Blockchains & Cryptocurrencies

Applications of Blockchains - III

Instructor: Abhishek Jain
Johns Hopkins University: Spring 2021

Today

- More cryptographic applications of Blockchains
- **Key Idea:** Using Blockchains as an immutable public ledger
  - Proof of Stake vs Proof of Work blockchains
  - Chances of creating a fork that “looks like” honest chain (monetary-cost security vs cryptographic security)

I. Overcoming Cryptographic Impossibility Results using Blockchains

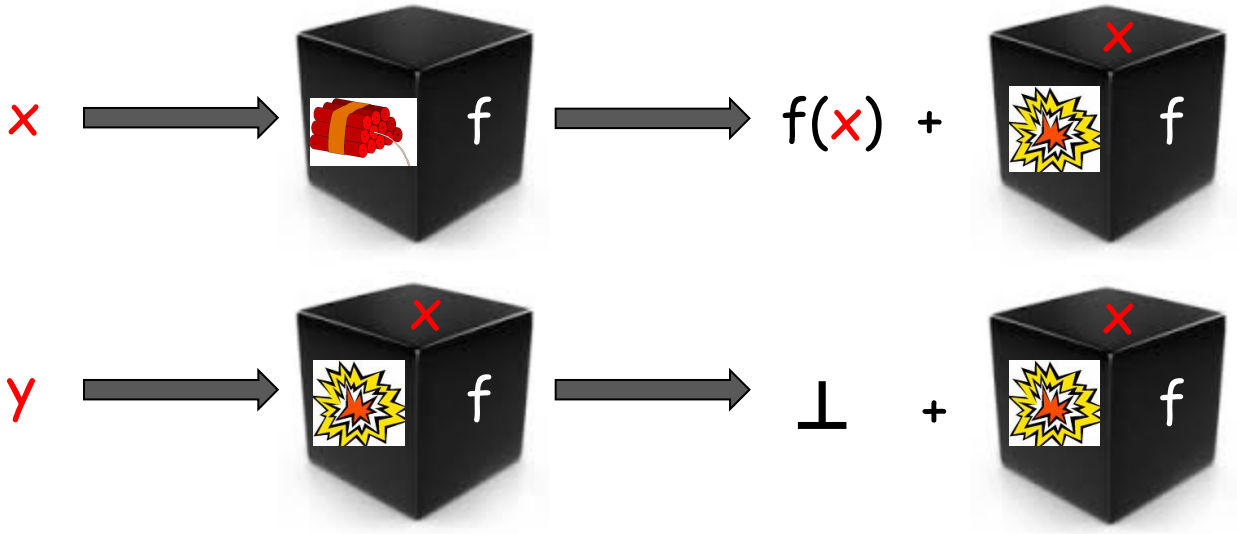
Rishab Goyal and Vipul Goyal

TCC 2017

Slides based on Rishab's talk at TCC'17

One-Time Programs [GoldwasserKalaiRothblum08]

- Can only be executed on *single* input
- Input chosen at run-time



Applications

- (Proprietary) Software Leasing
  - Charge money for every use
- Embed secrets in programs that can only be read once!
  - Yes, like in Mission Impossible

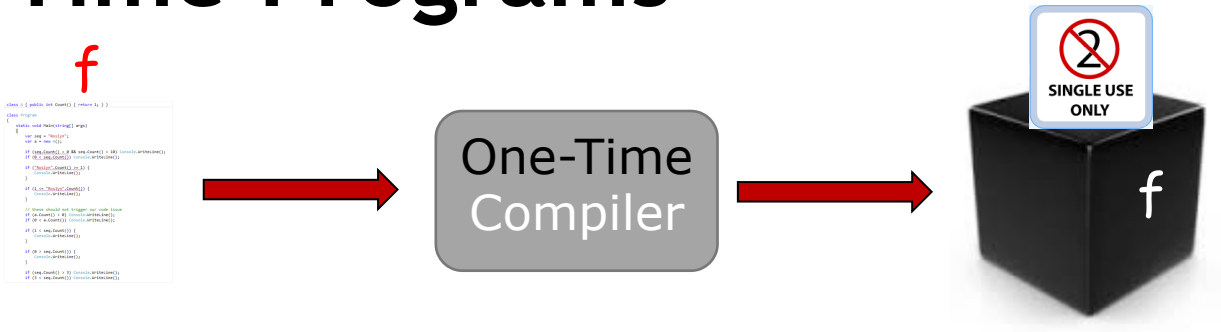
Applications

- (Proprietary) Software Leasing
  - Charge money for every use
- Embed secrets in programs that can only be read once!
  - Yes, like in Mission Impossible

Can we construct one-time programs?

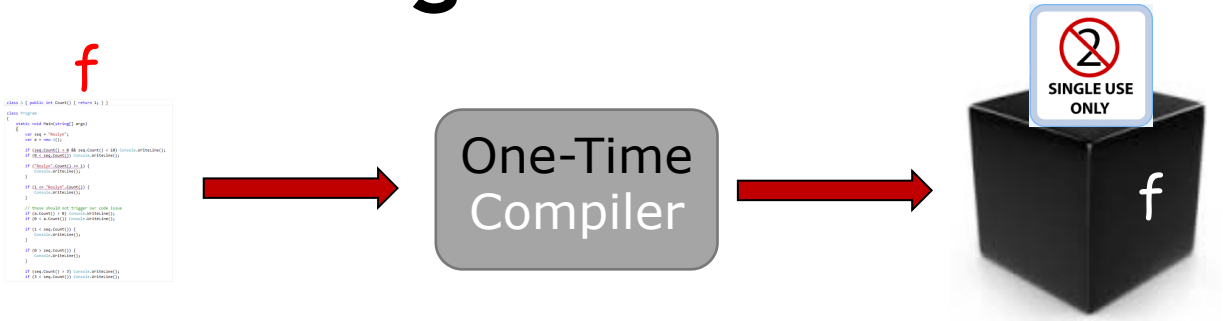
In general, the answer seems no

One-Time Programs



- [GKR'08]: Construction based on "tamper-proof hardware"
  - If the hardware is **stateful** and can execute arbitrary programs, then there is a straightforward solution (Think: Why?)
  - GKR solution uses a program-independent "simple" **one-time use memory token** (i.e., *it can only be read once*)

One-Time Programs



- [GKR'08]: Construction based on "tamper-proof hardware"
- [GG'17]: *Software-only construction using blockchains with specific properties*

Main Ingredients:

- Advanced Cryptographic primitives
  - (Extractable) *Witness Encryption* [Garg-Gentry-Sahai-Waters'13]
  - Garbled Circuits [Yao'82]
- Blockchains with specific properties

Main Ingredients:

- Advanced Cryptographic primitives
  - (Extractable) *Witness Encryption* [Garg-Gentry-Sahai-Waters'13]
  - Garbled Circuits [Yao'82]
- Blockchains with specific properties

Today: Simpler solution without using Garbled circuits and Witness encryption. Instead, we will use a **stateless** tamper-proof hardware chip

(Think: Why is the problem still hard if the hardware is stateless?)

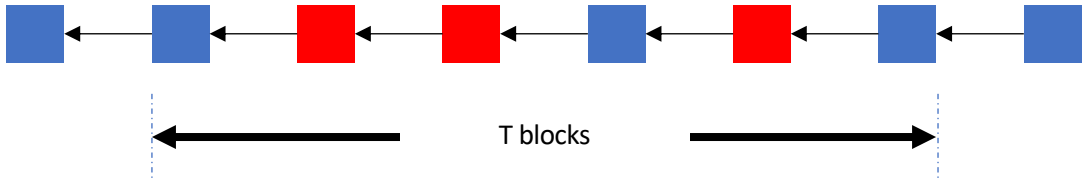
Blockchain Properties

[GarayKiayiasLeonardos15,PassSeemanShelat16]

Chain Quality

Number of blocks mined by **honest** parties is proportional to their voting power, for any T consecutive blocks

- This ensures that if majority of voting power is honest, then there will be a majority of “good” blocks within every T-window



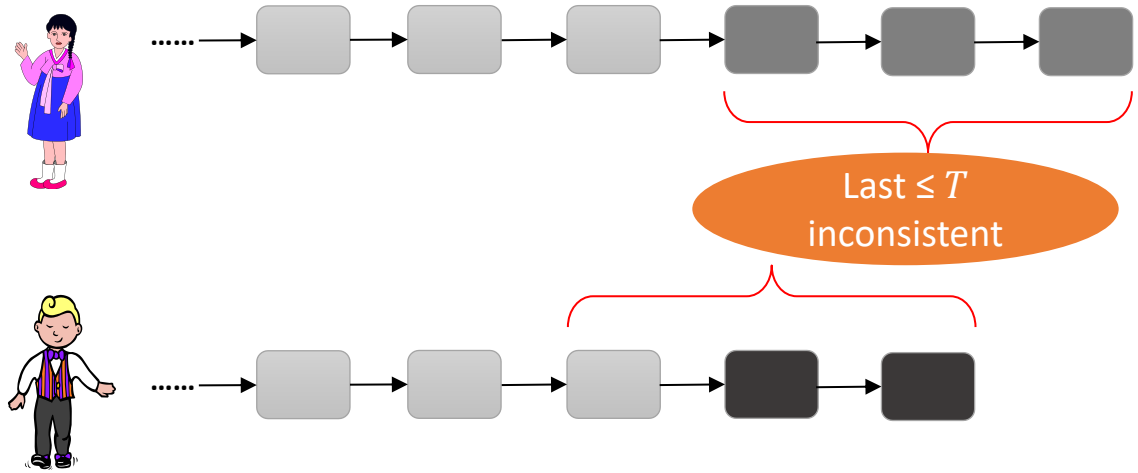
honest block

adversarial block

Chain T-consistency

Honest parties agree on all but last T blocks

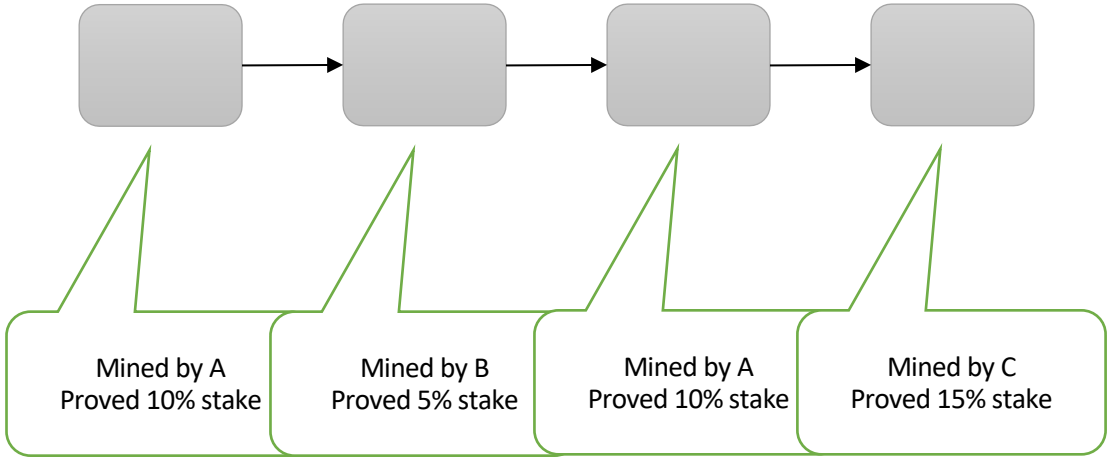
- This ensures that once a block is T blocks deep in a chain, it appears on every party's blockchain



[Goyal-Goyal’17]:
New *Proof-of-Stake* Specific Abstractions

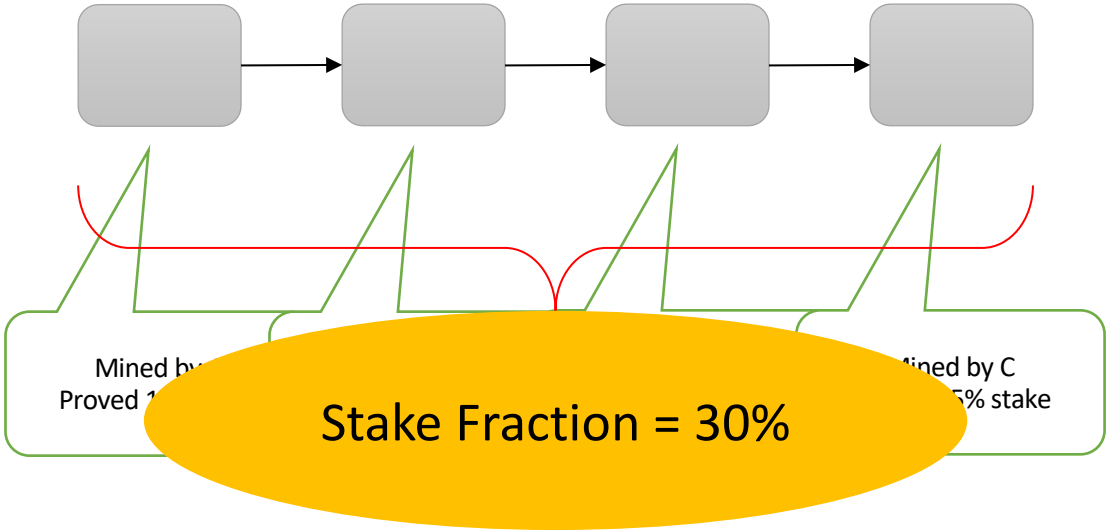
Defining *Stake Fraction*

- Measure of *combined* difficulty of POS puzzles solved



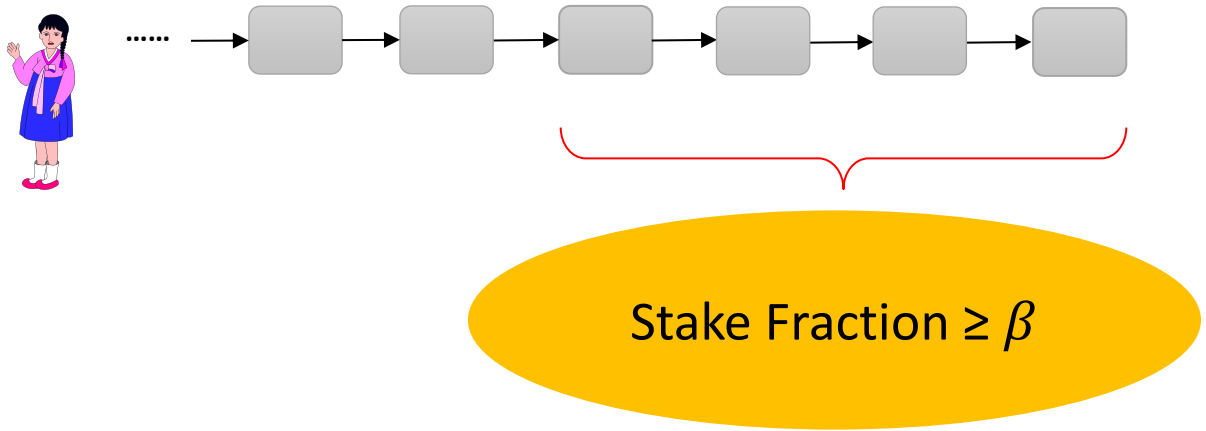
Defining *Stake Fraction*

- Measure of *combined* difficulty of POS puzzles solved



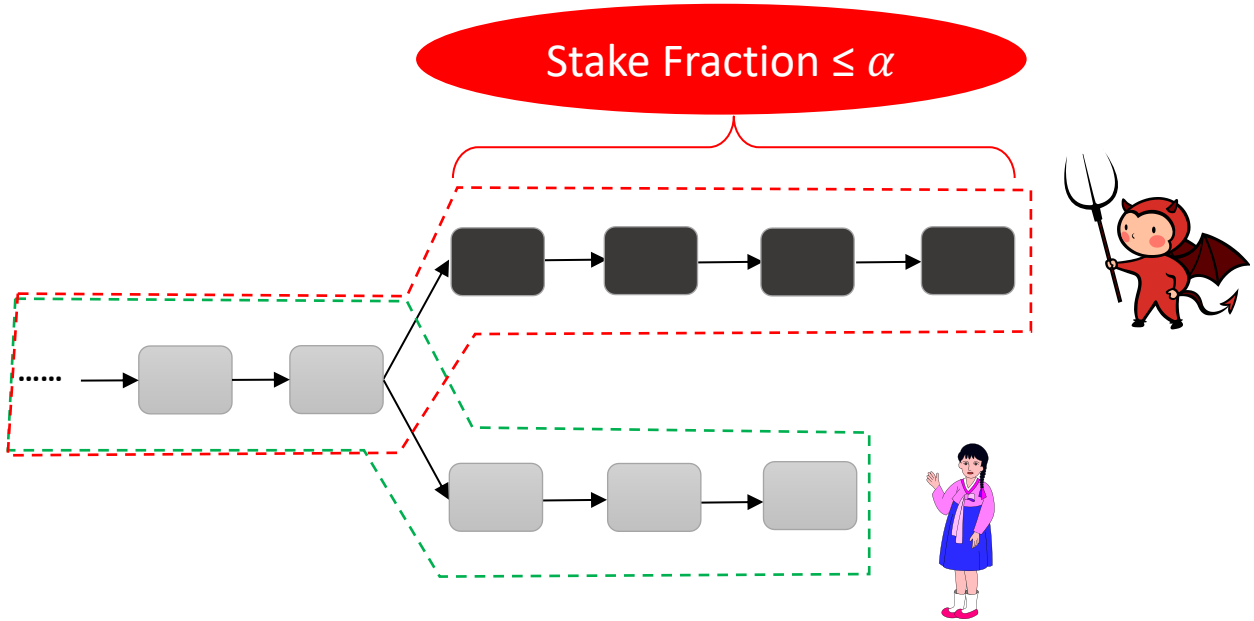
(β, ℓ)-Sufficient Stake Contribution

- Total 'stake-fraction' in last ℓ blocks is a (fairly) high fraction ($\geq \beta$)



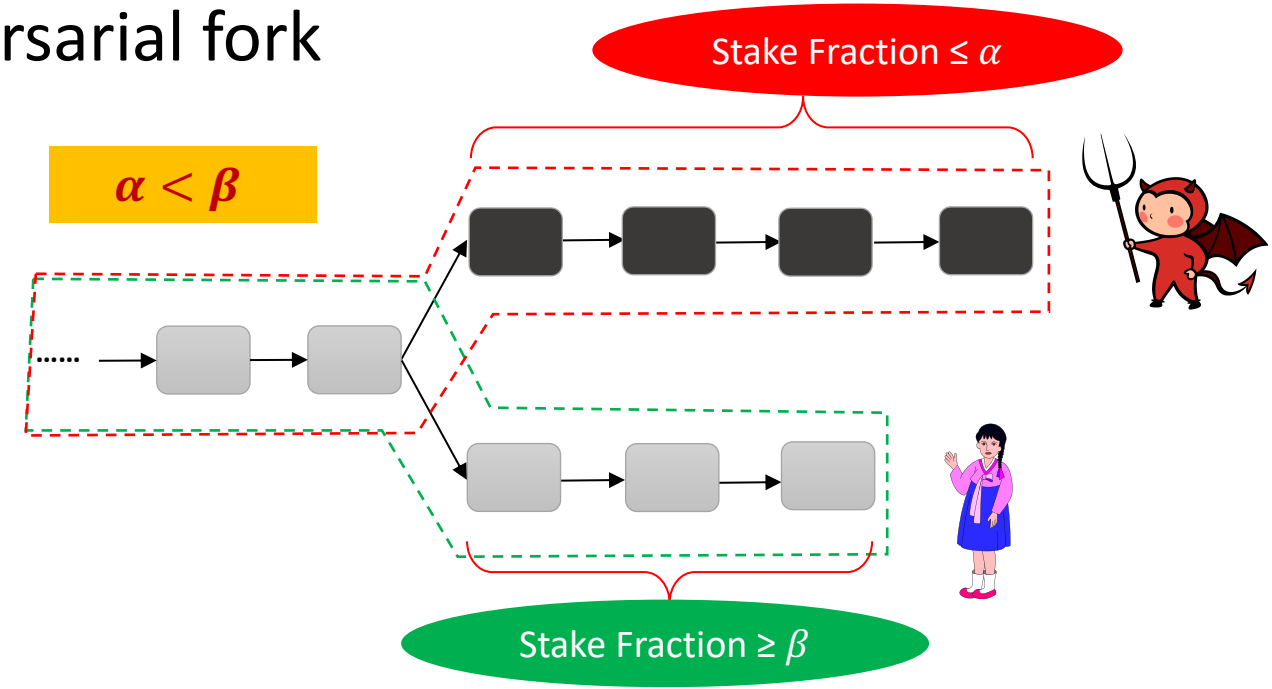
(α, ℓ)-Bounded Stake Forking

- No adversary can create a valid fork (length ≥ ℓ) with high stake-fraction (≥ α)



(α, β, ℓ)-Distinguishable Forking

- Honest chain of blocks can be distinguished from adversarial fork



Connection to Previous Properties

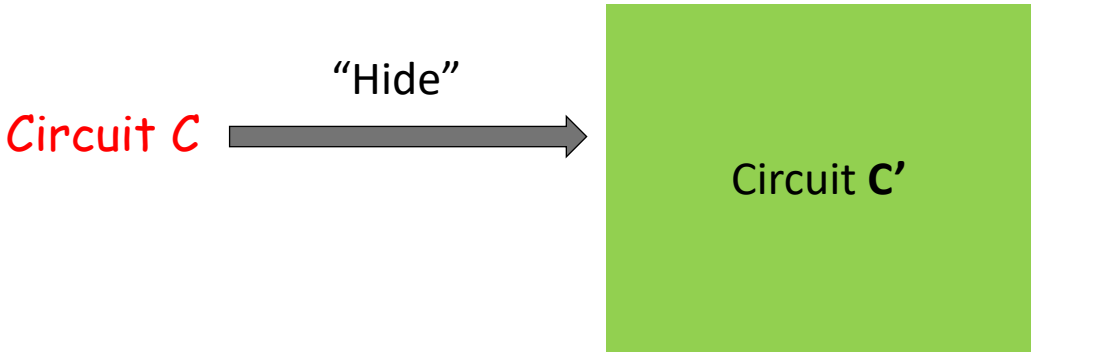
[GG'17]: If a PoS blockchain satisfies chain consistency and chain quality, then it also satisfies the following properties:

- sufficient honest- stake contribution
- bounded stake forking
- distinguishable forking

(for some choice of parameters)

One-Time Program
(using secure hardware)

Compilation:

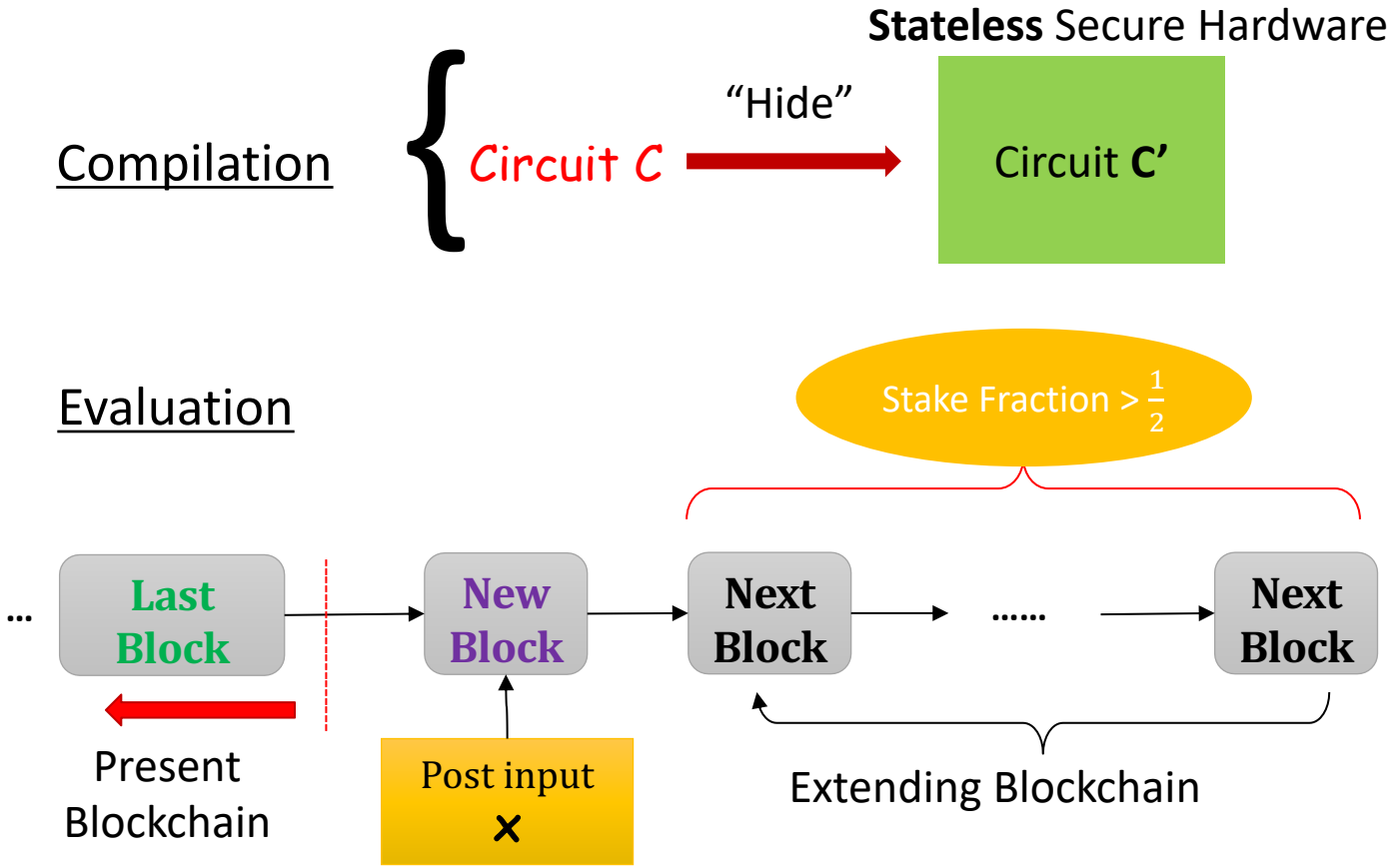


One-Time Program

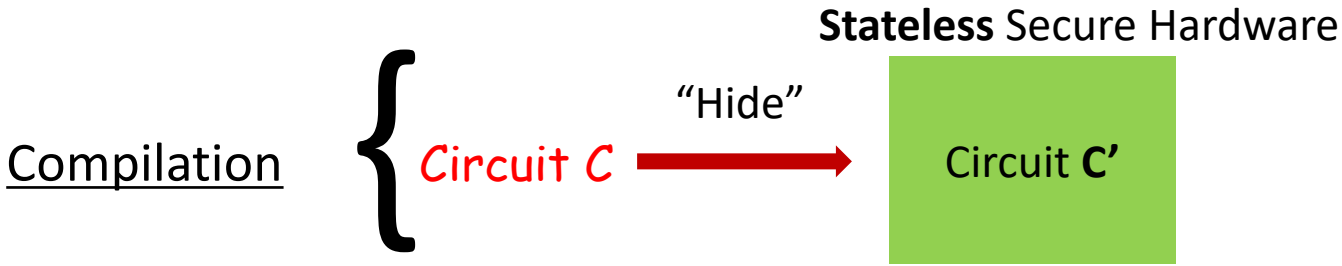
Evaluation steps (informal):

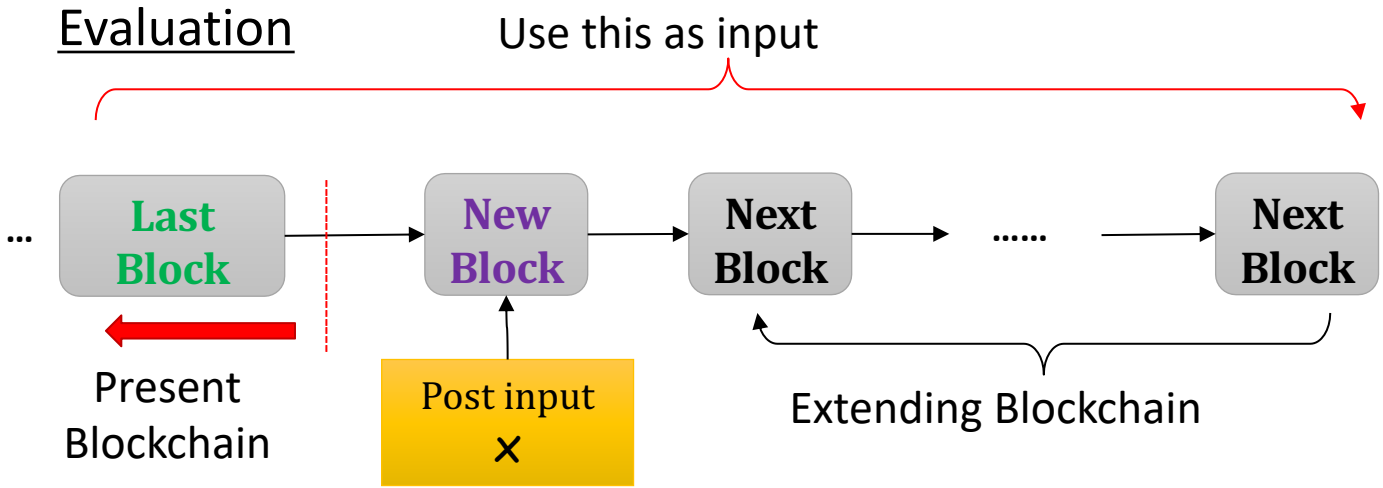
1. Evaluator posts its input **x** on the blockchain
2. Waits for the blockchain to be “sufficiently” extended
3. Provides the blockchain state as an input to secure hardware
4. If the blockchain state contains a “single” input **x**, circuit C’ in secure hardware evaluates and outputs **C(x)**

One-Time Program



One-Time Program





One-Time Program

Security (informal):

- To evaluate OTP on any input **x**, adversary must first post it on the blockchain
- If adversary posts **x'** ≠ **x** after already posting **x**, blockchain state will contain both of them
- **Such a blockchain state is not a valid input** (rejected by circuit C')

One-Time Program

Security (contd):

- But what if adversary creates a fork from “last block” and posts **x’** there?
- To create a valid input, adversary must extend the fork so that it has stake fraction > 1/2
- **Any such fork will be distinguishable from real chain due to distinguishable forking property**