se llama Función Pseudo-Aleatoria

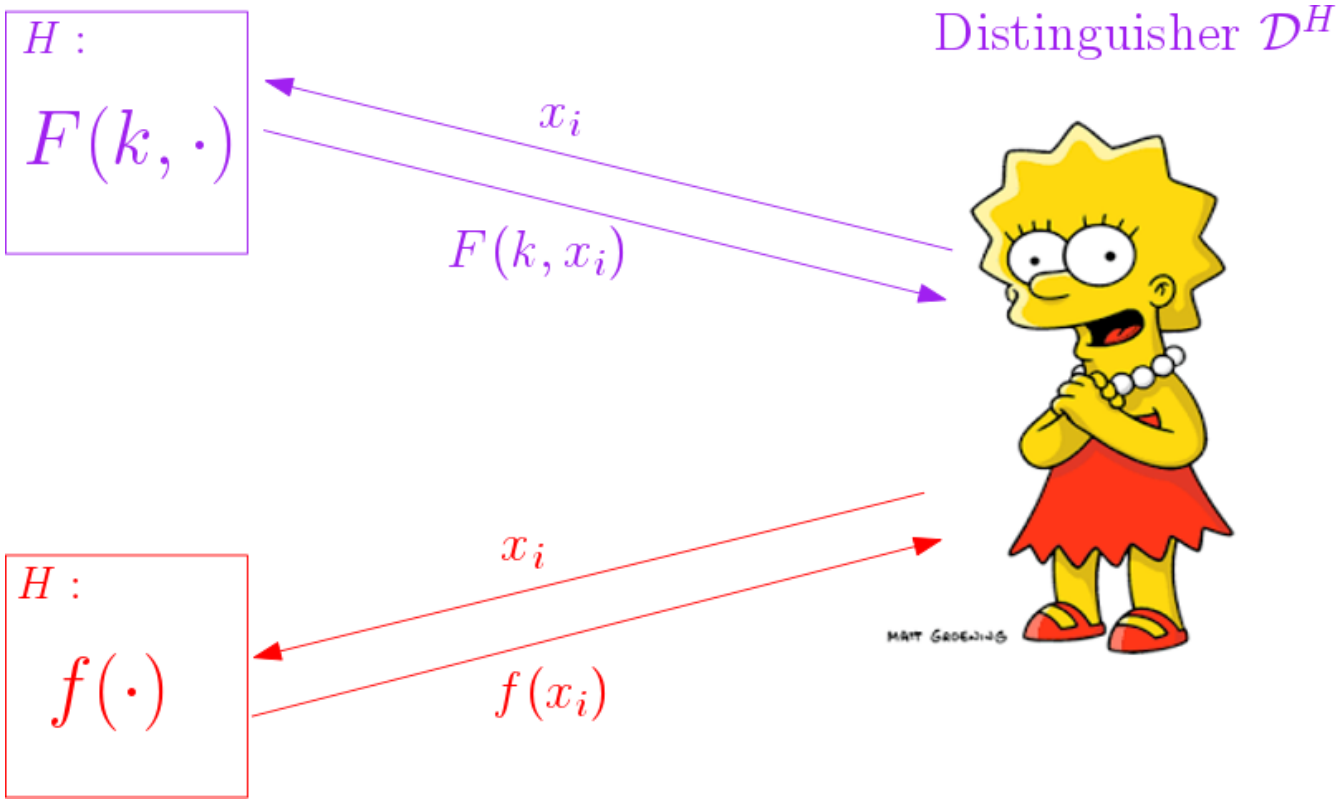
Definition (PRF)

Una función $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^n$ es pseudo-aleatoria si ningún algoritmo puede distinguir $F(k,\cdot) : \{0,1\}^\lambda \to \{0,1\}^n$, para k uniforme, de una función de verdad aleatoria $f : \{0,1\}^\lambda \to \{0,1\}^n$.

Nueva Herramienta: Funciones Pseudo-aleatorias

Definition (PRF)

Una función $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^n$ es pseudo-aleatoria si ningún algoritmo puede distinguir $F(k,\cdot) : \{0,1\}^\lambda \to \{0,1\}^n$, para k uniforme, de una función de verdad aleatoria $f : \{0,1\}^\lambda \to \{0,1\}^n$.



$\mathcal{D}$ no puede distinguir entre $H(\cdot) = F(k,\cdot)$ de $H(\cdot) = f(\cdot)$

Nuevo Esquema

Idea: Cada vez que ciframos, elegimos una llave k' nueva .

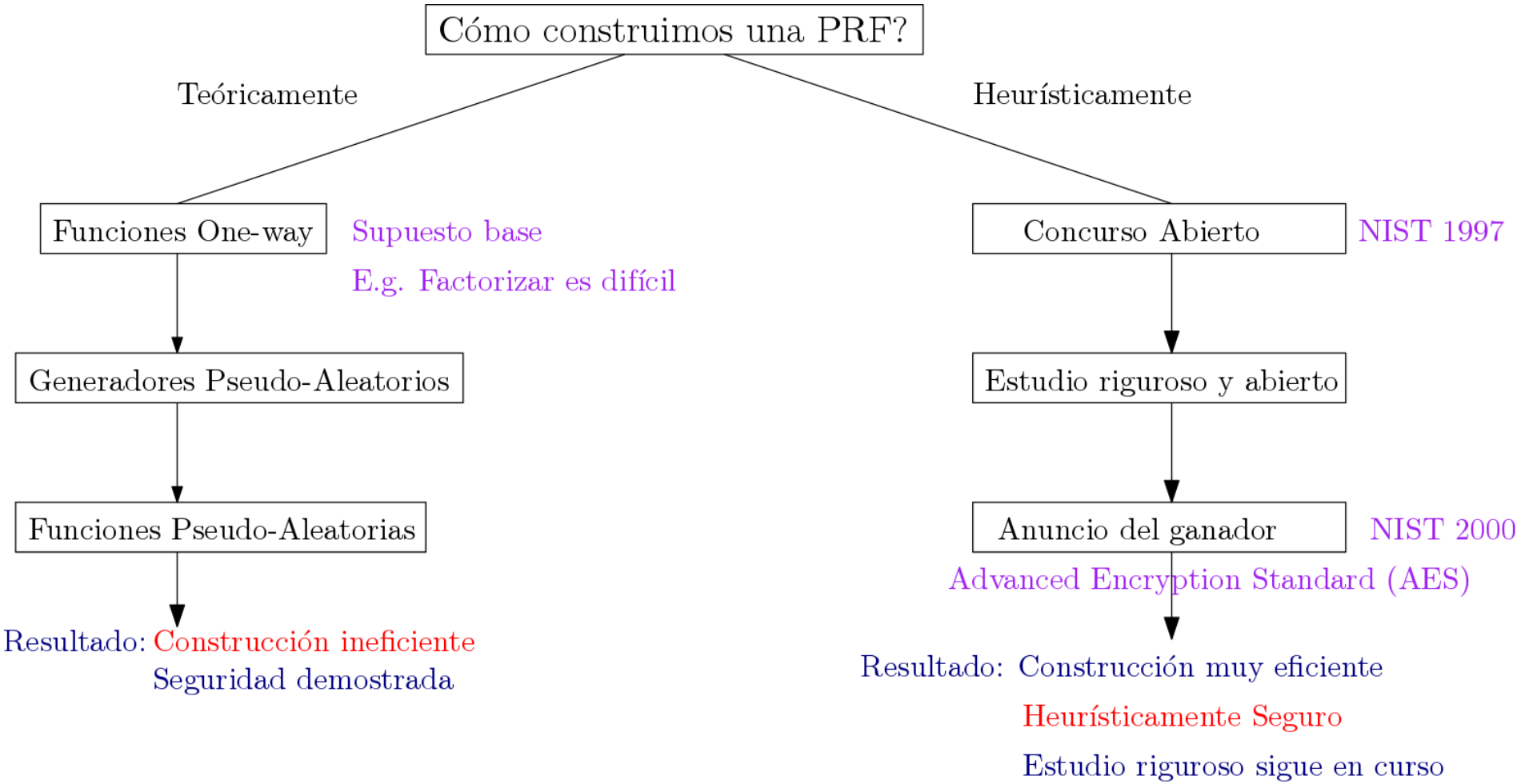
- $\texttt{Gen}(1^\lambda)$: Elegir una llave k uniforme en $\{0,1\}^\lambda$, output k.
- $\texttt{Enc}_k(m)$: Elegir r uniforme, computamos $k' = F_k(r)$, output $\langle k' \oplus m, r \rangle$.
- $\texttt{Dec}_k(c_1, c_2)$: Computamos $k' = F(k, c_2)$, output $m = k' \oplus c$.

Theorem

Si F es una función pseudo-aleatoria, entonces el esquema es seguro frente a ataques de texto plano escogido (CPA).

No lo demostraremos acá

Sub-problema 2.5: ¿Cómo construimos una PRF?



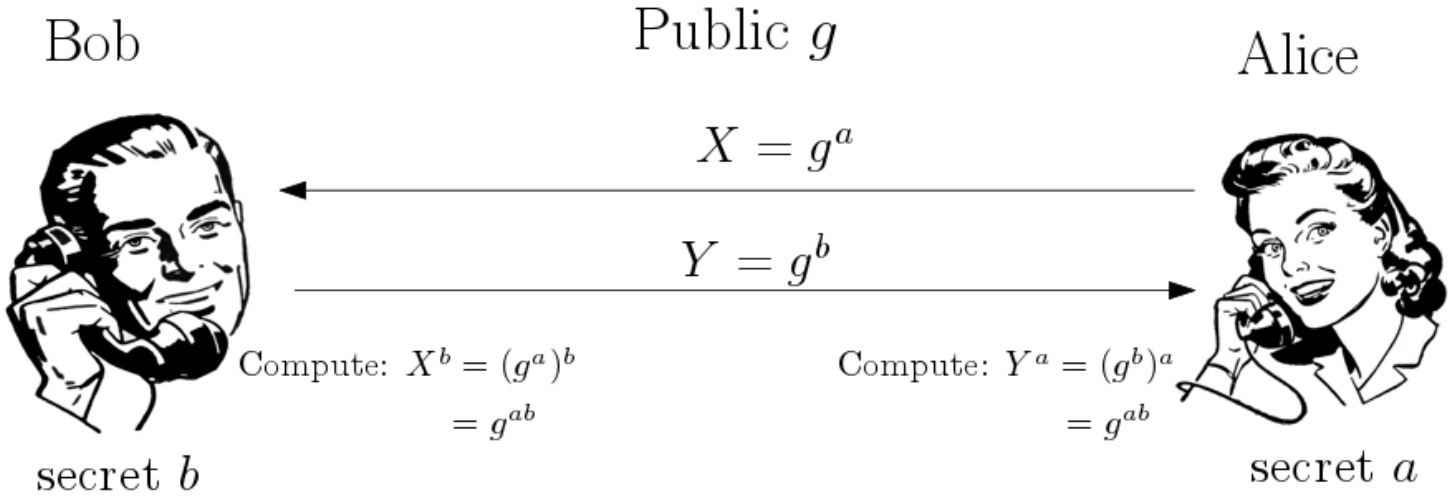
Problema 3: ¿Cómo Alice y Bob obtuvieron la llave?

1. Alice y Bob se reunieron en privado.

2. "New Directions in Cryptography" DH'76
   - Revolucionaron la criptografía (y el mundo).
   - Seguridad de Internet depende de este protocolo.
   - Premio Turing 2015!

Diffie-Hellman

Intercambio de llaves Diffie-Hellman



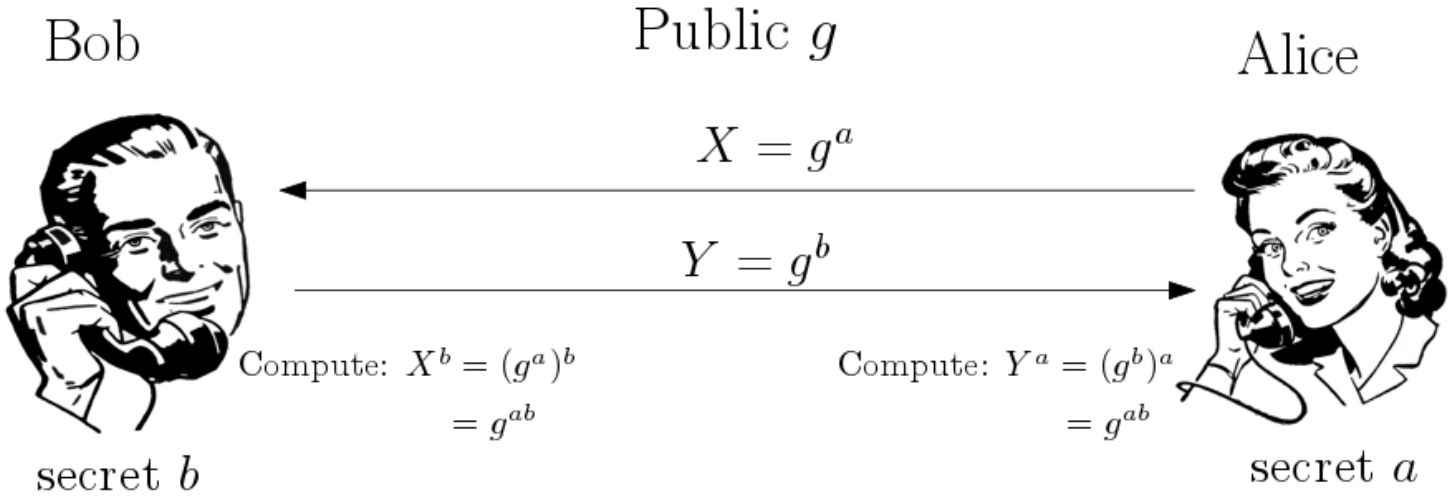
Use $K = g^{ab}$ for secure channel

Supuestos

- Necesario: Difícil de obtener x a partir de g, g^x (DLog).
- Suficiente: g^{ab} es pseudo-aleatorio aún sabiendo g, g^a, g^b. (Decisional DH)
- Si DLog es fácil, entonces DDH también lo es. .
- Si DDH es fácil, DLog aún puede ser difícil. Por lo tanto, DDH es un supuesto más *fuerte* que DLog .

Diffie-Hellman

Intercambio de llaves Diffie-Hellman



Supuestos

- Necesario: Difícil de obtener x a partir de g, g^x (DLog).
- Suficiente: g^{ab} es pseudo-aleatorio aún sabiendo g, g^a, g^b. (Decisional DH)
- Si DLog es fácil, entonces DDH también lo es. .
- Si DDH es fácil, DLog aún puede ser difícil. Por lo tanto, DDH es un supuesto más *fuerte* que DLog .

Cifrador de llave pública El Gamal

- Público: g
- $\texttt{Gen}$: Elegimos a aleatorio, computamos g^a Output $k_e = g^a$, $k_d = a$.
- $\texttt{Enc}_{k_e}(m)$: Elegimos r aleatorio, computamos g^r . Output $c = \langle m \cdot k_e^r,\ g^r \rangle$.
- $\texttt{Dec}_{k_d}(c_0, c_1)$: Computamos $R = c_1^{k_d}$. Output $m = c_0 \cdot R^{-1}$.
- ¿Por qué funciona?

$$\begin{aligned} c_0 \cdot R^{-1} &= c_0 \cdot (c_1^{k_d})^{-1} \\ &= c_0 \cdot (g^r)^{-k_d} \\ &= c_0 \cdot g^{-ra} \\ &= m \cdot k_e^r \cdot g^{-ra} \\ &= m \cdot g^{ar} \cdot g^{-ar} = m \end{aligned}$$

- Seguridad basada en DDH.

Cifrador de llave pública: RSA plano

- Gen: Elegimos P, Q, e aleatorios, computamos $N = P \cdot Q, d = e^{-1} \mod (P-1)\cdot(Q-1)(= \phi(N))$.
  Output $k_e = e, N$, $k_d = d, N$.
- $\mathrm{Enc}_{k_e}(m)$: Ouptut $c = m^e \mod N$.
- $\mathrm{Dec}_{k_d}(c)$: Output $m = c^d \mod N$.
- ¿Por qué funciona?

$$\begin{aligned} c^d &= (m^e)^d \mod N \\ &= m^{e\cdot d} \mod \phi(N) \mod N \\ &= m^{e\cdot e^{-1}} \mod \phi(N) \mod N \\ &= m \end{aligned}$$

- Supuesto Necesario: Dado N, difícil de obtener P, Q (Factor).
- Supuesto Suficiente: Dado e, N, difícil de obtener

$$d = e^{-1} \mod \phi(N)$$

No vimos en esta presentación

- Funciones de Hash $H : \{0,1\}^* \to \{0,1\}^\lambda$.
- Integridad de mensajes $\langle \mathtt{Gen}, \mathtt{MAC}, \mathtt{VRFY} \rangle$.

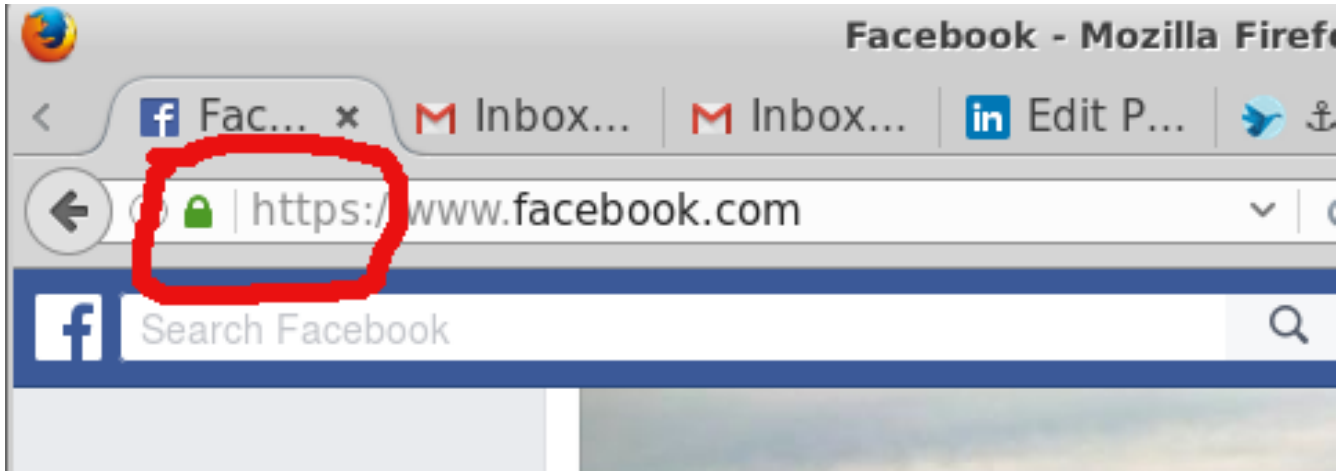
$$\mathtt{VRFY}_k(m,t) = \mathtt{True} \text{ if and only if } t = \mathtt{MAC}_k(m)$$

- Firmas Digitales $\langle \mathtt{Gen}, \mathtt{SIGN}, \mathtt{VRFY} \rangle$

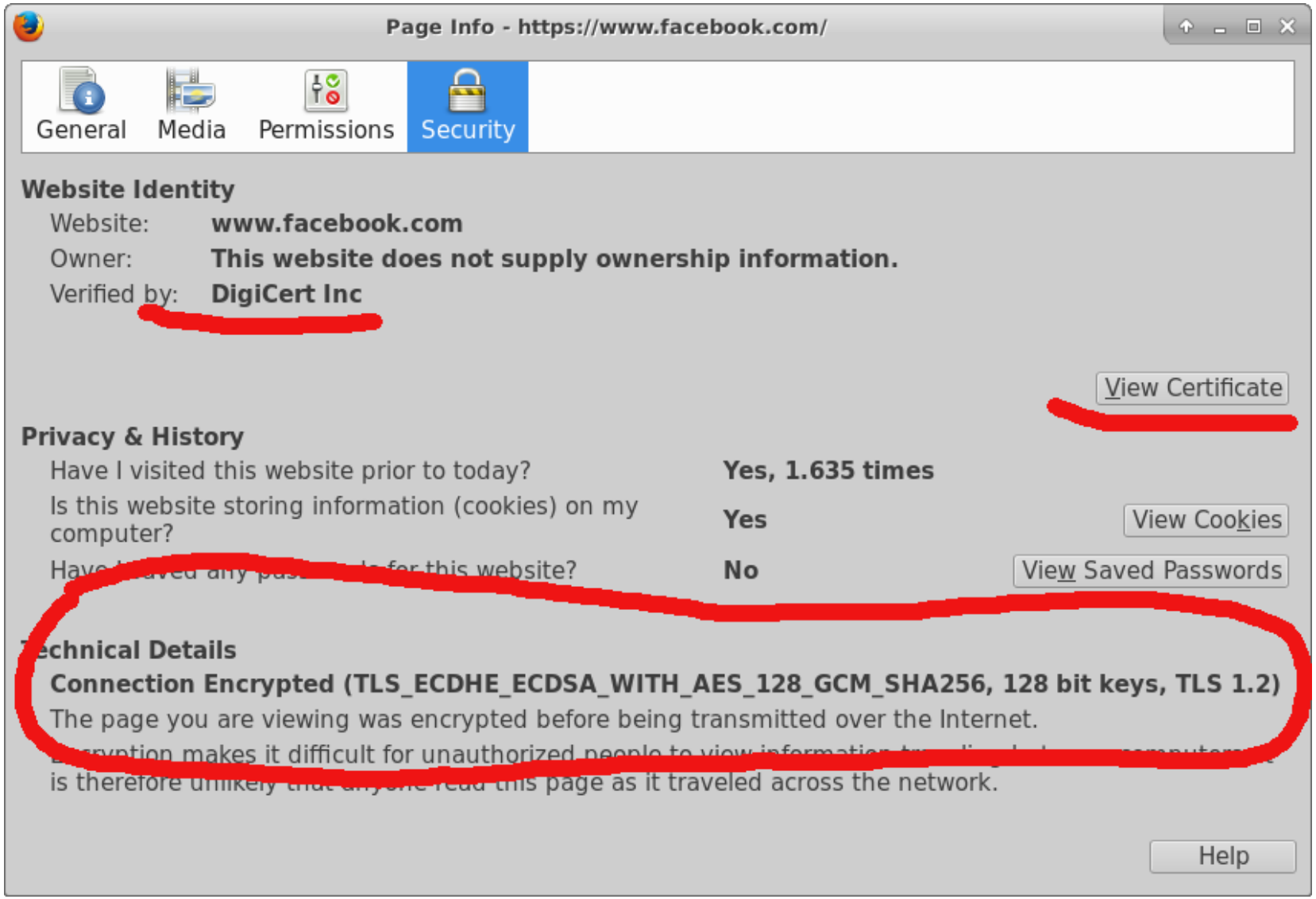
$$\mathtt{VRFY}_{pk}(m,s) = \mathtt{True} \text{ if and only if } s = \mathtt{SIGN}_{sk}(m)$$

- Certificados Digitales. Ej. Cómo puede tu navegador confiar en la llave pública del servidor web.

Applicación: TLS en tu navegador



Applicación: TLS en tu navegador



Applicación: TLS en tu navegador

- `TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`
- TLS: Transport Layer Security, nombre del protocolo.
- ECDHE: Intercambio de llaves Diffie-Hellman en Curvas Elípticas.
- ECDSA: DSA on Elliptic Curves, esquema de firmas digitales.
- AES-128: PRF (cifrador de bloque) usaba para cifrar datos.
- GCM: Modo de operación de la PRF y esquema de integridad.
- SHA256: Función de hash a utilizar.

Criptografía es mucho más

- Criptografía basada en la identidad.
  - Usar identidad como llave pública.
  - $pk = \texttt{fernando.krell@dreamlab.net}$
- Cifradores basados en atributos.
  - Puedes descifrar si tienes el atributo.
  - $\texttt{Enc}_{\texttt{ayudantes}}(\texttt{notas})$
- Cifrado funcional.
  - Puedes descifrar una funcíon sobre el mensaje.
  - $c = \texttt{Enc}_{\texttt{promedio}}(\texttt{notas})$
  - $\texttt{Dec}_{sk_{\texttt{promedio}}}(\texttt{notas}) = \sum_i^n \texttt{notas}_i/n$
- Cifradores Homomórficos: Computar sobre datos que estan cifrados.
  - $Enc(m_1) \oplus Enc(m_2) = Enc(m_1 + m_2)$
  - $Enc(m_1) \otimes Enc(m_2) = Enc(m_1 \times m_2)$.

Criptografía es mucho más (2)

- Demostraciones de Nula Divulgación
  - Puedo demostrar veracidad de una declaración sin revelar secreto alguno.
  - Ej. Yo sé la llave secreta correspondiente a esa llave pública.
  - Aplicación inmediata: Esquemas de Identificación.
- Computación Segura
  - Alice, Bob y Charlie quieren computar una función $f(\cdot, \cdot, \cdot)$ sin revelar sus inputs.
  - Muchas aplicaciones: Búsqueda privada en Bases de datos, votación electrónica, negociación de contratos, subastas electrónicas, etc.
- Criptografía Cuántica.
  - Uso de propiedades de mecánica cuántica para construir esquemas criptográficos.
- Criptografía post-cuántica:
  - Bajo computadores cuánticos, DLog y Factor son fáciles. Necesitaremos nuevos esquemas.

Conclusión

- Análisis formal de seguridad es la manera correcta de estudiarla.
- Cripto es difícil.
- Cripto es interesante!
- Muchas aplicaciones a problemas de la vida real.
- Futuro prometedor.
- Mucho trabajo por hacer.

Conclusión

- Análisis formal de seguridad es la manera correcta de estudiarla.
- Cripto es difícil.
- Cripto es interesante!
- Muchas aplicaciones a problemas de la vida real.
- Futuro prometedor.
- Mucho trabajo por hacer.

¿Preguntas?