Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

PART XIII

Outline

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

In a Private Communication Network
n users $\Rightarrow$ $\binom{n}{2}$ secret keys

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

In a Private Communication Network
n users $\Rightarrow$ $\binom{n}{2}$ secret keys
Every user stores $n-1$ independent secret keys!

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

In a Private Communication Network
n users $\Rightarrow$ $\binom{n}{2}$ secret keys
Every user stores $n-1$ independent secret keys!

Public Key Cryptography: Every user generates a key pair (pk, sk), publishes pk and keeps sk secret

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

In a Private Communication Network
n users $\Rightarrow$ $\binom{n}{2}$ secret keys
Every user stores $n-1$ independent secret keys!

Public Key Cryptography: Every user generates a key pair (pk, sk), publishes pk and keeps sk secret

Every user stores only one secret key

Cryptography in a Multiuser Setting

Private Storage: One secret key per user

Private Communication: One shared secret key per channel

In a Private Communication Network
n users $\Rightarrow$ $\binom{n}{2}$ secret keys
Every user stores $n-1$ independent secret keys!

Public Key Cryptography: Every user generates a key pair (pk, sk), publishes pk and keeps sk secret

Every user stores only one secret key . . . but public keys must be reliably distributed

The Diffie-Hellman Key Agreement Protocol

The seminal work in public key cryptography!

Let $G = \langle g \rangle$ be a cyclic (multiplicative) group of prime order q.

The Diffie-Hellman Key Agreement Protocol

The seminal work in public key cryptography!

Let $G = \langle g \rangle$ be a cyclic (multiplicative) group of prime order q.

Party $P_i(G, q, g)$:
  $x_i \leftarrow \mathbb{Z}_q^{\times}$;
  publish $y_i = g^{x_i}$;

Every two parties P_i, P_j can compute a common secret key
$k_{ij} = y_i^{x_j} = y_j^{x_i} = g^{x_i x_j}$

The Diffie-Hellman Key Agreement Protocol

The seminal work in public key cryptography!

Let $G = \langle g \rangle$ be a cyclic (multiplicative) group of prime order q.

Party $P_i(G, q, g)$:
  $x_i \leftarrow \mathbb{Z}_q^{\times}$;
  publish $y_i = g^{x_i}$;

Every two parties P_i, P_j can compute a common secret key $k_{ij} = y_i^{x_j} = y_j^{x_i} = g^{x_i x_j}$

Given only $(g, y_i = g^{x_i}, y_j = g^{x_j})$, $k_{ij} = g^{x_i x_j}$ is indistinguishable from random (Decision Diffie-Hellman Assumption)

The Diffie-Hellman Key Agreement Protocol

The seminal work in public key cryptography!

Let $G = \langle g \rangle$ be a cyclic (multiplicative) group of prime order q.

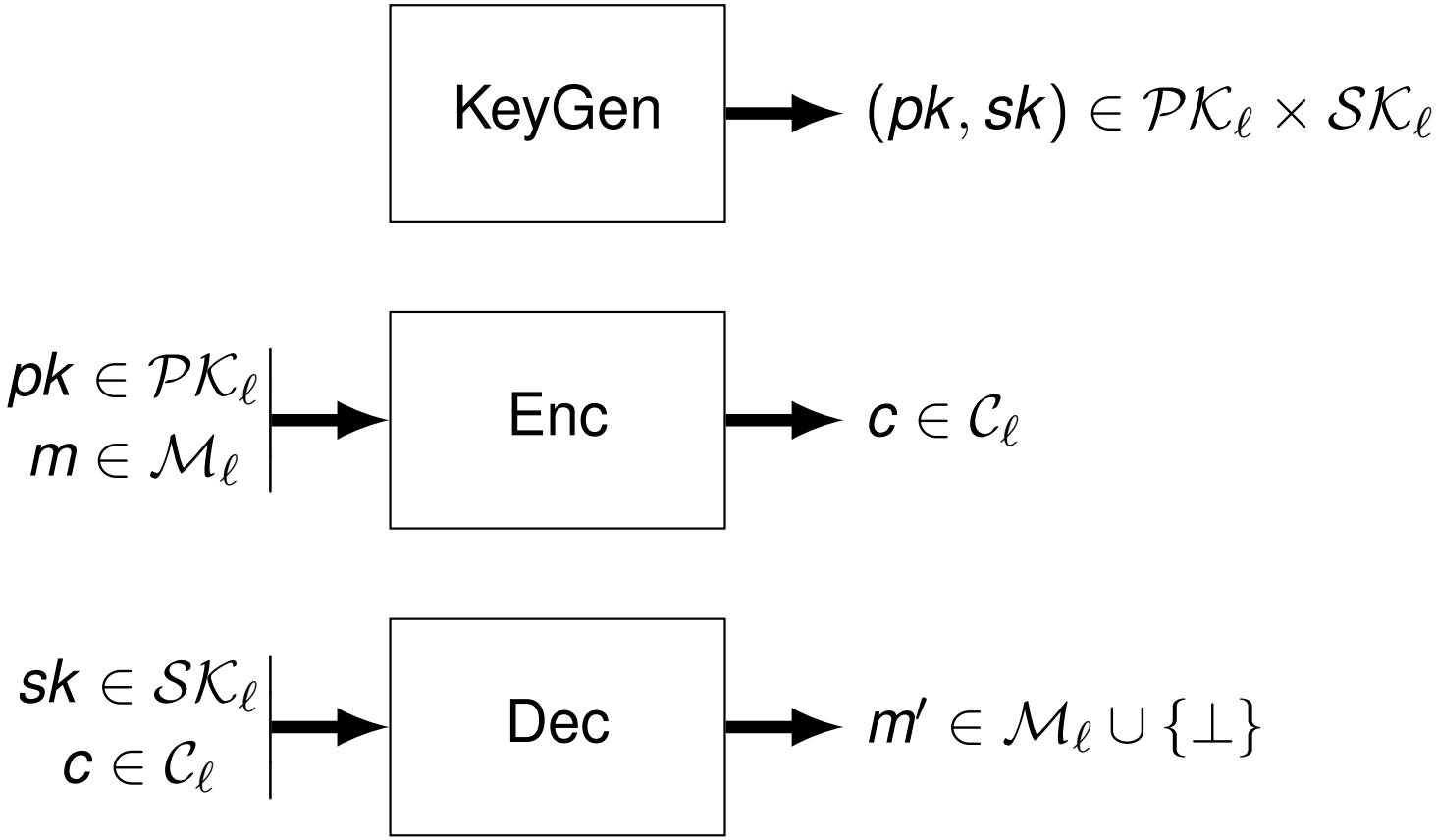
Party $P_i(G, q, g)$:
$x_i \leftarrow \mathbb{Z}_q^{\times}$;
publish $y_i = g^{x_i}$;

Every two parties P_i, P_j can compute a common secret key $k_{ij} = y_i^{x_j} = y_j^{x_i} = g^{x_i x_j}$

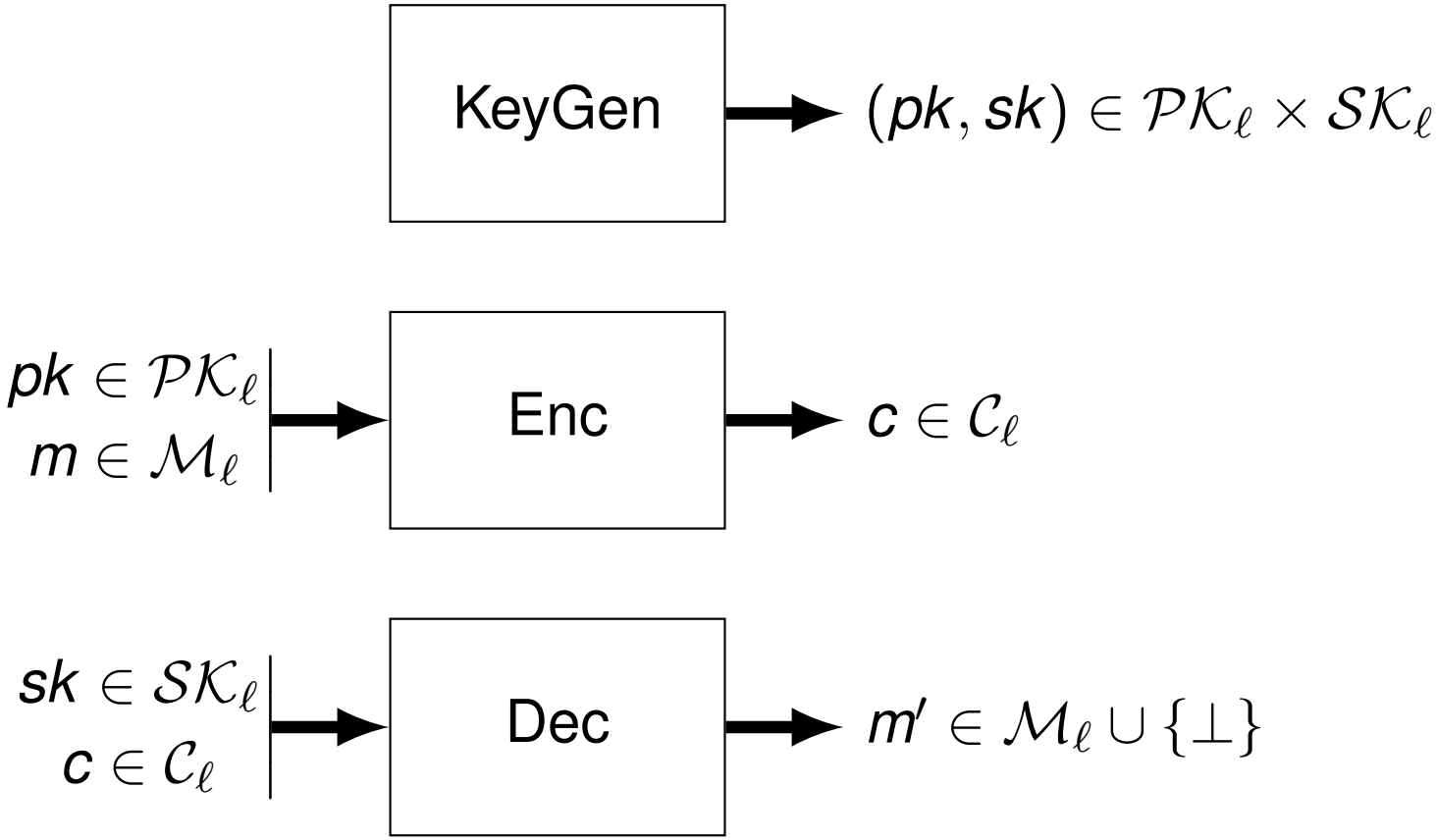
Given only $(g, y_i = g^{x_i}, y_j = g^{x_j})$, $k_{ij} = g^{x_i x_j}$ is indistinguishable from random (Decision Diffie-Hellman Assumption)

Combining Diffie-Hellman key agreement protocol with the one-time pad, we can build a **Public-Key Encryption Scheme**

Public Key Encryption: Syntax



Public Key Encryption: Correctness



$$\forall m \in \mathcal{M}_\ell \ \forall (pk, sk) \leftarrow \mathsf{KeyGen}(\ell), \quad m = \mathsf{Dec}(sk, \mathsf{Enc}(pk, m))$$

The Man-in-the-Middle Attack

A	B
send(B, pk_A);	**receive**(pk_A)
	$c_B \leftarrow \text{Enc}(pk_A, m_B)$;
receive(c_B)	**send**(A, c_B);
$m_B \leftarrow \text{Dec}(sk_A, c_B)$;	

The Man-in-the-Middle Attack

In a multiparty scenario, the adversary C can impersonate a user by replacing the public key

A	C	B
send(B, pk_A);	**intercept**; **send**(B, pk');	**receive**(pk')
		$c_B \leftarrow \text{Enc}(pk', m_B)$; **send**$(A, c_B)$;
	intercept; $m_B \leftarrow \text{Dec}(sk', c_B)$; $c' \leftarrow \text{Enc}(pk_A, m')$; **send**$(A, c')$;	
receive(c') $m' \leftarrow \text{Dec}(sk_A, c')$;		

The Man-in-the-Middle Attack

Public keys need to be **certified**

A	C	B
send(B, pk_A);	**intercept**; **send**(B, pk');	**receive**(pk') **verify**(A, pk');
		$c_B \leftarrow \text{Enc}(pk', m_B)$; **send**$(A, c_B)$;
	intercept; $m_B \leftarrow \text{Dec}(sk', c_B)$; $c' \leftarrow \text{Enc}(pk_A, m')$; **send**$(A, c')$;	
receive(c') $m' \leftarrow \text{Dec}(sk_A, c')$;		

The Man-in-the-Middle Attack

Public keys need to be **certified**

- **Public Key Infrastructure:** Public keys are validated by a trusted entity

The Man-in-the-Middle Attack

Public keys need to be **certified**

- **Public Key Infrastructure:** Public keys are validated by a trusted entity
- **Identity Based Cryptography:** Public keys are just the users' identities, and then they are not replaceable (e.g., they are just the users' names or their email addresses)

The Man-in-the-Middle Attack

Public keys need to be **certified**

- **Public Key Infrastructure:** Public keys are validated by a trusted entity
- **Identity Based Cryptography:** Public keys are just the users' identities, and then they are not replaceable (e.g., they are just the users' names or their email addresses) . . . but then a Key Generation Center generates and then knows all the secret keys

Public Key Encryption: Security

Experiment Exp-PKE-OW-CPA$(\Pi, \mathcal{A}, \ell)$:
$(pk, sk) \leftarrow$ KeyGen(ℓ);
$m_* \leftarrow \mathcal{M}_\ell$;
$c_* \leftarrow$ Enc(pk, m_*);
$m' \leftarrow \mathcal{A}(1^\ell, pk, c_*)$;
if $m' = m_*$ **output** 1; //$\mathcal{A}$ wins
else output 0;

Definition (PKE-OW-CPA)

The public key encryption scheme Π is PKE-OW-CPA secure if for all PPTM, $\mathcal{A}$,

$$\Pr[\text{Exp-PKE-OW-CPA}(\Pi, \mathcal{A}, \ell) = 1] \in \textbf{negl}(\ell)$$

Public Key Encryption: Security

Experiment Exp-PKE-IND-CPA$(\Pi, \mathcal{A}_1, \mathcal{A}_2, \ell)$:
$(pk, sk) \leftarrow \text{KeyGen}(\ell)$;
$(m_0, m_1, s) \leftarrow \mathcal{A}_1(1^\ell, pk)$;
$b_* \leftarrow \{0,1\}$;
$c_* \leftarrow \text{Enc}(pk, m_{b_*})$;
$b' \leftarrow \mathcal{A}_2(1^\ell, c_*, s)$;
if $b' = b_*$ **and** $|m_0| = |m_1|$ **output** 1; //$\mathcal{A}$ wins
else output 0;

Definition (PKE-IND-CPA)

The public key encryption scheme Π is PKE-IND-CPA secure if for all **Two-Stages** PPOTM, $(\mathcal{A}_1, \mathcal{A}_2)$,

$$|\Pr[\text{Exp-PKE-IND-CPA}(\Pi, \mathcal{A}_1, \mathcal{A}_2, \ell) = 1] - 1/2| \in \textbf{negl}(\ell)$$

Public Key Encryption: Security

Experiment Exp-PKE-IND-CCA$(\Pi, \mathcal{A}_1, \mathcal{A}_2, \ell)$:
$(pk, sk) \leftarrow \text{KeyGen}(\ell)$;
$(m_0, m_1, s) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\text{Dec}}}(1^\ell, pk)$;
$b_* \leftarrow \{0,1\}$;
$c_* \leftarrow \text{Enc}(pk, m_{b_*})$;
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\text{Dec}}}(1^\ell, c_*, s)$;
if $b' = b_*$ **and** $|m_0| = |m_1|$ **output** 1; //$\mathcal{A}$ wins
else output 0;

Oracle $\mathcal{O}_{\text{Dec}}(c)$:
if $c = c_*$ **output** $\perp$; //Illegal oracle query
else output $\text{Dec}(sk, c)$;

Definition (PKE-IND-CCA)

The public key encryption scheme Π is PKE-IND-CCA secure if for all **Two-Stages** PPOTM, $(\mathcal{A}_1, \mathcal{A}_2)$,

$$|\Pr[\text{Exp-PKE-IND-CCA}(\Pi, \mathcal{A}_1, \mathcal{A}_2, \ell) = 1] - 1/2| \in \textbf{negl}(\ell)$$

Deterministic vs. Probabilistic Encryption

No **deterministic** PKE can be PKE-IND-CPA secure!

Deterministic vs. Probabilistic Encryption

No **deterministic** PKE can be PKE-IND-CPA secure!

Generic Attack: Encrypt m_1 and compare the result with c_*

Deterministic vs. Probabilistic Encryption

No **deterministic** PKE can be PKE-IND-CPA secure!

Generic Attack: Encrypt m_1 and compare the result with c_*

The attack still works if there are only polynomially many different encryptions for a given message m and public key pk.

Deterministic vs. Probabilistic Encryption

No **deterministic** PKE can be PKE-IND-CPA secure!

Generic Attack: Encrypt m_1 and compare the result with c_*

The attack still works if there are only polynomially many different encryptions for a given message m and public key pk.

. . . but a deterministic encryption scheme can still be PKE-OW-CPA secure (and it can achieve some even stronger security)

One-Way Functions

PKE implies the existence of function families easy-to-compute (encryption) but hard-to-invert (decryption)

One-Way Functions

PKE implies the existence of function families easy-to-compute (encryption) but hard-to-invert (decryption)

Definition

A function family $\mathcal{F} = \{\mathcal{F}_\ell\}_{\ell \in \mathbb{Z}^+}$, $\mathcal{F}_\ell = \{f_k : \mathcal{X}_k \to \mathcal{Y}_k\}_{k \in \mathcal{K}_\ell}$ is **one-way** if it is efficiently computable, but for all PPTM $\mathcal{A}$

$$\Pr[\mathcal{A}(1^\ell, k, y) \in f_k^{-1}(y) \ : \ k \leftarrow \mathcal{K}_\ell;\ x \leftarrow \mathcal{X}_k;\ y \leftarrow f_k(x)] \in \mathbf{negl}(\ell)$$

One-Way Functions

PKE implies the existence of function families easy-to-compute (encryption) but hard-to-invert (decryption)

Definition

A function family $\mathcal{F} = \{\mathcal{F}_\ell\}_{\ell \in \mathbb{Z}^+}$, $\mathcal{F}_\ell = \{f_k : \mathcal{X}_k \to \mathcal{Y}_k\}_{k \in \mathcal{K}_\ell}$ is **one-way** if it is efficiently computable, but for all PPTM $\mathcal{A}$

$$\Pr[\mathcal{A}(1^\ell, k, y) \in f_k^{-1}(y) \ : \ k \leftarrow \mathcal{K}_\ell;\ x \leftarrow \mathcal{X}_k;\ y \leftarrow f_k(x)] \in \mathbf{negl}(\ell)$$

- 'Efficiently computable' means that there is a PPTM Eval such that $\text{Eval}(k, x) = f_k(x)$
- If f_k is one-way then f'_k defined by $f'_k(x, r) = (f_k(x), r)$ is also one-way
- The sets $\mathcal{X}_k$ and $\mathcal{Y}_k$ must be of size superpolynomial in ℓ

Injective Trapdoor One-Way Functions

PKE also requires the existence of a **trapdoor** which knowledge renders the decryption function easy to compute.

Injective Trapdoor One-Way Functions

PKE also requires the existence of a **trapdoor** which knowledge renders the decryption function easy to compute.

Definition

An injective one-way function family $\mathcal{F} = \{\mathcal{F}_\ell\}_{\ell \in \mathbb{Z}^+}$, $\mathcal{F}_\ell = \{f_k : \mathcal{X}_k \to \mathcal{Y}_k\}_{k \in \mathcal{K}_\ell}$ is called **trapdoor one-way** if there exists a family of trapdoors $\mathcal{T} = \{\mathcal{T}_\ell\}_{\ell \in \mathbb{Z}^+}$, and two PPTM Sample and Inv such that

$$\Pr[\text{Inv}(1^\ell, t, f_k(x)) = x \ : \ (k,t) \leftarrow \text{Sample}(1^\ell);\ x \leftarrow \mathcal{X}_k] = 1$$

and $(k,t) \leftarrow \text{Sample}(1^\ell)$ samples the uniform distribution in $\mathcal{K}_\ell$, and $t \in \mathcal{T}_\ell$

PKE From Injective TOW Functions

Let $\mathcal{F}$ be an injective trapdoor one-way function family.

KeyGen(ℓ) :
$(k, t) \leftarrow$ Sample(ℓ);
output (k, t);

Enc(k, m) :
output Eval(k, m);

Dec(t, c) :
output Inv(t, c);

PKE From Injective TOW Functions

Let $\mathcal{F}$ be an injective trapdoor one-way function family.

KeyGen(ℓ) :
$(k, t) \leftarrow$ Sample(ℓ);
output (k, t);

Enc(k, m) :
output Eval(k, m);

Dec(t, c) :
output Inv(t, c);

It is PKE-OW-CPA secure but not PKE-IND-CPA secure (because the encryption function is deterministic)

Hardcore Predicates of a One-Way Function

Let $\mathcal{F}$ be an injective one-way function family between the set families $\mathcal{X}$ and $\mathcal{Y}$. A family of predicates $\mathcal{H}$ (functions taking binary values) on $\mathcal{X}$ is hardcore for $\mathcal{F}$ if computing $h_k(x)$ from $f_k(x)$ is as hard as computing x from $f_k(x)$.

Hardcore Predicates of a One-Way Function

Let $\mathcal{F}$ be an injective one-way function family between the set families $\mathcal{X}$ and $\mathcal{Y}$. A family of predicates $\mathcal{H}$ (functions taking binary values) on $\mathcal{X}$ is hardcore for $\mathcal{F}$ if computing $h_k(x)$ from $f_k(x)$ is as hard as computing x from $f_k(x)$.

Examples:

- For an RSA public key $(n = pq, e)$, computing $LSB(x)$ from $x^e \mod n$ is as hard as computing x from $x^e \mod n$
- **Goldreich-Levin predicate:**

$$h(x, r) = x_1 r_1 + \ldots + x_n r_n \mod 2$$

is a hardcore predicate for $f'_k(x, r) = (f_k(x), r)$

PKE From Hardcore Predicates

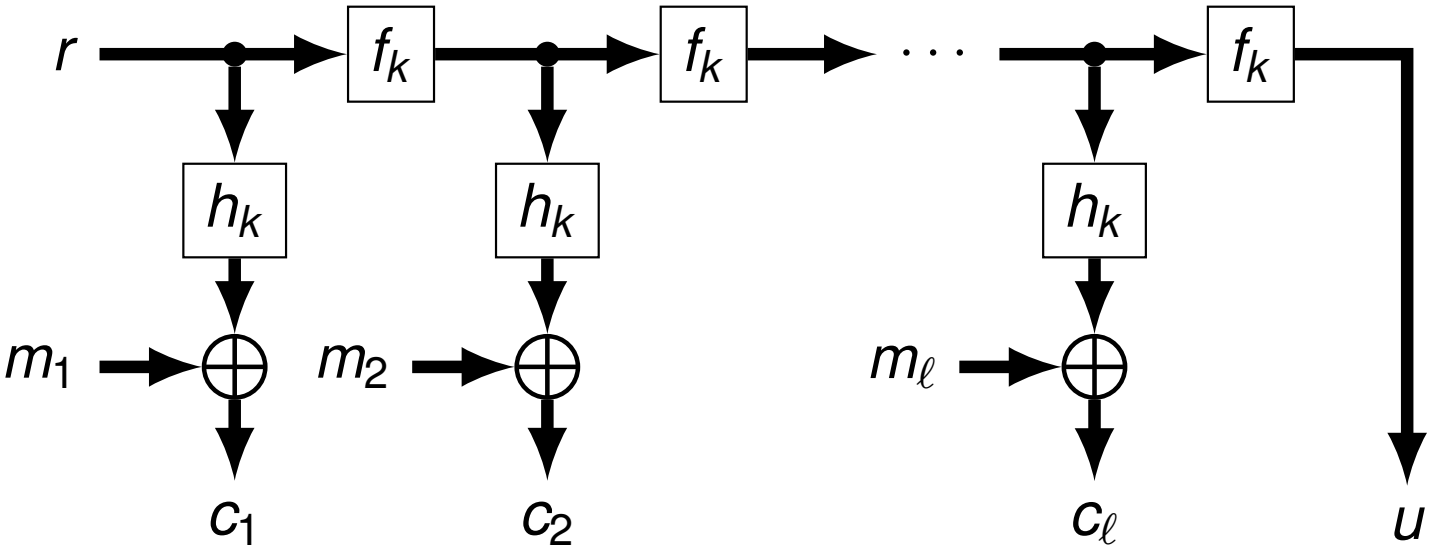
Trapdoor One-Way Permutation (TOWP) Family: A trapdoor one-way family of bijections $f_k : \mathcal{X}_k \to \mathcal{X}_k$
$\mathcal{H}$ hardcore predicate family for $\mathcal{F}$

KeyGen(ℓ) :
$(k, t) \leftarrow$ Sample$(\kappa(\ell))$;
output (k, t);

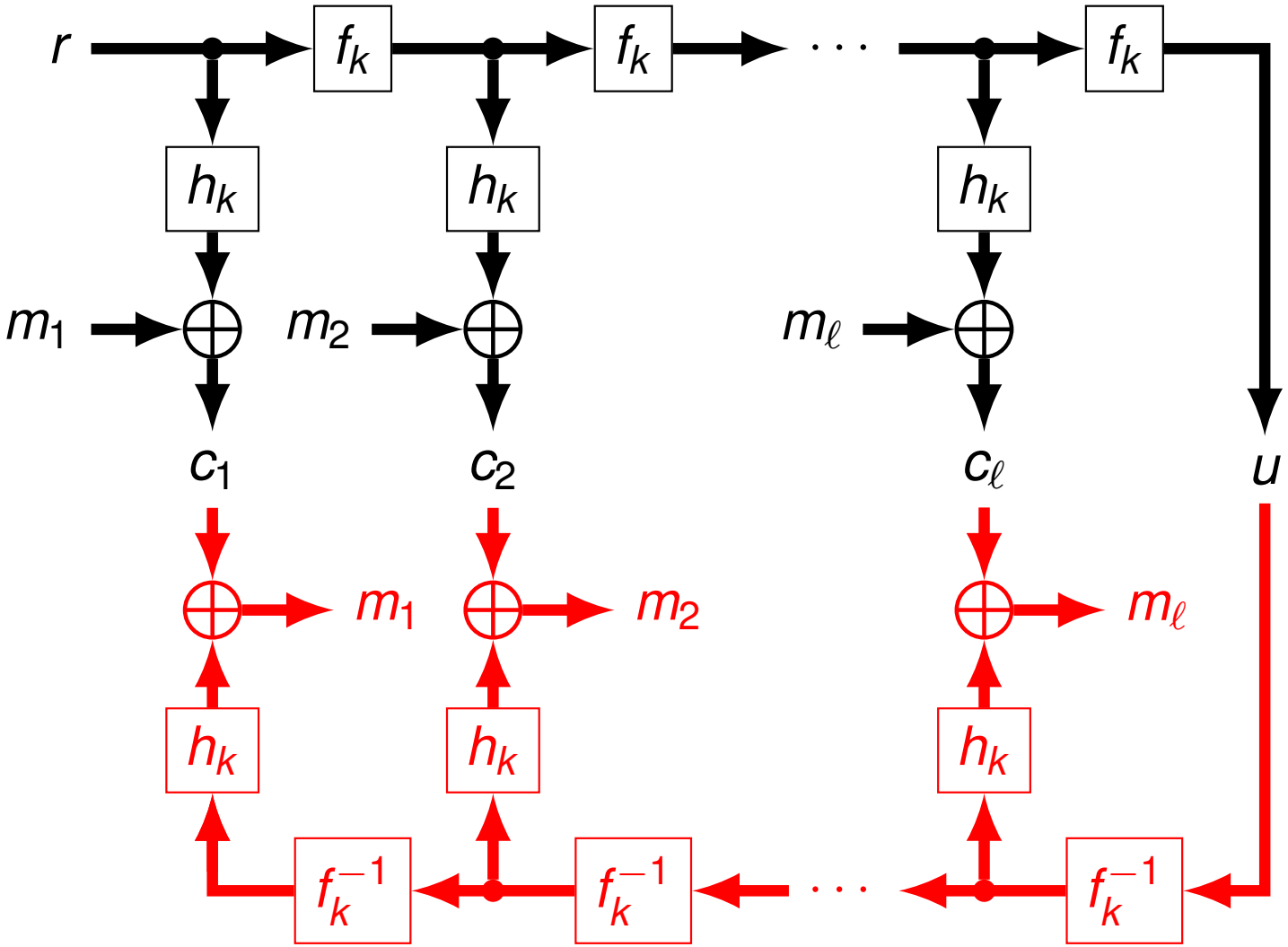
Enc(k, m) :
$(m_1, \ldots, m_\ell) = m$;
$r \leftarrow \mathcal{X}_k$;
$c_i \leftarrow m_i \oplus h_k(f_k^{i-1}(r)); \quad i = 1, \ldots, \ell$
output $(c_1, \ldots, c_\ell, f_k^\ell(r))$;

Dec(t, c) :
$(c_1, \ldots, c_\ell, u) = c$;
$m_i \leftarrow c_i \oplus h_k(f_k^{-(\ell-i+1)}(u)); \quad i = 1, \ldots, \ell$
output $m = (m_1, \ldots, m_\ell)$;

PKE From Hardcore Predicates



PKE From Hardcore Predicates



Outline

ElGamal PKE Scheme

Let $G = \langle g \rangle$ be a cyclic (multiplicative) group of ℓ bits long prime order q. Set $\mathcal{M} = \mathcal{PK} = G$, $\mathcal{SK} = \mathbb{Z}_q^\times$ and $\mathcal{C} = G \times G$.

KeyGen(ℓ) :
$(G, q, g) \leftarrow$ InstGen(ℓ);
$x \leftarrow \mathbb{Z}_q^\times$;
$y \leftarrow g^x$;
param $\leftarrow (G, q, g)$;
output (y, x);

Enc(param, y, m) :
$r \leftarrow \mathbb{Z}_q^\times$; //probabilistic encryption
output $(g^r, y^r m)$; //Diffie-Hellman Key & One-Time Pad

Dec(param, $x, (c_1, c_2)$) :
output $c_2 c_1^{-x}$; //randomness cancels out

ElGamal PKE Scheme

Theorem

ElGamal scheme is PKE-OW-CPA secure under the Computational Diffie-Hellman Assumption

Theorem

ElGamal scheme is PKE-IND-CPA secure under the Decision Diffie-Hellman Assumption

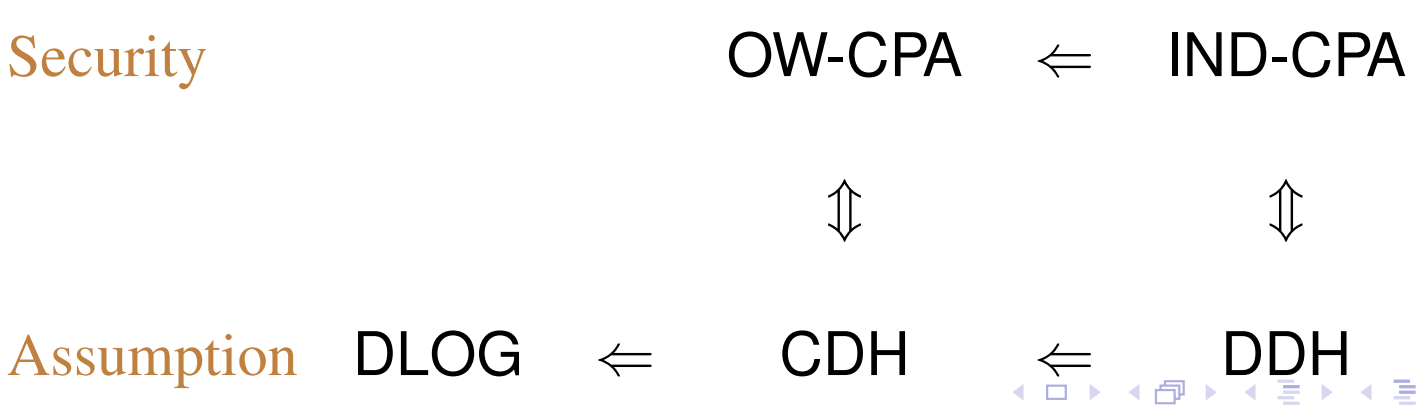
ElGamal PKE Scheme

Theorem

ElGamal scheme is PKE-OW-CPA secure under the Computational Diffie-Hellman Assumption

Theorem

ElGamal scheme is PKE-IND-CPA secure under the Decision Diffie-Hellman Assumption



Groups with Conjectured Hard DLOG

- Prime order subgroups of $GF(p)^{\times}$
  $|p| \approx 1024$, $|q| \approx 160$, due to some known subexponential attacks

Groups with Conjectured Hard DLOG

- Prime order subgroups of $GF(p)^{\times}$
  $|p| \approx 1024$, $|q| \approx 160$, due to some known subexponential attacks
- Prime order subgroups of the group of points of an elliptic curve over $GF(p)$
  $|p| \approx 160$, $|q| \approx 160$, only exponential attacks are known

Groups with Conjectured Hard DLOG

- Prime order subgroups of $GF(p)^{\times}$
  $|p| \approx 1024$, $|q| \approx 160$, due to some known subexponential attacks
- Prime order subgroups of the group of points of an elliptic curve over $GF(p)$
  $|p| \approx 160$, $|q| \approx 160$, only exponential attacks are known
  . . . but in some curves DDH is easy because of **efficient pairings**

Groups with Conjectured Hard DLOG

- Prime order subgroups of $GF(p)^{\times}$
  $|p| \approx 1024$, $|q| \approx 160$, due to some known subexponential attacks
- Prime order subgroups of the group of points of an elliptic curve over $GF(p)$
  $|p| \approx 160$, $|q| \approx 160$, only exponential attacks are known
  . . . but in some curves DDH is easy because of **efficient pairings**
- Jacobians of hyperelliptic curves can also be used

Groups with Conjectured Hard DLOG

- Prime order subgroups of $GF(p)^{\times}$
  $|p| \approx 1024$, $|q| \approx 160$, due to some known subexponential attacks
- Prime order subgroups of the group of points of an elliptic curve over $GF(p)$
  $|p| \approx 160$, $|q| \approx 160$, only exponential attacks are known
  . . . but in some curves DDH is easy because of **efficient pairings**
- Jacobians of hyperelliptic curves can also be used
- Diffie-Hellman Key Agreement can be generalized to some non-abelian groups

RSA PKE Scheme

Let $n = pq$, for two different $\ell/2$ bits long primes, and $e \geq 3$ coprime with $\phi(n) = (p-1)(q-1)$. Set $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n^\times$.

KeyGen(ℓ) :
$(p, q, e) \leftarrow$ InstGen(ℓ);
$d \leftarrow e^{-1} \mod \operatorname{lcm}(p-1, q-1)$;
param $\leftarrow n$;
output (e, d);

Enc(n, e, m) :
output $m^e \mod n$;

Dec(n, d, c) :
output $c^d \mod n$;

RSA PKE Scheme

Let $n = pq$, for two different $\ell/2$ bits long primes, and $e \geq 3$ coprime with $\phi(n) = (p-1)(q-1)$. Set $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n^\times$.

KeyGen(ℓ) :
$(p, q, e) \leftarrow$ InstGen(ℓ);
$d \leftarrow e^{-1} \mod \operatorname{lcm}(p-1, q-1)$;
param $\leftarrow n$;
output (e, d);

Enc(n, e, m) :
output $m^e \mod n$;

Dec(n, d, c) :
output $c^d \mod n$;

RSA is PKE-OW-CPA secure under the RSA assumption (not harder than the factoring assumption)

RSA PKE Scheme

Let $n = pq$, for two different $\ell/2$ bits long primes, and $e \geq 3$ coprime with $\phi(n) = (p-1)(q-1)$. Set $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n^\times$.

KeyGen(ℓ) :
$(p, q, e) \leftarrow$ InstGen(ℓ);
$d \leftarrow e^{-1} \mod \operatorname{lcm}(p-1, q-1)$;
param $\leftarrow n$;
output (e, d);

Enc(n, e, m) :
output $m^e \mod n$;

Dec(n, d, c) :
output $c^d \mod n$;

RSA is PKE-OW-CPA secure under the RSA assumption (not harder than the factoring assumption) . . . but cannot be PKE-IND-CPA secure!

Paillier PKE Scheme

$n = pq$, as in RSA, $\mathcal{M} = \mathbb{Z}_n$, $\mathcal{C} = \mathbb{Z}_{n^2}^{\times}$.

KeyGen(ℓ) :
$(p, q) \leftarrow$ InstGen(ℓ);
$\lambda \leftarrow \mathrm{lcm}(p-1, q-1)$;
output (n, λ);

Enc(n, m) :
$r \leftarrow \mathbb{Z}_n^{\times}$;
output $(1+mn)r^n \mod n^2$;

Dec(n, λ, c) : //RSA with $e = n$ gives another way to decrypt c
$c' \leftarrow c^{\lambda} \mod n^2$;
output $\frac{c'-1}{n}\lambda^{-1} \mod n$;

Paillier PKE Scheme

Theorem

Paillier PKE scheme is PKE-OW-CPA secure under the Composite Residuosity Assumption

Theorem

Paillier PKE scheme is PKE-IND-CPA secure under the Decision Composite Residuosity Assumption

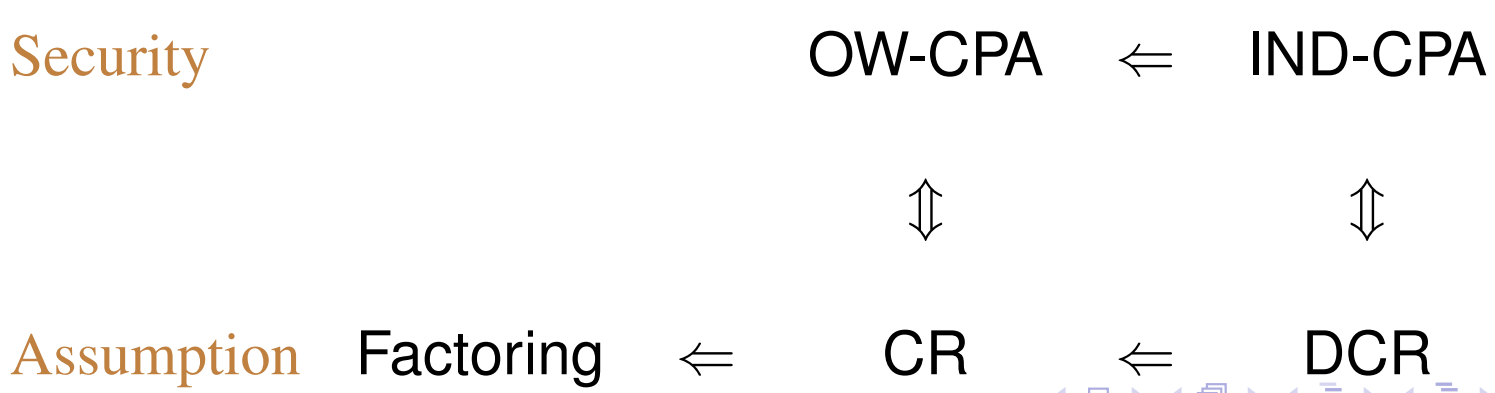
Paillier PKE Scheme

Theorem

Paillier PKE scheme is PKE-OW-CPA secure under the Composite Residuosity Assumption

Theorem

Paillier PKE scheme is PKE-IND-CPA secure under the Decision Composite Residuosity Assumption



Regev PKE

Discrete Learning With Errors (LWE) instance, for some parameters q, n, m and σ, and some fixed $\mathbf{s} \leftarrow \mathbb{Z}_q^n$:

$$(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + x)$$

where $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ and $x \leftarrow \lfloor qN(0, \sigma^2) \rceil \mod q$

Decision Discrete LWE Problem: Tell apart polynomially many Discrete LWE instances from Random instances

Regev PKE

For q, n, m and $\sigma(n)$ such that:

- $n^2 < q < 2n^2$, q prime
- $m = (1+\epsilon)(n+1)\log q$ for some ϵ
- $\sigma(n) \in o(1/\sqrt{n}\log n)$

Key Generation:
$A \leftarrow \mathbb{Z}_q^{m\times n}$; $\mathbf{s} \leftarrow \mathbb{Z}_q^{n\times 1}$; $\mathbf{x} \leftarrow \mathbb{Z}_q^{m\times 1}$;
according to $x_i \leftarrow \lfloor qN(0,\sigma^2)\rceil \mod q$
$pk = (A, \mathbf{b} = A\mathbf{s} + \mathbf{x})$; $sk = \mathbf{s}$

Encryption:
$\mathbf{r} \leftarrow \{0,1\}^{1\times m}$, $c = (\mathbf{r}A, \mathbf{rb} + m\frac{q-1}{2})$ for $m \in \{0,1\}$

Decryption:
Compute $c_2 - \mathbf{c_1}\cdot\mathbf{s}$ and decide by proximity to $\{0, \frac{q-1}{2}\}$

Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

END OF PART XIII