Zero Knowledge - Session 2

Elahe Sadeghi

October 2020

Symmetric Encryption/Private-Key Encryption

Symmetric Encryption/Private-Key Encryption

- The problem:

Symmetric Encryption/Private-Key Encryption

- Sol. use a shared key.

Alice

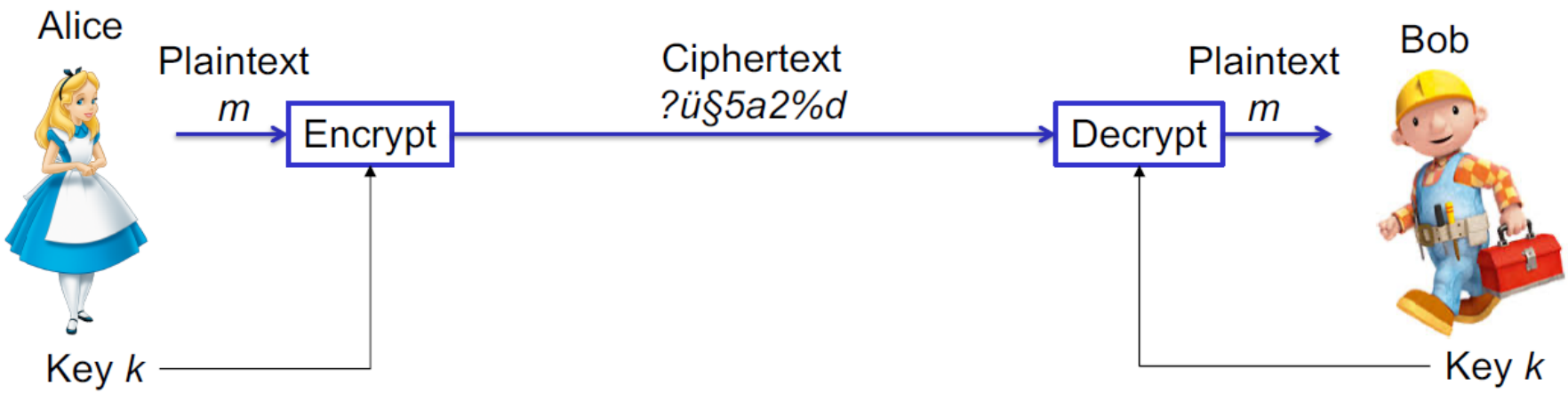
Key k

Bob

Key k

Symmetric Encryption/Private-Key Encryption

- using a private-key encryption scheme (Gen, Enc, Dec).



Symmetric Encryption/Private-Key Encryption

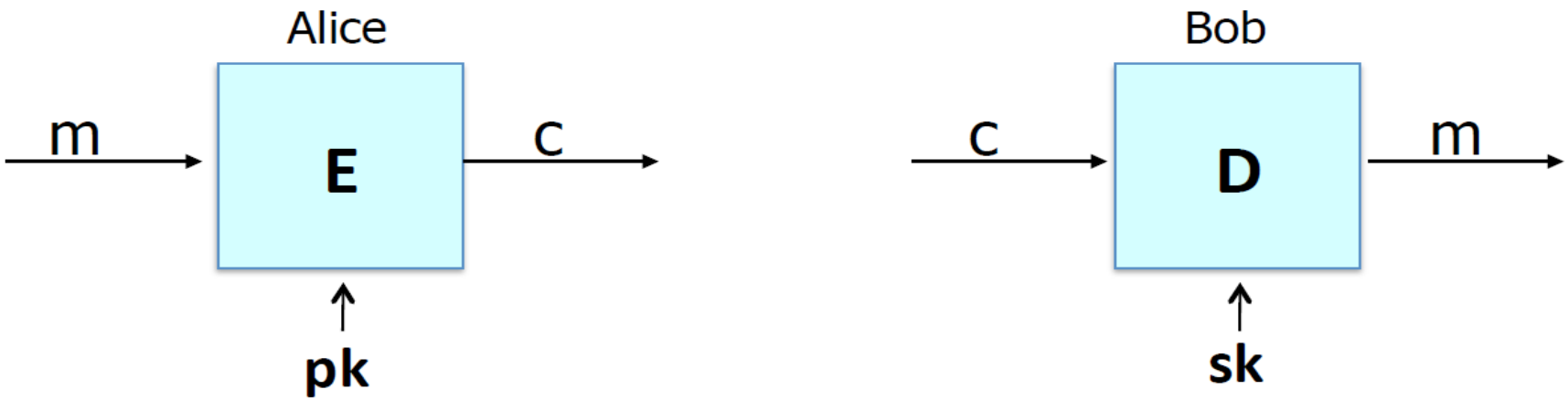
- **Correctness**: For all k and m, $\Pr[\mathrm{Dec}(k, \mathrm{Enc}(k, m)) = m] = 1$ holds.
- **Security**: The adversary should not obtain any non-trivial information about the encrypted message. ("semantic security")

Kerckhoffs' Principle

- A cryptosystem should be secure even if **all details** about the system are **public** except the key.

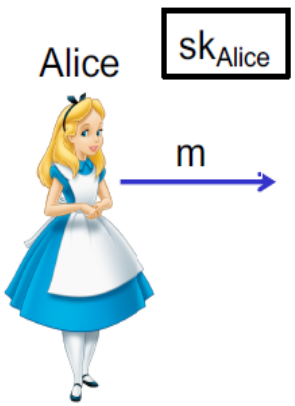
Asymmetric Encryption/Public-Key Encryption

- Bob generates (pk, sk) and gives pk to Alice.



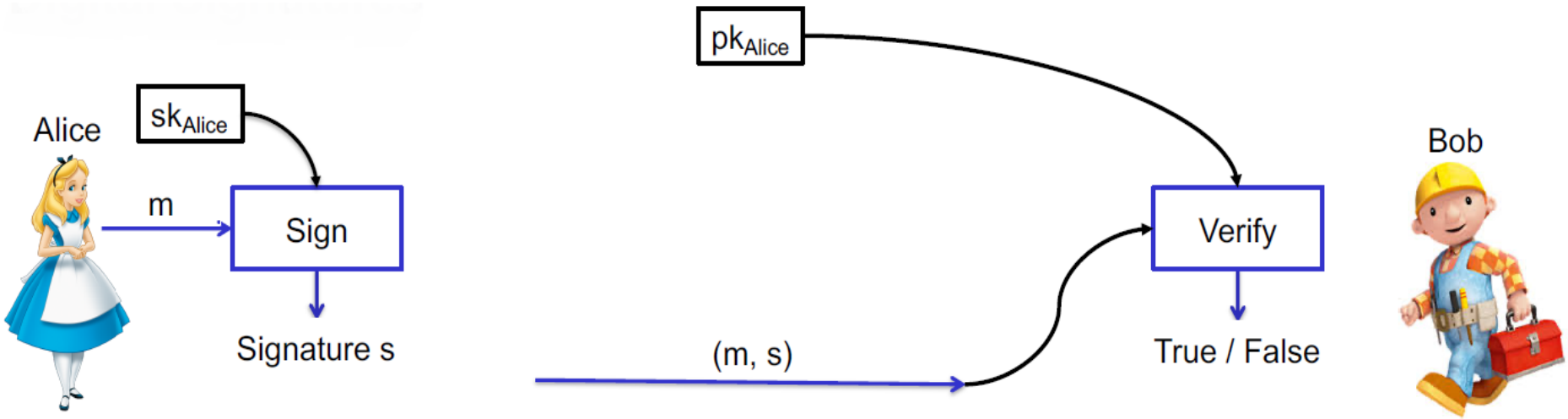
Digital Signatures

- Alice publishes pk_{Alice}, only she knows the matching secret key sk_{Alice}.



pk_{Alice}

Digital Signatures

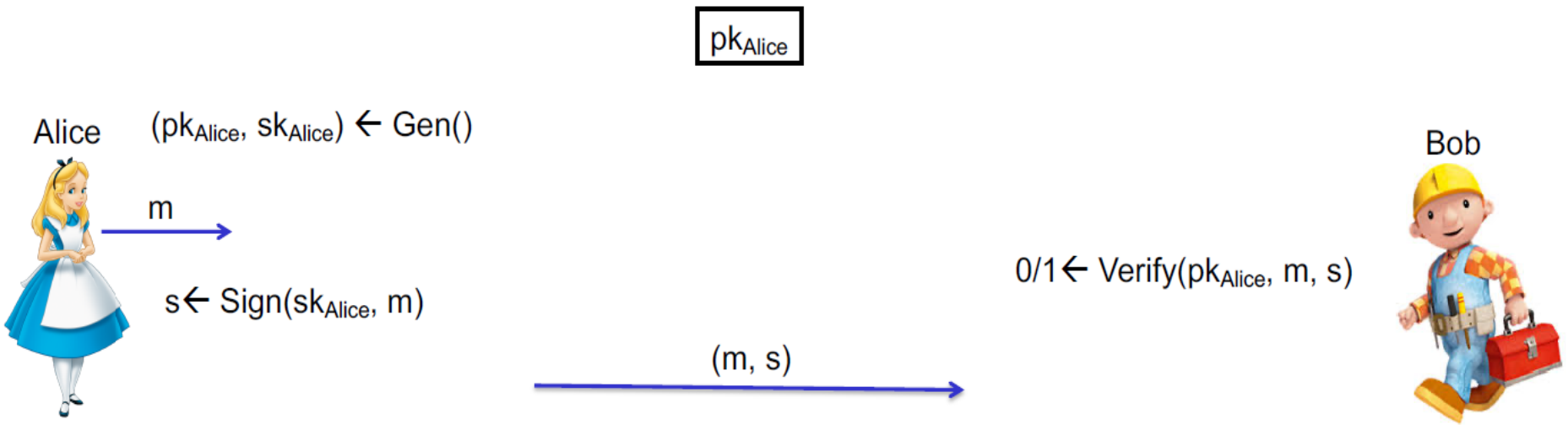


Digital Signatures

Digital Signature

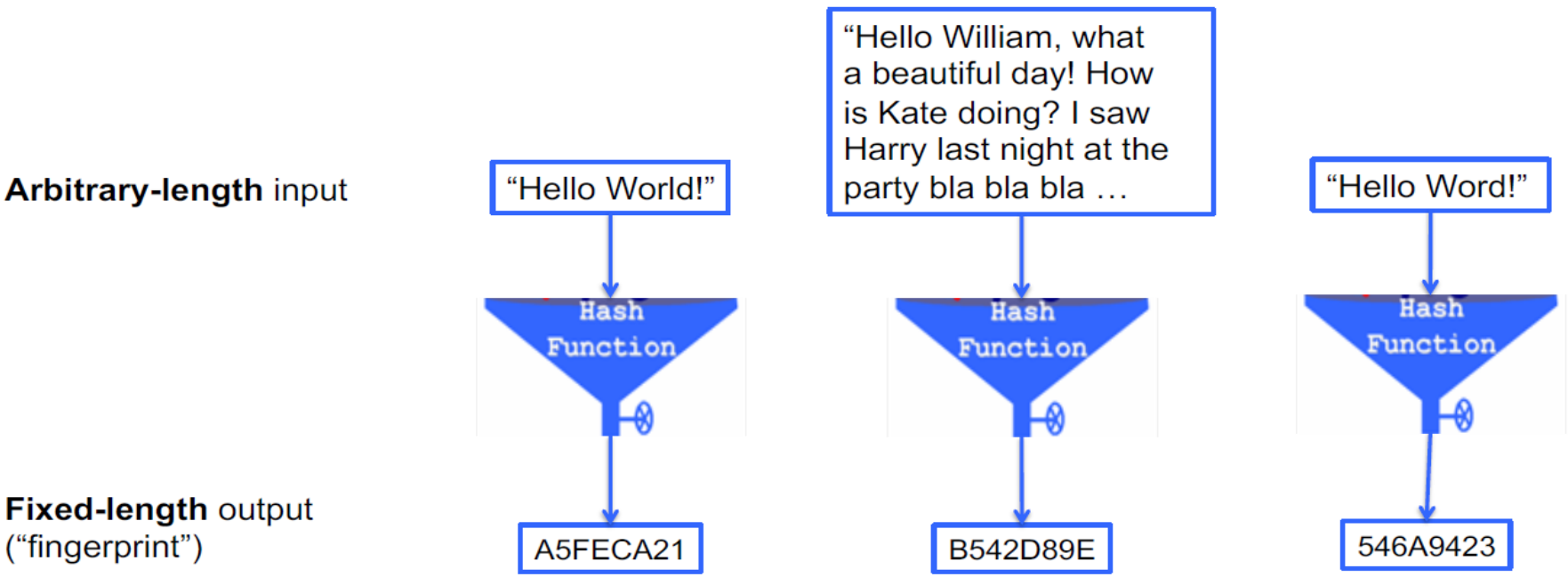
A digital signature scheme consists of three algorithms (Gen, Sign, Verify).

- **Correctness**: For all m and all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}()$, $\Pr[\mathsf{Verify}(\mathsf{pk}, m, \mathsf{Sign}) = 1] = 1$ holds.
- **Security**(intuitively): No adversary should not be able to create a valid signature, given only pk but not sk.



Cryptographic Hash Functions

- A cryptographic hash function is a function $H : \{0,1\}^* \rightarrow \{0,1\}^n$.



Cryptographic Hash Functions

- Security of Hash Functions:

Preimage resistance:

– **Pre-image resistance**: Given $y = H(x)$, one should not be able to find the x.

Cryptographic Hash Functions

- Security of Hash Functions:

Preimage resistance:

Second-preimage resistance:

– **Second pre-image resistance**: Given x, one should not be able to find a x' such that $H(x) = H(x')$.

Cryptographic Hash Functions

- Security of Hash Functions:

Preimage resistance:

Second-preimage resistance:

Collision resistance:

– **Collision resistance**: One should not be able to find a x and x' such that $H(x) = H(x')$.

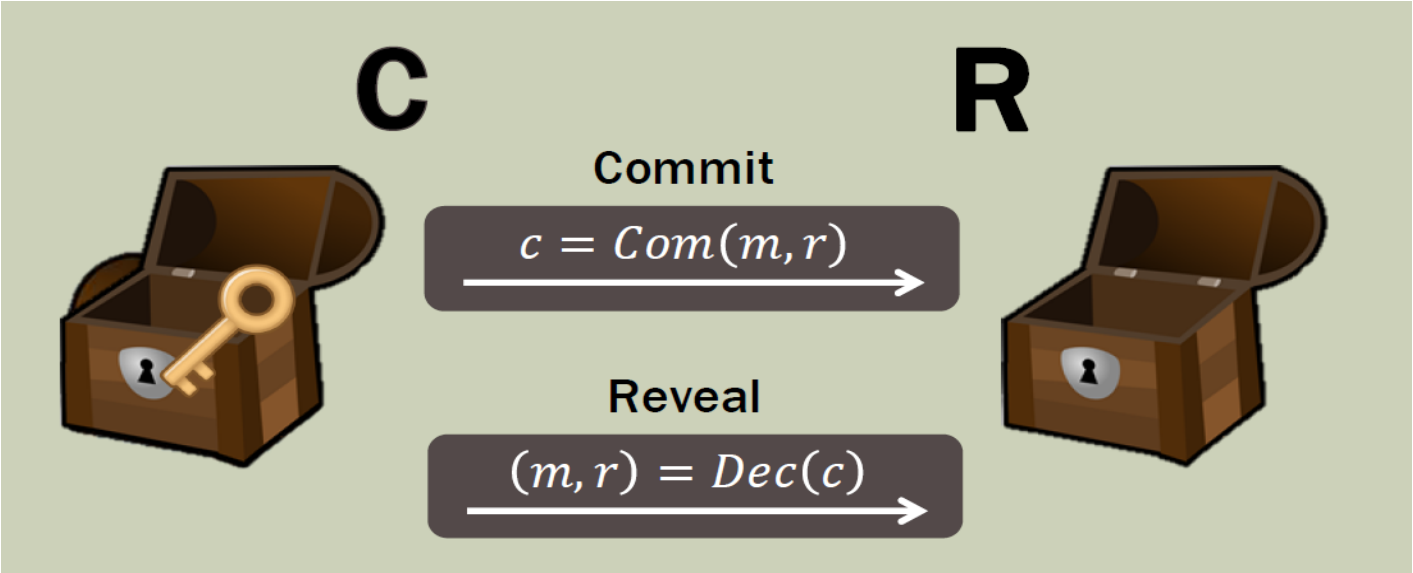
Cryptographic Hash Functions

- Popular Examples of Cryptographic Hash Functions:

Name	Output Length	Comment
MD5	128 b	• Considered **broken** • It is very easy to find collisions (1, 2)
SHA-1	160 b	• Considered **broken** • Possible to find collisions (3)
SHA-2	224 b, 256 b, 384 b, 512 b	• No practical attacks, but minor weaknesses (4) • Widely used, e. g., SHA-256 in Bitcoin's Proof-of-Work
SHA-3	224 b, 256 b, 384 b, 512 b	• No known attacks • Winner of SHA-3 competition • **Good choice for new applications**

Commitment Schemes

- A two stage protocol between Committer and Receiver, (Com, Dec).



  - **Completeness**: C can always generate valid $c = Com(m, r)$.

∗There is two kinds of commitment schemes:

1. statistically-binding
2. statistically-hiding

Commitment Schemes

A commitment scheme (Com, Dec) consists of two protocols:

- **Commit Phase**: A protocol in which the committer commits to a value m by computing $c = Com(m, r)$, where r is some randomness, and sending c to the receiver.
- **Reveal Phase**: A protocol in which the committer opens the commitment $c = Com(m, r)$ by sending m and r to the receiver. The receiver computes $Com(m, r)$ and verifies that it is equal to the previously received commitment.

The scheme should satisfy following properties:

- **Binding**: For any adversary $\mathcal{A}$, the probability of generating r, r', m, m' satisfying $Com(m, r) = Com(m', r')$ should be negligible.
- **Hiding**: For any $m \neq m', r, r'$, the distributions of $Com(m, r)$ and $Com(m', r')$ should be indistinguishable.

Commitment Schemes

Statistically-Binding Commitment Schemes

A statistically-binding commitment scheme (Com, Dec) satisfies:

- **Statistically-Binding**: For any unbounded adversary $\mathcal{A}$, the probability of generating r, r', m, m' satisfying $Com(m, r) = Com(m', r')$ should be negligible.
- **Computationally-Hiding**: For any $m \neq m', r, r'$, the distributions of $Com(m, r)$ and $Com(m', r')$ are computationally indistinguishable.

Statistically-Hiding Commitment Schemes

A statistically-binding commitment scheme (Com, Dec) satisfies:

- **Computationally-Binding**: For any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the probability of generating r, r', m, m' satisfying $Com(m, r) = Com(m', r')$ should be negligible.
- **Statistically-Hiding**: For any $m \neq m', r, r'$, the distributions of $Com(m, r)$ and $Com(m', r')$ are statistically indistinguishable.

A little bit of Complexity Theory :)

Decision Problems/Languages

An important special case of functions mapping strings to strings is the case of Boolean functions, whose output is a single bit. We identify such a function f with the subset $L_f = \{x : f(x) = 1\}$ for $\{0,1\}^*$ and call such sets languages or decision problems.

- A complexity class is a set of functions that can be computed within given resource bounds.

A little bit of Complexity Theory :)

The Class P

A language L is in P if there is a polynomial-time algorithm $\mathcal{A}$ such that $L = \{x \mid \mathcal{A}(x) = \text{accept}\}$.

- Bounded-error Probabilistic Polynomial-time (BPP) is the class of decision problems solvable by a probabilistic Turing machine in polynomial time with an error probability bounded away from $1/3$ ($\leq 1/3$) for all instances.

A little bit of Complexity Theory :)

The Class NP

A language L is in NP if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time Turing Machine $\mathcal{M}$ (called the verifier for L) such that for every x,

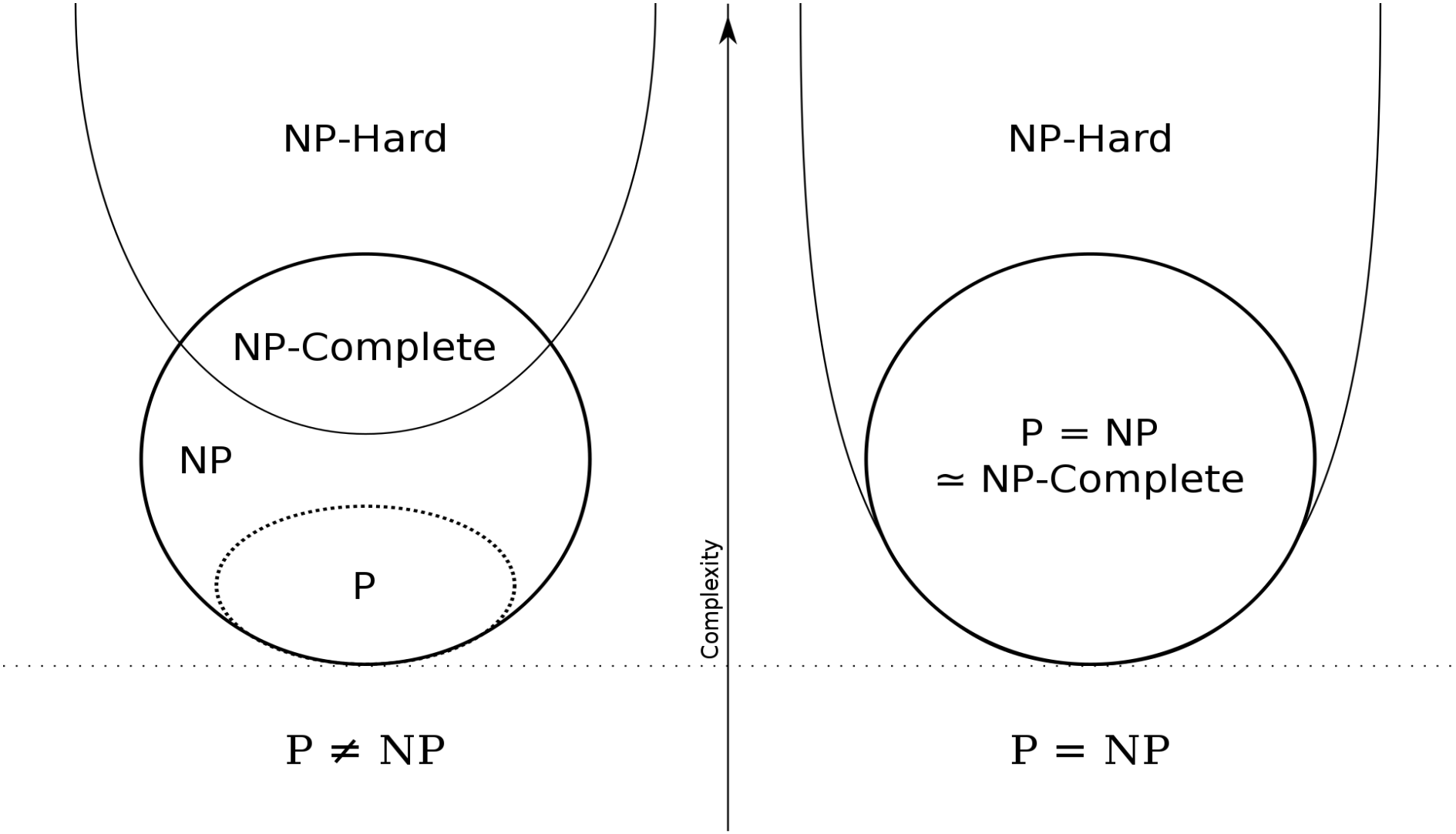
$$x \in L \iff \exists u \in \{0,1\}^{p(|x|)} \text{ s.t. } \mathcal{M}(x,u) = 1$$

If $x \in L$ and $u \in \{0,1\}^{p(|x|)}$ satisfy $\mathcal{M}(x,u) = 1$, then we call u a **certificate/witness** for x.

Reductions, NP-hardness, NP-completeness

A language L is polynomial-time reducible to a language L', denoted by $L \leq_p L'$, if there is a polynomial-time computable function f such that for every x, $x \in L$ if and only if $f(x) \in L'$.
We say that L' is NP-hard if $L \leq_p L'$ for every $L \in$ NP. We say that L' is NP-complete if L' is NP-hard and $L' \in$ NP.

A little bit of Complexity Theory :)



Some Helpful References

- Introduction to Modern Cryptography, Katz-Lindell
- Lecture Notes Cryptographic Protocols, Berry Schoenmakers
- Computational Complexity: A Modern Approach, Barak-Arora

Thank you!