The ABC of Composable Security in Cryptography

Mariano José Lemus Hernández

Instituto de Telecomunicações - Aveiro
January 8, 2025

Motivation: Defining Security in Cryptography

- Cryptographic protocols often arise from informal descriptions of communication tasks.

“A wants to send a message to B without C knowing what the message is”

- Provable security generally requires quantifying its security features.

“How do we measure how much of the message C knows?”

- More than one way of mathematically modeling protocols, information, and filling the details left from such descriptions.

“What is acceptable if the protocol fails?”
“What is acceptable if later C finds out half of the message?”

- The security of a protocol as a measure of how well it “does its job”.

A Classification of Security Definitions

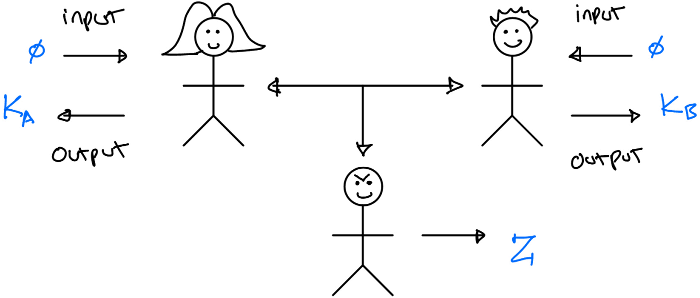
We can classify modern security definitions as follows:

- Stand-alone security – as list of properties expressed in terms of guessing probability, mutual information, entropy, etc.
- Indistinguishability-based security – as a list of indistinguishability relations between variables of the protocol and their respective “ideal” outcomes.
- Simulation-based security – as a list of (implied) indistinguishability relations between the executions of the protocol and its respective ideal functionality.

An Example: Key Distribution

Informal statement of the communication task:

"Alice wants to share a random n-bit key k with Bob without Eve knowing what the key is"



An Example: Key Distribution – Stand-Alone Security

1. $k_A = k_B$, except with negligible probability in n (The key is shared)
2. The distribution of K_A is uniform in the set of n-bit strings (The key is random)
3. The accessible information $I_{\mathsf{acc}}(K_A : E)$ is negligible in n (Eve does not know the key)

$$I_{\mathsf{acc}}(K_A : E) = \max_{\mathcal{M}} I(K_A : Z) \tag{1}$$

An Example: Key Distribution – Stand-Alone Security

Consider the following quantum state with:

$$\rho_{XYE} = \frac{1}{2 \cdot 3^m} \sum_{\substack{x \in \{0,1\} \\ y \in \{1,2,3\}^m}} |x\rangle\langle x|_X \otimes |y\rangle\langle y|_Y \otimes \rho_E^{x,y} \tag{2}$$

with

$$\rho_E^{x,y} = \frac{1}{2^m} \left(I + (-1)^x \sigma_y \right). \tag{3}$$

It can be shown that $I_{\text{acc}}(XY : E) \leq \frac{2}{3}^{\frac{m}{2}}$. However, for a fixed value of y the states $\rho_E^{0,y}$ and $\rho_E^{1,y}$ are orthogonal.

An Example: Key Distribution – Indistinguishability-based security

Using the second approach, all security features can be combined into one statement

$$\rho_{XYE} \approx \frac{1}{2^n} \sum_{k \in \{0,1\}^n} |k\rangle\langle k|_{K_A} \otimes |k\rangle\langle k|_{K_B} \otimes \rho_E \tag{4}$$

In other words, the trace distance between both sides of Eq.(4) is negligible in n.

An Example: Key Distribution – Simulation-based security

Functionality $\mathcal{F}'_{\text{KD}}$

Parameters:

- Parties Alice and Bob, eavesdropper Eve.
- Size n of the output key.

1. Upon receiving the message (*send keys*) from Alice, sample uniformly $k \leftarrow \{0,1\}^n$, output k to Alice and Bob, and the message (*key shared*) to Eve and halt.

An Example: Key Distribution – Simulation-based security

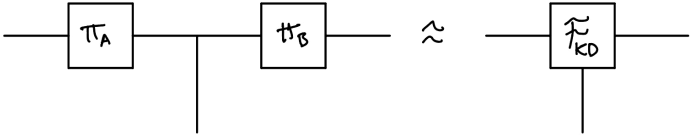


Figure: Emulation-based security statement: The ϕ_A, ϕ_B represent the local programs Alice and Bob run as part of executing the protocol and the wires represent communication channels.

More than one Flavor

Examples of simulation-based frameworks

- Universal Composability Framework (Canetti, 2001)
- Quantum Universal Composability Framework (Unruh, 2009)
- Abstract Cryptography Framework (Maurer, Renner, 2011)
- Simplified UC Framwork (Canetti, Cohen, Lindell, 2014)

Universal Composability Framework

- Main object of analysis are **Protocols**, which are understood as *algorithms* or *computer programs* written for a distributed system.
- A protocol consist of several separated programs called **Machines**:
  - Each program runs independently from the others and is able to send and receive messages to/from others
  - Each program has its own individual inputs/outputs

Machines and Protocols

- Formally, a machine is a triplet $\mu = (\text{Id}, C, \tilde{\mu})$, where
  - Id is the identifier of the machine within the communication network
  - C is a communication set; a set of communication channels with other machines within the network
  - $\tilde{\mu}$ is the program of the machine
- A protocol is a set of machines $\pi = (\mu_1, \ldots, \mu_n)$, satisfying a set of compatibility requirements. (Note: machines in a protocol may have communication channels to machines not in the protocol)
- Protocols may be parametrized by a security parameter k

Execution and Emulation

The model of execution for protocol π consists of the machines in π plus two additional special machines, called the environment $\mathcal{E}$ and the adversary $\mathcal{A}$:

- The environment $\mathcal{E}$ communication set allows it to provide inputs and receive outputs from the *external communication* channels of the machines in π, and to $\mathcal{A}$. Additionally, it has a single external channel for input/output. Its outputs are always binary.
- The adversary $\mathcal{A}$ communication set allows it receive backdoor information from *all* machines in π, who are augmented with an extra communication channel with $\mathcal{A}$.

The resulting set of machines can be understood as an associated protocol which can only receive inputs through $\mathcal{E}$.

Execution and Emulation

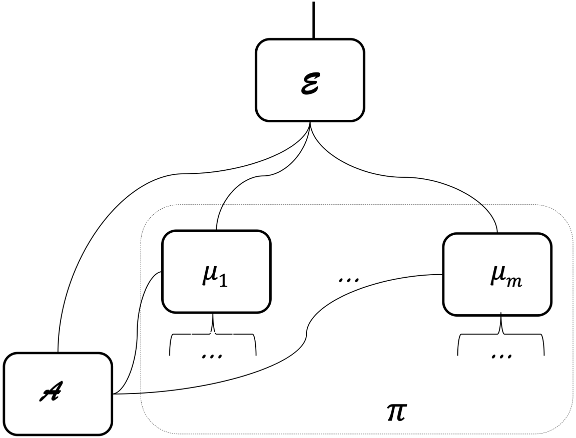


Figure: Diagram of a protocol execution. (Canetti, 2001)

Execution and Emulation

- Denote by $\text{EXEC}_{\pi,\mathcal{A},\mathcal{E}}(k, z)$ the random variable associated to the output of an execution of the joint programs of $\pi, \mathcal{A}, \mathcal{E}$ on input z and security parameter k.
- Denote by by $\text{EXEC}_{\pi,\mathcal{A},\mathcal{E}}(k)$ the ensemble $\{\text{EXEC}_{\pi,\mathcal{A},\mathcal{E}}(k, z)\}_{z\in\{0,1\}^*}$
- A protocol π **UC-emulates** a protocol ϕ if for any adversary $\mathcal{A}$, there exists an adversary $\mathcal{S}$, such that, for any environment $\mathcal{E}$, the ensembles $\text{EXEC}_{\pi,\mathcal{A},\mathcal{E}}(k)$ and $\text{EXEC}_{\phi,\mathcal{A},\mathcal{E}}(k)$ are indistinguishable in k.
- Statistical vs Computational security

Execution and Emulation

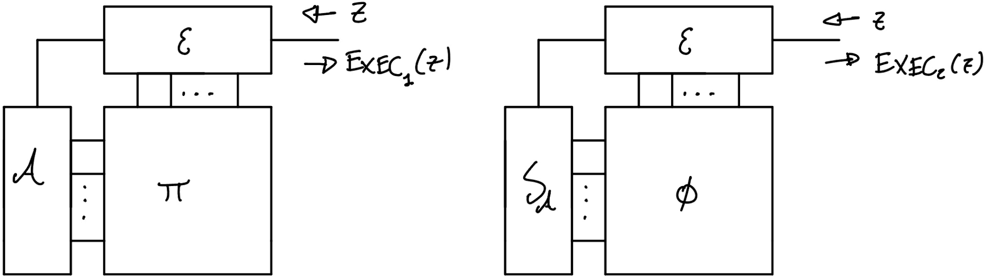


Figure: Execution of two protocols π and ϕ

Ideal Functionalities

- Ideal functionalities are understood as trusted machines that perform the desired task.
- The formalization of a cryptographic task is done by defining its respective ideal functionality.

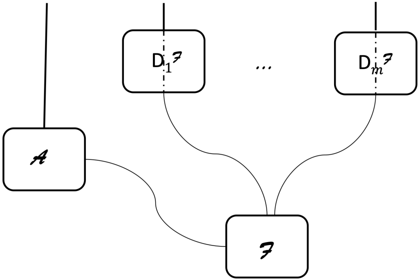


Figure: Protocol associated to an ideal functionality, $\text{IDEAL}_{\mathcal{F}}$. (Canetti, 2001)

Security in the UC framework

A protocol π UC-securely realizes an ideal functionality $\mathcal{F}$,
if π UC-emulates $\text{IDEAL}_{\mathcal{F}}$.

Universal Composition Theorem

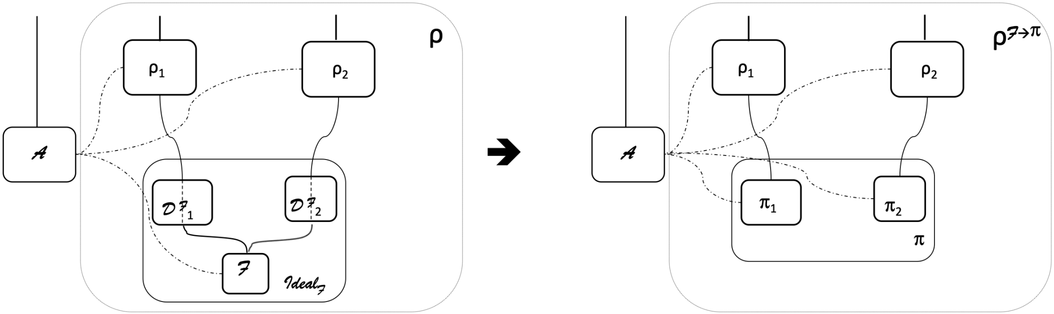


Figure: Universal composition operation. (Canetti, 2001)

Some additional features

- Party corruption – Introduced in the definition of the programs of each machine to allow for interaction with the Adversary.
- Hybrid models – Protocols can be assumed to have access to trusted ideal functionalities. Useful for finding reductions.
    - E.g. Random Oracle model, Public-key infrastructure model...

From Classical to Quantum

- Unruh's Quantum UC-security framework is a direct generalization of Canetti's
- It separates itself in the machine model and in the addition of quantum communication channels

Quantum lifting theorem
Let π and ϕ be classical protocols such that π statistically (classically) UC-emulates ϕ, then π statistically quantum UC-emulates ϕ.

MPC reduction to BC
It has been proven that Oblivious Transfer (and thus, secure multiparty computation) can be quantum UC-securely realized through a quantum protocol in the $\mathcal{F}_{\text{BC}}$-hybrid model, which is believed to be impossible classically.

An Example: Key Distribution – Simulation-based security

Functionality $\mathcal{F}'_{\text{KD}}$

Parameters:

- Parties Alice and Bob, eavesdropper Eve.
- Size n of the output key.

1. Upon receiving the message (*send keys*) from Alice, send the message (*keys requested*) to Eve.
2. Upon receiving m from Eve:
   - If $m =$ (*allow*), sample uniformly $k \leftarrow \{0,1\}^n$, output k to Alice and Bob and the message (*key shared*) to Eve and halt.
   - Else, output the message (*unable to send keys*) to Alice, Bob, Eve and halt.

Closing Thoughts

- Simulation security frameworks are powerful tools for abstracting and analyzing the security of cryptographic protocols
- Great security comes with great requirements, not all useful protocols need be UC-secure
- Cryptography is a game of trade-offs

References

- Canetti, Ran. ”Universally composable security: A new paradigm for cryptographic protocols.” Proceedings 42nd IEEE Symposium on Foundations of Computer Science. IEEE, 2001.
- Unruh, Dominique. ”Universally composable quantum multi-party computation.” Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
- Maurer, Ueli and Renato Renner. “Abstract Cryptography.” International Conference on Supercomputing, 2011.
- Renner, Renato. Security of quantum key distribution. International Journal of Quantum Information, 6(01), 1-127, 2008.
- König, Robert, Renato Renner, Andor Bariska, and Ueli Maurer. Small accessible quantum information does not imply security. Physical Review Letters, 98(14), 140502, 2007.
- Canetti, Ran, Asaf Cohen, and Yehuda Lindell. ”A simpler variant of universally composable security for standard multiparty computation.” Advances in Cryptology–CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015.

Thank you for your attention!