Using Task-Structured PIOAs to Analyze Cryptographic Protocols

Ran Canetti, Ling Cheung, Dilsun Kaynar, Moses Liskov,
Nancy Lynch, Olivier Pereira and Roberto Segala

March 5, 2006

Nondeterminism

Nondeterminism in models for protocols:

- in concurrency: keep it as much as you can!
  - generality: allows more implementations
  - clarity: no unnecessary constraints
  - used in IOAs, PIOAs, ...
- in crypto: get rid of it!
  - we want computational indistinguishability, functional behaviors, ...

One of our goals:

- Reconcile nondeterministic and probabilistic choices in a crypto setting

PIOAs

PIOAs are kinds of interacting, abstract, automata:

- state variables
- actions (input, output, internal)
- transitions: $(\mathit{state} \times \mathit{action}) \rightarrow \mathrm{Disc}(\mathit{states}) \cup \bot$

Internal nondeterminism for output and internal actions

- not algorithmically resolved
- not resolved in the analyzed systems

High-level nondeterminism algorithmically resolved (by *Adv*)
How do we resolve the low-level (internal) nondeterminism?

Task-PIOAs

Task-PIOAs are PIOAs with tasks: equivalence classes on actions (ex: send message 1, select key, . . .)

- given a task, at most one possible (probabilistic) action

Task schedulers resolve low-level nondeterminism and give probabilistic executions

- task schedulers do not give extra power to *Adv*

Conclusion

We hope task-PIOAs provide a framework for:

- More general, expressive, specifications
- More general, systematic, security proofs

Case-study on a simple OT procotol [GMW87]

Security

Implementation relation for task-PIOAs:

- $A \leq B$ means:
  $\forall$ env. E and $\forall$ task scheduler for $A||E$, $\exists$ task scheduler for $B||E$ s.t. E cannot distinguish A from B

UC-style security:

- Protocol P realizes specification F iff
  $\forall$ task-PIOA A, $\exists$ task-PIOA S: $P||A \leq F||S$

Proving Security

Two tools:

1. Sound simulation relation for $\leq_0$:
   - on probability distributions on execution fragments
   - $\forall$ task $T, \exists T_1, \ldots T_n$ s.t.
     $\epsilon_1 R \epsilon_2 \Rightarrow apply(\epsilon_1, T)\ \mathcal{E}(R)\ apply(\epsilon_2, T_1, \ldots, T_n)$
   - only available for perfectly indistinguishable systems
2. Composability of $\leq_{neg,pt}$:
   - Express computational assumptions as $C_1 \leq_{neg,pt} C_2$
     Ex: hard-core predicate B for f:
     C_1 outputs $f, f(x), B(x)$ and C_2 outputs $f, f(x), b$
   - Composability:
     $C_1 \leq_{neg,pt} C_2 \Rightarrow C_1 || Ifc \leq_{neg,pt} C_2 || Ifc$