How to Define Security for Protocols?

Sven Laur
swen@math.ut.ee

University of Tartu

Primitives and protocols

Cryptographic primitives. Primitives are tailor-made constructions that have to preserve their security properties in very specific scenarios.

▷ IND-CPA cryptosystem is guaranteed to be secure only with respect to the simplistic games that define IND-CPA security.

▷ A binding commitment is secure only against double opening.

Cryptographic protocols. Protocols must preserve security under the wide range of conditions that are implicitly specified by security model.

▷ In stand-alone setting, the adversary can choose any "plausible" security objective and a corresponding attack strategy.

▷ Universally composable protocols must also handle implicit information flow that comes from a surrounding computational context.

Well-defined security goals

A security goal is well defined if for any attack strategy it is possible to determine whether it was successful based on externally observable events.

- ▷ We can formalise the security goal as a game phase $\mathcal{G}_{\text{re-atk}}$ where the challenger emulates behaviour of honest protocol participants.
- ▷ Given the resulting outputs $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_n, \psi_a)$, the challenger must decide whether the attack was successful by evaluating a predicate $\mathcal{B}(\boldsymbol{\psi})$.

Usually, one is interested in a certain subset $\mathfrak{B}$ of all security objectives. For example, we might be interested in effects that become visible in reasonable time. Then it is appropriate to consider all t_{pred}-time predicates, where t_{pred} is large enough to capture all feasible computations.

Ideal vs real world paradigm

Since a protocol must preserve security in wide range of conditions, we cannot prove security for each separate attack scenario.

- ▷ Instead, we formalise an ideal world model and a corresponding game phase $\mathcal{G}_{\text{id-atk}}$ that models ideal world execution.
- ▷ As a result, we can compare success of real and ideal world adversaries $\mathcal{A}$ and $\mathcal{A}^\circ$ wrt the security goal $\mathcal{B}(\cdot)$ and input distribution $\mathfrak{D}$.
- ▷ Let $\mathcal{G}_{\text{real}}$ and $\mathcal{G}_{\text{ideal}}$ be the corresponding real and ideal world games. Then we want that for any $\mathcal{B} \in \mathfrak{B}$ and for any t_{re}-time real world adversary there exists a t_{id}-time ideal world adversary $\mathcal{A}^\circ$ such that

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon$$

For the proof, we need a simulator $\mathcal{S}$ that implements a mapping $\mathcal{A}, \mathcal{B} \mapsto \mathcal{A}^\circ$.

Canonical game description

There are many ways how to represent real and ideal world environments as security games, but all of them formalise the same process.

- ▷ The challenger simulates the execution of all computations.
- ▷ The challenger creates an illusion of a true attack to the adversary.
  - ◇ The adversary "sees" an interaction between protocol participants although all computations are done by the challenger.
- ▷ An adversary can issue corruption requests.
  - ◇ Then challenger reveals the internal state of the participant $\mathcal{P}_i$.
  - ◇ If $\mathcal{P}_i$ is maliciously corrupted then the control over $\mathcal{P}_i$ to the adversary.
  - ◇ If $\mathcal{P}_i$ is semi-honestly corrupted then the adversary can only observe the internal state.

The devil is in the details

Exact properties of the simulator construction $\mathcal{A}, \mathcal{B} \overset{\mathcal{S}}{\mapsto} \mathcal{A}^\circ$ depend on

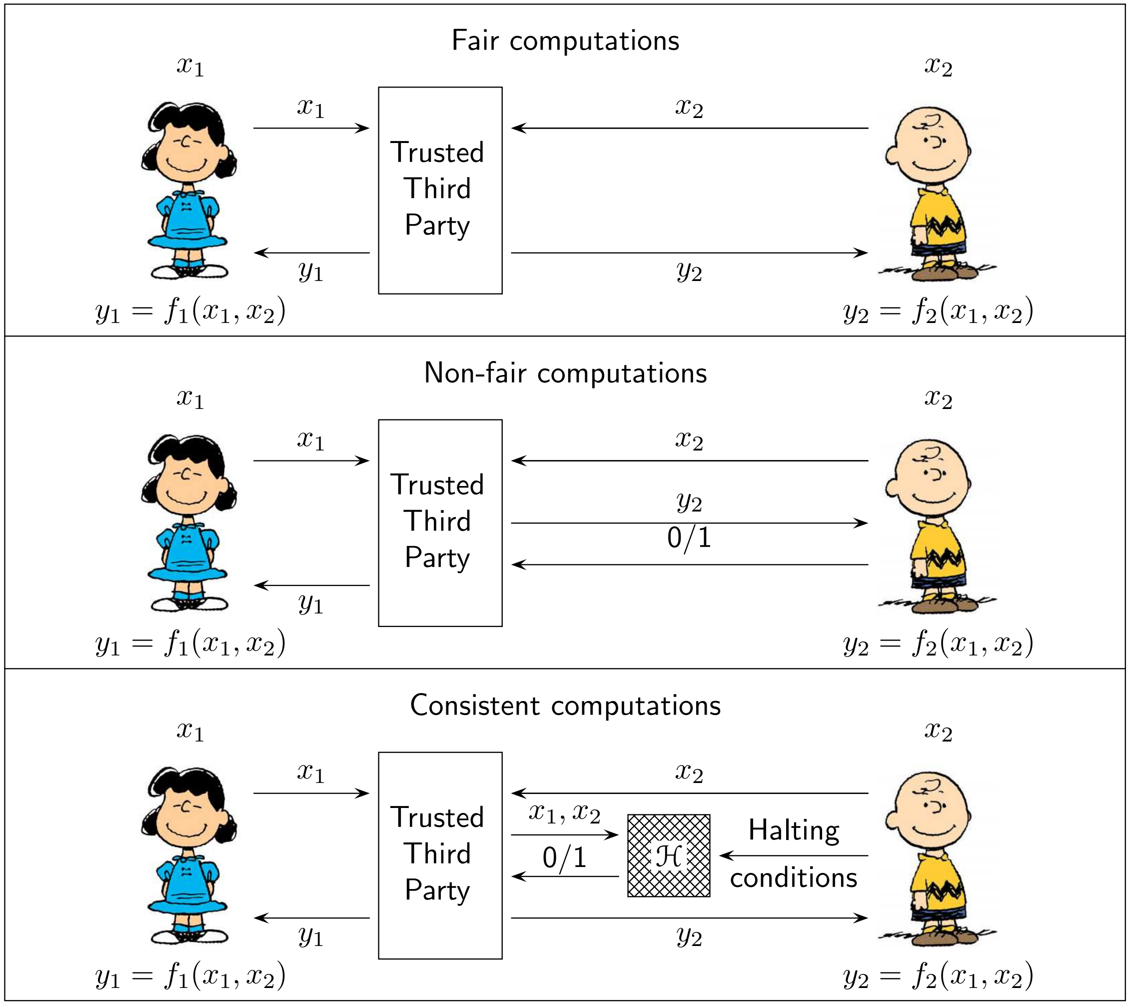
▷ **Quantitative properties:**
- ◇ How does $\varepsilon = \varepsilon(t_{\mathrm{re}}, t_{\mathrm{pred}})$ behave?
- ◇ How comparable is $t_{\mathrm{id}} = t_{\mathrm{id}}(t_{\mathrm{re}}, t_{\mathrm{pred}})$ with t_{re}?
- ◇ How large values of t_{pred} lead to reasonable values of t_{id} and ε?

▷ **Qualitative properties:**
- ◇ What is the tolerated adversarial behaviour?
- ◇ Which model is used for idealised computations?
- ◇ What type of predicates $\mathcal{B}(\cdot)$ are considered relevant?
- ◇ In which context the protocol is executed?

Taxonomy of security levels

	Input privacy	Output consistency	Complete security
Malicious behaviour	Privacy of inputs is guaranteed even against malicious behaviour.	Malicious behaviour that alters outputs is detectable. Public complaints reveal information about inputs.	Malicious behaviour is detectable. Public complaints reveal no information about inputs.
Semi-honest behaviour	Privacy of inputs is guaranteed. Privacy of outputs is not guaranteed.	Protocols implement the desired functionality.	Privacy of inputs and outputs is guaranteed. Protocols implement the desired functionality.

Taxonomy of ideal world models



Taxonomy of corruption models

Static corruption model. An adversary must choose which participants to corrupt before the execution of the computations.

▷ This model is adequate for small well-protected networks.

Dynamic corruption model. An adversary can choose which participants to corrupt during the execution of the computations.

▷ This model is adequate for larger networks provided that the protocol is executed in a relatively short time-frame.

Mobile corruption model. An adversary can corrupt participants and withdraw from them during the execution of the computations.

▷ This model is adequate for protocols that have a long life span.

Taxonomy of tolerated contexts (1/2)

Stand-alone security. Security of a protocol is considered in the setting where no other computations are carried out.

- ▷ The stand-alone setting is simple enough to analyse in practice.
- ▷ Security in more complex settings is often characterised through the stand-alone model by imposing additional constraints.

Sequential composability. A protocol is sequentially composable if it preserves security in the computational contexts where

- ▷ some computations are done before the protocol,
- ▷ some computations will be done after the protocol,
- ▷ no side computations are carried out during the protocol,
- ▷ the beginning and end of the protocol is clearly detectable for all parties.

Taxonomy of tolerated contexts (2/2)

Composability wrt specific contexts. In many cases, it is advantageous to use sub-protocols in order to implement complex functionality.

▷ The compound protocol is secure only if the sub-protocols preserve their security in the context induced by the compound protocol.

▷ As a result, we must either prove security directly for the compound protocol or show that the security is preserved in this context.

As an example, consider parallel self-composability of zero-knowledge proofs.

Universal composability. A protocol is universally composable if it preserves security in all contexts that use protocol as a black box:

▷ the context provides inputs and fresh randomness,

▷ the context uses only the outputs of the protocol.

Taxonomy of other taxonomies

A communication model specifies how messages are transferred between participants and how an adversary can influence message transmission.

An execution and timing model specifies how the participants carry out their computations and whether the adversary can use timing information.

Setup assumptions characterise how system wide parameters are generated. There are a wide spectrum of setup assumptions:

▷ plain model

▷ common reference string model

▷ public key infrastructure model

Finally, there is wide spectrum of models that describe possible side-channel attacks, e.g., power analysis, spectral analysis, hardware tampering, etc.