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

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"

$\phi \leq_E \psi \;\; \Leftrightarrow$

Simulation-Based Security

"securely emulates"

$\phi \leq_E \psi \ \Leftrightarrow$

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

Simulation-Based Security

"securely emulates"
$\phi \leq_E \psi \;\Leftrightarrow$

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

ϕ: *real* protocol
ψ: *ideal* protocol
$\approx$: indistinguishable (perfectly, statistically, computationally)

Composability: One-Page Proof

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

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

$$\forall \mathit{Env} \quad \mathit{Adv} \| \phi \| \mathit{Env} \approx \mathit{Sim} \| \psi \| \mathit{Env}$$

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

$$\forall \mathit{Env} \quad \mathit{Adv} \| \phi \| \mathit{Env} \approx \mathit{Sim} \| \psi \| \mathit{Env}$$

Let *Env* be given. Set $\mathit{Env}' := \eta \| \mathit{Env}$.

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

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

$\square$

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

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

$\square$

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

$$\forall \mathit{Env} \quad \mathit{Adv} \| \phi \| \mathit{Env} \approx \mathit{Sim} \| \psi \| \mathit{Env}$$

Let *Env* be given. Set $\mathit{Env}' := \eta \| \mathit{Env}$. Then

$$\mathit{Adv} \| \phi \| \eta \| \mathit{Env} \approx \mathit{Adv} \| \phi \| \mathit{Env}' \approx \mathit{Sim} \| \psi \| \mathit{Env}' \approx \mathit{Sim} \| \psi \| \eta \| \mathit{Env}.$$

$\square$

Hidden hurdles: associativity, compatibility, . . .

Most important*ly*, $\approx$ must be preserved under substitutions.

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 ϕ 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 ϕ_k can be
  - encoded with fewer than $p(k)$ bits and
  - recognized in fewer than $p(k)$ Turing steps;

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 ϕ_k can be
  - encoded with fewer than $p(k)$ bits and
  - recognized in fewer than $p(k)$ Turing steps;
- all single-step transitions of ϕ_k can be computable in at most $p(k)$ Turing steps;

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 ϕ_k can be
  - encoded with fewer than $p(k)$ bits and
  - recognized in fewer than $p(k)$ Turing steps;
- all single-step transitions of ϕ_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.

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 ϕ_k can be
  - encoded with fewer than $p(k)$ bits and
  - recognized in fewer than $p(k)$ Turing steps;
- all single-step transitions of ϕ_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 $\not\Longrightarrow$ bounded runtime.
(Distinctive feature of task-PIOA!)

Computational Implementation

$$
\begin{aligned}
\phi \leq_{\text{neg,pt}} \psi \;\Leftrightarrow\; & \forall p, q_1\ \exists q_2, \epsilon\ \forall k \\
& \forall p(k)\text{-bounded environment } Env \\
& \forall q_1(k)\text{-bounded task schedule } \rho_1 \\
& \exists q_2(k)\text{-bounded task schedule } \rho_2 \\
& |\,\mathbf{P}_{\text{acc}}(\phi_k \| Env, \rho_1) - \mathbf{P}_{\text{acc}}(\psi_k \| Env, \rho_2)| \leq \epsilon(k)
\end{aligned}
$$

Computational Implementation

$$
\begin{aligned}
\phi \leq_{\text{neg,pt}} \psi \;\; \Leftrightarrow \; & \forall p, q_1 \; \exists q_2, \epsilon \; \forall k \\
& \forall p(k)\text{-bounded environment } Env \\
& \forall q_1(k)\text{-bounded task schedule } \rho_1 \\
& \exists q_2(k)\text{-bounded task schedule } \rho_2 \\
& |\, \mathbf{P}_{\text{acc}}(\phi_k \| Env, \rho_1) - \mathbf{P}_{\text{acc}}(\psi_k \| Env, \rho_2)| \leq \epsilon(k)
\end{aligned}
$$

Theorem. If $\phi \leq_{\text{neg,pt}} \psi$, then $\phi \| \eta \leq_{\text{neg,pt}} \psi \| \eta$.

Computational Implementation

$$
\begin{aligned}
\phi \leq_{\text{neg,pt}} \psi \;\; \Leftrightarrow\; & \forall p, q_1 \; \exists q_2, \epsilon \; \forall k \\
& \forall p(k)\text{-bounded environment } \mathit{Env} \\
& \forall q_1(k)\text{-bounded task schedule } \rho_1 \\
& \exists q_2(k)\text{-bounded task schedule } \rho_2 \\
& |\, \mathbf{P}_{\text{acc}}(\phi_k \| \mathit{Env}, \rho_1) - \mathbf{P}_{\text{acc}}(\psi_k \| \mathit{Env}, \rho_2)| \leq \epsilon(k)
\end{aligned}
$$

Theorem. If $\phi \leq_{\text{neg,pt}} \psi$, then $\phi \| \eta \leq_{\text{neg,pt}} \psi \| \eta$.
Proof. Set $\mathit{Env}' := \eta \| \mathit{Env}$ 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-th copy of ϕ: $\phi_i = \{(\phi_i)_1, \ldots (\phi_i)_k, \ldots\}$
i-th copy of ψ: $\psi_i = \{(\psi_i)_1, \ldots (\psi_i)_k, \ldots\}$

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 ϕ: $\phi_i = \{(\phi_i)_1, \dots (\phi_i)_k, \dots\}$
i-th copy of ψ: $\psi_i = \{(\psi_i)_1, \dots (\psi_i)_k, \dots\}$

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

Polynomial Composition

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

i-th copy of ϕ: $\phi_i = \{(\phi_i)_1, \dots (\phi_i)_k, \dots\}$
i-th copy of ψ: $\psi_i = \{(\psi_i)_1, \dots (\psi_i)_k, \dots\}$

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

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

Naive Solution

Repeated application of the binary composition theorem.

$$(\phi_1)_k \| ((\phi_2)_k \| \dots \| (\phi_{b(k)})_k \| Env)$$
$$(\psi_1)_k \| ((\phi_2)_k \| \dots \| (\phi_{b(k)})_k \| Env)$$
$$(\phi_2)_k \| ((\psi_1)_k \| (\phi_3)_k \| \dots \| (\phi_{b(k)})_k \| Env)$$
$$(\psi_2)_k \| ((\psi_1)_k \| (\phi_3)_k \| \dots \| (\phi_{b(k)})_k \| Env)$$
$$\dots$$
$$(\psi_1)_k \| ((\psi_2)_k \| \dots \| (\psi_{b(k)})_k \| Env)$$

Naive Solution

Schedule length bounds:

$\forall q_1 \ \exists q_2$

$\quad \forall q_2 \ \exists q_3$

$\quad\quad \forall q_3 \ \exists q_4 \ \ldots$

Naive Solution

Schedule length bounds:

$\forall q_1 \ \exists q_2$

$\quad \forall q_2 \ \exists q_3$

$\quad\quad \forall q_3 \ \exists q_4 \ \ldots$

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.

Naive Solution

Schedule length bounds:

$\forall q_1 \ \exists q_2$

$\quad \forall q_2 \ \exists q_3$

$\quad\quad \forall q_3 \ \exists q_4 \ \ldots$

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 ϵ depends on schedule length bound q_i, so a different ϵ_i at every step!

$\hat{\epsilon}(k) = \sum_{i=1}^{b(k)} \epsilon_i(k)$ still negligible?

Computational Implementation (Take 2)

$$
\begin{aligned}
\phi \leq_{\text{neg,pt}}^{\text{strong}} \psi \quad \Leftrightarrow\ & \forall q_1\ \exists q_2\ \forall p, q\ \exists \epsilon\ \forall k \\
& \forall p(k)\text{-bounded environment } Env \\
& \forall \text{ task schedule } \rho_1 \text{ such that} \\
& \quad \text{proj}_{\phi}(\rho_1) \text{ is } q_1(k)\text{-bounded} \\
& \quad \text{proj}_{Env}(\rho_1) \text{ is } q(k)\text{-bounded} \\
& \exists \text{ task schedule } \rho_2 \text{ such that} \\
& \quad \text{proj}_{\psi}(\rho_2) \text{ is } q_2(k)\text{-bounded} \\
& \quad \text{proj}_{Env}(\rho_1) = \text{proj}_{Env}(\rho_2) \\
& |\,\mathbf{P}_{\text{acc}}(\phi_k \| Env, \rho_1) - \mathbf{P}_{\text{acc}}(\psi_k \| Env, \rho_2)| \leq \epsilon(k)
\end{aligned}
$$

Computational Implementation (Take 2)

Main changes.

- Separate schedule bounds.

Computational Implementation (Take 2)

Main changes.

- Separate schedule bounds.
- q_2 independent of q.

Computational Implementation (Take 2)

Main changes.

- Separate schedule bounds.
- q_2 independent of q.
- Environment tasks fixed.

Hybrid Argument

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

Hybrid Argument

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

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

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

Hybrid Argument

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

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

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

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

Hybrid Argument

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

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

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

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

$$
\begin{aligned}
&|\,\mathbf{P}_{\text{acc}}((\hat{\phi})_k \| Env, \rho_1) - \mathbf{P}_{\text{acc}}((\hat{\psi})_k \| Env, \rho_{b(k)+1})| \\
&\leq |\,\mathbf{P}_{\text{acc}}(H^0_k \| Env, \rho_1) - \mathbf{P}_{\text{acc}}(H^1_k \| Env, \rho_2)| \\
&\quad + |\,\mathbf{P}_{\text{acc}}(H^1_k \| Env, \rho_2) - \mathbf{P}_{\text{acc}}(H^2_k \| Env, \rho_3)| \\
&\quad + \ldots + |\,\mathbf{P}_{\text{acc}}(H^{b(k)-1} \| Env, \rho_{b(k)}) - \mathbf{P}_{\text{acc}}(H^{b(k)} \| Env, \rho_{b(k)+1})| \\
&< b(k) \cdot \epsilon(k)
\end{aligned}
$$

Compositional Security

"securely emulates"

$$\phi \leq_E \psi \;\Leftrightarrow\; \forall Adv\ \exists Sim\ Adv \| \phi \leq^{\text{strong}}_{\text{neg,pt}} Sim \| \psi$$

Compositional Security

"securely emulates"

$$\phi \leq_E \psi \;\Leftrightarrow\; \forall Adv\ \exists Sim\ Adv \| \phi \leq^{\text{strong}}_{\text{neg,pt}} Sim \| \psi$$

Remark: "$\forall Env$" is encapsulated in $\leq^{\text{strong}}_{\text{neg,pt}}$.

Compositional Security

"securely emulates"

$$\phi \leq_E \psi \iff \forall Adv\ \exists Sim\ Adv \| \phi \leq^{\text{strong}}_{\text{neg,pt}} Sim \| \psi$$

Remark: "$\forall Env$" is encapsulated in $\leq^{\text{strong}}_{\text{neg,pt}}$.

Theorem. If $\phi_i \leq_E \psi_i$ uniformly for every i, then $\hat{\phi} \leq^{\text{strong}}_{\text{neg,pt}} \hat{\psi}$
Proof. Dummy adversaries and composition theorem for $\leq^{\text{strong}}_{\text{neg,pt}}$.

Dummy Adversaries

Dummy adversary: forwarder between protocol and environment.

Dummy Adversaries

Dummy adversary: forwarder between protocol and environment.

Formal property: $f(\phi) \leq^{\text{strong}}_{\text{neg,pt}} \phi \| Adv_{\text{dummy}}$, where f is a renaming function.

$$\begin{tikzcd} f(\phi) \arrow[r, bend right, "f(AAct_\phi)"'] & Env \arrow[l, bend right] \end{tikzcd}$$

$$\begin{tikzcd} \phi \arrow[r, bend right, "AAct_\phi"'] & Adv_{\text{dummy}} \arrow[l, bend right] \arrow[r, bend right, "f(AAct_\phi)"'] & Env \arrow[l, bend right] \end{tikzcd}$$

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_{dummy}.

$$\begin{tikzcd}
\phi_i \arrow[r, bend right, "AAct_{\phi_i}"'] & Adv \arrow[l, bend right]
\end{tikzcd}$$

$$\begin{tikzcd}
f(\phi_i) \arrow[r, bend right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, bend right]
\end{tikzcd}$$

$$\begin{tikzcd}
\phi_i \arrow[r, bend right, "AAct_{\phi_i}"'] & Adv^i_{\text{dummy}} \arrow[l, bend right] \arrow[r, bend right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, bend right]
\end{tikzcd}$$

$$\begin{tikzcd}
\psi_i \arrow[r, bend right, "AAct_{\psi_i}"'] & Sim^i \arrow[l, bend right] \arrow[r, bend right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, bend right]
\end{tikzcd}$$

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_{dummy}.

$$\begin{tikzcd}
\phi_i \arrow[r, shift right, "AAct_{\phi_i}"'] & Adv \arrow[l, shift right]
\end{tikzcd}$$

$$\begin{tikzcd}
f(\phi_i) \arrow[r, shift right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, shift right]
\end{tikzcd}$$

$$\begin{tikzcd}
\phi_i \arrow[r, shift right, "AAct_{\phi_i}"'] & Adv^i_{\text{dummy}} \arrow[l, shift right] \arrow[r, shift right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, shift right]
\end{tikzcd}$$

$$\begin{tikzcd}
\psi_i \arrow[r, shift right, "AAct_{\psi_i}"'] & Sim^i \arrow[l, shift right] \arrow[r, shift right, "f(AAct_{\phi_i})"'] & f(Adv) \arrow[l, shift right]
\end{tikzcd}$$

Step 3. $Sim := (\|_i Sim^i)\|f(Adv)$. □

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.