A Composable Security Treatment of the Lightning Network

Aggelos Kiayias
Orfeas Stefanos Thyfronitis Litos
University of Edinburgh
24/6/2020

Part 1
The Lightning Network

VISA

20,000 tx/s

bitcoin

7 tx/s

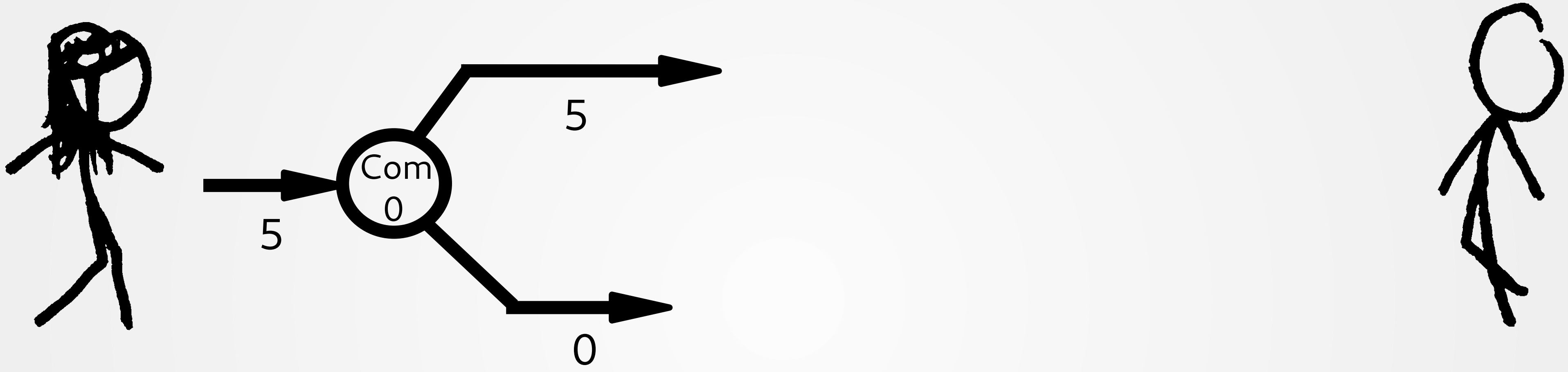
Problem

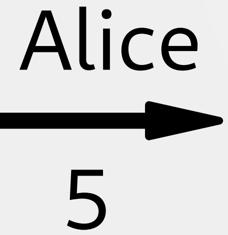
All txs validated by all wallets

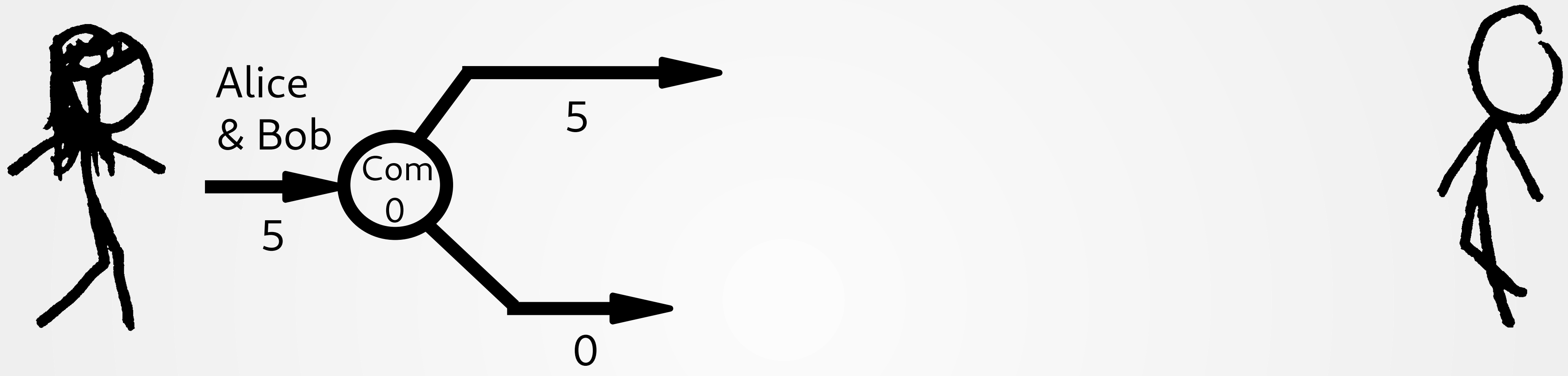
Solution

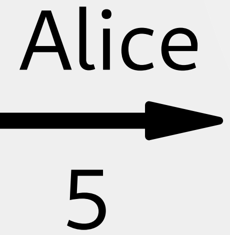
- Move most txs off-chain
- Resolve disputes on-chain

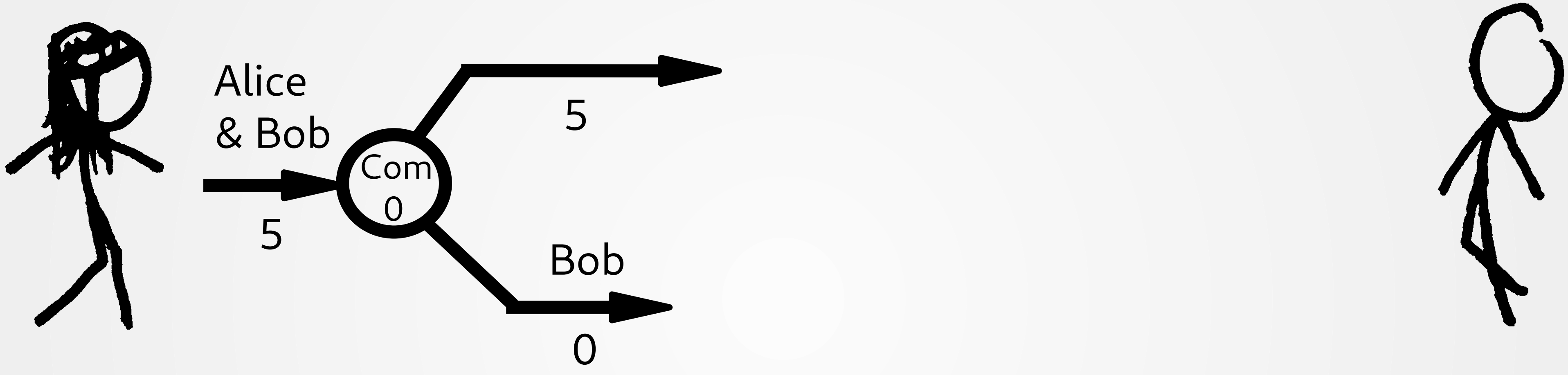
Alice
5

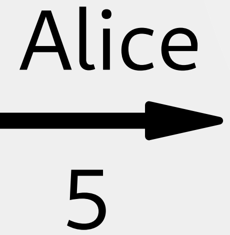

5
Com
0
5
0

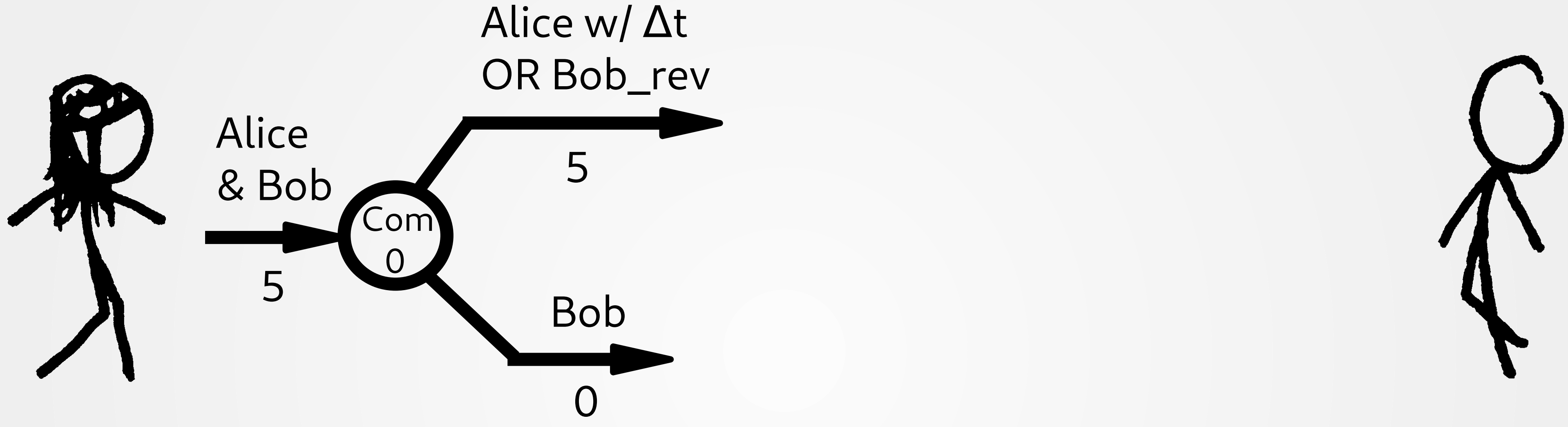

Alice
5

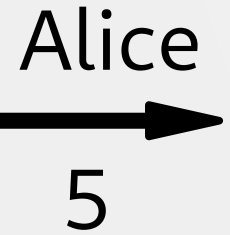

Alice
& Bob
5
Com
0
5
0

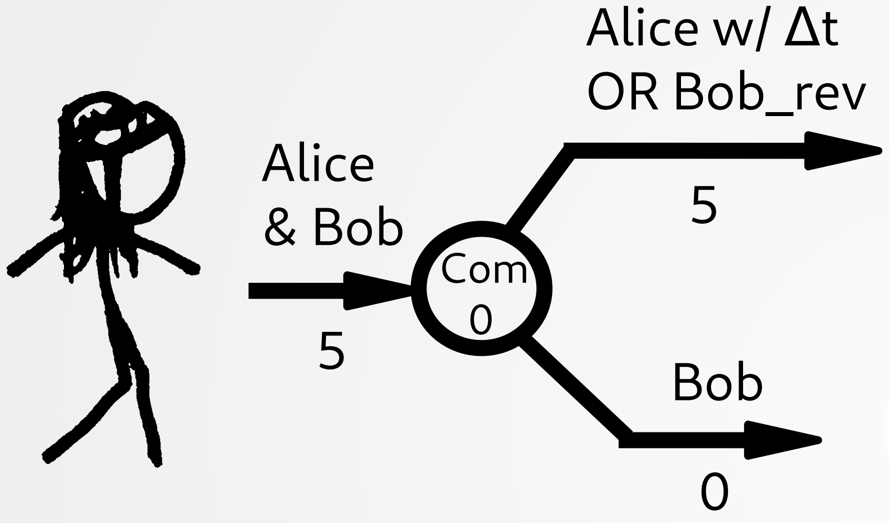

Alice
5

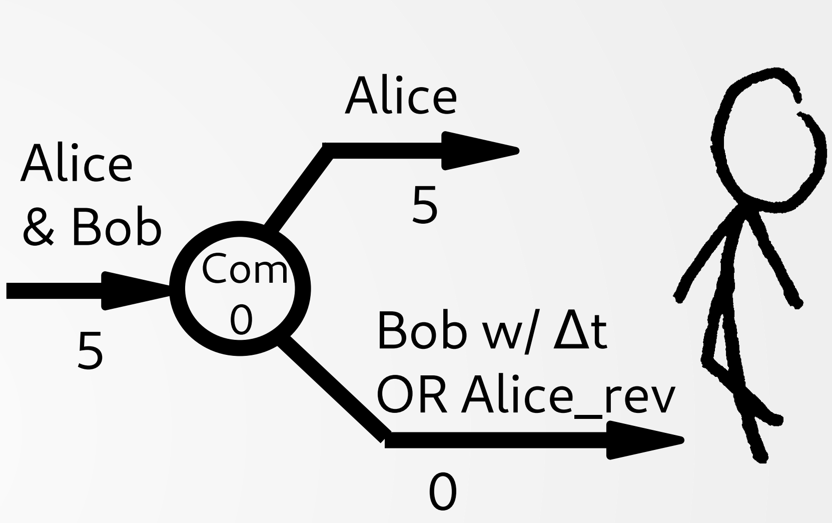

Alice
& Bob
5
Com
0
5
Bob
0


Alice
5

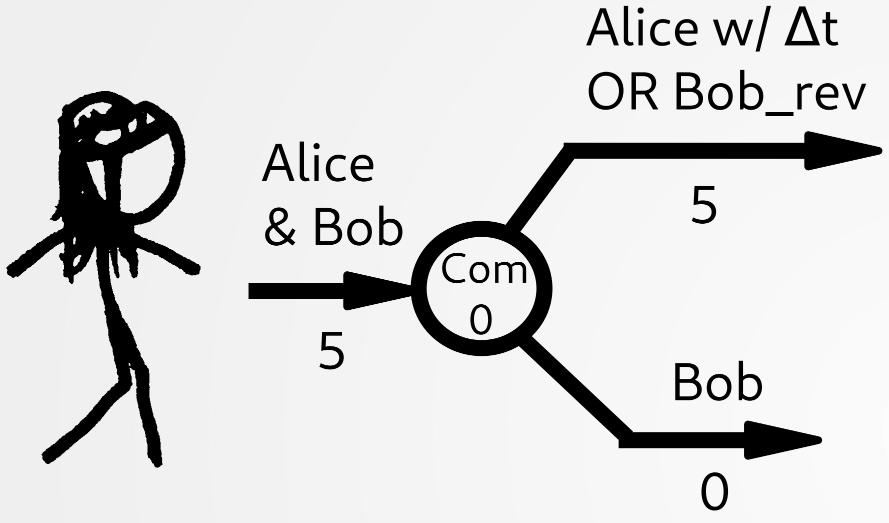

Alice w/ Δt
OR Bob_rev
5
Alice
& Bob
Com
0
5
Bob
0

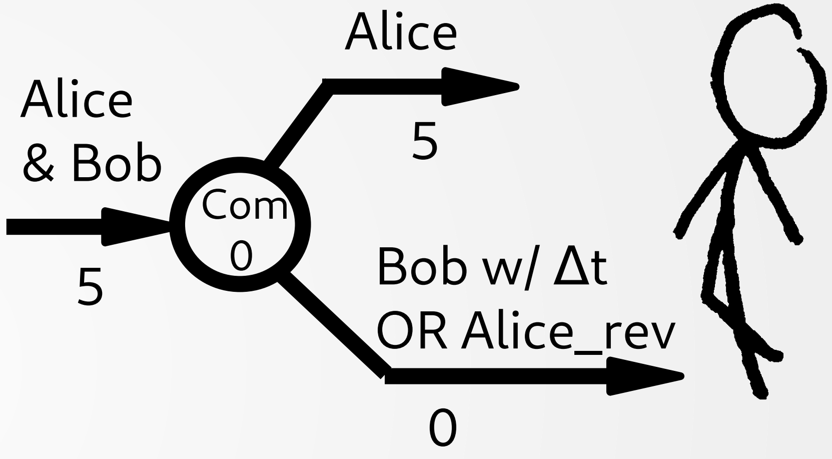

Alice
5

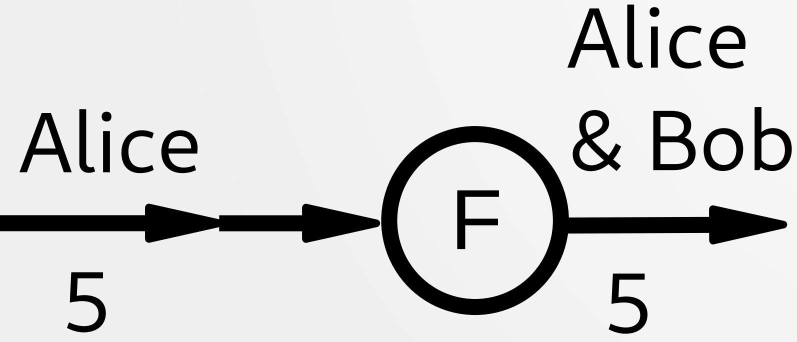

Alice w/ Δt
OR Bob_rev
5
Alice
& Bob
5
Com
0
Bob
0

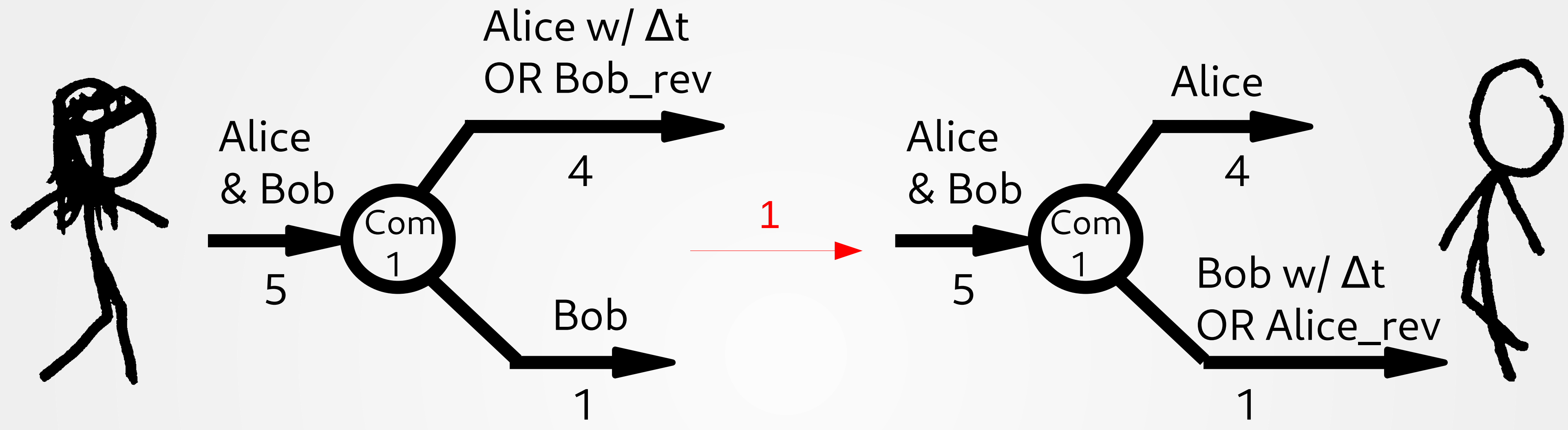

Alice
5
Alice
& Bob
5
Com
0
Bob w/ Δt
OR Alice_rev
0

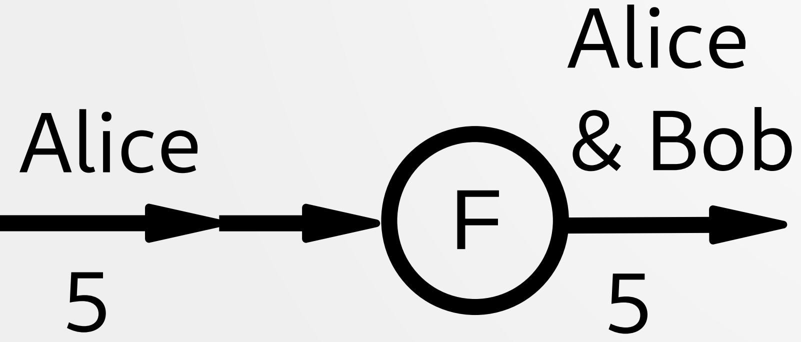
Alice
5

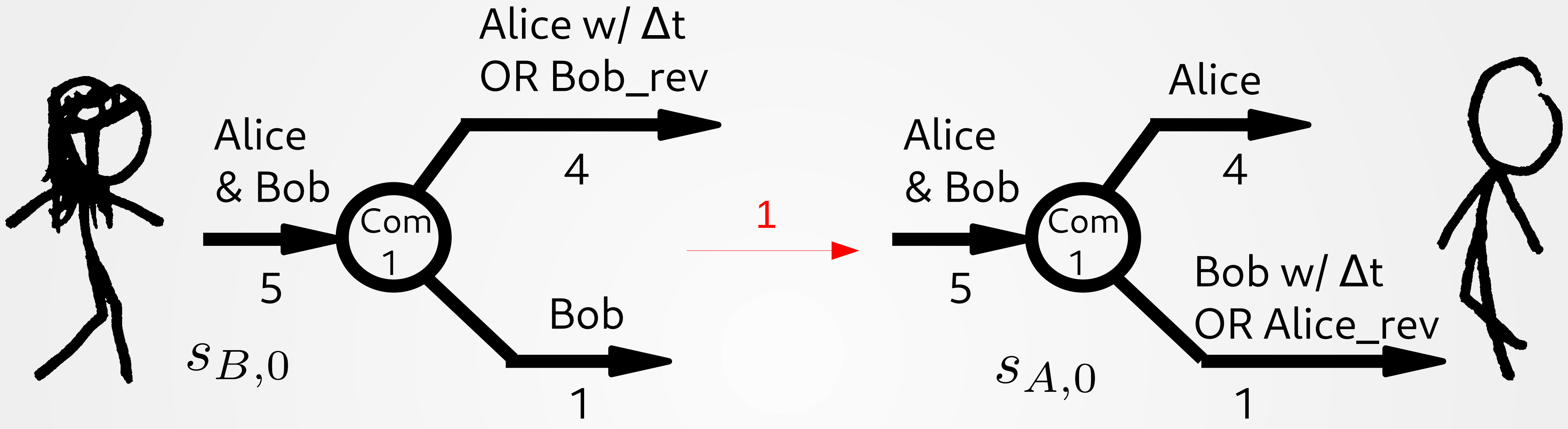

Alice & Bob
5
Com
0
Alice w/ Δt
OR Bob_rev
5
Bob
0

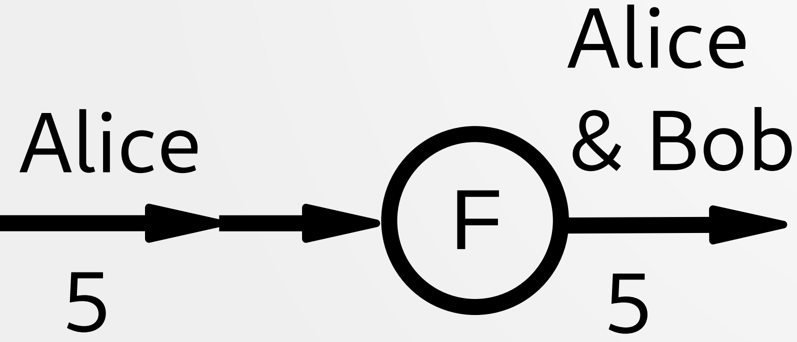

Alice & Bob
5
Com
0
Alice
5
Bob w/ Δt
OR Alice_rev
0

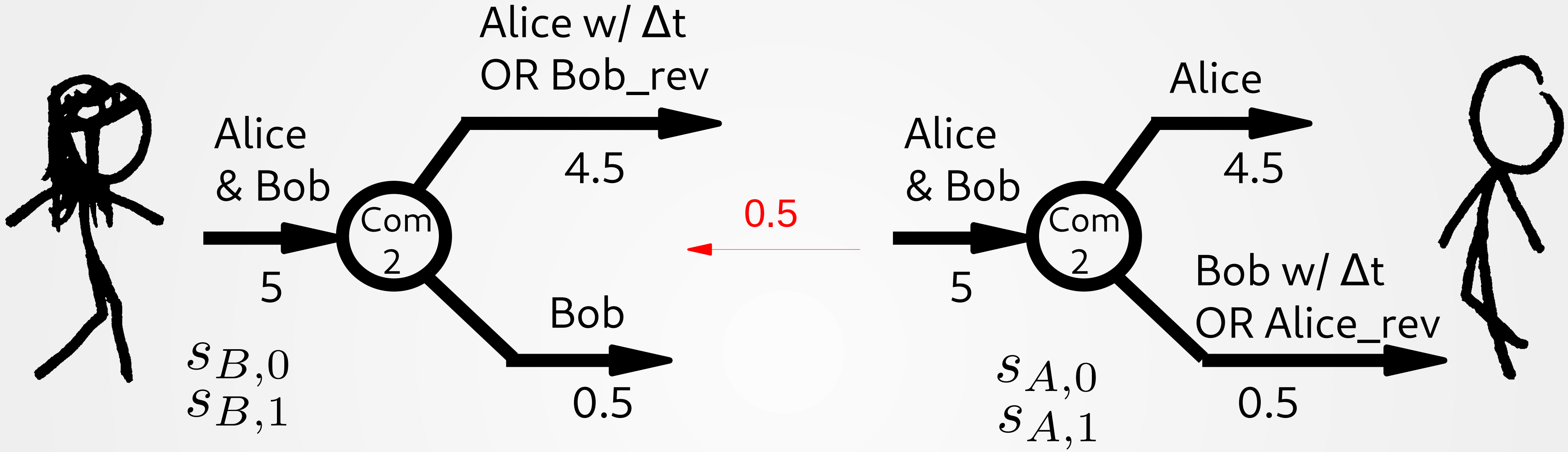

Alice
5
F
Alice & Bob
5

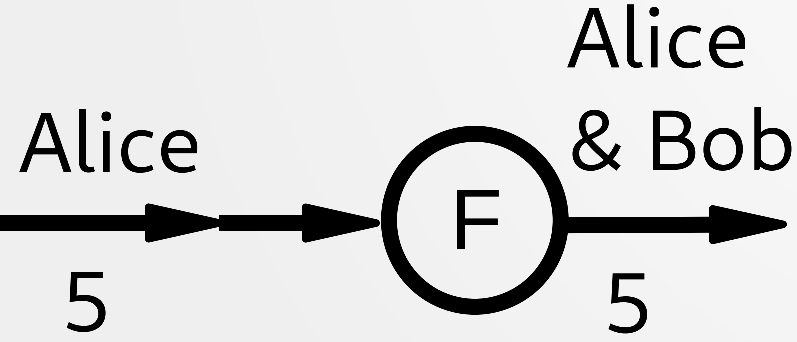

Alice w/ Δt
OR Bob_rev
Alice
& Bob
4
Com
1
5
Bob
1
1
Alice
Alice
& Bob
4
Com
1
5
Bob w/ Δt
OR Alice_rev
1

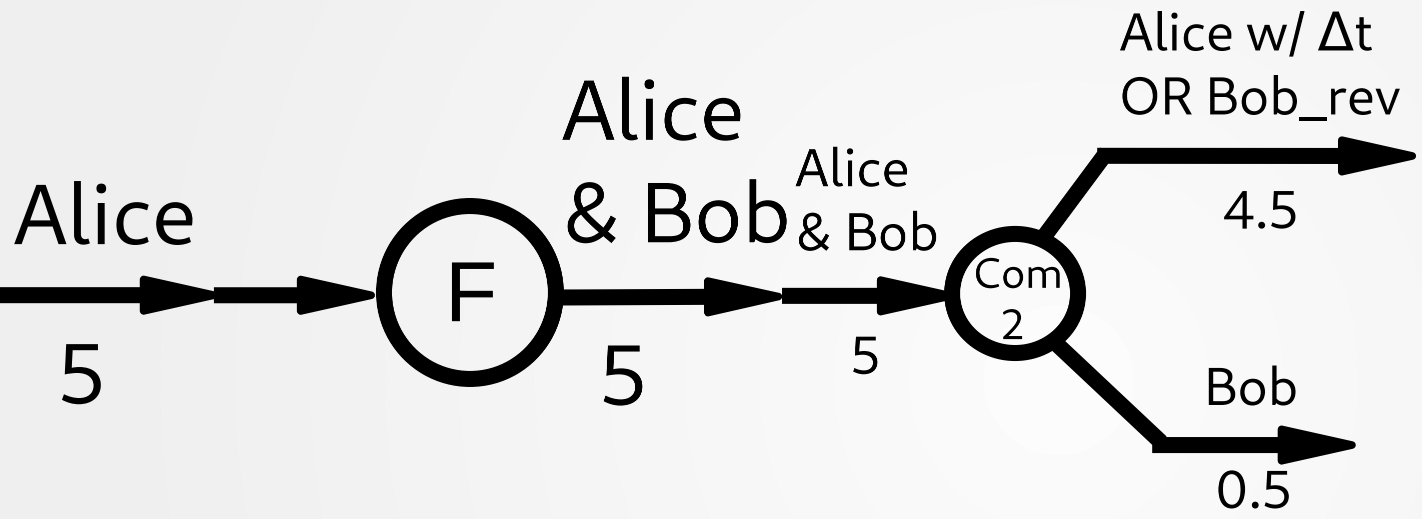

Alice
Alice
& Bob
F
5
5

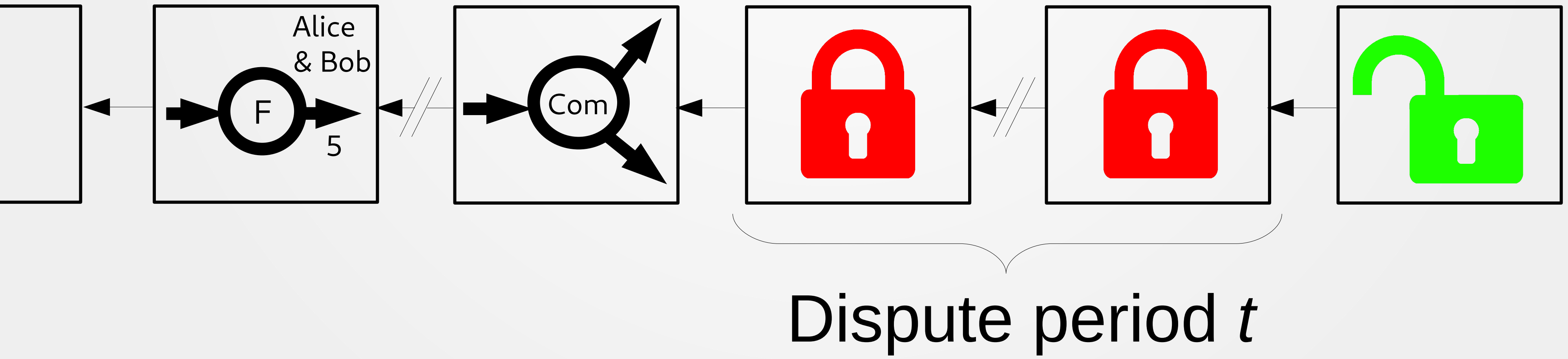

Alice w/ Δt
OR Bob_rev
Alice
& Bob
4
Com
1
5
Bob
1
$s_{B,0}$
1
Alice
Alice
& Bob
4
Com
1
5
Bob w/ Δt
OR Alice_rev
1
$s_{A,0}$

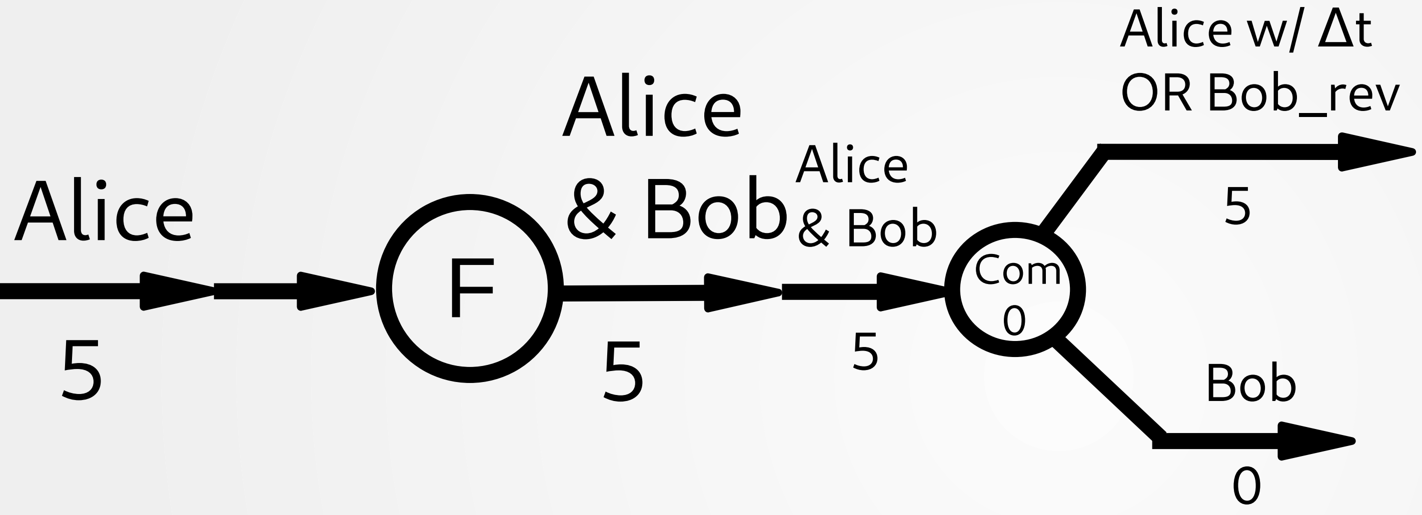
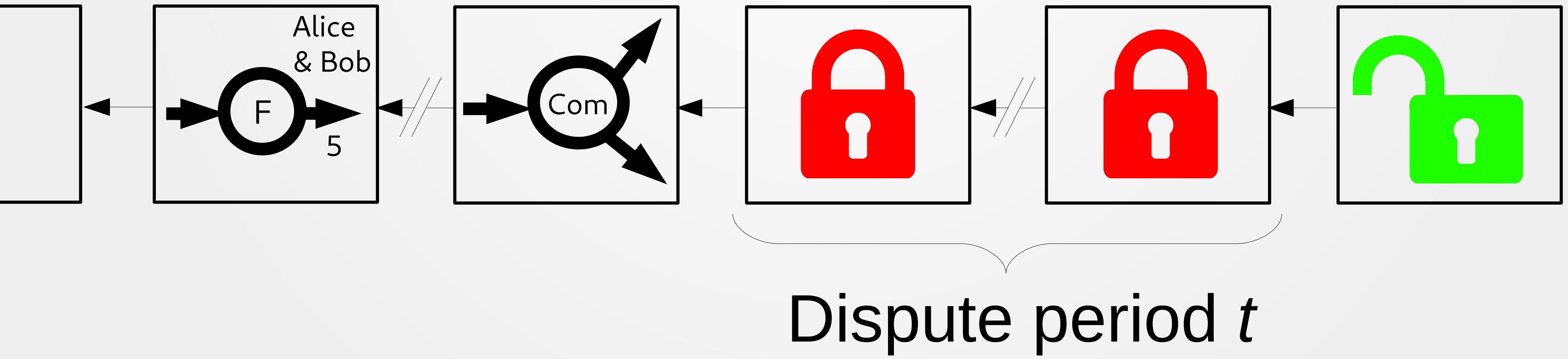

Alice
5
F
Alice
& Bob
5

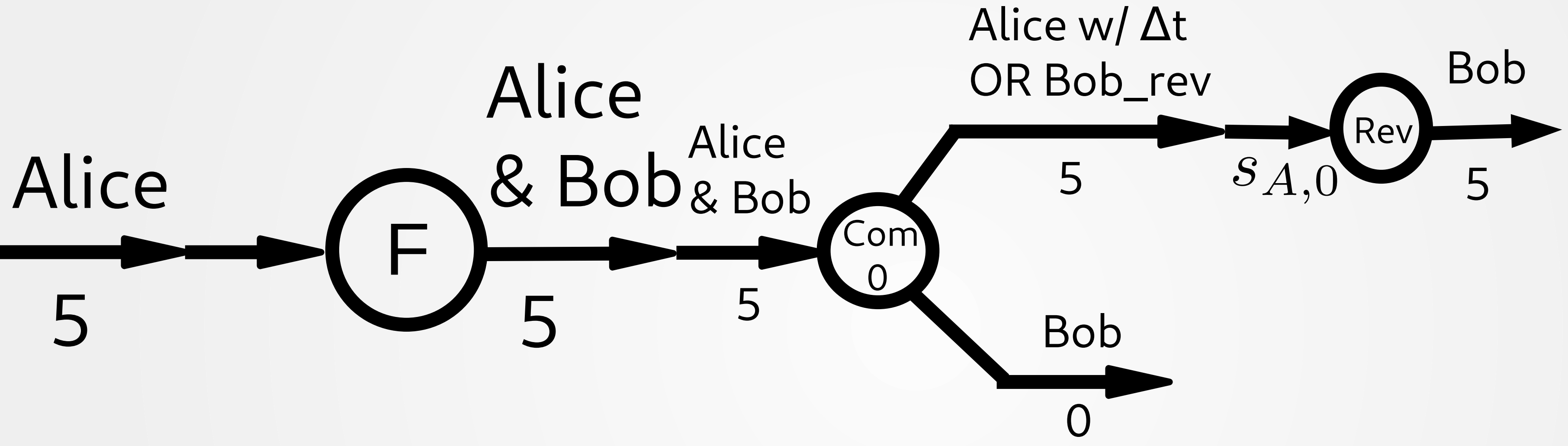

Alice w/ Δt
OR Bob_rev
Alice
& Bob
4.5
Com
2
5
Bob
0.5
$s_{B,0}$
$s_{B,1}$
0.5
Alice
& Bob
Com
2
5
Alice
4.5
Bob w/ Δt
OR Alice_rev
0.5
$s_{A,0}$
$s_{A,1}$

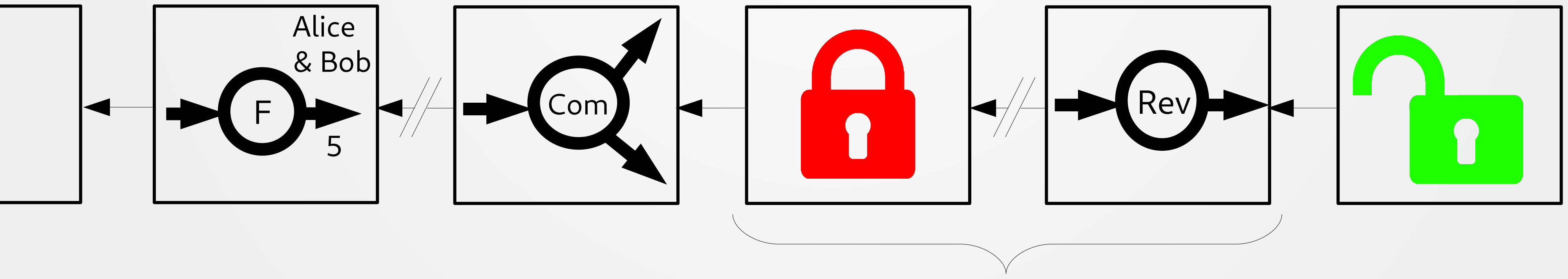

Alice
5
F
Alice
& Bob
5


Alice
5
F
Alice
& Bob
5
Alice
& Bob
5
Com
2
Alice w/ Δt
OR Bob_rev
4.5
Bob
0.5


Alice
& Bob
F
5
Com
Dispute period *t*

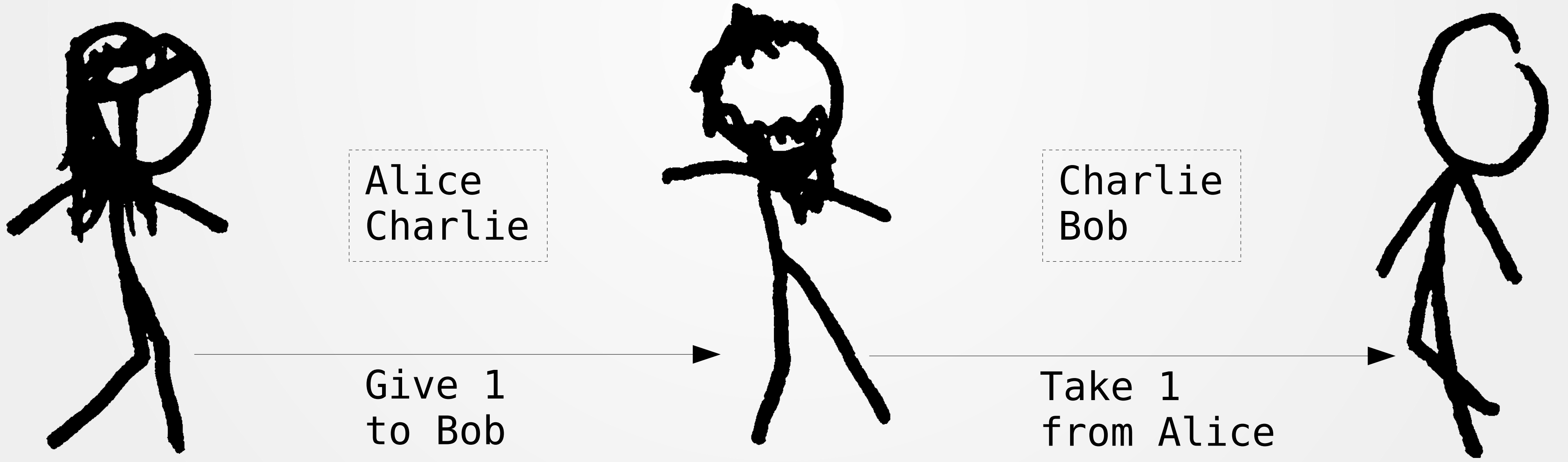

Alice
5
F
Alice & Bob
5
Alice & Bob
5
Com
0
Alice w/ Δt OR Bob_rev
5
Bob
0

Alice & Bob
F
5
Com
Dispute period *t*





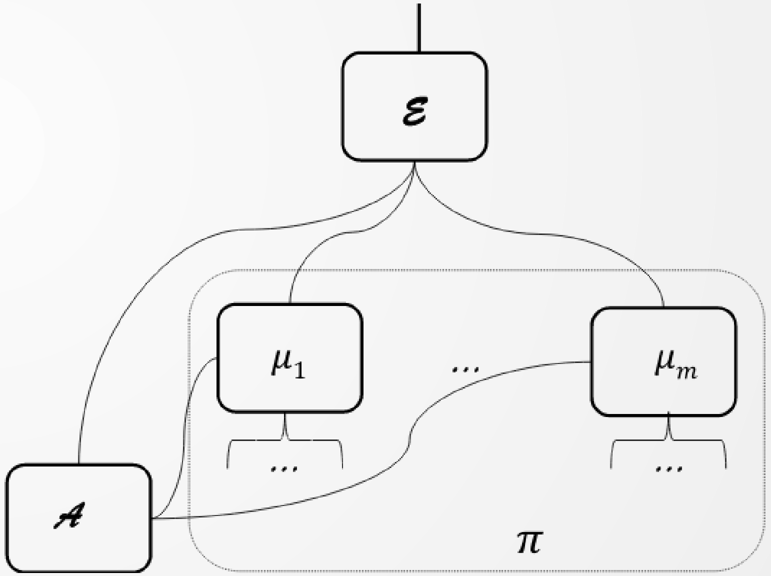
Dispute period *t*

Multi-hop payments

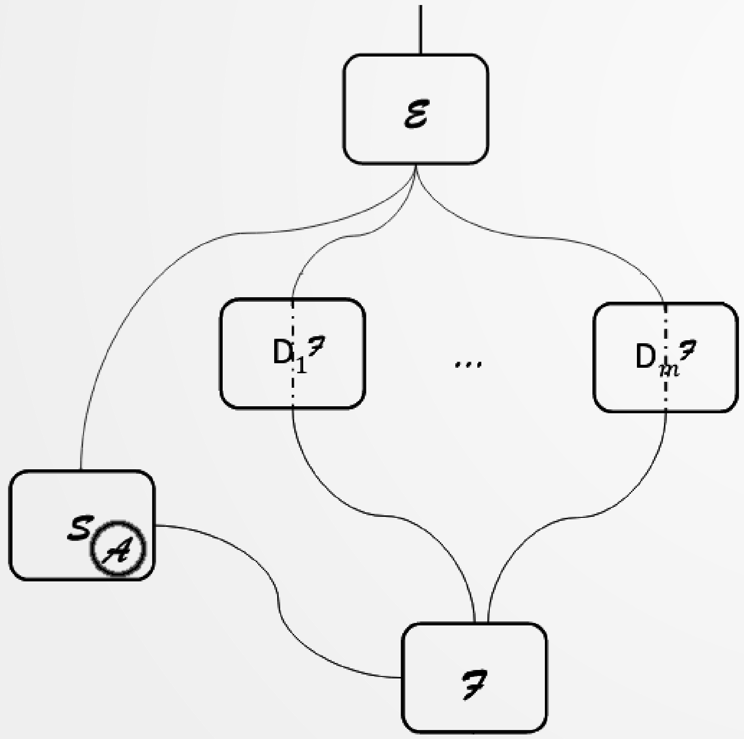


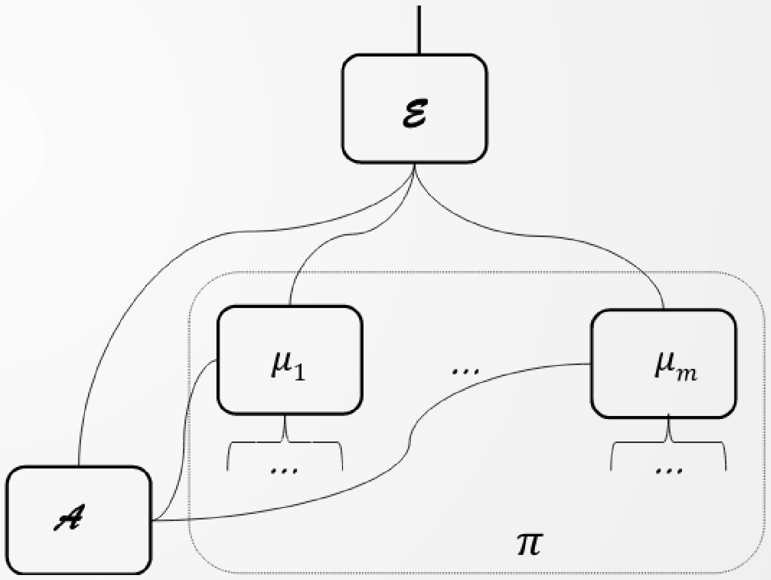
Part 2
Our contribution

Simulation-based Security



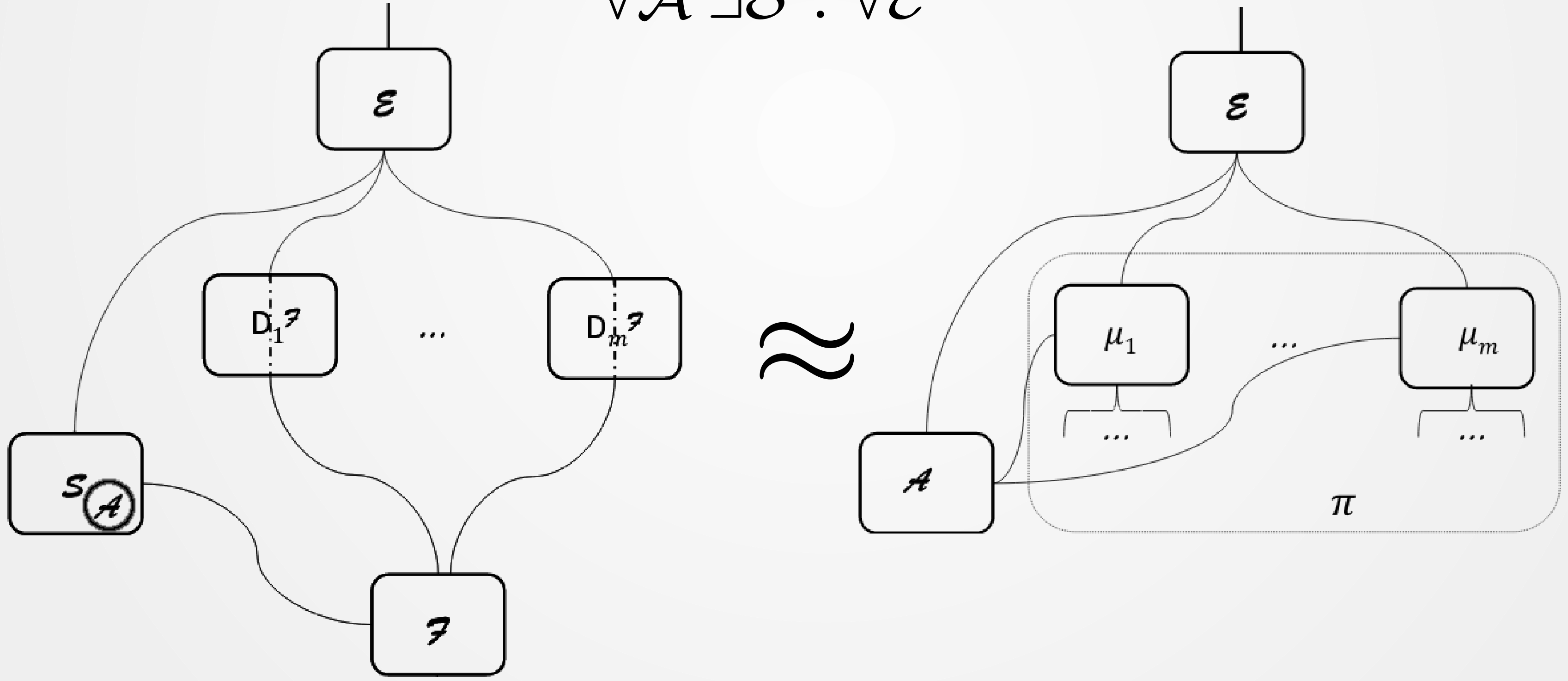
Simulation-based Security





Simulation-based Security

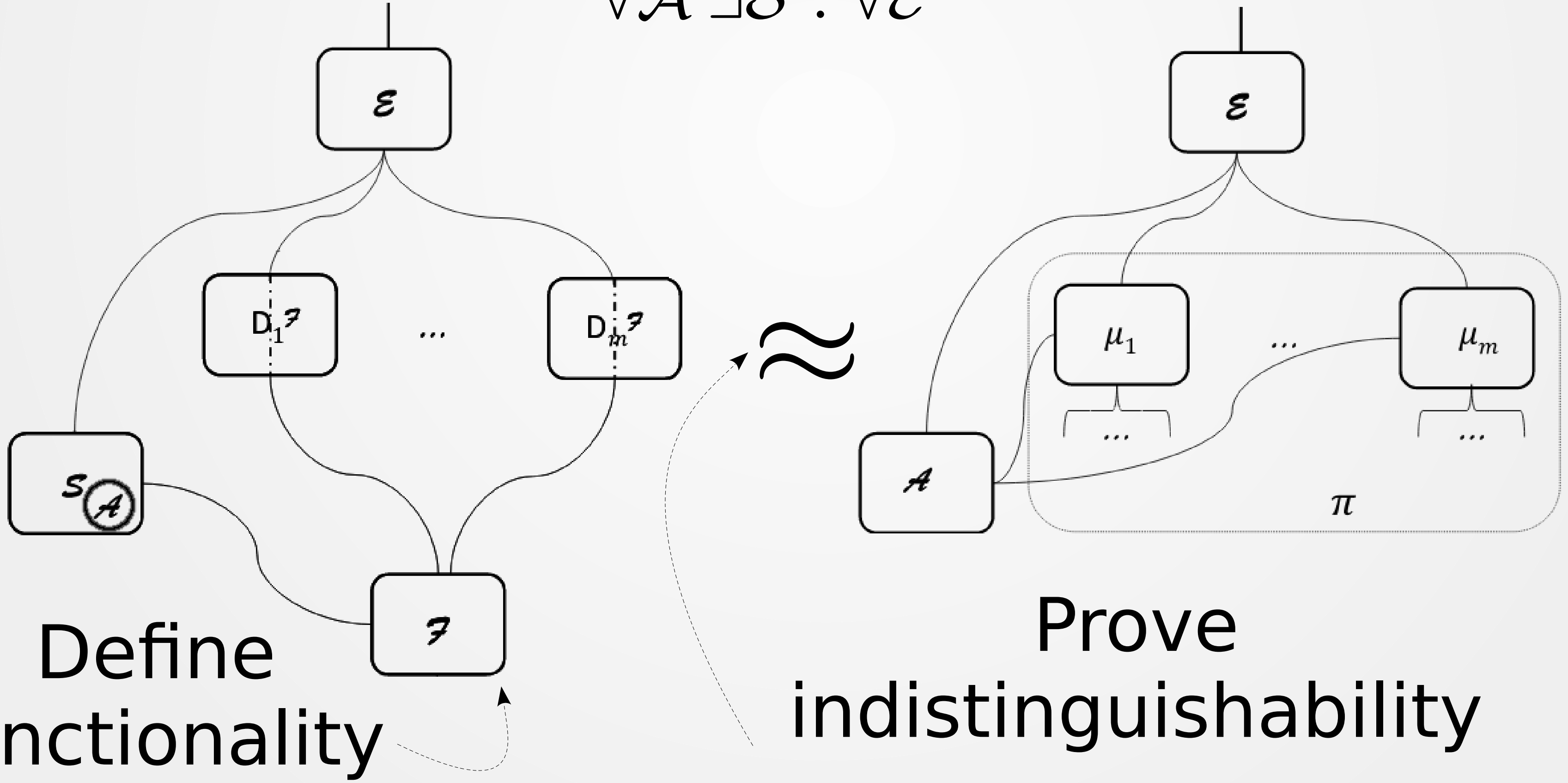
$$\forall \mathcal{A}\ \exists \mathcal{S} : \forall \mathcal{E}$$



Credits: “Universally Composable Security”, Ran Canetti
https://eprint.iacr.org/2000/067

Our paper

$$\forall \mathcal{A}\ \exists \mathcal{S} : \forall \mathcal{E}$$



Define functionality

Prove indistinguishability

Universal Composition

If functionality F is UC-realized by protocol P, then a protocol R that internally uses F is indistinguishable from R' that is like R but internally uses P

Universal Composition

$\mathcal{G}_{\mathrm{ledger}}$ [BMTZ’17, BGKRZ’18]

Functionality $\mathcal{F}_{\mathrm{PayNet}}$ – interface

- from $\mathcal{E}$:
  - (REGISTER, delay, relayDelay)
  - (TOPPEDUP)
  - (OPENCHANNEL, $Alice$, Bob, x, tid)
  - (CHECKFORNEW, $Alice$, Bob, tid)
  - (PAY, Bob, x, $\overrightarrow{\texttt{path}}$, `receipt`)
  - (CLOSECHANNEL, `receipt`, $pchid$)
  - (FORCECLOSECHANNEL, `receipt`, $pchid$)
  - (POLL)
  - (PUSHFULFILL, $pchid$)
  - (PUSHADD, $pchid$)
  - (COMMIT, $pchid$)
  - (FULFILLONCHAIN)
  - (GETNEWS)
- to $\mathcal{E}$:
  - (REGISTER, $Alice$, `delay`($Alice$), `relayDelay`($Alice$), pubKey)
  - (REGISTERED)
  - (NEWS, newChannels, closedChannels, updatesToReport)
- from $\mathcal{S}$:
  - (REGISTERDONE, $Alice$, pubKey)
  - (CORRUPTED, $Alice$)
  - (CHANNELANNOUNCED, $Alice$, $p_{Alice,F}$, $p_{Bob,F}$, $fchid$, $pchid$, tid)
  - (UPDATE, `receipt`, $Alice$)
  - (CLOSEDCHANNEL, `channel`, $Alice$)
  - (RESOLVEPAYS, $payid$, `charged`)
- to $\mathcal{S}$:
  - (REGISTER, $Alice$, delay, relayDelay)
  - (OPENCHANNEL, $Alice$, Bob, x, $fchid$, tid)
  - (CHANNELOPENED, $Alice$, $fchid$)
  - (PAY, $Alice$, Bob, x, $\overrightarrow{\texttt{path}}$, `receipt`, $payid$)
  - (CONTINUE)
  - (CLOSECHANNEL, $fchid$, $Alice$)
  - (FORCECLOSECHANNEL, $fchid$, $Alice$)
  - (POLL, Σ_{Alice}, $Alice$)
  - (PUSHFULFILL, $pchid$, $Alice$)
  - (PUSHADD, $pchid$, $Alice$)
  - (COMMIT, $pchid$, $Alice$)
  - (FULFILLONCHAIN, t, $Alice$)

Our contributions

- Use a realistic ledger functionality
- Prove Lightning Network security in UC framework
- Derive exact time bounds for how often parties need to check the chain

Our contributions

- Use a realistic ledger functionality
- Prove Lightning Network security in UC framework
- Derive exact time bounds for how often parties need to check the chain

Thank you!