Universal Composability (UC)

An Introduction

Julia Hesse
Reality vs. Dreamland
Formalizing the two worlds

Real execution
- real users modeled as ITMs
- execute real protocol
- adversary A controls communication
- A-leakage/influence possible

Ideal execution
- dummy parties (only relays), no communication
- trusted party F computes outputs

The Framework

Universal Composability (UC)
Communication model

Make your choice:
- asynchronous
- synchronous
- with guaranteed delivery
- with authentication
- ...

Most results: asynchronous with authentication
Indistinguishability

The Framework

Universal Composability (UC)
Indistinguishability
Indistinguishability

Protocol "UC-realizes" ideal functionality:

∀ \text{view} \in (A) \sim \text{view} \in (A)
Indistinguishability

Protocol "UC-realizes" ideal functionality:

\[ \forall \mathcal{A} \quad \mathcal{A} \text{ view} (\mathcal{A}) \sim \text{ view} (\mathcal{A}) \]
Attacks

What can the environment do to distinguish?

- decide about inputs
- corrupt a player (learn state, control it, via A)
- tamper with transcript (via A)
Attacks

What can the environment do to distinguish?

- decide about inputs
- corrupt a player (learn state, control it, via A)
- tamper with transcript (via A)
Attacks

What can the environment do to distinguish?

- decide about inputs
- corrupt a player (learn state, control it, via A)
- tamper with transcript (via A)
Hybrid model

UC supports modular protocol design

Users get access to trusted third parties (unbounded number)
Composition Theorem

\[ \rho \text{ UC-realizes } F \quad \land \quad \pi \text{ uses } F \]

\[ \implies \pi^F \approx \pi^\rho \]

* an unbounded number of copies of F \textit{concurrently}
Composition Theorem

\[ \rho \text{ UC-realizes } F \quad \land \quad \pi \text{ uses } ^* F \]
\[ \implies \pi^F \approx \pi^\rho \]

* an unbounded number of copies of F \textit{concurrently}

Modular protocol design and proofs:
Composition Theorem

\[ \rho \text{ UC-realizes } F \quad \land \quad \pi \text{ uses}^{*} F \quad \implies \quad \pi^{F} \approx \pi^{\rho} \]

* an unbounded number of copies of \( F \) \textit{concurrently}

Modular protocol design and proofs:
Composition Theorem

\[ \rho \text{ UC-realizes } F \land \pi \text{ uses } F \Rightarrow \pi^F \approx \pi^\rho \]

* an unbounded number of copies of F \emph{concurrently}

Modular protocol design and proofs:
Ideal functionalities

Design goals:
- capture functionality
- guarantee security properties
Example: ideal commitment

Between messages: sender is committed to $b$, but receiver does not know $b$ yet.
Example: ideal commitment

Commitment phase:

Unveil/opening phase:

- hiding ✓
- binding ✓
More sophisticated

Ideal password-authenticated key exchange (PAKE)

- If $pw = pw'$ then $k = k'$
- Else $k, k'$ random

secure for any password distribution since chosen by $Z$
More sophisticated

Ideal password-authenticated key exchange (PAKE)

\[ F_{PAKE} \]
- if \( pw = pw' \) then \( k = k' \)
- else \( k, k' \) random
- if guess correct: Adv determines key

- adversary can influence outputs
Proofs of UC security

- order of quantors: ...\(\forall Z \exists S\)...
- recall: \(Z\) can vary inputs, corrupt players at different times, tamper with messages...
- \(\Rightarrow\) not efficient to check every attack
Proofs of UC security

Common technique for ind-based notions: **hybrid argument**

- start with real protocol execution
- switch to ideal execution via a sequence of indistinguishable games ([Sho04])
  1. change layout without changing input/output behaviour
  2. show indistinguishability if *BAD* event does not happen, then show $\Pr[BAD] = \text{negl}$
  3. use a computational assumption
Let the games begin...
Let the games begin...
Let the games begin...

Universal Composability (UC)
Let the games begin...

Universal Composability (UC)
Let the games begin...
Main difficulties

- F cuts S from inputs (except leakage)
- F decides about outputs (except adversarial influence in F)
When proving fails

The Framework

Proofs

Universal Composability (UC)

Conclusion
When proving fails

But: \[ b = b' \implies \text{com is not hiding!} \]
Getting help

- $S$ chooses $CRS$
- $\Rightarrow S$ knows trapdoor (e.g., exponent, pre-image etc.)
- Caution: $Z$ can access helping functionalities via $S$

The Framework

Universal Composability (UC)
## Overview (Im-)possibilities

<table>
<thead>
<tr>
<th>Task</th>
<th>Impossibility</th>
<th>Realization</th>
<th>Comm. model</th>
<th>Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>any</td>
<td>-</td>
<td>servers with honest majority</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>Coin toss</td>
<td>plain model</td>
<td>-</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>Com</td>
<td>plain model</td>
<td>-</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>OT</td>
<td>plain model</td>
<td>-</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>ZK</td>
<td>plain model</td>
<td>-</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>any</td>
<td>-</td>
<td>with CRS</td>
<td>asynchr. &amp; auth.</td>
<td>[Can01]</td>
</tr>
<tr>
<td>SMT¹</td>
<td>plain model (for some f)</td>
<td>with RO</td>
<td>asynchr. &amp; auth.</td>
<td>[Nie02]</td>
</tr>
<tr>
<td>SFE²</td>
<td>plain model (for some f)</td>
<td>with CRS</td>
<td>asynchr. &amp; auth.</td>
<td>[CKL03]</td>
</tr>
<tr>
<td>PAKE</td>
<td>-</td>
<td>with CRS</td>
<td>asynchronous</td>
<td>[BCL⁺05]</td>
</tr>
</tbody>
</table>

---

¹ Secure Message Transfer
² Secure Function Evaluation

---

**Structured Communication Framework**

The Framework | Proofs | Conclusion

Universal Composability (UC)
Related frameworks

- UC with joint state ([CR03])
- UC with global setup ([CDPW07])
- UC/C with incoercibility ([UMQ10])
- GNUC ([HS11]), fixes some flawed details in UC
- UC with super-polynomial simulators (SPS) ([Pas03])
- UC with angels ([PS04])
- UC with shielded super-polynomial simulators ([BDH+17])
Pros and Cons

- UC can capture many cryptographic tasks
- allows modular protocol design and proofs
- strong security notion
  $\Rightarrow$ realistic model
  $\Rightarrow$ implies many other notions

- strong security notion $\Rightarrow$ hard to achieve
- proofs complicated
FAQ

- **What are session identifiers (SID) for and where do they come from?**
  SIDs are used for deciding which copy of $\mathcal{F}$ in the hybrid model a message is intended for. They can be precomputed by the parties or come from external entities. See [Can01], Section 3.1.3 “On the SID mechanism”.

- **Does the simulator learn anything upon corruption in the ideal world, since dummy parties do not have coins and do not compute anything?**
  Dummy parties keep a log file of everything that was sent through them. Upon corruption the simulator learns this log. This includes, e.g., inputs from $\mathcal{Z}$ and outputs from $\mathcal{F}$ that were already sent.

- **Does the real adversary control all the channels?**
  No. It does not have read/write access to messages between parties and hybrid functionalities, but it is still allowed to schedule them.

- **Does the simulator control any channels?**
  No. This is why it has to get notified by $\mathcal{F}$, e.g., as soon as the protocol started and $\mathcal{Z}$ waits for a transcript.

- **Which inputs can $\mathcal{Z}$ send to parties?**
  Basically everything. But if the party is honest, to produce any output or transcript at all the inputs have to match the format required by $\mathcal{F}$. If the party is corrupted, then even malformed inputs might lead to a successful attack (see, e.g., the aforementioned attack on commitment protocols in the plain model).
Pointers

- [Extended Abstract of the original UC paper of Ran Canetti](http://ieeexplore.ieee.org/document/959888/)
- [Figures 1, 2 and 3 show the order of execution in the real, ideal and hybrid models](https://ia.cr/2002/059)
- [Nice UC summary in Section 2](https://ia.cr/2007/464)
- [Survey (im-)possibility results w.r.t. different (including global) setup assumptions](Can07)


Ran Canetti and Marc Fischlin. 
Universally composable commitments. 

Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. 
On the limitations of universally composable two-party computation without set-up assumptions. 

Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. 
Universally composable two-party and multi-party secure computation. 

Ran Canetti and Tal Rabin. 
Universal composition with joint state. 

Dennis Hofheinz and Victor Shoup. 
References III

Jesper Buus Nielsen.
Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case.

Rafael Pass.
Simulation in quasi-polynomial time, and its application to protocol composition.

Manoj Prabhakaran and Amit Sahai.
New notions of security: Achieving universal composability without trusted setup.

Victor Shoup.
Sequences of games: a tool for taming complexity in security proofs.

Dominique Unruh and Jörn Müller-Quade.
Universally composable incoercibility.