Modern Cryptography
explained via Two-View Principle

Phan Duong Hieu

Telecom Paris, IPP
hieu.phan@telecom-paris.fr
https://www.di.ens.fr/~phan/

New Technologies and the Risks for Privacy

Privacy

- *Privacy:* “the right to be left alone”
- *Privacy protection* allows individuals to have control over how their personal information is collected and used

Big Data, Cloud computing

- Easy to collect and store user data
- Combined with powerful tools (e.g., machine learning)

$\rightarrow$ Attractive applications but Huge risk of mass surveillance, social credit systems.

Cryptography

Security of Data

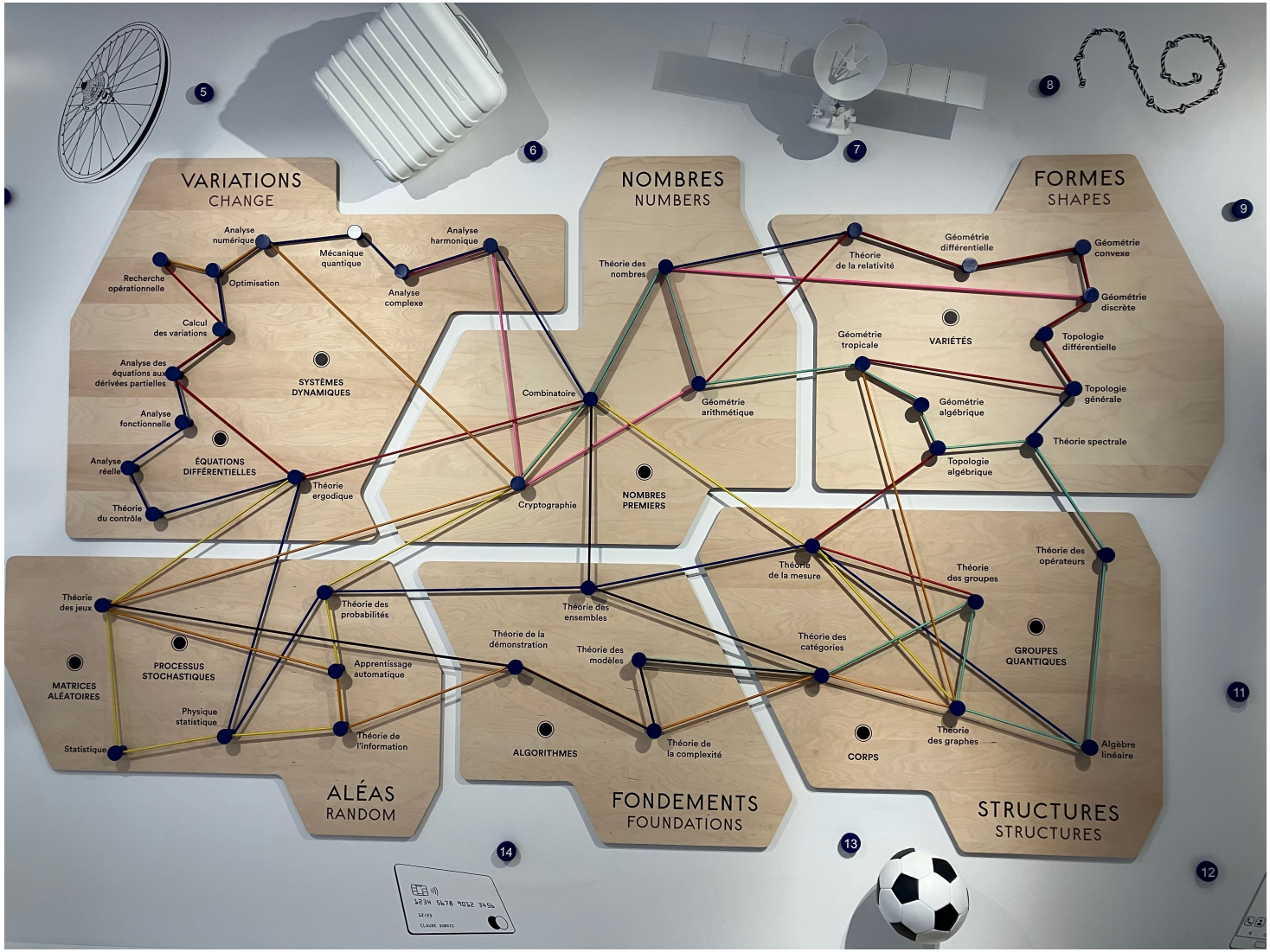
- Integrity with hash function
- Confidentiality with encryption
- Authenticity with MAC, signature

New Technologies $\rightarrow$ Advanced cryptographic primitives

- Big Data, Cloud Computing $\rightarrow$ widespread real-life applications
- Privacy: protect personal information.
  - Security
  - Trust on Authorities

$\rightarrow$ **Security of Computation on Untrusted Machines.**

Cryptography in Museum of Mathematics



Some directions for the future:

- How to protect privacy in the AI era
- How to protect privacy against powerful adversaries (e.g., anamorphic encryption)
- How to implement the "Right to be Forgotten"
- How to use powerful tools (e.g., quantum machines) to protect data and privacy (e.g., key leasing)

Some directions for the future:

- How to protect privacy in the AI era
- How to protect privacy against powerful adversaries (e.g., anamorphic encryption)
- How to implement the "Right to be Forgotten"
- How to use powerful tools (e.g., quantum machines) to protect data and privacy (e.g., key leasing)

This course

- How new concepts were invented and their impact
- How security can be proven

Modern Cryptography from the **Two-View Principle**

Modern Cryptography from the **Two-View Principle**

Secret Communication: Sender vs. Receiver Views

- **Symmetric Encryption:** The same key is used for both encryption (locking) and decryption (unlocking).

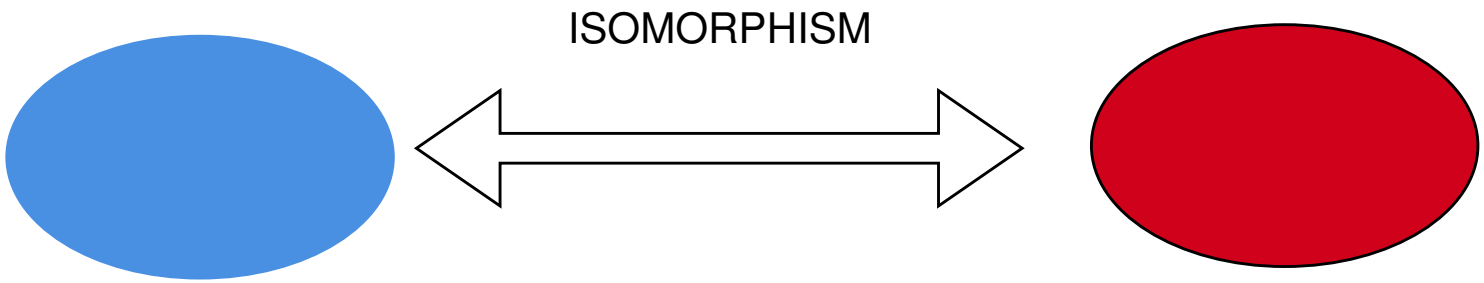
Modern Cryptography from the **Two-View Principle**

Secret Communication: Sender vs. Receiver Views

- **Symmetric Encryption:** The same key is used for both encryption (locking) and decryption (unlocking).
- **Asymmetric Encryption:** Different keys are used for encryption and decryption $\rightarrow$ the public key is used for encryption, and the private key for decryption.

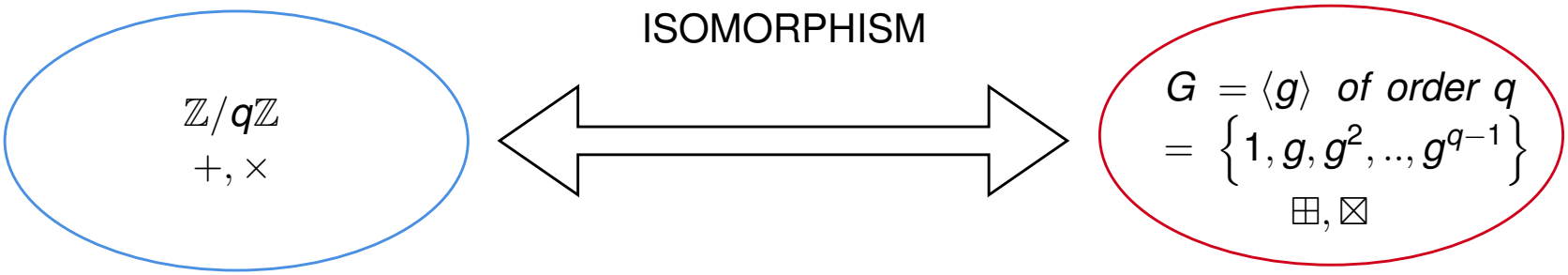
Outline

Mathematical vs. Computational Views



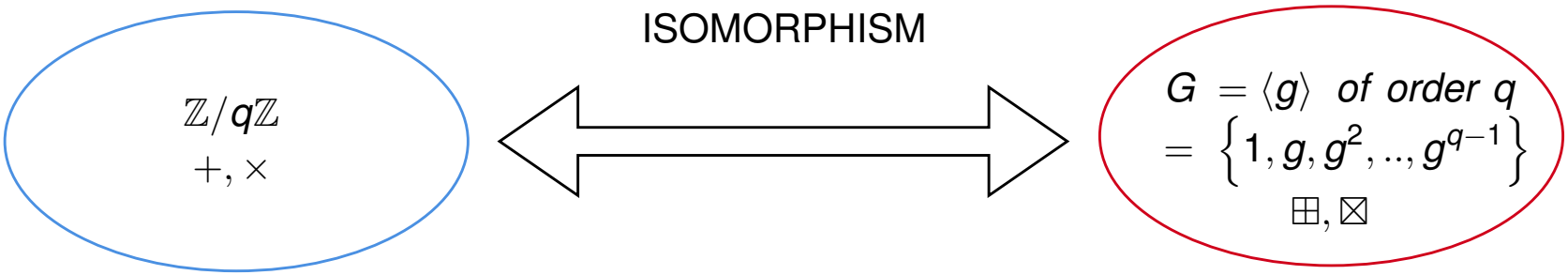
Mathematical Views: They are the same

Mathematical vs. Computational Views



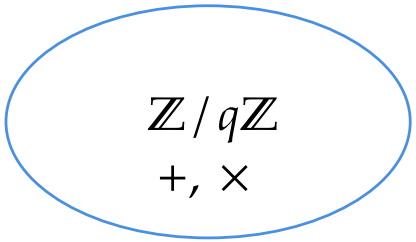
$$g^x \boxplus g^y := g^{x+y}$$
$$g^x \boxtimes g^y := g^{xy}$$

Mathematical vs. Computational Views



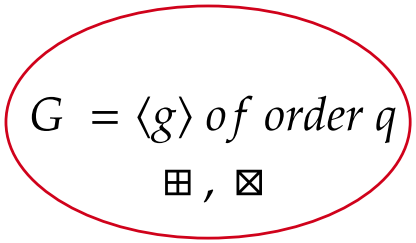
$$\begin{aligned} \varphi : \mathbb{Z}/q\mathbb{Z} &\to G = \langle g \rangle \text{ of order } q \\ x &\mapsto g^x \\ x + y &\mapsto g^x \boxplus g^y := g^{x+y} \\ x \times y &\mapsto g^x \boxtimes g^y := g^{xy} \end{aligned}$$

Mathematical vs. Computational Views



$\varphi : x \mapsto g^x$ *is easy*

$\varphi^{-1} : g^x \mapsto x$ *is hard*
(*Discrete Log problem*)

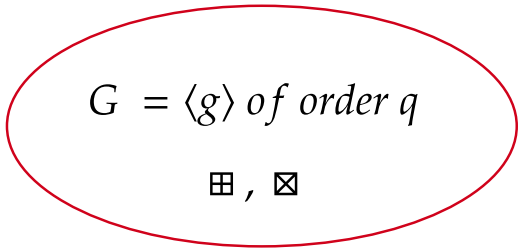


Given $X = g^x$; $Y = g^y$
$X \boxtimes Y := g^{xy}$ *is hard*
(*Computational Diffie – Hellman*)

with x : $X \boxtimes Y$ *is easy*
x : *local*, *does not depend on* y

Choice of G *in practice* :
a subgroup of a finite field, or of the group of points on an elliptic curve

Insiders' vs. Outsiders' Views



$\left(X = g^x;\ Y = g^y\right) : X \boxtimes Y := g^{xy}$ *is hard to compute*

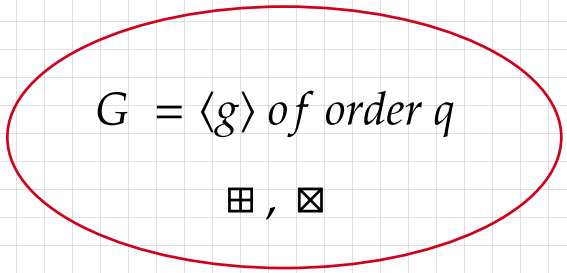
But with x *or* y : $X \boxtimes Y = Y \boxtimes X \left(= X^y = Y^x\right)$ *are easy to compute*

$\Longrightarrow$ *Set up a commun key* $K = X \boxtimes Y$

Insiders (generating in local x *or* y*)*
$\rightarrow$ *mathematical view, able to compute all operations*
Outsiders (adversaries, only see public X, Y*)*
$\rightarrow$ *computational view, unable to compute* $\boxtimes$

$\Longrightarrow$ **Revolution in Cryptography :**
Secure communication is possible without directly sharing any secret

Diffie-Hellman Key Exchange '76 to ElGamal Encryption '85



$\left(X = g^x;\ Y = g^y\right) : X \boxtimes Y := g^{xy}$ *is hard to compute*

But with x *or* y : $X \boxtimes Y = Y \boxtimes X \left(= X^y = Y^x\right)$ *are easy to compute*

ElGamal Encryption :
Secret key : x
Public key : $X = g^x$
Encryption : randomly generates y, *set* $Y = g^y$ *and send*

$$\left(Y,\ (X \boxtimes Y) \times M\right) = \left(g^y, g^{xy} \times M\right)$$

Decryption : from x, Y, *compute* $(X \boxtimes Y)$ *and recover* M

Insiders (encryptor, decryptor) → *mathematical view, able compute all operations*
Outsiders (adversaries) → *computational view, unable to compute* $\boxtimes$

Modern Cryptography

Public-key Encryption (Diffie-Helmann 1976)

- Encryption key could be published $\rightarrow$ encryption can be publicly computed.

Modern Cryptography

Public-key Encryption (Diffie-Helmann 1976)

- Encryption key could be published $\rightarrow$ encryption can be publicly computed.
- **Elgamal scheme**

$$\frac{m(g^d)^r}{(g^r)^d} = m, \text{ where } g \text{ is a generator of a prime-order cyclic group}$$

- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

Digital Signature

- **RSA Signature** $\text{Sign}(m) := m^d \mod N$

Security
from the **Two-View Principle**

Security
from the **Two-View Principle**

Communication: Insider vs. Outsider Views

- Security is often established by showing that :
  - “insider” adversaries have no advantage w.r.t an outsider adversary.
  - communication generated by an **insider** (who knows the secret) can be **simulated** by an **outsider** (who does not know the secret).

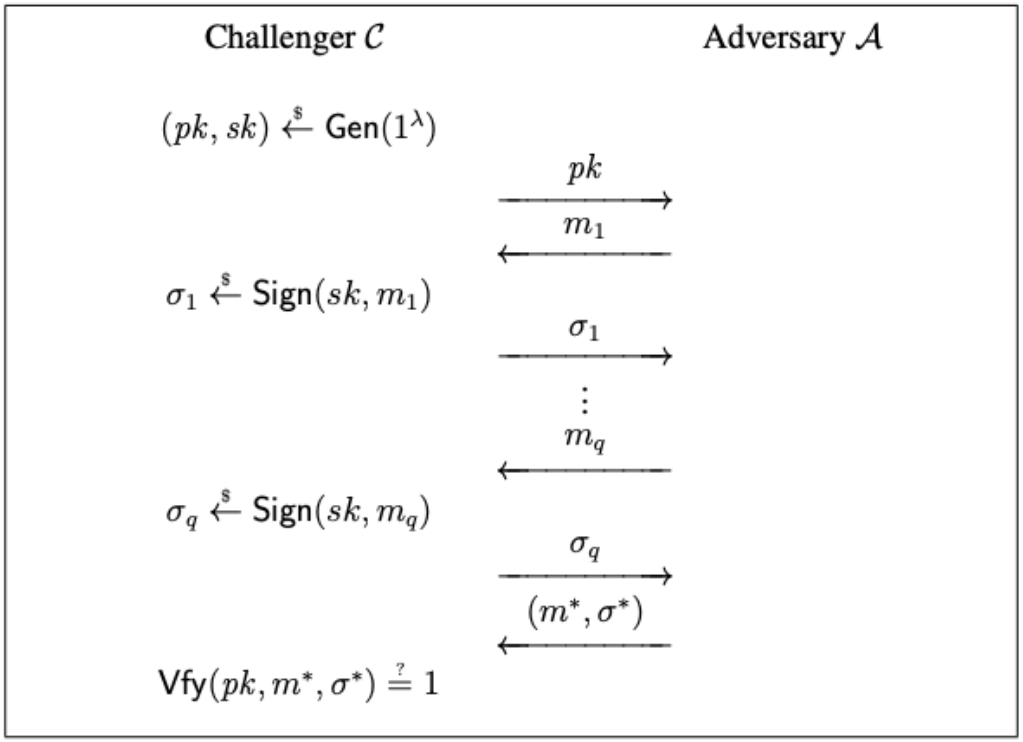
Security
from the **Two-View Principle**

Communication: Insider vs. Outsider Views

- Security is often established by showing that :
  - "insider" adversaries have no advantage w.r.t an outsider adversary.
  - communication generated by an **insider** (who knows the secret) can be **simulated** by an **outsider** (who does not know the secret).

$\rightarrow$ This implies that the communication does not leak the secret.

Digital Signatures: attack model (EUF-CMA)



Existential unforgeability under adaptive chosen message attacks

$$Adv(\mathcal{A}) = \Pr[Vfy(pk, m^\star, \sigma^\star) = 1]$$

The scheme is EUF-CMA secure if $\forall \mathcal{A}$, $Adv(\mathcal{A})$ is negligible.

Lamport's One-time Signatures from OWF f

- $Gen(1^\lambda) \rightarrow (pk, sk)$:

$$sk = \begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{\ell,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{\ell,1} \end{pmatrix}$$

$$pk = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{\ell,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{\ell,1} \end{pmatrix}$$

  where $x_{i,b} \in \{0,1\}^n, y_{i,b} = f(x_{i,b})$

- $Sign(sk, m = m_1 m_2 \ldots m_\ell \in \{0,1\}^\ell) \rightarrow \sigma$

$$\sigma = x_{1,m_1} x_{2,m_2} \ldots x_{\ell,m_\ell}$$

- $Vfy(pk, m, \sigma)$ check if $y_{i,m_i} = f(\sigma_i = x_{i,m_i}), \forall i = 1 \ldots \ell$

Theorem

If f is one-way, then the one-time signature is EUF-CMA.

Public-Key Encryption

What we discussed

If factorization or DL problems are easy, then we can attack crypto systems (RSA, ElGamal) that based on these problems

Question

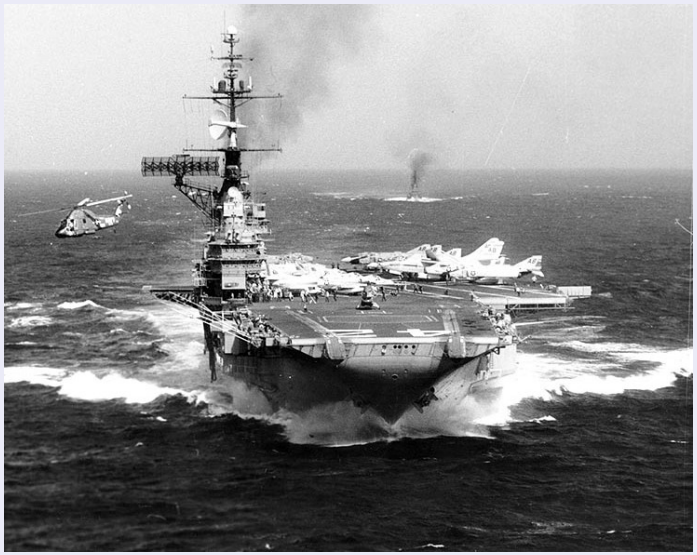
Suppose that factorization and DL problems are hard. Could we prove the security for proposed crypto systems?

One-wayness is enough?

$$E'(m1||m_2) := E(m_1)||m_2$$

- If E is one-way, then E' is also one-way
- But the security of E' is clearly not enough: at least half the message leaks!

In many situation, one bit (attack or not) is important...



Semantic security [Goldwasser-Micali '82]

Perfect Security vs. Semantic security

- Perfect security: the distribution of the ciphertext is perfectly independent of the plaintext
- Semantic security (computational version of perfect security): the distribution of the ciphertext is computationally independent of the plaintext

Semantic security [Goldwasser-Micali '82]

Perfect Security vs. Semantic security

- Perfect security: the distribution of the ciphertext is perfectly independent of the plaintext
- Semantic security (computational version of perfect security): the distribution of the ciphertext is computationally independent of the plaintext

Semantic Security - IND

- Semantic Security is equivalent to the notion of Indistinguishability (IND): No adversary (modeled by a poly-time Turing machine) can distinguish a ciphertext of m_0 from a ciphertext of m_1.

IND Security Notion

- **IND:**
  - **Security Game:**
    1. The adversary $\mathcal{A}$ receive pk and chooses two plaintexts m_0 and m_1.
    2. A random bit $b \in \{0, 1\}$ is selected, and the challenger encrypts m_b to get the ciphertext $c = \mathsf{Enc}(pk, m_b)$.
    3. The ciphertext c is given to $\mathcal{A}$.
    4. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$.
  - **Advantage:**
$$\mathsf{Adv}_{\mathcal{A}}^{\text{IND-CPA}} = \left| \Pr[b' = b] - \frac{1}{2} \right|$$
  - **IND-CPA Security:**
$$\forall \text{polynomial-time } \mathcal{A}, \mathsf{Adv}_{\mathcal{A}}^{\text{IND-CPA}} \text{ is negligible}$$

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

- **ElGamal scheme**

$$\frac{m(g^d)^r}{(g^r)^d} = m, \text{ where } g \text{ is a generator of a cyclic group}$$

Exercises

- Is RSA IND? Is ElGamal IND?

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

- **ElGamal scheme**

$$\frac{m(g^d)^r}{(g^r)^d} = m, \text{ where } g \text{ is a generator of a cyclic group}$$

Exercises

- Is RSA IND? Is ElGamal IND?
- For public-key encryption: Probabilistic encryption is required!
- For secret-key encryption: deterministic encryption could be semantically secure [Phan-Pointcheval '04]

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

- **ElGamal scheme**

$$\frac{m(g^d)^r}{(g^r)^d} = m, \text{ where } g \text{ is a generator of a cyclic group}$$

Exercises

- Is RSA IND? Is ElGamal IND?

Security of RSA & ElGamal PKE

Recall:

- k_e could be published $\rightarrow$ encryption can be publicly computed.
- **RSA scheme**

$$(m^e)^{(e^{-1} \mod \phi(N))} = m \mod N, \text{ where } N = pq$$

- **ElGamal scheme**

$$\frac{m(g^d)^r}{(g^r)^d} = m, \text{ where } g \text{ is a generator of a cyclic group}$$

Exercises

- Is RSA IND? Is ElGamal IND?
- For public-key encryption: Probabilistic encryption is required!
- For secret-key encryption: deterministic encryption could be semantically secure [Phan-Pointcheval '04]

Semantic security/IND is enough?

ElGamal Encryption

- Elgamal encryption can be proven to be IND, based on Decisional Diffie-Hellman assumption (given g^a, g^b, it is hard to distinguish between g^{ab} and a random element g^z).
- Elgamal encryption is homomorphic: $E(m_1 m_2) = E(m_1)E(m_2)$

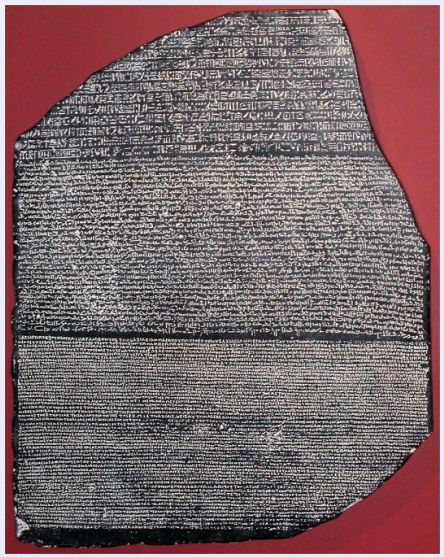
Private Auctions

The bids are encrypted. The authority then opens all the encrypted bids and the highest bid wins

- IND guarantees privacy of the bids
- Malleability: from $c = E(pk, b)$, without knowing b, one can generate $c' = E(pk, 2b)$: an unknown higher bid!
- Should consider adversaries with some more information.

Adversaries with additional information

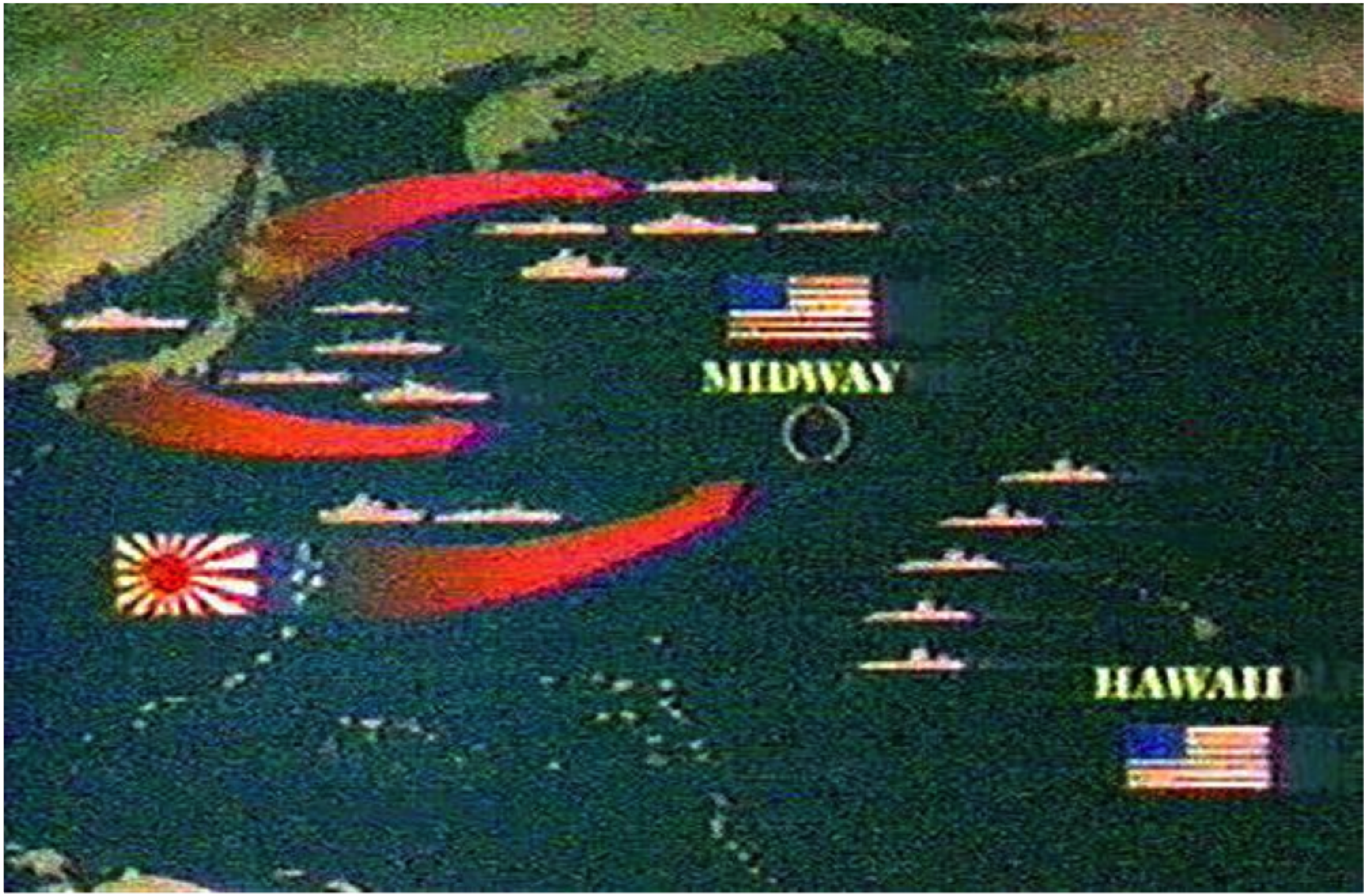
Rosetta Stone: A key element to decode Ancient Egyptian hieroglyphs



Chosen plaintext attacks (CPA)

The adversary can have access to encryption oracle (this only makes sense for symmetric encryption)

Interactive Adversaries: CCA attacks



IND-CCA Security Notion in Encryption

- **IND-CCA Security Game:**
  1. The adversary $\mathcal{A}$ is given pk and also given access to an oracle that decrypts ciphertexts.
  2. $\mathcal{A}$ chooses two plaintexts m_0 and m_1.
  3. A random bit $b \in \{0,1\}$ is selected, and the challenger encrypts m_b to get the challenge ciphertext $c^* = \mathsf{Enc}(pk, m_b)$.
  4. The ciphertext c^* is given to $\mathcal{A}$.
  5. $\mathcal{A}$ continues to have access to the decryption oracle, except for the challenge ciphertext (*i.e.,* cannot query c^*). (*)
  6. $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$.
- **Advantage:**

$$\mathsf{Adv}_{\mathcal{A}}^{\text{IND-CCA}} = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

- **IND-CCA Security:**

$$\forall \text{polynomial-time } \mathcal{A}, \mathsf{Adv}_{\mathcal{A}}^{\text{IND-CCA}} \text{ is negligible}$$

CCA1: CCA without access to the decryption oracle in the second phase (*)

Chosen plaintext and chosen ciphertext attacks

IND-CCA Security

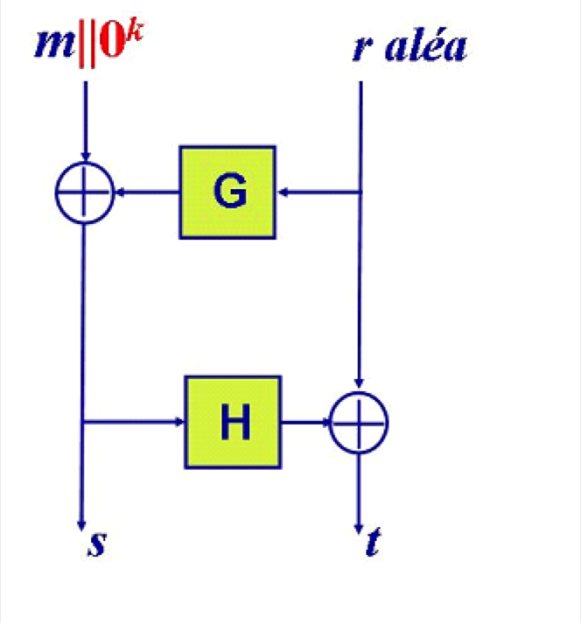
- IND-CCA also implies non-malleability (NM-CCA)
- This is the standard notion for public-key encryption
- Exercise: Is ElGamal IND-CCA?

Major problem in cryptography

Construction of IND-CCA encryption schemes.

OAEP (Bellare-Rogaway94)

Random oracle model



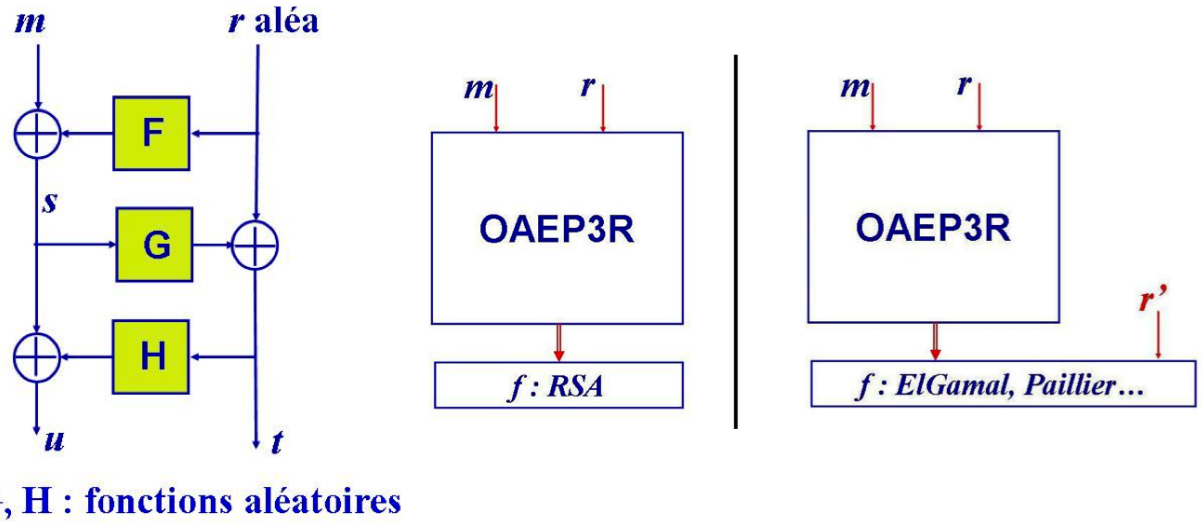
- It is believed that *f*-OAEP is IND-CCA for any trapdoor one-way permutation.
- In 2000, Shoup presented an attack on a very special trapdoor one-way permutation.

RSA-OAEP

RSA-OAEP is proven IND-CCA secure [Fujisaki-Okamoto-Pointcheval-Stern01]

- If f is partially one-way, then f-OAEP is secure
- RSA is partially one-way

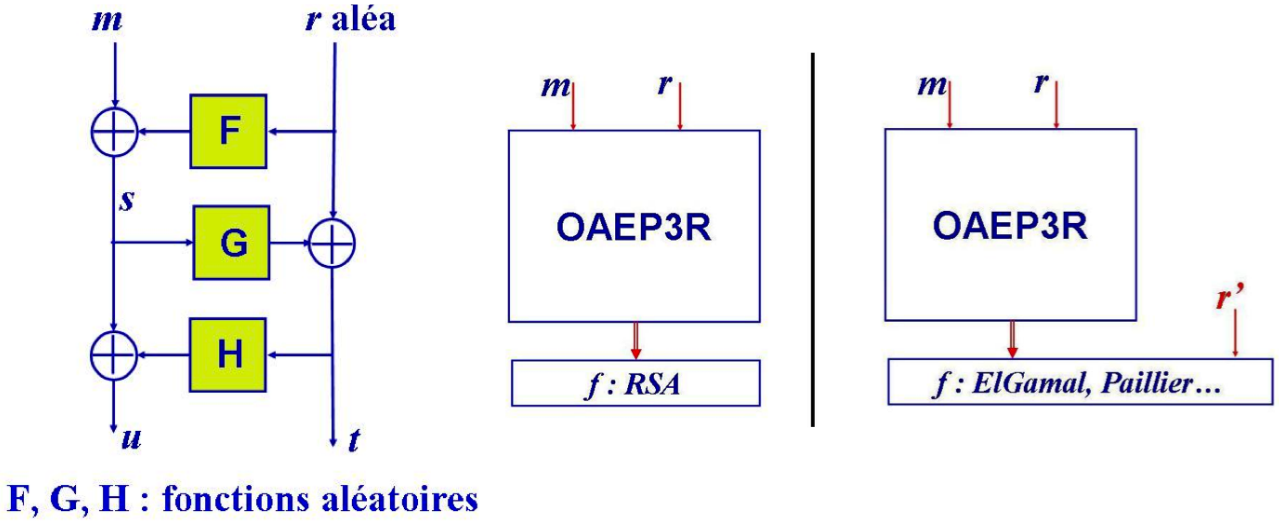
3-round OAEP (among others varieties of OAEP)



Advantages

- f does not need to be partially one-way
- f could also be one-way function (such as Elgamal, Paillier encryptions...)

3-round OAEP (among others varieties of OAEP)



Advantages

- f does not need to be partially one-way
- f could also be one-way function (such as Elgamal, Paillier encryptions...)

Current state

Many solutions in the standard model (without random oracle) but the practical implementations mostly rely on RSA-OAEP.

Security Proofs: Game Sequence technique

Proof of IND-CPA of ElGamal scheme, under DDH assumption

Let $\mathbb{G} = \langle g \rangle$ with generator g of order $|\mathbb{G}| = q$ where q is a prime.
Public key $pk = (g, h = g^x)$ and secret key $sk = x$.
Encryption:Enc$(pk, m) = (g^r, h^r \cdot m)$ where $r \leftarrow \mathbb{Z}_q$.

- **Game 0:** Real IND-CPA game, challenge ciphertext is $(g^r, h^r \cdot m_b)$
- **Game 1:** Replace (g, h, g^r, h^r) by $(g, h, g^r, h^{r'})$, for random r, r'
  The adversary cannot distinguish Game 0 and Game 1, otherwise we can solve DDH
- In Game 1: the adversary has no information about m_b.

Modern Cryptography

Beyond Encryption:

- Interactive proofs, zero-knowledge proofs, PCP
- Identification, Digital Signature
- Computation on Encrypted Data (Functional Encryption, FHE)
- Decentralized computation/ Verifiable computation (beyond data security)
- Multi-party computation (for doing any cryptographic task imaginable!)

Beyond Standard models

An example: Anamorphic Encryption $\rightarrow$
`https://www.di.ens.fr/~phan/anamorphic.html`

Two Assumptions for the Design of a Cryptographic Protocol

Encryption guarantees message confidentiality only with respect to parties that do not have access to the receiver's private key

The receiver-privacy assumption

The receiver keeps his secret key in a private location

A ciphertext carries the message that was provided as an input, not the one that the sender wishes to encrypt

The sender-freedom assumption

The sender is free to pick the message to be encrypted

Receiver privacy and Sender freedom

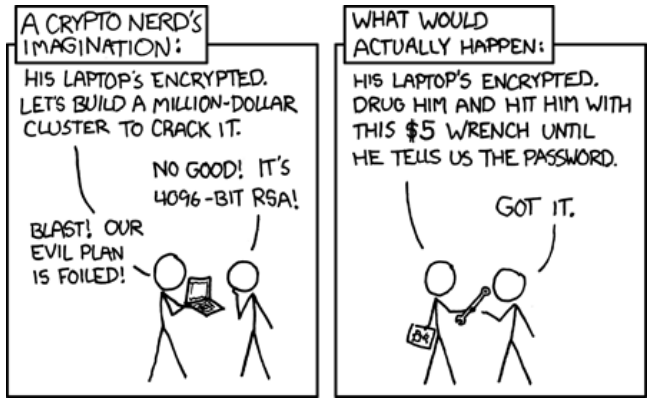
- Both assumptions are realistic for “normal” settings

Receiver privacy and Sender freedom

- Both assumptions are realistic for “normal” settings
- In a dictatorship, instead

Receiver privacy and Sender freedom

- Both assumptions are realistic for “normal” settings
- In a dictatorship, instead
  - No receiver privacy: citizens might be invited to surrender their private keys



(Source: `https://xkcd.com/538/`)

  - No sender freedom: dissidents might be forced to send messages to international newspapers to make the dictator look good

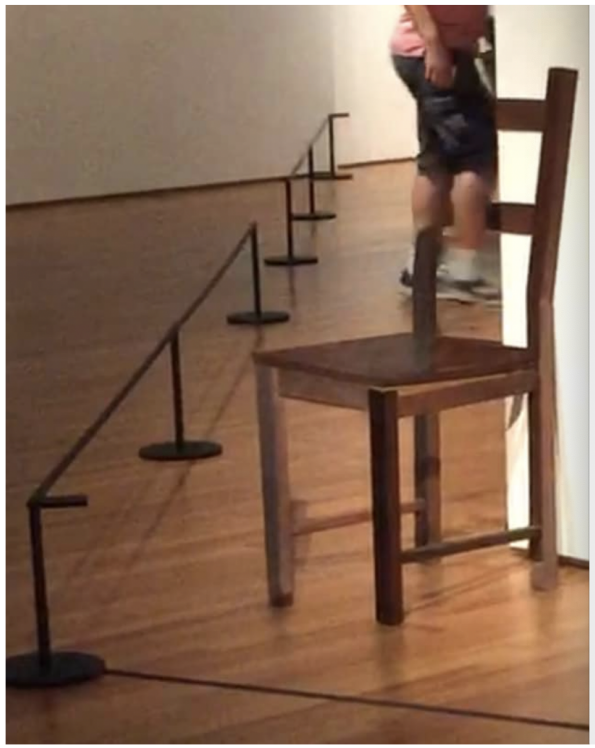
Two-View Principle

Anamorphic Art: Two Views on the Same Object

In anamorphic art, an object can be seen from many viewpoints, but the “complete” image only appears from a specific angle.

Two-View Principle

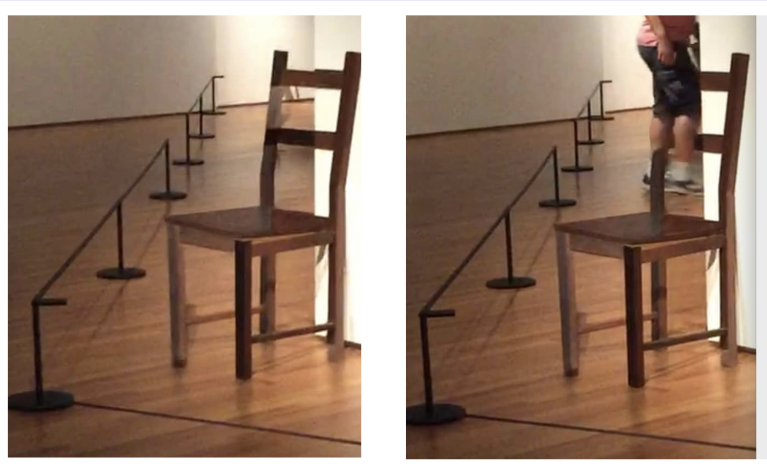
Anamorphic Art: Two Views on the Same Object



These images (*which I took at the National Gallery Singapore*) contain a proof of *anamorphism*: only from my specific position does the complete chair become visible.

Two-View Principle

Anamorphic Art: Two Views on the Same Object



Cryptography: Two Views on the "Same" Communication

- **Real view:** Actual interactions between insiders (holders of secret keys).
- **Simulated view:** Interactions are produced by a simulator.

Security: Real view $\approx$ *Simulated view* (e.g., ZKP, MPC)

Anamorphic Cryptography

[PPY22]

Anamorphic Ciphertext

$\xrightarrow{\textit{View with normal\_key}}$ a regular message

$\xrightarrow{\textit{View with double\_key}}$ *an anamorphic message*

Everything should look normal to a powerful adv. $\mathcal{D}$ (dictator):

Anamorphic Cryptography

[PPY22]

Anamorphic Ciphertext

$\xrightarrow{View\ with\ normal\_key}$ a regular message

$\xrightarrow{View\ with\ double\_key}$ *an anamorphic message*

Everything should look normal to a powerful adv. $\mathcal{D}$ (dictator):

- **Ciphertext rule:**

  Anamorphic ciphertexts $\overset{\mathcal{D}}{\approx}$ normal ciphertexts,

  for an allowed encryption, with a specific public key.
- **Key rule:** $\mathcal{D}$ can request the secret key corresponding to any public key (unlike in *steganography*).

Anamorphic Cryptography

[PPY22]

Anamorphic Ciphertext

$\xrightarrow{\textit{View with normal\_key}}$ a regular message

$\xrightarrow{\textit{View with double\_key}}$ *an anamorphic message*

Everything should look normal to a powerful adv. $\mathcal{D}$ (dictator):

- **Ciphertext rule:**

  Anamorphic ciphertexts $\overset{\mathcal{D}}{\approx}$ normal ciphertexts,

  for an allowed encryption, with a specific public key.
- **Key rule:** $\mathcal{D}$ can request the secret key corresponding to any public key (unlike in *steganography*).
- **Blocking rule:** *D* can block any party from receiving communications (the fraction of blocked parties must remain bounded).

Crypto War: On the User's Side

The dictator enacts a law mandating weakened encryption or a built-in backdoor.
$\rightarrow$ A huge risk for **everyone**, including the dictator (e.g., Clipper chip, Dual_EC_DRBG).
Solution: Public debate, petitions, or technical demonstrations to oppose the approval of such laws.

The dictator permits standard encryption but remotely and massively controls all users:

- Require receivers to surrender their **secret keys** so all messages can be decrypted.
- **Block** any suspected users.

Anamorphic model provides a way to preserve users' privacy in this scenario.

How can we fix this?

Not by designing new schemes

- Suppose we design a new encryption scheme that is secure without assuming receiver privacy and/or sender freedom
- What is the problem?
  - It will be considered illegal
  - The simple act of using the new scheme will be self accusatory
  - The encryption scheme and its use will be seen as provocations

How can we fix this?

Not by designing new schemes

- Suppose we design a new encryption scheme that is secure without assuming receiver privacy and/or sender freedom
- What is the problem?
  - It will be considered illegal
  - The simple act of using the new scheme will be self accusatory
  - The encryption scheme and its use will be seen as provocations

Rather, we should find a way with existing *schemes.*

Existing schemes cannot be disallowed as there are legitimate uses for them.

Further Reading:
`https://www.di.ens.fr/~phan/anamorphic.html`

Main Theoretical Question (Complexity)

Does Cryptography really exist?

Central Question of Complexity: P vs. NP from the **Two-View Principle**

Central Question of Complexity: P vs. NP from the **Two-View Principle**

On a Mathematical Problem: Solver vs. Verifier Views

- In mathematics: **Solving** a problem is often more difficult than **Verifying** a proposed solution.

Central Question of Complexity: P vs. NP from the **Two-View Principle**

On a Mathematical Problem: Solver vs. Verifier Views

- In mathematics: **Solving** a problem is often more difficult than **Verifying** a proposed solution.
- In computer science: Tackle this distinction $\rightarrow$ formal notion of efficiency and difficulty $\rightarrow$ Computational Models & Algorithms.

Centre question of Complexity: P vs. NP

- P: Problems for which solutions can be "efficiently" found
- NP: Problems for which solutions can be "efficiently" verified

Centre question of Complexity: P vs. NP

- P: Problems for which solutions can be "efficiently" found
- NP: Problems for which solutions can be "efficiently" verified

Efficiency

- Formal definition of algorithm (Turing machine)
- Church-Turing Thesis: everything that nature computes, can be emulated on a Turing machine
- Efficient algorithm: number of basic steps is bounded by a polynomial on the size of the input

Centre question of Complexity: P vs. NP

- P: Problems for which solutions can be "efficiently" found
- NP: Problems for which solutions can be "efficiently" verified

Efficiency

- Formal definition of algorithm (Turing machine)
- Church-Turing Thesis: everything that nature computes, can be emulated on a Turing machine
- Efficient algorithm: number of basic steps is bounded by a polynomial on the size of the input
- Example
  - P: multiplication, exponentiation modulo a prime number,...
  - NP: factorisation, discrete logarithm, 3-coloring problem, sudoku,...

Centre question of Complexity: P vs. NP

- P: Problems for which solutions can be "efficiently" found
- NP: Problems for which solutions can be "efficiently" verified

Centre question of Complexity: P vs. NP

- P: Problems for which solutions can be "efficiently" found
- NP: Problems for which solutions can be "efficiently" verified

Definition of an NP Language

A language $\mathcal{L}$ is an NP-language if there is a polynomial-time verifier V such that:

- **Completeness:** True theorems have (short) proofs.
  For all $x \in \mathcal{L}$, there is a polynomial($|x|$))-size witness (proof) $w \in \{0,1\}^*$ such that $V(x,w) = 1$.
- **Soundness:** False theorems have no short proofs.
  For all $x \notin \mathcal{L}$, there is no witness.
  i.e., for all polynomially long $w \in \{0,1\}^*$, $V(x,w) = 0$.

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$: for all PPT adversary A:

$$\Pr_{x \in_R \{0,1\}^n}\left[A(1^n, f(x)) = x' \text{ and } f(x) = f(x')\right] \text{ is negligible.}$$

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$: for all PPT adversary A:

$$\Pr_{x \in_R \{0,1\}^n}\left[A(1^n, f(x)) = x' \text{ and } f(x) = f(x')\right] \text{ is negligible.}$$

- Trapdoor: given a trapdoor, it is easy to invert the function f.

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$: for all PPT adversary A:

$$\Pr_{x \in_R \{0,1\}^n}\left[A(1^n, f(x)) = x' \text{ and } f(x) = f(x')\right] \text{ is negligible.}$$

- Trapdoor: given a trapdoor, it is easy to invert the function f.

Necessary conditions for the existence of cryptography

- One-way function for secret-key cryptography

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$: for all PPT adversary A:

$$\Pr_{x \in_R \{0,1\}^n}\left[A(1^n, f(x)) = x' \text{ and } f(x) = f(x')\right] \text{ is negligible.}$$

- Trapdoor: given a trapdoor, it is easy to invert the function f.

Necessary conditions for the existence of cryptography

- One-way function for secret-key cryptography
- Trapdoor one-way function for public-key cryptography

Cryptography and the P vs. NP problem

(Trapdoor) one-way functions

A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a (trapdoor) function if it is

- Efficiently computable: $f(x)$ is efficiently computable for any $x \in_R \{0,1\}^n$
- Hard to invert: for a random $x \in_R \{0,1\}^n$, given $y = f(x)$, it is hard to find a $\bar{x}$ such that $y = f(\bar{x})$: for all PPT adversary A:

$$\Pr_{x \in_R \{0,1\}^n}\left[A(1^n, f(x)) = x' \text{ and } f(x) = f(x')\right] \text{ is negligible.}$$

- Trapdoor: given a trapdoor, it is easy to invert the function f.

Necessary conditions for the existence of cryptography

- One-way function for secret-key cryptography
- Trapdoor one-way function for public-key cryptography

5 Worlds in Impagliazzo's view

W1-Algorithmica: P = NP

One could use the method of verifying the solution to automatically solve the problem!

5 Worlds in Impagliazzo's view

W1-Algorithmica: P = NP

One could use the method of verifying the solution to automatically solve the problem!

W2-Heuristica: NP problems are hard in the worst case but easy on average.

There exist hard instances of NP problem, but to find such hard instances is itself a hard problem.

5 Worlds in Impagliazzo's view

W1-Algorithmica: P = NP

One could use the method of verifying the solution to automatically solve the problem!

W2-Heuristica: NP problems are hard in the worst case but easy on average.

There exist hard instances of NP problem, but to find such hard instances is itself a hard problem.

W3-Pessiland: NP problems hard on average but no one-way functions exist

It's easy to generate many hard instances of NP-problems, but no way to generate hard instances where we know the solution.

5 Worlds in Impagliazzo's view (cont.)

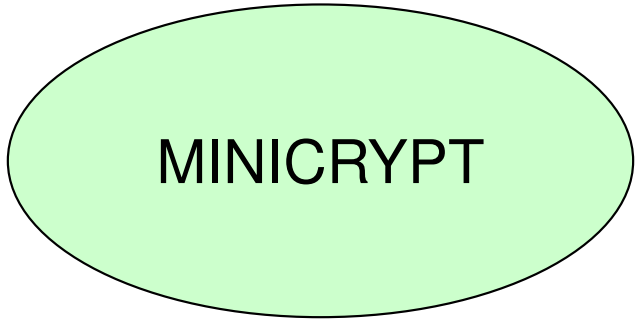
Minicrypt: One-way functions exist but public-key cryptography does not exist.

5 Worlds in Impagliazzo's view (cont.)

Minicrypt: One-way functions exist but public-key cryptography does not exist.

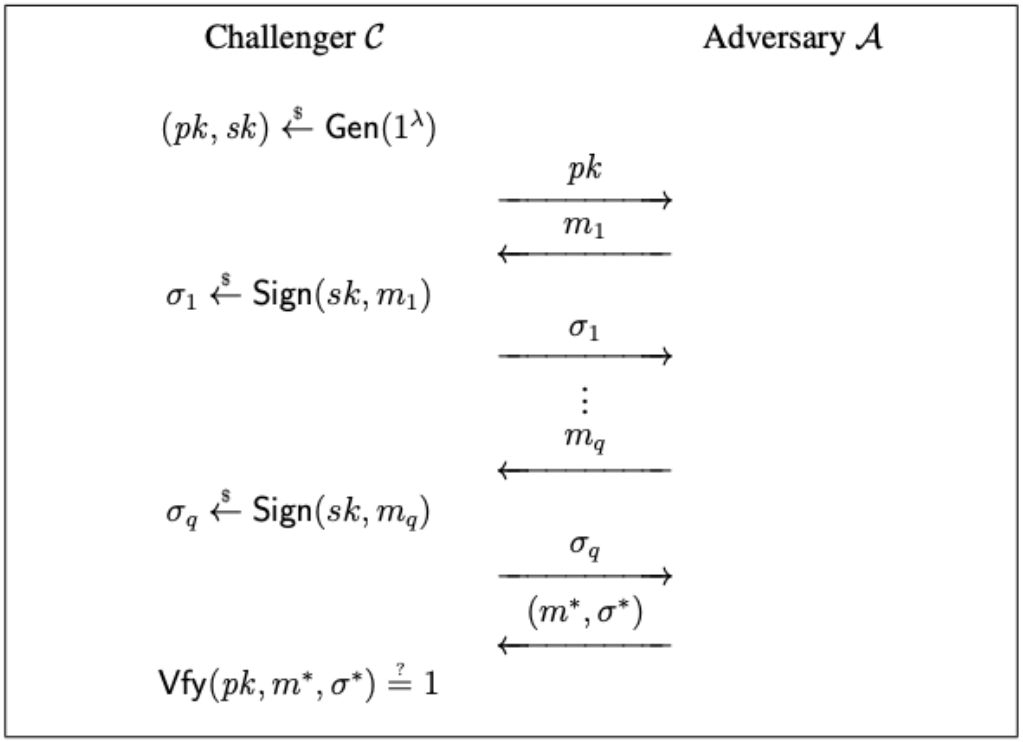
Cryptomania: Public-key cryptography is possible

It is possible for two parties to agree on a secret message using only public accessible channels


MINICRYPT

Outline

Digital Signatures: attack model (EUF-CMA) - Recall



Existential unforgeability under adaptive chosen message attacks

$$Adv(\mathcal{A}) = \Pr[Vfy(pk, m^\star, \sigma^\star) = 1]$$

The scheme is EUF-CMA secure if $\forall \mathcal{A}$, $Adv(\mathcal{A})$ is negligible.

Lamport's One-time Signatures from OWF f

- $Gen(1^\lambda) \to (pk, sk)$:

$$sk = \begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{\ell,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{\ell,1} \end{pmatrix}$$

$$pk = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{\ell,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{\ell,1} \end{pmatrix}$$

  where $x_{i,b} \in \{0,1\}^n, y_{i,b} = f(x_{i,b})$

- $Sign(sk, m = m_1 m_2 \ldots m_\ell \in \{0,1\}^\ell) \to \sigma$

$$\sigma = x_{1,m_1} x_{2,m_2} \ldots x_{\ell,m_\ell}$$

- $Vfy(pk, m, \sigma)$ check if $y_{i,m_i} = f(\sigma_i = x_{i,m_i}), \forall i = 1 \ldots \ell$

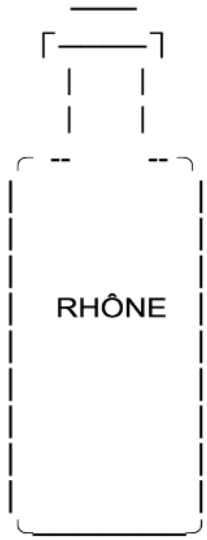
Theorem

If f is one-way, then the one-time signature is EUF-CMA.

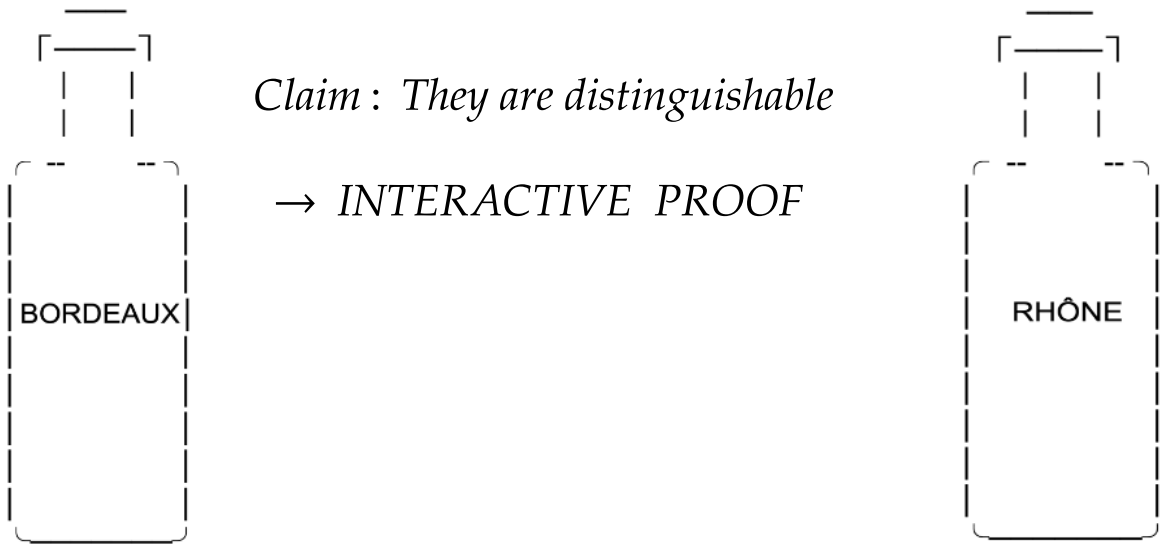
Mathematical vs. Interactive Proofs

Claim : They are distinguishable

Mathematical proof ?



Mathematical vs. Interactive Proofs



Interactive proofs [Goldwasser, Micali, Rackoff '85]

"A proof is whatever convinces me" (Shimon Even)

Mathematical vs. Interactive Proofs

Claim : They are distinguishable

→ INTERACTIVE PROOF

Verifier :

i) Convinced about the claim

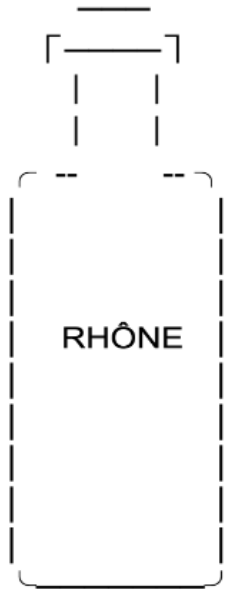
ii) Gain no knowledge

(Zero – knowledge proof)

→ *Learnability is hard!*

→ *Duality between :*

Cryptography *vs.* (Machine) Leaning



There are zero-knowledge proofs for all NP problems under the existence of one-way functions (Goldreich–Micali–Wigderson, 1991).

ZKP for the Discrete Logarithm Problem

ZKP: $G = \langle g \rangle$. Given g and $h = g^x$.
If the Discrete Logarithm problem is hard, I can convince you that I know x without revealing it.

ZKP for the Discrete Logarithm Problem

ZKP: $G = \langle g \rangle$. Given g and $h = g^x$.
If the Discrete Logarithm problem is hard, I can convince you that I know x without revealing it.
Idea: Consider the space generated by g and h over $\mathbb{Z}_q$:

- Scalar multiplication: $a \times g := g^a$
- Linear combination: $a \times g \boxplus b \times h := g^a h^b$
- Computational independence: g and h are considered independent if we cannot compute x such that $h = g^x$

ZKP for the Discrete Logarithm Problem

ZKP: $G = \langle g \rangle$. Given g and $h = g^x$.
If the Discrete Logarithm problem is hard, I can convince you that I know x without revealing it.
Idea: Consider the space generated by g and h over $\mathbb{Z}_q$:

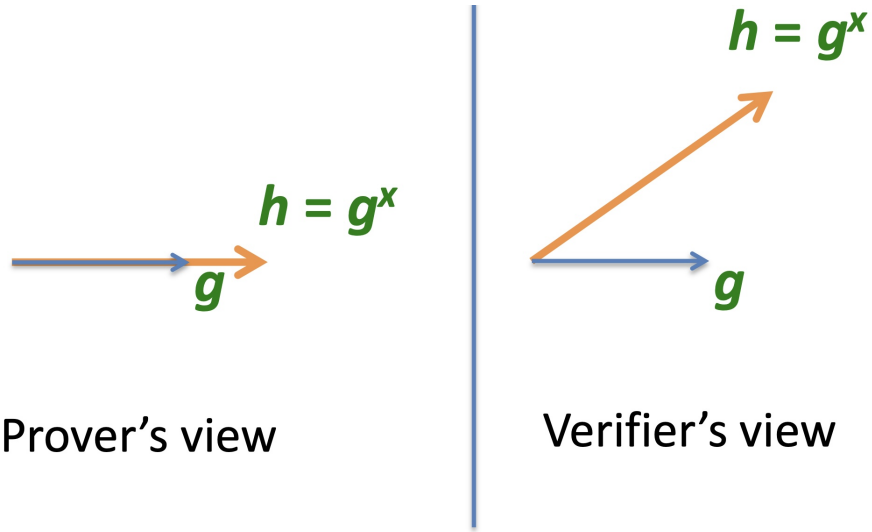
- Scalar multiplication: $a \times g := g^a$
- Linear combination: $a \times g \boxplus b \times h := g^a h^b$
- Computational independence: g and h are considered independent if we cannot compute x such that $h = g^x$

This space looks different from the Insider and Outsider views:

- **Insider (with x):** mathematical view $\rightarrow$ 1-dimensional
- **Outsider:** computational view $\rightarrow$ 2-dimensional

Proving knowledge of x: Convince the verifier that I live in a 1-dimensional space.

ZKP for Discrete Logarithm Problem



2 representions of $R = g^r$ (given r) in the basis of (g, $h = g^x$) require the knowledge of x.
1 represention of $R = g^r$ (given r) in the basis of (g, $h = g^x$) gives no information of x.

R →

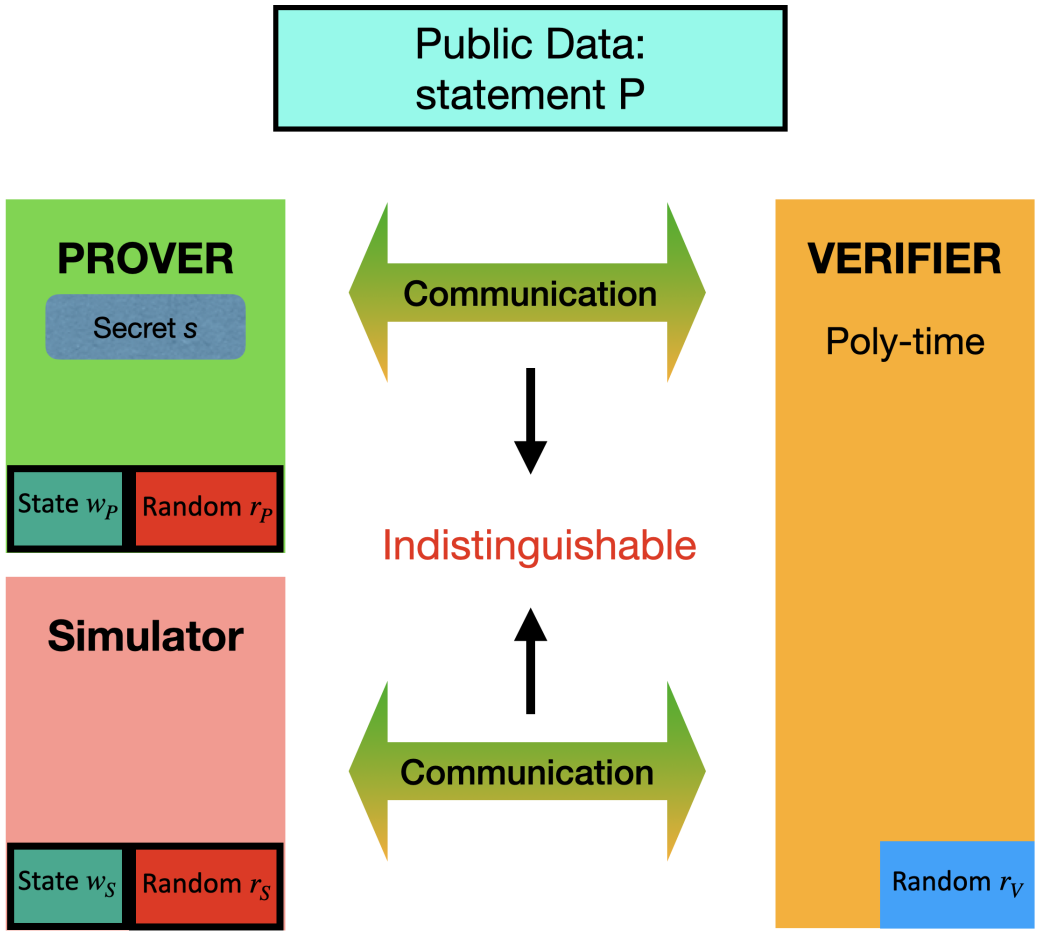
Random challenge b ←

$a = r - bx$ →

Check $R = g^a h^b$

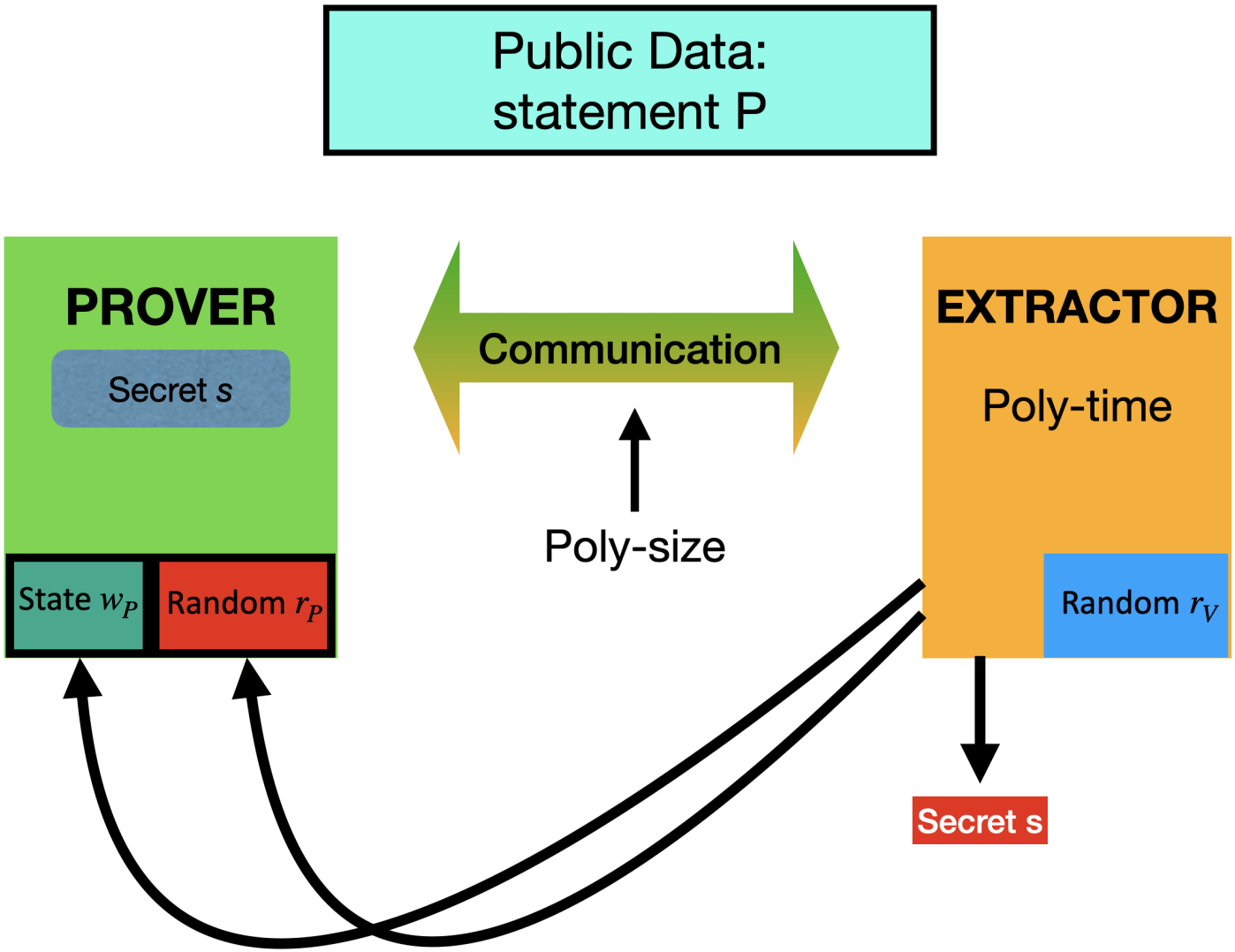
Set $b = H(R,M)$
☞ Digital Signature

Zero-knowledge Proof: Simulator



- Honest verifier: easy to simulate
- Dishonest verifier: more challenging

Zero-knowledge Proof of Knowledge: Extractor



(Zero-knowledge) Interactive Proof: Idea

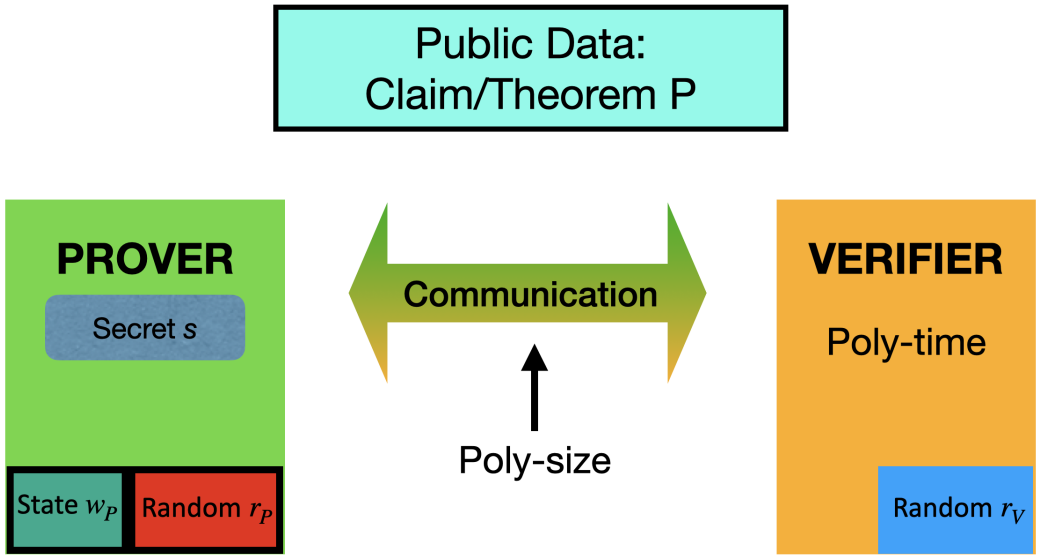
Zero-knowledge proofs: the verifier gets no information

- Verifier is poly-time TM, Prover could be all powerful
- Example: Graph non-isomorphism
- Simulation (zero-knowledge)

Zero-knowledge proof of knowledge.

- Verifier is poly-time TM, Prover is often poly-time TM as well
- Simulation (zero-knowledge) + Extraction (proof of knowledge)

Interactive Proofs



$\mathcal{L}$ is an **IP-language** if there is a **probabilistic poly-time** verifier V:

- **Completeness:** If $x \in \mathcal{L}$,

$$\Pr[(P, V)(x) = \text{accept}] = 1.$$

- **Soundness:** If $x \notin \mathcal{L}$, for every P^*,

$$\Pr[(P^*, V)(x) = \text{accept}] \text{ is negligible.}$$

Security
from the **Two-View Principle**

Security
from the **Two-View Principle**

Communication: Insider vs. Outsider Views

- Security is often established by showing that communication generated by an **insider** (who knows the secret) can be **simulated** by an **outsider** (who does not know the secret).

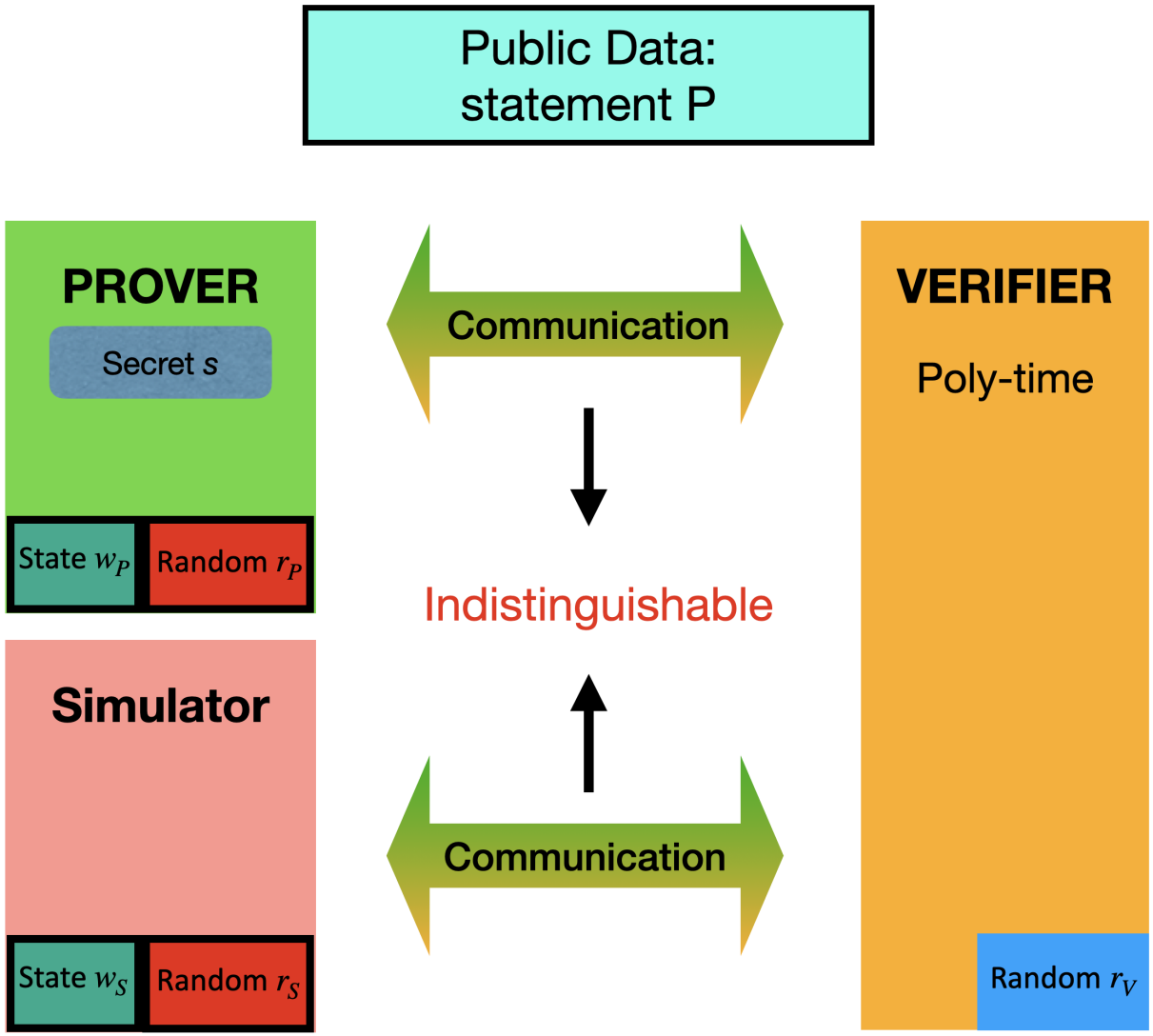
Security
from the **Two-View Principle**

Communication: Insider vs. Outsider Views

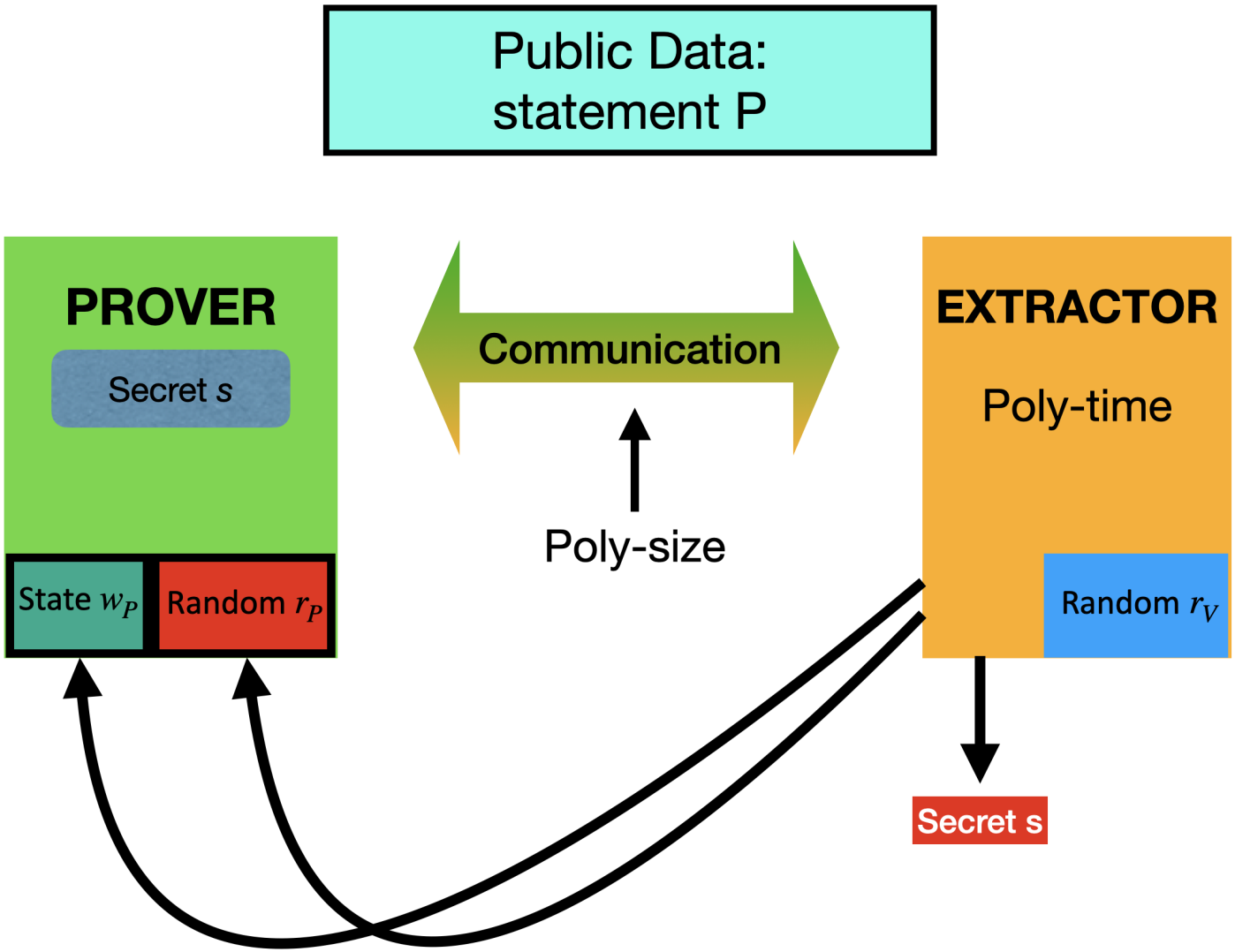
- Security is often established by showing that communication generated by an **insider** (who knows the secret) can be **simulated** by an **outsider** (who does not know the secret).

$\rightarrow$ This implies that the communication does not leak the secret.

Zero-knowledge Proof: Simulator



Zero-knowledge Proof of Knowledge: Extractor



Minicrypt: Commitment

- Alice **commits** herself to some message m by giving Bob: $c = \text{Commit}(m, r)$, for a random r.
- Bob should not learn anything about m given the commitment c.
- Alice can **open** the commitment by giving (m, r) to Bob to convince him that m was the value she committed herself to.

Two properties:

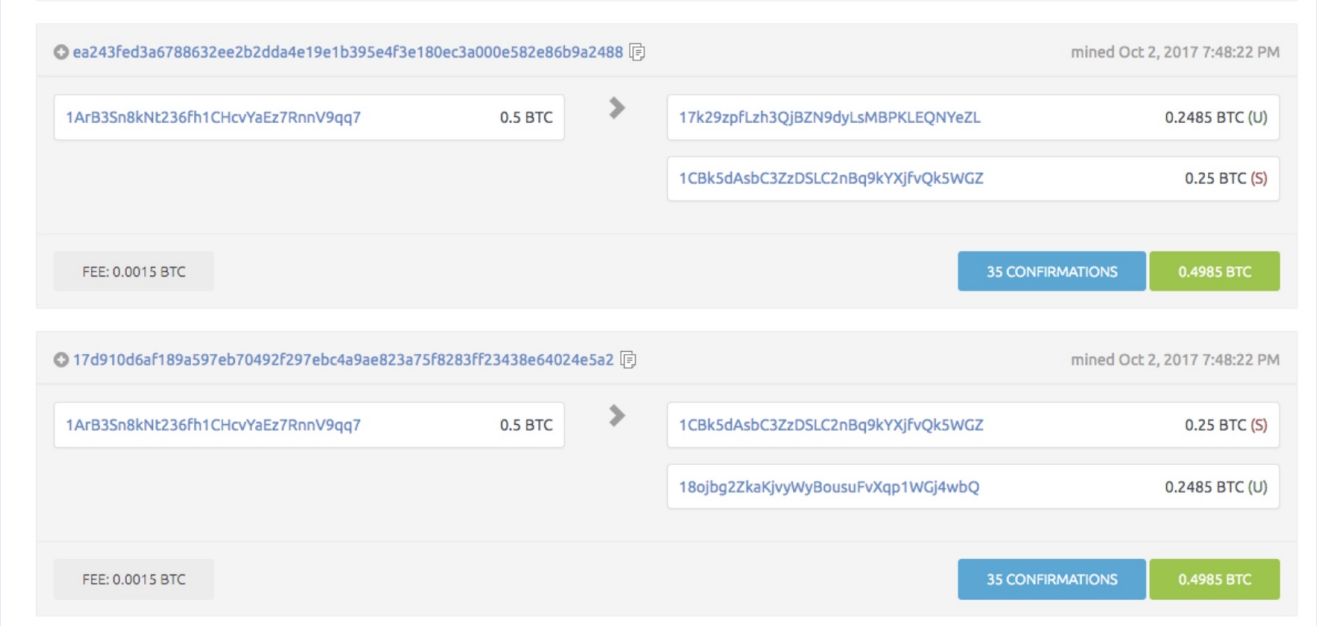
- **Hiding** Commitment c hides information on m
- **Binding** Alice cannot open c to $(m', r') \neq (m, r)$

Example & Application

- Pedersen's construction
- General construction from one-way function
- $\rightarrow$ In Minicrypt: ZKP for all NP problem (on ZKP for G3C)

ZKP in Practice: Privacy in Blockchain

A Bitcoin transaction



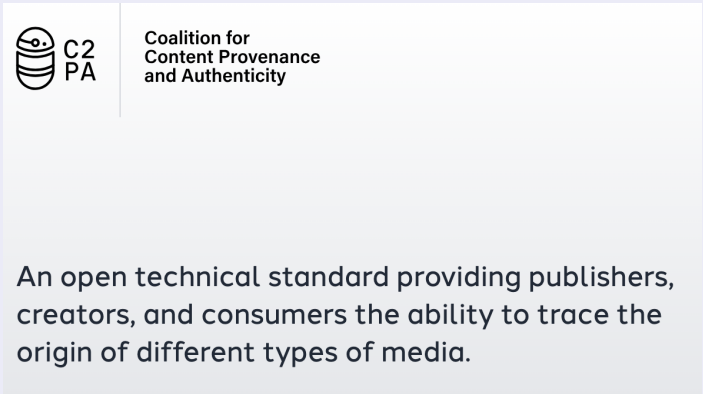
Privacy

- What is the problem with privacy in bitcoin?
- How we can use ZKP to solve this? $\rightarrow$ zkSNARKS.

Signatures/Commitment in Practice

(beyond classical examples)

C2PA and the need of Short Polynomial Commitment



Polynomial Commitment

Given a polynomial $P(x) = \sum_{i=0}^{n-1} a_i x^i$. We want the sender to commit P in such a way that it can prove to the receiver that (u, v) satisfies: $P(u) = v$.

- linear-size commitment: exercise
- constant-size commitment: KZG10, using pairings

KZG10 Polynomial Commitment: Setup

- **Setup:**
  1. Select a prime field $\mathbb{F}_p$ and a generator g of a group $\mathbb{G}$ of prime order p.
  2. Choose a random $s \in \mathbb{F}_p$.
  3. Compute $\{g_0 = g, g_1 = g^s, g_2 = g^{s^2}, \ldots, g_d = g^{s^d}\}$ for a polynomial of degree d.
  4. Publish the setup parameters $\text{PP} = \{g_0, g_1, g_2, \ldots, g_d\}$.
- **Trusted Setup Assumption:** The trusted setup randomly generates s, compute $\{g_0, g_1, g_2, \ldots, g_d\}$, then erase s.

KZG10 Polynomial Commitment: Commitment

- **Polynomial:** $P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_d x^d$
- **Commitment:**

$$\begin{aligned} C &= g_0^{a_0} \cdot g_1^{a_1} \cdot g_2^{a_2} \cdot \cdots \cdot g_d^{a_d} \\ &= g^{a_0} \cdot (g^s)^{a_1} \cdot (g^{s^2})^{a_2} \cdot \cdots \cdot (g^{s^d})^{a_d} \\ &= g^{P(s)} \end{aligned}$$

- **Result:** The commitment C is a single group element in $\mathbb{G}$.

KZG10 Polynomial Commitment: Opening

- **Opening:**
  - To open the commitment at point $x = u$, compute the evaluation $v = P(u)$.
  - u is a root of $P(x) - v$.
  - We can write thus $P(x) - v = (x - u)Q(x)$ and can compute $Q(x)$.
  - Generate proof $\pi = g^{Q(s)}$.
- **Send to Verifier:** $\{u, v, \pi\}$

KZG10 Polynomial Commitment: Verification

- **Verification:**
  1. Verifier receives $\{u, v, \pi\}$ and the commitment C.
  2. IDEA: Check at the random point s if $P(s) - v = (s - u)Q(s)$
  3. This check can only performed with a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$
  4. Compute:

$$e(g_1 g^{-u}, \pi) \stackrel{?}{=} e(g, Cg^{-v})$$

- **Result:** If the equality holds, the proof is valid and the polynomial evaluates to $v = P(u)$ at point u.