CRYPTOGRAPHY

(Lecture 1)

Literature:

“**Handbook of Applied Cryptography**” **(ch** 1, **2.0, 2.1.1,2.1.2,2.1.3,9.1,9.2.2)**, optional 2.2.1
“Lecture Notes on Introduction to Cryptography” by V. Goyal (ch2.0-2.3, **11.1-11.3**)
“A Graduate Course in Applied Cryptography” by D. Boneh and V. Shoup (ch 3.12)
“Commitment Schemes and Zero Knowledge Protocols” by I. Damgård, J. Buus Nielsen

Lecture Agenda

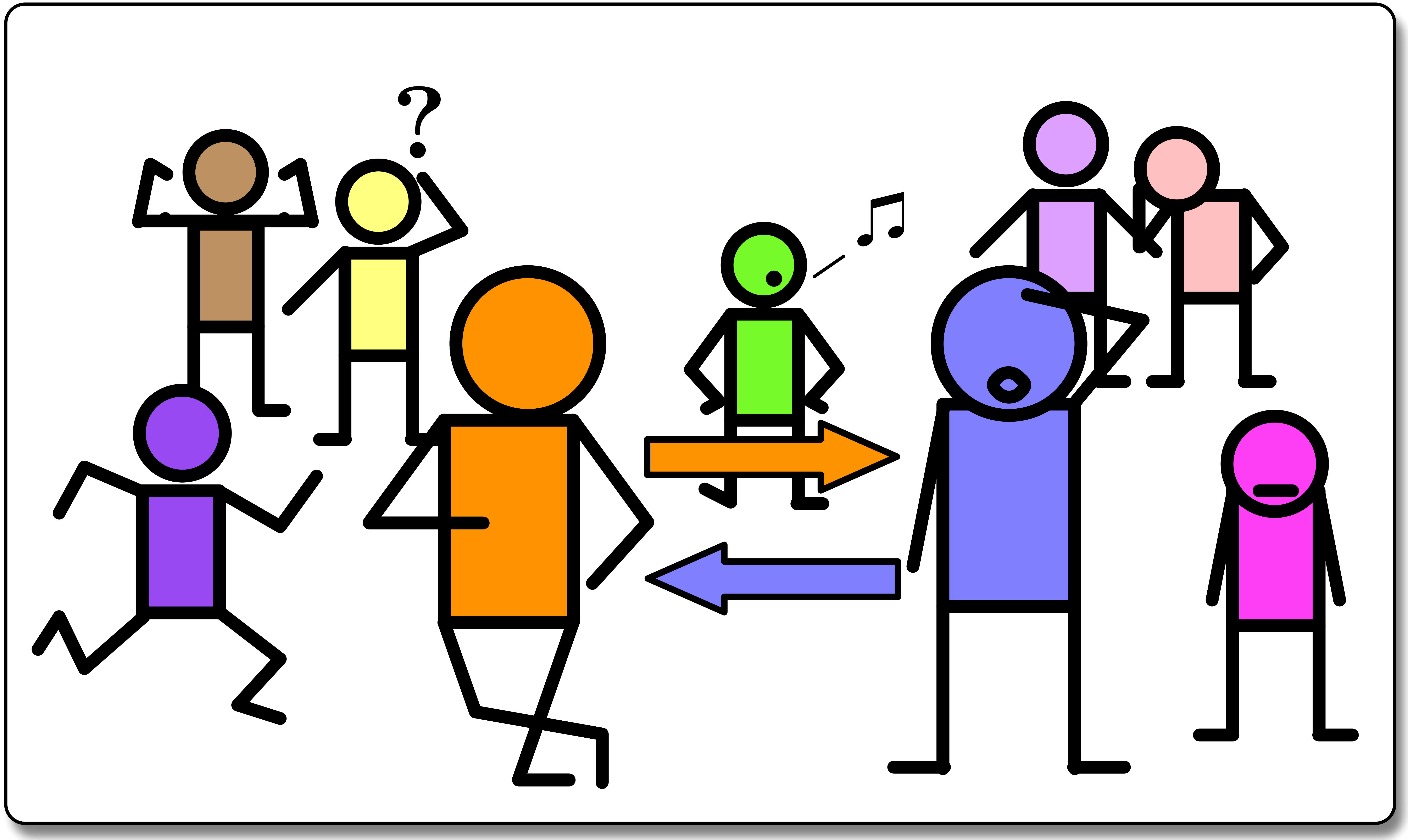
Introduction

- Cryptography: Meaning and Aims
- Core Concepts in Modern Cryptography
- The Attacker's Resources
- Terminology

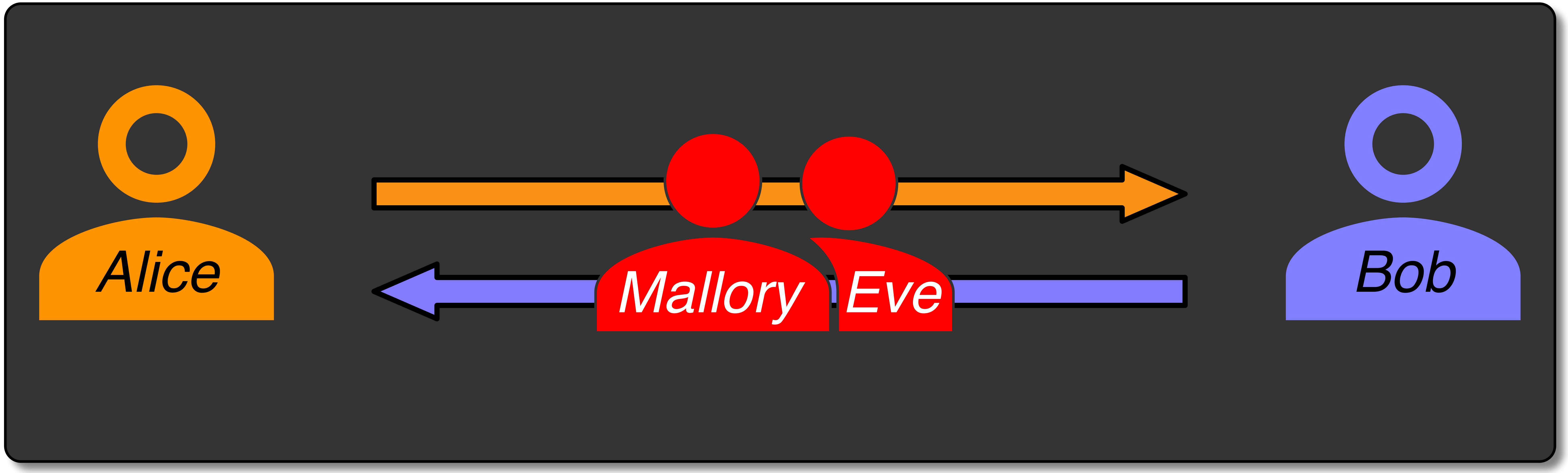
Commitment Schemes & One-Way Functions

- Intuition
- Cryptographic Hash Functions
- Definitions (Syntax & Properties)
- Constructions

The Real World



The World to the Eyes of Cryptography



The Goal of Cryptography: “Make our Digital World Safe”

- 🔒 Confidentiality
- 💌 Data integrity
- 🔏 Authenticity
- 🪪 Entity identification
- ⛔ Access control / authorisation
- 🤡 Anonymity
- ✅ Non-repudiation
- 👁️ Privacy

Foundations of *Modern* Cryptography (1980-Now)

CRYPTOGRAPHY

CRYPTANALYSIS

CRYPTOLOGY

Rigorous definitions

- What does security mean?
- What are the attacker's goal and resources?
- Precise mathematical security assumptions (formally define "hard")

Rigorous logic reasoning to prove security

Lots of heuristics to define exact security levels

Solutions need to work in practice

- Efficient algorithms
- Use the best size/security ratios

When I say "crypto" I mean "cryptography" not "cryptocurrency"

Useful Terminology

Deterministic : refers to a value that is set, or to a function that given an input always returns the same output.

Notation: $b = 0$, $Alg(x) = y$

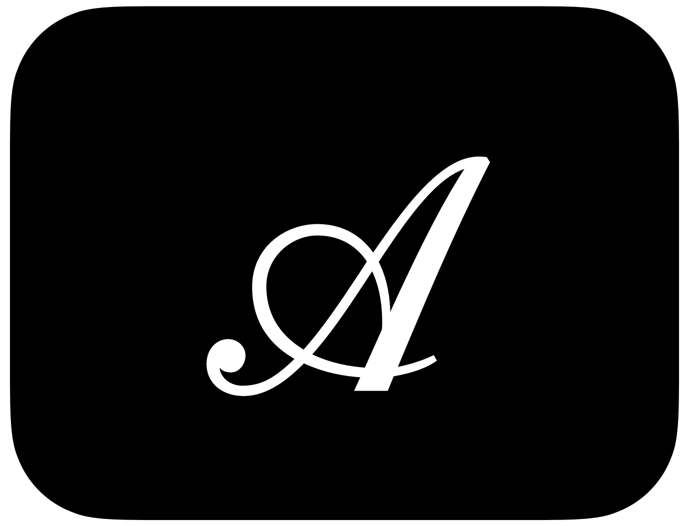
Random : refers to a value that is drawn from a set using the uniform distribution (all possibilities are equiprobable).

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

Randomised or **Probabilistic** : refers to a function or algorithm that involves sampling and using randomness, thus the output is non-deterministic (unless the randomness is specified).

Notation: $y \leftarrow Alg(x)$ and there exists $rnd \in \{0,1\}^n$ such that $y = Alg(x; rnd)$

The Adversary in Cryptography



The Attacker's Resources

$\mathscr{A}$

Adversarial Behaviour: the actions that corrupted parties are allowed to take.

- **Passive**: $\mathscr{A}$ monitors the communication channel as an eavesdropper, but does not modify messages between parties.
- **Active**: $\mathscr{A}$ monitors the communication channel as an eavesdropper and additionally can drop, alter or stop information sent between parties.

Adversarial (Computational) **Power:**

- **Polynomial time** (classical) **:** $\mathscr{A}$ is allowed to run in (probabilistic) polynomial time (and sometimes, expected polynomial time). This is abbreviated in **PPT** or "**efficient**".
- **Computationally unbounded**: $\mathscr{A}$ has no computational limits whatsoever, is not bound to any complexity class and is not assumed to run in polynomial time.
- **Quantum**: $\mathscr{A}$ has access to a quantum computer.

One Fundamental Definition

"too small to matter"

Negligible A function $negl(x) : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$ is **negligible** in x if *for any* positive polynomial $p(x)$ it holds that

$$negl(x) \leq \frac{1}{p(x)} \quad \text{for all } x \geq x_0 \in \mathbb{N}$$

Intuition: Events that occur with negligible probability occur so seldom that polynomial time algorithms will never see them happening.

This definition is asymptotic (“it holds from a certain point onwards”). This is a common approach in complexity-based cryptography.

In practice, if one needs to pick a value, then $negl(x) < 2^{-128}$ is considered to be negligible (but this depends on the context, and may yield inefficient constructions).

Lecture Agenda

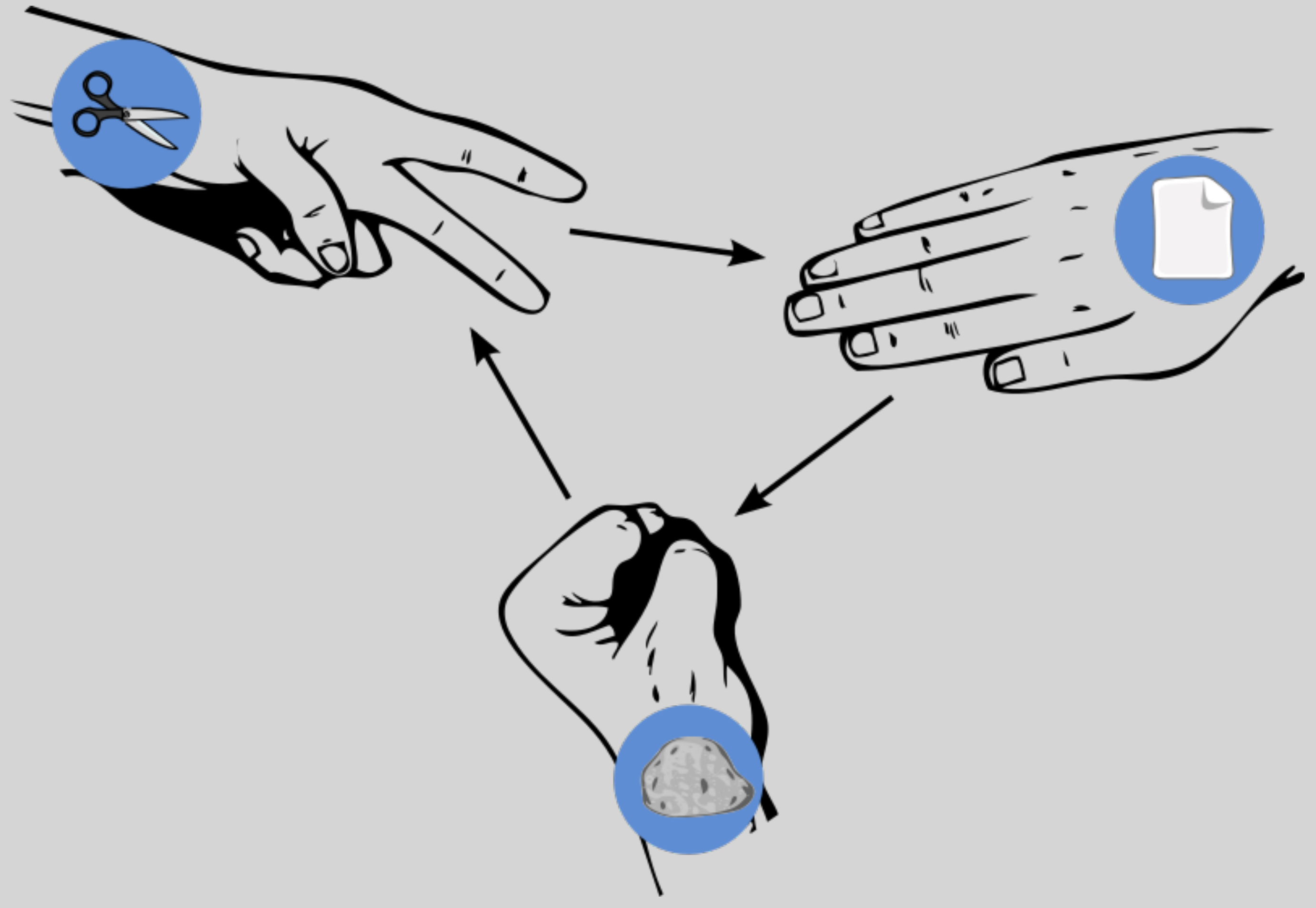
Introduction

- Cryptography: Meaning and Aims
- Core Concepts in Modern Cryptography
- The Attacker's Resources
- Terminology

Commitment Schemes & One-Way Functions

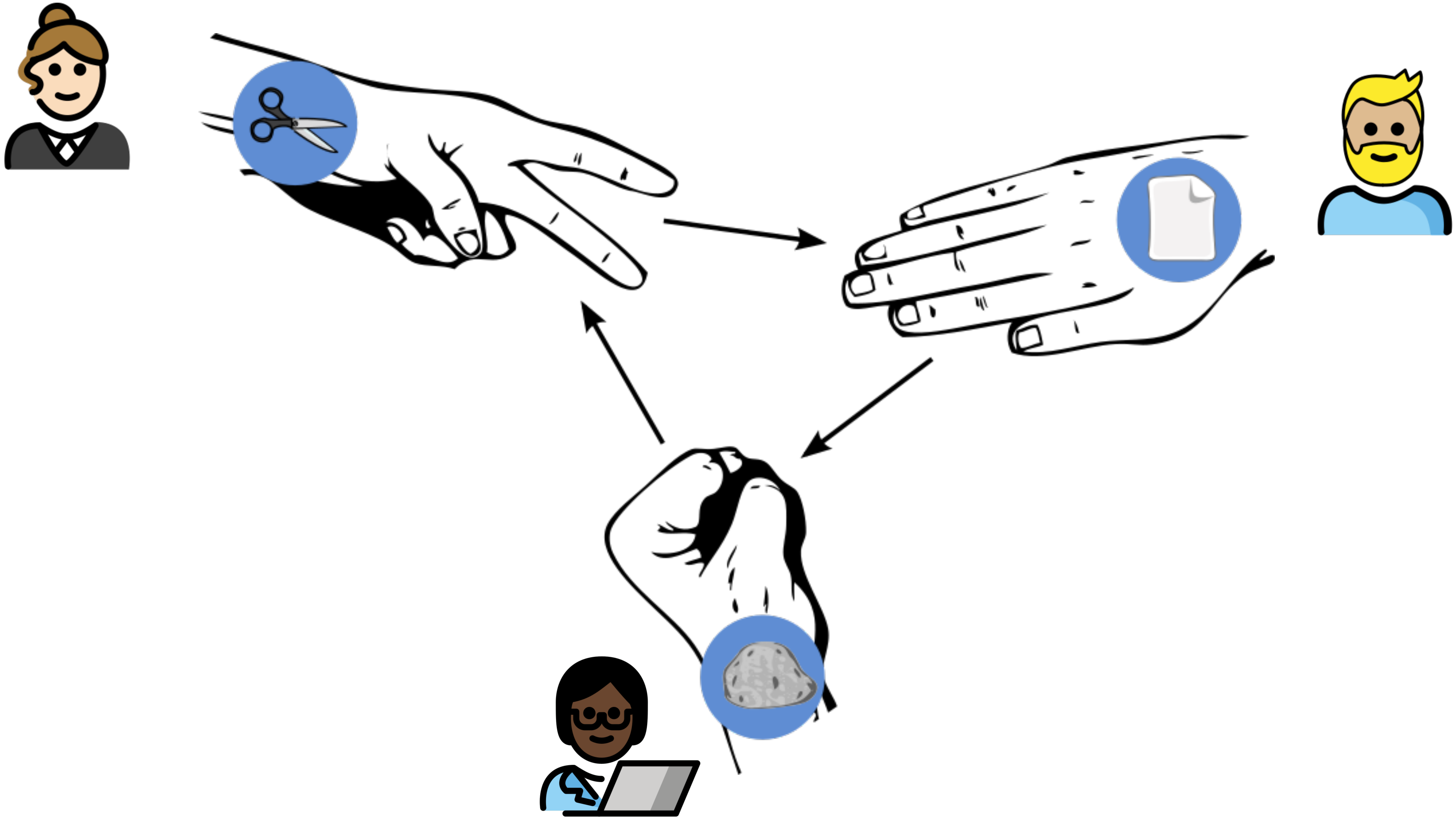
- Intuition
- Cryptographic Hash Functions
- Definitions (Syntax & Properties)
- Constructions

Home Assignment 1

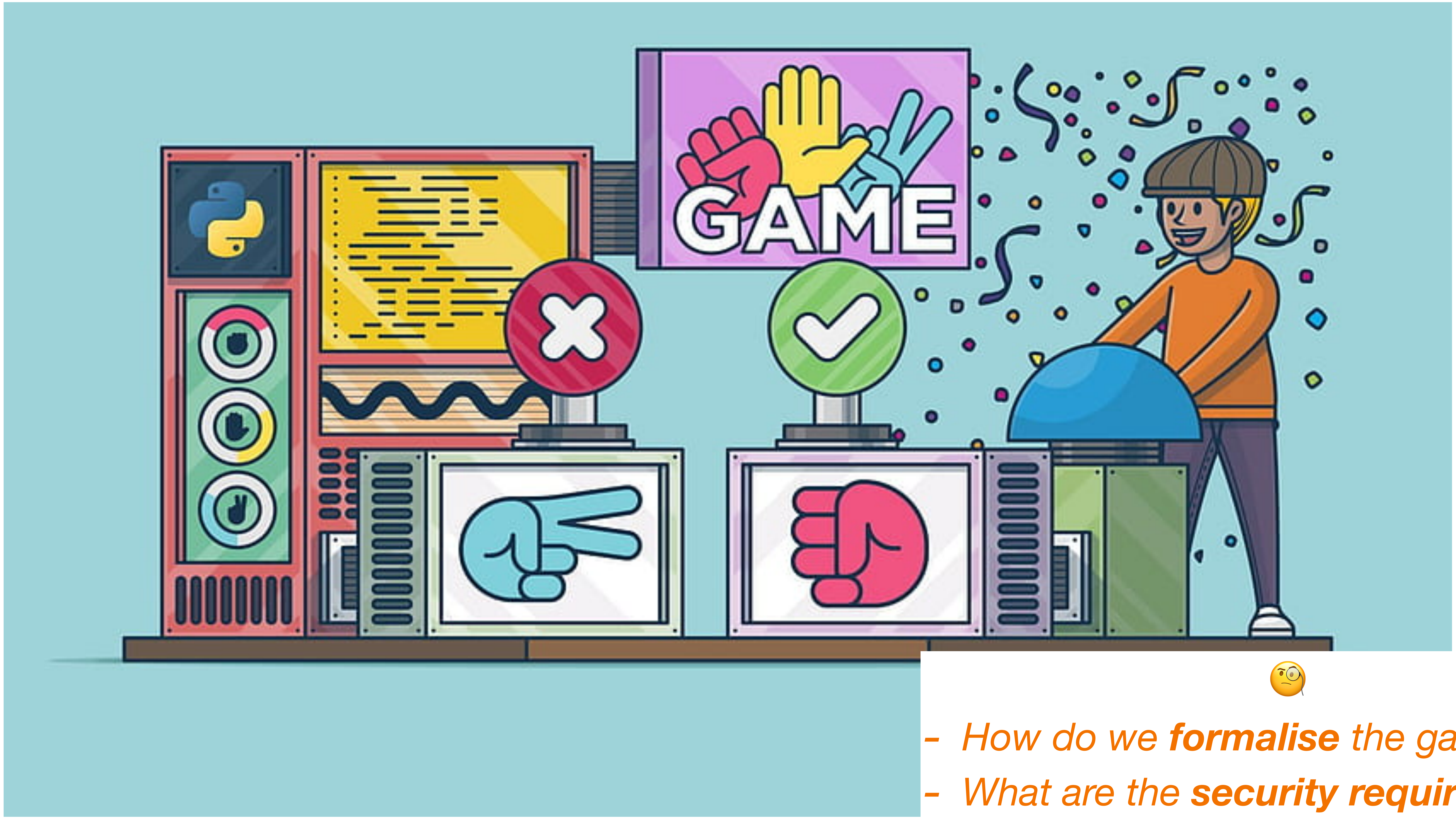


Deadline: Nov 15th (1st Submission)

Use Case: Playing Rock-Paper-Scissors



Rock-Paper-Scissors Over the Internet



🧐

- *How do we **formalise** the game?*
- *What are the **security requirements**?*
- *What **tool** can we use to **realise** this?*

One-Way Functions

"easy" to compute and "hard" to invert

One-Way Functions

Definition: ONE-WAY FUNCTION

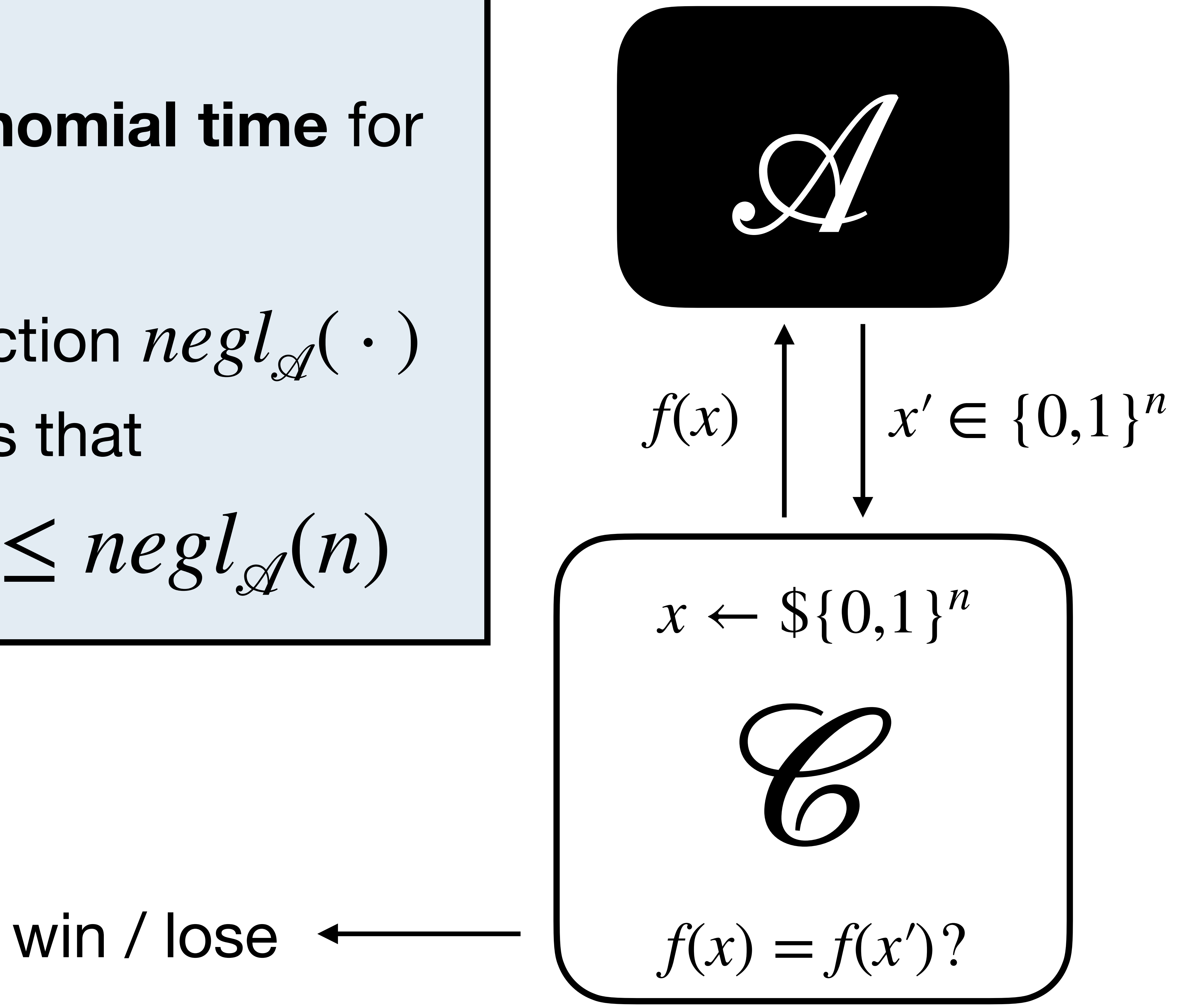
A function $f : \{0,1\}^n \rightarrow \{0,1\}^d$ is one-way if:

(1) There exists an algorithm that computes $f(x)$ in **polynomial time** for all inputs $x \in \{0,1\}^n$ (f is efficiently computable)

(2) For every PPT algorithm $\mathscr{A}$ there is a **negligible** function $negl_{\mathscr{A}}(\cdot)$ such that for sufficiently large values of $n \in \mathbb{N}$ it holds that

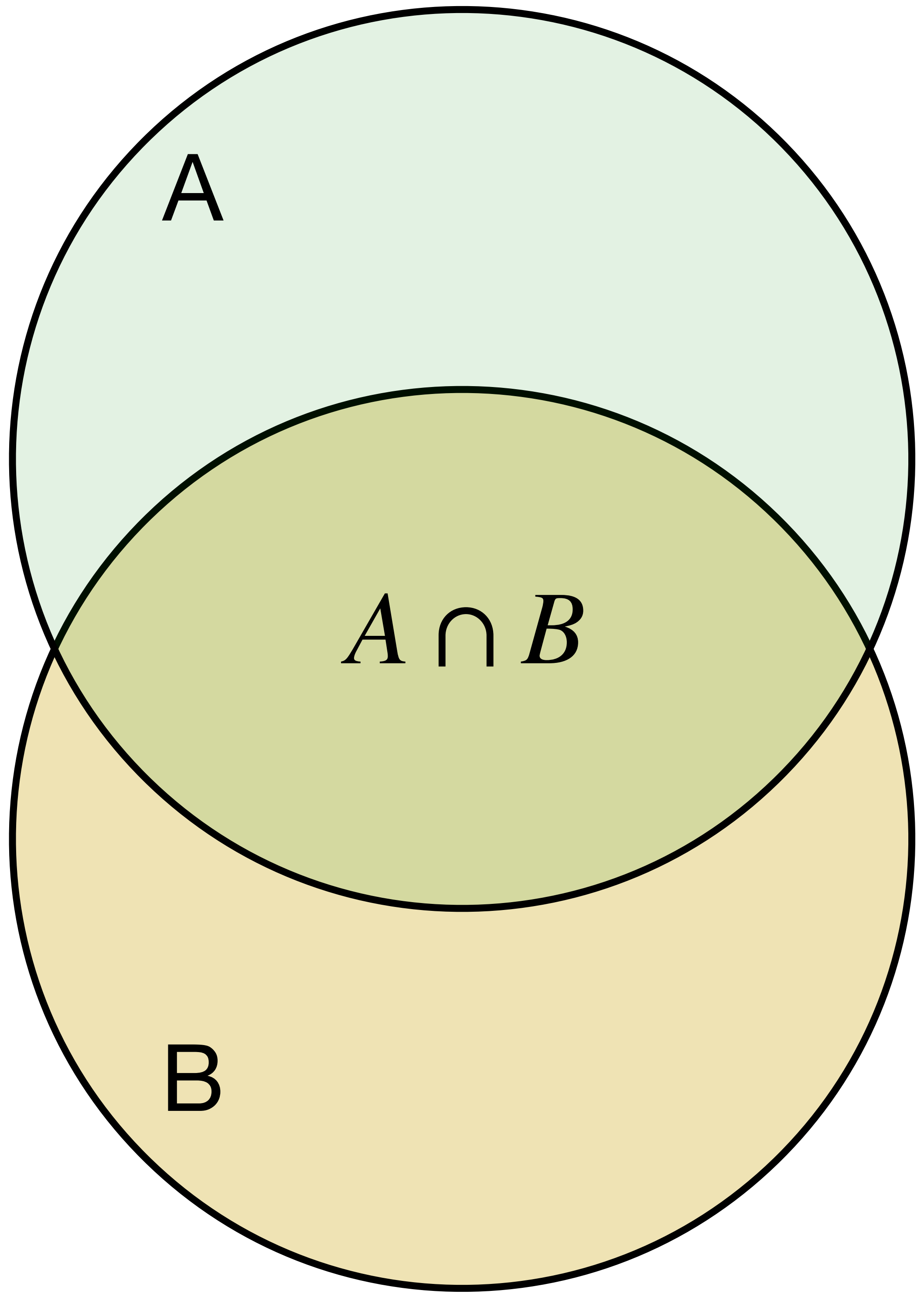
$$Pr[f(x) = f(x') \,|\, x \leftarrow \$\{0,1\}^n, x' \leftarrow \mathscr{A}(f(x))] \leq negl_{\mathscr{A}}(n)$$

conditional probability



Probability Theory Lightning-Fast Recap

Probability Theory provides rigorous foundation to measures the likelihood that an event happens.



🧐 *..“It's not at all hard to understand a person; it's only hard to listen without bias.”*

<u>Definition: CONDITIONAL PROBABILITY</u>

Given two events A,B with $Pr[B] > 0$, the conditional probability of event A *given* B (that is, the probability that A happens assuming B has happened) is denoted as $Pr[A \mid B]$ and it is computed as:

$$Pr[A \mid B] = \frac{Pr[A \cap B]}{Pr[B]}$$

<u>Bayes’ Theorem</u> (very useful when calculating values)

$$Pr[A \mid B] = \frac{Pr[A] \cdot Pr[B \mid A]}{Pr[B]}$$

Constructing One Way Functions (OWF)

Example: OWF from integer factorisation

Consider $f : \{k - bit\ primes\} \times \{k - bit\ primes\} \rightarrow \mathbb{N}$ defined as: $f(p, q) = p \cdot q$.

$f(\cdot)$ is a one-way function if integer factorisation is (computationally) hard.

🧐 *what happens if we consider* $f : \{primes\} \times \{primes\} \rightarrow \mathbb{N}, \ \ f(p, q) = p \cdot q$ *?*

Plenty more provable secure examples…but we need more math (Module 2)

A Special Case of OWF: Cryptographic Hash Functions

Definition: HASH FUNCTION

A function $H : \{0,1\}^n \rightarrow \{0,1\}^d$ is a cryptographic hash function if:

(1) H is a one-way function (efficient to compute, hard to invert)

And at least one of the following holds

(2) **Preimage resistance** (hard to invert when $d < n$ and n is large enough)

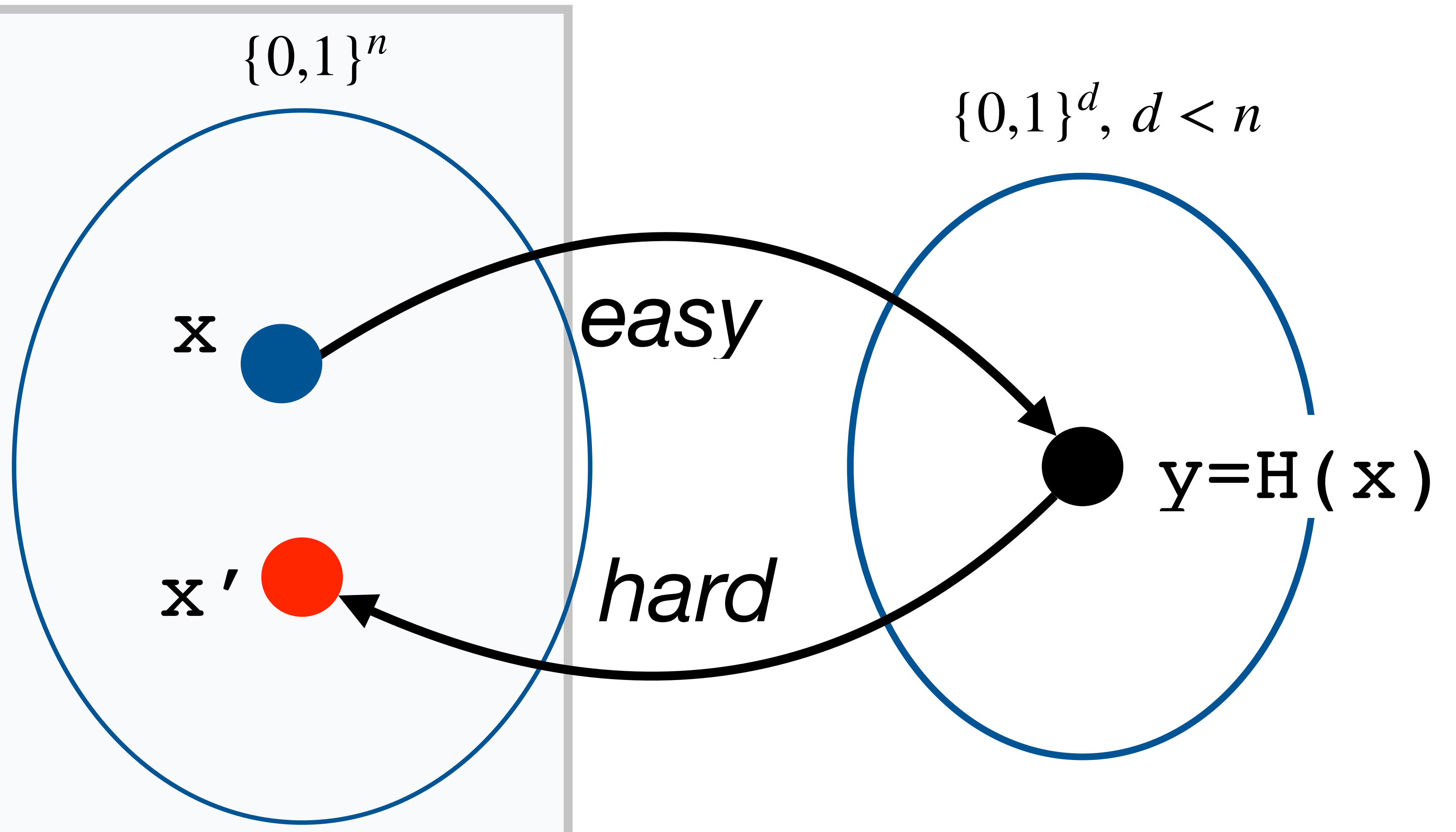
$$Pr[f(x) = f(x') \,|\, x \leftarrow \$\{0,1\}^n, x' \leftarrow \mathscr{A}(f(x))] \leq negl(n)$$

(3) **2nd preimage resistance**

$$Pr[f(x) = f(x') \,|\, x \leftarrow \$\{0,1\}^n, x' \leftarrow \mathscr{A}(x, f(x)), x \neq x'] \leq negl(n)$$

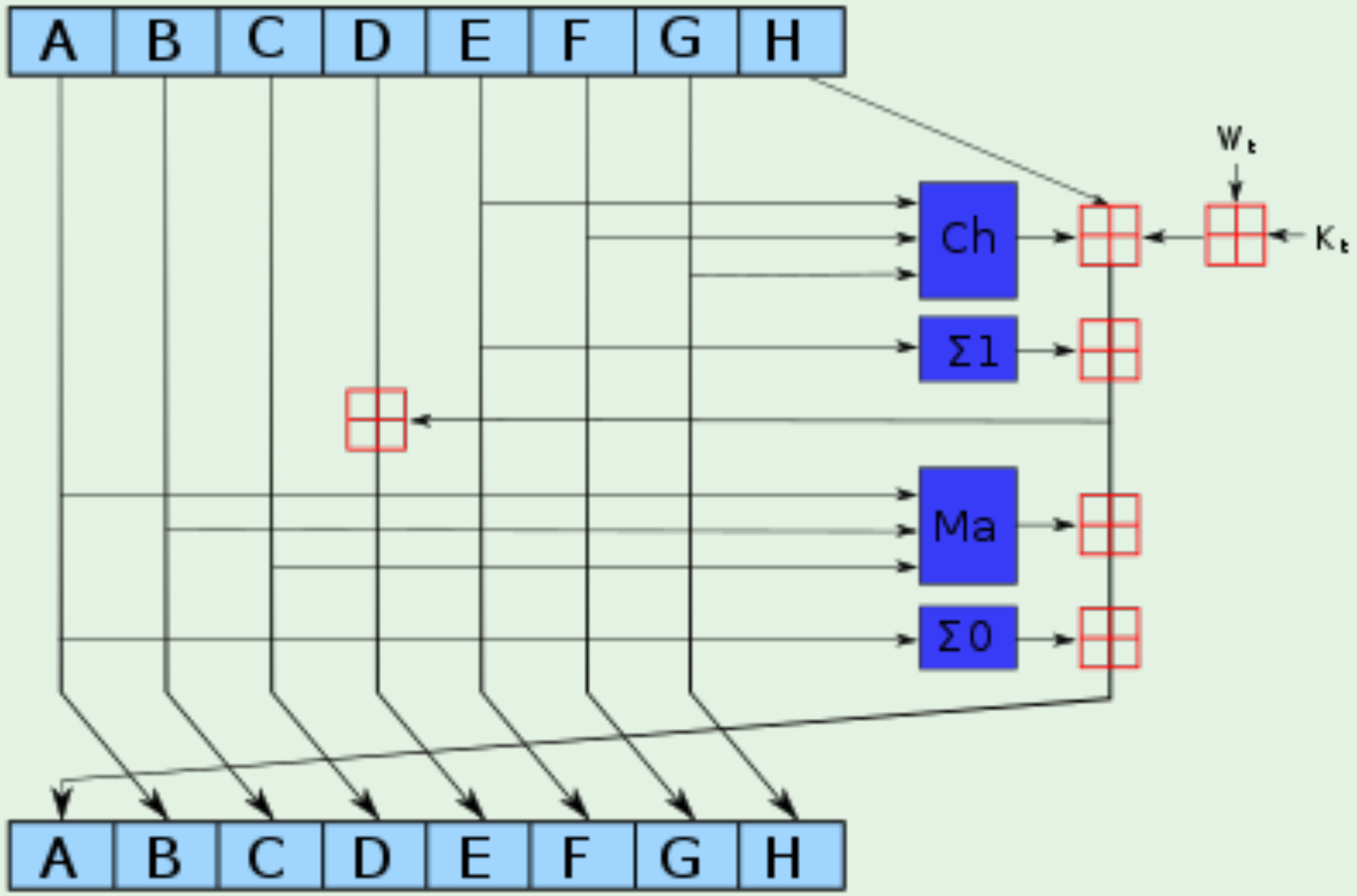
(4) **Collision resistance**

$$Pr[f(x) = f(x') \,|\, x, x' \leftarrow \mathscr{A}(f), x \neq x'] \leq negl(n)$$



State-of-the-Art: Secure Hash Algorithm (SHA2)

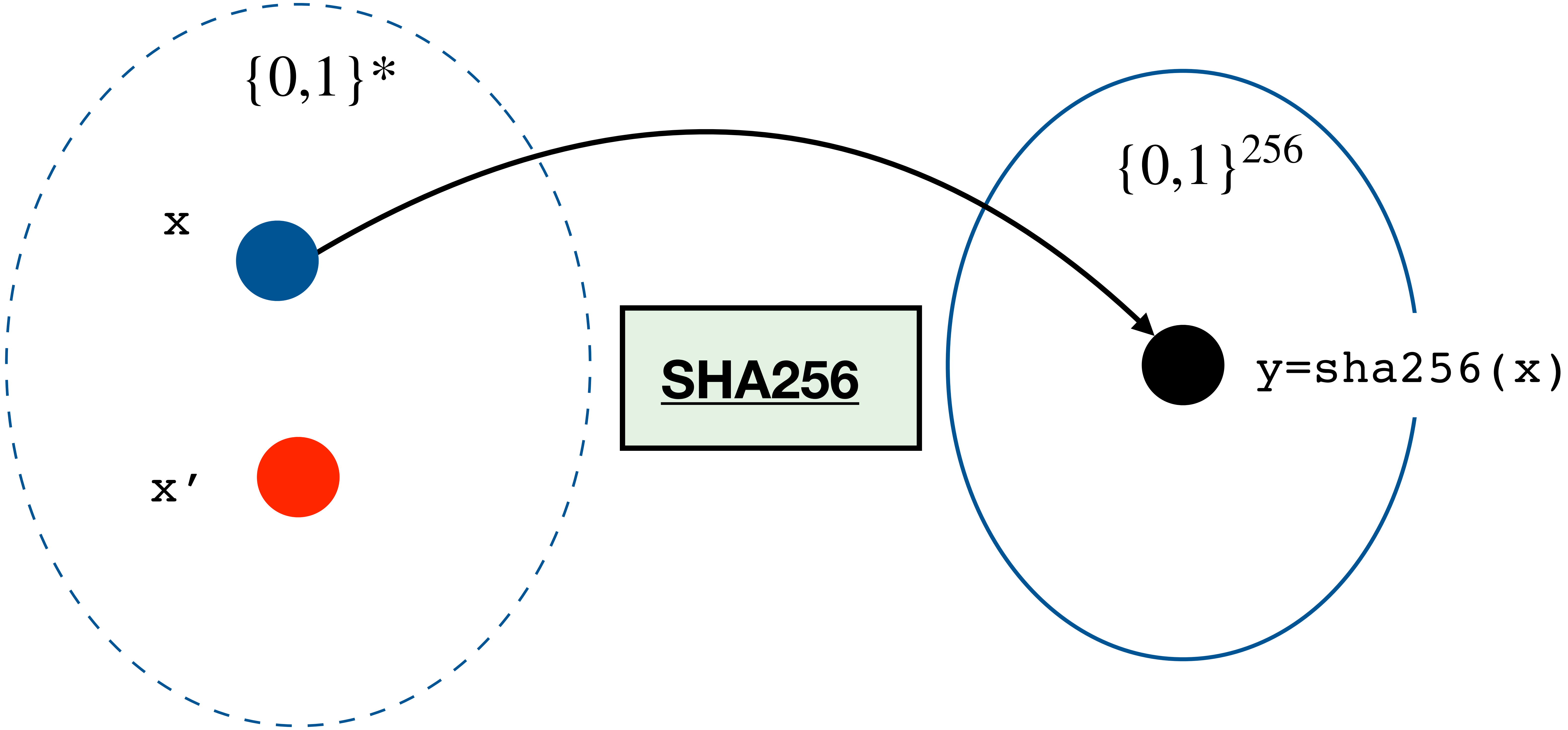
SHA-2 family



sha256(“T**D**A352”) = 3956a5541f782d61b7ca95e80496871e0d1f92a91b4836f65f21cc18e430ee86

sha256(“T**B**A352”) = 99d626fd9c74f8e7a1267ad7512ad13b92b841cdb11a0b132b1e43d8dfc80ed3

About SHA256



Preimage resistance attack:

$\mathscr{A}$ will eventually find x (given y) : it will take at most $2^{256} \approx 10^{78}$ trials

Collision resistance

$\mathscr{A}$ will eventually find x and x' that both hash to a y… and this is ***expected**** to take 2^{128} trials $\approx 10^{13}$ years on the world's fastest super computer

** By the birthday paradox, we will find out more about it in Module 2*

Classification of Hash Functions and Their Applications

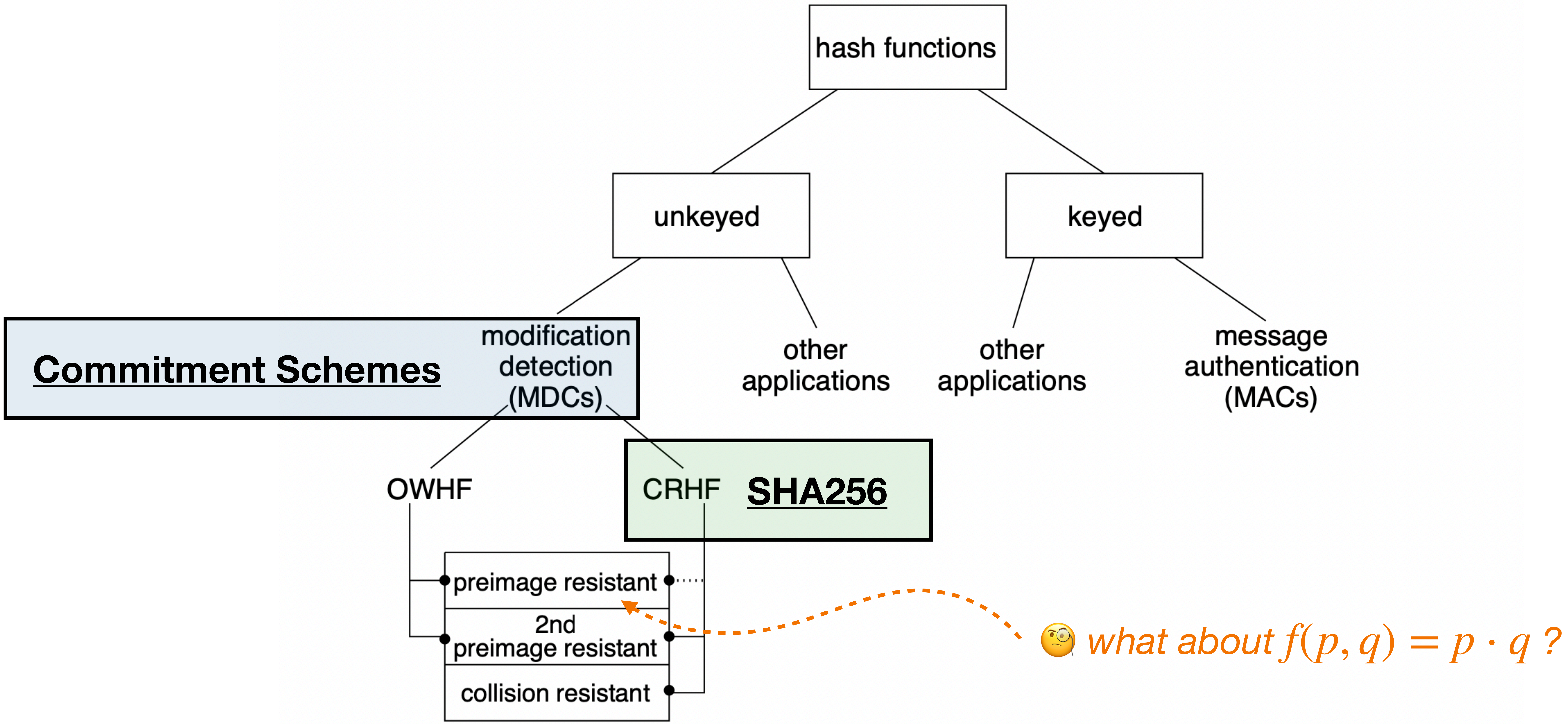


Figure 9.1: *Simplified classification of cryptographic hash functions and applications.*

OWF: an Important Security Note

OWF only guarantee that the input x is not leaked *entirely.* This means that it is still possible that $f(x)$ leaks a substantial amount of information about x.

Example:

Let $f : \{0,1\}^n \rightarrow \{0,1\}^n$ be a OWF.

Consider the function $g : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$ defined as $g(x_0 || x_1) := f(x_0) || x_1$.

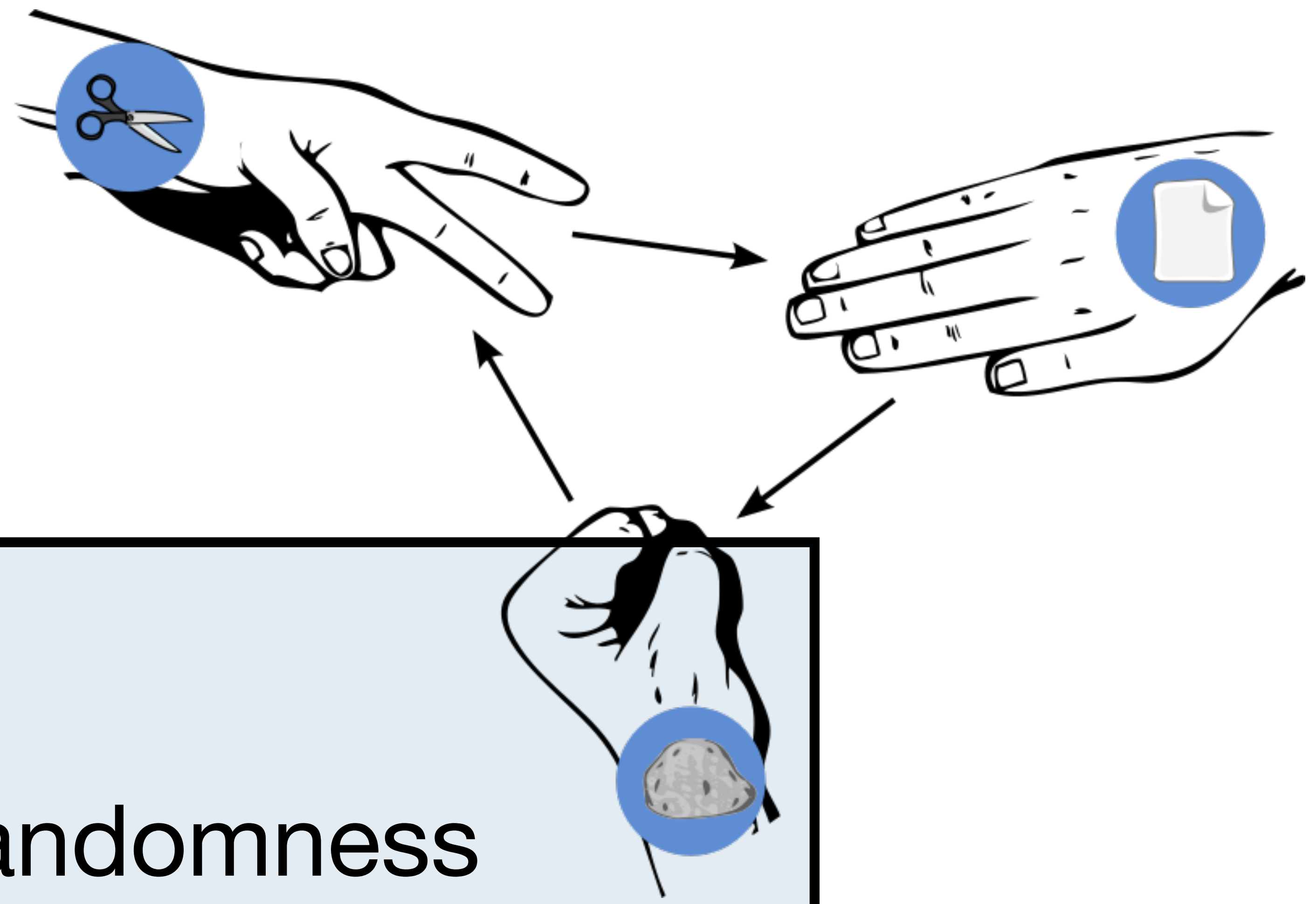
Even if $g()$ reveals half of its input, it is still a OWF! 🧐 Why?

Back to Commitment Schemes

Not in crypto:

Once you commit, you cannot change your mind!

Commitment Schemes Definitions



Syntax

A commitment scheme over a set of messages $\mathscr{M}$, a set of keys/randomness $\{0,1\}^n$ and a set of commit values C is defined by the two following PPT algorithms:

- Commit$(m, r) = c$ is a deterministic algorithm that takes in input a message m and a random string r; and outputs a commitment c to m.

- Open$(m, r, c) \in \{0,1\}$ this is a deterministic algorithm that takes in input a message m and a random string r, and a commitment c, and returns 1 (accept) if c is a valid commitment (for m, r); and 0 (reject) otherwise.

… and satisfying the **binding** and **hiding** properties (given next)

Commitment Schemes Definitions

Binding A commitment scheme is said to be **binding** if no adversary $\mathscr{A}$ can win the following game:

$\mathscr{A}$ must output two *distinct* messages $m, m^* \in \mathscr{M}$ and two keys $r, r^* \in \{0,1\}^n$ such that $m \neq m^*$ and Commit(m, r) = Commit(m^*, r^*).

$$Pr[\text{Commit}(m, r) = c = \text{Commit}(m^*, r^*) \,|\, m \neq m^*] \leq negl(n)$$

Commitment Schemes Definitions

Binding A commitment scheme is said to be information-theoretically (resp. *computationally*) **binding** if no infinitely powerful (resp. *computationally bounded*) adversary $\mathscr{A}$ can win the following game:

$\mathscr{A}$ must output two *distinct* messages $m, m^* \in \mathscr{M}$ and two keys $r, r^* \in \{0,1\}^n$ such that $m \neq m^*$ and $\text{Commit}(m, r) = \text{Commit}(m^*, r^*)$.

computational
(complexity-based)

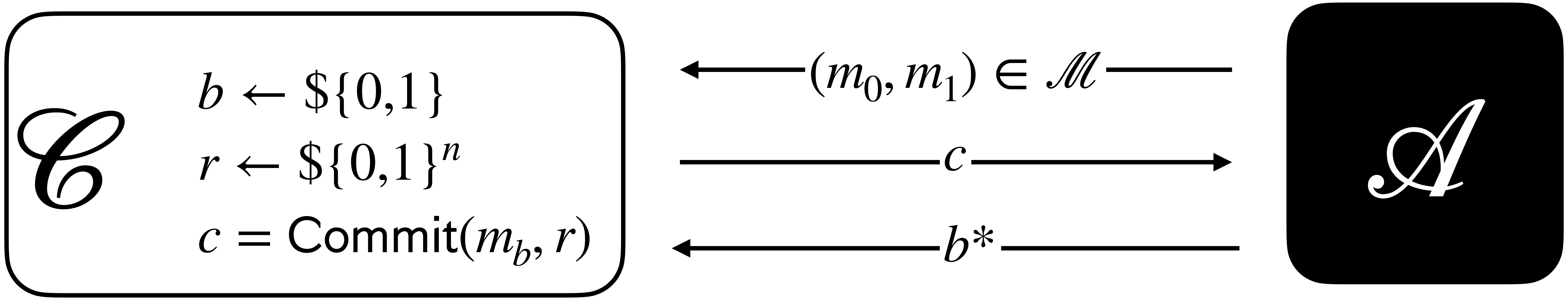
vs

information-theoretic
(unconditional)

Commitment Schemes Definitions

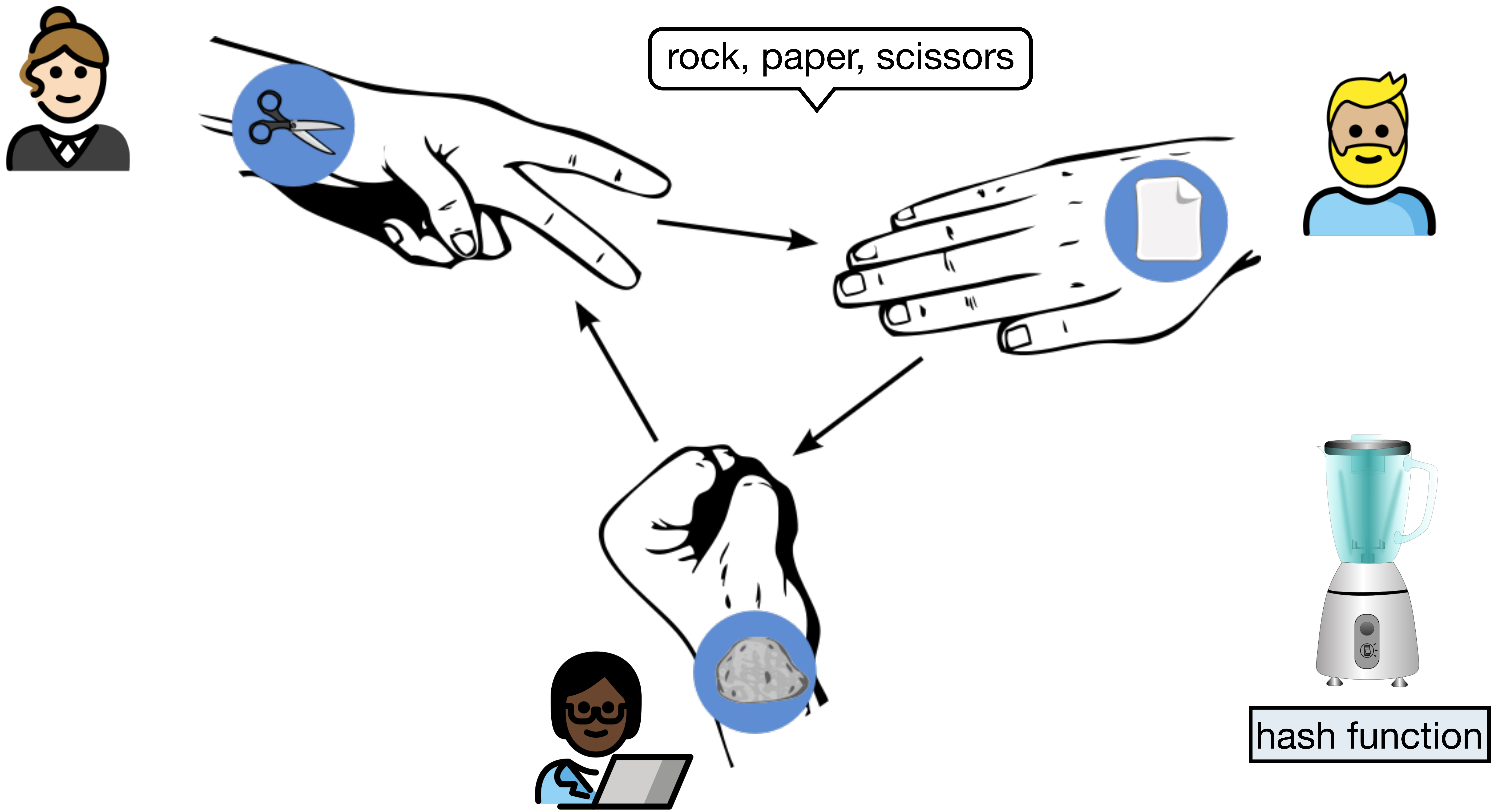
Hiding A commitment scheme is said to be information-theoretically (resp. *computationally*) **hiding** if no infinitely powerful (resp. *computationally bounded*) adversary can win the following game:

1. $\mathscr{A}$ outputs two messages m_0 and m_1.
2. $\mathscr{C}$ selects a random bit $b \leftarrow \$\{0,1\}$; picks a random $r \leftarrow \$\{0,1\}^n$; computes $c = \text{Commit}(m_b, r)$; and returns c to $\mathscr{A}$.
3. $\mathscr{A}$ outputs a bit b^* as a guess for b.



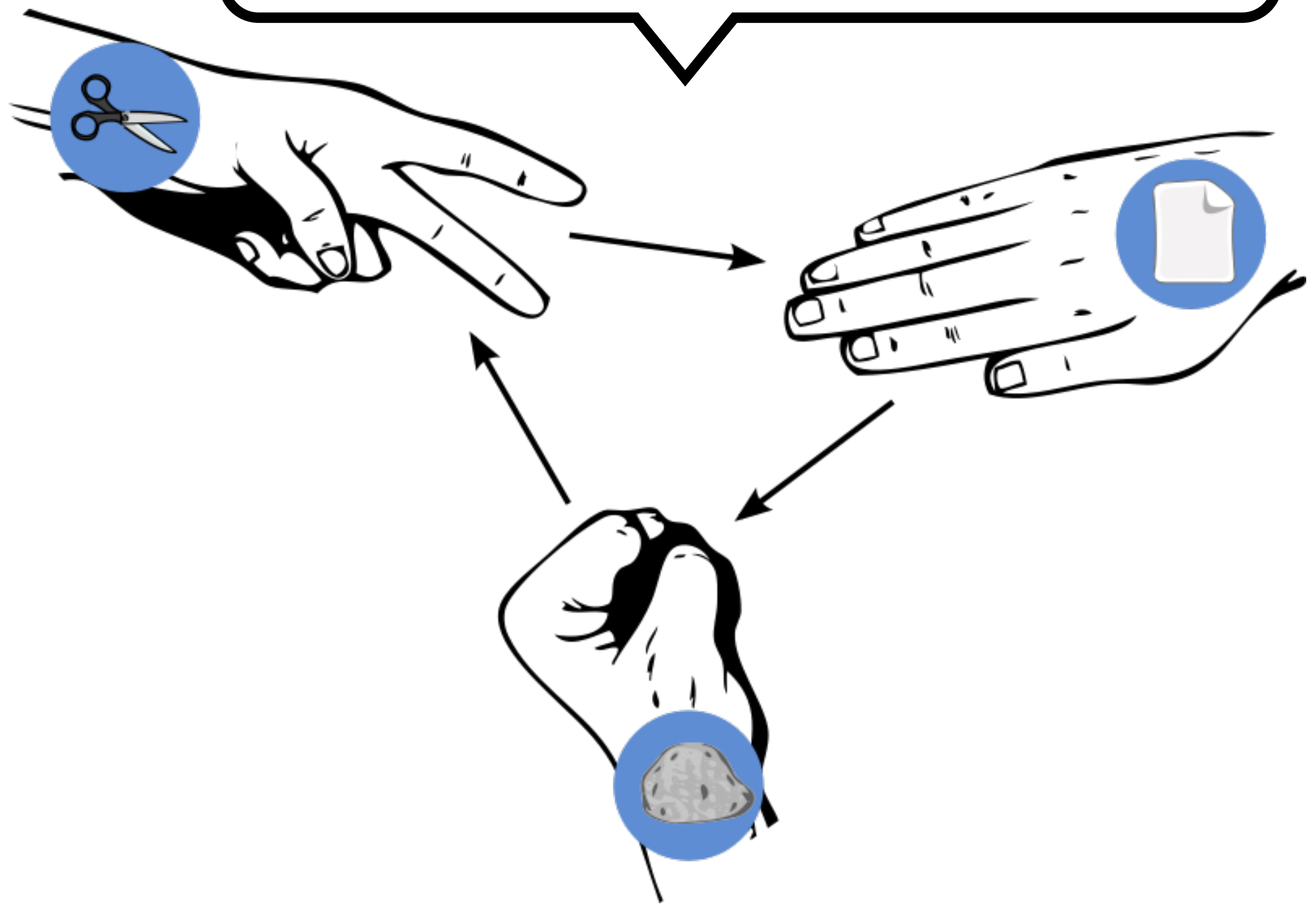
$$|Pr[b^* = b] - 1/2| \leq negl(n)$$

Let's Construct a Secure Commitment Scheme Using A Cryptographic Hash Function

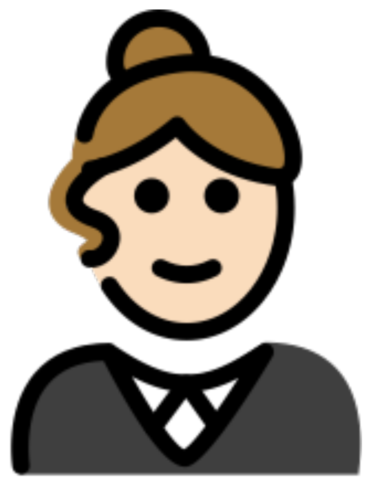


Commitment Schemes: a Simple Construction

rock, paper, scissors

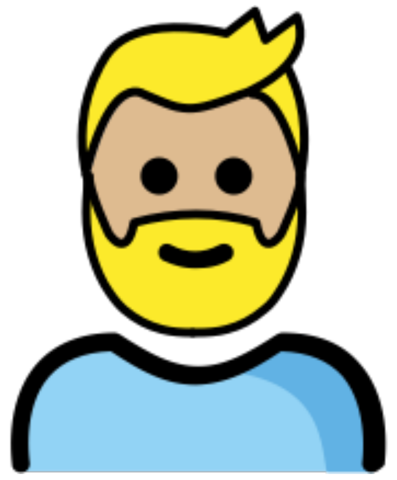


hash function

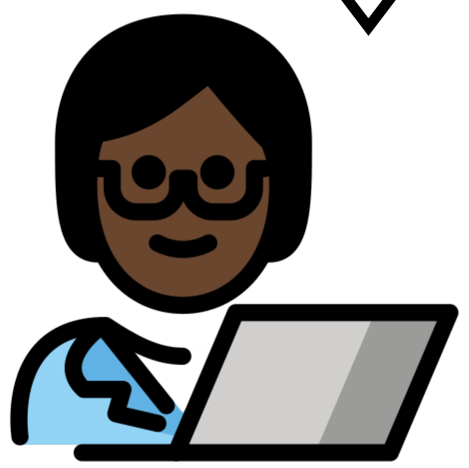


H(✂)

🧐 *what's the problem?*



H(✂ || r)



wait until everyone else ‘commits’,
and then ‘reveal’ r

r

A Hash-Based Commitment Scheme

$$\text{Commit}(m, r) = H(m||r) =: c$$
$$\text{Open}(m, r, c) = 1 \ \ if\ c = H(m||r); otherwise\ \ return\ 0$$

🧐 **Binding?** Yes!

$$Pr[\text{Commit}(m, r) = c = \text{Commit}(m^*, r^*) | m \neq m^*] \leq negl(n)$$

$$Pr[H(m||r) = H(m*||r^*) | m \neq m^*] \leq negl$$

second preimage resistance of the hash function H

🧐 **Hiding?** Yes!

$$|Pr[b^* = b] - 1/2| \leq negl(n)$$

$$Pr[b* = b | m_0, m_1, H(m_b||r)] \leq negl$$

preimage resistance of H

An Insecure Construction

$$\text{Commit}(m, r) = m + r =: c$$
$$\text{Open}(m, r, c) = 1 \ \textit{if} \ c = m + r; \ \ \textit{otherwise} \ 0$$

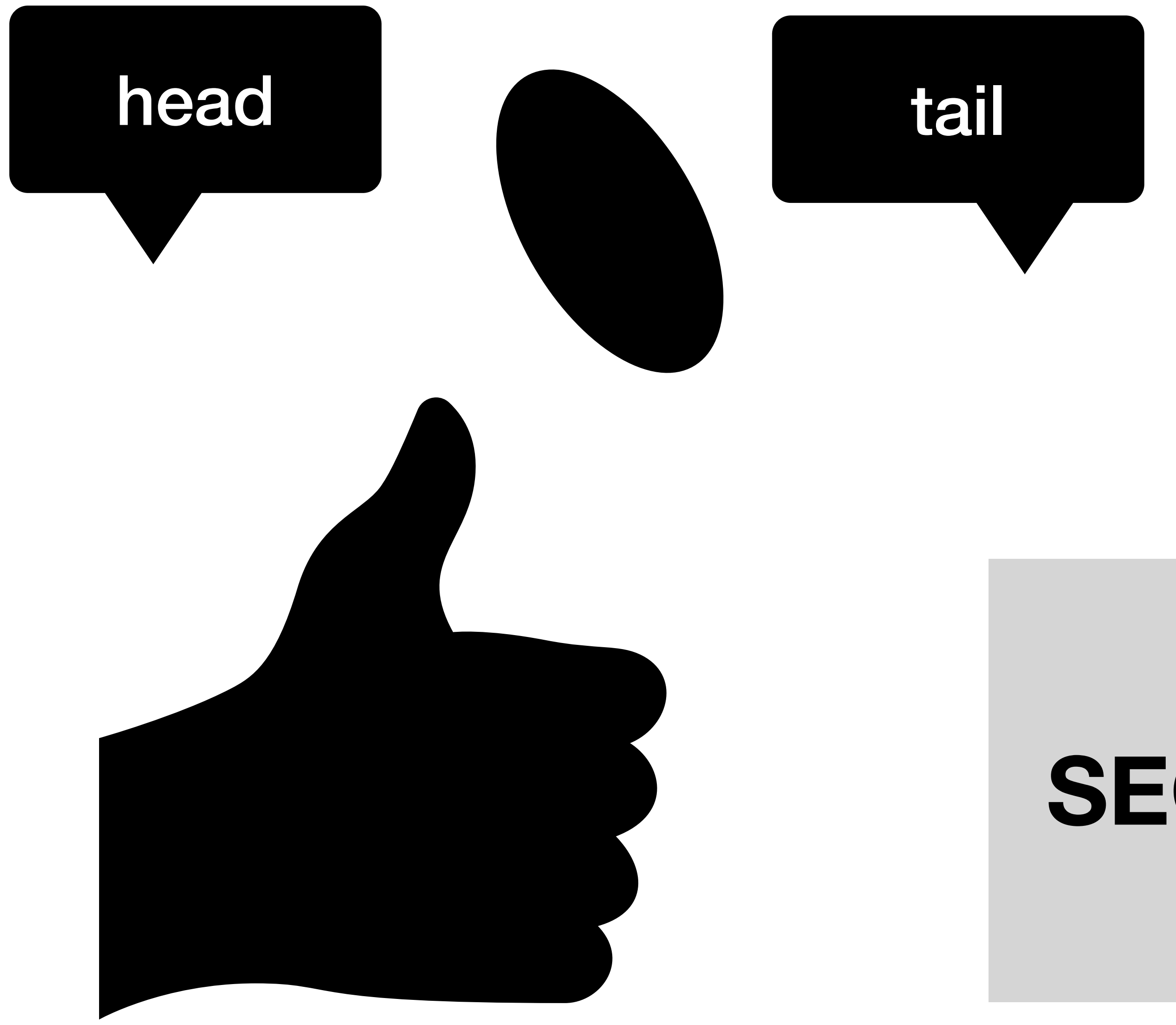
🧐 **Hiding?**

🧐 **Binding?**

There are plenty provable secure constructions of commitment schemes… we will see more in Module 3

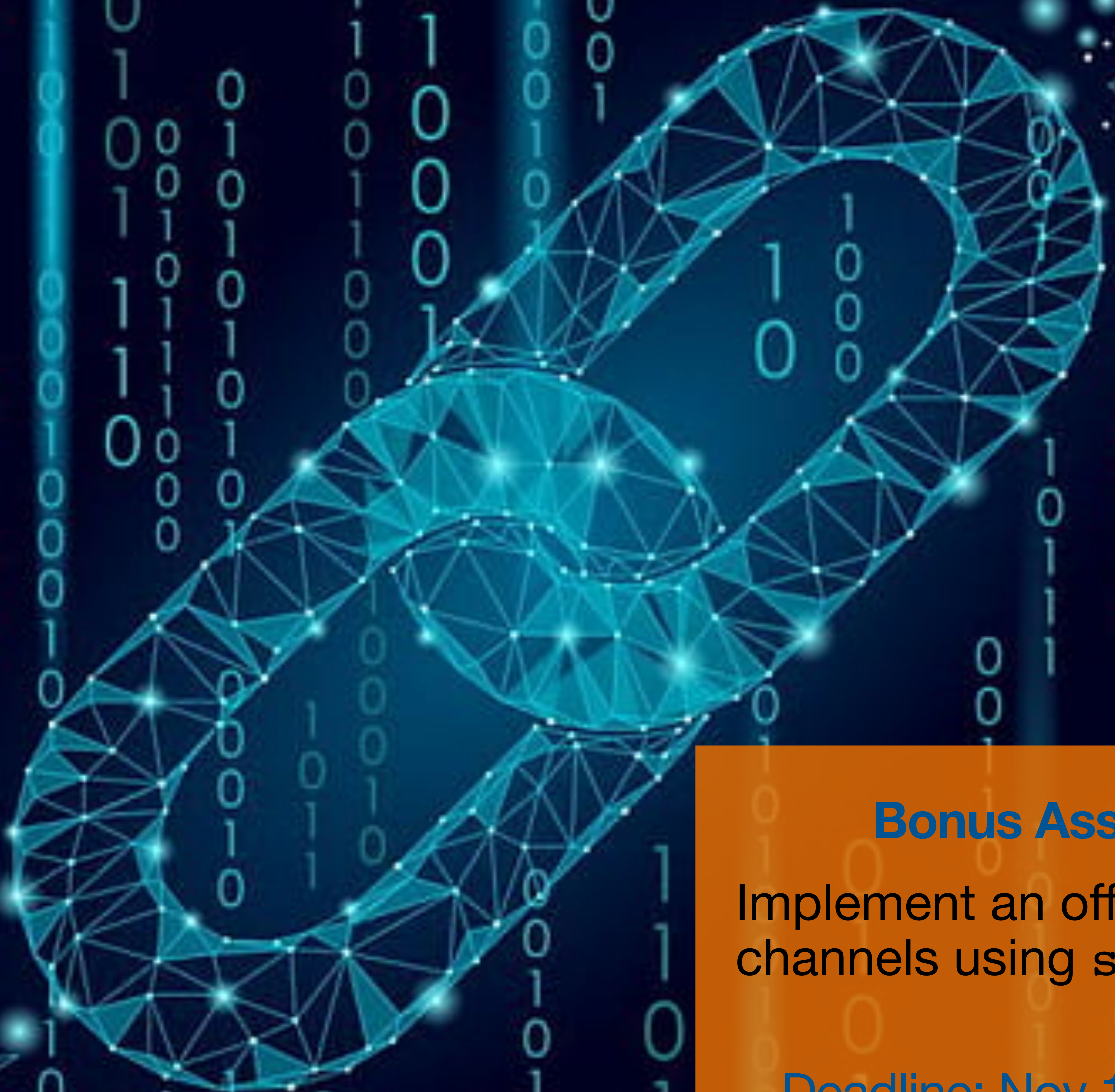
What Can You Do With Commitment Schemes?

PRIVATE AUCTIONS



SECURE COIN FLIPPING

Teaser for the Next Lecture



Bonus Assignment 1

Implement an off-chain payment channels using `solidity`

Deadline: Nov 18th

Blockchain Technology, Symmetric Encryption, Perfect Security