

Games and the Impossibility of Realizable Ideal Functionality

A. Datta A. Derek J. C. Mitchell
A. Ramanathan A. Scedrov

Stanford University

University of Pennsylvania

November 10, 2005

The Problem

- ◆ Specifying security of cryptographic primitives and protocols
 - Games:
 - Between challenger and adversary
 - Defines specific moves for each player
 - Not composable
 - Example: IND-CPA, IND-CCA for encryption
 - Universal Composability[Can01, PW01]:
 - Simulation relation between real protocol and ideal functionality, which is "secure by construction"
 - Composable (Main advantage)
 - Example: SMT using trusted third party
- ◆ How are these specification methods related?

Impossibility Theorem

- ◆ If F is *any* ideal functionality for bit-commitment, then no real protocol UC-securely realizes F
- ◆ **Intuition:** Can construct information-theoretically hiding and binding protocol for BC that does not use TTP
- ◆ Similarly, symmetric encryption, group signatures,...
- ◆ **Implication of theorem:**
 - Develop other **composable** notions of security
 - Conditional composability as opposed to universal

Outline

◆ Background:

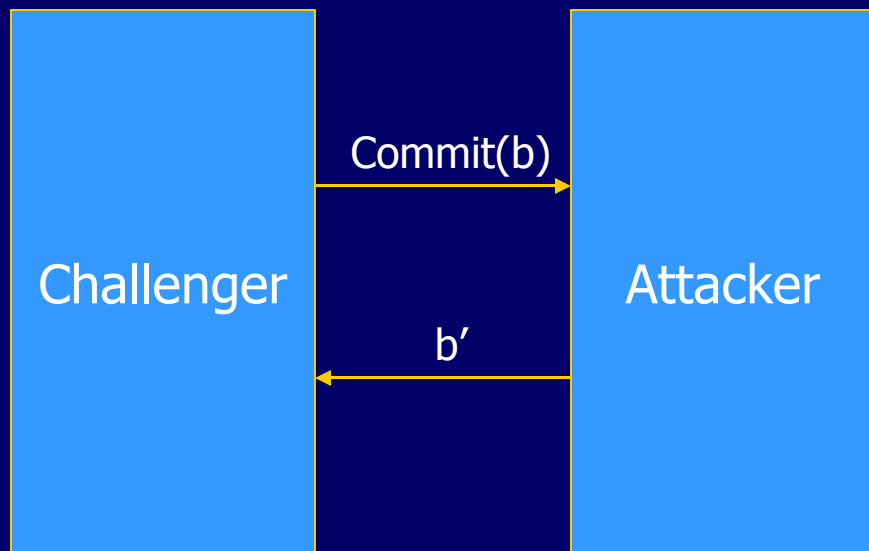
- Game-based specification
- UC-based specification
- Formalism: PPC



◆ Contribution of this work:

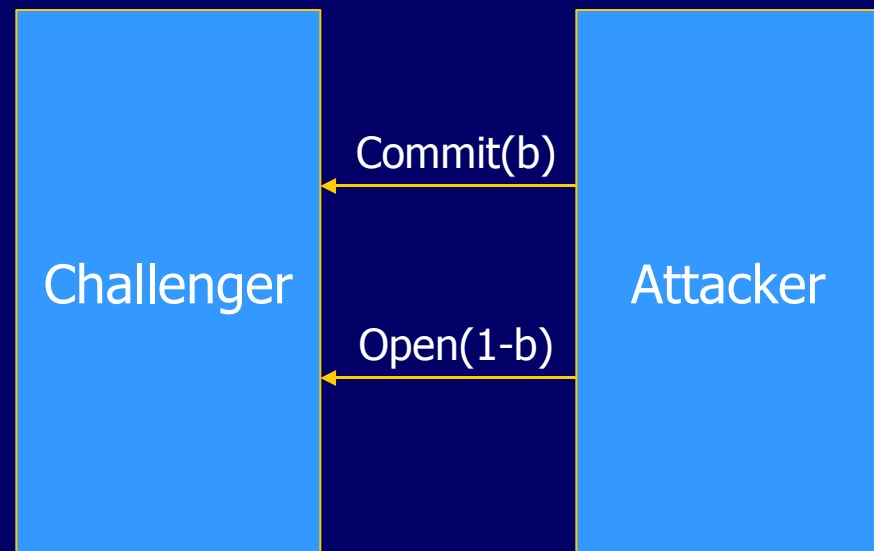
- Definition of Ideal Functionality
 - Connects UC with games
- Impossibility Theorem

Games for bit-commitment



Attacker wins if $b' = b$

Hiding Game

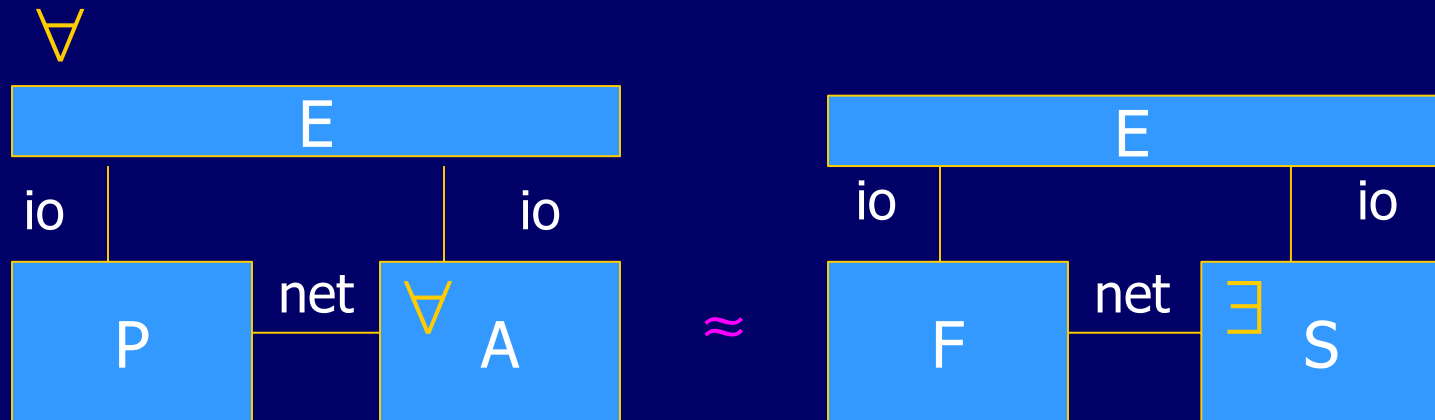


Attacker wins she can produce 2nd message

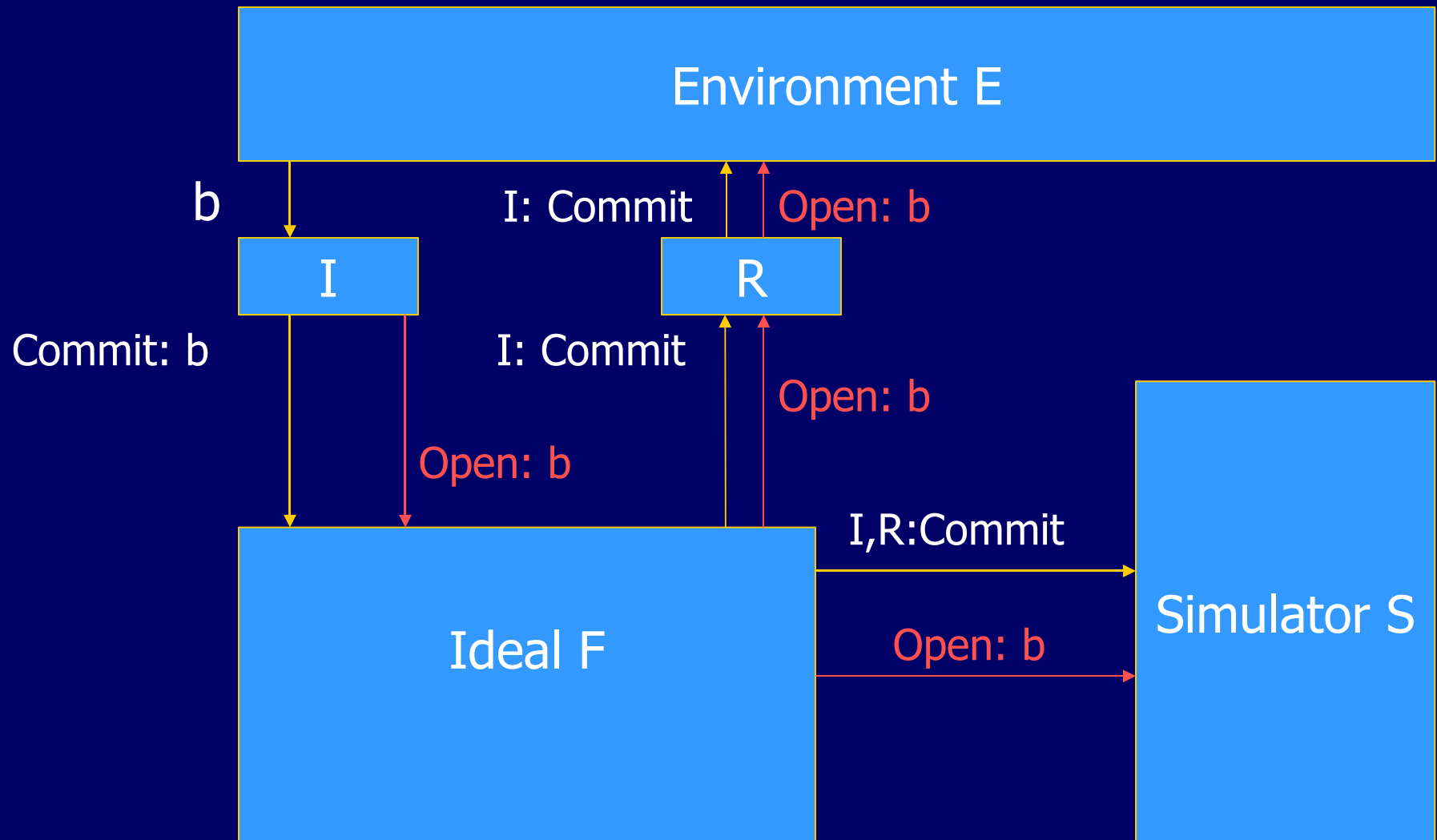
Binding Game

Functionalities (UC)

- ◆ Will use UC (Canetti). Similar idea used in Pfitzmann-Waidner
 - Two worlds: Real protocol P and Ideal functionality F
- ◆ Require
 - For every adversary A_1 for P, there exists an adversary A_2 for F revealing same information in any environment E



UC Bit-commitment specification



PPC (MMS98, LMMS98, LMMS99, MMS03, RMST04, etc.)

◆ Process Algebra

- Convenient for expressing both games and functionalities [DKMRS04,DKMR05] in the same language
- Probabilistic computation model
- Provides bit-level representations of secrets
- Can express any poly-time (in security parameter) computation or adversary

Described in previous reviews; used in this work

Outline

◆ Background:

- Game-based specification
- UC-based specification
- Formalism: PPC

◆ Contribution of this work:

- Definition of Ideal Functionality
 - Connects UC with games
- Impossibility Theorem



What is an "ideal" functionality

- ◆ Proposal: Ideal functionality for a primitive should satisfy corresponding game-conditions *information-theoretically*
- ◆ Intuition: "secure by construction"
- ◆ Example: Bit-commitment - two games for hiding and binding properties

A wrinkle

- ◆ Standard game-based definitions are given for non-interactive algorithms
 - Encryption has KeyGen, Encrypt, Decrypt
- ◆ We allow protocols
 - Need a mechanism to “call” an implementation of a protocol
- ◆ Solution: Call and return interface

Call and Return

- ◆ Principal sends a message with all params on a dedicated private channel
- ◆ Implementation listens on private channel and conducts protocol.
- ◆ Implementation returns values to principal
 - `out(impl,<params>).in(impl,<return vals>)`
| `Implementation(impl)`

2-Party Bit-Commitment (Game)

◆ 4 protocols:

- $\text{SendCommit}(b, C)$ returns σ
- $\text{GetCommit}(C)$ returns σ
- $\text{Open}(\sigma, C)$ returns ε
- $\text{Verify}(\sigma, C)$ returns $\{0, 1, \#\}$

◆ 3 properties:

- Correctness
- Hiding
- Binding

2-Party Bit-Commitment (Games)

◆ Hiding

- $\text{SendCommit}(b, C)$ returns $\sigma.\text{in}(c, b').\text{out}(c, \text{"yes"} \text{ if } b = b')$

≈

$\text{SendCommit}(b, C)$ returns $\sigma.\text{in}(c, b').\text{out}(c, \text{"yes"} \text{ 0.5 of the time})$

◆ Binding

- $\text{GetCommit}(C)$ returns $\sigma.\text{new}(b).\text{out}(b).\text{Verify}(\sigma, C)$ returns r .
 $\text{out}(c, \text{"yes"} \text{ if } r = b)$

≈

$\text{GetCommit}(C)$ returns $\sigma.\text{new}(b).\text{out}(b).\text{Verify}(\sigma, C)$ returns r .
 $\text{out}(c, \text{if } r = \# \text{ then "no" else "yes" 0.5 of the time})$

◆ Correctness

Ideal Functionality for BC

Any implementation of the calling interface that satisfies the games

INFORMATION THEORETICALLY

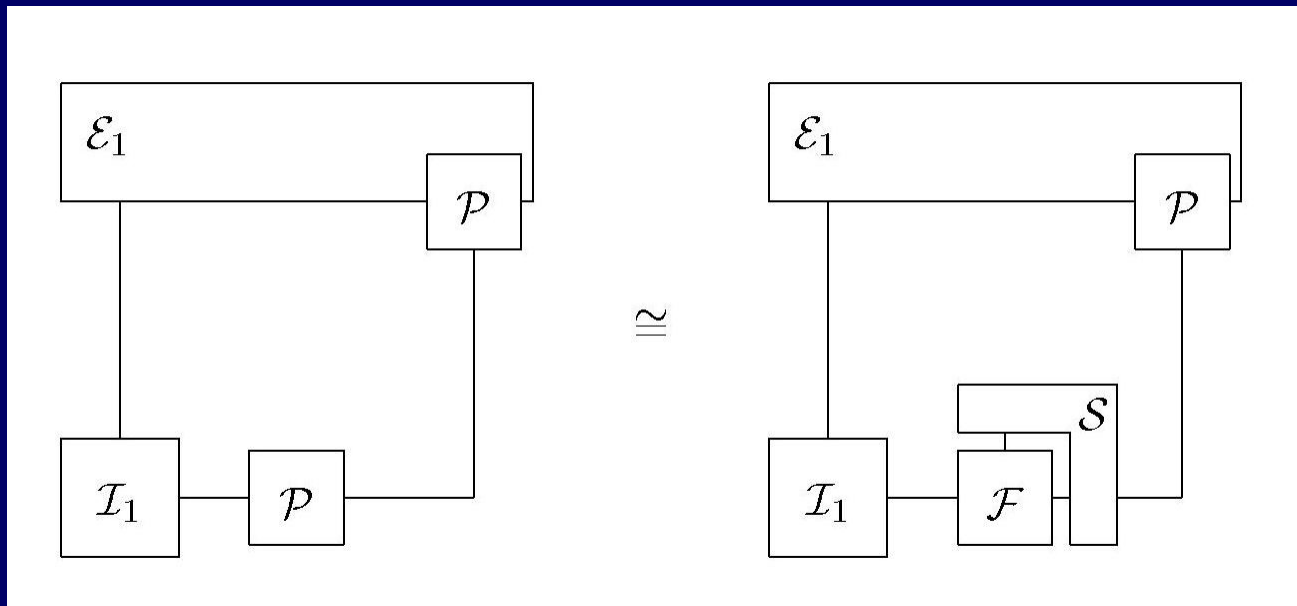
Intuition: Secure by construction

(may use unrealistic mechanism like TTP, secure and authenticated channels)

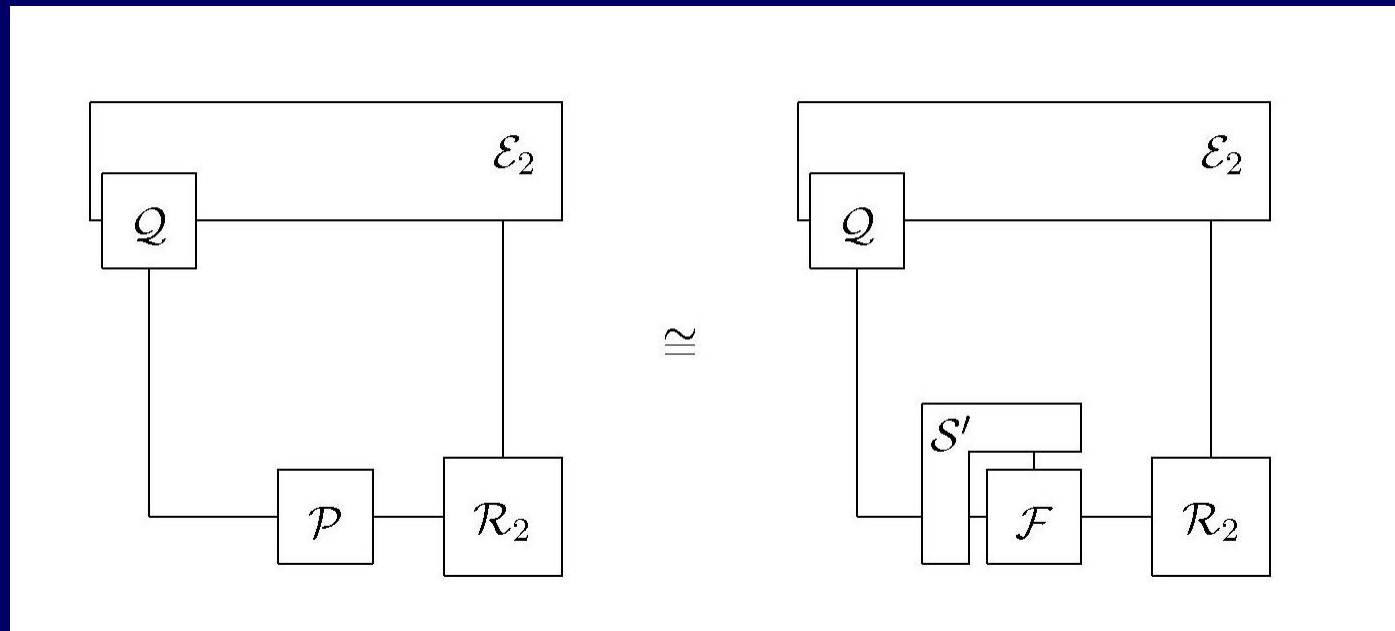
Impossibility Theorem

- ◆ If F is *any* ideal functionality for bit-commitment, then no real protocol UC-securely realizes F
- ◆ Proof idea: Can construct information-theoretically hiding and binding protocol for BC that does not use TTP

Proof, phase 1 of



Proof, phase 2 of

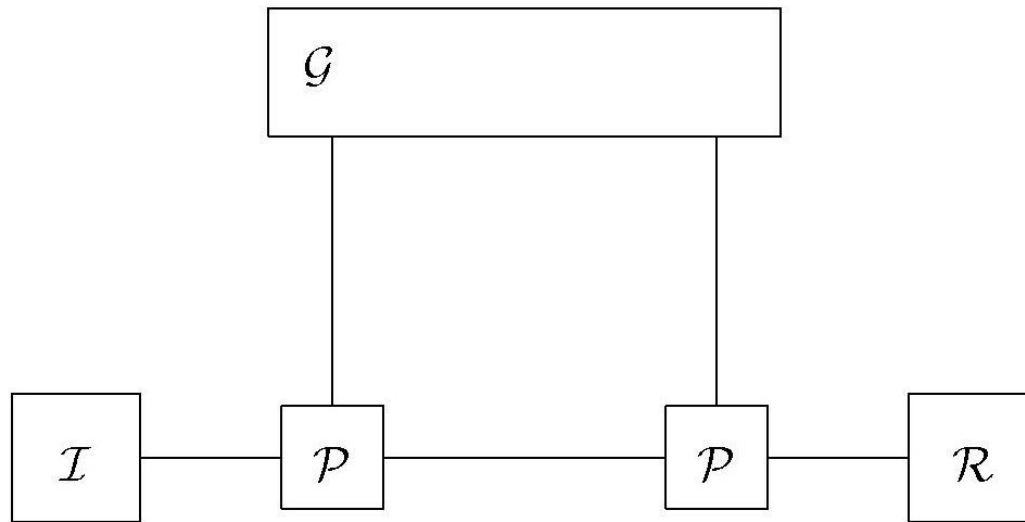


Payoff, the big

- ◆ So Q and S' and F together constitute a real implementation for BC that is
 - Info-theoretically binding
 - Info-theoretically hiding
 - Correct

Reductions

- ◆ Can show that any property that gives BC can't be realized
 - Uses reductions



Other things you can't do

◆ Variant of Symmetric encryption

- Semantic security and Ciphertext integrity

◆ Variant of Group signatures

- Anonymity and Traceability (strong variant)

Related work

◆ Bit-commitment

- For a particular F , no protocol securely realizes F [CF2001]
 - Allows Canetti to reason about what the simulator must do
 - Shows that simulator does not have enough info. to simulate

◆ Zero-knowledge, secure function evaluation, oblivious transfer

- Similar results

Conclusions and Future Work

- ◆ UC-security cannot be achieved for important cryptographic tools
- ◆ Need for alternative approaches to compositional security
 - More general versions of ideal functionalities
 - Modification of UC framework
 - Conditional composability instead of universal composability

Questions?
