Brick

Asynchronous Payment Channels

Zeta Avarikioti

L. Kokoris-Kogias, R. Wattenhofer, D. Zindros

ETH Zurich – Distributed Computing Group – www.disco.ethz.ch , EPFL - Decentralized and Distributed Systems Lab, UoA - Decrypto.org

Fundamentals of Channels

◀◀

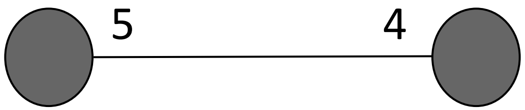
Fundamentals of Channels

Fundamentals of Channels

Funding transaction

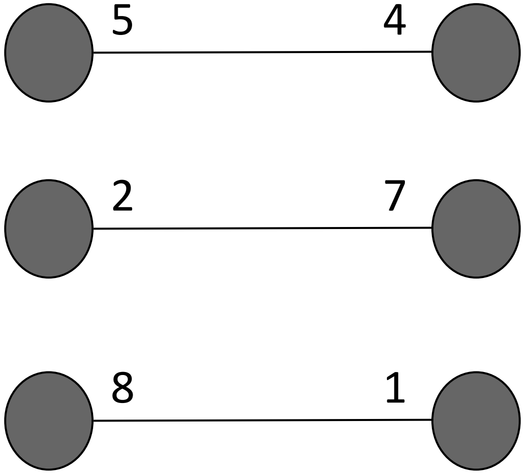
Fundamentals of Channels

Commitment transaction

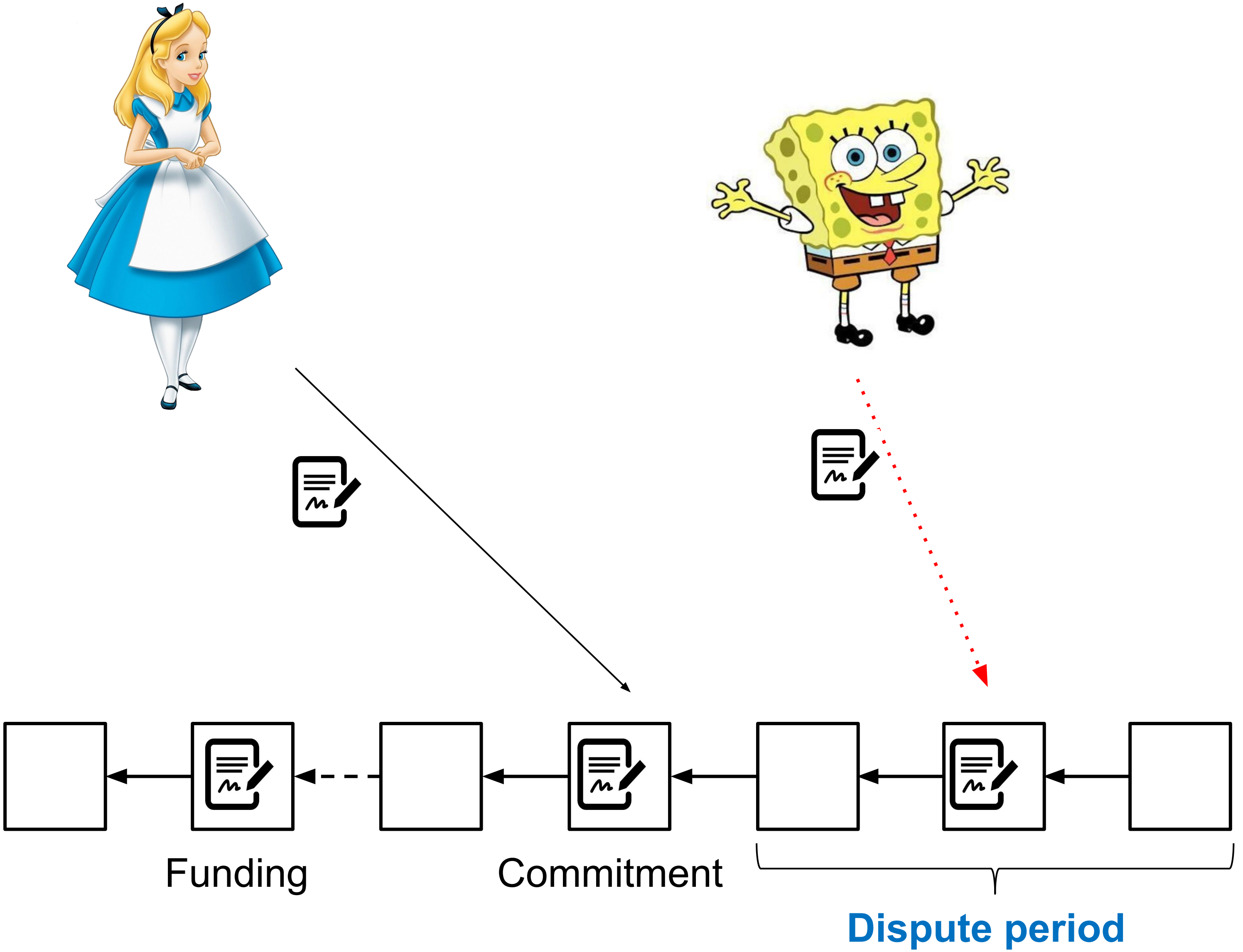


Fundamentals of Channels

Commitment transaction



Fundamentals of Channels

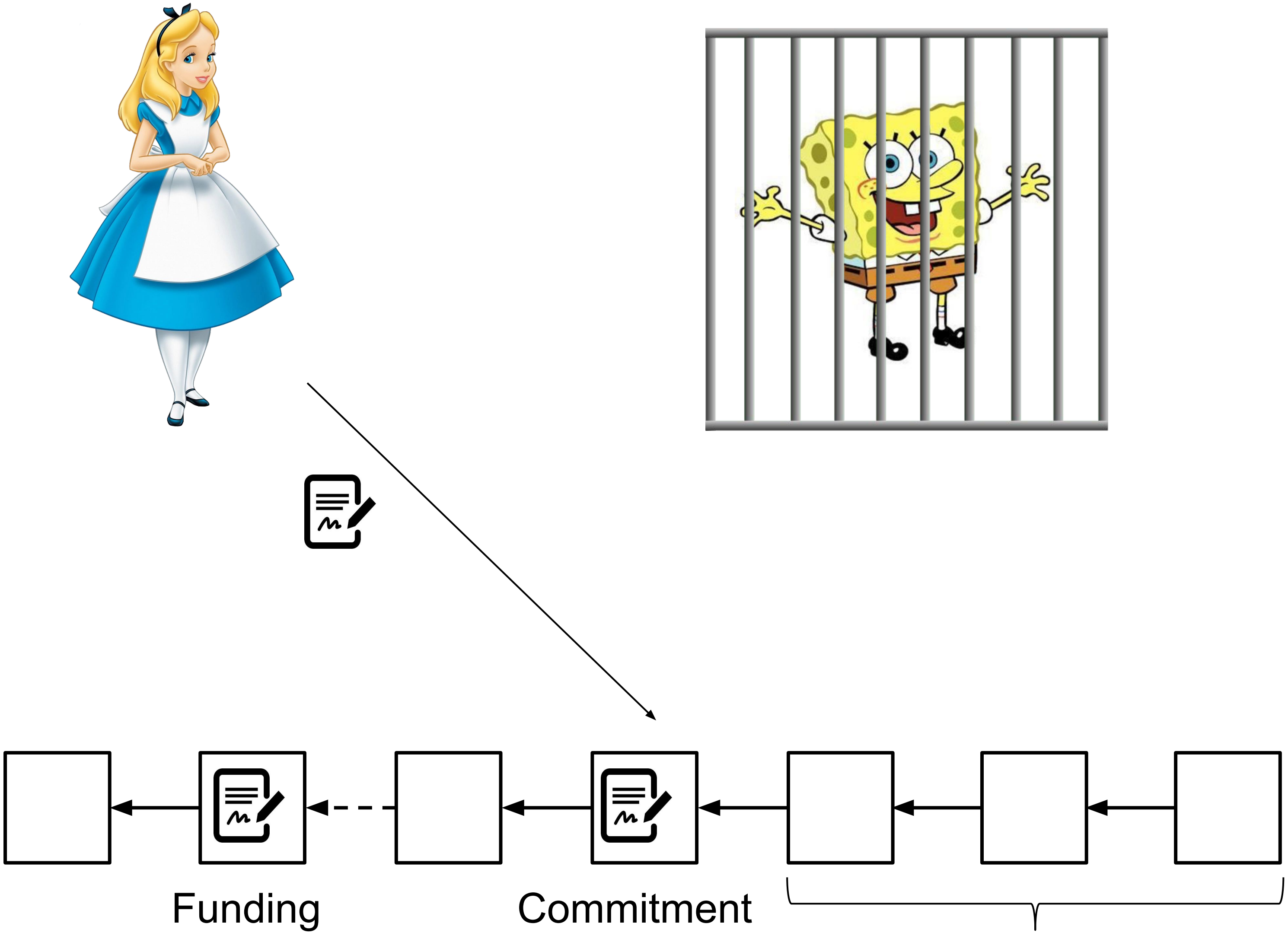


Funding

Commitment

Dispute period

Inactive Counter Party

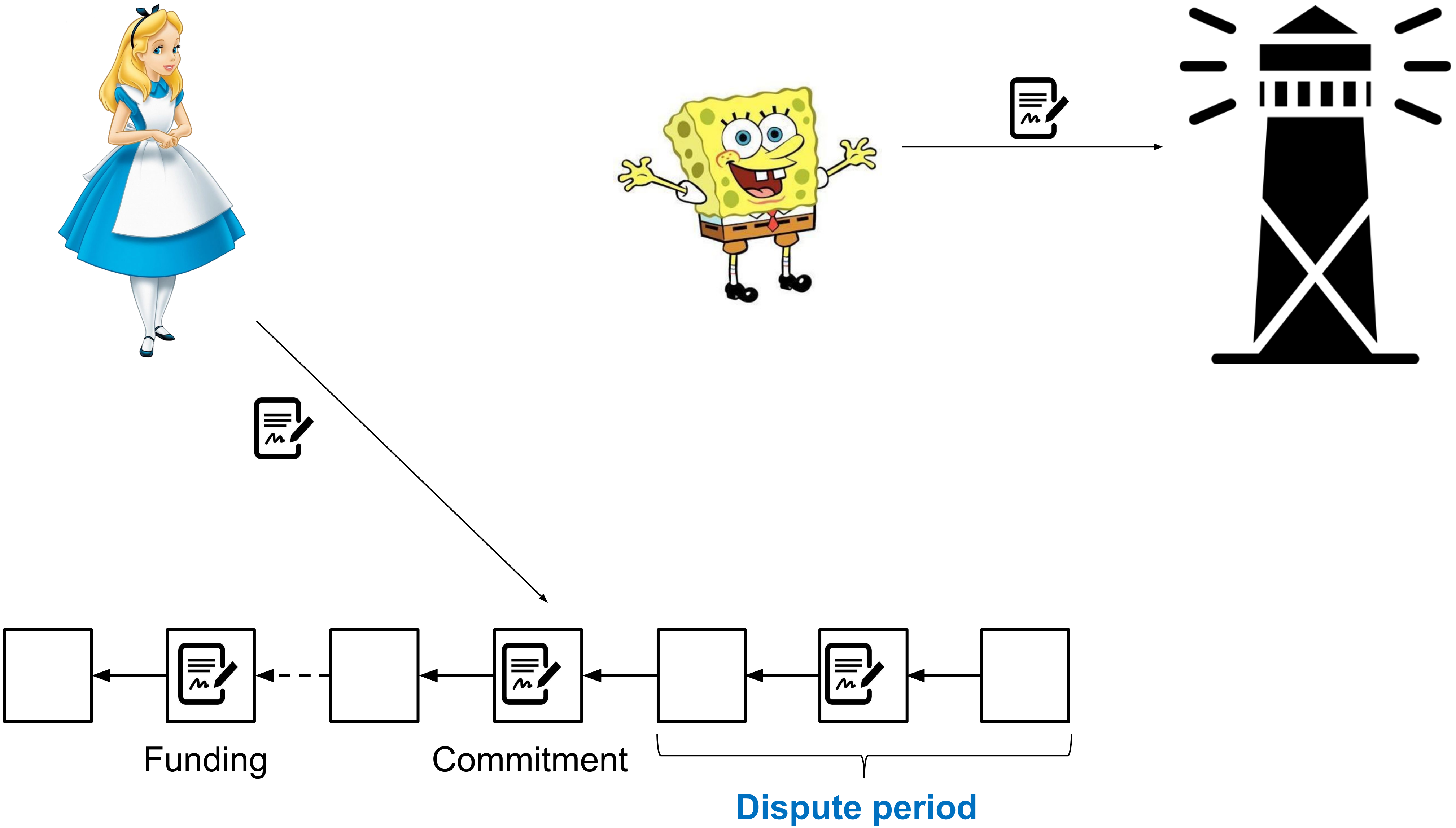


Funding

Commitment

Dispute period

Watchtowers

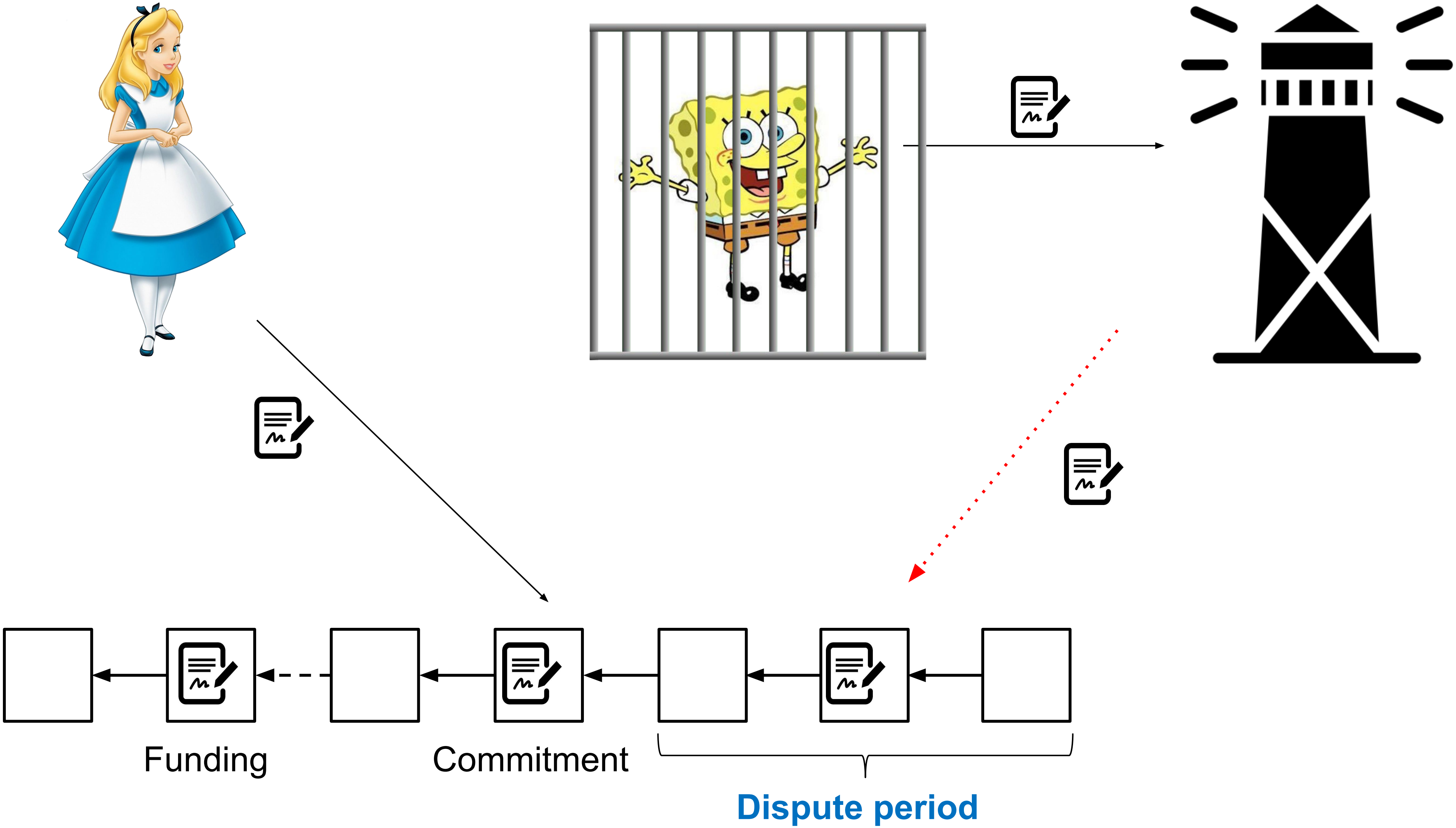


Funding

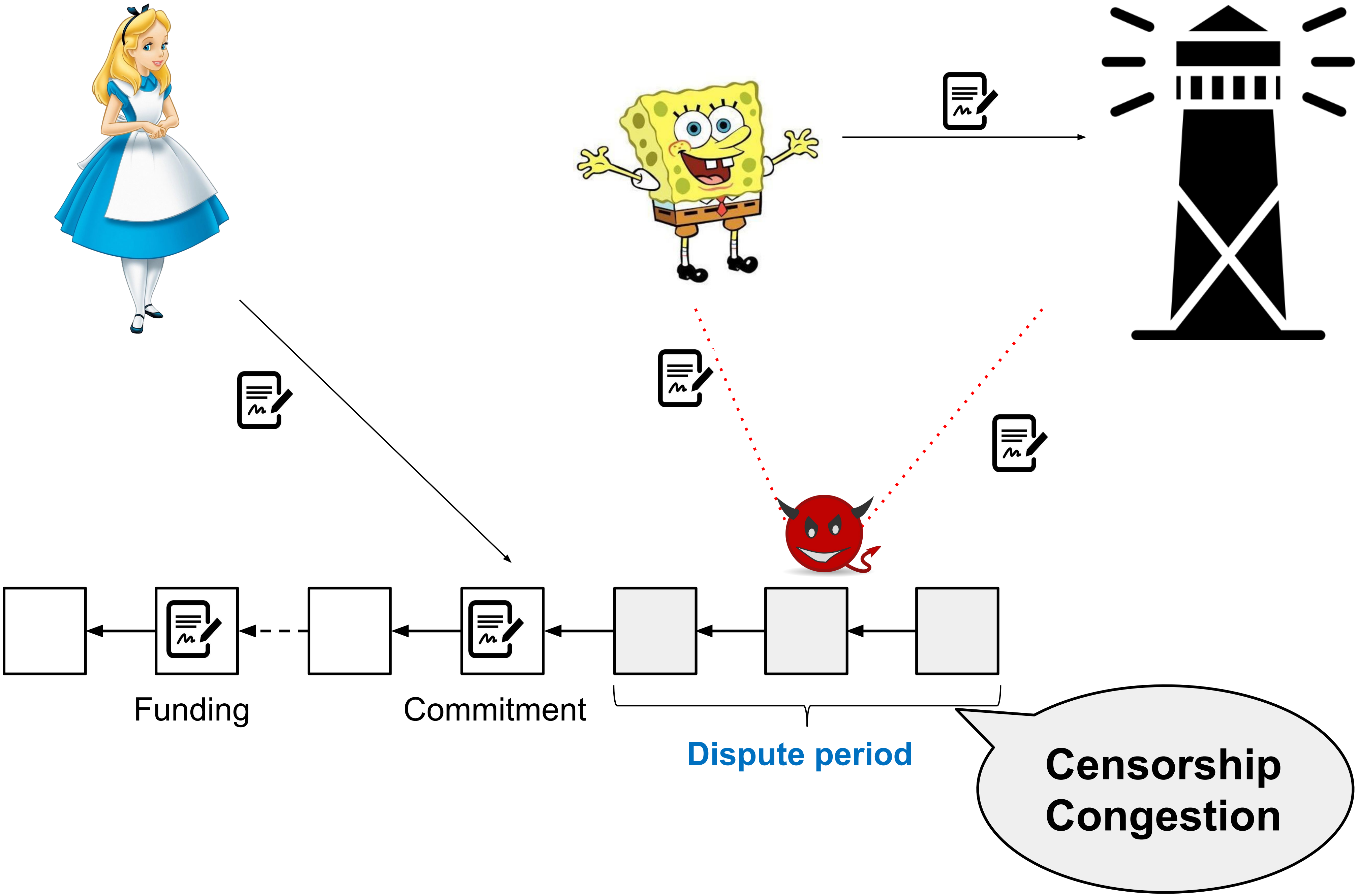
Commitment

Dispute period

Watchtowers



Attack the Liveness of the Blockchain



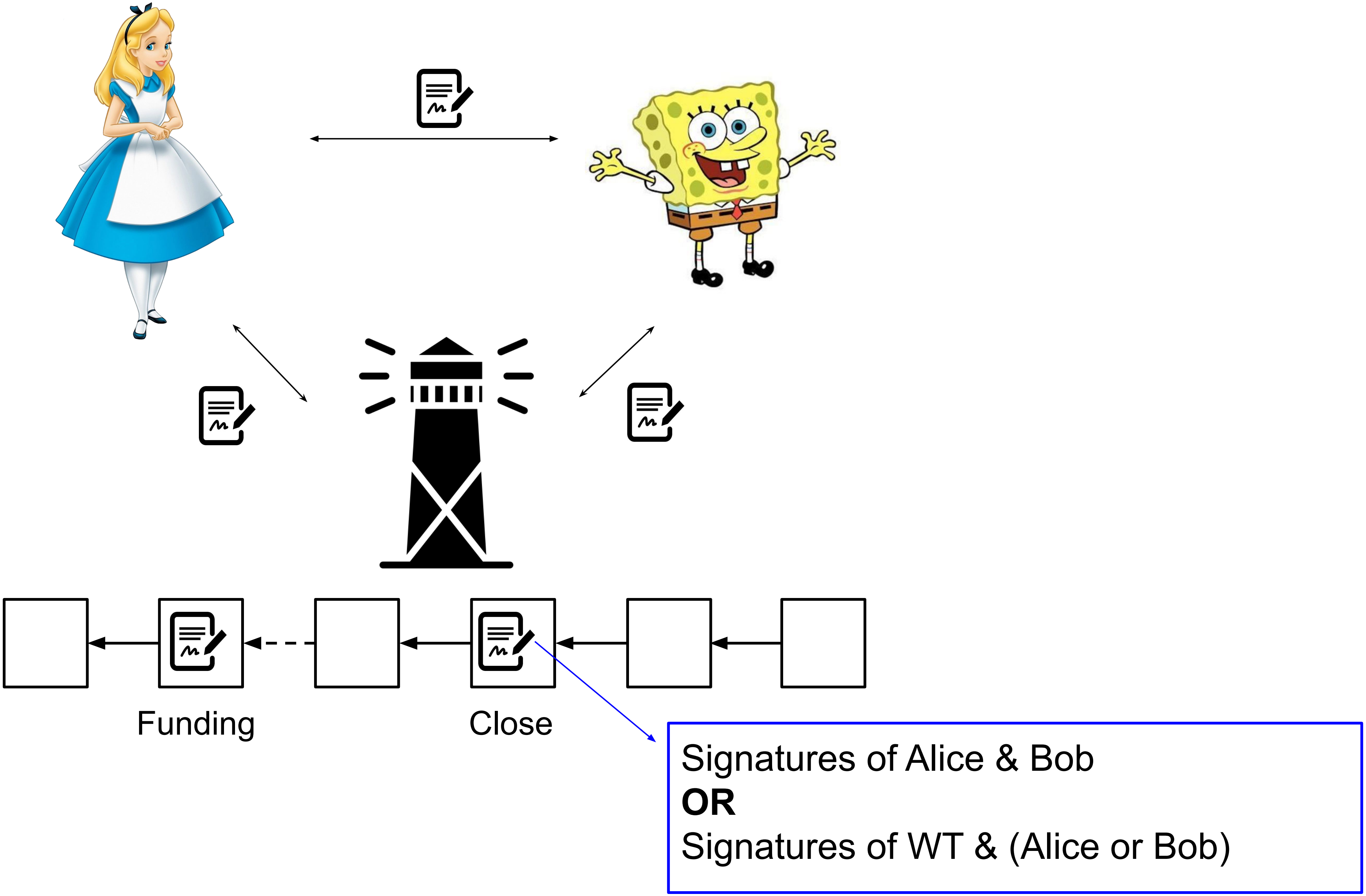
Time = CryptoMoney!

Time = CryptoMoney!

Asynchronous channels?

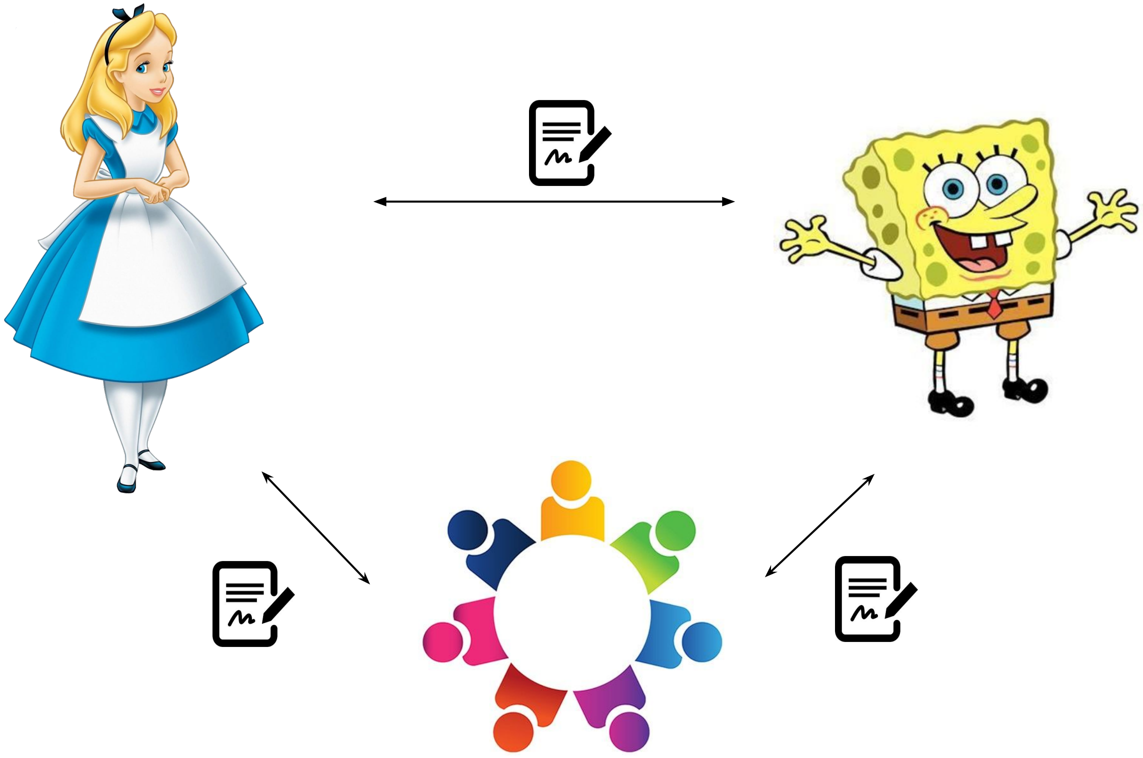
Be proactive, not reactive

Be proactive, not reactive

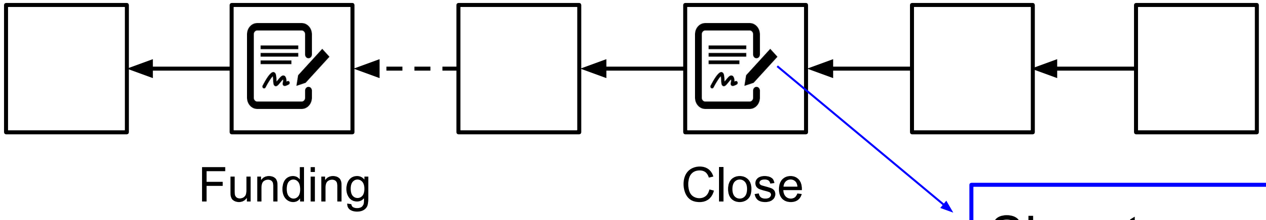


Signatures of Alice & Bob
OR
Signatures of WT & (Alice or Bob)

Watchtower Committee

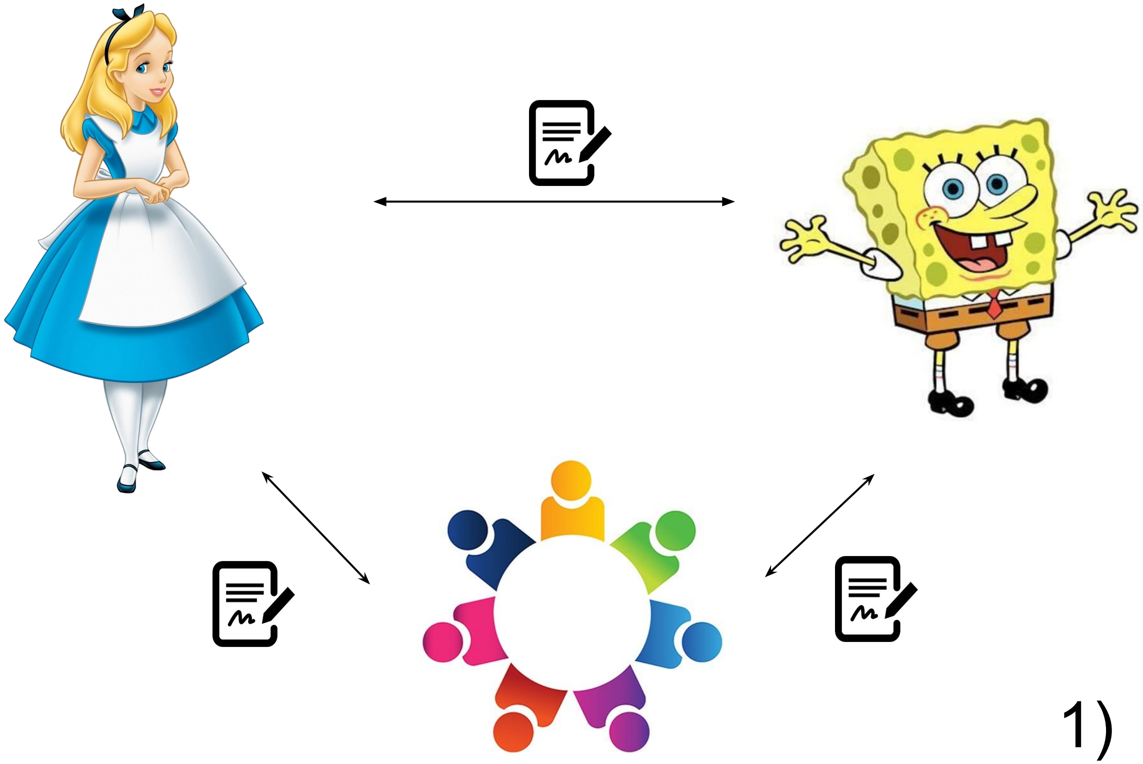


Committee
n = 3f+1
f Byzantine



Signatures of Alice & Bob
OR
Signatures of 2f+1 WTs & (Alice or Bob)

Challenges

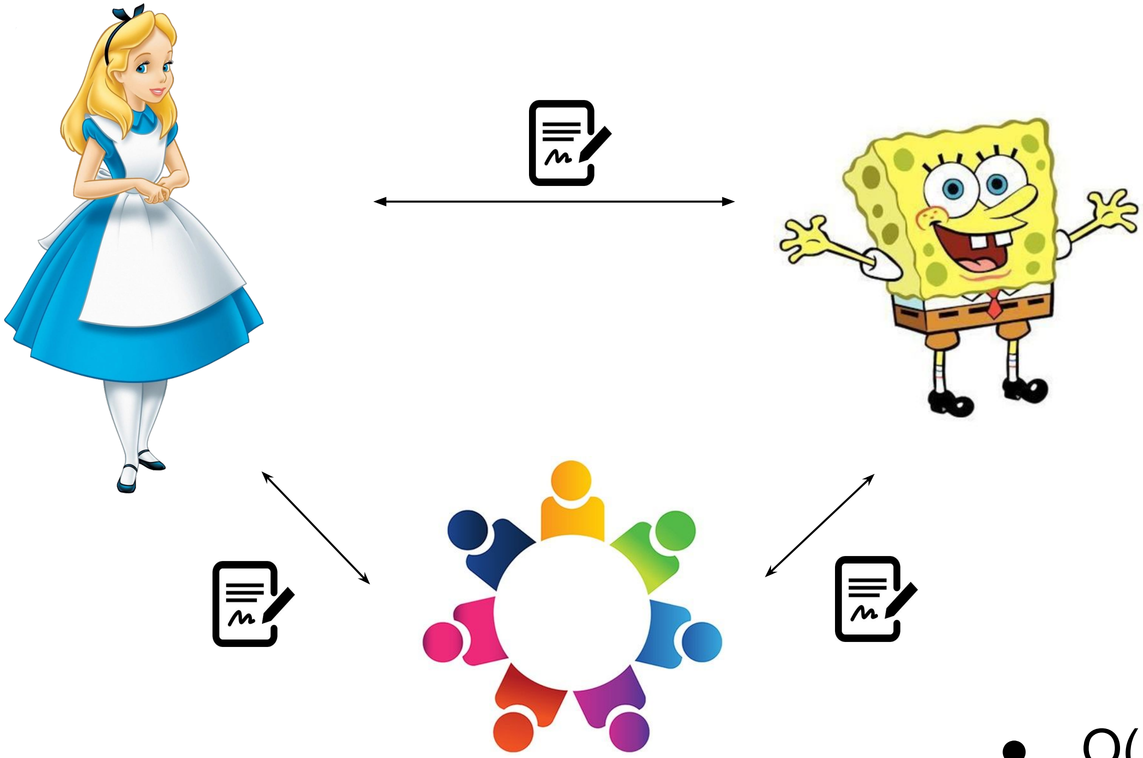


1) Consensus is costly

2) Privacy is important

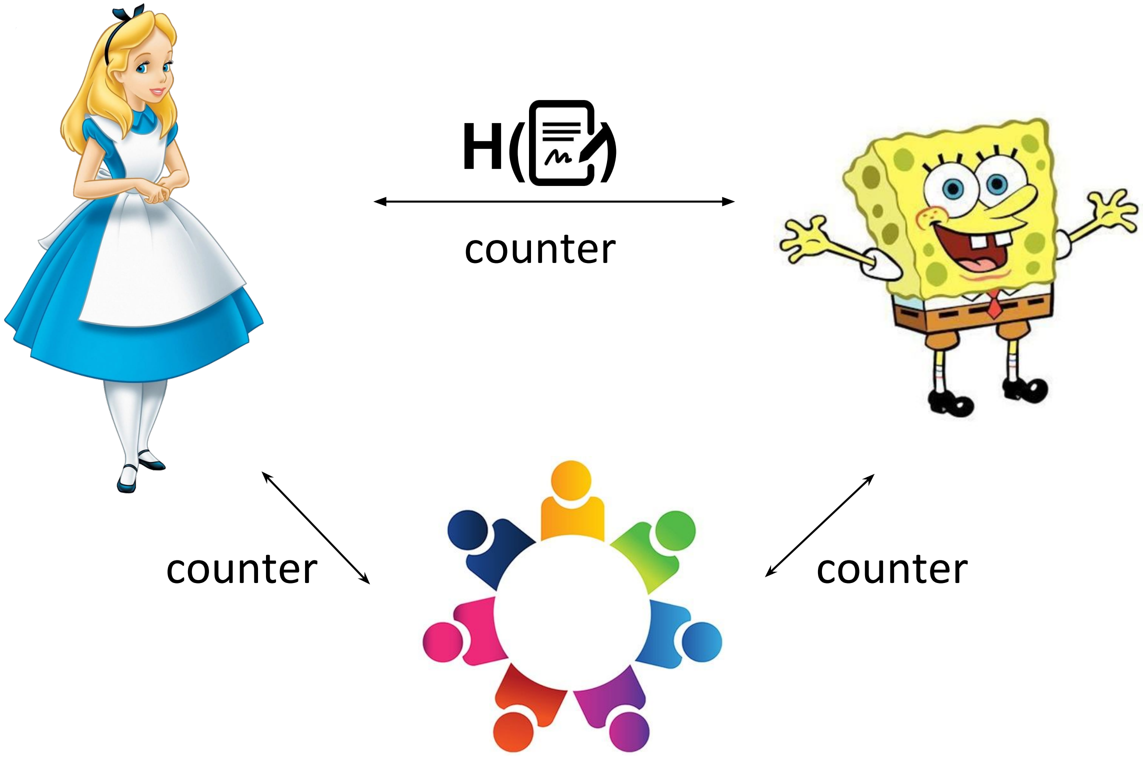
3) Incentives are critical

Consistent Broadcast



- O(n) communication complexity for state updates
- Verification of consensus between Alice & Bob
- No guarantees, if Alice & Bob both misbehave

Privacy



H( )

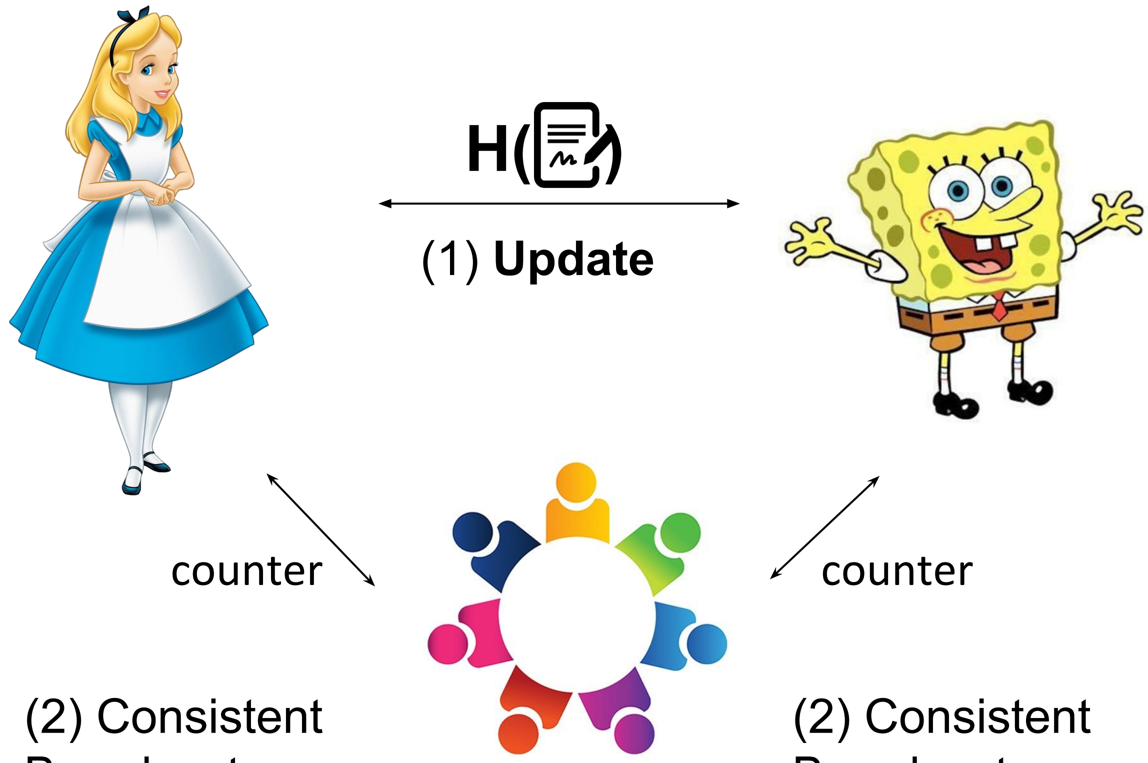
counter

counter

counter

- Privacy preserving
- Alice/Bob cannot publish a previous transaction

Brick Architecture



(3) Execute

H(📝)

(1) **Update**

(3) Execute

counter

counter

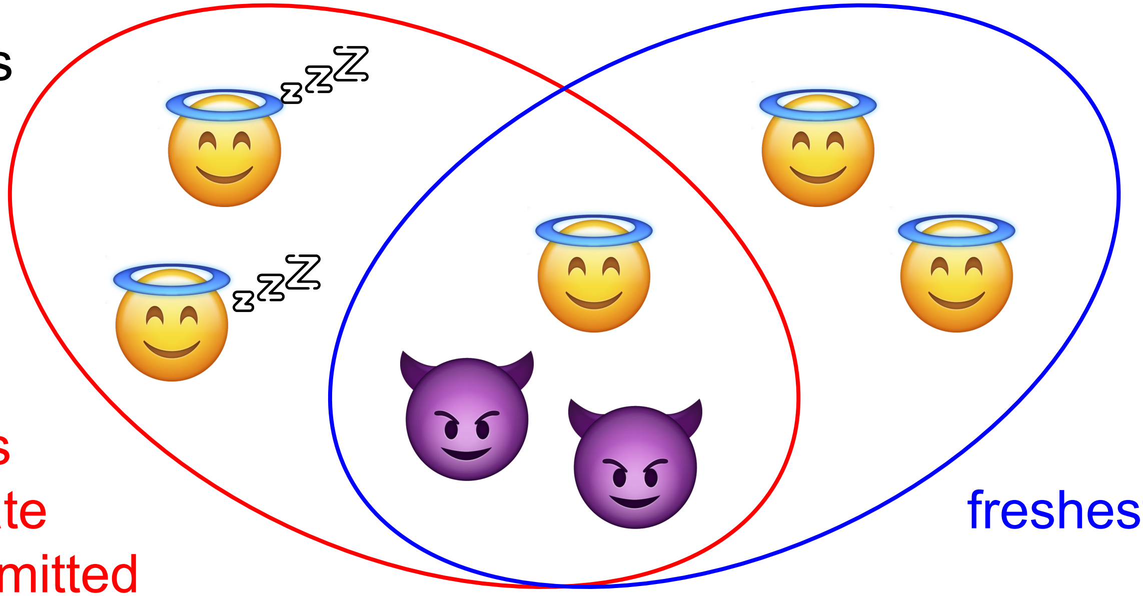
(2) Consistent
Broadcast

(2) Consistent
Broadcast

Close: max state of 2f+1 submitted states.

Brick Security Analysis

Safety
A channel will only close in the freshest committed state



f slow honest WTs

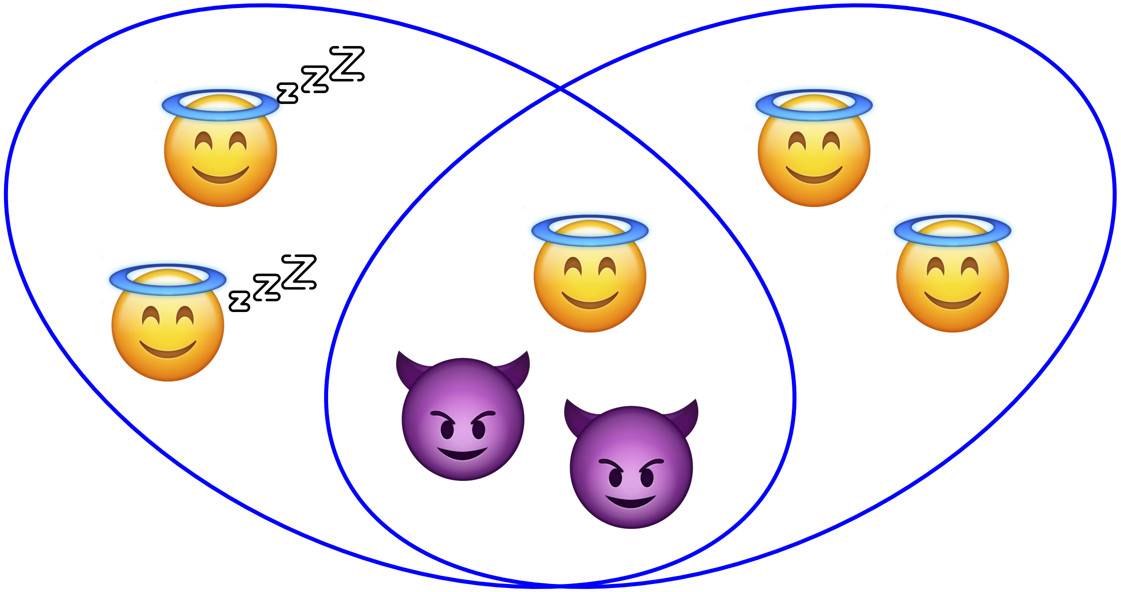
2f+1 WTs
closing state
(previous committed state)

2f+1 WTs
freshest committed state

Brick Security Analysis

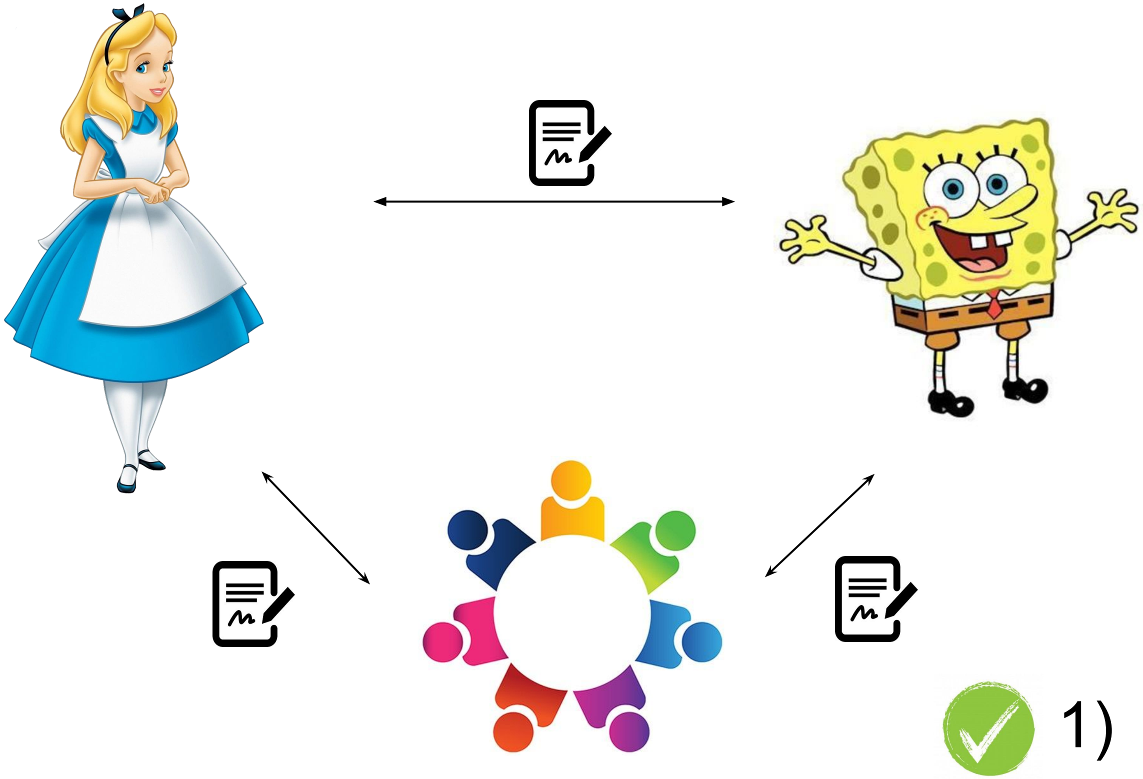
Liveness

Any valid operation (close, update)
will eventually be committed



Not committed = Invalid operation (failed verification)

Challenges



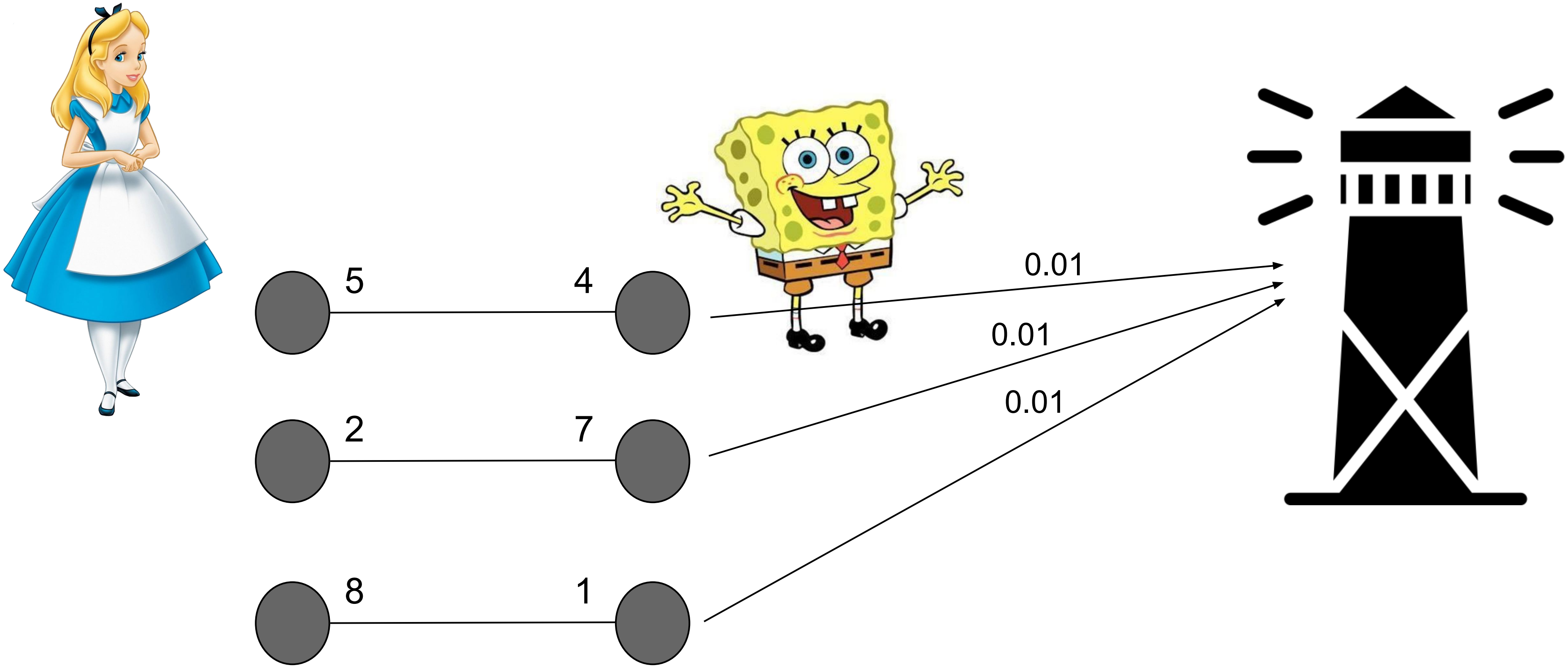
✓ 1) Consensus is costly

✓ 2) Privacy is important

3) Incentives are critical

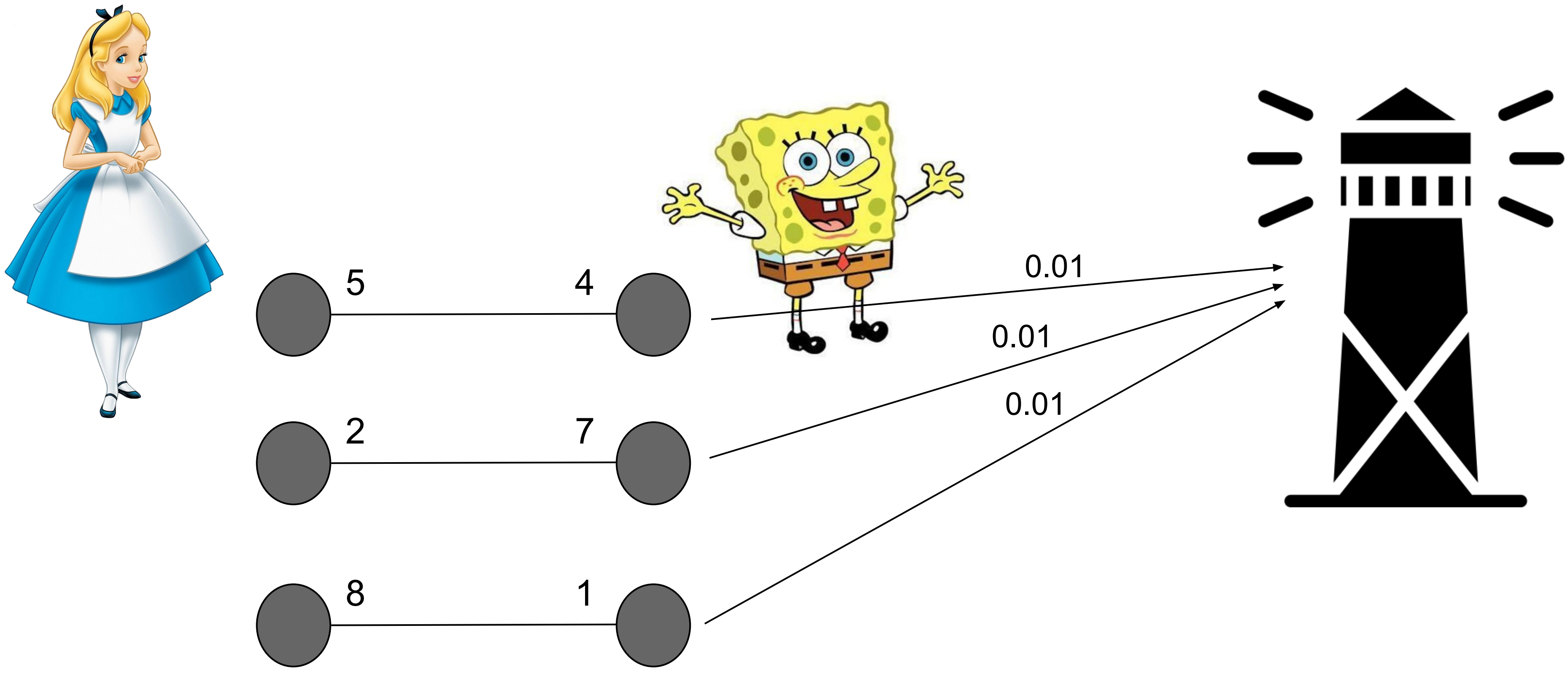
Why be a Watchtower?

Per-update fees



Repeated game lifts the fair-exchange impossibility

Per-update fees



Watchtower paid while channel is alive!
Incentives to close?

Why assist to close honestly?

Collateral

Why assist to close honestly?

Collateral

Asynchronous channels?

Collateral

Fraud proofs
two signed conflicting states

Party claims the collateral

Collateral

Fraud proofs
two signed conflicting states

Party claims the collateral

channel value
v

claimed collateral
$v/f * (f+1)$

Collateral

Where do we close?
when **>f fraud proofs** are submitted

all channel value→ counterparty

Collateral

Where do we close?
when **≤f fraud proofs** are submitted

run close again without the malicious → max state of 2f+1

Collateral

Profit =

channel balance (c) + fraud proofs (v/f) - bribes (v/f + ε)

v = channel value
f = Byzantine watchtowers
y = bribed watchtowers

Collateral

Profit =
channel balance (c) + fraud proofs (v/f) - bribes ($v/f + \varepsilon$)

1. **0 FPs**: profit = $c \leq v$

v = channel value
f = Byzantine watchtowers
y = bribed watchtowers

Collateral

Profit =
channel balance (c) + fraud proofs (v/f) - bribes ($v/f + \varepsilon$)

1. **0 FPs**: profit = $c \le v$

2. **> f FPs**: profit $\le v + y*v/f - y*(v/f-\varepsilon) = v - \varepsilon$

v = channel value
f = Byzantine watchtowers
y = bribed watchtowers

Collateral

Profit =
channel balance (c) + fraud proofs (v/f) - bribes (v/f + ε)

1. **0 FPs**: profit = $c \leq v$

2. **> f FPs**: profit $\leq v + y*v/f - y*(v/f-\varepsilon) = v - \varepsilon$

3. **f FPs and “correct” close:** profit = $c + v$

v = channel value
f = Byzantine watchtowers
y = bribed watchtowers

Collateral

Will a party close in a “incorrect” state?

Action	Proof-of-fraud	Close	Total
Byzantine	m	$f - m$	f
Bribed (rational)	y	$f + 1 - (f - m)$ $= m + 1$	$y + m + 1$
Total	$m + y$	$f + 1$	-

$$\textbf{profit} = \underbrace{v \quad +}_{\textbf{channel value}} \quad \underbrace{(m+y)*v/f}_{\textbf{fraud proofs}} \quad - \quad \underbrace{(y+m+1)*(v/f+\varepsilon)}_{\textbf{bribes}} \quad \leq \quad v - v/f - \varepsilon \boldsymbol{< c + v}$$

Collateral

Profit =
channel balance (c) + fraud proofs (v/f) - bribes (v/f + ε)

1. **0 FPs**: profit = $c \leq v$

2. **> f FPs**: profit ≤ $v + y*v/f - y*(v/f-\varepsilon) = v - \varepsilon$

3. **f FPs and "correct" close:** profit = $c + v$

4. **f FPs and "incorrect" close:** profit = $v - v/f - \varepsilon$

v = channel value
f = Byzantine watchtowers
y = bribed watchtowers

Why assist to close?

WTs collude → Hostage situations

Closing fees

prisoner’s dilemma

Why request close?

Parties collude → Hostage situations

Committee size > 7

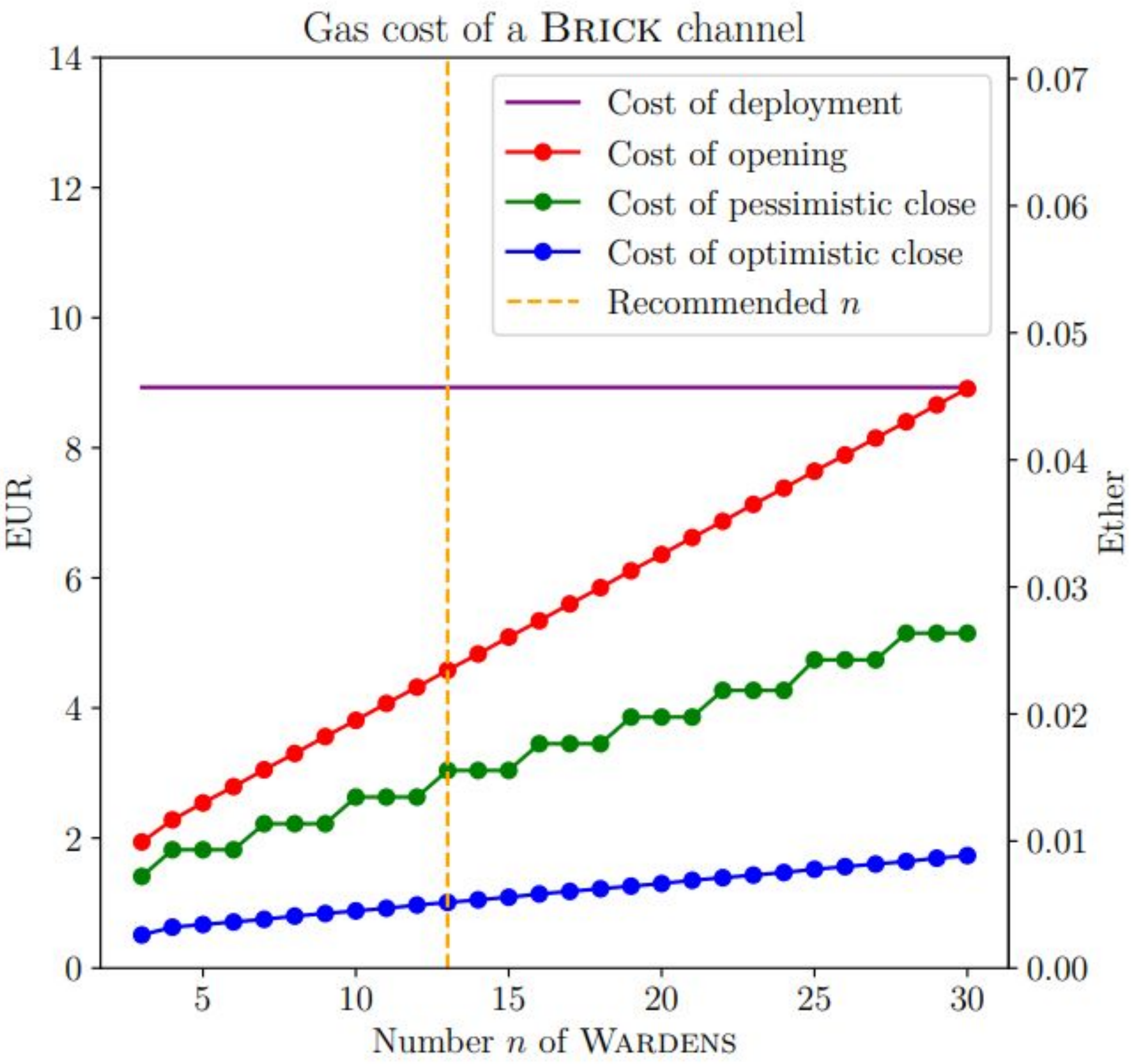
richest party loses more

Committee size

The more (WTs) the merrier!

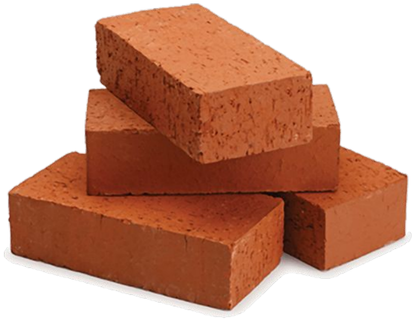
↑ robustness
↓ collateral per WT
≃ cost for parties

Brick Cost


Gas cost of a BRICK channel
EUR
0
2
4
6
8
10
12
14
Ether
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
Number n of WARDENS
5
10
15
20
25
30
Cost of deployment
Cost of opening
Cost of pessimistic close
Cost of optimistic close
Recommended n

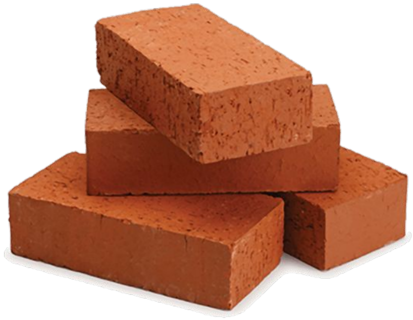
Brick Advantages

- Privacy
- Incentive-compatible
- Good performance
- **Asynchronous**
  - censorship
  - congestion
  - liveness attacks



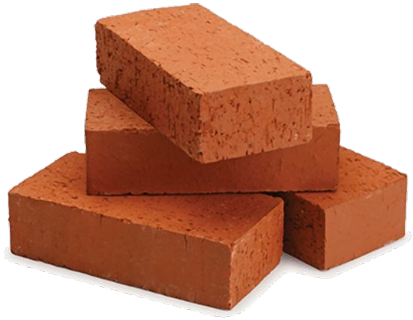
Limitations, Extensions & Future Work

- Minimum collateral
- Update fees via one-way channel



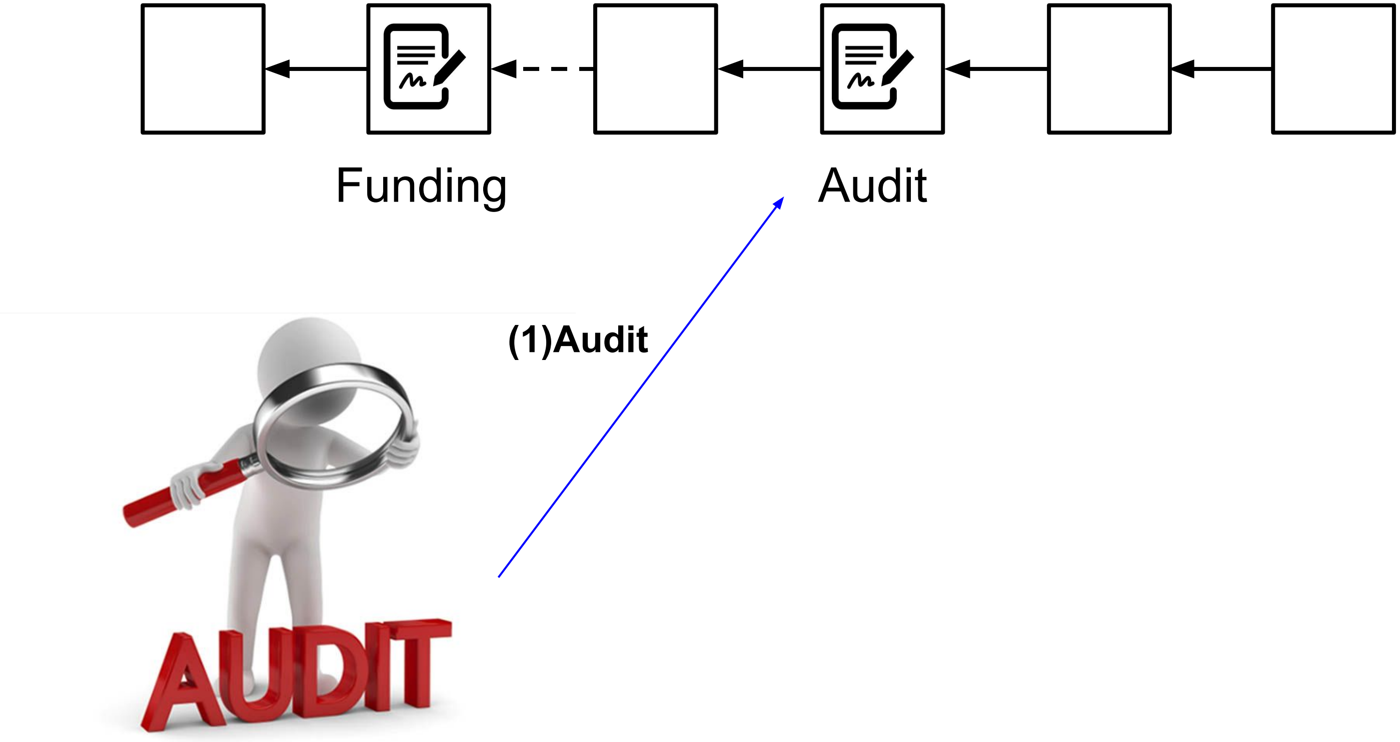
Limitations, Extensions & Future Work

- Minimum collateral
- Update fees via one-way channel
- Watchtower replacement
- Consensus → fork resilient
- Auditability



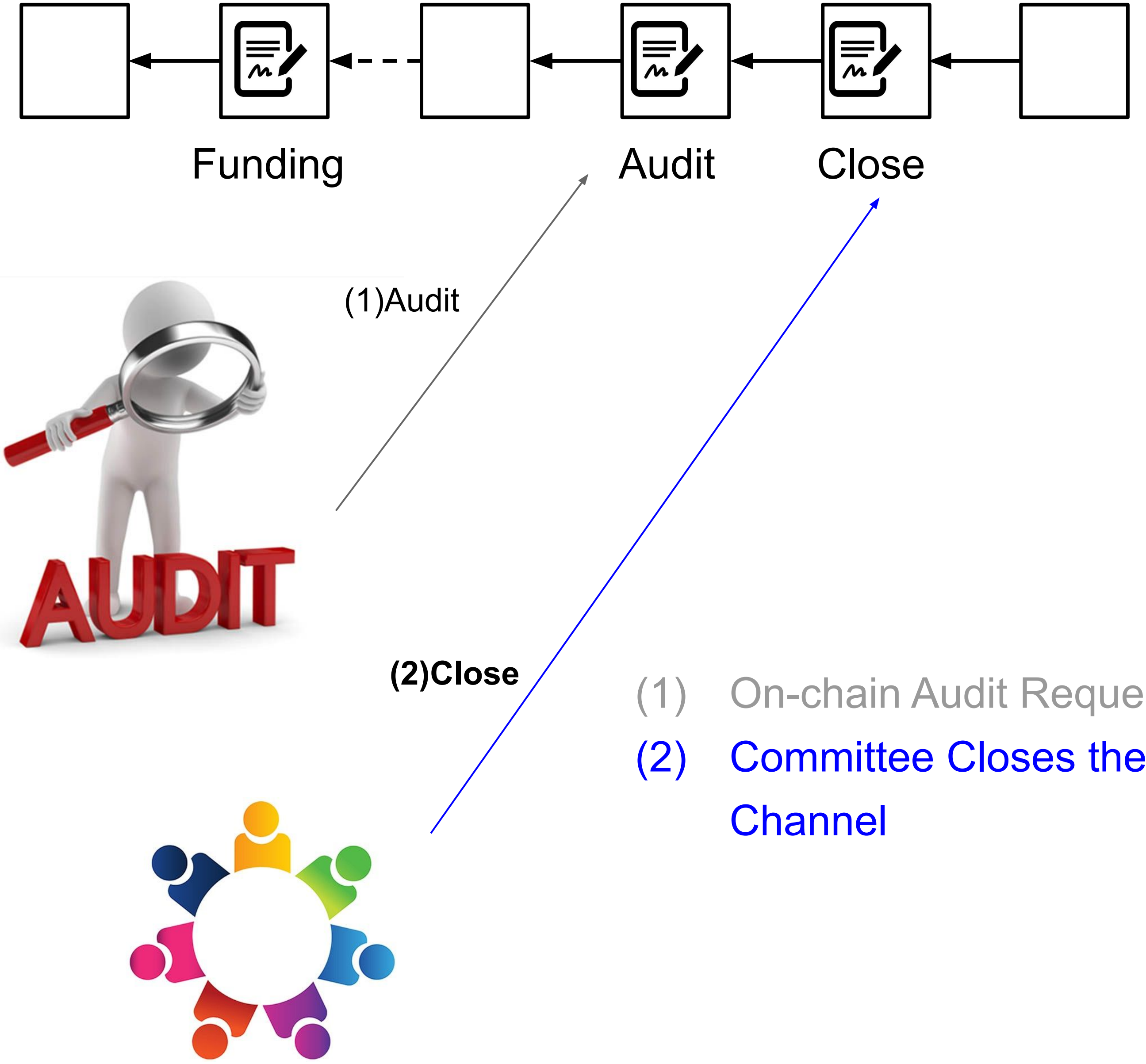
Brick+

Brick+ Workflow



(1) On-chain Audit Request

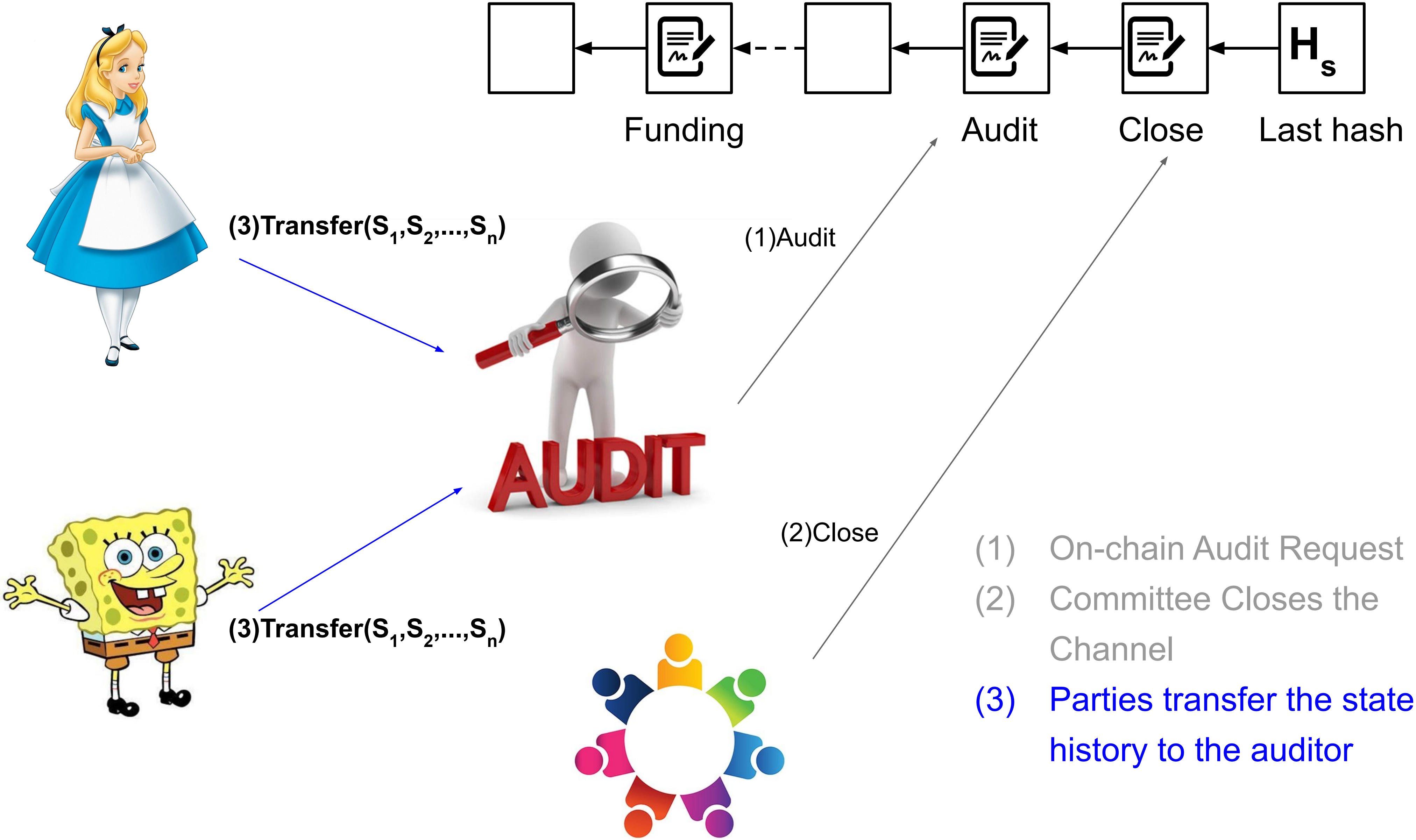
Brick+ Workflow



(1) On-chain Audit Request

(2) Committee Closes the Channel

Brick+ Workflow



H_s

Funding

Audit

Close

Last hash

(3)Transfer(S_1,S_2,...,S_n)

(1)Audit

(2)Close

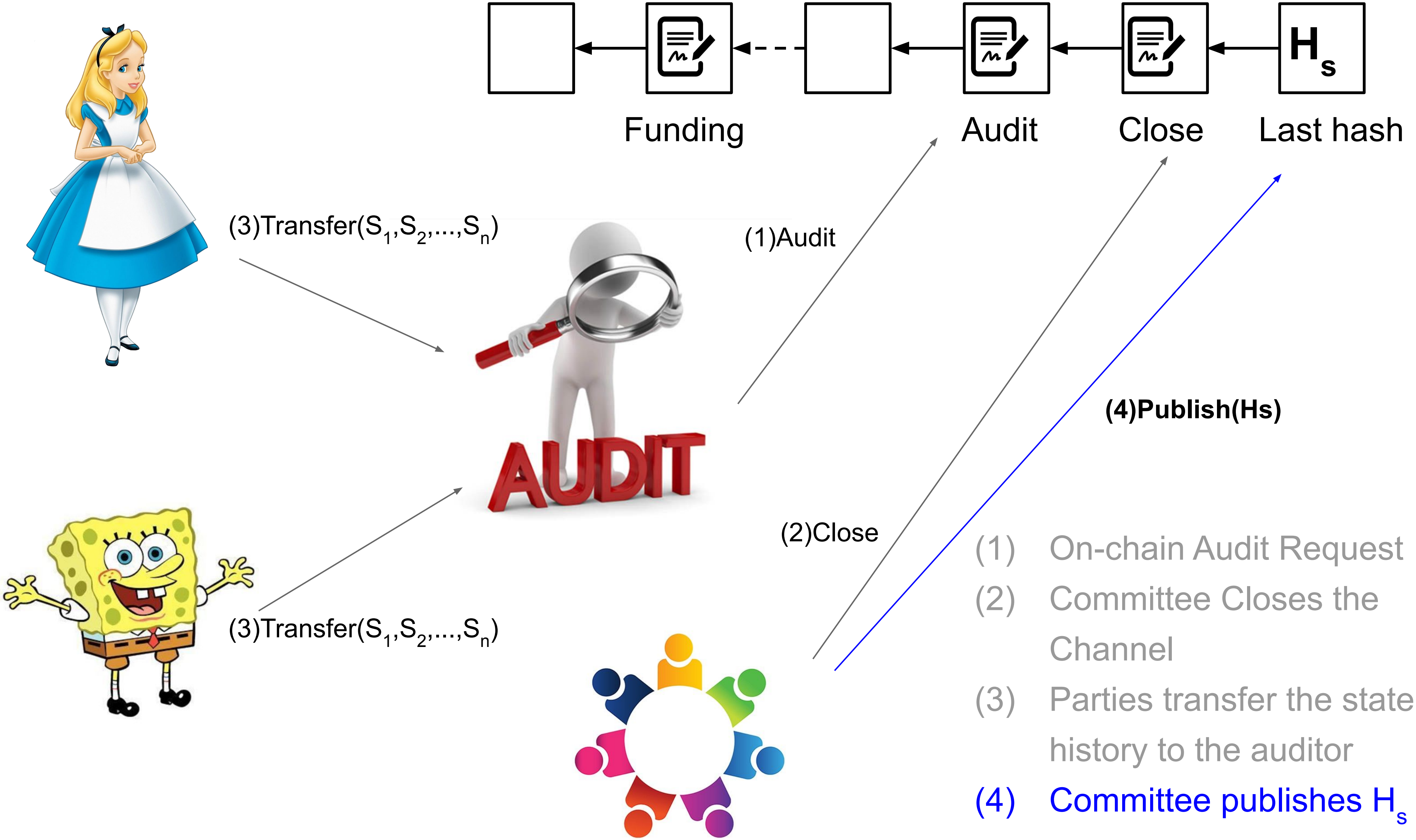
(3)Transfer(S_1,S_2,...,S_n)

(1) On-chain Audit Request

(2) Committee Closes the Channel

(3) Parties transfer the state history to the auditor

Brick+ Workflow


H_s
Funding
Audit
Close
Last hash
(3)Transfer(S_1,S_2,...,S_n)
(1)Audit
AUDIT
(4)Publish(Hs)
(2)Close
(3)Transfer(S_1,S_2,...,S_n)

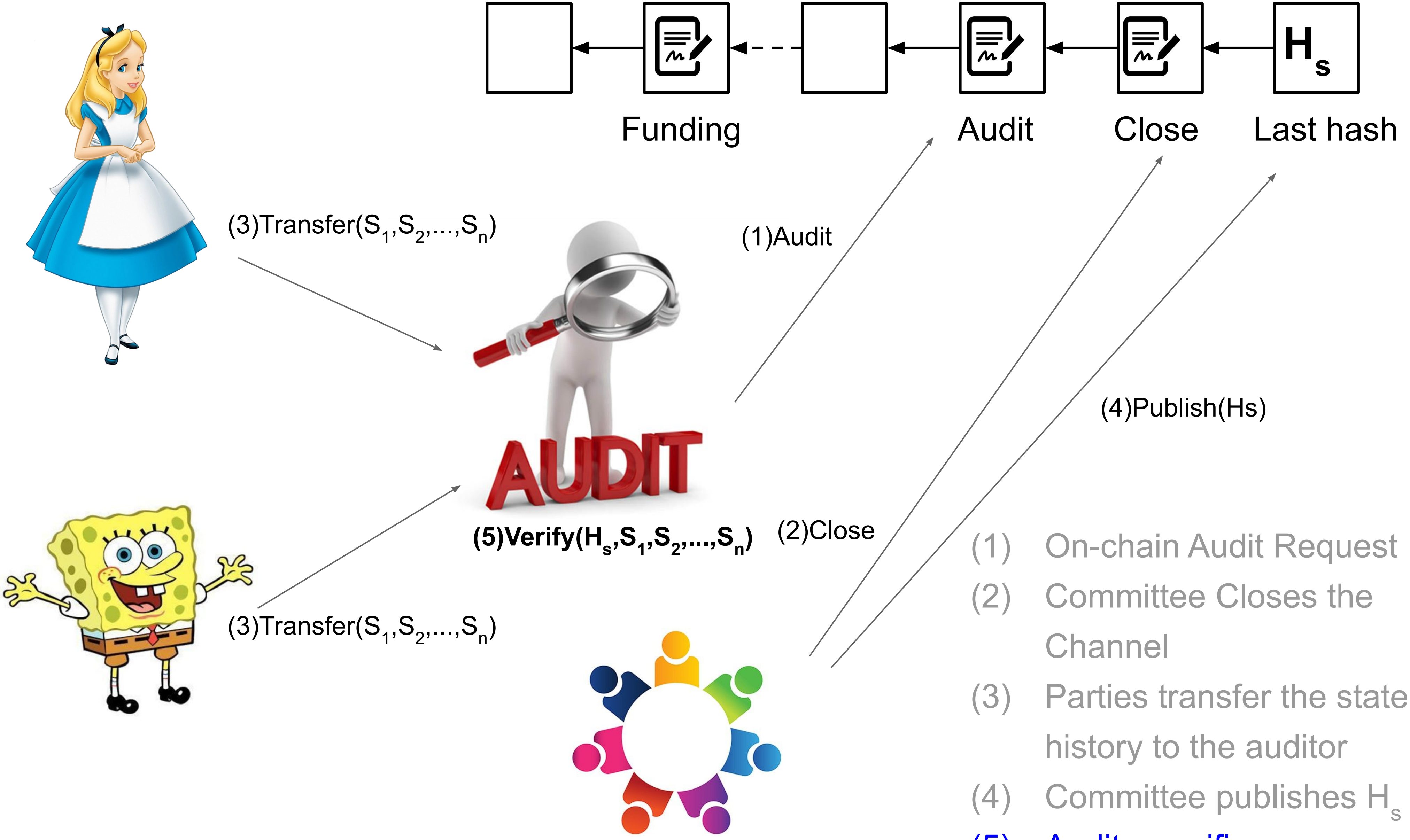
(1) On-chain Audit Request

(2) Committee Closes the Channel

(3) Parties transfer the state history to the auditor

(4) Committee publishes H_s

Brick+ Workflow


H_s
Funding
Audit
Close
Last hash
(3)Transfer(S_1,S_2,...,S_n)
(1)Audit
AUDIT
(4)Publish(Hs)
(5)Verify(H_s,S_1,S_2,...,S_n)
(2)Close
(3)Transfer(S_1,S_2,...,S_n)

(1) On-chain Audit Request

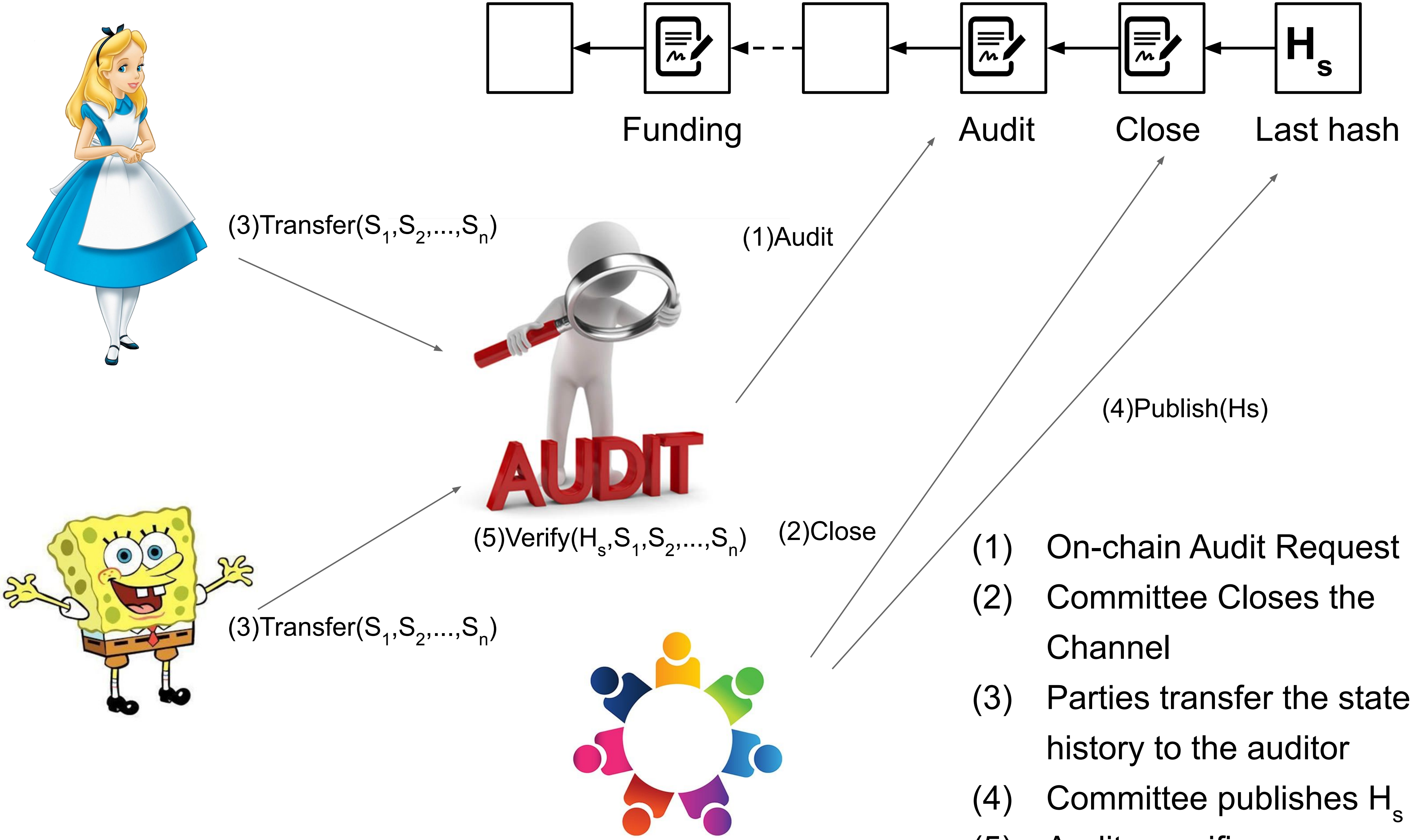
(2) Committee Closes the Channel

(3) Parties transfer the state history to the auditor

(4) Committee publishes H_s

(5) Auditor verifies ($H_{s\,,}$ S_1,S_2,...,S_n)

Brick+ Workflow


H$_s$
Funding
Audit
Close
Last hash
(3)Transfer(S_1,S_2,...,S_n)
(1)Audit
AUDIT
(4)Publish(Hs)
(5)Verify(H_s,S_1,S_2,...,S_n)
(2)Close
(3)Transfer(S_1,S_2,...,S_n)

(1) On-chain Audit Request

(2) Committee Closes the Channel

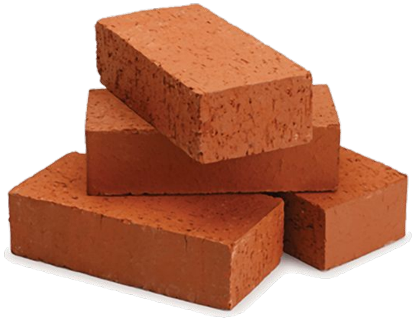
(3) Parties transfer the state history to the auditor

(4) Committee publishes H_s

(5) Auditor verifies (H_s, S_1,S_2,...,S_n)

Limitations, Extensions & Future Work

- Minimum collateral
- Update fees via one-way channel
- Watchtower replacement
- Consensus → fork resilient
- Auditability
- Multiple parties



Thank you!

Questions?

Z. Avarikioti, E. Kokoris-Kogias, R. Wattenhofer, D. Zindros. *Brick: Asynchronous State Channels.* arXiv:1905.11360