State-Separating Proofs
A Reduction Methodology for Real-World Protocols

Chris Brzuska
Antoine Delignat-Lavaud
Konrad Kohbrok
Markulf Kohlweiss

I sometimes suffer.

And the goal of this paper is to ease my suffering.

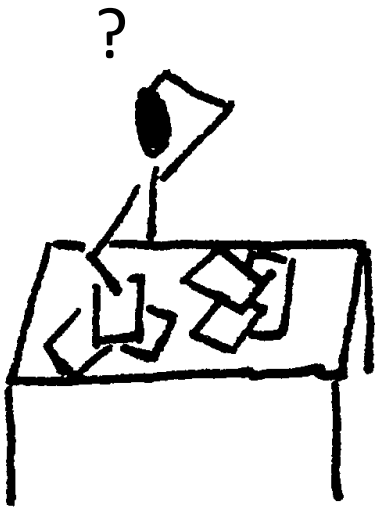
Key exchange researchers sometimes suffer.

And the goal of this paper is to ease our suffering.

Once upon a time…

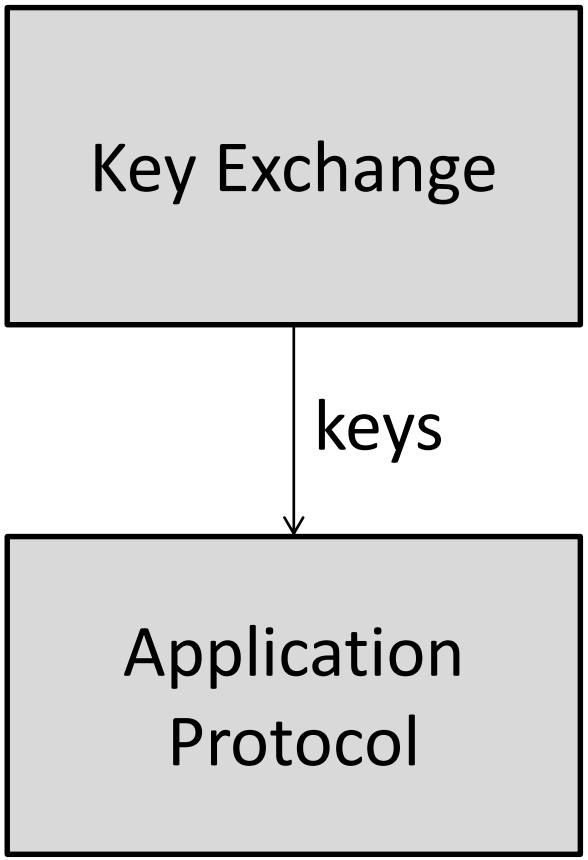
…I was a PhD student and I worked on key exchange.

At night, when I wrote proofs, I suffered.



a) tons of work
b) many *seemingly* simple steps
c) not human-verifiable.

…one of the worst proofs was to prove that Bellare-Rogaway secure key exchange protocols are composable.

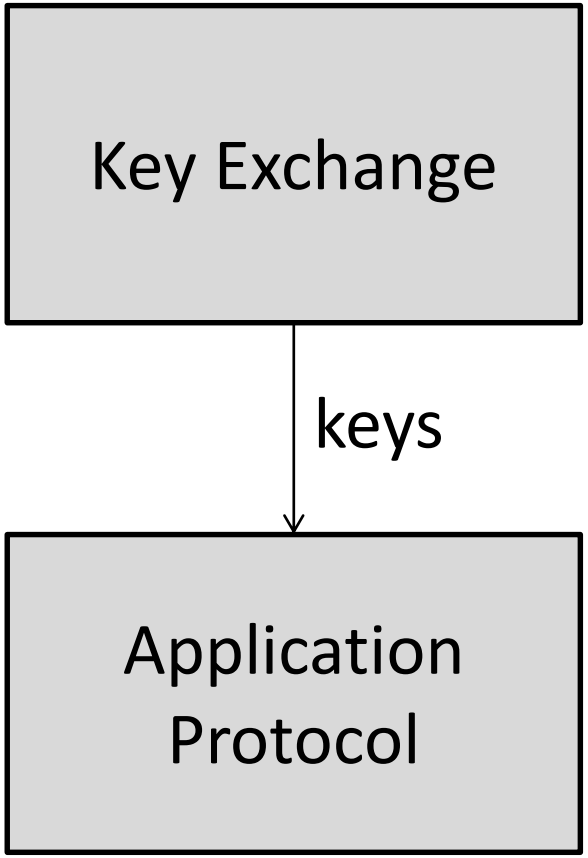


Outputs symmetric keys that look random.

Is secure with random keys

What I was really concerned about: The theorem seems so simple, why is the proof so hard?

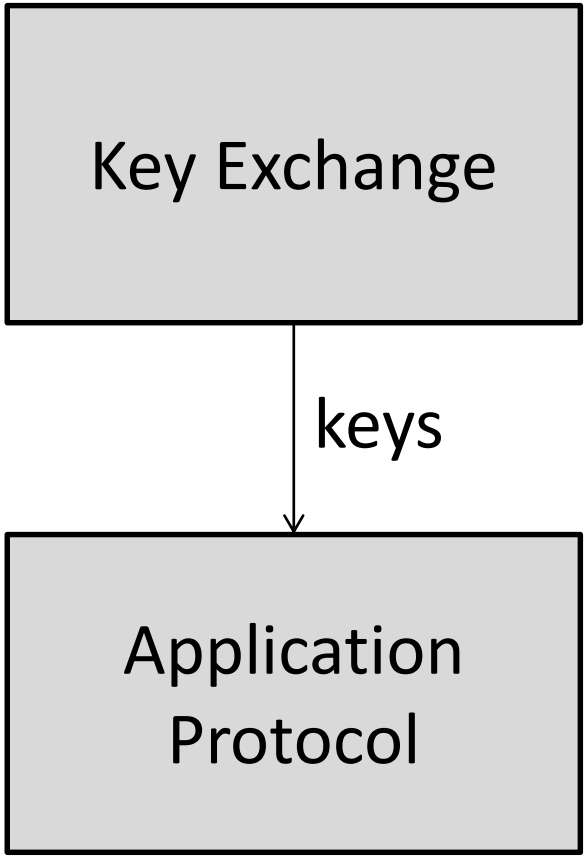
Tried in
CCS 2011 paper, my thesis, Stephen Williams's thesis…



Outputs symmetric keys that look random.

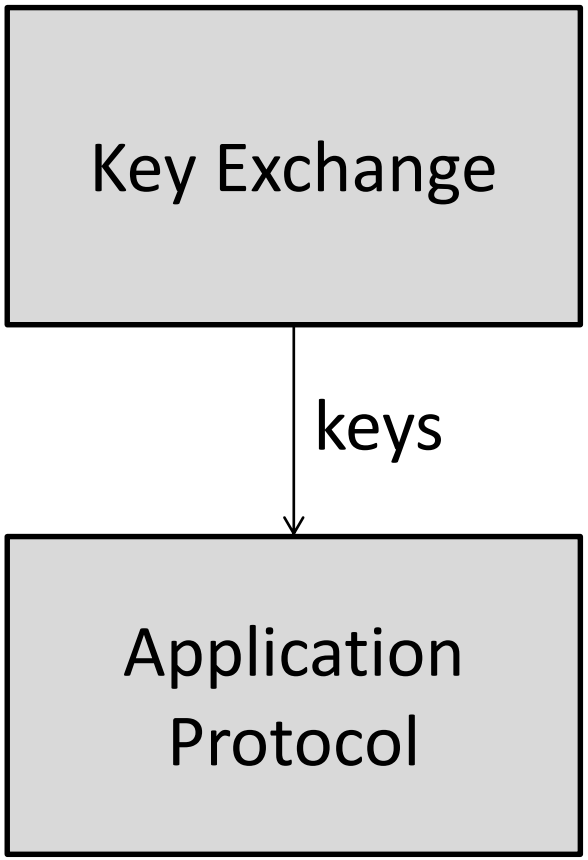
Is secure with random keys

What is the difficulty?



Outputs symmetric keys that look random.

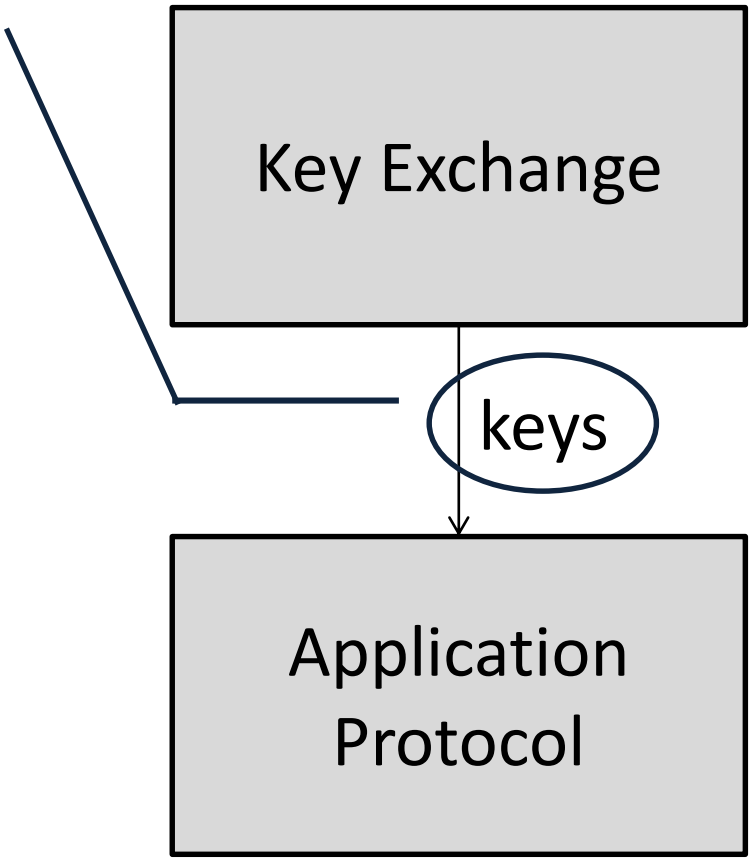
Is secure with random keys



Outputs symmetric keys that look random.

Is secure with random keys

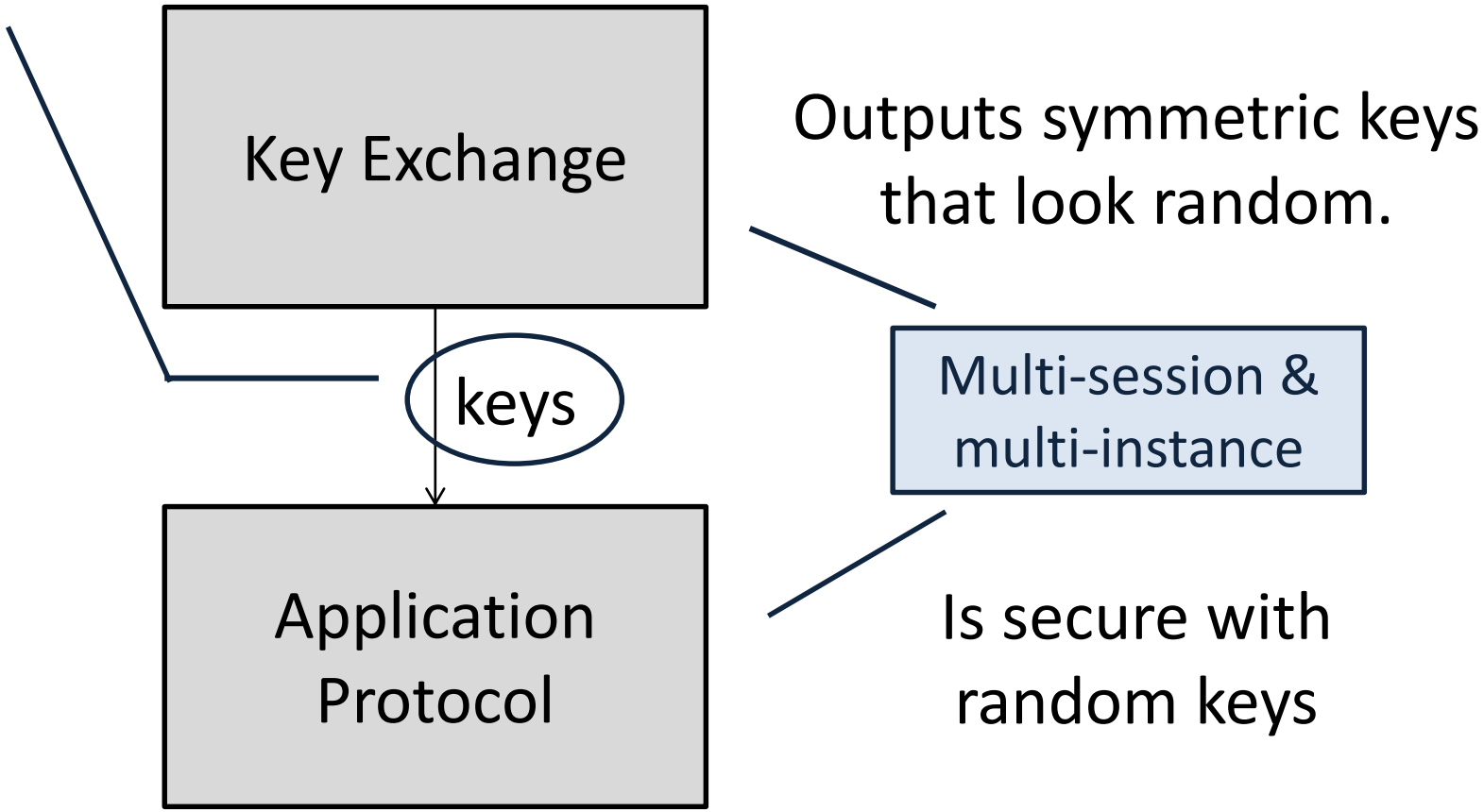
State is passed from one game to another, *defining* the composition is already annoying.

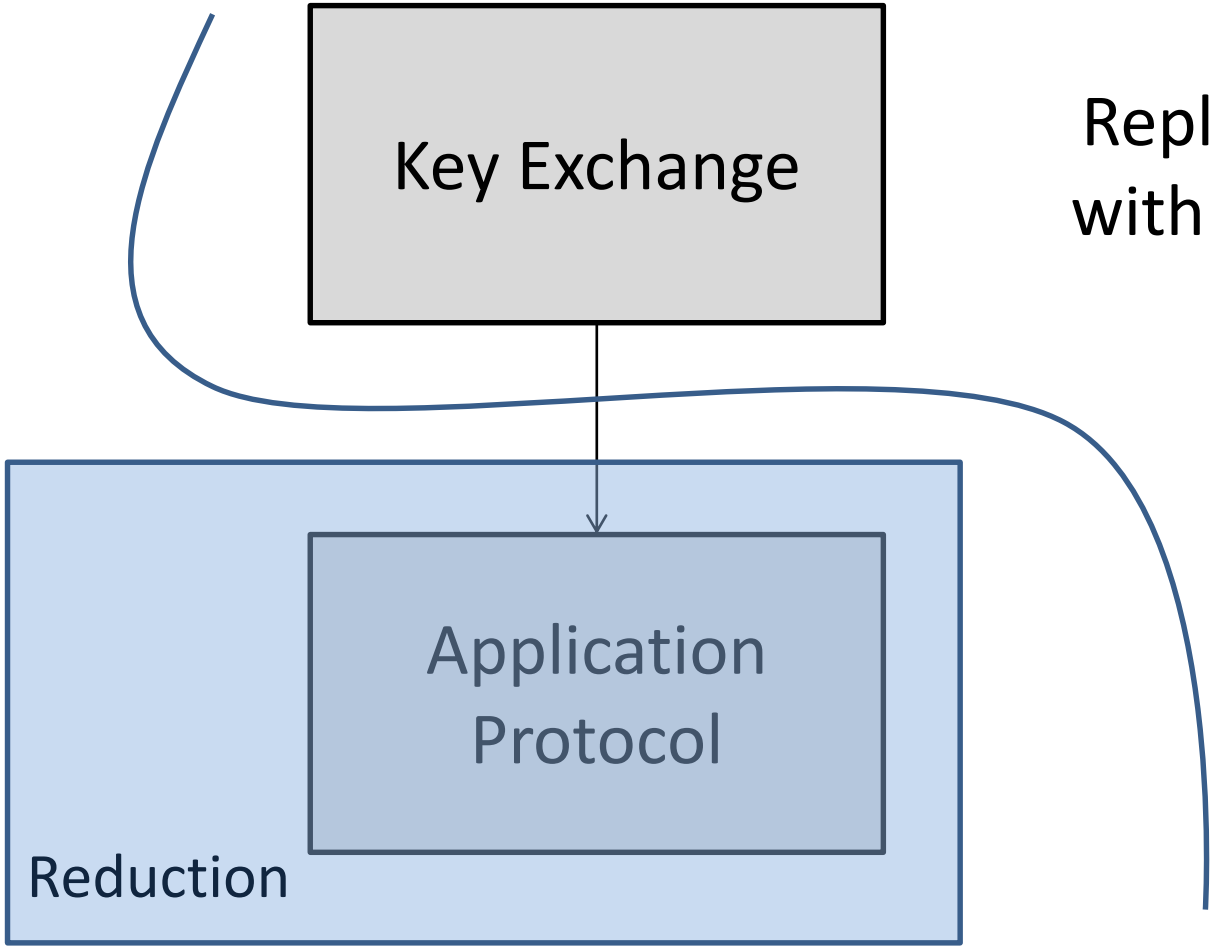


Outputs symmetric keys that look random.

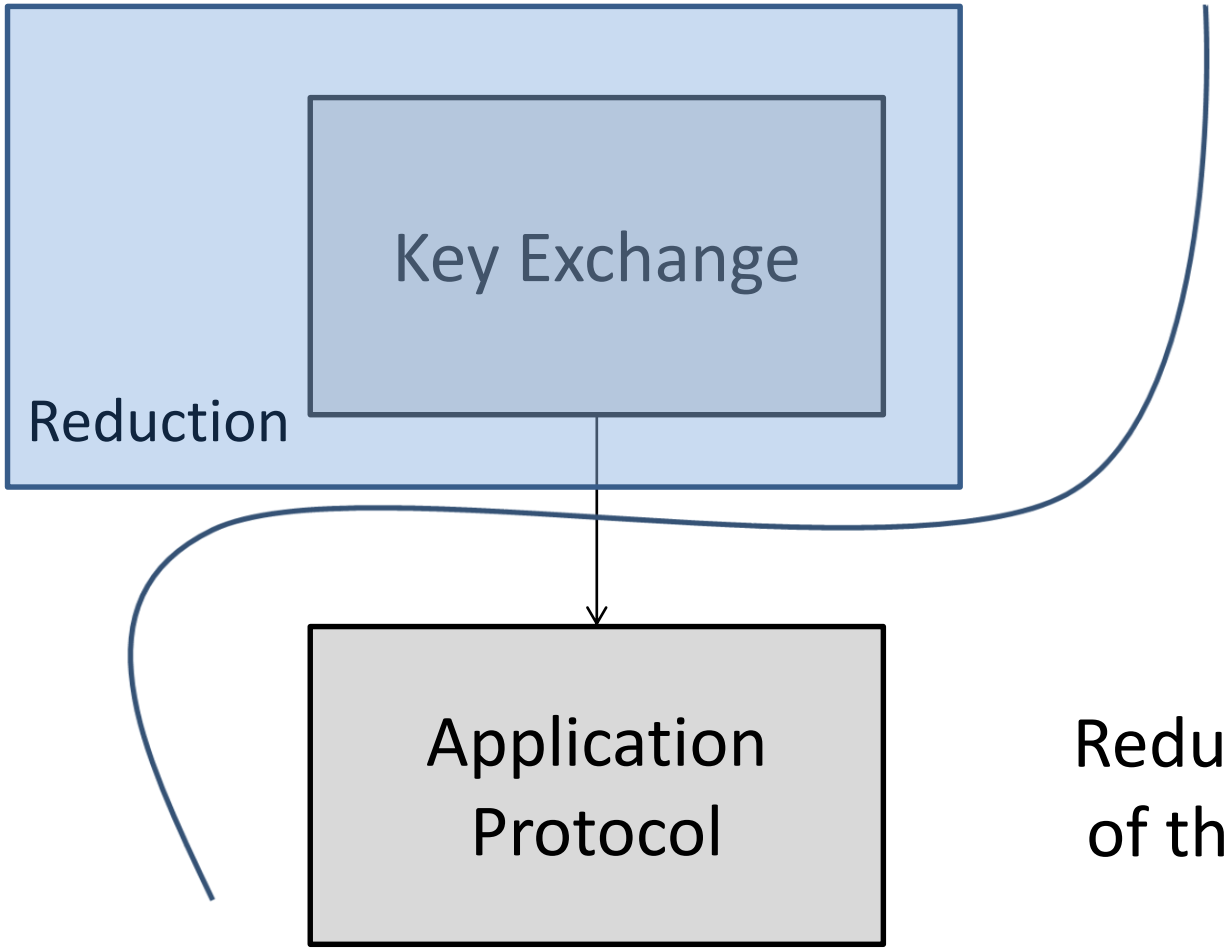
Is secure with random keys

State is passed from one game to another, *defining* the composition is already annoying.

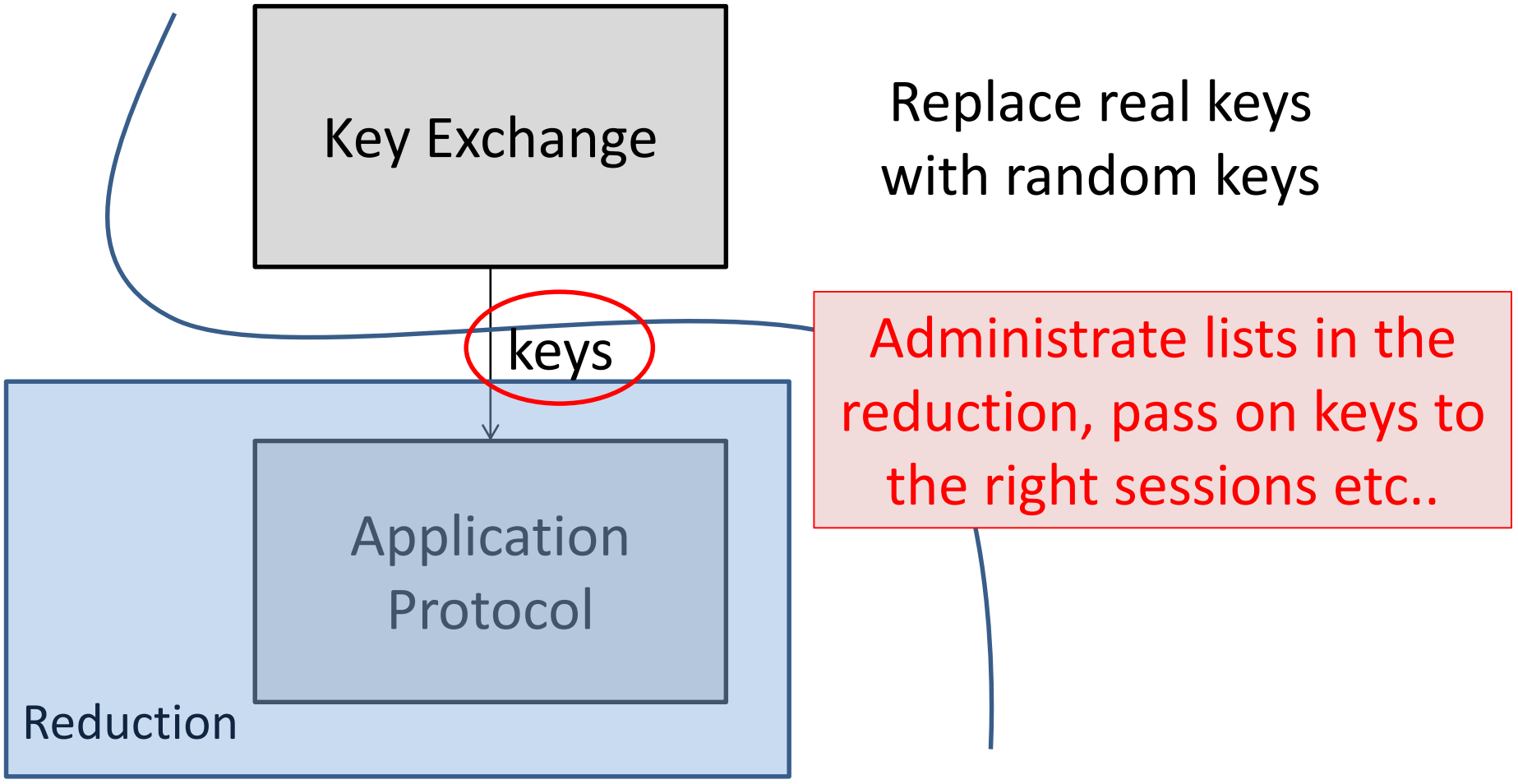


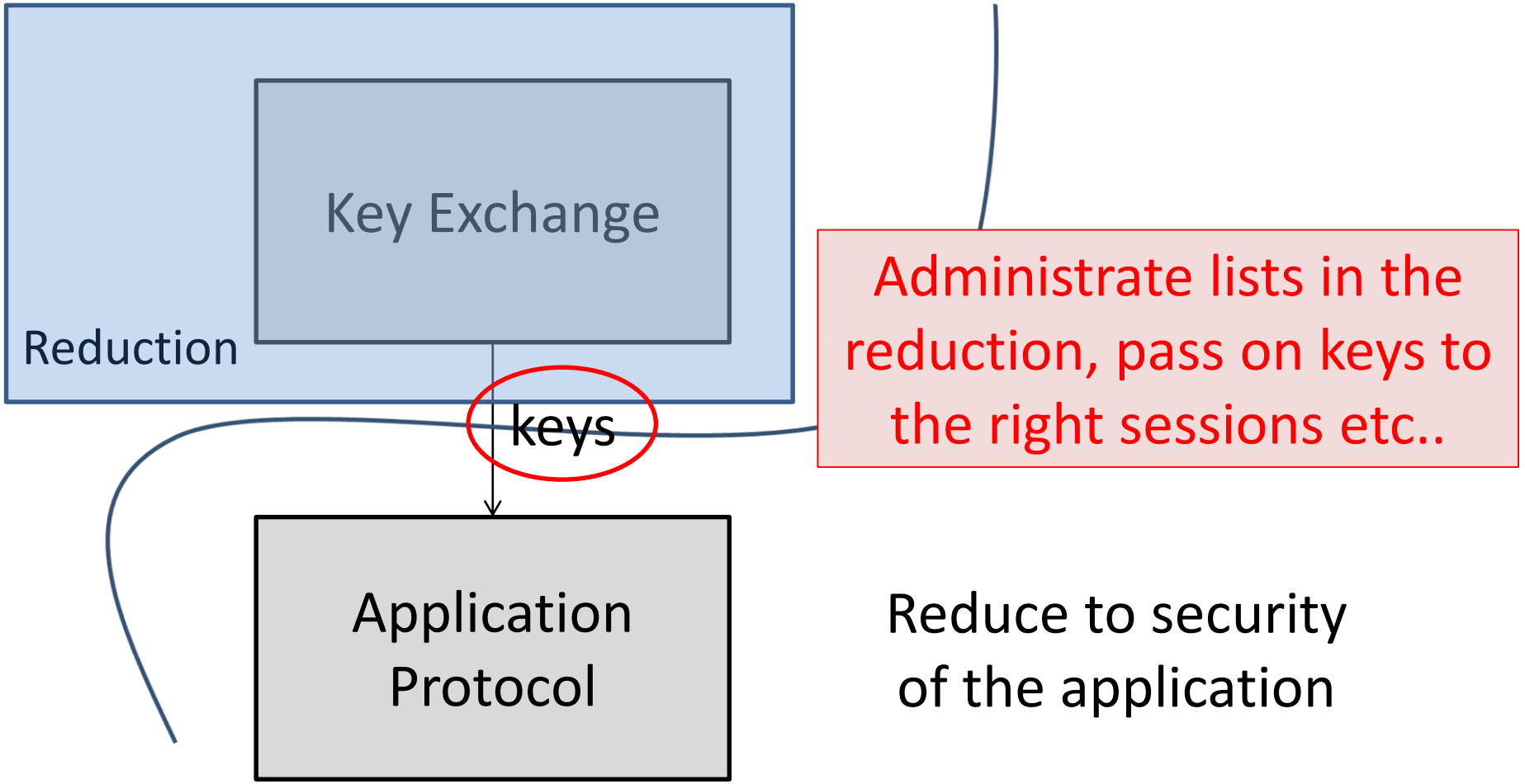


Replace real keys with random keys

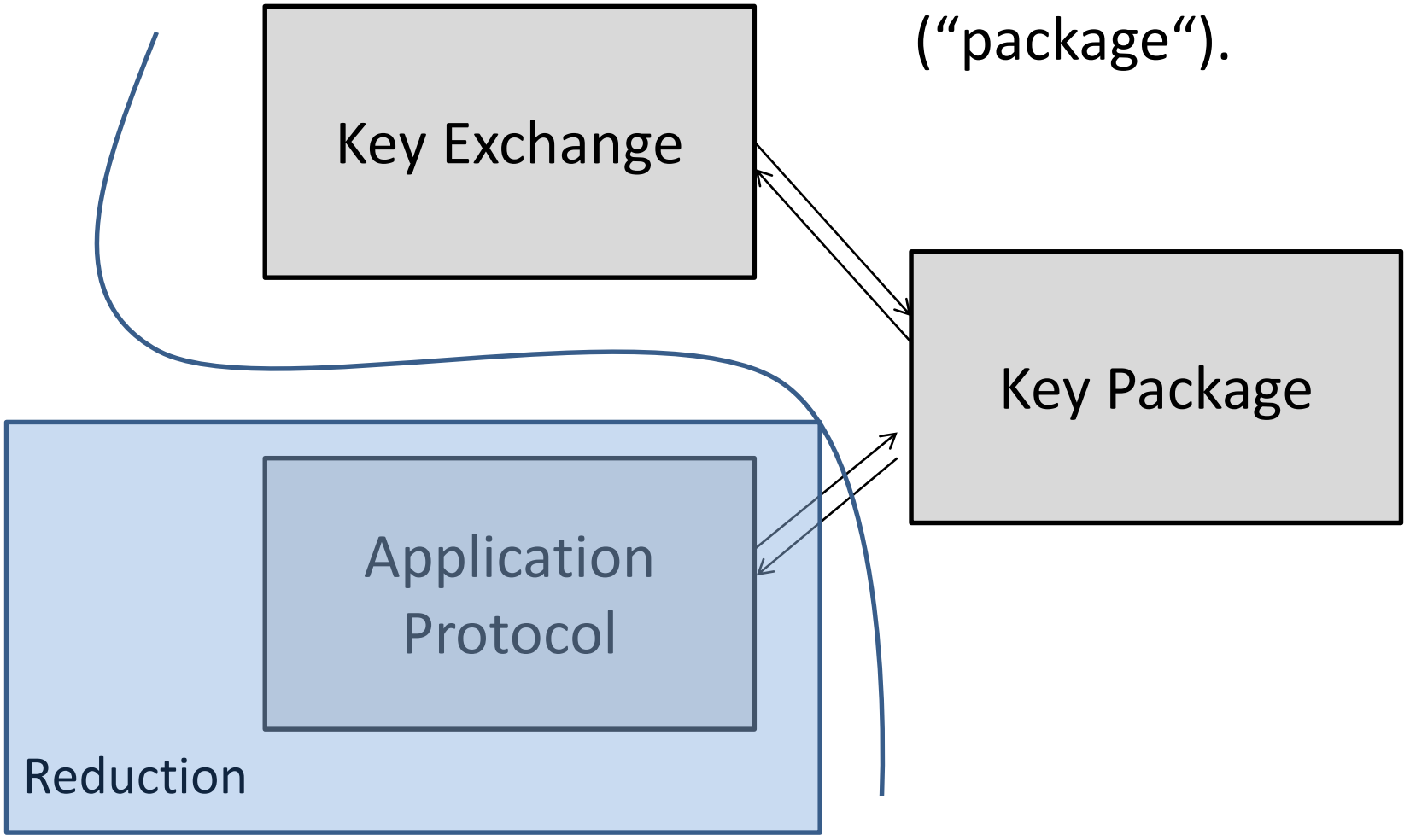


Reduce to security
of the application

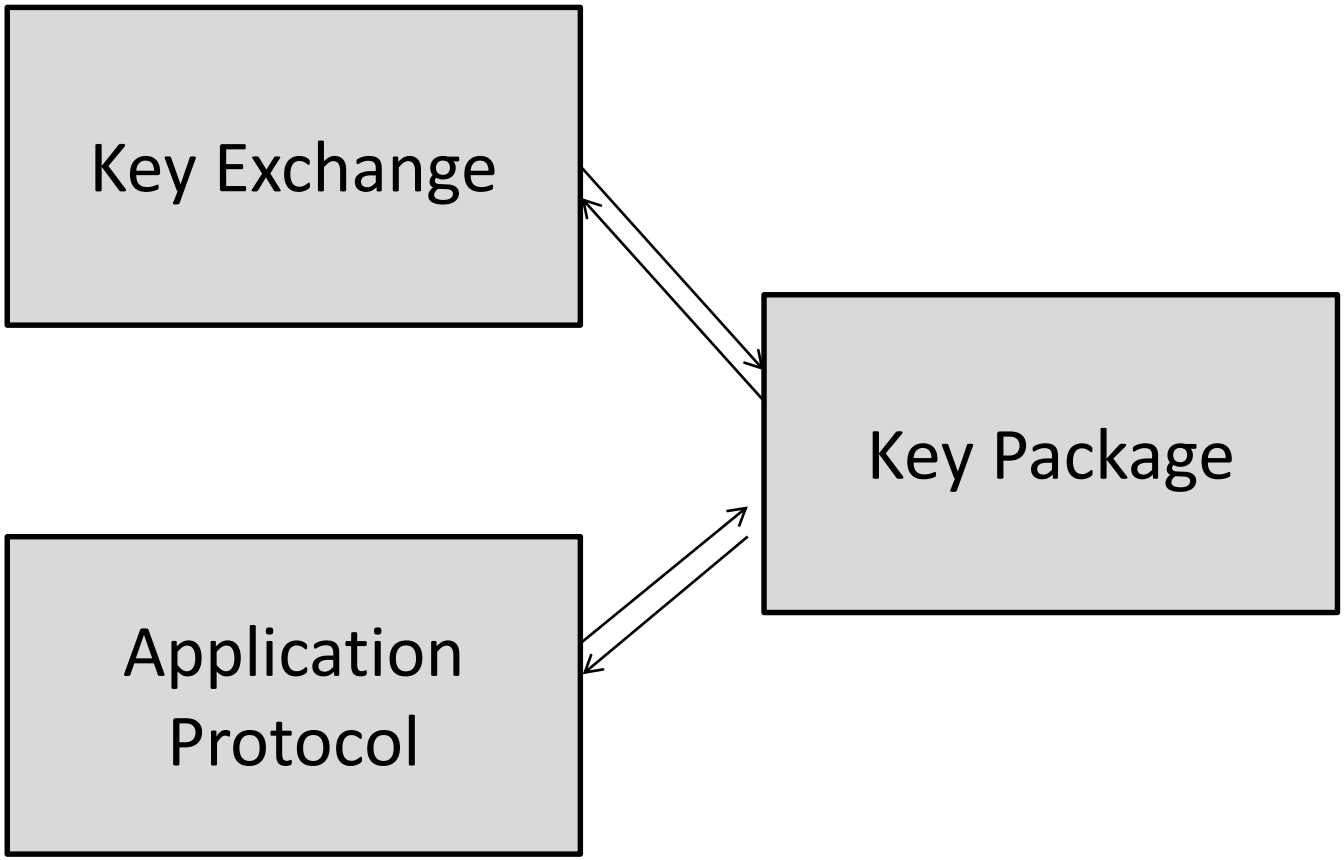

Key Exchange
keys
Application Protocol
Reduction
Replace real keys with random keys
Administrate lists in the reduction, pass on keys to the right sessions etc..


Key Exchange
Reduction
keys
Administrate lists in the reduction, pass on keys to the right sessions etc..
Application Protocol
Reduce to security of the application

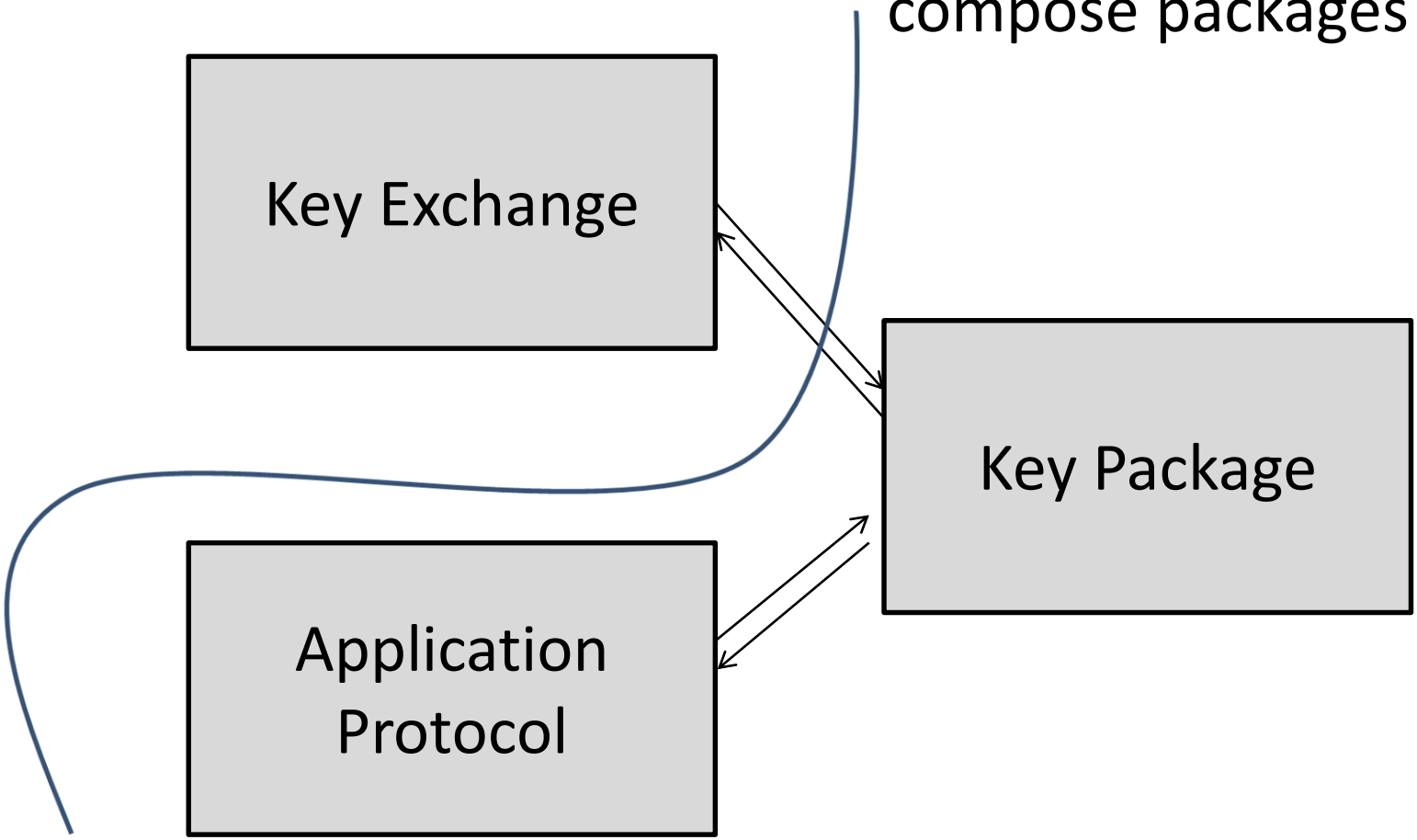
Idea 1: Move shared state to an external algorithm (“package“).



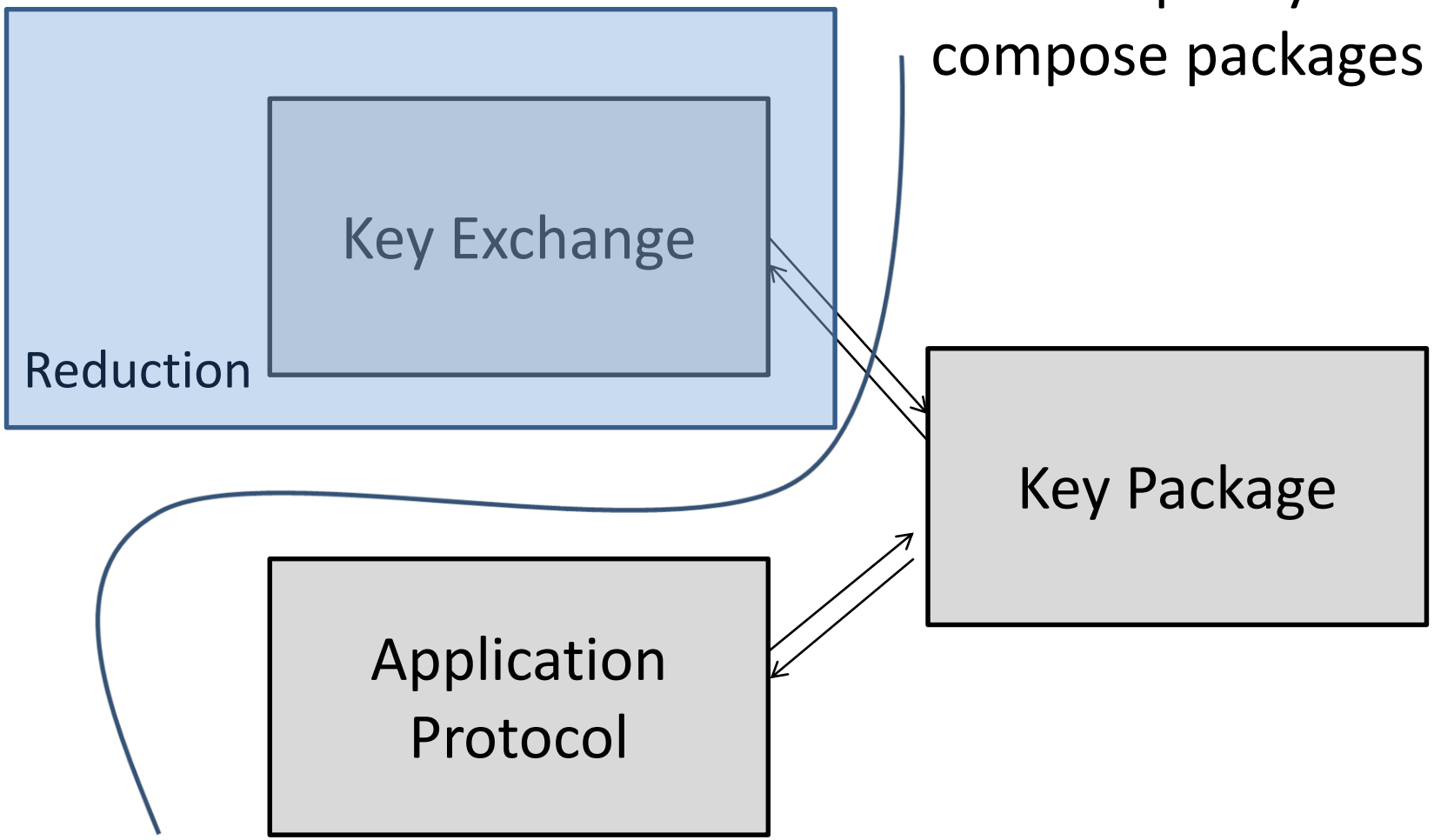
Idea 2: Specify rules to compose packages



Idea 2: Specify rules to compose packages



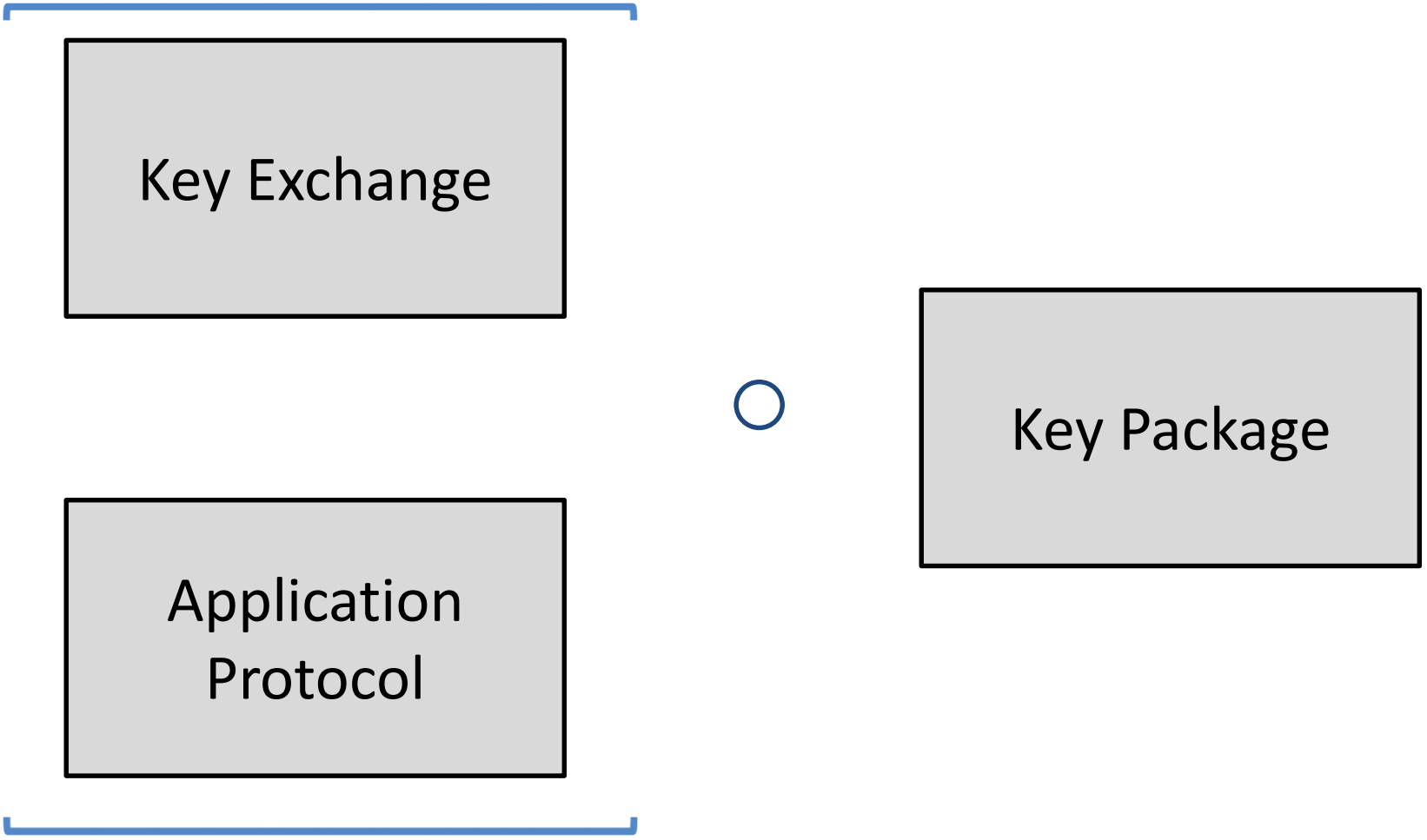
Idea 2: Specify rules to compose packages



The reduction is defined *automatically* and *precisely*.
How? The main focus of the paper is package composition.

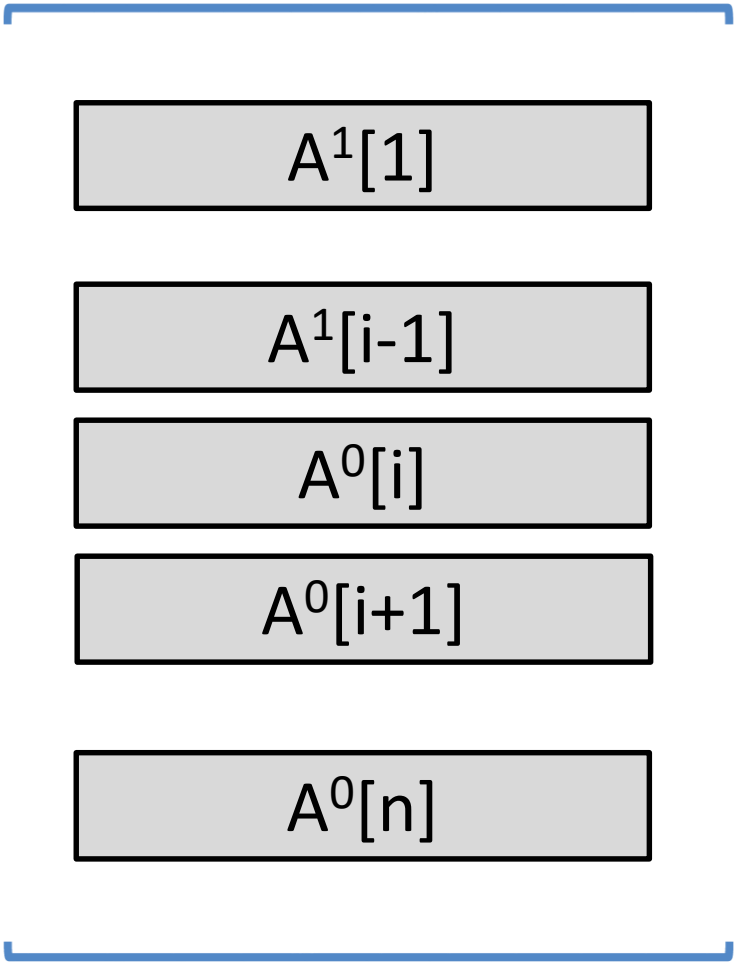
Bracket operator for parallel composition

Circle operator for packages that interact



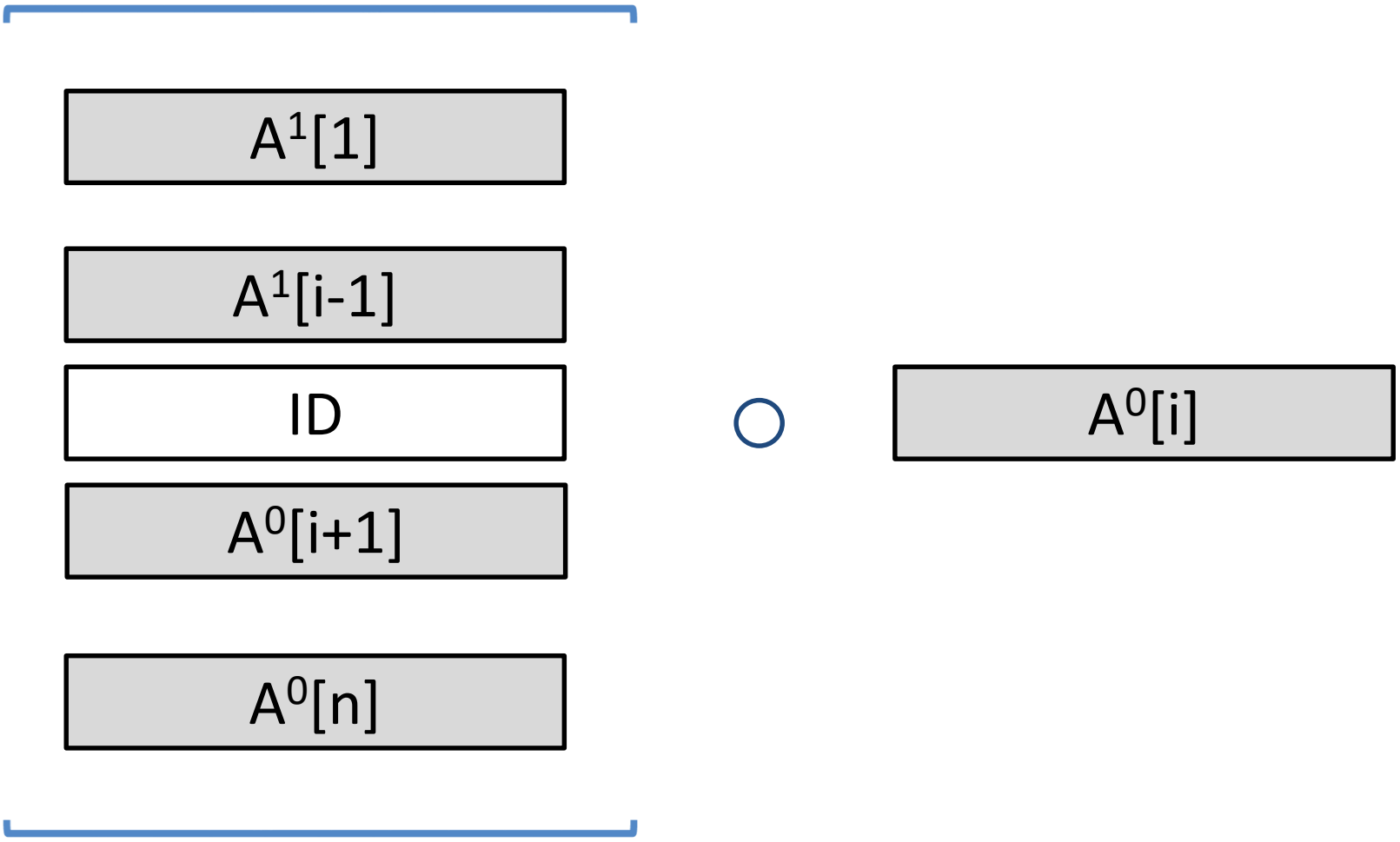
Hybrid

Idea 3: Specify algebraic rules



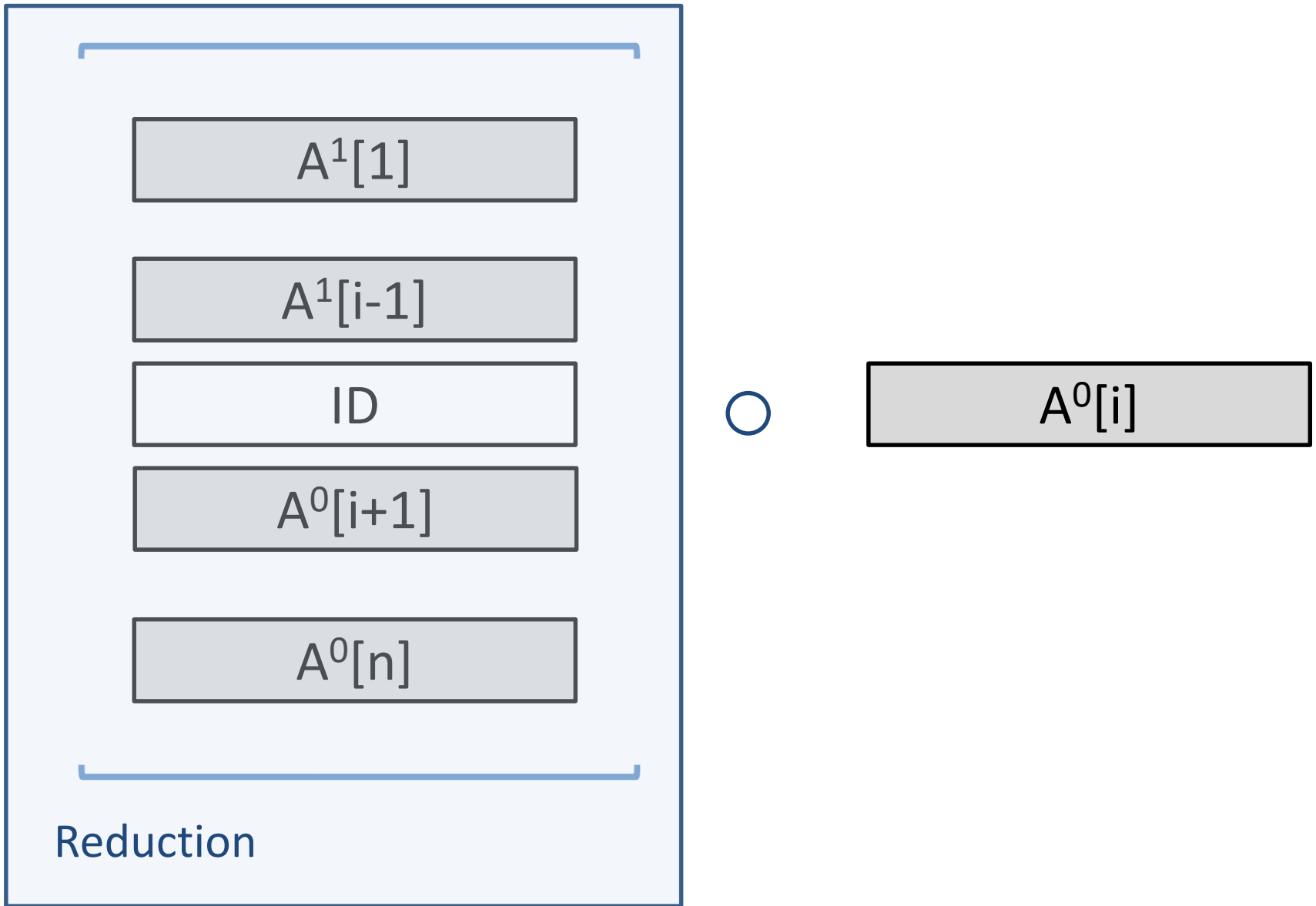
Hybrid

Idea 3: Specify algebraic rules



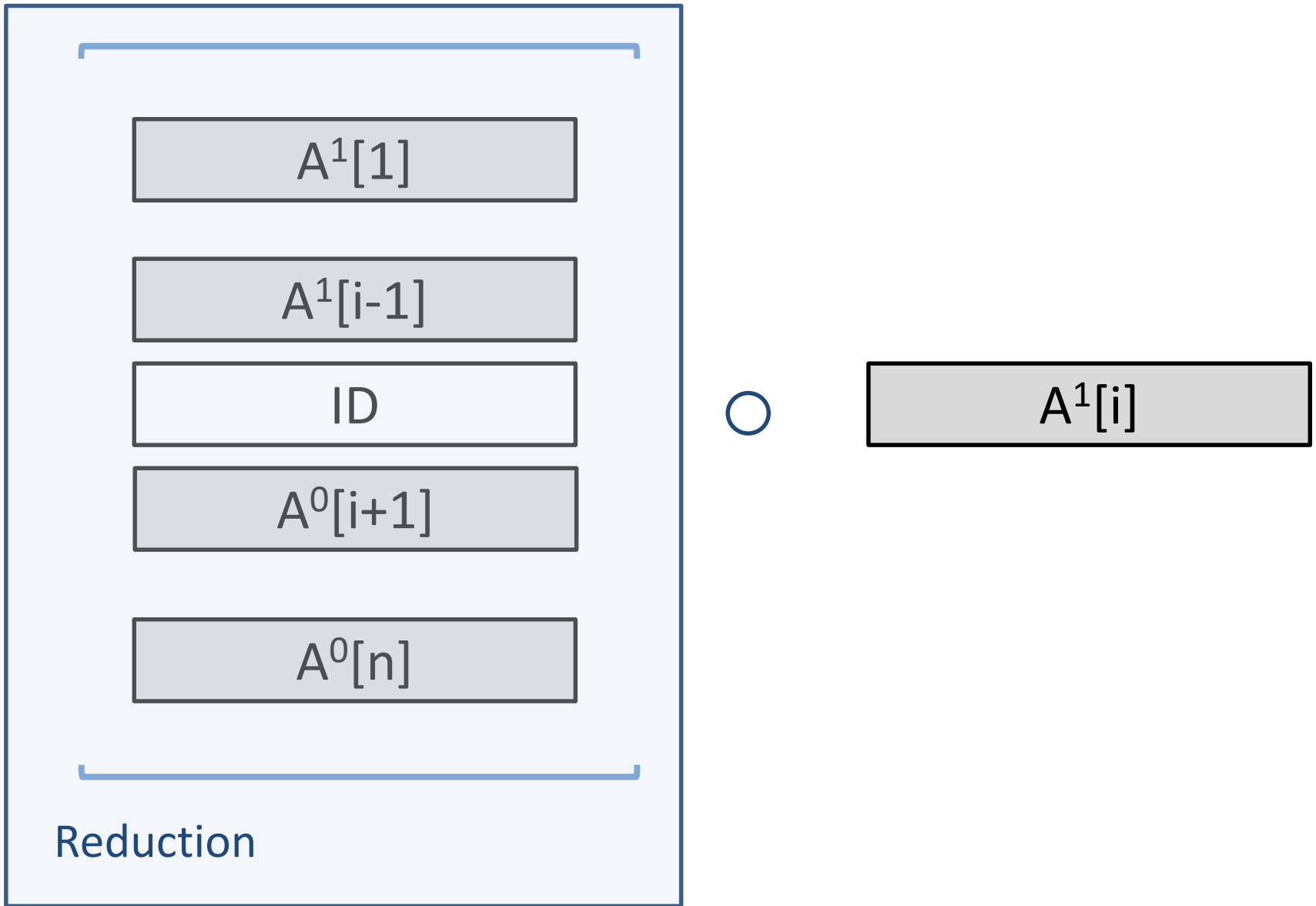
Hybrid

Idea 4: Automatic precise description of reductions

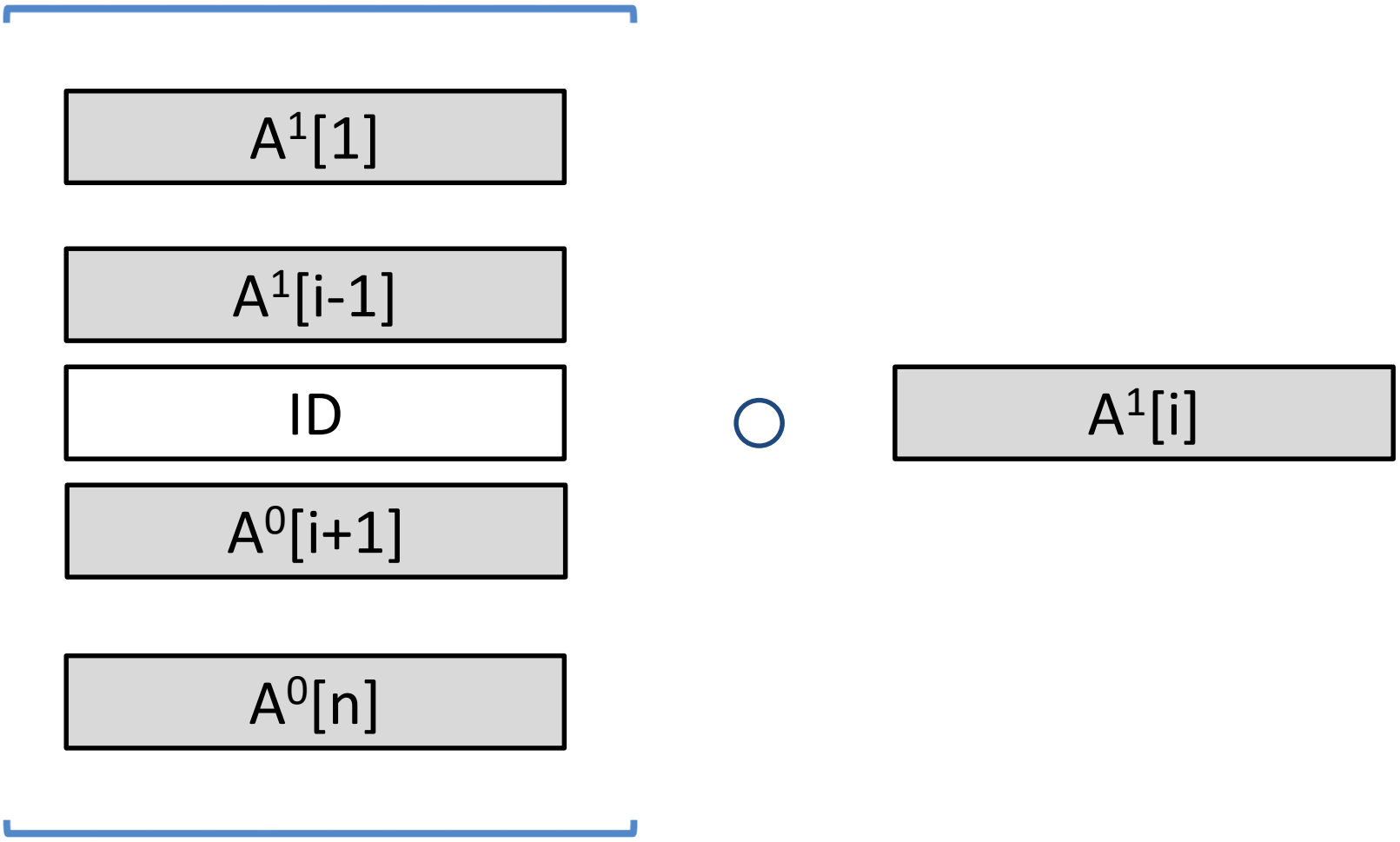


Hybrid

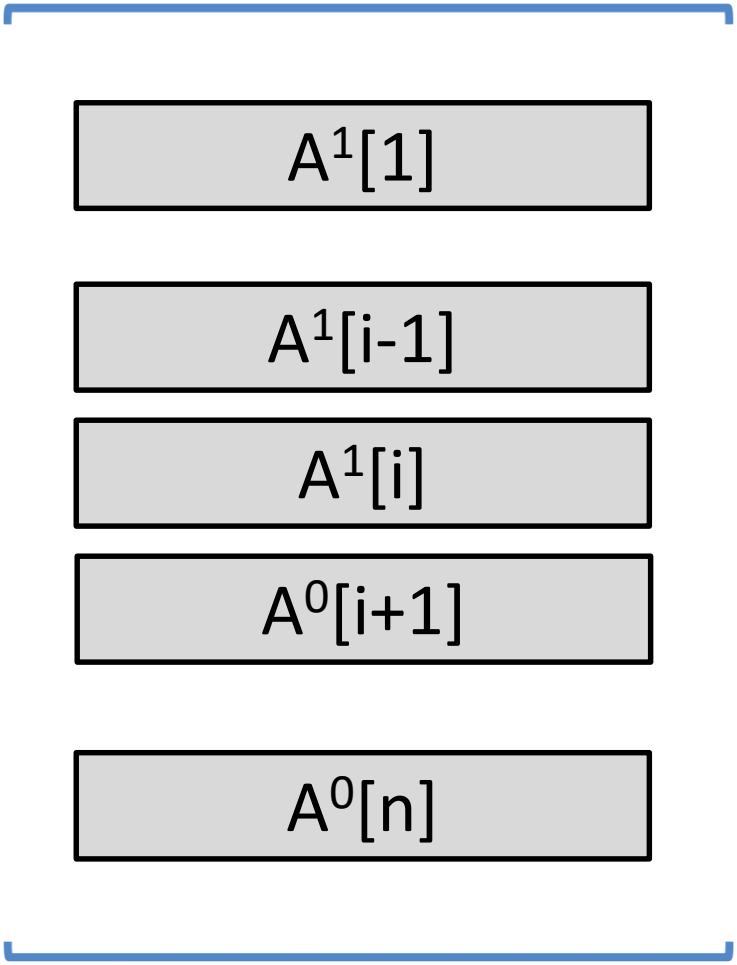
Idea 4: Automatic precise description of reductions



Hybrid



Hybrid



Motivation: Simple steps are

a) little work
b) precise
c) human-verifiable

Usability

Mike Rosulek uses some ideas and especially *similar notation* in the draft of his book

"The Joy of Cryptography"

which he has used in his *undergraduate* classes for years.

Our Hope:

a) Use method to prove TLS 1.3
b) Use method for meta-reductions
c) Make key exchange papers readable again
d) Suffer less, understand more ☺

Request/Suggestion

If you suffer from writing seemingly simple proof steps or from making them accessible to readers, check whether our notation can help.

If you have the same struggle in teaching, check whether Mike‘s book can help you.

Selection of Acknowledgements & Inspirations

- Universal composability, Ran Canetti
- Random systems, abstract crypto, constructive crypto, Ueli Maurer & Renato Renner
- miTLS, Microsoft Research & Inria Paris
- Pi-Calculus

We put existing ideas together with a focus on *proofs* and real-life protocol proofs in mind.

Now on ePrint:

State-Separating Proofs
A Reduction Methodology for Real-World Protocols

Chris Brzuska, Antoine Delignat-Lavaud,
Konrad Kohbrok, Markulf Kohlweiss

Now on ePrint:

State-Separating Proofs
A Reduction Methodology for Real-World Protocols

Chris Brzuska, Antoine Delignat-Lavaud,
Konrad Kohbrok, Markulf Kohlweiss

Maybe, it can help you, too ☺