Privacy Enhancing Technologies (BSc)

Lecture V: Public Key Crypto and Security Reductions

Univ.-Prof. Dr. Dominique Schröder

November 11, 2024

Overview

KW	Type	Lectures
42	**Lecture**	Internet and Privacy in the Web
43	**Lecture**	Fingerprinting & Censorship
44	**Exercise**	Introduction to Rust & Challenge Start
45	**Lecture**	Cryptographic Preliminaries
46	**Lecture**	**Public Key Cryptography and Security Reductions**
47	**Exercise**	Proofs and Theoretical Deepening
48	**Lecture**	TLS
49	**Exam**	**Midterm Exam**
50	**Exercise**	Challenge Hands-On
51	**Lecture**	Messaging I
3	**Lecture**	Messaging II (Forward Security, Post Compromise Security)
4	**Lecture**	Onion
5	**Lecture**	Q & A
6	**Exam**	**Final Exam**
9	**Exam**	**Reexam** (Both for Midterm and Final)

This Lecture's Agenda

AES, ...

Basic build blocks

Crypto primitive

We introduce two encryption schemes and formally prove security.

- CPA-Secure Private Key Encryption Scheme ✓
- Decisional Diffie-Hellman Assumption (∈ Into to Crypto) $N = pq$ P.f. GP
- Diffie-Hellman Key Exchange
- ElGamal Encryption Scheme
- Security Reductions

CPA-Secure Encryption from PRFs

CPA-Secure Encryption from PRFs

Construction Idea

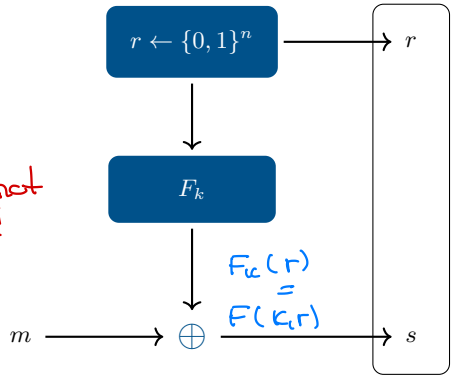
Intuition

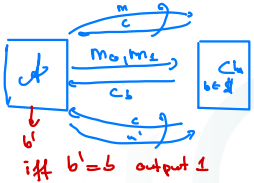
Enc(k, m)

$c \leftarrow f(k, m)$ // PRF

Dec(k, c)

$f^{-1}(k, c)$?? // might not exist!





$F(k, \cdot) \approx f(\cdot)$

PRF random func

Dec(k, c)

$c = (s, r)$

$m = F(k, r) \oplus s$

One-time pad:

$c = k \oplus m \rightsquigarrow (F(k,r) \oplus m, r) = (s, r)$

rnd $\oplus m$,

TU WIEN PRIVACY ENHANCING TECHNOLOGIES

CPA-Secure Encryption from PRFs

Formal Construction

Let F be a function. We define a private-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ for messages of length n as follows:

Construction

$\mathsf{Gen}(1^\lambda)$: Return a key $k \leftarrow \{0,1\}^\lambda$ of size λ chosen uniformly at random.

$\mathsf{Enc}(k, m)$: Choose $r \leftarrow \{0,1\}^\lambda$ uniformly at random and compute

$$c = (r, s) = (r, F(k, r) \oplus m).$$

$\mathsf{Dec}(k, c)$: Parse c as (r, s) and output the plaintext message computing

$$m = F(k, r) \oplus s.$$

CPA-Secure Encryption from PRFs

Formal Construction

Let F be a function. We define a private-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ for messages of length n as follows:

Construction

$\mathsf{Gen}(1^\lambda)$: Return a key $k \leftarrow \{0,1\}^\lambda$ of size λ chosen uniformly at random.

$\mathsf{Enc}(k, m)$: Choose $r \leftarrow \{0,1\}^\lambda$ uniformly at random and compute

$$c = (r, s) = (r, F(k, r) \oplus m).$$

$\mathsf{Dec}(k, c)$: Parse c as (r, s) and output the plaintext message computing

$$m = F(k, r) \oplus s.$$

CPA-Secure Encryption from PRFs

Formal Construction

Let F be a function. We define a private-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ for messages of length n as follows:

Construction

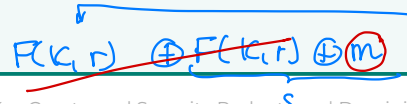
$\mathsf{Gen}(1^\lambda)$: Return a key $k \leftarrow \{0,1\}^\lambda$ of size λ chosen uniformly at random.

$\mathsf{Enc}(k, m)$: Choose $r \leftarrow \{0,1\}^\lambda$ uniformly at random and compute

$$c = (r, s) = (r, F(k, r) \oplus m).$$

$\mathsf{Dec}(k, c)$: Parse c as (r, s) and output the plaintext message computing

$$m = F(k, r) \oplus s$$



Proof – CPA-Secure Encryption from PRFs

Theorem

If F is a pseudorandom function, then the previous construction is a CPA-secure private-key encryption scheme for messages of length λ.

How do we show security?

Reduction Proofs

Complexity Theory
Crypto

Reduction Proofs

Say we want to prove the following theorem with a proof by reduction.

Theorem

Let H be an known primitive. From that we construct a new primitive G, which uses H in some way and has some specific properties.

This is secure! (PRF) → CPA encryption

What do we want to achieve?

We assume that there is an efficient adversary $\mathcal{A}$ against the new primitive G. From that we construct an efficient adversary $\mathcal{B}$ against the known primitive H, which uses $\mathcal{A}$ to break the primitive H. Because we know, that the primitive H has to hold, this is a contradiction. We can conclude, that an adversary $\mathcal{A}$ against the new primitive G can't exist, so this new primitive G has to fulfill the required properties.

Reduction Proofs

Say we want to prove the following theorem with a proof by reduction.

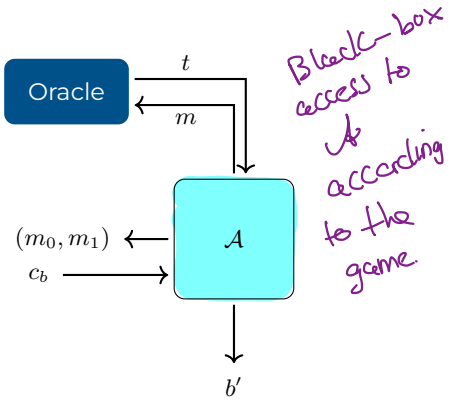
Theorem

Let H be an known primitive. From that we construct a new primitive G, which uses H in some way and has some specific properties.

What do we want to achieve?

We assume that there is an efficient adversary $\mathcal{A}$ against the new primitive G. From that we construct an efficient adversary $\mathcal{B}$ against the known primitive H, which uses $\mathcal{A}$ to break the primitive H. Because we know, that the primitive H has to hold, this is a contradiction. We can conclude, that an adversary $\mathcal{A}$ against the new primitive G can't exist, so this new primitive G has to fulfill the required properties.

Reduction Proofs – $\mathcal{A}$'s World

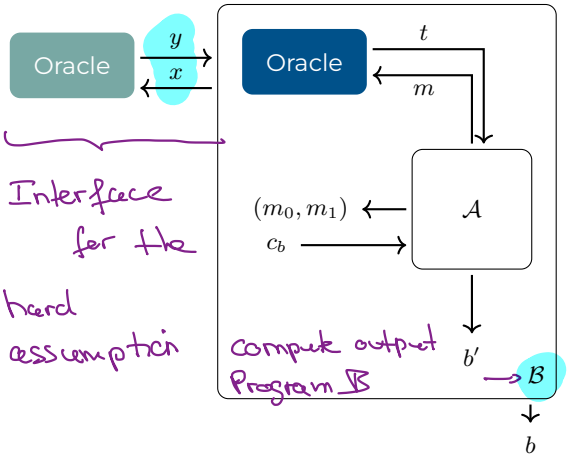


An overview of how the reduction works.

How to construct the reduction?

- We start with the *security game* of the known primitive H recalling the expected interfaces for adversary $\mathcal{A}$.
- These interfaces *may* include decryption/encryption oracles, or those outputting a PRF/PRG or their random equivalents.
- The adversary *may* interact directly with the challenger, with in/outgoing messages.
- The figure beside illustrates this setup for the CPA game discussed before.

Reduction Proofs – $\mathcal{B}$ Simulates the Game for $\mathcal{A}$

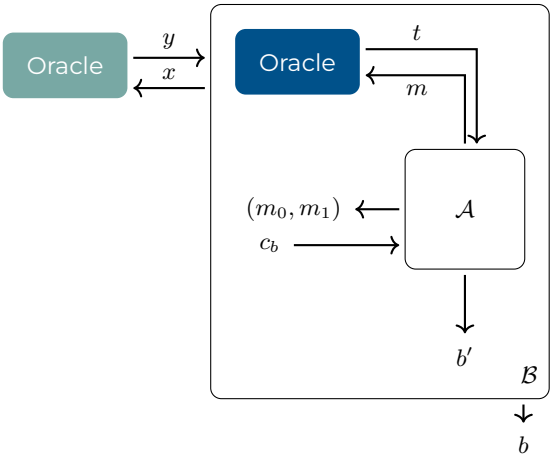


An overview of how the reduction works.

How to construct the reduction?

- $\mathcal{B}$'s goal is to solve the underlying hard problem.
- $\mathcal{B}$ *might* get an instance of this problem as input and *possibly* access to oracles.
- $\mathcal{B}$ simulates *all* interfaces for $\mathcal{A}$ such that $\mathcal{A}$ does not realize that is executed within another algorithm.
- The figure beside illustrates this setup, showing necessary oracles and I/O between $\mathcal{A}$ and $\mathcal{B}$ for the security reduction to a PRF.

Reduction Proofs



An overview of how the reduction works.

What now?

- We need to show, how $\mathcal{B}$ can break the secure primitive (or assumption) H, whenever there exists an adversary $\mathcal{A}$, that breaks G (that is the hard part).
- If we do so, no efficient adversary $\mathcal{A}$ against G can exist, since otherwise the secure primitive (or assumption) H can also be broken.
- We conclude, that G is secure.

Reduction Proofs

Warning

- Security reductions are highly exam-relevant.
- We will practice security reductions in the upcoming exercise sessions.
- In addition, you must practice on your own and ask us if you get lost.
- You will not pass this course if you do not understand security reductions.

We expect that you read about security reductions in the book of Katz and Lindell.

Reduction Proofs

An example (2)

$F(k,x)||0$ — insecure

Underlying assumption (H) → "secure"

Theorem

Let $F : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^n$ a pseudorandom function. Then the function $F'(k,x) := F(k,x||0)$ is a pseudorandom function as well.

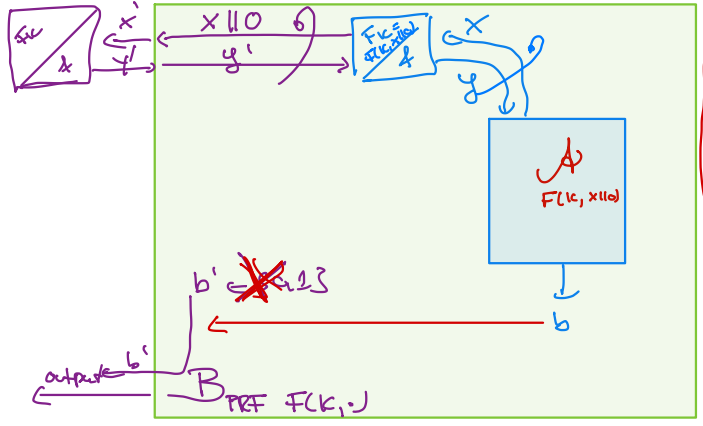
→ new construction G.

→ modification!

We want to prove this theorem with a proof by reduction.

Reduction Proofs

An example (2)



Construction

Contradiction: $F(k, x\|0)$ is not a PRF.

$$|Pr[A^{F'}=1] - Pr[A^{f}=1]| \not\approx 0$$

$$|Pr[D^{F(k,\cdot)}=1] - Pr[D^{f}=1]| \approx 0$$

Analysis

$$B\,|Pr[B^{F}=1] - |Pr[B^{f}=1]|$$

$$|\underbrace{Pr[A^{F'}=1]}_{F'} - |Pr[A^{f}=1]|$$

$\Rightarrow$ ↯

$\Rightarrow$ contradiction to F is a PRF. ☺

① Contradiction $F'(k,x) := F(k, x\|0)$ is not a PRF.

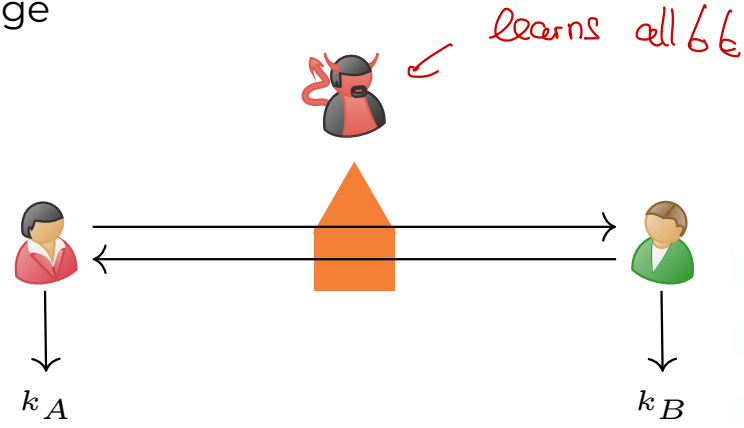
Def: PRF: <u>secure:</u> $|Pr[D^F = 1] - Pr[D^f = 1]| \approx 0$

insecure $\Rightarrow$ $\exists$ ppt D, s.t.

$$|Pr[D^F = 1] - Pr[D^f = 1]| \not\approx 0$$

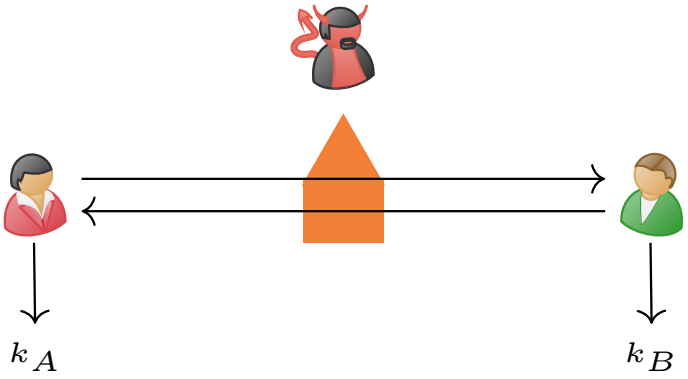
Key Exchange

Key Exchange



- Correctness: $k_A = k_B = k$
- Security (intuitive): Π is secure if k is completely unknown to the adversary.

Key Exchange



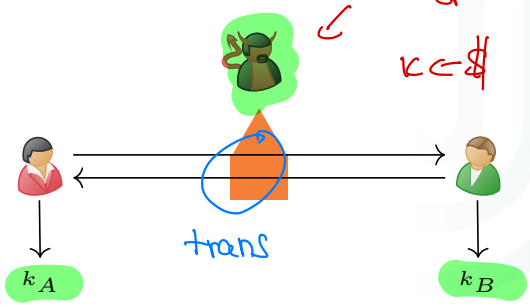
- Correctness: $k_A = k_B = k$
- Security (intuitive): Π is secure if k is completely unknown to the adversary.

Key-Exchange Experiment

We define an experiment, where Π is a key-exchange protocol.

$\mathsf{KE}^{eav}_{\mathcal{A},\Pi}(\lambda)$

1 : $(trans, k) \leftarrow < A(1^\lambda), B(1^\lambda) >$
2 : $b \leftarrow \{0,1\}$
3 : **if** $b = 0 : k' \leftarrow k$
4 : **if** $b = 1 : k' \leftarrow \{0,1\}^\lambda$
5 : $b' \leftarrow \mathcal{A}(k')$
6 : **if** $b' = b$ **return** 1
7 : **else return** 0



We write $\mathsf{KE}^{eav}_{\mathcal{A},\Pi}(\lambda) = 1$ if the output is 1 and we say that $\mathcal{A}$ succeeded.

Eavesdropper Security

Definition (Eavesdropper Security of Key-Exchange Protocols)

A key-exchange protocol Π is secure in the presence of an eavesdropper if for every PPT adversary $\mathcal{A}$, there exists a negligible function negl such that

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

$\Rightarrow$ PRF style is possible.

Diffie-Hellman Key-Exchange Protocol

$(\mathbb{G}, q, g) \leftarrow \mathsf{Gen}(1^\lambda)$

$x \leftarrow \mathbb{Z}_q$

$h_A \leftarrow g^x$

$(\mathbb{G}, q, g, h_A)$

h_B

$y \leftarrow \mathbb{Z}_q$

$h_B \leftarrow g^y$

$k_A \leftarrow h_B^x$

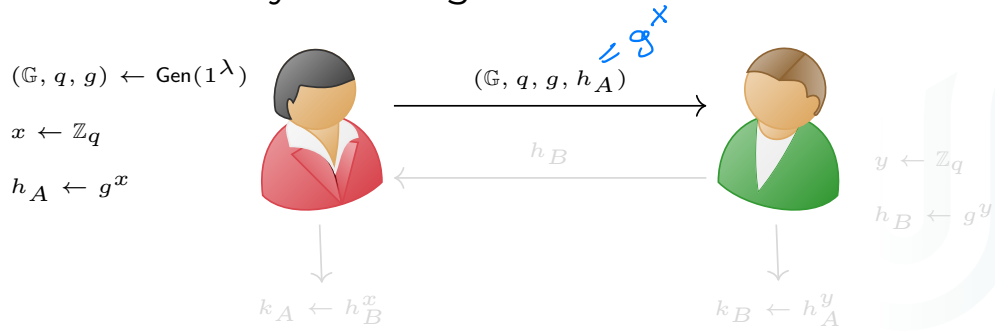
$k_B \leftarrow h_A^y$

Correctness:

$$k_A = h_B^x = (g^y)^x = g^{x \cdot y}$$

$$k_B = h_A^y = (g^x)^y = g^{x \cdot y}$$

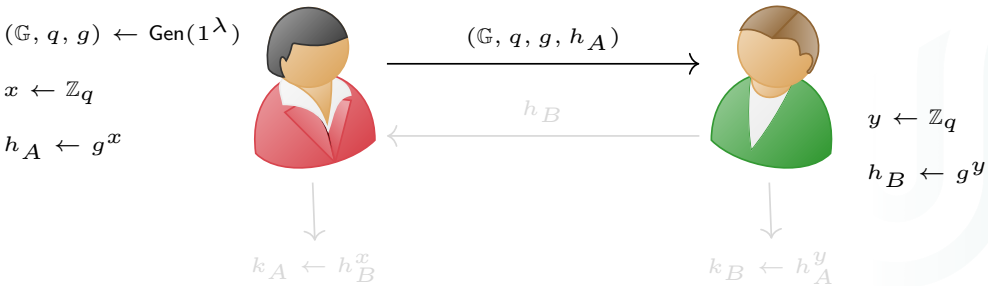
Diffie-Hellman Key-Exchange Protocol



Correctness:

$$k_A = h_B^x = (g^y)^x = g^{x \cdot y}$$
$$k_B = h_A^y = (g^x)^y = g^{x \cdot y}$$

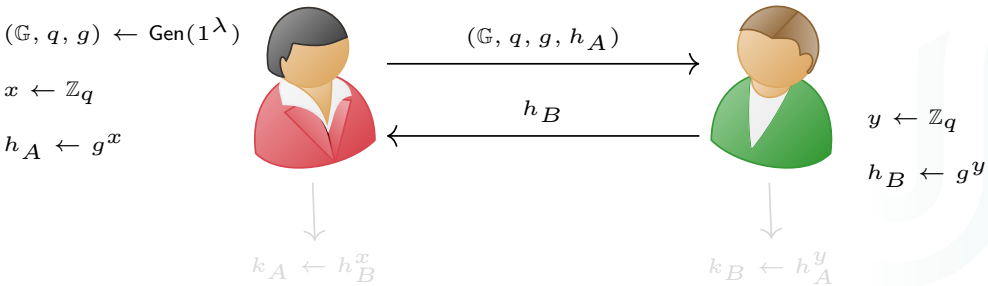
Diffie-Hellman Key-Exchange Protocol



Correctness:

$$k_A = h_B^x = (g^y)^x = g^{x \cdot y}$$
$$k_B = h_A^y = (g^x)^y = g^{x \cdot y}$$

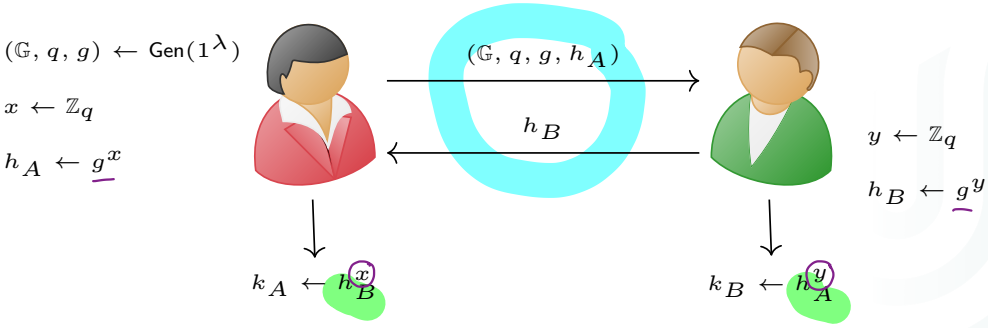
Diffie-Hellman Key-Exchange Protocol



Correctness:

$$k_A = h_B^x = (g^y)^x = g^{x \cdot y}$$
$$k_B = h_A^y = (g^x)^y = g^{x \cdot y}$$

Diffie-Hellman Key-Exchange Protocol



Correctness:

$$k_A = h_B^x = (g^y)^x = g^{x \cdot y}$$
$$k_B = h_A^y = (g^x)^y = g^{x \cdot y}$$

Security of Diffie-Hellman Key-Exchange

Let $\mathsf{KE}'^{eav}_{\mathcal{A},\Pi}(\lambda)$ denote a modified experiment where if $b = 1$ the adversary is given k' chosen uniformly from G instead of a uniform n-bit string.

Theorem

If the decisional Diffie-Hellman problem is hard relative to Gen, then the Diffie-Hellman key-exchange protocol Π is secure in the presence of an eavesdropper (w.r.t. the modified experiment $\mathsf{KE}'^{eav}_{\mathcal{A},\Pi}(\lambda)$).

Cryptographic Hardness Assumptions

Decisional Diffie-Hellman Assumption

Definition (Decisional Diffie-Hellman)

The Decisional Diffie-Hellman (DDH) problem is hard relative to Gen if for all PPT algorithms $\mathcal{A}$ there exists a negligible function negl such that

$$\Pr[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - \Pr[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1] \leq \mathsf{negl}(\lambda),$$

where in each case the probabilities are taken over the experiment in which the group generation algorithm $\mathsf{Gen}(1^\lambda)$ outputs $(\mathbb{G}, q, g)$, and then uniform $x, y, z \in \mathbb{Z}_q$ are chosen.

Discrete Logarithm Assumption

We define an experiment, where Gen is a group generation algorithm.

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

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

2 : $h \leftarrow\!\!\$\ \mathbb{G}$

3 : $x \leftarrow \mathcal{A}(\mathbb{G}, q, g, h)$

4 : **return** $g^x \stackrel{?}{=} h$

We write $\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(\lambda) = 1$ if the output is 1 and we say that $\mathcal{A}$ succeeded.

Discrete Logarithm Assumption

Definition (Discrete Logarithm)

The discrete logarithm problem is hard relative to Gen if for all PPT algorithms $\mathcal{A}$ there exists a negligible function negl such that

$$\Pr\left[\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda).$$

Examples

Where Do These Assumptions Hold?

- **Cyclic groups of prime order**
  - DLog problem the hardest in such groups (Pohlig-Hellman algorithm).
  - DDH problem is easy if the group order q has small prime factors.
  - Finding generators is easy: Every element is a generator.
- **(Subgroups of)** $\mathbb{Z}_p^*$
  - Problem: The DDH assumption is *not hard* in general.
  - Solution: Work in subgroups of $\mathbb{Z}_p^*$
    Let $p = rq + 1$, where both p, q are primes. Then $\mathbb{Z}_p^*$ has a subgroup of order q.

Intro to Crypto

$q \in$ check prime $\cup$

$p = r \cdot q + 1$ check prime

Public-Key Encryption

Public-Key Encryption

A Formal Definition

Definition (Public-Key Encryption)

A public-key encryption scheme consists of three PPT algorithms:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda)$: The key generation algorithm outputs a pair $(\mathsf{sk}, \mathsf{pk})$. We refer to the first of these as the private key and the second as the public key.

$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$: The encryption algorithm takes as input a public-key pk and a plaintext message m and outputs a ciphertext $c \in C$.

$m \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$: The decryption algorithm takes as input a private-key sk and a ciphertext c and outputs some plaintext m. We assume w.l.o.g. that Dec is deterministic.

Public-Key Encryption

Correctness

An encryption scheme must satisfy the following **(perfect) correctness** requirement:

Definition (Correctness of Public-Key Encryption Schemes)

For every security parameter $\lambda \in \mathbb{N}$, every key-pair $(\mathsf{sk}, \mathsf{pk})$ output by $\mathsf{KGen}(1^\lambda)$ and every message $m \in \{0,1\}^*$, it holds that

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m.$$

CPA Security For Public-Key Encryption Schemes

An Experiment

We define an experiment, where $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is a public-key encryption scheme.

$\mathsf{PubK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(\lambda)$

1 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$

2 : $(m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pk}), \quad |m_0| = |m_1|$

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

4 : $c_b \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$

5 : $b' \leftarrow \mathcal{A}(c_b)$

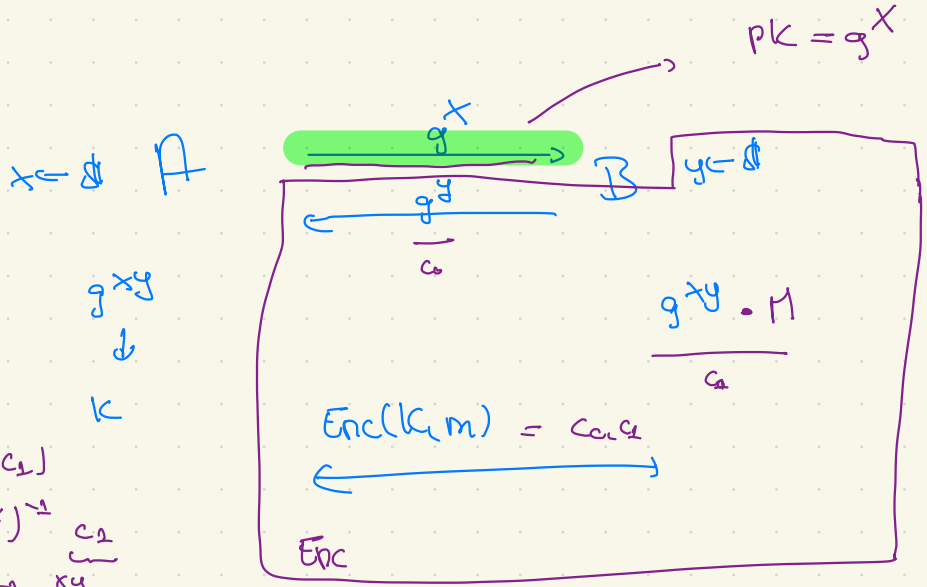
6 : **return** $b' \stackrel{?}{=} b$

$\mathsf{Enc}(\mathsf{pk}, \cdot)$?

No oracle blc Enc(pk, ·) is public
⇒ A can do it locally.

We write $\mathsf{PubK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(\lambda) = 1$ if the output is 1 and we say that $\mathcal{A}$ succeeded.

El Gamal Encryption


pk = g^x
x ← $
A
g^x
B
y ← $
g^y
c0
g^xy
k
g^xy · M
c0
Enc(k, m) = c0, c1
Enc
(c0, c1)
(c0^x)^-1
c1
=
g^-xy g^xy · m

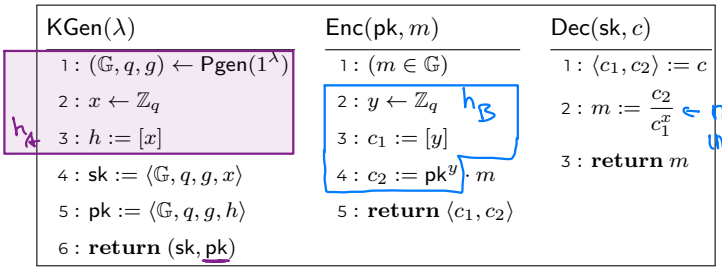
Notation

If we talk about group elements $g^x \in \mathbb{G}$, where the generator g is known, we use the notation $g^x = [x]$.
By $x \leftarrow\!\$\ X$, we denote the uniform sampling of x from the set X, and λ represents the security parameter. By $y \leftarrow \mathsf{A}(x; r)$ we denote a probabilistic polynomial time (PPT) algorithm A that on input x and randomness r, outputs y. When A is a deterministic polynomial time (DPT) algorithm, we use the notation $x := \mathsf{A}(y)$. We call a function $\mathsf{negl}(\lambda) : \mathbb{N} \to \mathbb{R}$ negligible in λ if for every $k \in \mathbb{N}$, there exists $\lambda_0 \in \mathbb{N}$ s.t. for every $\lambda \geq \lambda_0$ it holds that $|\mathsf{negl}(\lambda)| \leq \lambda^{-k}$.

El Gamal Encryption Scheme

Construction

We define the El Gamal encryption scheme as follows:



Security of El Gamal Encryption

Theorem

If the DDH problem is hard relative to $\mathbb{G}$, then the El Gamal encryption scheme is CPA secure.

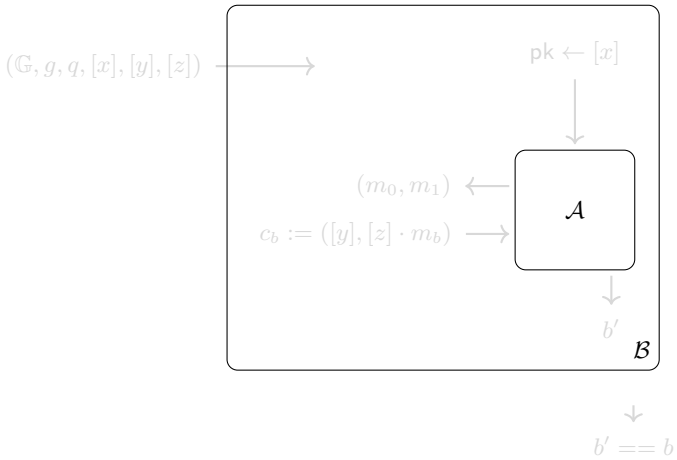
Lemma

Let $\mathbb{G}$ be a finite group, and let $m \in \mathbb{G}$ be arbitrary. Then choosing uniform $k \leftarrow \mathbb{G}$ and setting $k' := k \cdot m$ gives the same distribution for k' as choosing uniform $k' \leftarrow \mathbb{G}$. Put differently, for any $g'' \in \mathbb{G}$ we have

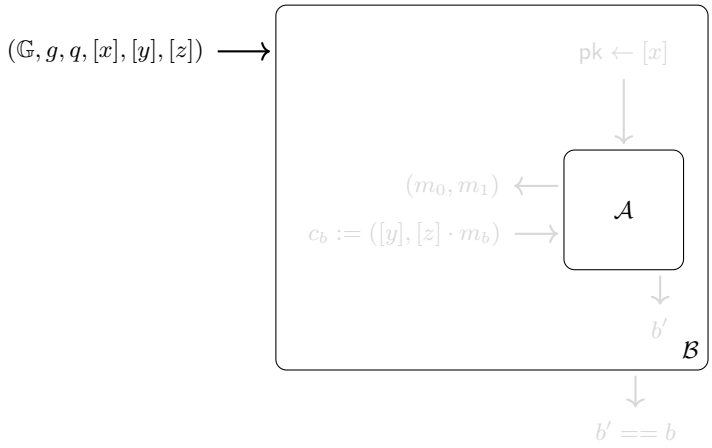
$$\Pr[k \cdot m = g''] = \frac{1}{|\mathbb{G}|},$$

where the probability is taken over a uniform choice of k.

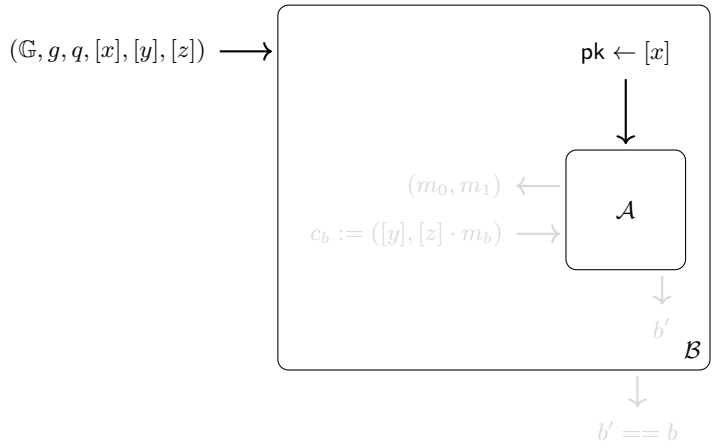
Proof of El Gamal



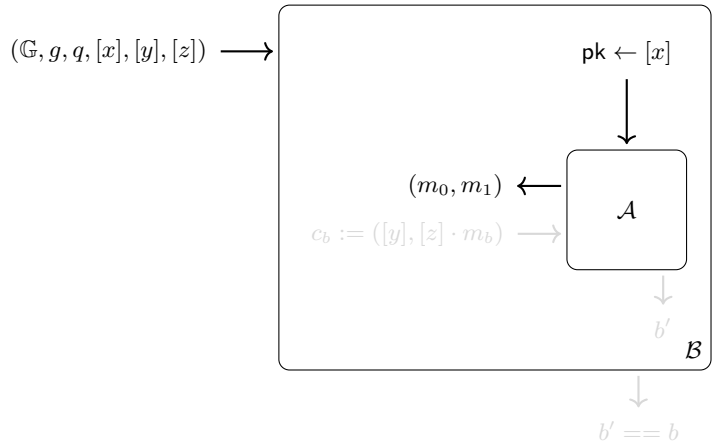
Proof of El Gamal



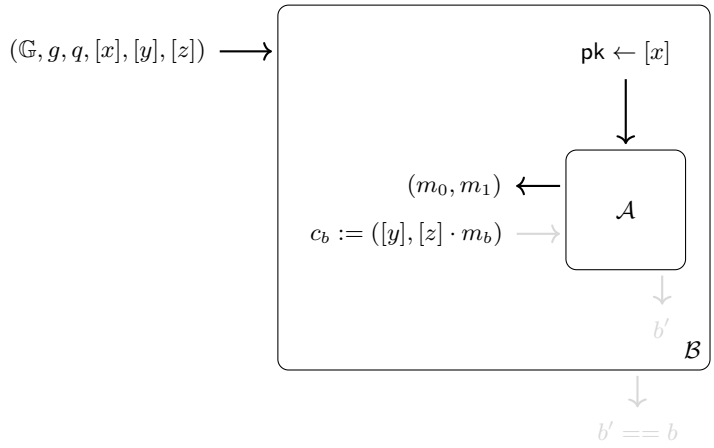
Proof of El Gamal



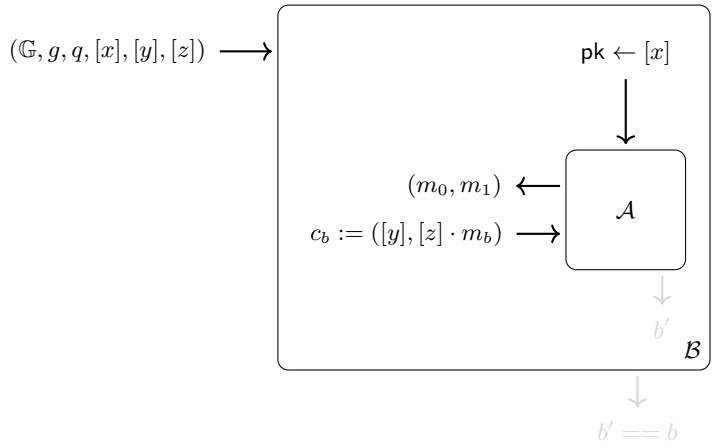
Proof of El Gamal



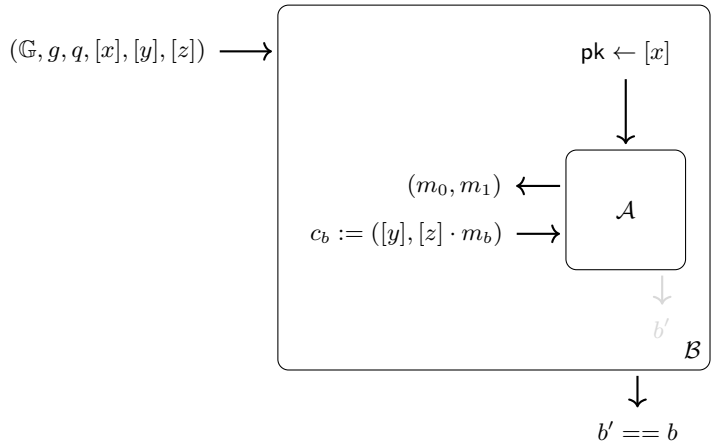
Proof of El Gamal



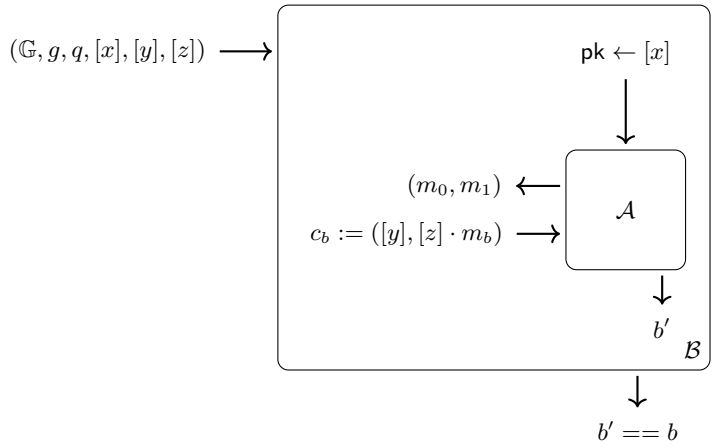
Proof of El Gamal



Proof of El Gamal



Proof of El Gamal



Feedback

Your *anonymous feedback* is invaluable to us, and we are truly grateful for your time and insights.

Thank you for helping us improve!

https://tuwel.tuwien.ac.at/mod/feedback/view.php?id=2456987

PRIVACY
ENHANCING
TECHNOLOGIES