Digital Money & Blockchain Forum, April 24th, 2018

Proofs-of-space – an alternative for Proofs-of-work

Stefan Dziembowski
University of Warsaw

Talk based on joint work with **Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak (CRYPTO 2015)**

General idea

We introduce **Proofs-of-Space** – a type of a "proof of effort", where the "**effort**" is measured in terms of "**wasted memory**"

(an alternative to **Proofs-of-Work**).

Bitcoin in a nutshell: a "digital analogue" of the paper money

A digital currency introduced by "Satoshi Nakamoto" in 2008.

Based on the assumption that ``the majority of the computing power is honest".

currency unit: **Bitcoin (BTC) 1 BTC = 10^8 Satoshi**

as of April 24, 2018:

Market cap ≈ USD 160 billion

1 BTC ≈ USD 9000

How is this the “honest computing power majority” verified?

Main idea:

- use **Proofs of Work**
- **incentivize** honest users to constantly participate in the process

The honest users can use their **idle CPU cycles**.

Nowadays: often done on **dedicated hardware**.

Proofs of work

Introduced by **Dwork and Naor** [Crypto 1992] as a countermeasure against spam.

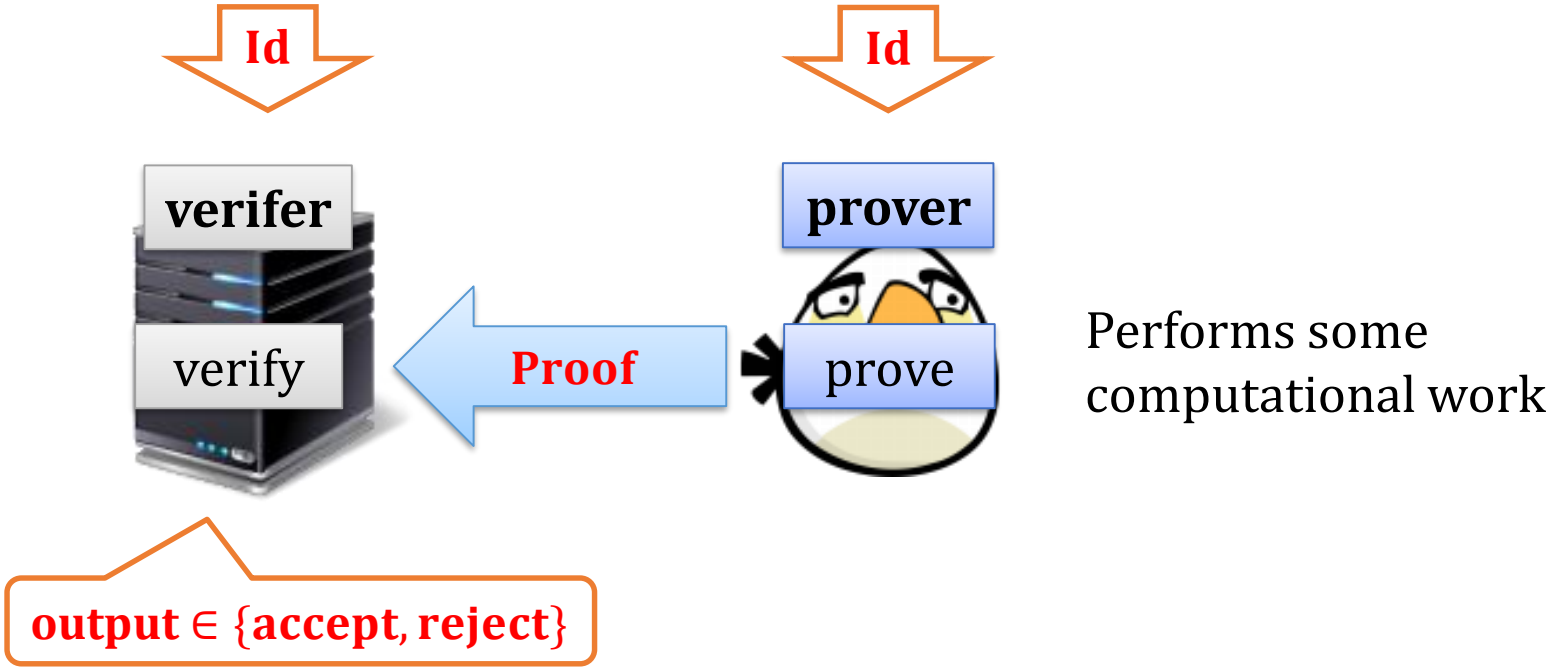
Basic idea:
Force users to do some computational work:

solve a **moderately difficult** "puzzle"

(checking correctness of the solution has to be fast)

Proofs of work – the general scenario

The proof is done with respect to an identifier **Id**



Performs some computational work

A drawback of PoW systems

1. **high energy consumption**

costs money

bad for environment

2. advantage for people with **dedicated hardware**

What to do?

This problem seems unavoidable:

The only way to prove that one "invested a lot of computing power" is to do **a lot of computation**.

What is the other resource that we could use?

Proofs of Space (PoSpace):

instead of **CPU** use **disk space**!

Example of an application

Goal: prevent malicious users from opening lots of fake accounts.

Method: force each account owner to "waste" large part of his local space.

Important: the space needs to be allocated as long as the user uses the service.

Main difference from PoWs

To prove that one wasted **n CPU cycles** one **needs to perform these cycles**.

while:

To prove that one wasted **n bytes** one **does <u>not</u> need touch all of them.**

Advantages

- **more energy-efficient**
- no "**hardware acceleration**"
- **cheaper** (user can devote their unused disk space)

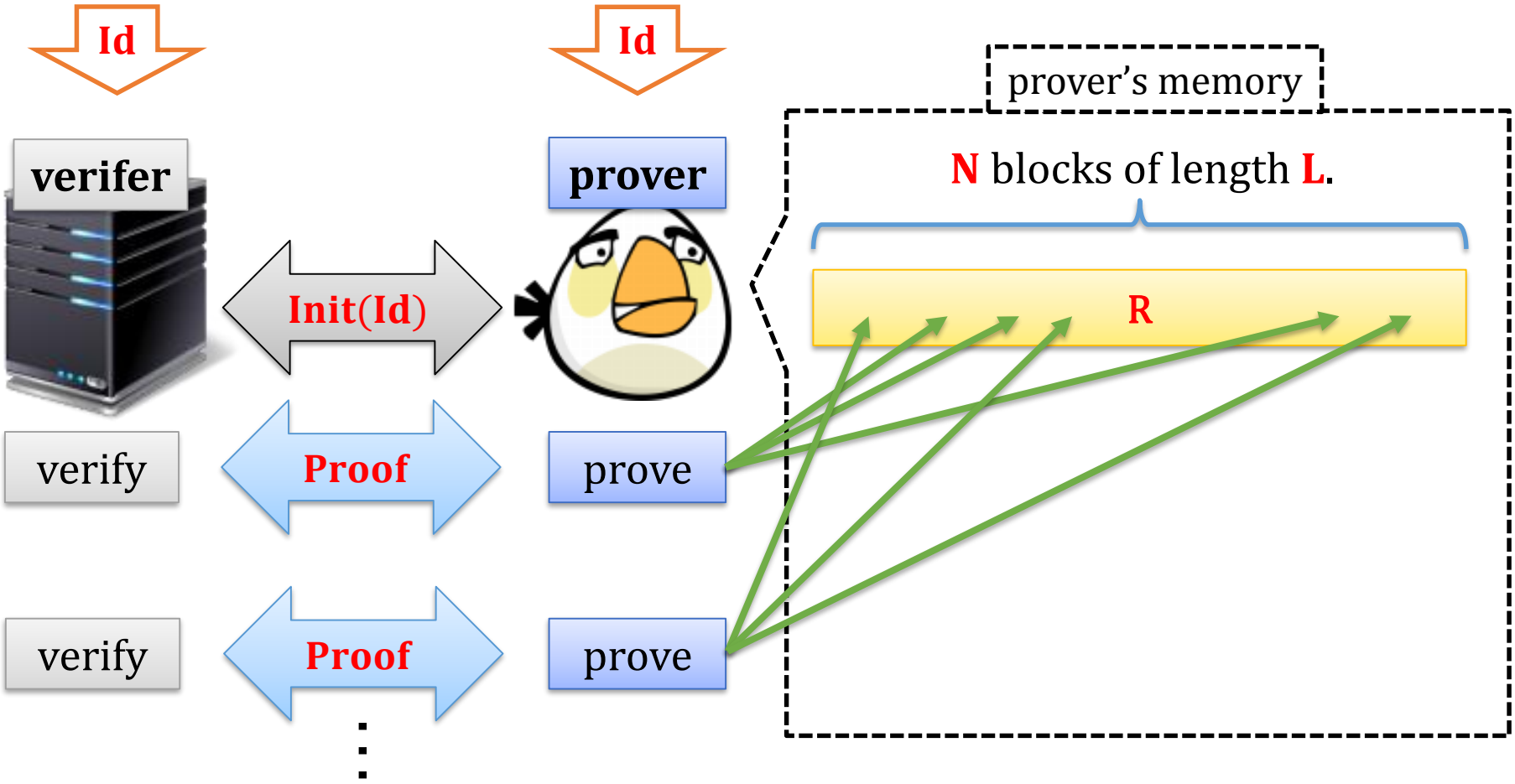
The security definition

How to measure time and space

Time is measured in terms of calls to **H**.

Space is measured in blocks of length ***L*** (outputs of **H**)

The proof is done with respect to an identifier **Id**



output ∈ {accept, reject}

How to define security of a PoSpace

Properties:

- **completeness**,
- **soundness**, and
- **efficiency**.

If the prover is **honest** then the verifier will **always accept** the proof.

less trivial to define

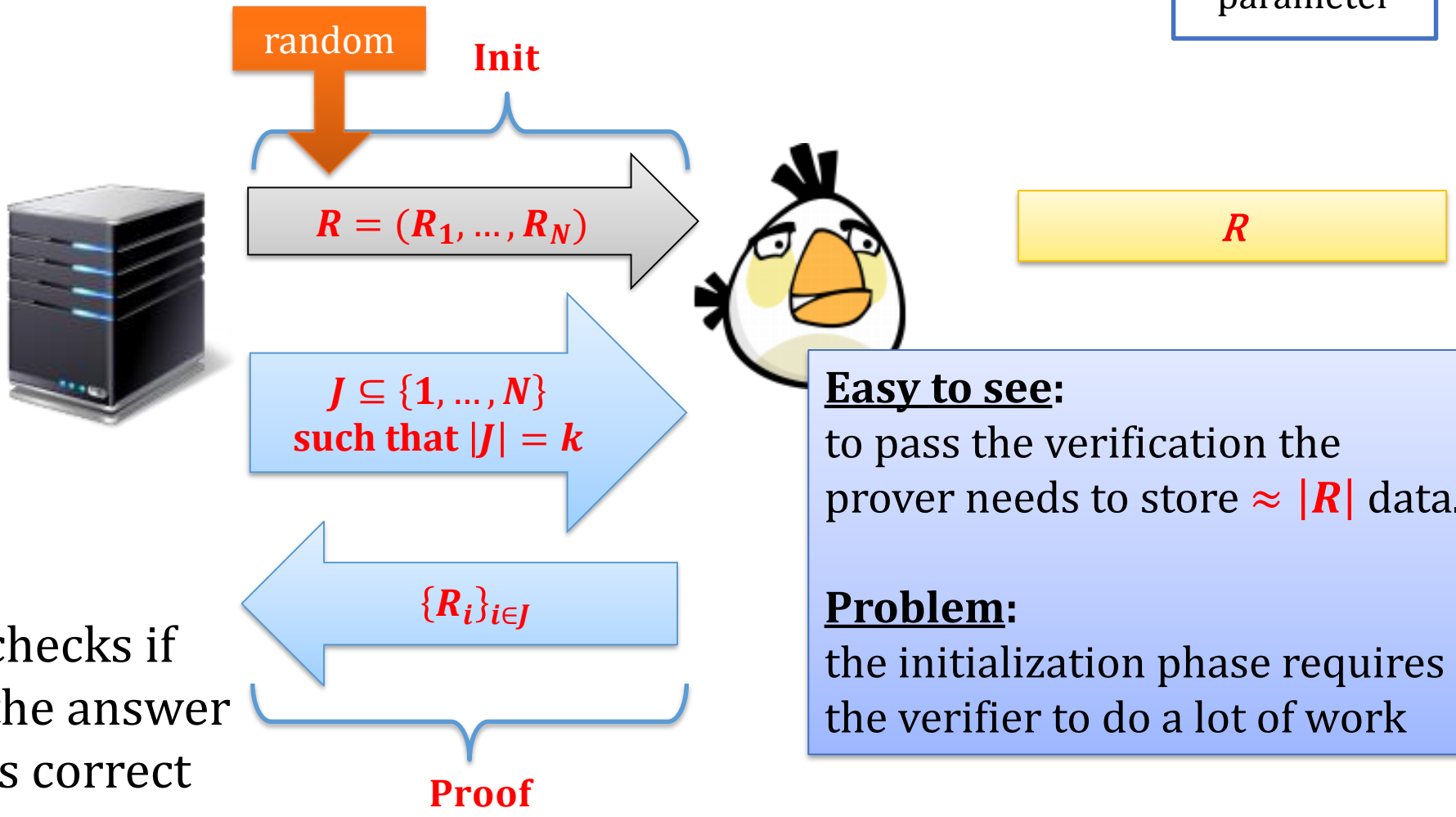
How to define the efficiency?

Let us show a very simple (but **not** efficient) **PoSpace**.

Note: we have **not** defined the security yet, so it's just an "*informal example*".

A “trivial PoSpace”

k – security parameter



checks if the answer is correct

Easy to see:
to pass the verification the prover needs to store $\approx |R|$ data.

Problem:
the initialization phase requires the verifier to do a lot of work

Note: if R is generated pseudorandomly then he need to store only the seed.

Efficiency

We **require** that the **computing time** of the parties is as follows:

	verifier	prover
Init	very fast	slow (needs to depend on the length of **R**)
Proof	very fast	very fast

Note:
this also imposes limit on communication complexity.

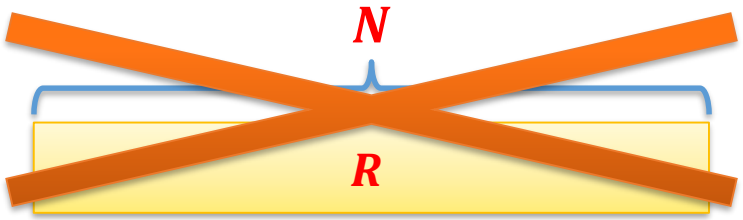
How to define soundness?

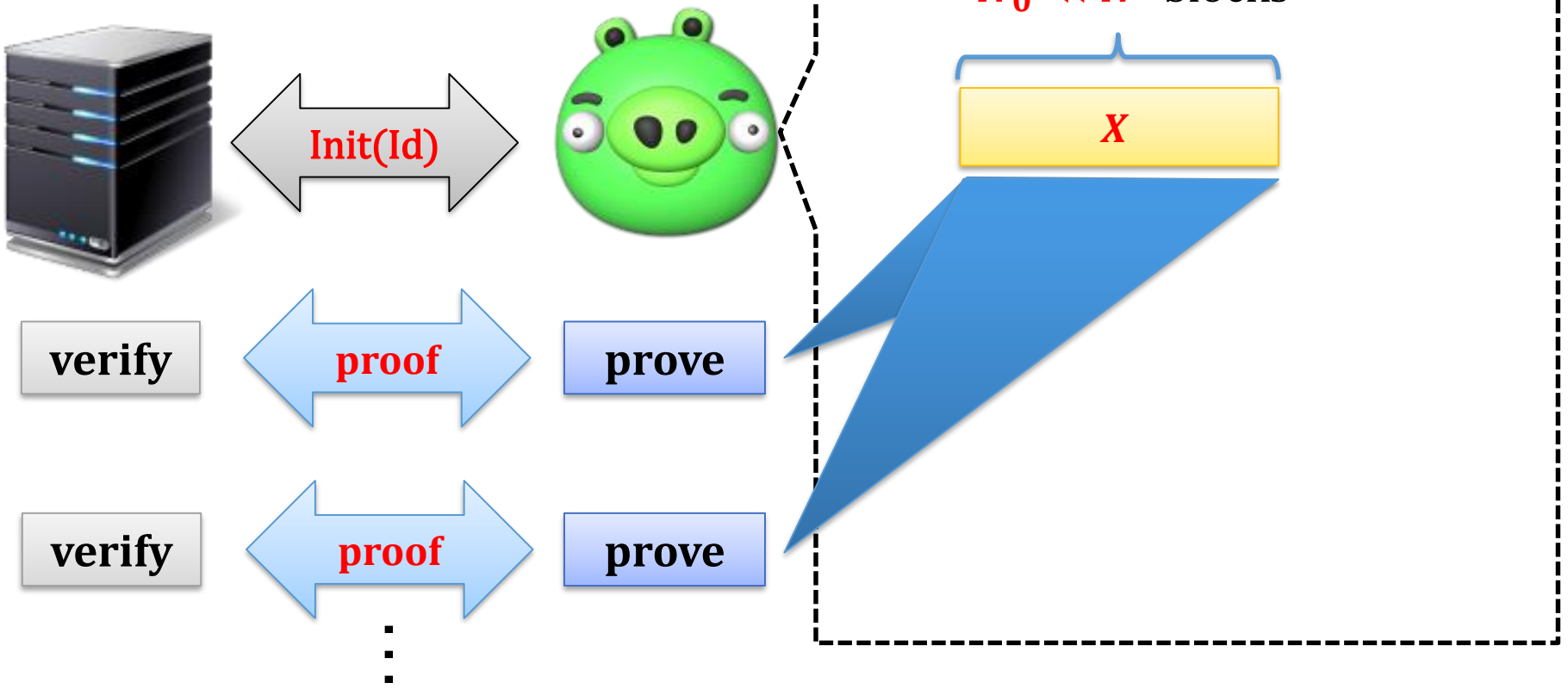
Informally:

we want to force a cheating prover to constantly waste a lot of memory.

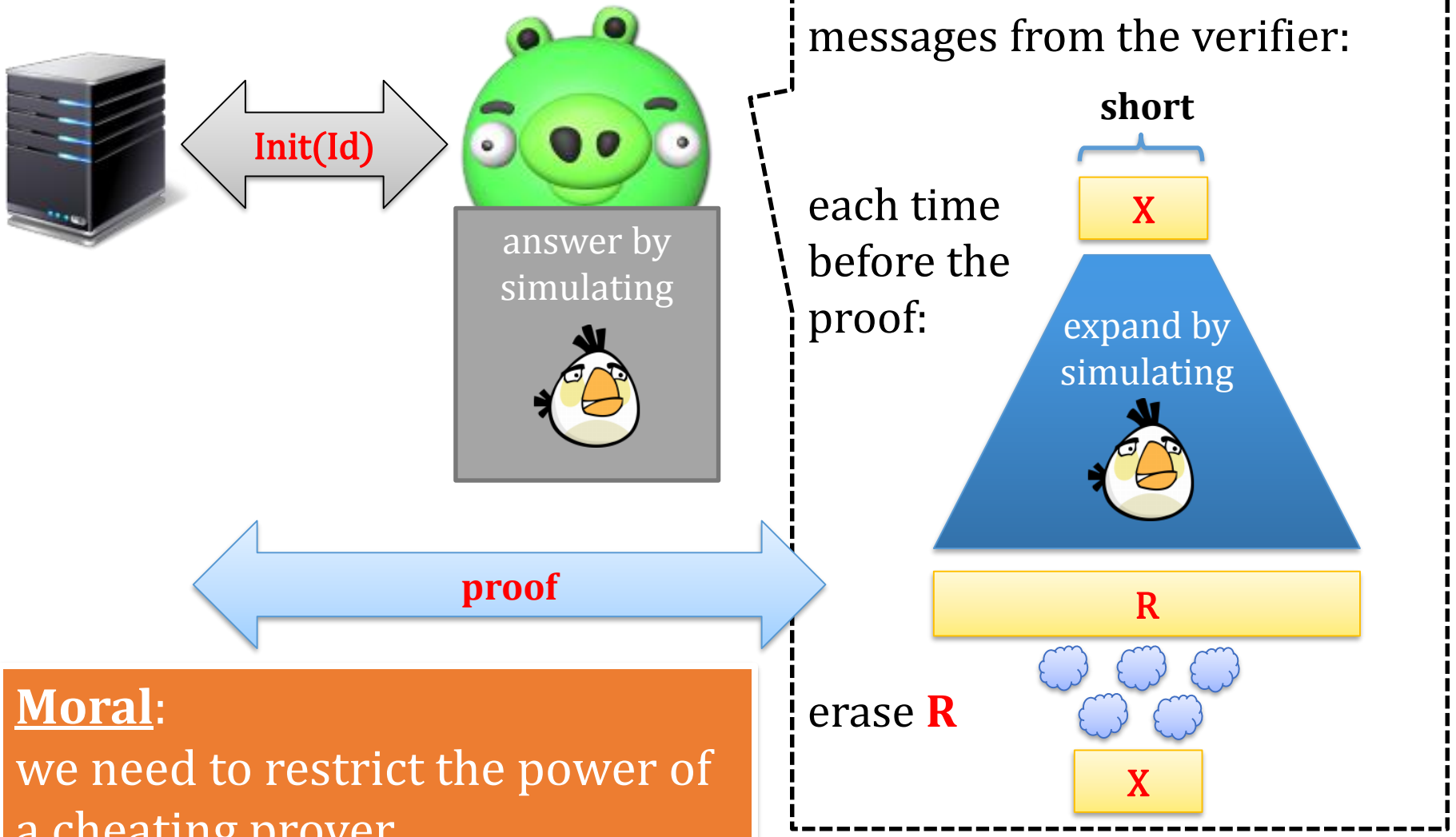
What would be the goal of a cheating prover?

"Compress" R:





Observation: a cheating prover has a simple (but inefficient) winning strategy.



Moral:
we need to restrict the power of a cheating prover.

Solution

Restrict the **cheating prover's** computing power **during the "proof phase"**.

Informally, a cheating prover is forced to do one of the following things:

- constantly **use a lot of memory** (but then he doesn't win anything), or
- invest **a lot of computing effort during the proof** (but this requires time!)

Constructions

Cautionary note

Simple ideas do not work!

This is due to the time-memory tradeoffs ☹

Our construction

We construct PoSpace schemes using techniques from **graph theory**.

- **"hard to pebble graphs"** of **Paul, Tarjan, Celoni, 1976**,
- **"superconcentrators"**, **"random bipartite expander graphs",** and
- **graphs** of **Erdos, Graham, Szemeredi, 1975**.

The details are in the paper.

Alternative constructions

- **Ling Ren and Srinivas Devadas**
  Proof of Space from Stacked Bipartite Graphs,
  Theory of Cryptography. TCC 2016.

  also uses advanced graph theory…

- A much simpler construction:
  Hamza Abusalah, Joel Alwen, Bram Cohen; Danylo Khilko, Krzysztof Pietrzak, and Leonid Reyzin
  Beyond Hellman's Time-Space Trade-Offs with Applications to Proofs of Space
  ASIACRYPT 2017

A natural question

How to construct a cryptocurrency based on the PoSpace.

Not so trivial...

An academic proposal:

Park, Pietrzak, Kwon, Alwen, Fuchsbauer, and Gazi:
SpaceMint: A Cryptocurrency Based on Proofs of Space.
Cryptology ePrint Archive: Report 2015/528

A currency based on these ideas (developed by Bram Cohen):

Thank you!