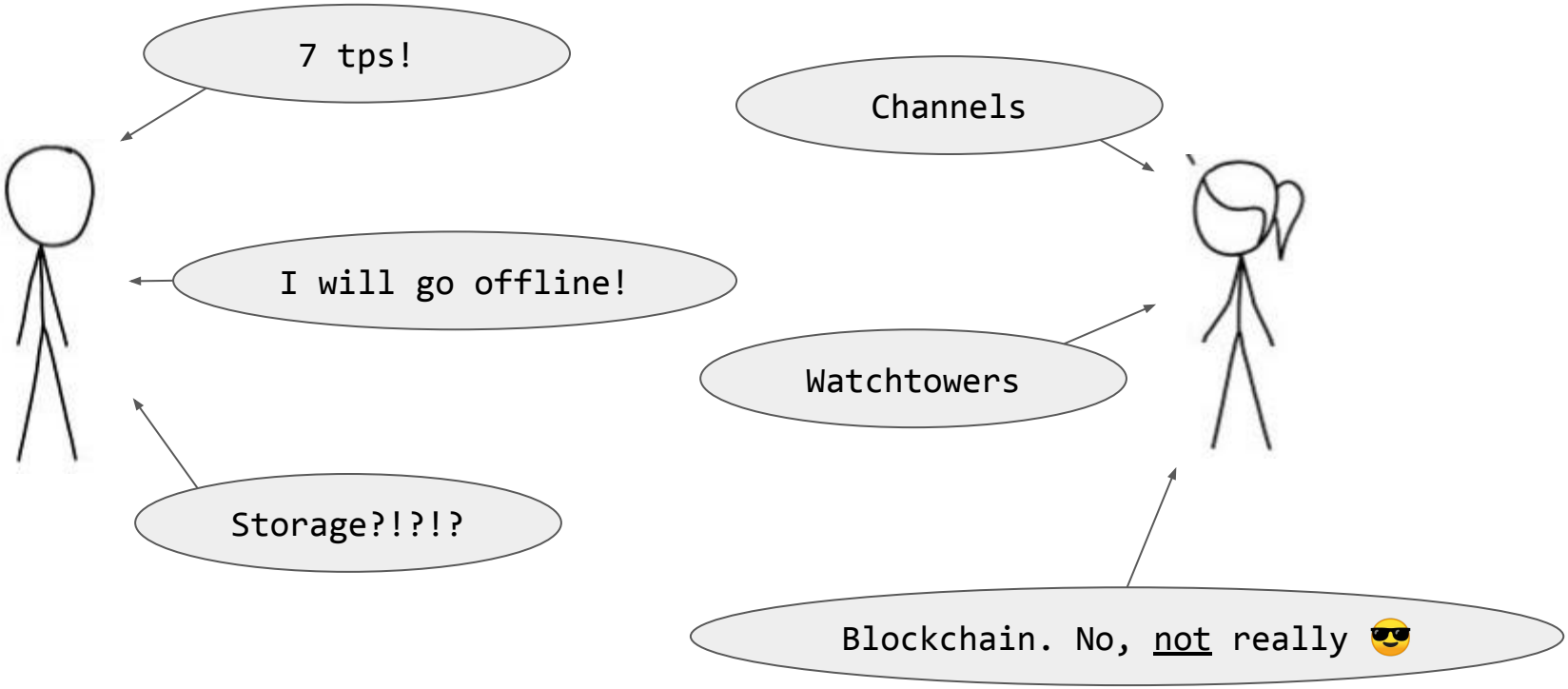
Outpost: A Lightweight Responsive Watchtower

eth = Eidgenössische Technische Hochschule
(Federal Institute of Technology)

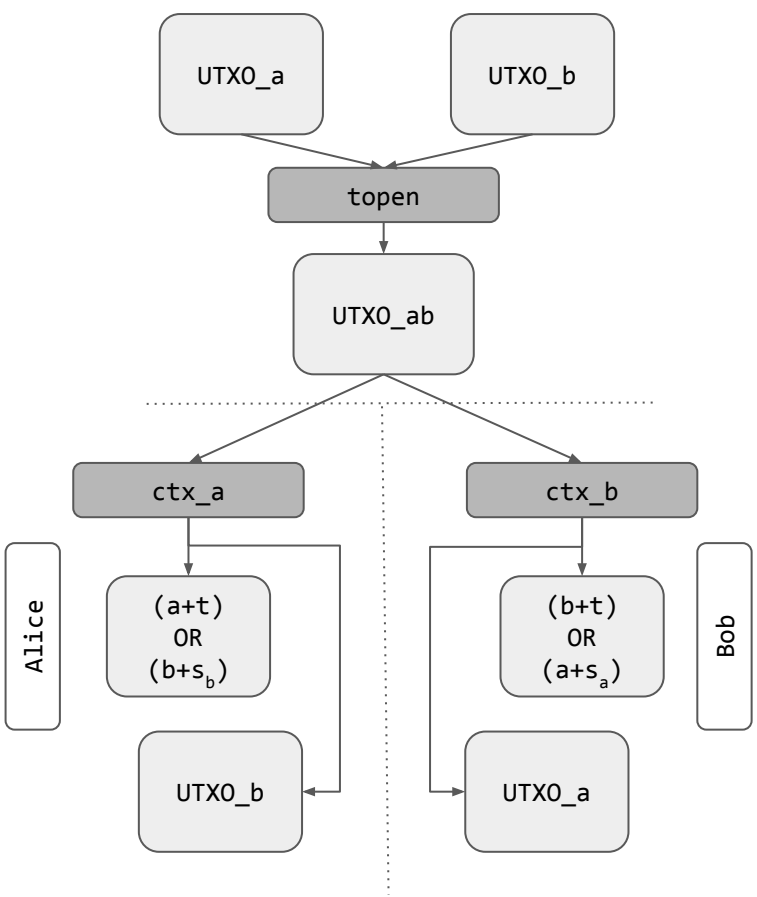
Tejaswi Nadahalli (*ETH Zürich*)
Majid Khabazzian (*Univ. of Alberta*)
Roger Wattenhofer (*ETH Zürich*)

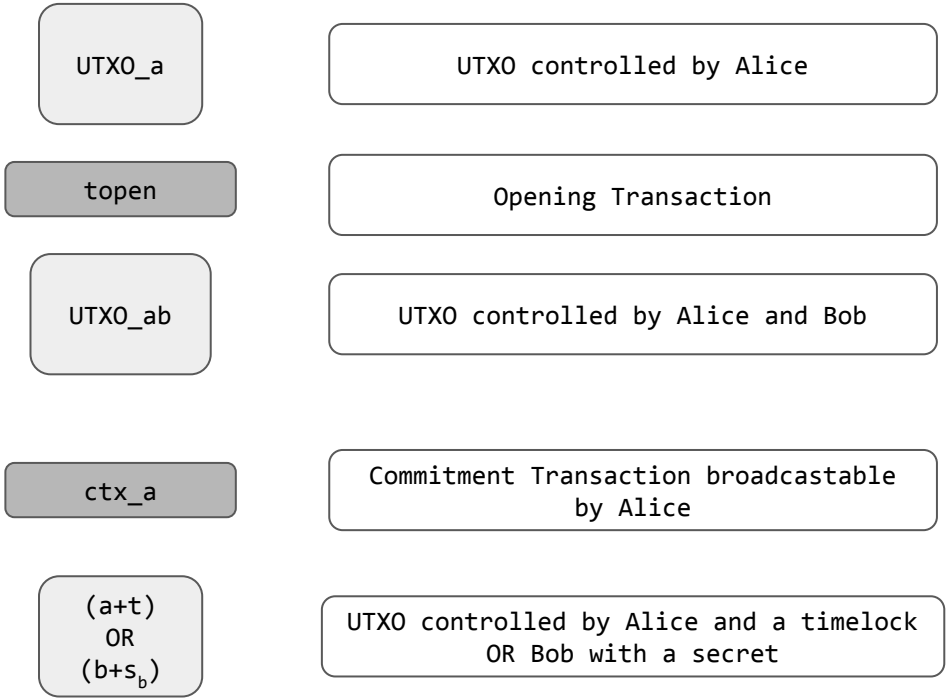
What’s happening?



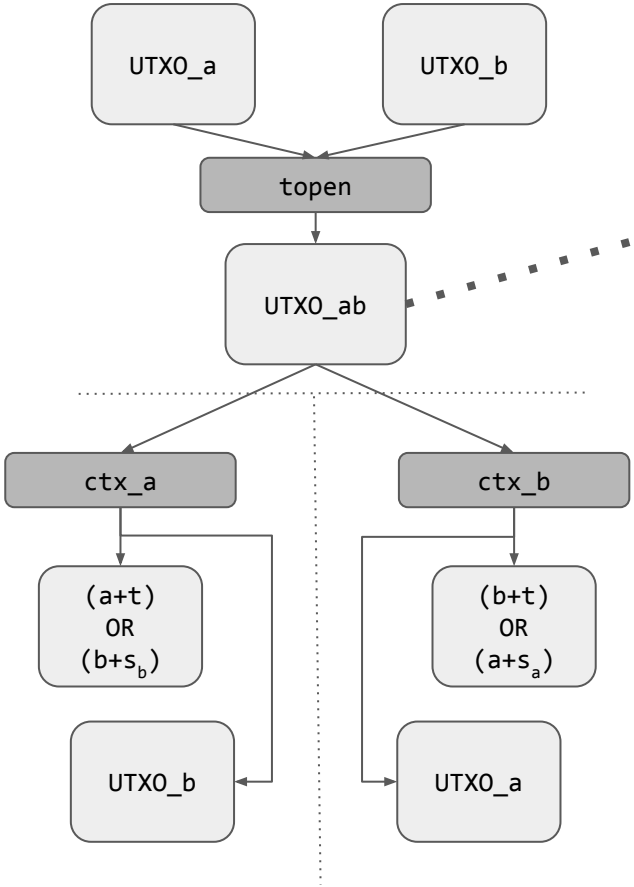
stick figures: XKCD

Lightning Channel

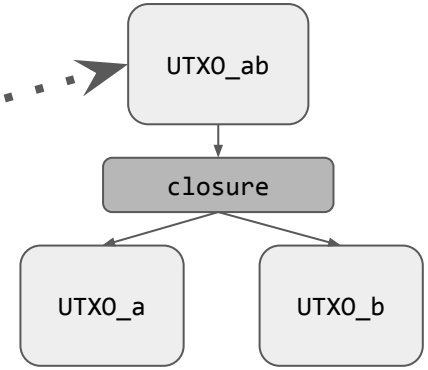




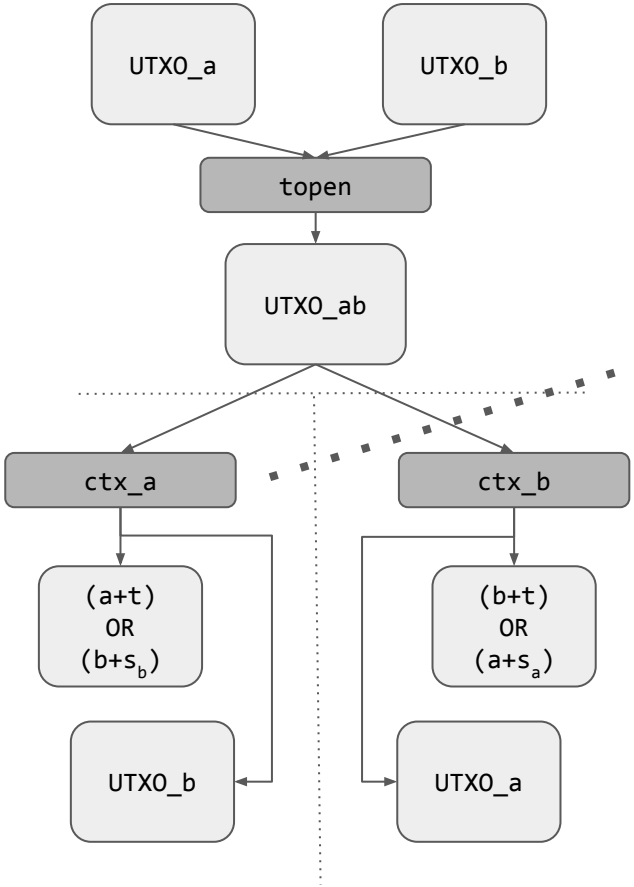
Lightning Channel



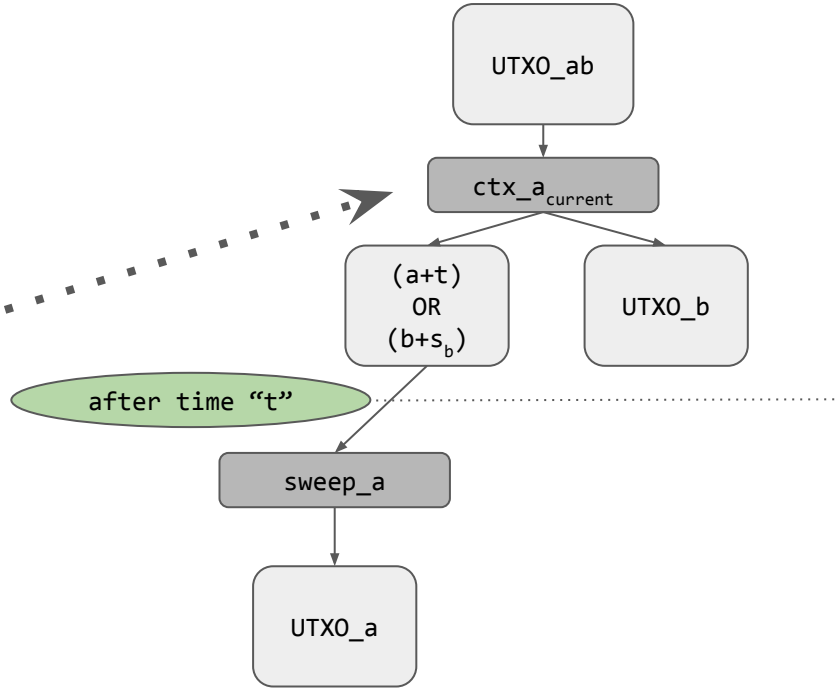
Bilateral Closure



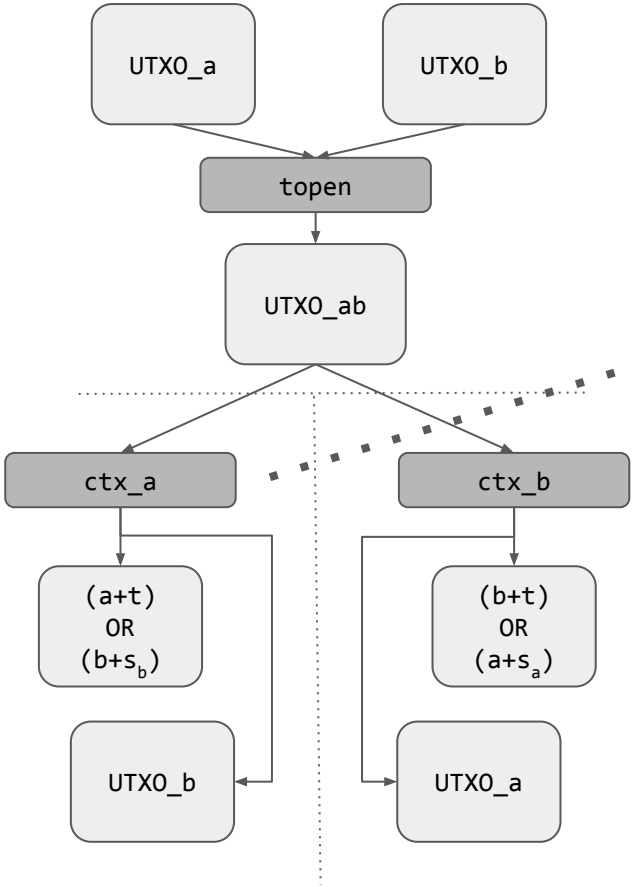
Lightning Channel



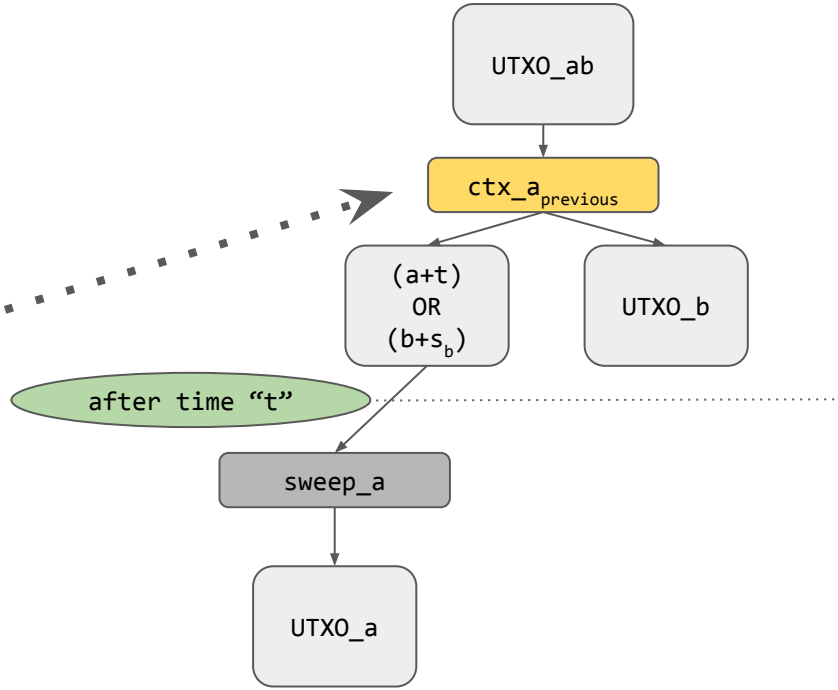
Unilateral Closure



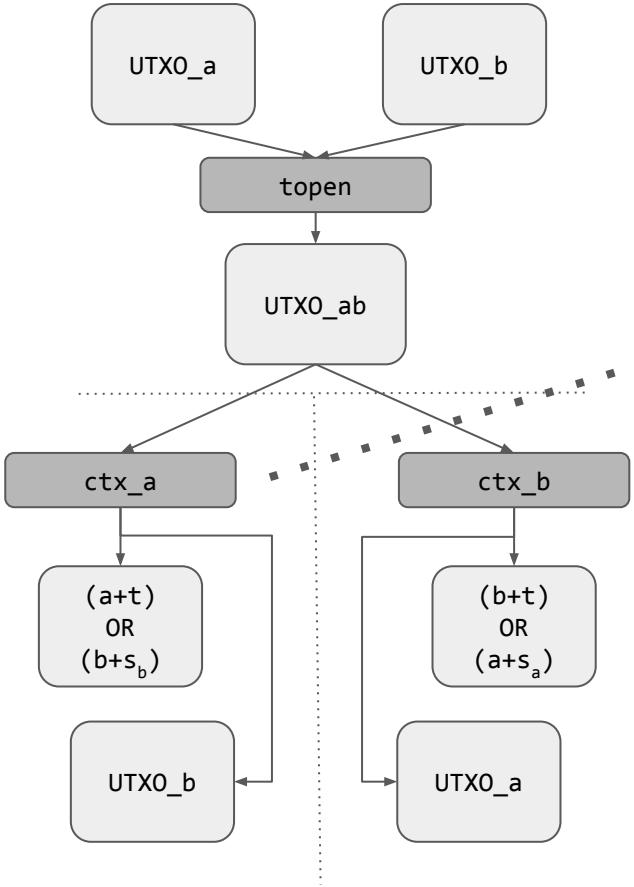
Lightning Channel



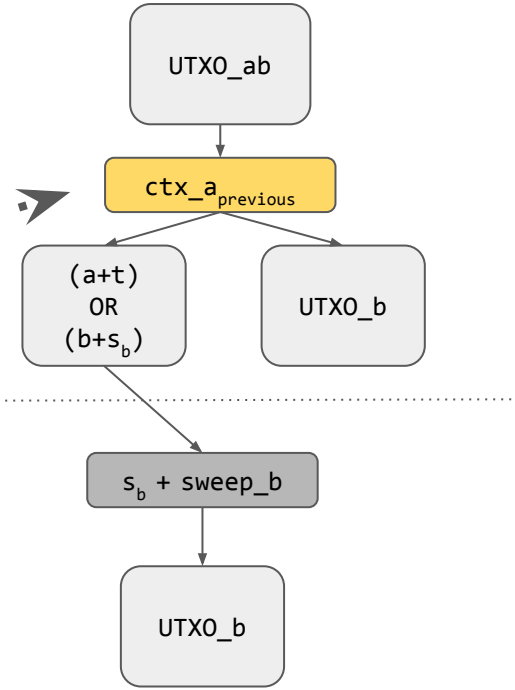
Cheating Closure



Lightning Channel



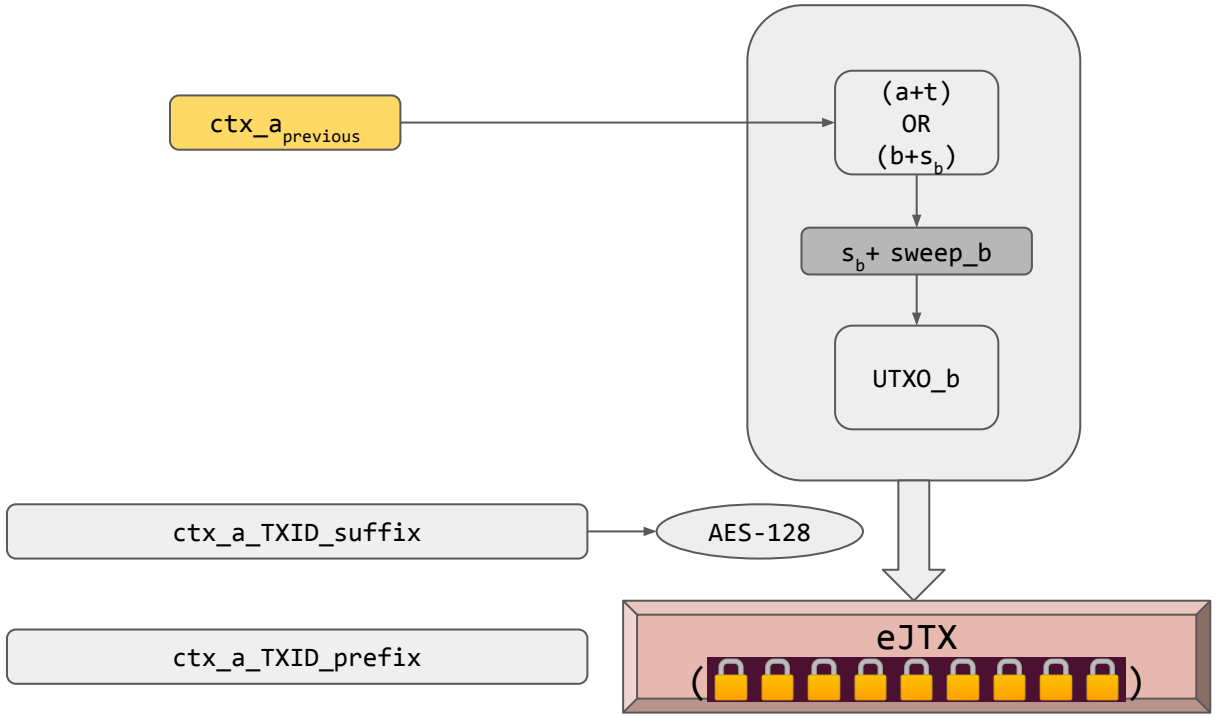
Justice Transaction



Offline...

Watchtower

Justice Kit


ctx_a previous
(a+t)
OR
(b+s_b)
s_b+ sweep_b
UTXO_b
ctx_a_TXID_suffix
AES-128
ctx_a_TXID_prefix
eJTX
(
)

Watchtower

e3b0c44298...	
6e340b9cff...	
96a296d224...	
709e80c884...	
df3f619804...	
8855508aad...	
...	...
...	...

Observations

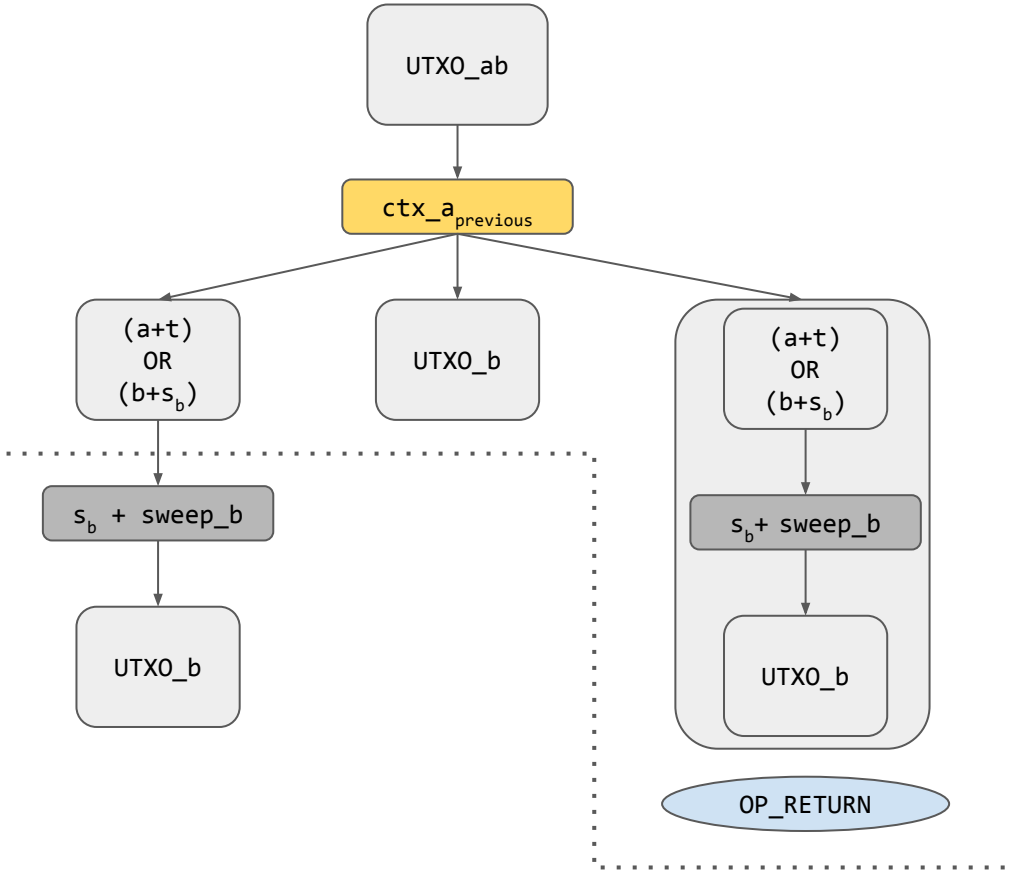
Cheater has to store cheating CTX(s)

Every CTX has a corresponding JTX

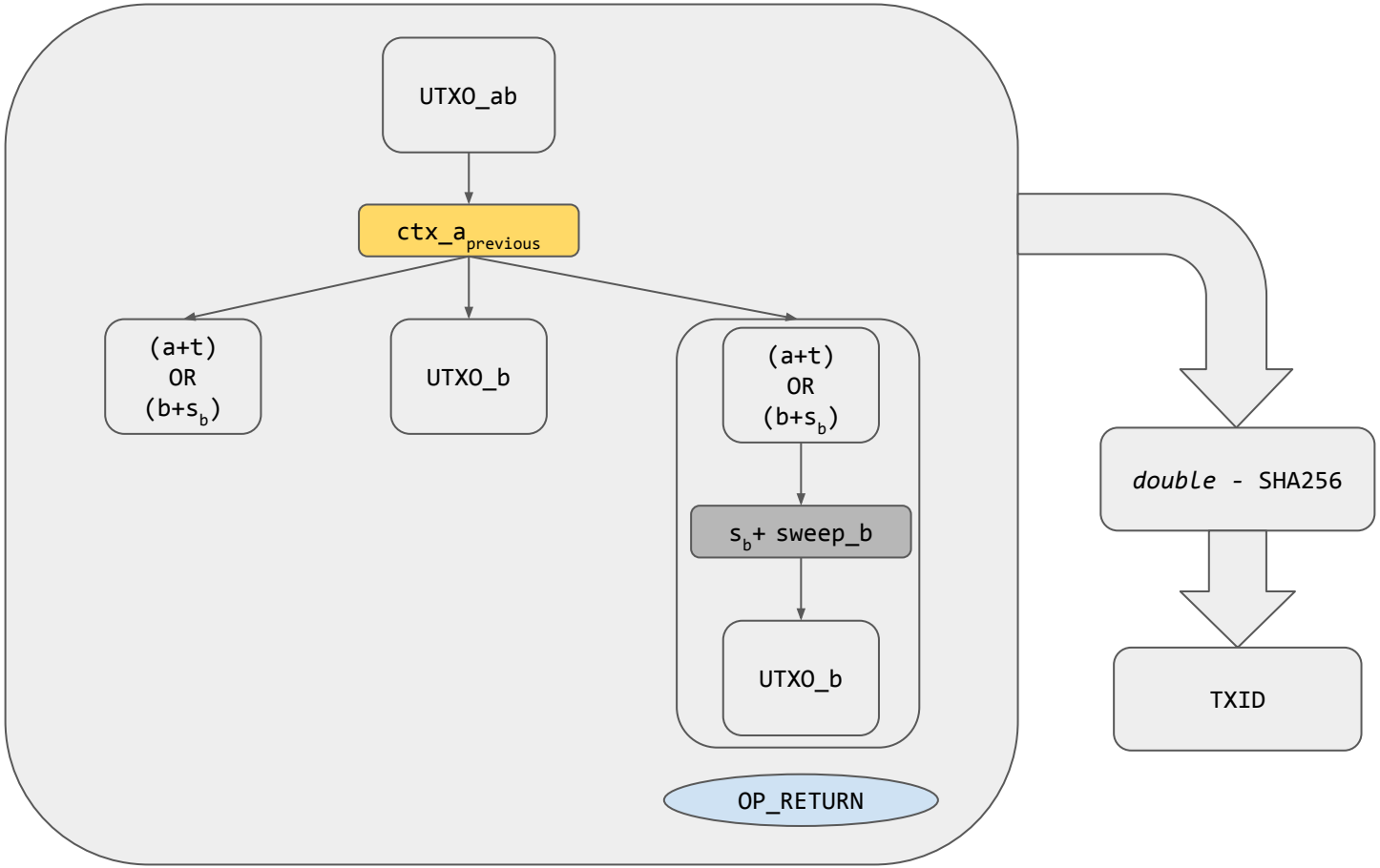
CTX has to be published on the blockchain

Store the corresponding JTX inside this CTX?

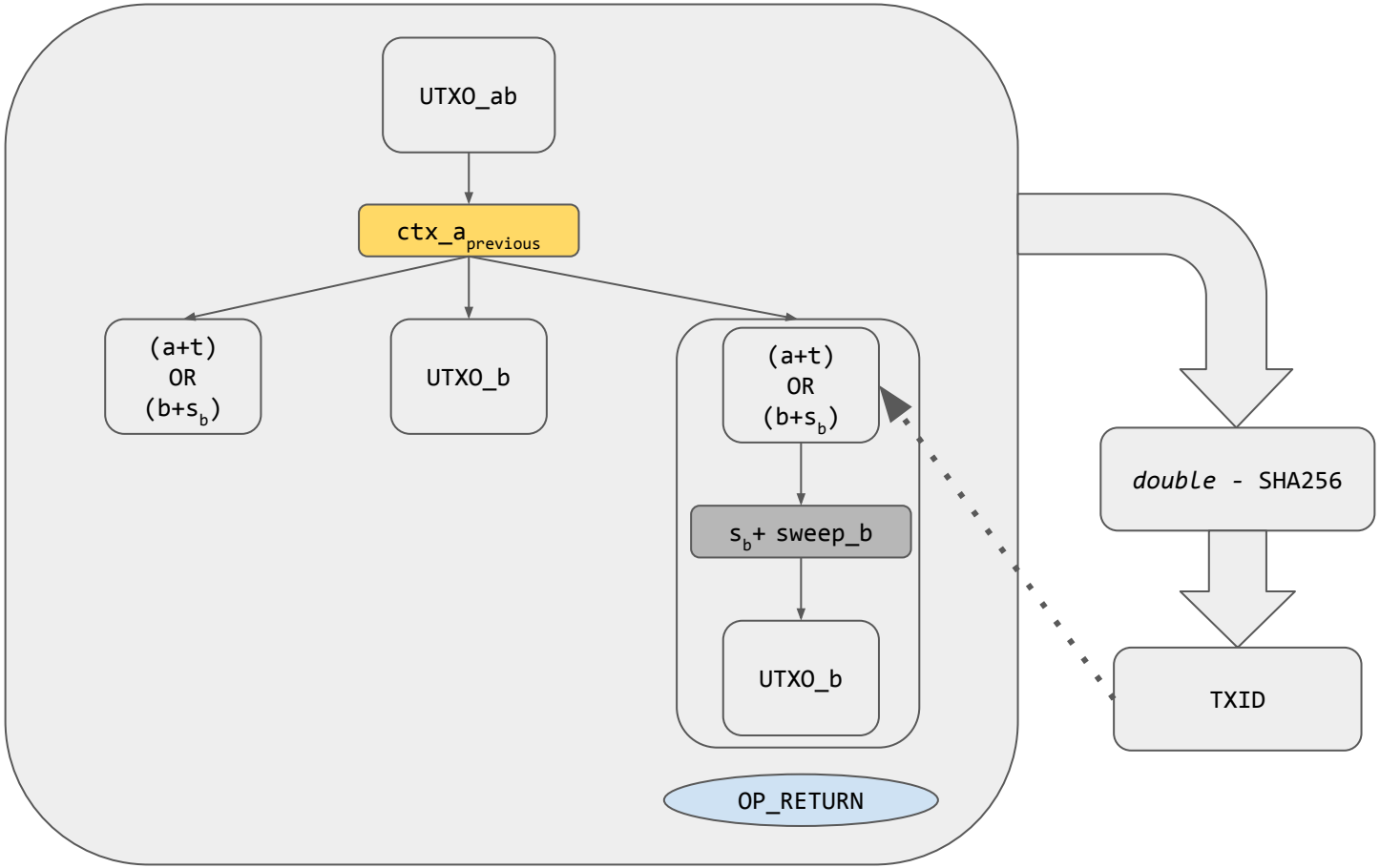
Can we store JTX inside CTX?



TXID


UTXO_ab
ctx_a previous
(a+t)
OR
(b+s_b)
UTXO_b
(a+t)
OR
(b+s_b)
s_b+ sweep_b
UTXO_b
OP_RETURN
double - SHA256
TXID

TXID makes it self-referential



OP_RETURN alternatives

P2SH Data Drop

- vulnerable to scriptSig malleability

P2SH Data Hash

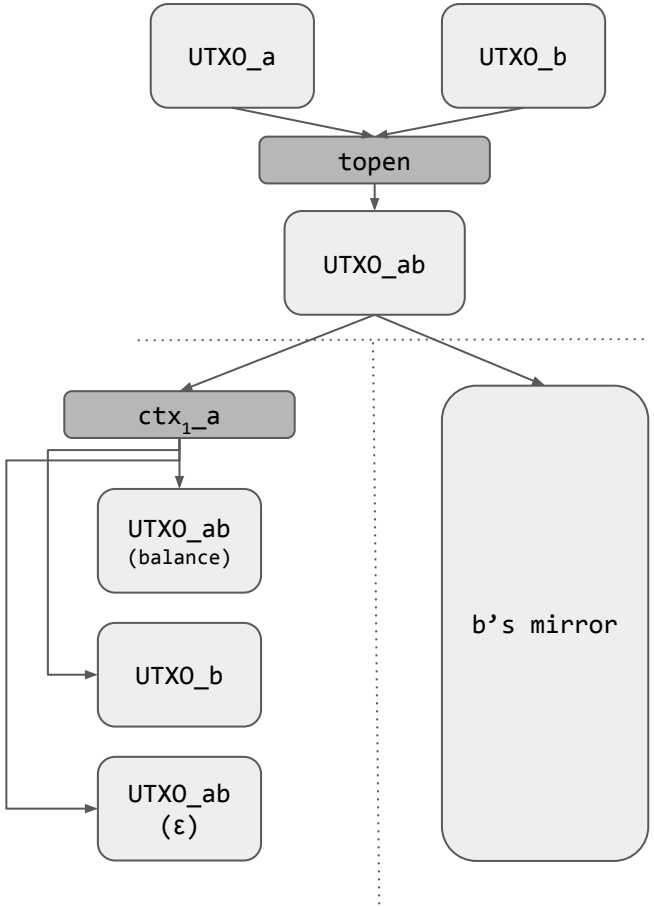
- vulnerable to self-loop in the redeem_script hash

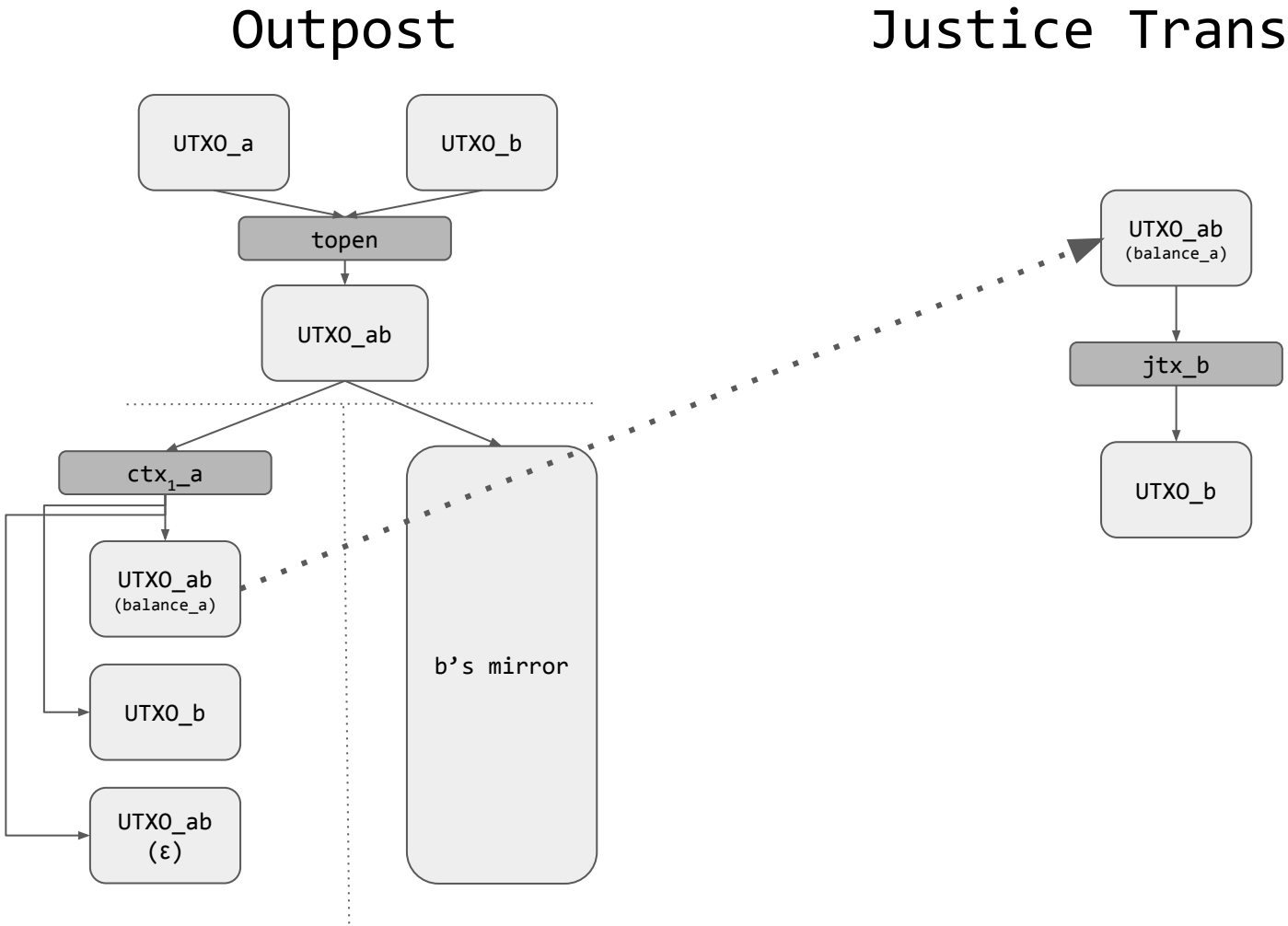
SIGHASH_NOINPUT

- If Eltoo, why Lightning?

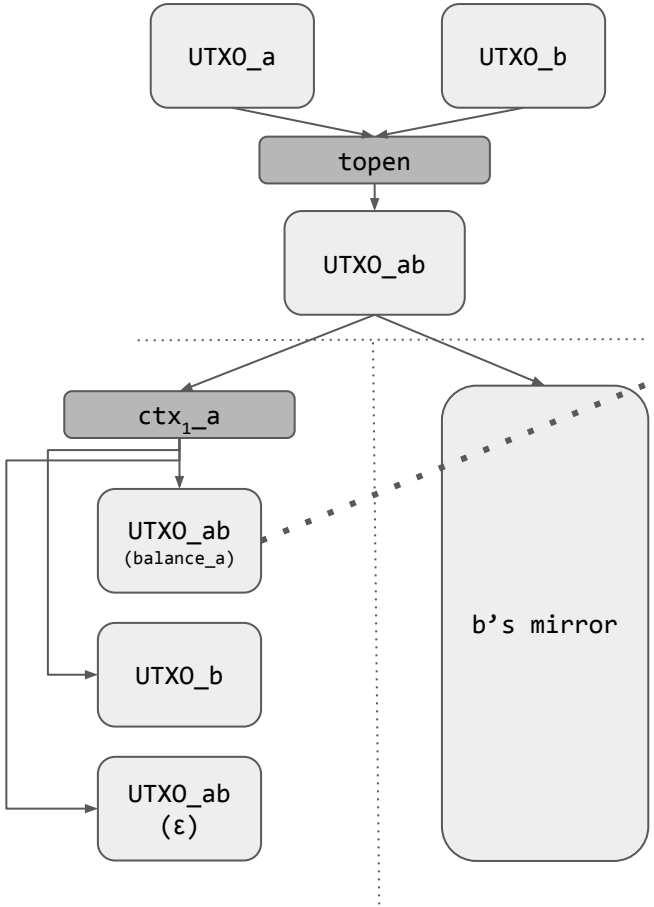
Outpost

Outpost

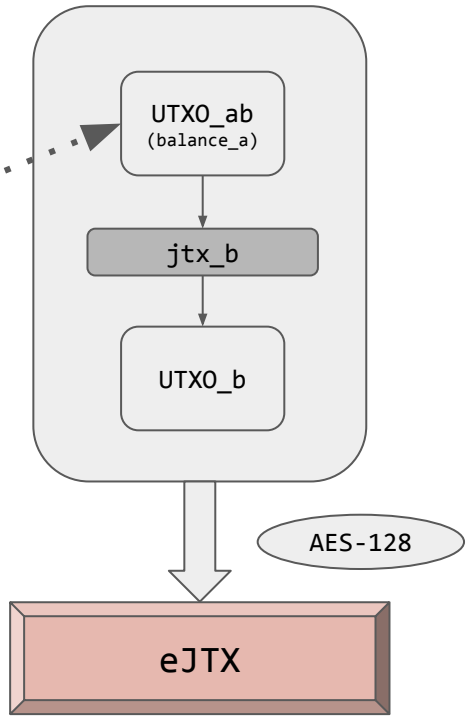

UTXO_a
UTXO_b
topen
UTXO_ab
ctx1_a
UTXO_ab
(balance)
UTXO_b
UTXO_ab
(ε)
b's mirror

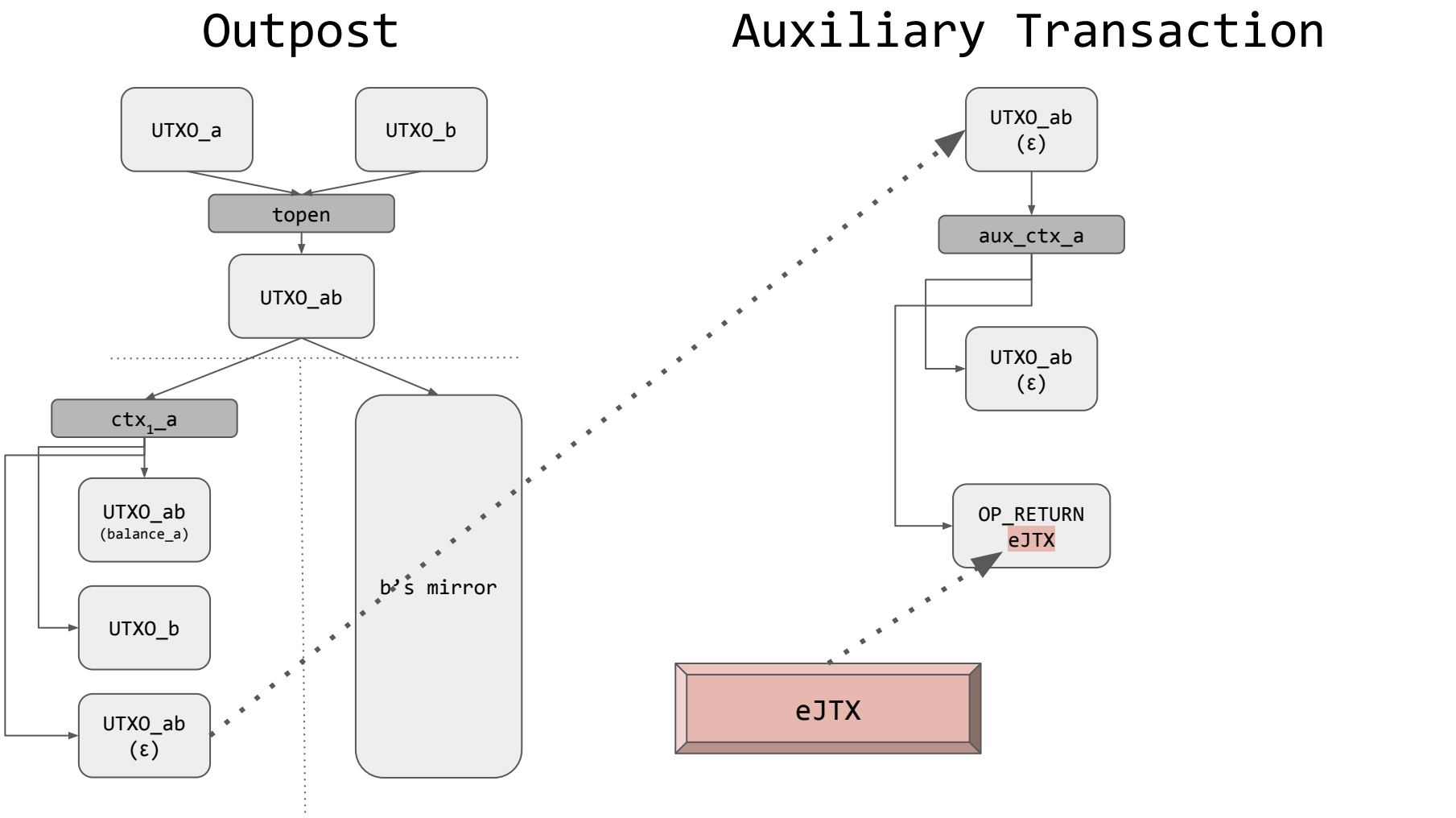

Outpost
Justice Transaction
UTXO_a
UTXO_b
topen
UTXO_ab
ctx1_a
UTXO_ab
(balance_a)
UTXO_b
UTXO_ab
(ε)
b's mirror
UTXO_ab
(balance_a)
jtx_b
UTXO_b

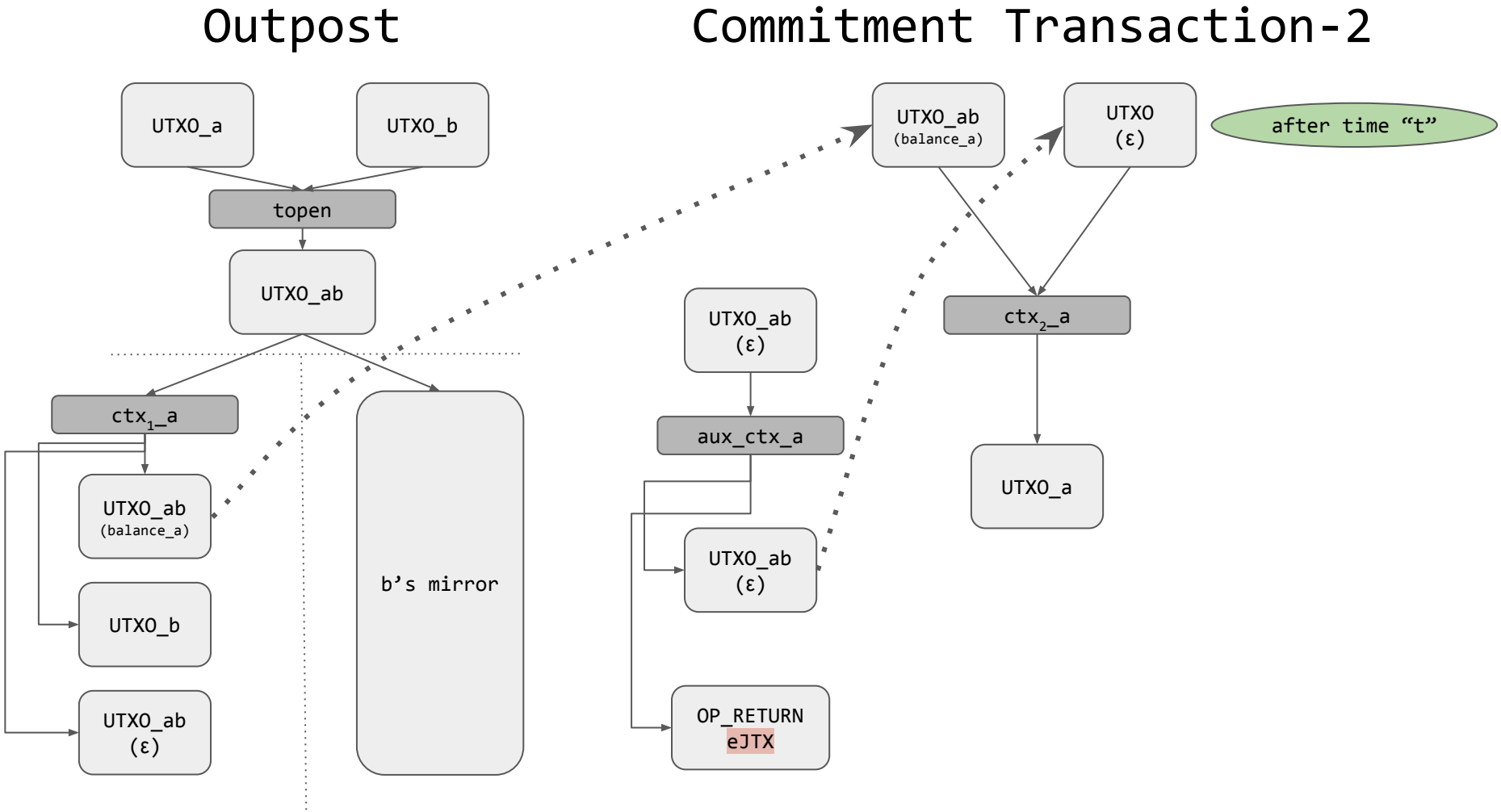
Outpost



Encrypted Justice Transaction




Outpost
Auxiliary Transaction
UTXO_a
UTXO_b
topen
UTXO_ab
ctx1_a
UTXO_ab
(balance_a)
UTXO_b
UTXO_ab
(ε)
b's mirror
UTXO_ab
(ε)
aux_ctx_a
UTXO_ab
(ε)
OP_RETURN
eJTX
eJTX


Outpost
Commitment Transaction-2
UTXO_a
UTXO_b
topen
UTXO_ab
ctx1_a
UTXO_ab
(balance_a)
UTXO_b
UTXO_ab
(ε)
b's mirror
UTXO_ab
(balance_a)
UTXO
(ε)
after time "t"
UTXO_ab
(ε)
ctx2_a
aux_ctx_a
UTXO_a
UTXO_ab
(ε)
OP_RETURN
eJTX

The money slide

Classic Lightning

	Per channel, with N updates	20k channels, 1M updates
Known Channel	N·size(ejtx) + 1·size(txid)	7.00 TB
Unknown Channel	N·size(ejtx) + N·size(txid))	7.65 TB

Outpost

Known Channel	size(key) + size(txid)	0.96 **MB** (WTF)
Unknown Channel	N·size(key) + N·size(txid)	0.96 TB

Note: size(key) << size(ejtx) i.e. 16 << 350

Outpost keeps Lightning’s key features

- Unilateral closure: broadcaster has to wait
    - Not cheating
    - Cheating

- Exchange revocation keys vs. AES-128 decryption keys

Cheating (Alice wants to profit)

Time

Alice broadcasts older ctx_1_a, aux_ctx_a	Bob is watching the blockchain
Wait	Bob sees aux_ctx_a
Wait	Bob has key to decrypt eJTX & get JTX_b
Wait	Bob broadcasts JTX_b
Wait	JTX_b is confirmed on the blockchain 😎
Wait	ctx_2_a is now invalid
Alice can broadcast ctx_2_a - but…	

Unilateral Closure(Bob has disappeared)

Time

Alice broadcasts latest ctx_1_a, aux_ctx_a	Bob is secretly watching the blockchain
Wait	Bob sees aux_ctx_a
Wait	Bob cannot decrypt eJTX
Wait	ctx_2_a is still valid
Alice can broadcast ctx_2_a - 😎	

Griefing ¯_(ツ)_/¯

Alice broadcasts ctx_1_a	Bob is watching the blockchain
Wait	Bob sees ctx_1_a confirmed
Wait	Bob's own ctx_1_b is now invalid 😡
Wait	Bob's JTX_b is valid, if he has it
Gone...	

Can we have ctx_1_a send all its balance to Bob? Much later.

Profit

Outpost

- CTX and eJTX on the blockchain
- eJTX’s key in the node (16 bytes)

Classic Lightning

- CTX on the blockchain
- JTX in the node (~350 bytes)

Cost

Outpost

- 3 txns
- ε

Classic Lightning

- 1 txn
- No ε

Limitations

- OP_RETURN limited to 80 bytes.
    - IsStandard ಠ_ಠ
    - Split aux_ctx into 2; P2SH Data-hash across them

- Bloat
    - Not on the blockchain (happy case)
    - On the blockchain, 3 txns vs 1 txn

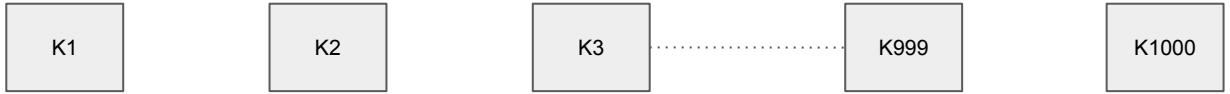
- But
    - No changes to Bitcoin, whatsover.

Optimize!!

SHACHAIN (Fast)

Pre-compute

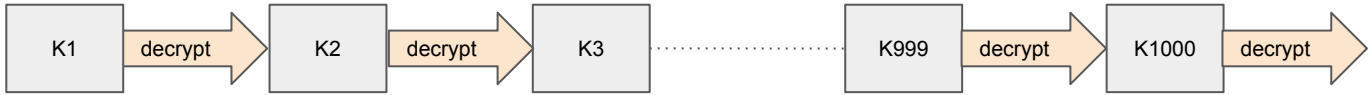
Use*



RSA-CHAIN (Slow)

Pre-compute No

Use*



*Key derivation function

Size estimates

Classic Lightning

	Per channel, with N updates	20k channels, 1M updates
Known Channel	N·size(ejtx) + 1·size(txid)	7.00 TB
Unknown Channel	N·size(ejtx) + N·size(txid))	7.65 TB

Outpost

Known Channel	size(key) + size(txid)	0.96 MB (WTF)
Unknown Channel	N·size(key) + N·size(txid)	0.96 TB

Note: size(key) << size(ejtx) i.e. 16 << 350

What’s next?

- HTLC
- Implementation

Thanks

- Store justice transactions inside commitment transactions

- tejaswin@ethz.ch
- mkhabbazian@ualberta.ca
- wattenhofer@ethz.ch