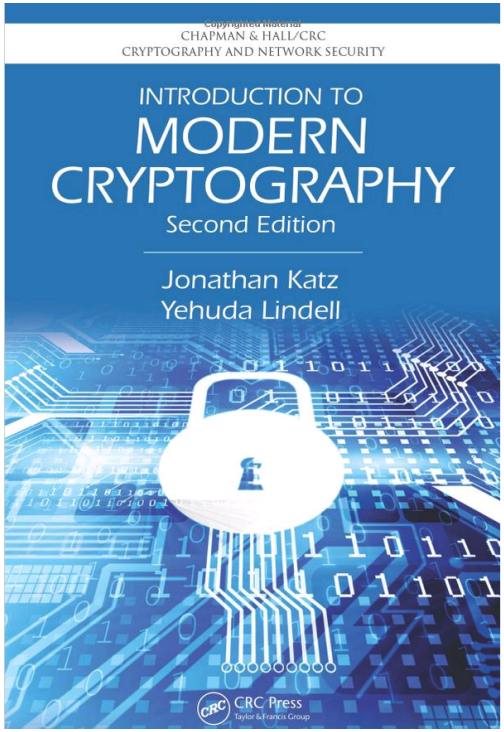
Public Key Cryptography

Federico Pintore [1]

[1] Mathematical Institute,
University of Oxford

Outline

Course main reference



Outline

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.
- **How can communicating parties share a secret key?**

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.
- **How can communicating parties share a secret key?**
- They may have access to a secure channel (e.g. being in the same place at some time, or using a trusted courier).

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.
- **How can communicating parties share a secret key?**
- They may have access to a secure channel (e.g. being in the same place at some time, or using a trusted courier).
- A secure channel is usually slow and costly!

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.
- **How can communicating parties share a secret key?**
- They may have access to a secure channel (e.g. being in the same place at some time, or using a trusted courier).
- A secure channel is usually slow and costly!
- It does not work well for *open systems*.

Private keys

- Assuming that two communicating parties share a secret key, private-key cryptography ensures secrecy and integrity.
- **How can communicating parties share a secret key?**
- They may have access to a secure channel (e.g. being in the same place at some time, or using a trusted courier).
- A secure channel is usually slow and costly!
- It does not work well for *open systems*.
- Each user has to securely store a big number of private keys.

Key-Distribution Centers (KDCs)

- A KDC is a trusted third party.
- Each user can share a key with the KDC through a secure channel.
- When Alice and Bob want to communicate, they query the KDC, which chooses a new, random key k and sends it over (encrypted using k_A to Alice, encrypted using k_B to Bob).
- Each user has to store only one long-term secret key.
- However, each user must trust the KDC. Furthermore the KDC is a single point of failure and a high-value target.
- **Still requires the use of a private channel!**

New Directions in Cryptography

In 1976, Diffie and Hellman published a paper, titled *New Directions in Cryptography*, that revolutionised Cryptography.

- They proposed an interactive protocol allowing two parties to share a secret key via communication over a public channel.
- They posed the first steps toward Public-key Cryptography, but they did not give any candidate construction.
- In 1977, Ron Rivest, Adi Shamir and Len Adleman introduced the RSA problem, and presented the first public-key encryption and digital signature schemes based on its hardness.

Outline

Key-exchange protocol

It is a probabilistic protocol Π to generate a shared, secret key.

- Alice and Bob begin by holding the security parameter n.
- They run Π using independent random bits.
- At the end of the protocol, they output k_A and k_B, respectively.

Correctness: with overwhelming probability $k_A = k_B$.

The key-exchange Experiment $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}$

Challenger Ch	**Adversary** $\mathcal{A}$
Execution of Π	Access to the transcript trans
$b \leftarrow \{0,1\}$	
If $b = 0, \hat{k} = k$	
else $\hat{k} \leftarrow \{0,1\}^n$	
$\xrightarrow{\hat{k}}$	
	Outputs his guess b'

Definition

The key-exchange protocol Π is secure if, for all PPT $\mathcal{A}$, the following holds:

$$\mathsf{Adv}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = \Pr[\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

The Discrete Logarithm Problem (Dlog)

Let $\mathcal{G}$ be a PPT group generation algorithm:

- On input n, it outputs a description of a cyclic group $\mathbb{G}$, its order q and a generator $g \in \mathbb{G}$.
- $||q|| = \lfloor \log_2 q \rfloor + 1 = n$
- The group operation is efficient in $\mathbb{G}$.
- Given $h \in \mathbb{G}$, $\log_g h$ denotes the unique $x \in \mathbb{Z}_q$ s.t. $h = g^x$.
- **Discrete logarithm (DLog) problem relative to $\mathcal{G}$:** given $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(n)$ and a uniform $h = g^x$, compute x.
- The DLog problem is hard relative to $\mathcal{G}$ if, for all PPT adversaries $\mathcal{A}$, the success probability is negligible in n.

Variants of Diffie-Hellman Problem

- Computational Diffie-Hellman (CDH): given uniform $g^a, g^b \in \mathbb{G}$, compute g^{ab}.
- If the DLog problem is easy relative to $\mathcal{G}$, then also the CDH problem is.
- The reverse implication is not clear.
- Decisional Diffie-Hellman (DDH): given $h, g^a, g^b \in \mathbb{G}$, decide if $h = g^{ab}$ or it is a uniform bit-string.
- If the CDH problem is easy relative to $\mathcal{G}$, then also the DDH problem is.
- The reverse implication does not appear to be true.
- There is a huge list of members in the DH family of problems!

Diffie-Hellman Key Exchange Algorithm

- Public elements: $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(n)$.
- Alice chooses random $a \in \mathbb{Z}_q$ and sends $h_A = g^a$ to Bob.
- Bob chooses random $b \in \mathbb{Z}_q$ and sends $h_B = g^b$ to Alice.
- Alice computes $(g^b)^a = g^{ab}$.
- Bob computes $(g^a)^b = g^{ab}$.

Diffie-Hellman security

- The hardness of the DLog problem is necessary for the security of the Diffie-Hellman key exchange.
- It may be not sufficient.
- The security follows *almost* directly from the hardness of the DDH problem.

Outline

Public Key Cryptosystems

In the public-key setting, a party generates a pair of keys: a public key and a private key.

They can be used for obtaining:

- secrecy for messages it receives using a **public-key encryption scheme**,
- integrity for messages it sends using a **digital signature scheme**.

Key distribution can be done over public, but authenticated channels. The need to store many secret keys is reduced. Suitable for *open systems*.

Public Key Cryptosystems

A public-key encryption scheme consists of the following algorithms:

- KeyGen(1^n): is a randomised algorithm that takes the security parameters as input and returns a pair of keys $(\mathsf{PK}, \mathsf{SK})$, the public key PK and its matching secret key SK, respectively.
- Enc(PK, m): An algorithm (possibly randomised) that takes a public key PK, a plaintext m and returns a ciphertext c.
- Dec(SK, c): A deterministic algorithm that takes the secret key SK and a ciphertext c, and returns a message $m \in \mathcal{M} \cup \perp$.

Correctness:

$$\forall m \in \mathcal{M}, \Pr[(\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{KeyGen}(n) : \mathsf{Dec}(\mathsf{Enc}(\mathsf{PK}, m), \mathsf{SK}) = m] = 1$$

The eavesdropping indistinguishability Experiment $\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},E}$

Challenger Ch

$\mathsf{PK}, \mathsf{SK} \leftarrow \mathsf{KeyGen}(1^n)$

Adversary $\mathcal{A}$

Access to PK

$\xleftarrow{m_0, m_1, |m_0| = |m_1|}$

$b \leftarrow \{0, 1\}$

$\xrightarrow{c = \mathsf{Enc}(\mathsf{PK}, m_b)}$ Outputs his guess b'

Definition

An encryption scheme E has indistinguishable encryptions in the presence of an eavesdropper if for all PPT adversaries $\mathcal{A}$ the following holds:

$$\mathsf{Adv}^{\mathsf{eav}}_{\mathcal{A},E}(n) = \Pr[\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},E}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

$\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},E}(n) = 1$ if $b' = b$; it is 0 otherwise.

CPA security

Since the adversary $\mathcal{A}$ knows the public key PK, it has access to an encryption oracle for free.

Consequently, if E has indistinguishable encryptions in the presence of an eavesdropper, then it is CPA-secure.

This is in contrast to the private-key setting.

Also in the public-key setting, a deterministic encryption scheme cannot be CPA-secure.

CCA Indistinguishability Experiment $\mathsf{PubK}^{\mathsf{cca}}_{\mathcal{A},E}$

Challenger Ch	**Adversary** $\mathcal{A}$
$\mathsf{PK}, \mathsf{SK} = \mathsf{KeyGen}(n)$	Access to PK and to the oracle $\mathsf{Dec}(\mathsf{SK}, \cdot)$
$\xleftarrow{m_0, m_1, \lvert m_0\rvert = \lvert m_1\rvert}$	
$b \leftarrow \{0,1\}$	
$\xrightarrow{c = \mathsf{Enc}(\mathsf{PK}, m_b)}$	Access to the oracle $\mathsf{Dec}(\mathsf{SK}, \cdot)^c$
	Outputs his guess b'

Definition

An encryption scheme is CCA-secure if for all PPT adversaries $\mathcal{A}$ the following holds:

$$\mathsf{Adv}^{\mathsf{cca}}_{\mathcal{A},E}(n) = \Pr[\mathsf{PubK}^{\mathsf{cca}}_{\mathcal{A},E}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

Dealing with arbitrary-length messages

In the indistinguishability of multiple encryptions experiment, the adversary is given access to a *letf-or-right* encryption oracle which, on input a pair of messages m_0, m_1 (with $|m_0| = |m_1|$), returns $c \leftarrow \text{Enc}(\text{PK}, m_b)$.

Theorem

If a public-key encryption scheme is CPA-secure, then it also has indistinguishable multiple encryptions.

- As a consequence, any CPA-secure public-key encryption scheme for fixed-length messages (down to one bit!) can be used as a CPA-secure public key-encryption scheme for arbitrary-length messages.

Hybrid Encryption

- Let E be a public-key encryption scheme for ℓ-bit messages. Using E for encrypting an ℓ'-bit message requires $\gamma = \lceil \ell'/\ell \rceil$ applications of E.
- A better approach to deal with arbitrary-length messages is possible.
- Exploit a private-key encryption scheme to obtain a public-key encryption scheme.
- Private-key encryption schemes are significantly faster (2 or 3 orders of magnitude) than public ones.
- This approach is called *key-encapsulation mechanism* (KEM), and *data-encapsulation mechanism* (DEM).

KEM

A key-encapsulation mechanism (KEM) consists of the following PPT algorithms:

- KeyGen(1^n): takes the security parameter as input and returns a pair of keys $(\mathsf{PK}, \mathsf{SK})$, the public key PK and its matching secret key SK, respectively, each of length n.
- Encaps$(\mathsf{PK}, 1^n)$: it returns a ciphertext c and a key $k \in \{0,1\}^{\ell(n)}$.
- Decaps(SK, c): a deterministic algorithm that takes a secret key SK and a ciphertext c, and returns a key k or $\perp$.

Correctness:

$$\Pr[(\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{KeyGen}(n), (c, k) \leftarrow \mathsf{Encaps}(\mathsf{PK}, 1^n) : \\ : \mathsf{Decaps}(\mathsf{SK}, c) = k] = 1$$

Hybrid Encryption

A hybrid encryption scheme $(\text{KeyGen}^{hy}, \text{Enc}^{hy}, \text{Dec}^{hy})$ is a public-key encryption scheme obtained combining a KEM $(\text{KeyGen}, \text{Encaps}, \text{Decaps})$ and a private-key encryption scheme $(\text{KeyGen}', \text{Enc}, \text{Dec})$ as follows:

- $\text{KeyGen}^{hy}(1^n)$: is a randomized algorithm that takes the security parameter as input and returns a pair of keys (PK, SK).
- $\text{Enc}^{hy}(\text{PK}, m \in \{0,1\}^*)$: takes a public key PK, a plaintext m and does the following:
  - Compute $(c, k) \leftarrow \text{Encaps}(\text{PK}, 1^n)$.
  - Compute $c' \leftarrow \text{Enc}(k, m)$.
  - Output the ciphertext (c, c').
- $\text{Dec}^{hy}(\text{SK}, (c, c'))$: takes a secret key SK and a ciphertext (c, c') and does the following:
  - Compute $k \leftarrow \text{Decaps}(\text{SK}, c)$.
  - Output $m \leftarrow \text{Dec}(k, c')$.

Hybrid Encryption: Efficiency

- Fix n. Let $\alpha = \text{cost}(\text{Encaps}(\cdot, 1^n))$ and $\beta = \text{cost}(\text{Enc}(\cdot, 1 \text{ bit}))$. Suppose $|m| > n$. Then

$$\text{cost}(\text{Enc}^{hy}(\cdot, 1 \text{ bit})) = \frac{\alpha + \beta \cdot |m|}{|m|} = \frac{\alpha}{|m|} + \beta$$

- For sufficiently long m, $\text{cost}(\text{Enc}^{hy}(1 \text{ bit}))$ approaches β, i.e. $\text{cost}(\text{Enc}^{hy}(1 \text{ bit})) \approx \text{cost}(\text{Enc}(1 \text{ bit}))$. In other words, the cost of encrypting one bit using the constructed public-key encryption scheme is approximately the cost of encrypting one bit using the private-key encryption scheme!

Security of KEM

Intuitively speaking, for a KEM to be CPA-secure, we require the encapsulated key to be indistinguishable from a uniform key that is independent of the ciphertext.

Experiment (CPA Indistinguishability $\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$)

- *Run* $\mathsf{KeyGen}(1^n)$ *to get* $(\mathsf{PK}, \mathsf{SK})$*, then run* $\mathsf{Encaps}(\mathsf{PK}, 1^n)$ *to generate* (c, k)*, where we assume* $k \in \{0,1\}^n$*.*
- *Choose random* $b \in \{0,1\}$*: if* $b = 0$ *set* $\bar{k} := k$*, otherwise choose* $\bar{k}$ *uniformly at random from* $\{0,1\}^n$*.*
- *The tuple* $(\mathsf{PK}, c, \bar{k})$ *is given to* $\mathcal{A}$*, who outputs a bit* b'*.*
- *Experiment output: 1 if* $b' = b$*, 0 otherwise.*

A KEM Π is CPA-secure if, for all PPT adversaries $\mathcal{A}$, we have

$$\mathsf{Adv}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = \Pr[\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

Security of the Hyprid Encryption Scheme

Theorem

If Π is a CPA-secure key-encapsulation mechanism and E is a private-key encryption scheme which has indistinguishable encryptions in the presence of an eavesdropper, the corresponding hybrid encryption scheme E^{hy} is a CPA-secure public-key encryption scheme.

Security of the Hybrid Encryption Scheme

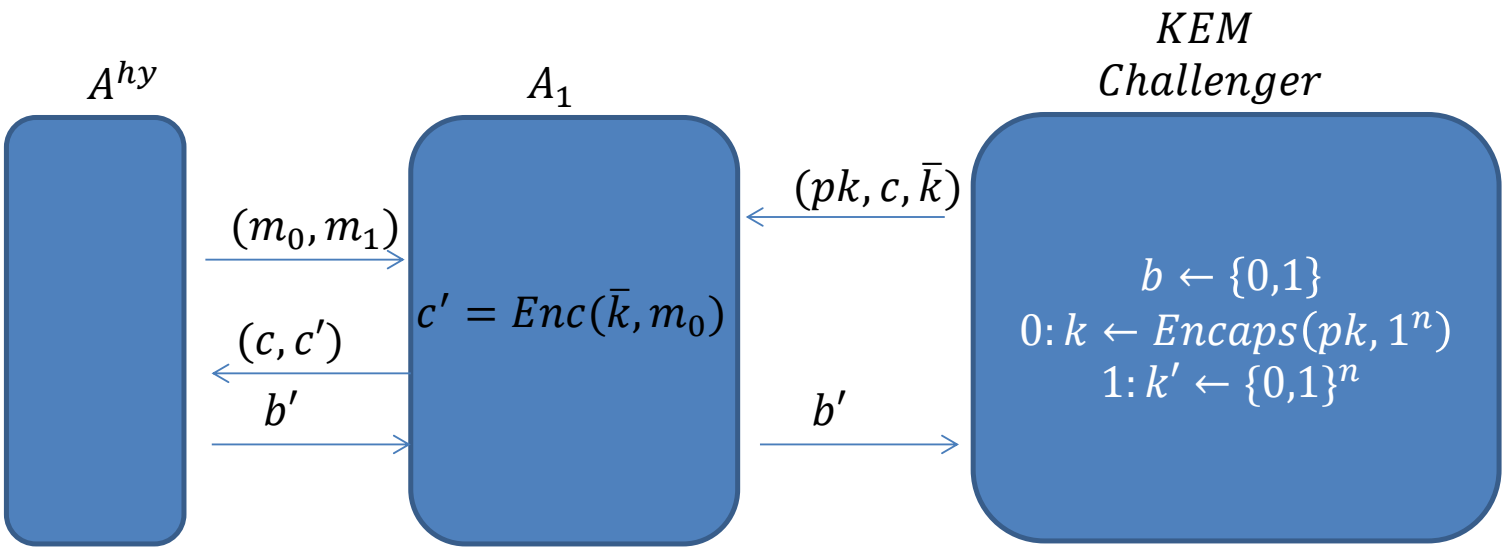
Let $\mathcal{A}^{hy}$ be an adversary playing the $\mathsf{PubK}^{eav}_{\mathcal{A}^{hy},S^{hy}}(n)$ game. We need to prove the following:

$$\Pr[\mathsf{PubK}^{eav}_{\mathcal{A}^{hy},S^{hy}}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

where

$$\begin{aligned}\Pr[\mathsf{PubK}^{eav}_{\mathcal{A}^{hy},S^{hy}}(n) = 1] = \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | m = m_0] \\ + \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | m = m_1]\end{aligned}$$

Security of the Hybrid Encryption Scheme



$$\mathbf{Pr}[A_1' s\ output = 0 | b = 0] = \mathbf{Pr}[A^{hy'} s\ output = 0 | \bar{k} = k, m = m_0]$$

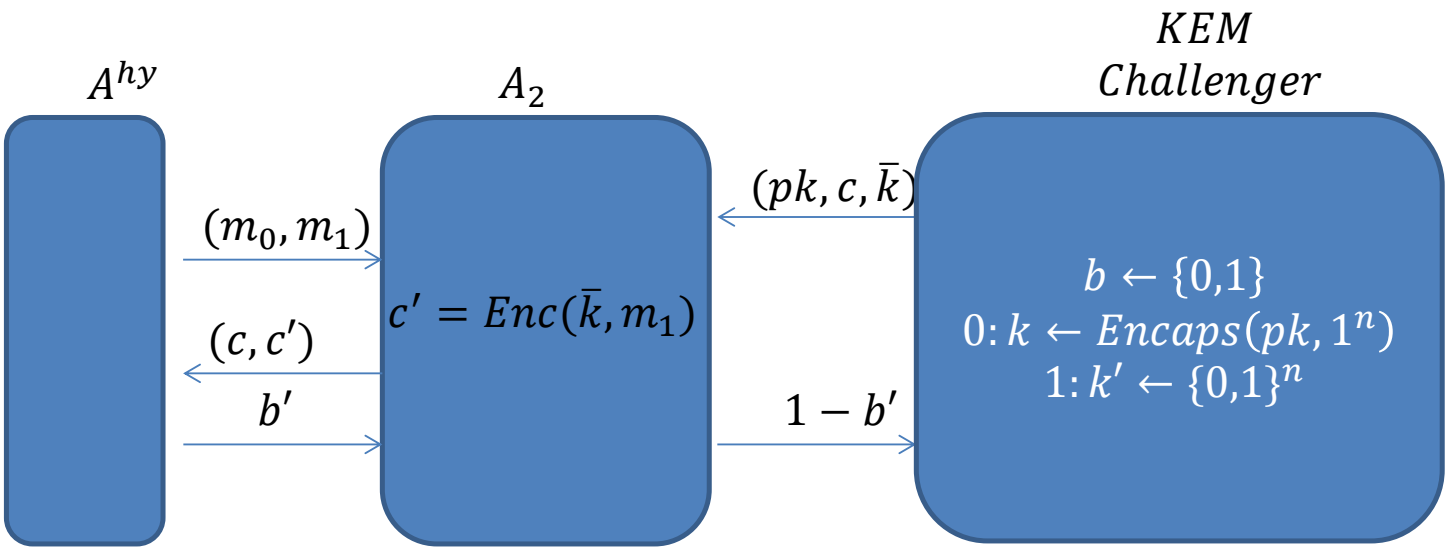
$$\mathbf{Pr}[A_1' s\ output = 1 | b = 1] = \mathbf{Pr}[A^{hy'} s\ output = 1 | \bar{k} = k', m = m_0]$$

Security of the Hybrid Encryption Scheme

Since the key-encapsulation scheme Π is CPA-secure, we have:

$$\begin{aligned}
\Pr[\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A}_1,\Pi}(n) = 1] &= \frac{1}{2}\Pr[\mathcal{A}_1 \text{ outputs } 0 | b = 0] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}_1 \text{ outputs } 1 | b = 1] = \\
&= \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k, m = m_0] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k', m = m_0] \leq \\
&\leq \frac{1}{2} + \mathsf{negl}_1(n)
\end{aligned}$$

Security of the Hybrid Encryption Scheme



$$\Pr[A_2\text{'s output} = 0 | b = 0] = \Pr[A^{hy'}\text{s output} = 1 | \bar{k} = k, m = m_1]$$

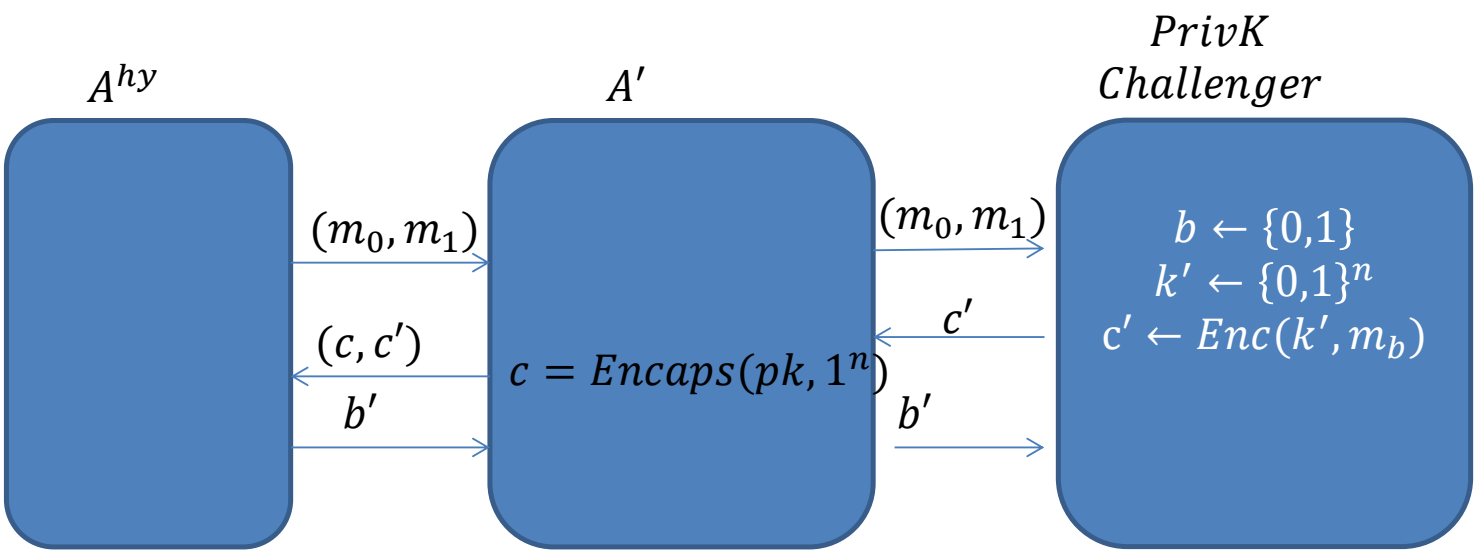
$$\Pr[A_2\text{'s output} = 1 | b = 1] = \Pr[A^{hy'}\text{s output} = 0 | \bar{k} = k', m = m_1]$$

Security of the Hybrid Encryption Scheme

Since the key-encapsulation scheme Π is CPA-secure, we have:

$$\begin{aligned}
\Pr[\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A}_2,\Pi}(n) = 1] &= \frac{1}{2}\Pr[\mathcal{A}_2 \text{ outputs } 0 | b = 0] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}_2 \text{ outputs } 1 | b = 1] = \\
&= \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k, m = m_1] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k', m = m_1] \leq \\
&\leq \frac{1}{2} + \mathsf{negl}_2(n)
\end{aligned}$$

Security of the Hybrid Encryption Scheme



$$\mathbf{Pr}[A' \textbf{ output} = 0 | b = 0] = \mathbf{Pr}[A^{hy'} \textbf{s output} = 0 | \bar{k} = k', m = m_0]$$

$$\mathbf{Pr}[A' \textbf{ output} = 1 | b = 1] = \mathbf{Pr}[A^{hy'} \textbf{s output} = 1 | \bar{k} = k', m = m_1]$$

Security of the Hybrid Encryption Scheme

Since the private-key encryption scheme E has indistinguishable encryptions in the presence of an eavesdropper, we have:

$$\begin{aligned}
\Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A}',E}(n) = 1] &= \frac{1}{2}\Pr[\mathcal{A}' \text{ outputs } 0 | b = 0] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}' \text{ outputs } 1 | b = 1] = \\
&= \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k', m = m_0] + \\
&+ \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k', m = m_1] \leq \\
&\leq \frac{1}{2} + \mathsf{negl}_3(n)
\end{aligned}$$

Security of the Hybrid Encryption Scheme

The sum of negligible functions is negligible as well. Summing all the above inequalities we obtain:

$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0|\bar{k}=k, m=m_0] + \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1|\bar{k}=k', m=m_0]$$
$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1|\bar{k}=k, m=m_1] + \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0|\bar{k}=k', m=m_1]$$
$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0|\bar{k}=k', m=m_0] + \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1|\bar{k}=k', m=m_1]$$
$$\leq \frac{3}{2} + \mathsf{negl}(n)$$

Security of the Hybrid Encryption Scheme

Furthermore, we have:

$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k', m = m_0] +$$
$$+ \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k', m = m_0] = \frac{1}{2}$$

and

$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k', m = m_1] +$$
$$+ \frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k', m = m_1] = \frac{1}{2}$$

Security of the Hybrid Encryption Scheme

Hence it remains

$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 0 | \bar{k} = k, m = m_0] +$$
$$\frac{1}{2}\Pr[\mathcal{A}^{hy} \text{ outputs } 1 | \bar{k} = k, m = m_1] =$$
$$\Pr[\mathsf{PubK}^{eav}_{\mathcal{A}^{hy}}] \leq \frac{1}{2} + \mathsf{negl}(n)$$

which concludes the proof.

Security of the Hybrid Encryption Scheme

For the CCA-security of a key-encapsulation mechanism, we consider an experiment similar to $\mathrm{KEM}^{\mathrm{cpa}}_{\mathcal{A},\Pi}(n)$, where the adversary $\mathcal{A}$ is also given access to a decapsulation oracle $\mathrm{Decaps}(\mathrm{SK},\cdot)$.

Theorem

If KEM *is a CCA-secure key-encapsulation mechanism and* E *is a CCA-secure private-key encryption scheme, the corresponding hybrid encryption scheme* E^{hy} *is a CCA-secure public-key encryption scheme.*

Outline

ElGamal Encryption Scheme

The security of the scheme relies on the following main result.

- **Lemma**: Let $\mathbb{G}$ be a finite group. Given an arbitrary element $m \in \mathbb{G}$, if m is multiplied by an uniform group element $k \in \mathbb{G}$, the result $k \cdot m$ is a uniform group element as well.
- **Proof**: let g be an arbitrary element of $\mathbb{G}$, then

$$\Pr[k \cdot m = g] = \Pr[k = g \cdot m^{-1}].$$

and, because k is uniform, we obtain

$$\Pr[k = g \cdot m^{-1}] = 1/|\mathbb{G}|.$$

ElGamal Encryption Scheme - Construction

The ElGamal public-key encryption scheme is defined as follows:

- KeyGen(1^n): on input 1^n, it runs $\mathcal{G}$ to generate a description of a cyclic group $\mathbb{G}$ - having order q, with $||q|| = n$ - together with a generator g. Then, it picks a uniform $x \in \mathbb{Z}_q$ to compute $h \leftarrow g^x$. The public key is PK $= (\mathbb{G}, g, q, h)$ and the private/secret key is SK $= x$. The message space is $\mathbb{G}$.
- Enc(PK, $m \in \mathbb{G}$): it chooses a uniform $y \in \mathbb{Z}_q$, and outputs the following ciphertext

$$c = (c_1, c_2) \leftarrow (g^y, h^y \cdot m).$$

- Dec(SK, c): it outputs

$$m' = c_2 / c_1^x$$

Correctness: $c_2/c_1^x = h^y \cdot m/(g^y)^x = m$.

ElGamal Encryption scheme - Example

Example (Katz-Lindell book)

Let $q = 83$ and $p = 2q + 1 = 167$. Let $\mathbb{G}$ denote the group of quadratic residues mod p. Both p and q are primes, and $\mathbb{G}$ is a subgroup of $\mathbb{Z}_p^*$ with order q. Then, any element $g \in \mathbb{G} \setminus \{1\}$ is a generator. Take $g = 2^2 = 4 \mod 167$, pick $x = 37 \in \mathbb{Z}_{83}$, compute $h = g^x = 4^{37} \mod 167 = 76$. The public key becomes
$\text{PK} = (p, q, g, h) = (167, 83, 4, 76)$

- $\text{Enc}(\text{PK}, m = 65 \in \mathbb{G})$: [a] it picks $y = 71$ and compute the ciphertext,

$$c = (c_1, c_2) = (4^{71}, 76^{71} \cdot 65) = (132, 44) \mod 167$$

[a] 65 is indeed in $\mathbb{G}$ as $65 = 30^2 \mod 167$.

ElGamal Encryption Scheme - Example

Example (Katz-Lindell book)

- $\mathrm{Dec}(\mathrm{SK}, c)$:

$$\begin{aligned} m &= c_2 / c_1^x \\ &= 44 / 132^{37} \mod 167 \\ &= 44 / 124 \mod 167 \\ &= 44 \cdot 124^{-1} \mod 167 \\ &= 44 \cdot 66 \mod 167 \\ &= 65 \end{aligned}$$

Security of the ElGamal Encryption Scheme

Theorem

If the DDH problem is hard relative to $\mathcal{G}$, then the ElGamal encryption scheme is CPA-secure.

Sketch Proof.

***Idea**: we consider a PPT adversary $\mathcal{A}$ who is attacking the ElGamal scheme S, and we construct a PPT distinguisher $\mathcal{D}$ that attempts to solve the DDH problem relative to $\mathcal{G}$. $\mathcal{D}$ first receives an instance of the DDH problem, i.e. $(\mathbb{G}, q, g, h_1 = g^{x_1}, h_2 = g^{x_2}, h_3)$, and its challenge is to determine whether $h_3 = g^{x_1 x_2}$ or $h_3 = g^z$ for uniform $z \in \mathbb{Z}_q$.*

Security of ElGamal Encryption Scheme

Sketch Proof.

Algorithm $\mathcal{D}$ will simulate the ElGamal scheme to $\mathcal{A}$ as follows:

- *On input $(\mathbb{G}, q, g, h_1, h_2, h_3)$, it sets* $\mathsf{PK} = (\mathbb{G}, q, g, h_1)$.
- *On input (m_0, m_1) received from $\mathcal{A}$, it picks $b \in \{0, 1\}$, sets $c_1 = h_2$ and $c_2 = h_3 \cdot m_b$, and sends (c_1, c_2) to $\mathcal{A}$.*
- *It receives the bit b' from $\mathcal{A}$, and outputs 1 if $b' = b$, 0 otherwise.*

Now, let S' be a modified version of ElGamal, working as follows:

- *It has the same key generation algorithm.*
- *Encryption algorithm: it chooses uniform $y, z \in \mathbb{Z}_q$, and outputs the ciphertext $(g^y, g^z \cdot m)$. Note that the decryption algorithm doesn't work here, but we don't actually need it in the experiment* $\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},S'}(n) = 1$.

Security of ElGamal Encryption Scheme

Sketch Proof.

For the modified encryption scheme S', since c_2 is a uniformly distributed group element, we have

$$\Pr[\mathsf{PubK}^{eav}_{\mathcal{A},S'}(n) = 1] = 1/2$$

Case 1 - random tuple: *the view of the adversary $\mathcal{A}$ when run as a subroutine by $\mathcal{D}$ is distributed identically to its view in experiment* $\mathsf{PubK}^{eav}_{\mathcal{A},S'}$*. Therefore*

$$\Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] = \Pr[\mathsf{PubK}^{eav}_{\mathcal{A},S'}(n) = 1] = 1/2 \quad (1)$$

Security of ElGamal Encryption Scheme

Sketch Proof.

Case 2 - DH tuple: *the view of the adversary $\mathcal{A}$ when run as a subroutine by $\mathcal{D}$ is distributed identically to its view in experiment* $\mathsf{PubK}^{eav}_{\mathcal{A},S}$. *Therefore*

$$\Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1] = \Pr[\mathsf{PubK}^{eav}_{\mathcal{A},S}(n) = 1] \tag{2}$$

Concluding, if the DDH problem is hard relative to $\mathcal{G}$, then

$$|\Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - \Pr[\mathcal{D}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1]| \leq \mathsf{negl}(n) \tag{3}$$

From equations (1), (2) and (3) we deduce

$$\Pr[\mathsf{PubK}^{eav}_{\mathcal{A},S}(n)] \leq 1/2 + \mathsf{negl}(n)$$

ElGamal Encryption Scheme - CCA-secure?

- The ElGamal encryption scheme is malleable, hence it is not CCA-secure.
- **Malleability**: given a ciphertext c, which is the encryption of a message m, it is possible to generate the encryption c' of a message m' having some known relation with m.
- Consider the public key $(\mathbb{G}, q, g, h)$, the ciphertext $c = (c_1, c_2)$ and its modification $c' = (c_1, c_2' = \alpha \cdot c_2)$, where $\alpha \in \mathbb{G}$. If c is the encryption of m, we have $c_1 = g^y, c_2 = h^y \cdot m$. Hence c' is a valid encryption of $\alpha \cdot m$.

A CPA-secure KEM based on DDH

Consider the key-encapsulation mechanism defined as follows:

- KeyGen(1^n): it runs $\mathcal{G}$ to generate $(\mathbb{G}, q, g)$. It then chooses $x \in \mathbb{Z}_q$ and computes $h = g^x$. It also specifies a hash function $H : \mathbb{G} \to \{0,1\}^{\ell(n)}$. The public key is PK $= (\mathbb{G}, q, g, h, H)$ and the private key is x.
- Encaps(PK): it chooses a uniform $y \in \mathbb{Z}_q$ and outputs the ciphertext $c := g^y$ and the key $H(h^y)$.
- Decaps(SK, c): it outputs $H(c^x)$.

If H is modelled as a random oracle, then the above KEM is CPA-secure based on the hardness of the CDH problem relative to $\mathcal{G}$ (weaker assumption).

Outline

Cramer-Shoup Encryption Scheme

- The first public-key encryption scheme that can be proven CCA-secure in the standard model.
- It is based on the ElGamal Encryption Scheme.
- Its CCA-security relies on the hardness of the DDH problem.

Cramer-Shoup Encryption Scheme

- KeyGen(n): first, it runs $\mathcal{G}(n)$ to obtain a description of a cyclic group $\mathbb{G}$ - having prime order q, where $||q|| = \lfloor \log_2 q + 1 \rfloor = n$ - and a couple of generators g_1, g_2 for $\mathbb{G}$. Then, it picks uniform $x_1, x_2, y_1, y_2, z_1, z_2 \in \mathbb{Z}_q$ and computes:
  - $c \leftarrow g_1^{x_1} g_2^{x_2}$
  - $d \leftarrow g_1^{y_1} g_2^{y_2}$
  - $h \leftarrow g_1^{z_1} g_2^{z_2}$

  The public key is $\text{PK} = (\mathbb{G}, q, g_1, g_2, c, d, h, H)$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is a collision-resistant hash function. The private/secret key is $\text{SK} = (x_1, x_2, y_1, y_2, z_1, z_2)$. The message space is $\mathbb{G}$.

Cramer-Shoup Encryption Scheme

- Enc(PK, $m \in \mathbb{G}$): it chooses a uniform $k \in \mathbb{Z}_q$, and computes:
  - $u_1 = g_1^k, u_2 = g_2^k$
  - $e = h^k m$
  - $\alpha = H(u_1, u_2, e)$
  - $v = c^k d^{k\alpha}$

  The ciphertext is $CT = (u_1, u_2, e, v)$

Cramer-Shoup Encryption Scheme

- Dec(CT, SK):
  - It computes $\alpha = H(u_1, u_2, e)$.
  - If $u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha} \neq v$, it outputs $\perp$.
  - Otherwise it outputs $m' = e/(u_1^{z_1} u_2^{z_2})$

Correctness:

$$m' = e/(u_1^{z_1} u_2^{z_2}) = h^k m / g_1^{kz_1} g_2^{kz_2} = h^k m / h^k = m$$

Cramer-Shoup: Security Proof

Let $\mathcal{A}$ be an arbitrary PPT adversary in experiment $\mathsf{PubK}^{\mathsf{cca}}_{\mathcal{A},CS}$, where CS it the Cramer-Shoup scheme. $\mathcal{A}$ is exploited to construct a distinguisher $\mathcal{D}$ for the DDH problem relative to $\mathcal{G}$.

Proof.

Distinguisher $\mathcal{D}(\mathbb{G}, q, g_1, g_2, g_3, g_4)$

- $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow \mathbb{Z}_q$.
- $\mathsf{PK} = (\mathbb{G}, q, g_1, g_2, c := g_1^{x_1} g_2^{x_2}, d := g_1^{y_1} g_2^{y_2}, h := g_1^{z_1} g_2^{z_2}, H)$.
- $(m_0, m_1) \leftarrow \mathcal{A}(\mathsf{PK}, \mathsf{Dec}(\mathsf{SK}, \cdot))$.
- $b \leftarrow \{0, 1\}$.
- $e^* = g_3^{z_1} g_4^{z_2} m_b$, $\alpha^* = H(g_3, g_4, e^*)$,
  $CT^* = (g_3, g_4, g_3^{z_1} g_4^{z_2} m_b, g_3^{x_1 + \alpha^* y_1} g_4^{x_2 + \alpha^* y_2})$.
- $b' \leftarrow \mathcal{A}(\mathsf{PK}, CT^*, \mathsf{Dec}(\mathsf{SK}, \cdot)^{CT^*})$.
- The distinguisher outputs 1 if $b' = b$, otherwise 0.

Cramer-Shoup: Security Proof

Proof.

Decryption queries:

On input $(u_1, u_2, e, v) \in \mathbb{G}^4$, $\mathcal{D}$ computes $\alpha = H(u_1, u_2, e)$. If

$$u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2} \neq v$$

it outputs $\perp$, otherwise it outputs

$$m' = \frac{e}{u_1^{z_1} u_2^{z_2}}.$$

Cramer-Shoup: Security Proof

Proof.

Fact 1:

$$\left| \Pr[\mathcal{D} = 1 | DH] - \Pr[\mathcal{D} = 1 | \text{Random}] \right| \leq \text{negl}(n)$$

[It follows from the DDH assumption.]

Fact 2:

$$\Pr[\mathcal{D} = 1 | \text{DH}] = \Pr[b' = b | \mathcal{A} \text{ attacks } CS]$$

Fact 3:

$$\left| \Pr[\mathcal{D} = 1 | \text{Random}] \right| \leq \frac{1}{2} + \text{negl}(n)$$

Cramer-Shoup: Security Proof

Proof.

Proof of Fact 2;

When $\mathcal{D}$ gets a valid DH tuple, then there exist γ, r s.t.:

$$(g_1, g_2 = g_1^{\gamma}, g_3 = g_1^{r}, g_4 = g_2^{r}).$$

It is easy to verify that the distribution of PK, CT^* and the decryption answers are exactly the same of those obtained from the proper Cramer-Shoup challenger (and not from the distinguisher who is simulating the game). Therefore,

$$\Pr[\mathcal{D} = 1 | \mathsf{DH}] = \Pr[b' = b | \mathcal{A} \text{ attacks } CS] = \Pr[\mathsf{PubK}^{cca}_{\mathcal{A},CS}(n) = 1]$$

Cramer-Shoup: Security Proof

Proof.

Proof of Fact 3 (a bit longer...)

When $\mathcal{D}$ gets a random tuple, it will look like

$$(g_1, g_2 = g_1^{\gamma}, g_3 = g_1^{r}, g_4 = g_2^{r'})$$

where $\gamma \neq 0$ and $r \neq r'$ with overwhelming probability $((2q^2 - 1)/q^3)$.

We show that, even if $\mathcal{A}$ can compute discrete logarithms (in real world, it cannot, since it runs in polynomial time), we have

$$\left| \Pr[\mathcal{D} = 1 | \text{Random}] \right| \leq \frac{1}{2} + \text{negl}(n)$$

if $\mathcal{A}$ can make a polynomial number $\ell(n)$ of decryption queries.

Cramer-Shoup: Security Proof

Proof.

The only value in CT^* which directly depends on m_b is e^*. If e^* is uniformly distributed from the point of view of $\mathcal{A}$, then $\mathcal{A}$ has no information about which message was encrypted.

What does $\mathcal{A}$ learn about z_1, z_2?

From the public key PK, $\mathcal{A}$ learns

$$\log_{g_1} h = z_1 + \gamma z_2. \tag{4}$$

□

Cramer-Shoup: Security Proof

From the decryption oracle on $CT = (u_1, u_2, e, v)$.

We distinguish between two cases, legal and illegal ciphertexts. In particular, CT is

- illegal if $\log_{g_1} u_1 \neq \log_{g_4} u_2$;
- legal otherwise.

We will prove that

1. $\mathcal{A}$ learns nothing from legal ciphertexts and from illegal ciphertext for which $\mathrm{Dec}(\mathrm{SK}, \cdot)$ returns $\perp$;
2. the probability that $\mathcal{D}$ *decrypts* illegal ciphertexts is negligible.

Cramer-Shoup: Security Proof

Assuming the validity of the above two points, consider an arbitrary $\mu \in \mathbb{G}$. We are interested in the probability that $\mu = g_3^{z_1} g_2^{z_2}$. In order for this to occur, we must have:

$$\log_{g_1} \mu = rz_1 + \gamma r' z_2 \tag{5}$$

Equations (4) and (5) form a system of linear equations in z_1 and z_2 (over $\mathbb{Z}_q$) with matrix of coefficients

$$B = \begin{pmatrix} 1 & \gamma \\ r & \gamma r' \end{pmatrix}$$

which is non singular since $r' \neq r$.

Cramer-Shoup: Security Proof

Each arbitrary $\mu \in \mathbb{G}$ is a possible value for $g_3^{z_1} g_4^{z_2}$, and each value is equally likely. Indeed, we have q possible values for z_1, z_2 from (4), and the map sending (z_1, z_2) in $g_3^{z_1} g_4^{z_2}$ is a bijection.

The adversary $\mathcal{A}$ cannot predict the value of $g_3^{z_1} g_4^{z_2}$ with probability better than $1/q$.

Since $g_3^{z_1} g_4^{z_2}$ is uniformly distributed in $\mathbb{G}$, also $g_3^{z_1} g_4^{z_2} m_b$ is uniformly distributed, thus $\mathcal{A}$ has no information about m_b.

Cramer-Shoup: Security Proof

Proof.

1. When $\log_{g_1} u_1 = \log_{g_2} u_2 = r''$, then $\mathcal{A}$ learns from m that

$$\log_{g_1} m = \log_{g_1} e - r'' z_1 - r'' \gamma z_2 \tag{6}$$

But equation (6) is linearly dependent with equation (4), so no extra information about z_1, z_2 in this case.

When $\mathrm{Dec}(\mathrm{SK}, \cdot)$ returns $\perp$, it means that

$$v \neq u_1^{x_1 + y_1 H(u_1, u_2, e)} u_2^{x_2 + y_2 H(u_1, u_2, e)}$$

But z_1, z_2 are not involved in this check, so no information about them also in this case.

□

Cramer-Shoup: Security Proof

Proof.

1. We consider two phases: before the challenge ciphertext is given, and after.

   Before the challenge ciphertext

   From the public key PK, $\mathcal{A}$ learns the following about x_1, x_2, y_1, y_2:

$$\log_{g_1} c = x_1 + \gamma x_2 \tag{7}$$

$$\log_{g_1} d = y_1 + \gamma y_2 \tag{8}$$

   So there are exactly q^2 possibilities for x_1, x_2, y_1, y_2. Each of them is equally likely from the point of view of $\mathcal{A}$.

□

Cramer-Shoup: Security Proof

Given an arbitrary $\mu \in \mathbb{G}$, we are interested in the probability that $\mu = u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$. In order for this to occur, we must have:

$$\log_{g_1} \mu = r(x_1 + \alpha y_1) + \gamma r'(x_2 + \alpha y_2) \tag{9}$$

Equations (7), (8) and (9) form a system of linear equations in x_1, x_2, y_1, y_2 (over $\mathbb{Z}_q$) with matrix of coefficients

$$C = \begin{pmatrix} 1 & \gamma & 0 & 0 \\ 0 & 0 & 1 & \gamma \\ r & \gamma r' & \alpha r & \alpha\gamma r' \end{pmatrix}$$

which has rank 3 since $r' \neq r$.

Cramer-Shoup: Security Proof

Each arbitrary $\mu \in \mathbb{G}$ is a possible value for $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$, and each value is equally likely. Indeed, we have q^2 possible values for x_1, x_2, y_1, y_2 from (7), (8), and the map sending (x_1, x_2, y_1, y_2) in $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$ is surjective, and the preimage of each $\mu \in \mathbb{G}$ contains q distinct elements.

Fixed u_1, u_2, e, the adversary $\mathcal{A}$ cannot predict the value of $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$ with probability better than $1/q$.

If the first illegal decryption query (u_1, u_2, e, v) is rejected, $\mathcal{A}$ learns that $v \neq u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$. This eliminates q of q^2 possibilities for (x_1, x_2, y_1, y_2). The $\ell(n)$-th decryption queries of this form will be rejected except with probability $1/(q^2 - (\ell(n) - 1)q)$. Thus the probability that one of these queries is not rejected is at most $\ell(n)/(q^2 - (\ell(n) - 1)q)$ (q is exponential in n, $\ell(n)$ is polynomial).

Cramer-Shoup: Security Proof

Proof.

After the challenge ciphertext

From the challenge ciphertext, $\mathcal{A}$ learns:

$$\log_{g_1} v^* = (x_1 + \alpha^* y_1)r + (x_2 + \alpha^* y_2)\gamma r' \tag{10}$$

Given the challenge ciphertext $CT^* = (u_1^*, u_2^*, e^*, v^*)$, we have three possible types of illegal queries (u_1, u_2, e, v):

- $(u_1, u_2, e) = (u_1^*, u_2^*, e^*)$ with $v \neq v^*$. Since the computed hash values are equal but $v \neq v^*$, the decryption oracle will reject it.
- $(u_1, u_2, e) \neq (u_1^*, u_2^*, e^*)$ and $\alpha = \alpha^*$. It means a collision in H has been found, but H is collision-resistant, so this happens only with negligible probability.

Cramer-Shoup: Security Proof

Proof.

- $(u_1, u_2, e) \neq (u_1^*, u_2^*, e^*)$ and $\alpha \neq \alpha^*$. The decryption oracle will accept the query only if

$$\log_{g_1} v = (x_1 + \alpha y_1)\tilde{r} + (x_2 + \alpha y_2)\gamma\tilde{r}' \tag{11}$$

where $\tilde{r} = \log_{g_1} u_1 \neq \tilde{r}' = \log_{g_2} u_2$.
BUT, in this case, the equations (7),(8),(10) and (11) having unknowns x_1, x_2, y_1, y_2 are linearly independent because

$$\det \begin{pmatrix} 1 & \gamma & 0 & 0 \\ 0 & 0 & 1 & \gamma \\ r & r'\gamma & r\alpha^* & r'\alpha^*\gamma \\ \tilde{r} & \tilde{r}'\gamma & \tilde{r}\alpha & \tilde{r}'\alpha\gamma \end{pmatrix} = (\gamma^2)(r' - r)(\tilde{r} - \tilde{r}')(\alpha - \alpha^*) \neq 0$$

Cramer-Shoup: Security Proof

Proof.

Each arbitrary $v \in \mathbb{G}$ is a possible value for $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$, and each value is equally likely. Indeed, we have q possible values for x_1, x_2, y_1, y_2 from (7),(8),(10), and the map sending (x_1, x_2, y_1, y_2) in $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$ is surjective, and hence a bijection.

Fixed u_1, u_2, e, the adversary $\mathcal{A}$ cannot predict the value of $u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$ with probability better than $1/q$.

If the first illegal decryption query (u_1, u_2, e, v) is rejected, $\mathcal{A}$ learns that $v \neq u_1^{x_1+\alpha y_1} u_2^{x_2+\alpha y_2}$. This eliminates 1 of q possibilities for (x_1, x_2, y_1, y_2). The $\ell(n)$-th decryption queries of this form will be rejected except with probability $1/(q - (\ell(n) - 1))$. The probability that one of these queries is not rejected is at most $\ell(n)/(q - (\ell(n) - 1))$ (q is exponential in n, $\ell(n)$ is polynomial).

Collision-Resistant Hash Functions based on Dlog

Theorem

If the discrete logarithm is hard, then collision-resistant hash functions exist.

We define a fixed-length hash function based on the discrete-logarithm assumption in prime order groups. It consists of the algorithms (KeyGen, H) as follows:

- $\text{KeyGen}(1^n)$: It runs $\mathcal{G}(n)$ to obtain a description of a cyclic group $\mathbb{G}$ of prime order q - with $\|q\| = n$ - and a generator g. It then selects a uniform $h \in \mathbb{G}$. It outputs the key $s = (\mathbb{G}, q, g, h)$.
- $H(s, (x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q)$: It outputs $H^s(x_1, x_2) := g^{x_1} h^{x_2} \in \mathbb{G}$.

Collision-Resistant Hash Functions based on Dlog

- Can you solve the Dlog problem if a collision in H^s is found?

Collision-Resistant Hash Functions based on Dlog

- Can you solve the Dlog problem if a collision in H^s is found?
- $H^s(x_1, x_2) = H^s(x_1', x_2')$, with $(x_1, x_2) \neq (x_1', x_2') \Longrightarrow g^{x_1} h^{x_2} = g^{x_1'} h^{x_2'} \Longrightarrow g^{x_1 - x_1'} = h;^{x_2' - x_2} \Longrightarrow \log_g h = [(x - x_1') \cdot (x_2' - x_2)^{-1} \mod q]$.
- Note that $x_2' - x_2 \neq 0 \mod q$, otherwise we will have $x_1 = x_1' \mod q$ and therefore no collision is found.
- As q is prime, the inverse of $(x_2' - x_2)$ exists.

Gap Diffie-Hellman Assumption

Definition

A group generation algorithm $\mathcal{G}$ is a gap-DH if the DDH problem in relative to $\mathcal{G}$ is easy but the CDH problem is still hard.

Further Reading (1)

Mihir Bellare, Alexandra Boldyreva, and Silvio Micali.
Public-key encryption in a multi-user setting: Security proofs and improvements.
In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 259–274. Springer Berlin Heidelberg, 2000.

Dan Boneh.
Simplified OAEP for the RSA and Rabin Functions.
In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 275–291. Springer Berlin Heidelberg, 2001.

Further Reading (2)


Ronald Cramer and Victor Shoup.
Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack.
SIAM Journal on Computing, 33(1):167–226, 2003.


Whitfield Diffie and Martin E Hellman.
New directions in cryptography.
Information Theory, IEEE Transactions on, 22(6):644–654, 1976.

Further Reading (3)

Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir.
New attacks on feistel structures with improved memory complexities.
In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 433–454, 2015.

Naofumi Homma, Atsushi Miyamoto, Takafumi Aoki, Akashi Satoh, and Adi Shamir.
Collision-based power analysis of modular exponentiation using chosen-message pairs.
In *Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings*, pages 15–29, 2008.