Advanced Tools from Modern Cryptography

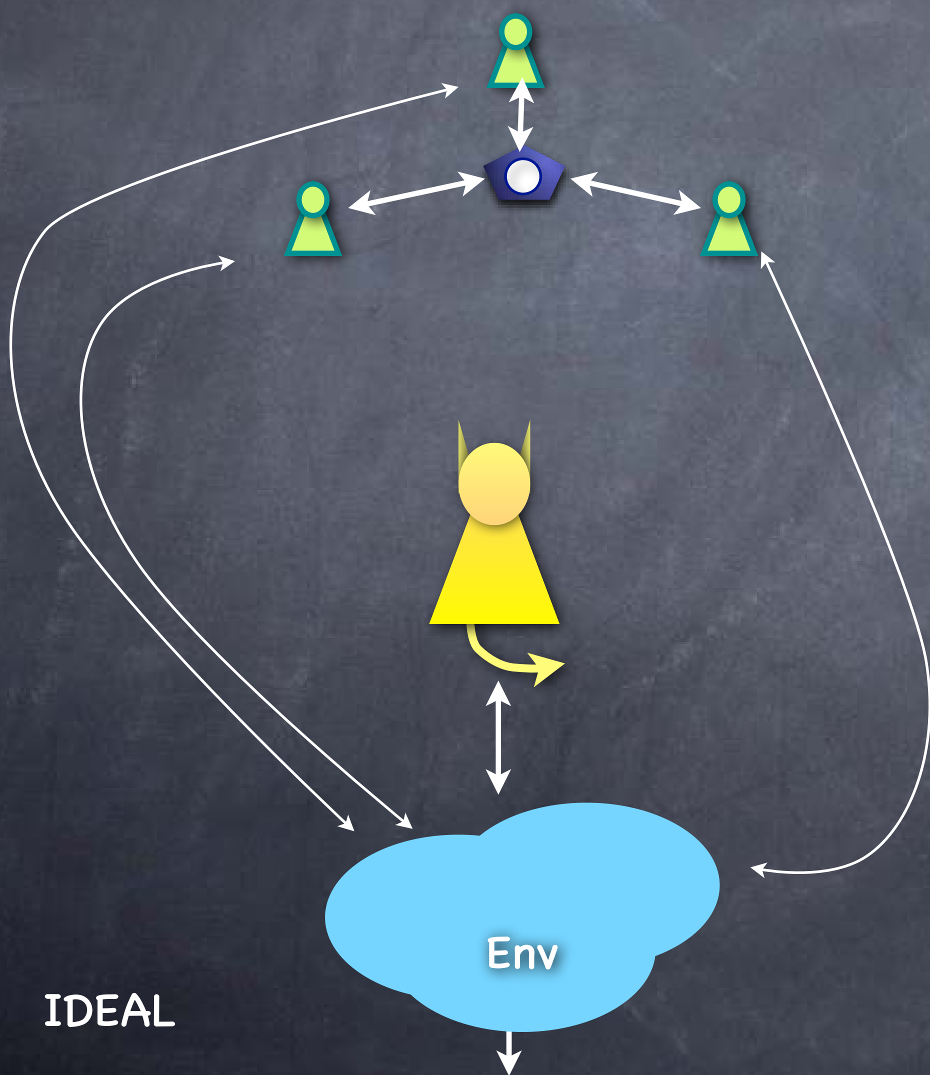
Lecture 11

MPC: UC Theorem. UC Limitations.

RECALL

UC Security

REAL is as secure as IDEAL if:



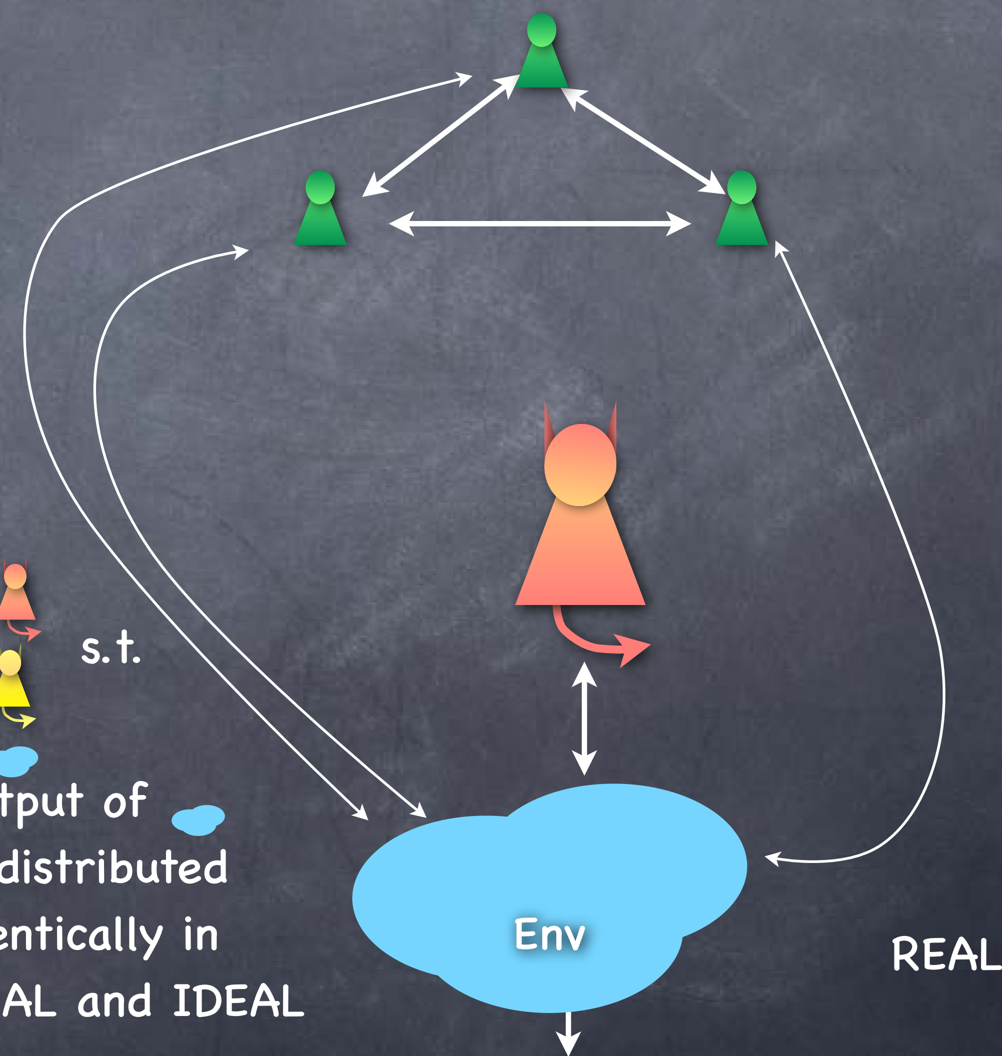
∀

∃ s.t.

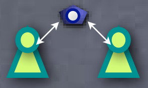
∀

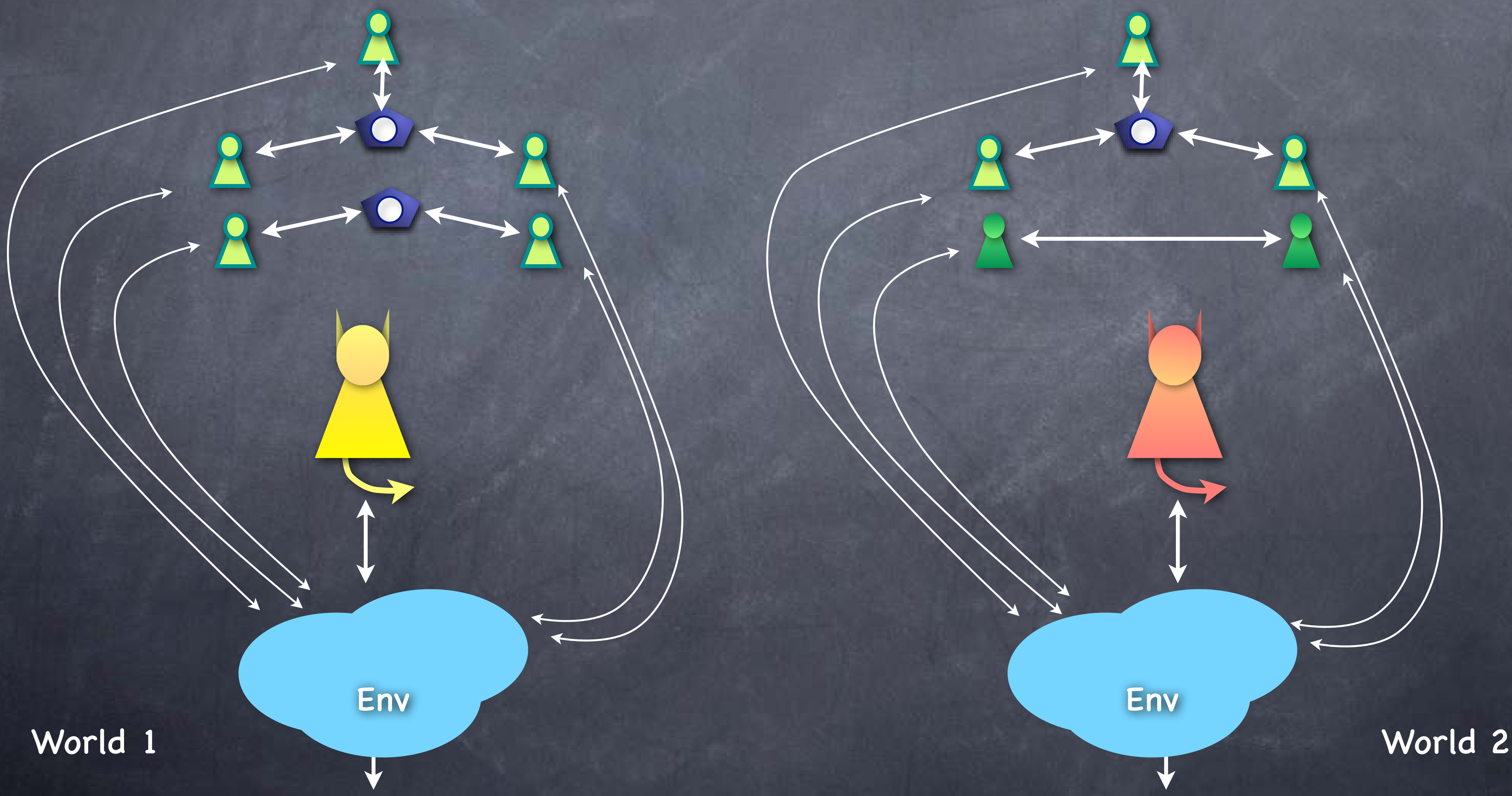
output of

is distributed identically in REAL and IDEAL



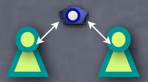
Universal Composition

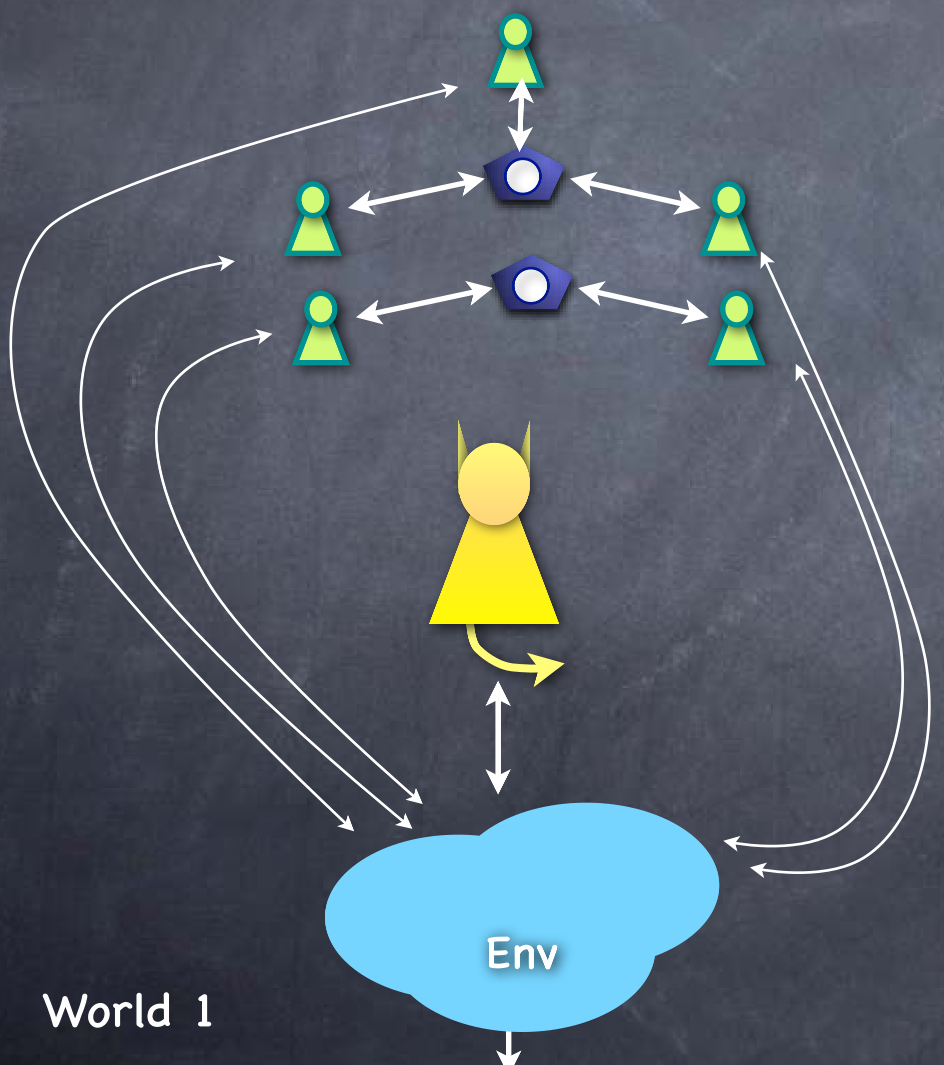
Replace protocol  with  which is as secure, etc.

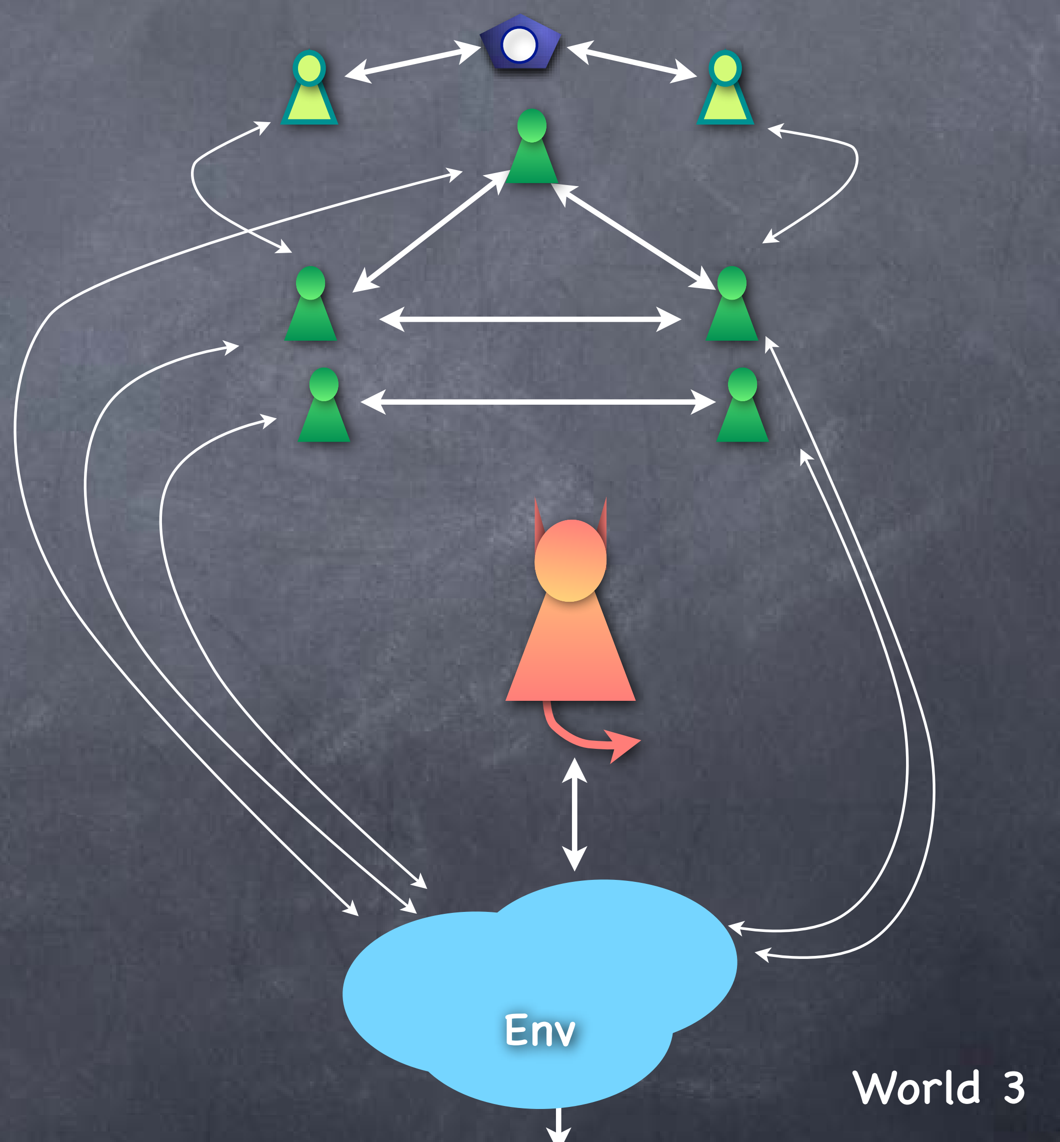


RECALL

Universal Composition

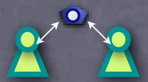
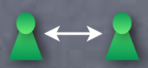
Replace protocol  with  which is as secure, etc.

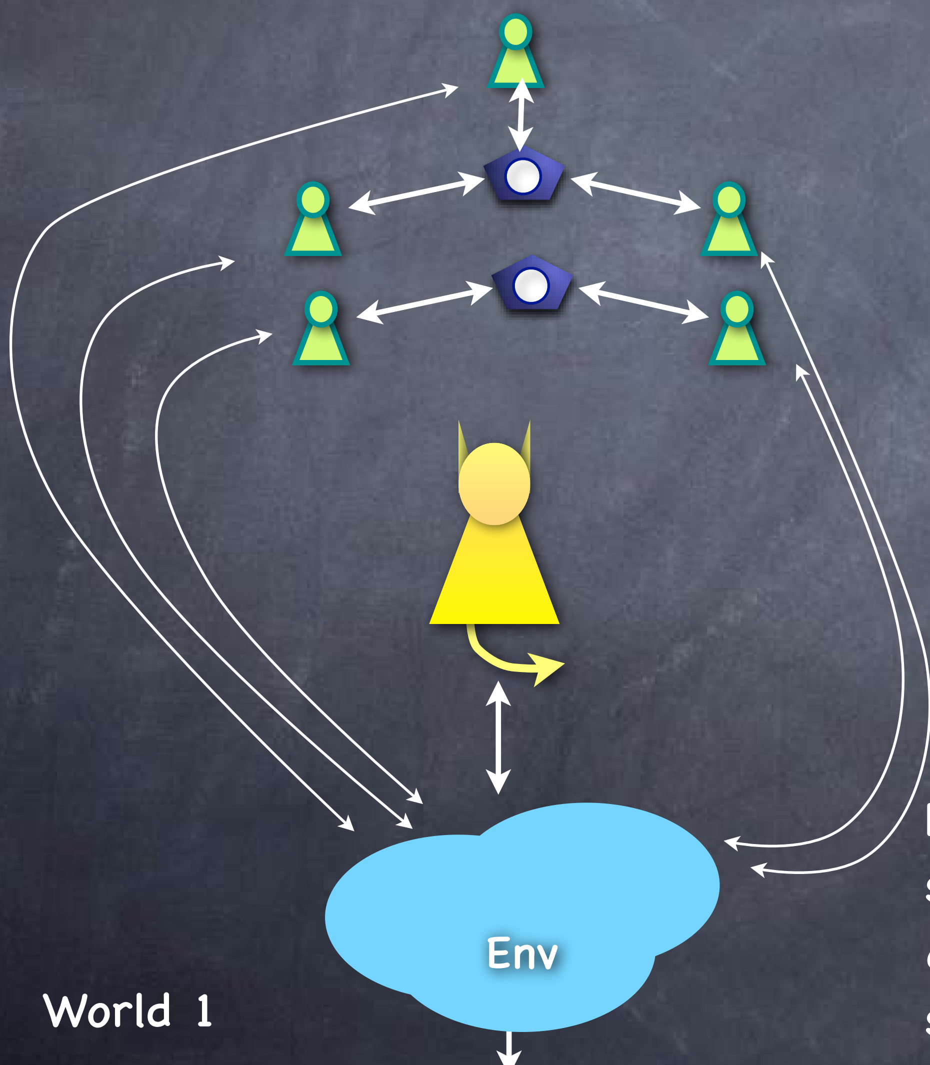




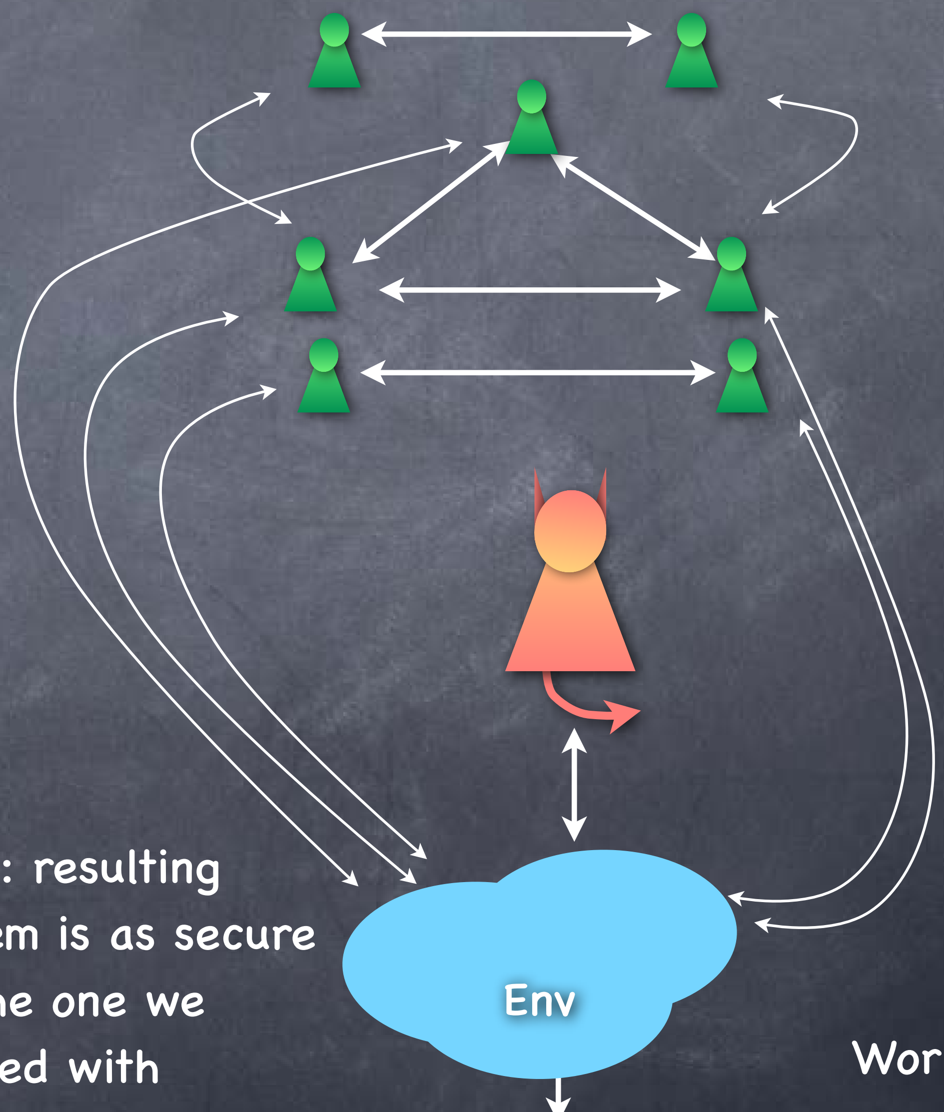
Universal Composition

RECALL

Replace protocol  with  which is as secure, etc.



Hope: resulting system is as secure as the one we started with



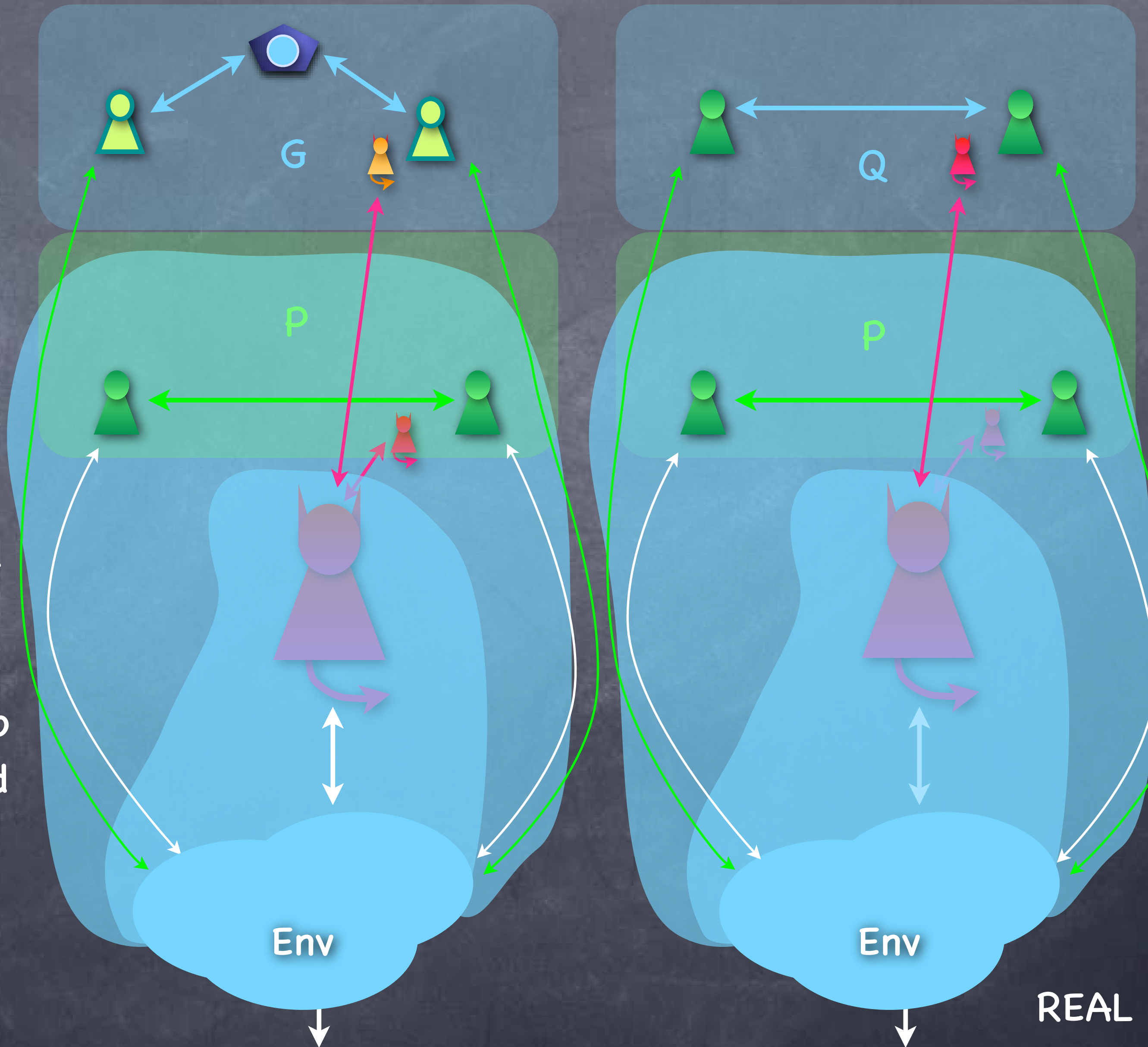
RECALL

Universal Composition

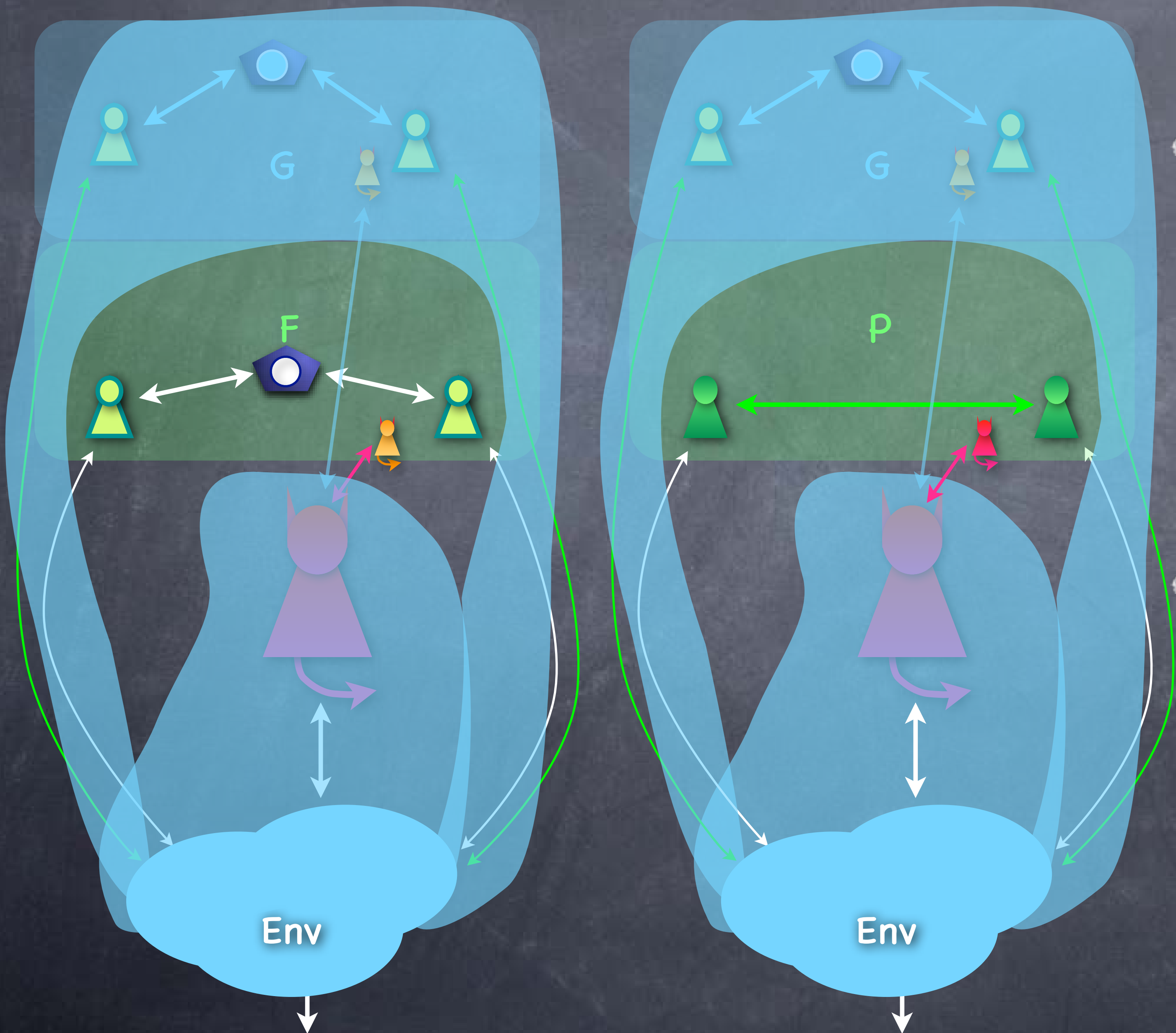
- Start from world A (think "IDEAL")
  - Repeat (for any poly number of times):
    - For some 2 "protocols" (that possibly make use of ideal functionalities) I and R such that R is as secure as I, substitute an I-session by an R-session
  - Say we obtain world B (think "REAL")
  - UC Theorem: Then world B is as secure as world A
- Gives a modular implementation of the IDEAL world

Proving the UC theorem

- Consider the environment which runs the adversary internally, and depends on "dummy adversaries" to interface with the protocols
- Now consider the new environment s.t. only Q (and its adversary) is outside it
- Use "Q is as secure as G" to get a new world with G and a new adversary

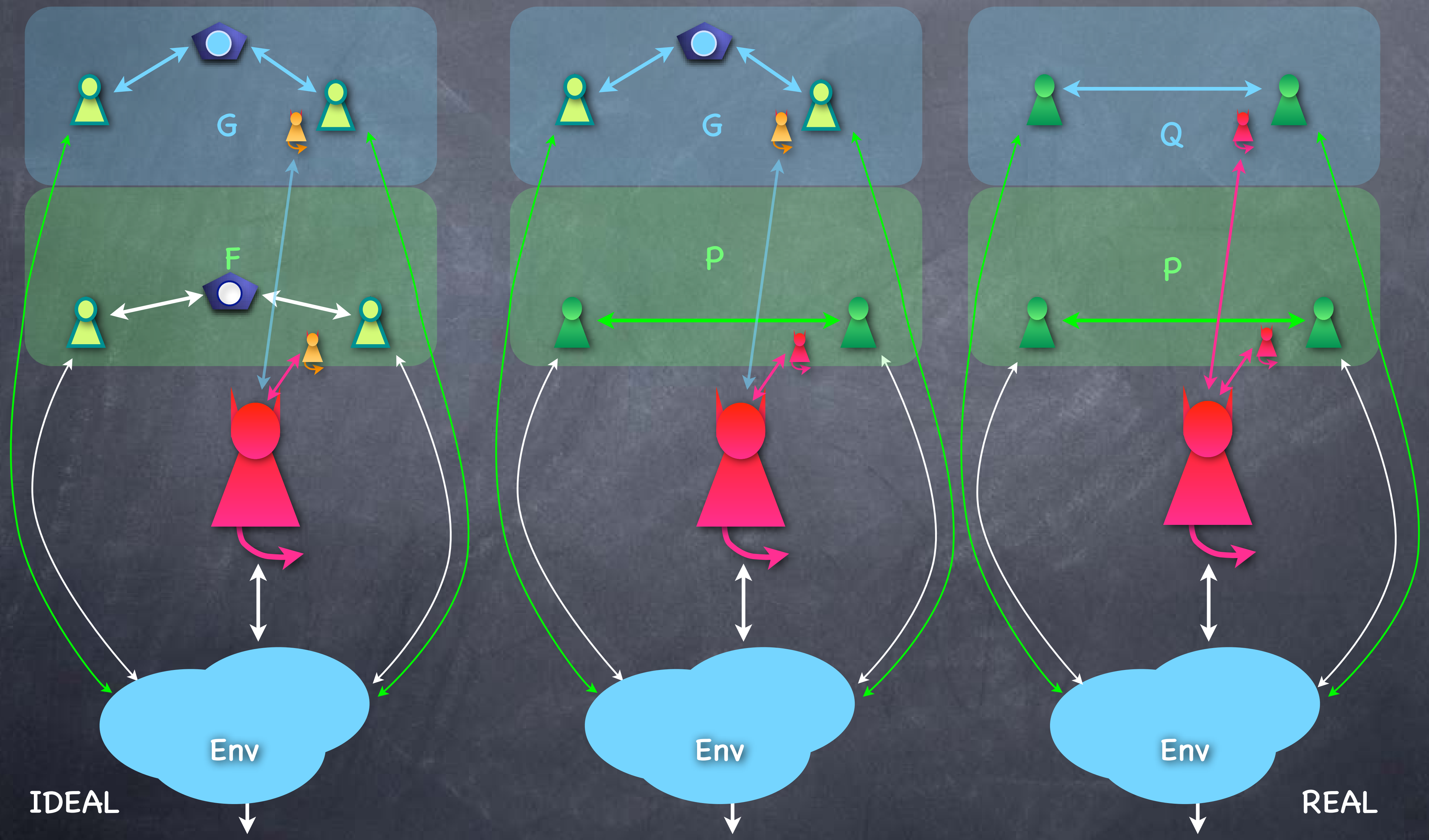


Proving the UC theorem

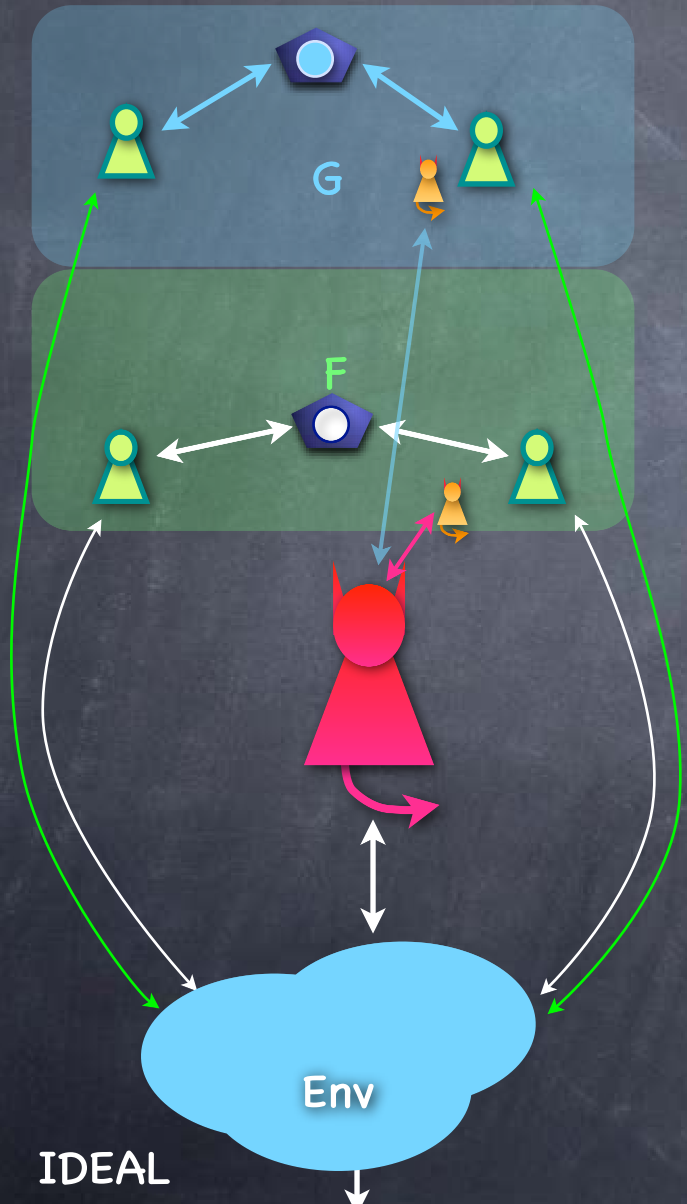


- Now consider the new environment s.t. only P (and adversary) is outside it
    - Note: G and simulator for Q/G are inside the new environment
- Use "P is as secure as F" to get a new world with F and a new adversary

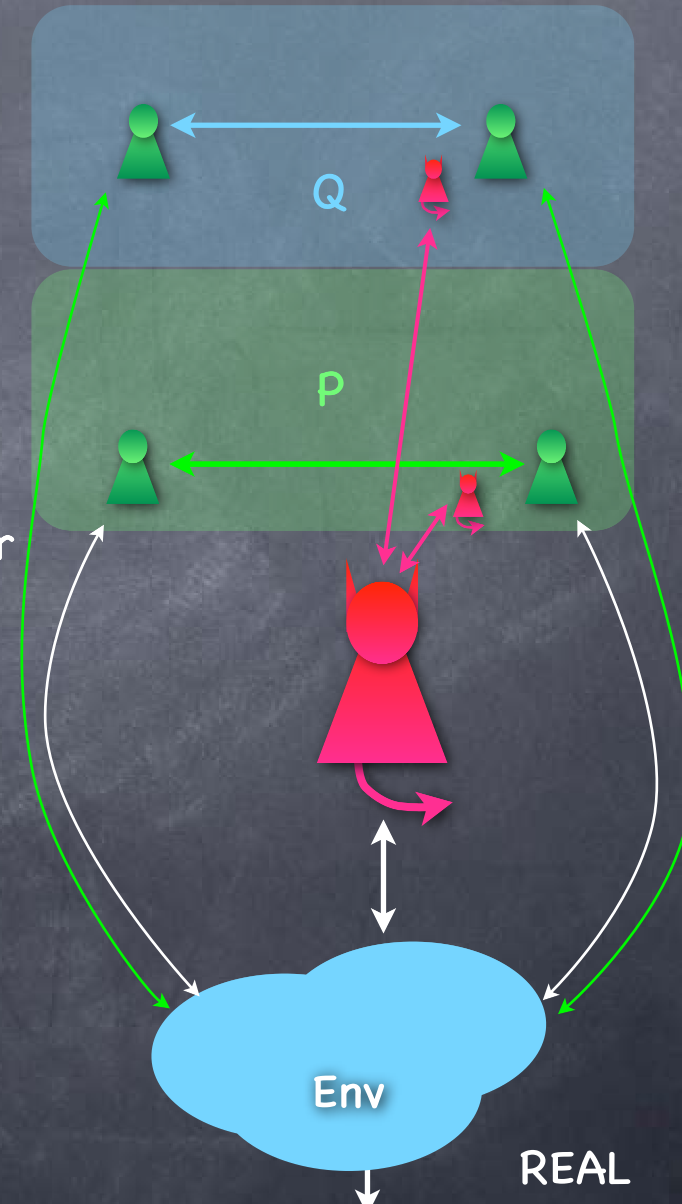
Proving the UC theorem



Proving the UC theorem



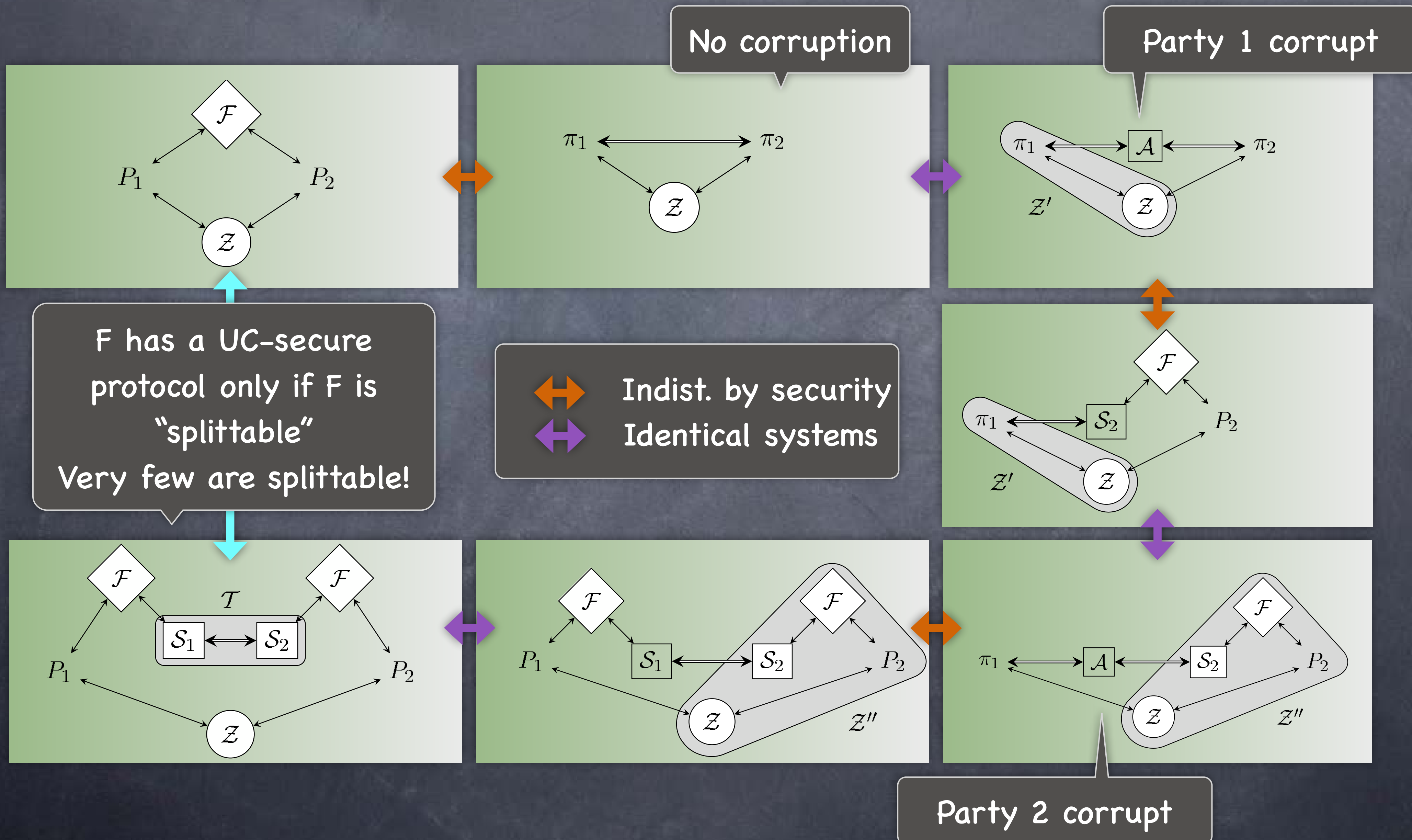
- Hence REAL ≈ IDEAL
- Main idea: Environment can model other sessions (real or ideal)



UC Secure MPC?

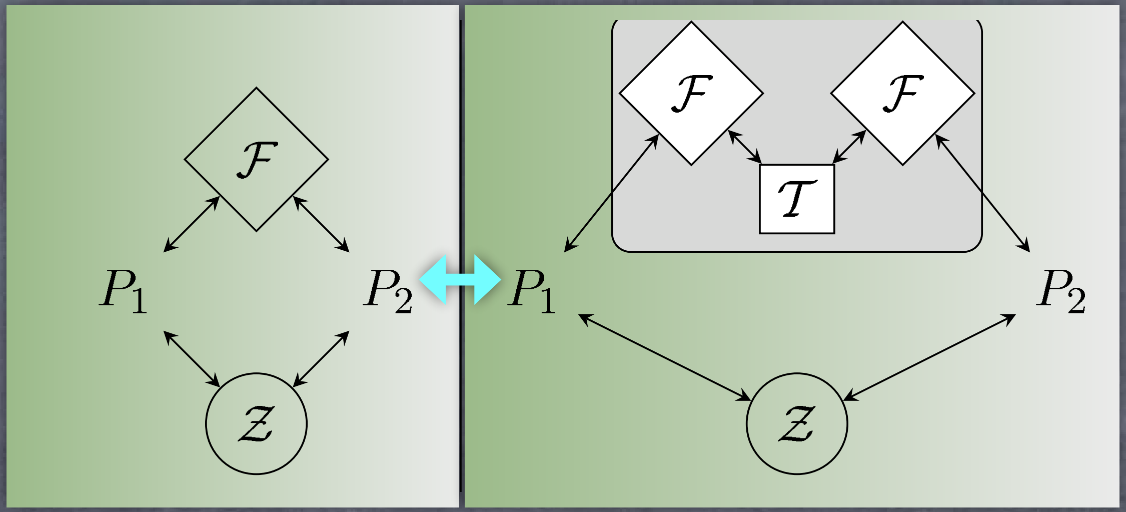
- UC-security is a strong security definition, and also enjoys the UC property
- But impossible to have "non-trivial" UC-secure MPC (for 2 parties)!
- Universal Composition possible when:
  - Passive corruption, or
  - Honest majority, or
  - Given trusted setups (e.g., OT, Common Reference String), or
  - Using alternate security definitions
    (e.g., "Angel-aided simulation": still meaningful and UC)

Impossibility of UC Security



Splittable Functionalities

- F splittable if ∃T ∀Z the outputs of Z in the following two experiments are negligibly far from each other:



- A splittable functionality essentially involves only communication and local computation. All splittable functionalities have UC-secure protocols.
- Most interesting functionalities are unsplittable. E.g., coin-tossing, commitment, XOR, OT, ...

UC Security Beyond 2 Parties Without Honest-Majority

- Any multi-party function F such that a 2-way partition of it is unsplittable is impossible to UC-securely realise
  - Consider F with an unsplittable partition f. Protocol Π_F gives a 2-party protocol Π_f. Π_F tolerates corruption of either part $\rightarrow$ Π_f tolerates corruption of either party
- So only "disseminating" and "aggregating" functionalities
- Disseminating: Only one party has input that influences the output of the others (e.g., broadcast, secret-sharing)
- Aggregating: Only one party has output that is influenced by the input of the others (e.g., group summation)

UC Security Beyond 2 Parties

- All disseminating functionalities are UC-securely realisable!
  - e.g., Broadcast protocol
    - Sender sends m to all Receivers
    - Each Receiver sends m that it received to all others
    - Each Receiver outputs m if it received the same m from all other Receivers. Else Aborts.
  - Note: Here selective abort allowed. UC-Secure [Why?]
- Open: which aggregating functionalities are UC-securely realisable?
  - e.g. additive-sharing based summation protocol (input parties play servers, only one output client) [Why UC-Secure?]