Compositional Security for Task-PIOAs

Ran Canetti, Ling Cheung, Dilsun Kaynar, Nancy Lynch, and Olivier Pereira

MIT Computer Science and Artificial Intelligence Laboratory

CSF, 6-8 July 2007, Venice, Italy
Outline

1. Introduction
2. Time Bounds in Task-PIOA
3. Polynomial Composition
4. Compositional Security
Analysis of Cryptographic Protocols

Three main targets:

- correctness
- efficiency
- security
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?

- Security game: e.g., IND-CPA, IND-CCA1, IND-CCA2.
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?

- **Security game**: e.g., IND-CPA, IND-CCA1, IND-CCA2.
- **Simulation-based security**: e.g., Universally Composable (UC) Security, Reactive Simulatability (RSIM).
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?

- **Security game**: e.g., IND-CPA, IND-CCA1, IND-CCA2.
- **Simulation-based security**: e.g., Universally Composable (UC) Security, Reactive Simulatability (RSIM).

Common theme: *indistinguishability.*
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?

- **Security game**: e.g., IND-CPA, IND-CCA1, IND-CCA2.
- **Simulation-based security**: e.g., Universally Composable (UC) Security, Reactive Simulatability (RSIM).

Common theme: *indistinguishability.*

Differences:
- security games are easier to prove;
Analysis of Cryptographic Protocols

Three main targets:
- correctness
- efficiency
- security

How do we define security?
- *Security game*: e.g., IND-CPA, IND-CCA1, IND-CCA2.
- *Simulation-based security*: e.g., Universally Composable (UC) Security, Reactive Simulatability (RSIM).

Common theme: *indistinguishability*.
Differences:
- security games are easier to prove;
- simulation-based security is composable.
Simulation-Based Security

"securely emulates"

\[ \phi \trianglelefteq E \psi \iff \]

Canetti et al. Compositional Security
Simulation-Based Security

“securely emulates”

\[ \phi \leq^E \psi \iff \forall \text{Adv} \exists \text{Sim} \forall \text{Env} \quad \text{Adv} \parallel \phi \parallel \text{Env} \approx \text{Sim} \parallel \psi \parallel \text{Env} \]
Simulation-Based Security

“securely emulates”
\[ \phi \leq_E \psi \iff \forall \text{Adv} \exists \text{Sim} \forall \text{Env} \; \text{Adv} \| \phi \| \text{Env} \approx \text{Sim} \| \psi \| \text{Env} \]

\( \phi: \) real protocol
\( \psi: \) ideal protocol
\( \approx: \) indistinguishable (perfectly, statistically, computationally)
Composability: One-Page Proof

**Theorem.** If $\phi \leq_E \psi$, then $\phi \parallel \eta \leq_E \psi \parallel \eta$. 

**Proof.** Let $\text{Adv}$ be given. Choose $\text{Sim}$ such that $\forall \text{Env} \\text{Adv} \parallel \phi \parallel \text{Env} \approx \text{Sim} \parallel \psi \parallel \text{Env}$. Then $\text{Adv} \parallel \phi \parallel \eta \parallel \text{Env} \approx \text{Adv} \parallel \phi \parallel \text{Env}' \approx \text{Sim} \parallel \psi \parallel \text{Env}' \approx \text{Sim} \parallel \psi \parallel \eta \parallel \text{Env}$. 

Hidden hurdles: associativity, compatibility, ... 

Most importantly, $\approx$ must be preserved under substitutions.
Composability: One-Page Proof

Theorem. If $\phi \leq_E \psi$, then $\phi \| \eta \leq_E \psi \| \eta$.

Proof. Let $Adv$ be given. Choose $Sim$ such that

$$\forall Env \quad Adv \| \phi \| Env \approx Sim \| \psi \| Env$$
Composability: One-Page Proof

**Theorem.** If $\phi \leq_E \psi$, then $\phi \parallel \eta \leq_E \psi \parallel \eta$.

**Proof.** Let $Adv$ be given. Choose $Sim$ such that

$$\forall Env \quad Adv \parallel \phi \parallel Env \approx Sim \parallel \psi \parallel Env$$

Let $Env$ be given. Set $Env' := \eta \parallel Env$. 

Hidden hurdles: associativity, compatibility, 

Most importantly, $\approx$ must be preserved under substitutions.

Canetti et al. Compositional Security
Composability: One-Page Proof

Theorem. If \( \phi \leq_E \psi \), then \( \phi \| \eta \leq_E \psi \| \eta \).

Proof. Let \( \text{Adv} \) be given. Choose \( \text{Sim} \) such that

\[
\forall \text{Env} \quad \text{Adv}\|\phi\|\text{Env} \approx \text{Sim}\|\psi\|\text{Env}
\]

Let \( \text{Env} \) be given. Set \( \text{Env}' := \eta \| \text{Env} \). Then

\[
\text{Adv}\|\phi\|\eta\|\text{Env} \approx \text{Adv}\|\phi\|\text{Env}' \approx \text{Sim}\|\psi\|\text{Env}' \approx \text{Sim}\|\psi\|\eta\|\text{Env}.
\]
Composability: One-Page Proof

**Theorem.** If $\phi \leq_E \psi$, then $\phi \parallel \eta \leq_E \psi \parallel \eta$.

**Proof.** Let $Adv$ be given. Choose $Sim$ such that

$$\forall Env \quad Adv \parallel \phi \parallel Env \approx Sim \parallel \psi \parallel Env$$

Let $Env$ be given. Set $Env' := \eta \parallel Env$. Then

$$Adv \parallel \phi \parallel \eta \parallel Env \approx Adv \parallel \phi \parallel Env' \approx Sim \parallel \psi \parallel Env' \approx Sim \parallel \psi \parallel \eta \parallel Env.$$  

Hidden hurdles: associativity, compatibility, . . .
Composability: One-Page Proof

**Theorem.** If $\phi \leq_E \psi$, then $\phi\|\eta \leq_E \psi\|\eta$.

**Proof.** Let $Adv$ be given. Choose $Sim$ such that

$$\forall Env \quad Adv\|\phi\|Env \approx Sim\|\psi\|Env$$

Let $Env$ be given. Set $Env' := \eta\|Env$. Then

$$Adv\|\phi\|\eta\|Env \approx Adv\|\phi\|Env' \approx Sim\|\psi\|Env' \approx Sim\|\psi\|\eta\|Env.$$
Two Layers of Composability Claims

Hard: Composability in the underlying model of concurrent computation.

Easy: Composability in the security layer.
Stop Being Sloppy . . .

A protocol $\phi$ is a family $\{\phi_1, \phi_2, \ldots, \phi_k, \ldots\}$, indexed by security parameter $k$. 
Description Bounds

\( \phi = \{ \phi_1, \phi_2, \ldots, \phi_k, \ldots \} \) is said to have polynomially bounded description if there is a polynomial \( p(k) \) such that, for all \( k \),

- every constituent (e.g., state, action, task) of \( \phi_k \) can be
  - encoded with fewer than \( p(k) \) bits and
  - recognized in fewer than \( p(k) \) Turing steps;

Caution: This is not polynomial-time in the traditional sense. Bounded description \( \neq \) bounded runtime. (Distinctive feature of task-PIOA!)
Description Bounds

\( \phi = \{ \phi_1, \phi_2, \ldots, \phi_k, \ldots \} \) is said to have *polynomially bounded description* if there is a polynomial \( p(k) \) such that, for all \( k \),

- every constituent (e.g., state, action, task) of \( \phi_k \) can be
  - encoded with fewer than \( p(k) \) bits and
  - recognized in fewer than \( p(k) \) Turing steps;
- all single-step transitions of \( \phi_k \) can be computable in at most \( p(k) \) Turing steps;

Caution: This is not polynomial-time in the traditional sense. Bounded description \( \neq \) bounded runtime. (Distinctive feature of task-PIOA!)

Canetti et al. Compositional Security
Description Bounds

$\phi = \{\phi_1, \phi_2, \ldots, \phi_k, \ldots\}$ is said to have *polynomially bounded description* if there is a polynomial $p(k)$ such that, for all $k$,

- every constituent (e.g., state, action, task) of $\phi_k$ can be encoded with fewer than $p(k)$ bits and recognized in fewer than $p(k)$ Turing steps;
- all *single-step* transitions of $\phi_k$ can be computable in at most $p(k)$ Turing steps;
- all relevant (probabilistic) Turing machines can be encoded with fewer than $p(k)$ bits.

Caution: This is *not* polynomial-time in the traditional sense. Bounded description $\neq$ bounded runtime. (Distinctive feature of task-PIOA!)
Description Bounds

\( \phi = \{ \phi_1, \phi_2, \ldots, \phi_k, \ldots \} \) is said to have \textit{polynomially bounded description} if there is a polynomial \( p(k) \) such that, for all \( k \),

- every constituent (e.g., state, action, task) of \( \phi_k \) can be
  - encoded with fewer than \( p(k) \) bits and
  - recognized in fewer than \( p(k) \) Turing steps;
- all single-step transitions of \( \phi_k \) can be computable in at most \( p(k) \) Turing steps;
- all relevant (probabilistic) Turing machines can be encoded with fewer than \( p(k) \) bits.

\textbf{Caution}: This is \textit{not} polynomial-time in the traditional sense. Bounded description \( \not\implies \) bounded runtime.

(Distinctive feature of task-PIOA!)
Computational Implementation

\[ \phi \leq_{\text{neg,pt}} \psi \iff \forall p, q_1 \exists q_2, \epsilon \forall k \]

\[ \forall p(k)\text{-bounded environment } Env \]

\[ \forall q_1(k)\text{-bounded task schedule } \rho_1 \]

\[ \exists q_2(k)\text{-bounded task schedule } \rho_2 \]

\[ | P_{\text{acc}}(\phi_k \parallel Env, \rho_1) - P_{\text{acc}}(\psi_k \parallel Env, \rho_2) | \leq \epsilon(k) \]
Computational Implementation

\[ \phi \leq_{\text{neg,pt}} \psi \iff \forall p, q_1 \exists q_2, \epsilon \forall k \]

- \forall p(k)-bounded environment \( Env \)
- \forall q_1(k)-bounded task schedule \( \rho_1 \)
- \exists q_2(k)-bounded task schedule \( \rho_2 \)

\[ |P_{\text{acc}}(\phi_k \parallel Env, \rho_1) - P_{\text{acc}}(\psi_k \parallel Env, \rho_2)| \leq \epsilon(k) \]

**Theorem.** If \( \phi \leq_{\text{neg,pt}} \psi \), then \( \phi \parallel \eta \leq_{\text{neg,pt}} \psi \parallel \eta \).
Computational Implementation

\[ \phi \leq_{\text{neg,pt}} \psi \iff \forall p, q_1 \exists q_2, \epsilon \forall k \]
\[ \forall p(k)\text{-bounded environment } Env \]
\[ \forall q_1(k)\text{-bounded task schedule } \rho_1 \]
\[ \exists q_2(k)\text{-bounded task schedule } \rho_2 \]
\[ |P_{\text{acc}}(\phi_k \parallel Env, \rho_1) - P_{\text{acc}}(\psi_k \parallel Env, \rho_2)| \leq \epsilon(k) \]

**Theorem.** If \( \phi \leq_{\text{neg,pt}} \psi \), then \( \phi \parallel \eta \leq_{\text{neg,pt}} \psi \parallel \eta \).

**Proof.** Set \( Env' := \eta \parallel Env \) and use associativity.
Polynomial Composition

What if we compose multiple instances? (E.g., a parent process that invokes dynamically multiple copies of the same protocol.)

\[ i\text{-th copy of } \phi: \phi_i = \{(\phi_i)_1, \ldots (\phi_i)_k, \ldots \} \]

\[ i\text{-th copy of } \psi: \psi_i = \{(\psi_i)_1, \ldots (\psi_i)_k, \ldots \} \]
Polynomial Composition

What if we compose multiple instances? (E.g., a parent process that invokes dynamically multiple copies of the same protocol.)

$i$-th copy of $\phi$: $\phi_i = \{(\phi_i)_1, \ldots (\phi_i)_k, \ldots\}$

$i$-th copy of $\psi$: $\psi_i = \{(\psi_i)_1, \ldots (\psi_i)_k, \ldots\}$

Let $b$ be a polynomial.

$(\hat{\phi})_k := (\phi_1)_k \parallel \ldots \parallel (\phi_{b(k)})_k$

$(\hat{\psi})_k := (\psi_1)_k \parallel \ldots \parallel (\psi_{b(k)})_k$
Polynomial Composition

What if we compose multiple instances? (E.g., a parent process that invokes dynamically multiple copies of the same protocol.)

\[ i \text{-th copy of } \phi: \quad \phi_i = \{(\phi_i)_1, \ldots (\phi_i)_k, \ldots \} \]
\[ i \text{-th copy of } \psi: \quad \psi_i = \{(\psi_i)_1, \ldots (\psi_i)_k, \ldots \} \]

Let \( b \) be a polynomial.
\[ (\hat{\phi})_k := (\phi_1)_k \| \cdots \| (\phi_{b(k)})_k \]
\[ (\hat{\psi})_k := (\psi_1)_k \| \cdots \| (\psi_{b(k)})_k \]

"Theorem". If \( \phi_i \leq_{\text{neg,pt}} \psi_i \) for every \( i \), then \( \hat{\phi} \leq_{\text{neg,pt}} \hat{\psi} \).
Naive Solution

Repeated application of the binary composition theorem.

\[
\begin{align*}
(\phi_1)_k \parallel ((\phi_2)_k \parallel \ldots \parallel (\phi_{b(k)})_k \parallel Env) \\
(\psi_1)_k \parallel ((\phi_2)_k \parallel \ldots \parallel (\phi_{b(k)})_k \parallel Env) \\
(\phi_2)_k \parallel ((\psi_1)_k \parallel (\phi_3)_k \parallel \ldots \parallel (\phi_{b(k)})_k \parallel Env) \\
(\psi_2)_k \parallel ((\psi_1)_k \parallel (\phi_3)_k \parallel \ldots \parallel (\phi_{b(k)})_k \parallel Env) \\
\ldots \\
(\psi_1)_k \parallel ((\psi_2)_k \parallel \ldots \parallel (\psi_{b(k)})_k \parallel Env)
\end{align*}
\]
Naive Solution

Schedule length bounds:

\[ \forall q_1 \exists q_2 \exists q_3 \exists q_4 \ldots \]
Naive Solution

Schedule length bounds:
\[ \forall q_1 \exists q_2 \exists q_3 \exists q_4 \ldots \]

Problem!
\[ q_i \text{'s may grow exponentially: } \forall i \ q_{i+1} = 2 \cdot q_i \]
Schedule length bound for \( \hat{\psi} \) is \( \hat{q}(k) = 2^{b(k)} \cdot q_1(k) \).
Not polynomial.
Naive Solution

Schedule length bounds:
\[ \forall q_1 \exists q_2 \\exists q_3 \\exists q_4 \ldots \]

Problem!
\( q_i \)'s may grow exponentially: \( \forall i \ q_{i+1} = 2 \cdot q_i \)
Schedule length bound for \( \hat{\psi} \) is \( \hat{q}(k) = 2^{b(k)} \cdot q_1(k) \).
*Not* polynomial.

Worse yet: error \( \epsilon \) depends on schedule length bound \( q_i \), so a different \( \epsilon_i \) at every step!
\( \hat{\epsilon}(k) = \sum_{i=1}^{b(k)} \epsilon_i(k) \) still negligible?
Computational Implementation (Take 2)

\[ \phi \leq^{\text{strong}}_{\text{neg,pt}} \psi \iff \forall q_1 \exists q_2 \forall p, q \exists \epsilon \forall k \]
\[ \forall p(k)\text{-bounded environment } Env \]
\[ \forall \text{task schedule } \rho_1 \text{ such that } \]
\[ \text{proj}_\phi(\rho_1) \text{ is } q_1(k)\text{-bounded} \]
\[ \text{proj}_{Env}(\rho_1) \text{ is } q(k)\text{-bounded} \]
\[ \exists \text{task schedule } \rho_2 \text{ such that } \]
\[ \text{proj}_\psi(\rho_2) \text{ is } q_2(k)\text{-bounded} \]
\[ \text{proj}_{Env}(\rho_1) = \text{proj}_{Env}(\rho_2) \]
\[ \left| P_{\text{acc}}(\phi_k \parallel Env, \rho_1) - P_{\text{acc}}(\psi_k \parallel Env, \rho_2) \right| \leq \epsilon(k) \]
Computational Implementation (Take 2)

Main changes.

- Separate schedule bounds.
Main changes.

- Separate schedule bounds.
- $q_2$ independent of $q$. 

Canetti et al.
Computational Implementation (Take 2)

Main changes.
- Separate schedule bounds.
- $q_2$ independent of $q$.
- Environment tasks fixed.
Hybrid Argument

Theorem. If $\phi_i \leq_{\text{neg,pt}} \psi_i$ for every $i$, then $\hat{\phi} \leq_{\text{neg,pt}} \hat{\psi}$
Hybrid Argument

Theorem. If $\phi_i \leq_{\text{neg,pt}} \psi_i$ for every $i$, then $\hat{\phi} \leq_{\text{neg,pt}} \hat{\psi}$

Proof. Fix $k$. Define hybrid automata: $H_k^0, \ldots, H_k^i, \ldots H_k^{b(k)}$.

$H_k^i := (\psi_1)_k \parallel \ldots \parallel (\psi_i)_k \parallel (\phi_{i+1})_k \parallel \ldots \parallel (\phi_{b(k)})_k$
Hybrid Argument

*Theorem.* If $\phi_i \leq_{\text{neg,pt}} \psi_i$ for every $i$, then $\hat{\phi} \leq_{\text{neg,pt}} \hat{\psi}$

*Proof.* Fix $k$. Define *hybrid automata*: $H_k^0, \ldots, H_k^i, \ldots, H_k^{b(k)}$

$$H_k^i := (\psi_1)_k \parallel \ldots \parallel (\psi_i)_k \parallel (\phi_{i+1})_k \parallel \ldots \parallel (\phi_{b(k)})_k$$

Note that $H_k^0 = (\hat{\phi})_k$ and $H_k^{b(k)} = (\hat{\psi})_k$. 

Canetti et al.
Hybrid Argument

Theorem. If $\phi_i \leq_{\text{neg,pt}} \psi_i$ for every $i$, then $\hat{\phi} \leq_{\text{neg,pt}} \hat{\psi}$

Proof. Fix $k$. Define hybrid automata: $H^0_k, \ldots, H^i_k, \ldots, H^b(k)_k$.

$H^i_k := (\psi_1)_k \parallel \ldots \parallel (\psi_i)_k \parallel (\phi_{i+1})_k \parallel \ldots \parallel (\phi_{b(k)})_k$

Note that $H^0_k = (\hat{\phi})_k$ and $H^b(k)_k = (\hat{\psi})_k$

\[
\begin{align*}
&| P_{\text{acc}}(\hat{\phi}_k \parallel \text{Env}, \rho_1) - P_{\text{acc}}(\hat{\psi}_k \parallel \text{Env}, \rho_{b(k)+1}) | \\
&\leq | P_{\text{acc}}(H^0_k \parallel \text{Env}, \rho_1) - P_{\text{acc}}(H^1_k \parallel \text{Env}, \rho_2) | \\
&\quad + | P_{\text{acc}}(H^1_k \parallel \text{Env}, \rho_2) - P_{\text{acc}}(H^2_k \parallel \text{Env}, \rho_3) | \\
&\quad + \ldots + | P_{\text{acc}}(H^{b(k)-1}_k \parallel \text{Env}, \rho_{b(k)}) - P_{\text{acc}}(H^{b(k)}_k \parallel \text{Env}, \rho_{b(k)+1}) | \\
&< b(k) \cdot \epsilon(k)
\end{align*}
\]
Compositional Security

“securely emulates”

\[ \phi \leq_E \psi \iff \forall \text{Adv} \exists \text{Sim Adv} \| \phi \leq_{\text{strong neg, pt}} \text{Sim} \| \psi \]
Compositional Security

“securely emulates”

\[
\phi \leq_E \psi \iff \forall \text{Adv} \ \exists \text{Sim Adv} \parallel \phi \leq_{\text{strong neg, pt}} \text{Sim} \parallel \psi
\]

Remark: “\(\forall \text{Env}\)” is encapsulated in \(\leq_{\text{neg, pt}}\).
Compositional Security

"securely emulates"

\[ \phi \leq_E \psi \iff \forall \text{Adv} \exists \text{Sim Adv} \parallel \phi \leq_{\text{neg,pt}} \text{Sim} \parallel \psi \]

Remark: “∀Env” is encapsulated in \( \leq_{\text{neg,pt}} \).

Theorem. If \( \phi_i \leq_E \psi_i \) uniformly for every \( i \), then \( \hat{\phi} \leq_{\text{neg,pt}} \hat{\psi} \).

Proof. Dummy adversaries and composition theorem for \( \leq_{\text{neg,pt}} \).
Dummy Adversaries

*Dummy adversary*: forwarder between protocol and environment.
**Dummy Adversaries**

*Dummy adversary:* forwarder between protocol and environment.

Formal property: \( f(\phi) \leq_{\text{neg,pt}} \phi \parallel \text{Adv}_{\text{dummy}} \), where \( f \) is a renaming function.

\[
\begin{align*}
  \phi & \xleftarrow{} A\text{Act}_\phi \\
  f(\phi) & \xrightarrow{} Env \\
  f(A\text{Act}_\phi) & \\
  \phi & \xrightarrow{} \text{Adv}_{\text{dummy}} \\
  f(A\text{Act}_\phi) & \xleftarrow{} Env
\end{align*}
\]
Proof of Secure Composition

Step 1. Get “big” Adv for \( \hat{\phi} \). Try to construct Sim for \( \hat{\psi} \).
Proof of Secure Composition

Step 1. Get “big” Adv for $\hat{\phi}$. Try to construct Sim for $\hat{\psi}$.

Step 2. Get Sim$^i$ for each Adv$^i_{\text{dummy}}$.

\[
\begin{align*}
\phi_i &\xleftarrow{} AAct_{\phi_i} \xrightarrow{} Adv \\
&\quad \quad f(\phi_i) \xleftarrow{} f(AAct_{\phi_i}) \xrightarrow{} f(Adv) \\
\phi_i &\xleftarrow{} AAct_{\phi_i} \xrightarrow{} Adv^i_{\text{dummy}} \\
&\quad \quad f(\phi_i) \xleftarrow{} f(AAct_{\phi_i}) \xrightarrow{} f(Adv) \\
\psi_i &\xleftarrow{} AAct_{\psi_i} \xrightarrow{} Sim^i \\
&\quad \quad f(\psi_i) \xleftarrow{} f(AAct_{\psi_i}) \xrightarrow{} f(Adv)
\end{align*}
\]
Proof of Secure Composition

Step 1. Get “big” $Adv$ for $\hat{\phi}$. Try to construct $Sim$ for $\hat{\psi}$.

Step 2. Get $Sim^i$ for each $Adv^i_{\text{dummy}}$.

Step 3. $Sim := (\bigparallel_i Sim^i) \parallel f(Adv)$. \[\square\]
Conclusions and Future Work

- Unbounded forwarder.
Conclusions and Future Work

- Unbounded forwarder.
- Dynamic process creation.
Conclusions and Future Work

- Unbounded forwarder.
- Dynamic process creation.
- Timed computational analysis: Haber’s protocol.
Conclusions and Future Work

- Unbounded forwarder.
- Dynamic process creation.
- Timed computational analysis: Haber’s protocol.
- More case studies: statistical ZK, ABE, etc.