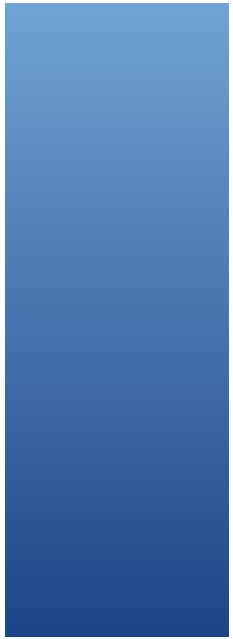

Transaction Relay Policy
for L2 Developers
@glozow

So you submitted the transaction to your mempool and you're waiting for it to confirm.

This Talk



01 Design Goals for Transaction Relay

02 Defining Policy

03 Why DoS Protection Isn’t the Only Concern

04 Known Policy Issues, Lightning Attacks

05 Let’s be friends?

We want a P2P transaction relay network in which

Anyone should be able to send a Bitcoin payment.

Minimal User Requirements

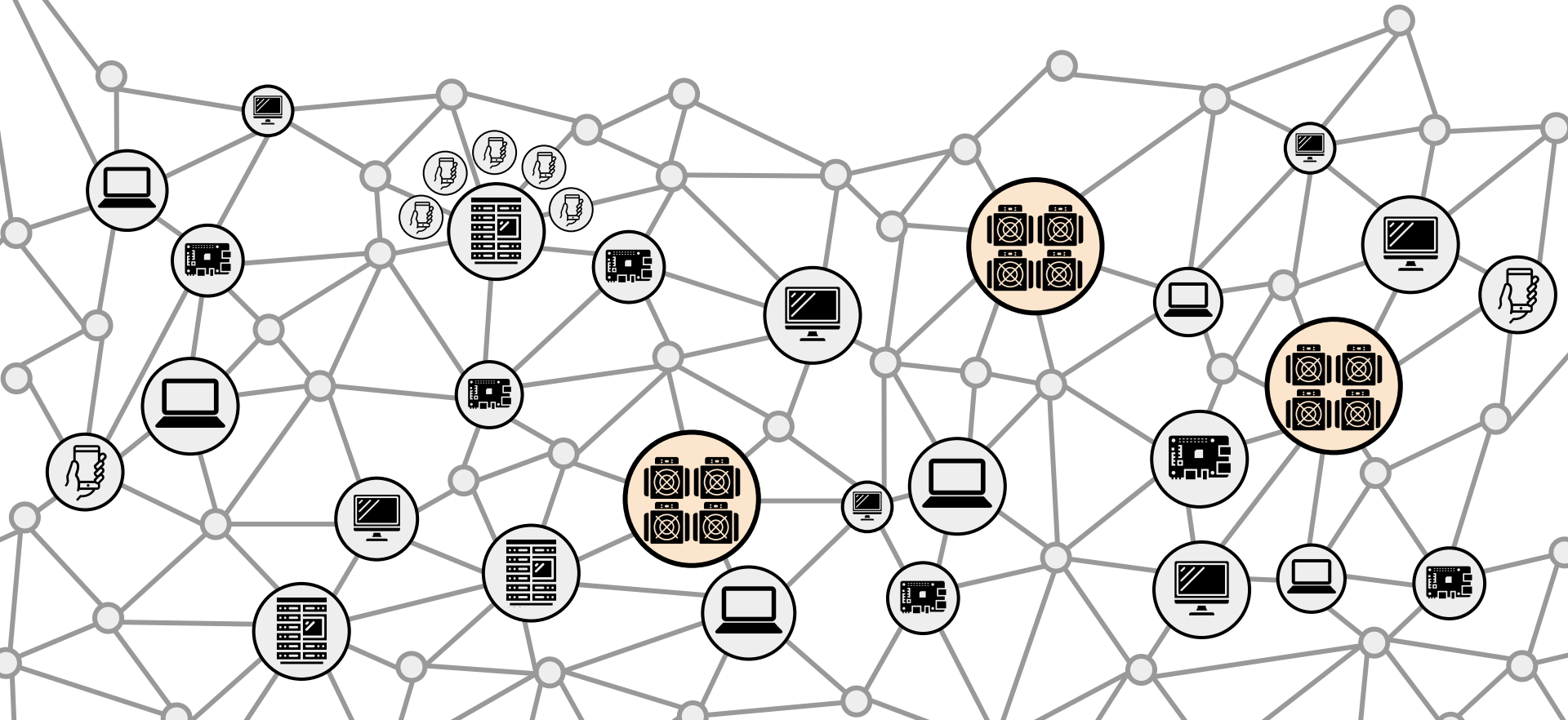
Censorship Resistance

Security Against DDoS Attacks

Incentive Compatibility

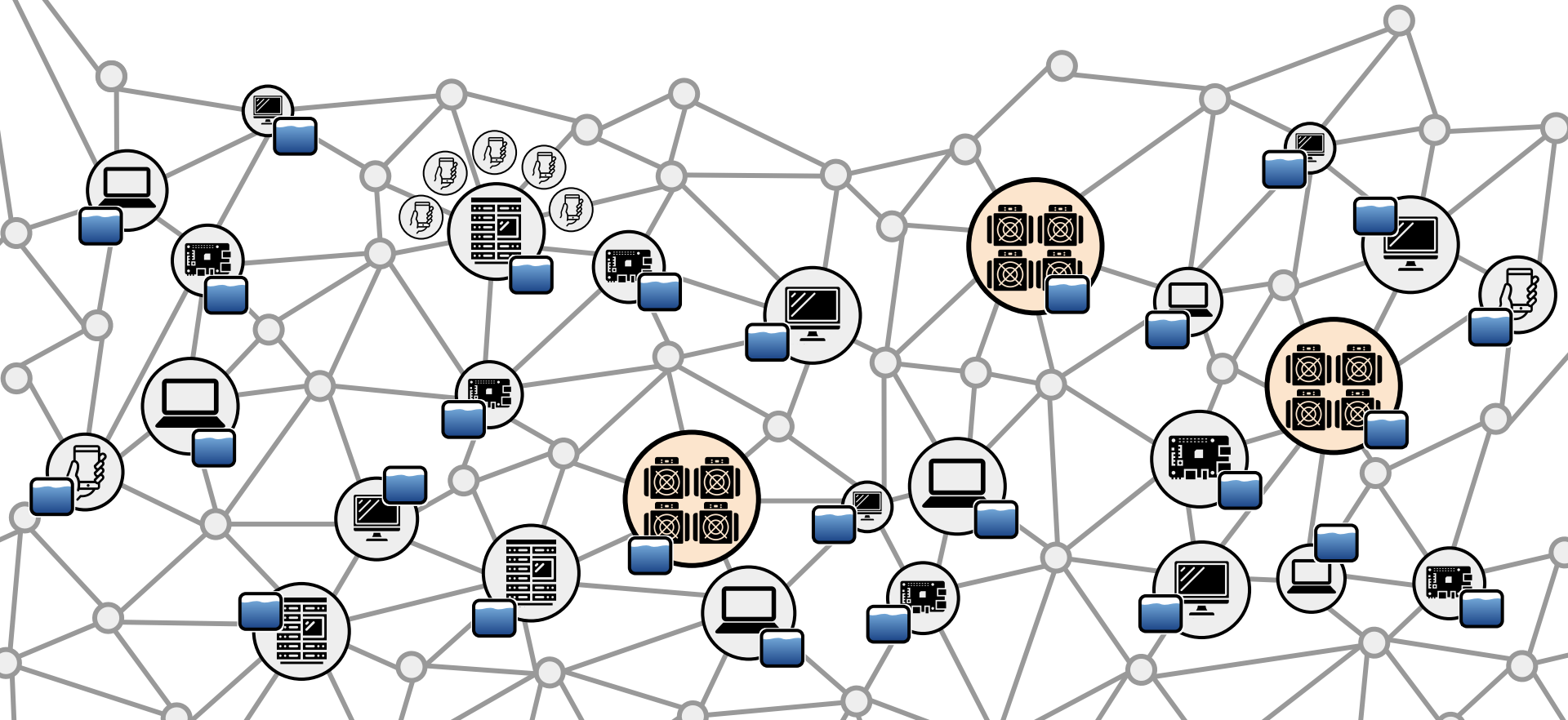
P2P Transaction Relay Network

Allows anyone to connect a computer to the Bitcoin network and propagate their transactions to miners

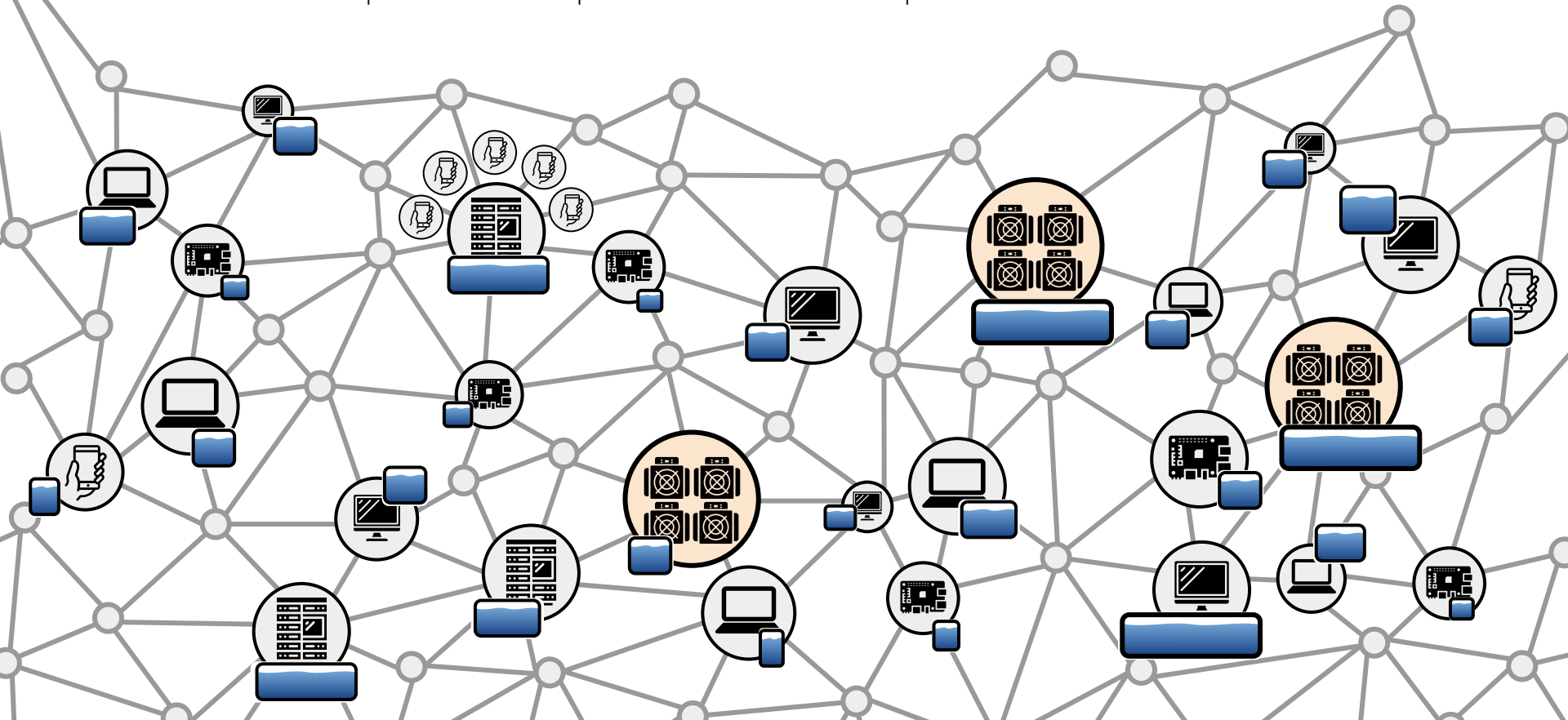


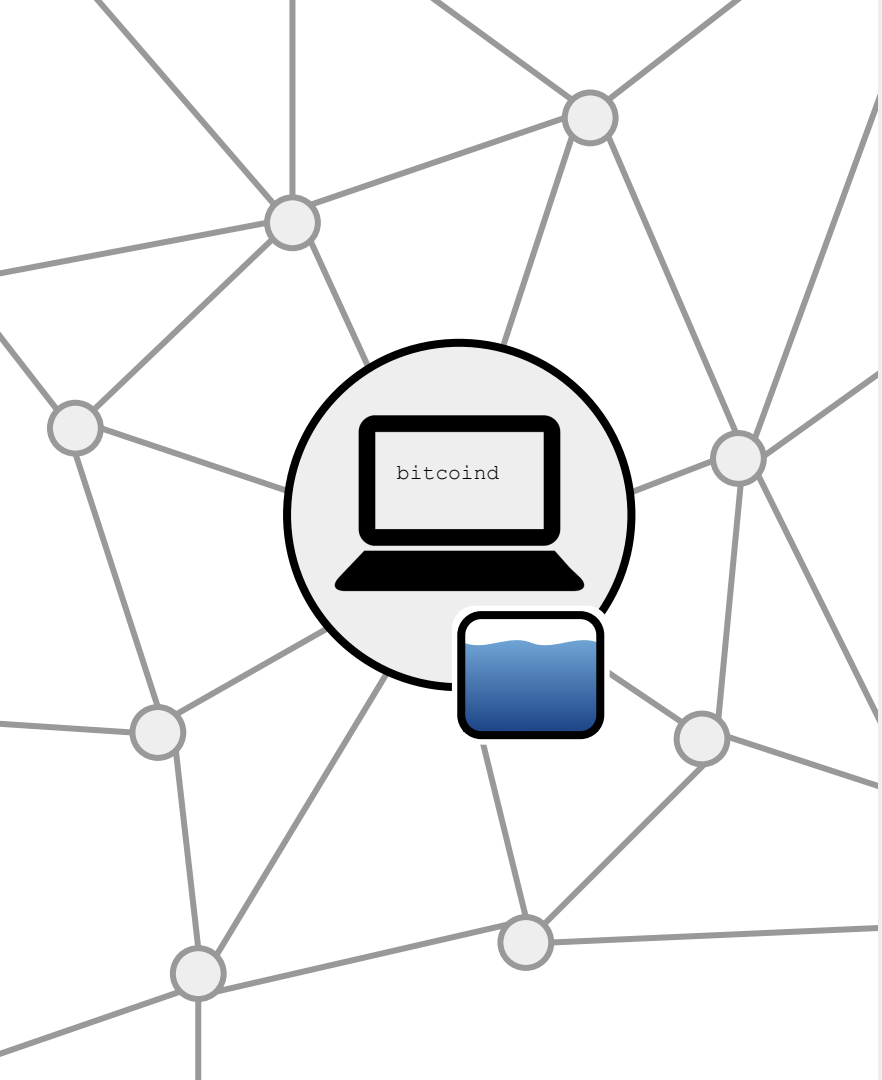
Mempools

Serve as caches of unconfirmed transactions



Mempool Policy: a node's set of validation rules, in addition to consensus, which all unconfirmed transactions must pass to be accepted to the node's mempool.





From a protocol development perspective,
we focus on protecting the bitcoind user.

Possible Peers:

- Honest Users
- DoSer trying to exhaust our CPU
- DoSer trying to cause OOM
- Attacker trying to fill mempools with garbage
- DDoSer trying to stall the network for 0.5sec
- Attacker trying to cause network splits
- Lightning counterparty trying to pin or censor the honest user's package
- Spy node trying to deanonymize transactions
- Spy node trying to analyze network topology

“Ideal” Mempool
(No Policy)

Always validate, accept all
consensus-valid transactions

“Ideal” Mempool (No Policy)

Always validate, accept all consensus-valid transactions

Perfectly Defensive Mempool Policy

Only validate transactions from trusted parties

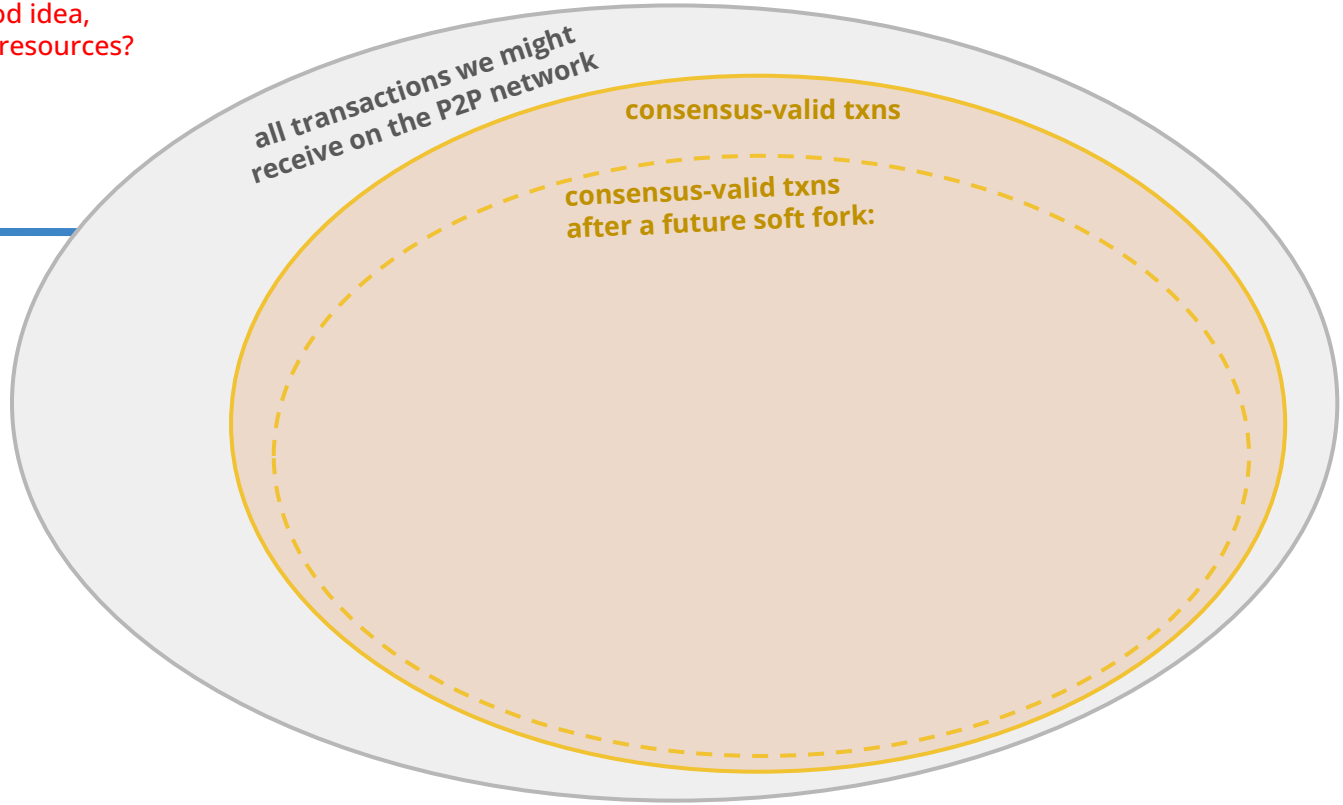
DoS protection isn’t the full story.


Would this be a good idea,
even if we had infinite resources?
“Ideal” Mempool
(No Policy)
Always validate, accept all
consensus-valid transactions

Would this be a good idea,
even if we had infinite resources?

“Ideal” Mempool
(No Policy)

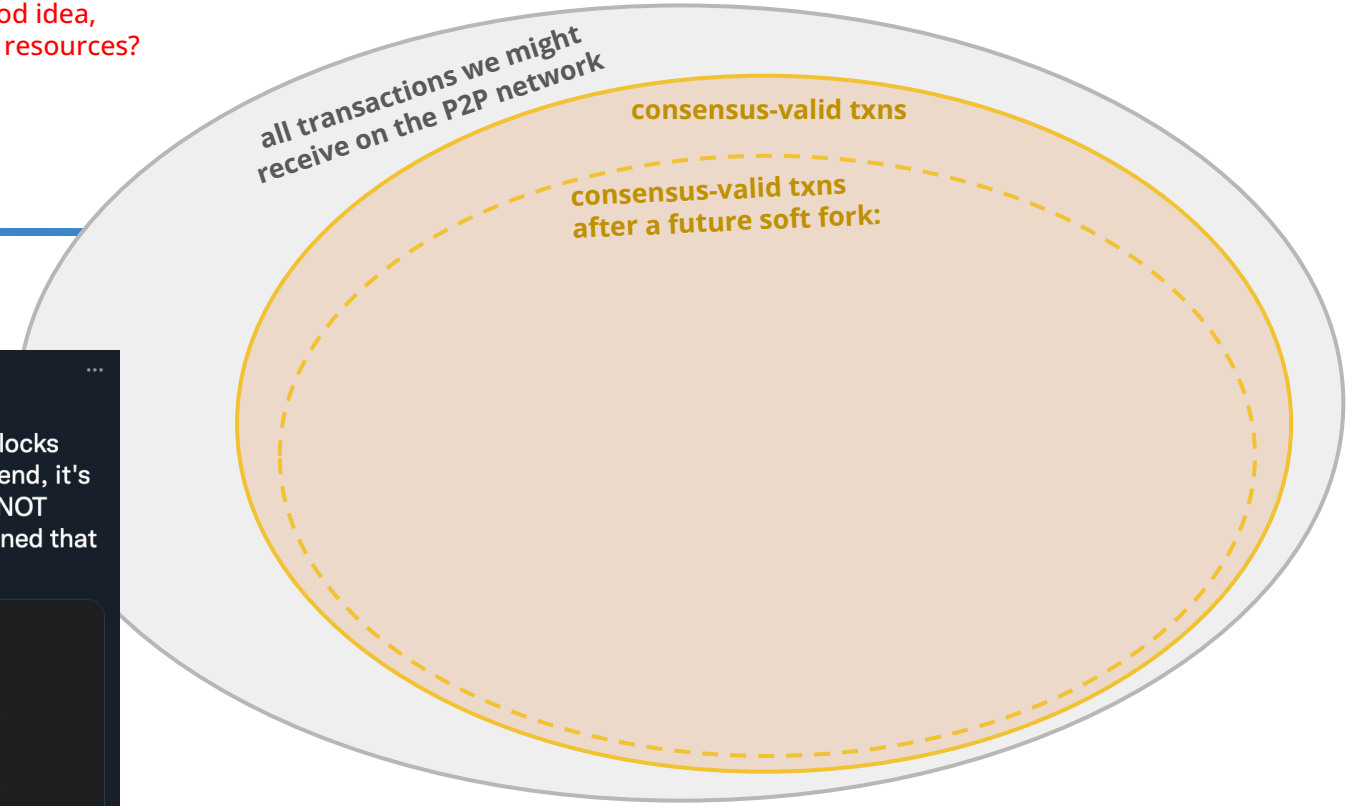
Always validate, accept all
consensus-valid transactions



Would this be a good idea,
even if we had infinite resources?

“Ideal” Mempool
(No Policy)

Always validate, accept all
consensus-valid transactions



b10c
@0xB10C

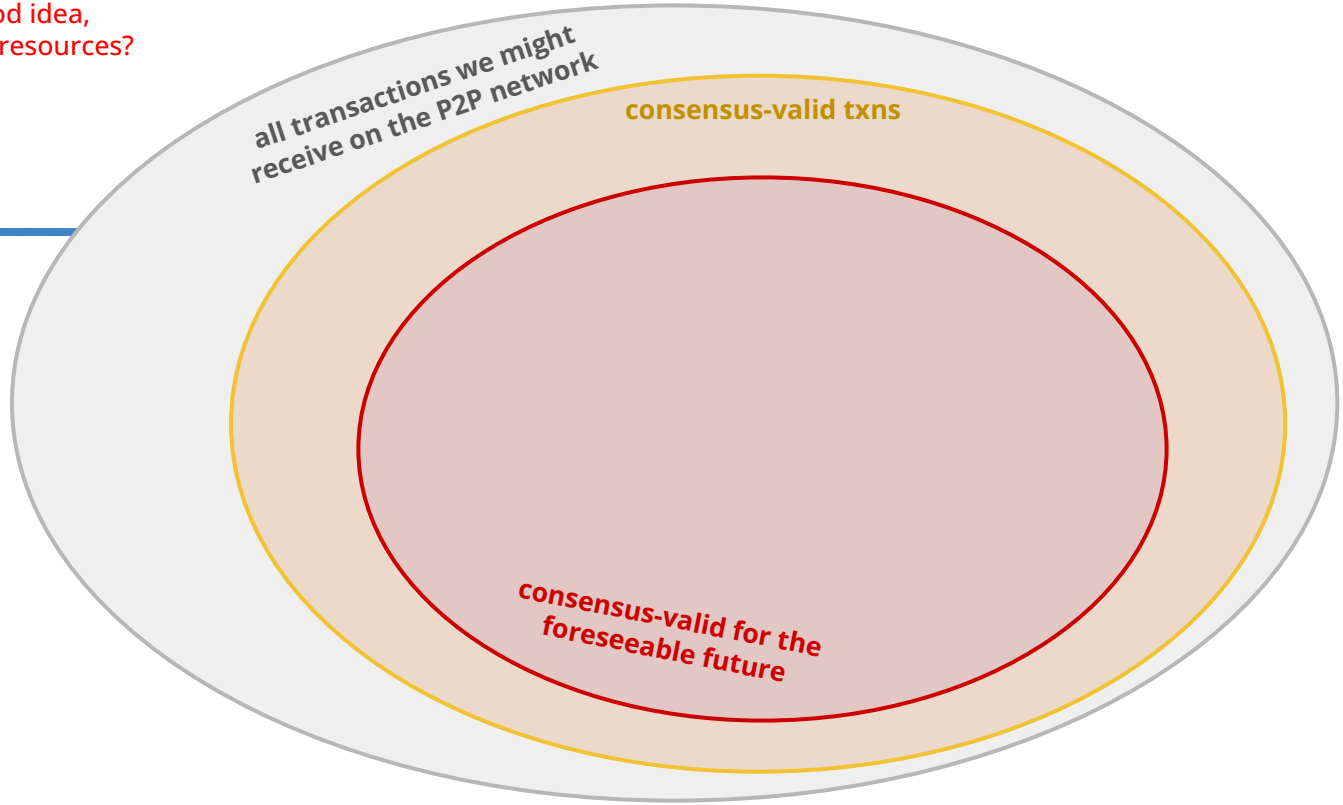
Looking at the pools that mined more than 3 blocks since taproot activation or included a P2TR spend, it's clear that F2Pool and AntPool are, very likely, NOT including P2TR spends. F2Pool already mentioned that they will upgrade their infrastructure soon.

```
up to block 709749:
BTC.com included 9 P2TR spends in 11 mined blocks
SlushPool included 6 P2TR spends in 5 mined blocks
AntPool included 0 P2TR spends in 19 mined blocks
F2Pool included 0 P2TR spends in 23 mined blocks
Poolin included 16 P2TR spends in 19 mined blocks
MARA Pool included 1 P2TR spends in 1 mined blocks
Luxor included 8 P2TR spends in 3 mined blocks
Binance Pool included 3 P2TR spends in 6 mined blocks
Foundry USA included 18 P2TR spends in 11 mined blocks
ViaBTC included 14 P2TR spends in 17 mined blocks
```

Would this be a good idea,
even if we had infinite resources?

“Ideal” Mempool
(No Policy)

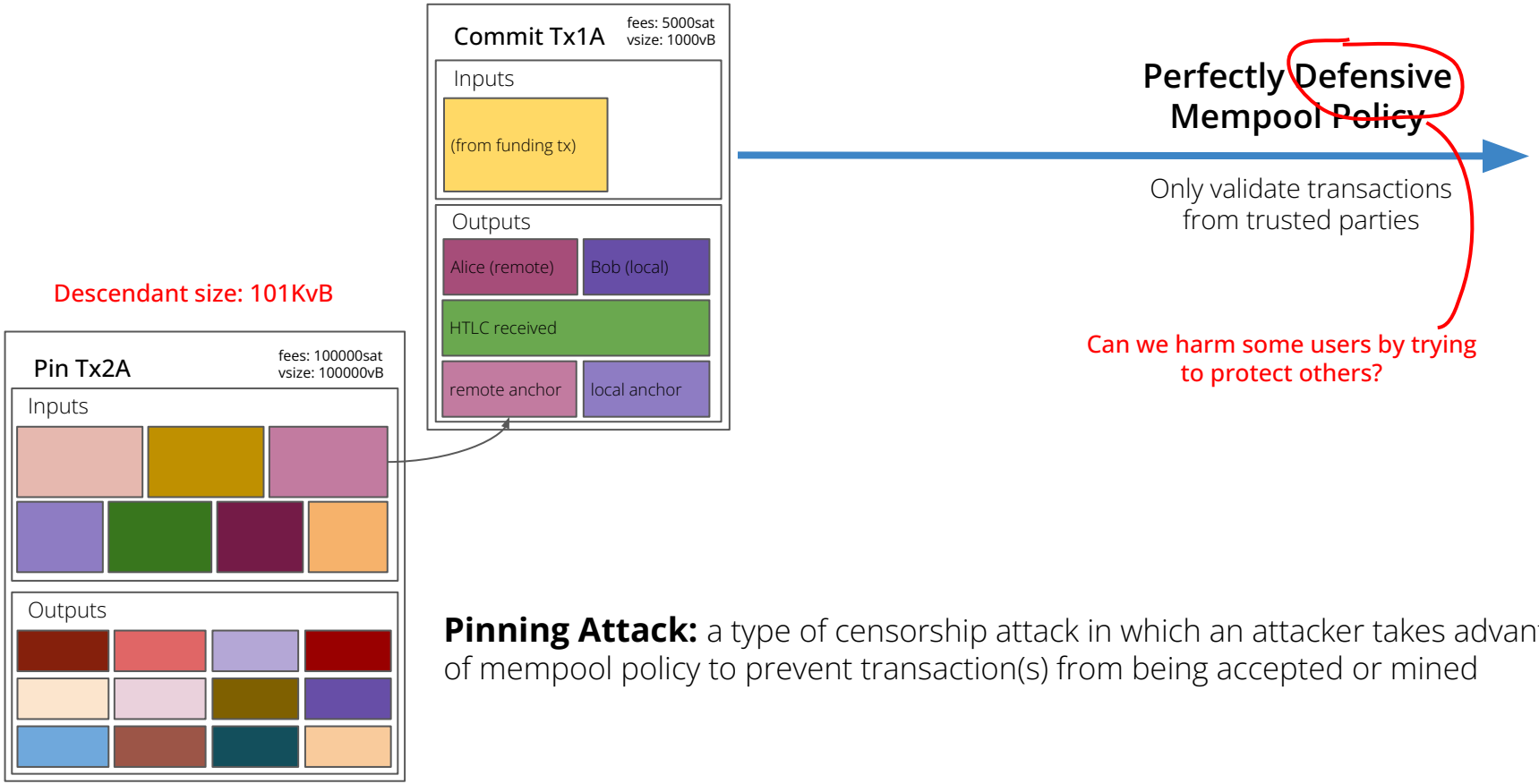
Always validate, accept all
consensus-valid transactions



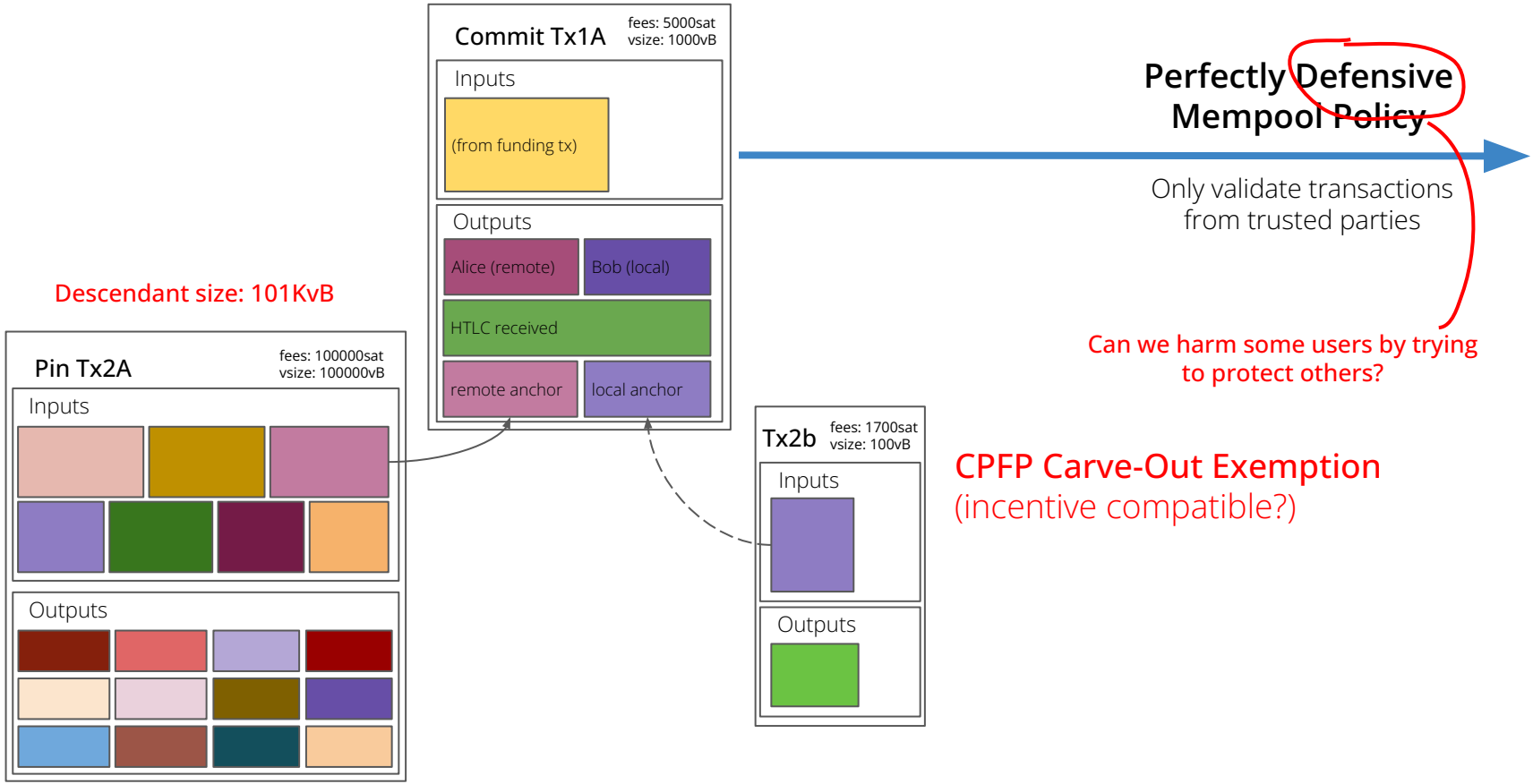
Perfectly Defensive Mempool Policy

Only validate transactions from trusted parties

Can we harm some users by trying to protect others?



Pinning Attack: a type of censorship attack in which an attacker takes advantage of mempool policy to prevent transaction(s) from being accepted or mined

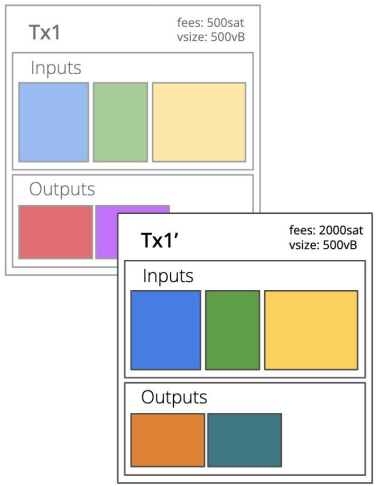

Commit Tx1A
fees: 5000sat
vsize: 1000vB
Inputs
(from funding tx)
Outputs
Alice (remote)
Bob (local)
HTLC received
remote anchor
local anchor
Perfectly Defensive Mempool Policy
Only validate transactions from trusted parties
Can we harm some users by trying to protect others?
Descendant size: 101KvB
Pin Tx2A
fees: 100000sat
vsize: 100000vB
Inputs
Outputs
Tx2b
fees: 1700sat
vsize: 100vB
Inputs
Outputs
CPFP Carve-Out Exemption
(incentive compatible?)

You probably like:

Policy designed for incentive compatibility enables fee-bumping:

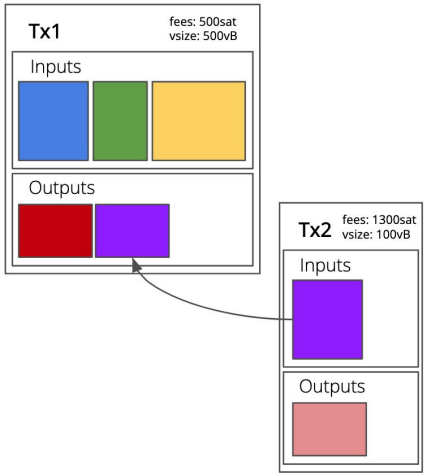
In the case of conflicting transactions,
accept replacements paying higher* fees

Replace By Fee (RBF)



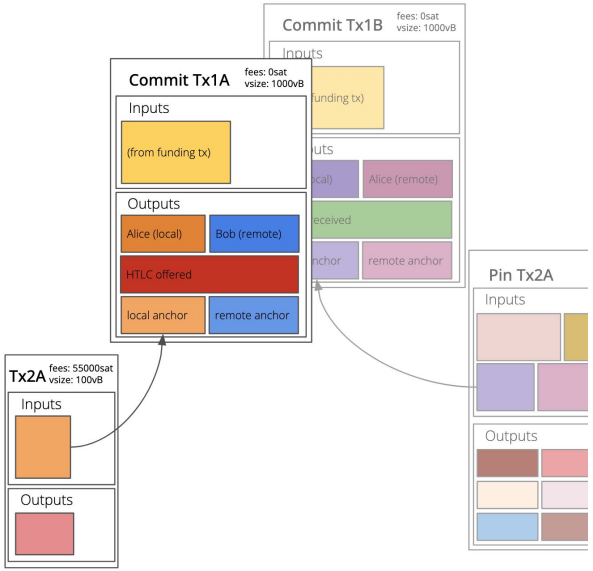
Build blocks by ancestor feerate,
evict by descendant feerate

Child Pays for Parent (CPFP)



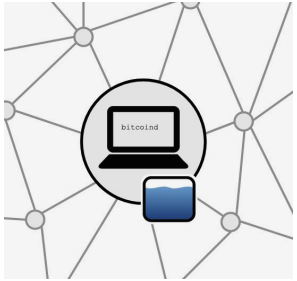
Validate multiple transactions at a time
when individual feerates are insufficient

[WIP] Package RBF

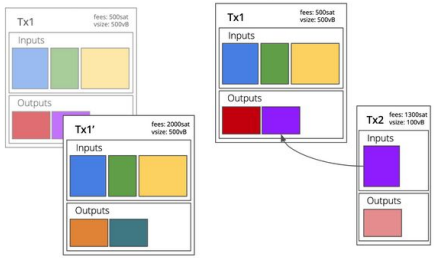


*must follow various other rules, see BIP125

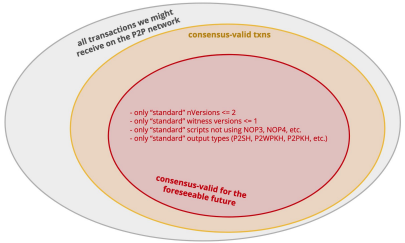
Mempool Policy: a node's set of validation rules, in addition to consensus, which all unconfirmed transactions must pass to be accepted to the node's mempool.



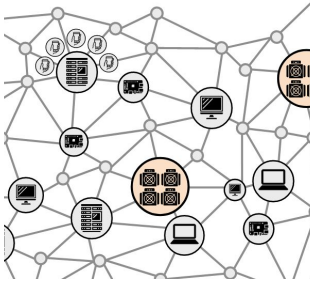
Denial of Service Protection



Incentive Compatibility



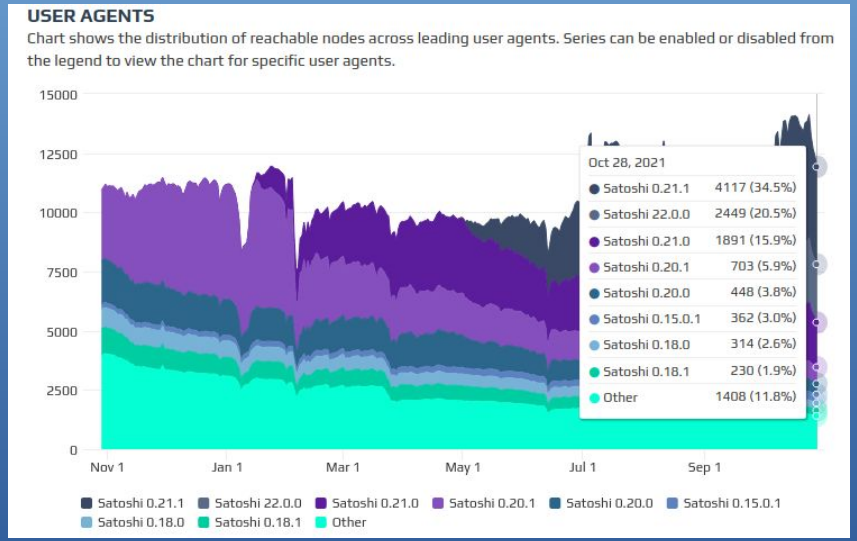
Network Upgradability



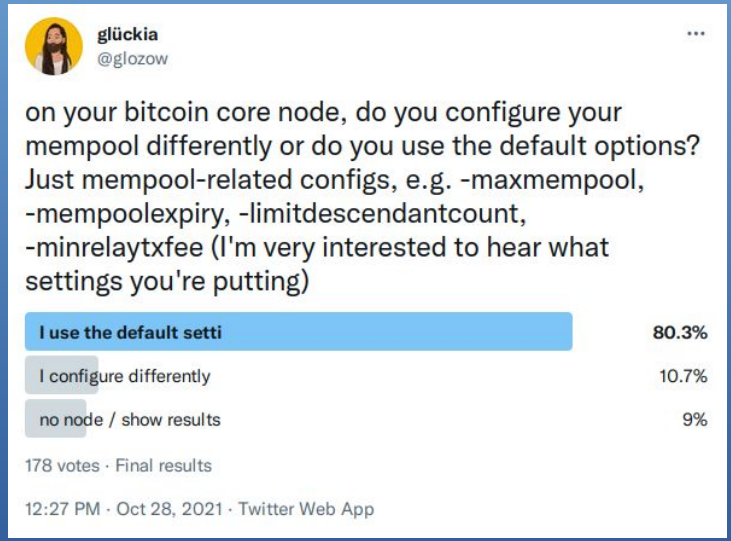
Best Practices, "Standardness"

transaction relay

Mempool Policy: a node's set of validation rules, in addition to consensus, which all unconfirmed transactions must pass to be accepted to the node's mempool.



From https://bitnodes.io on October 29, 2021



Known Policy Issues

Policies can seem arbitrary/opaque and make transaction relay unpredictable

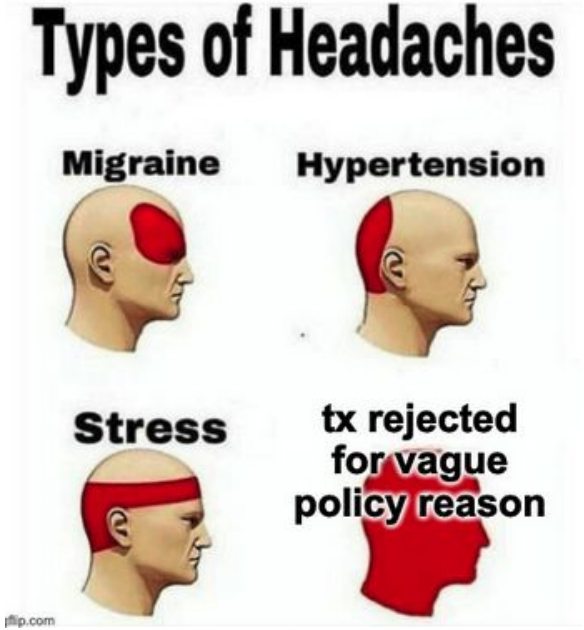
"Standardness" of transaction itself:

- The Dust Limit
- Script rules
  - SCRIPT_VERIFY_{MINIMALIF, CLEANSTACK, LOW_S}
  - Maximum 1 OP_RETURN, maximum 80B NULL_DATA

Evaluation of transaction in mempool context:

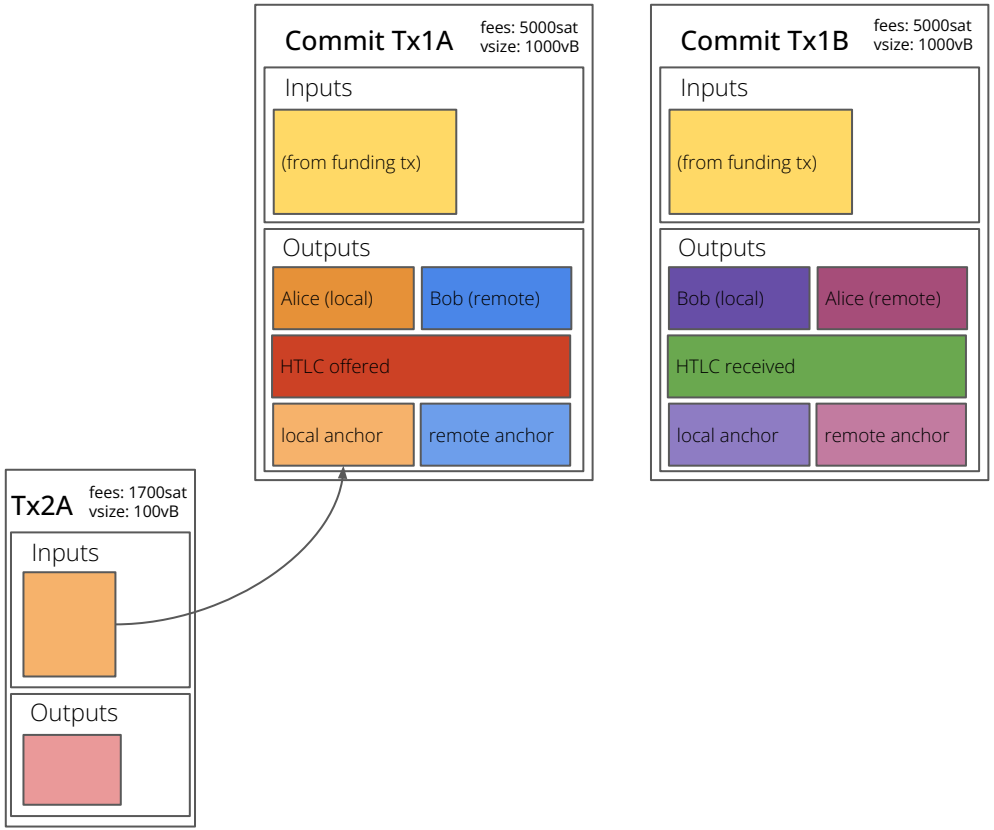
- Ancestor/Descendant niche exemptions
- BIP125 RBF Rules
- In high transaction volume, fee-bumping not guaranteed

The worst one: every mempool may have different policies.



Known Policy Issues

Commitment Transactions cannot replace one another



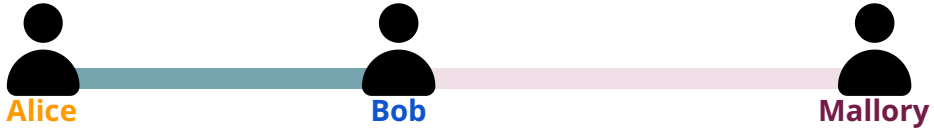
RBF only applies for a single replacement transaction.

Mempools accept the one they see first.

Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

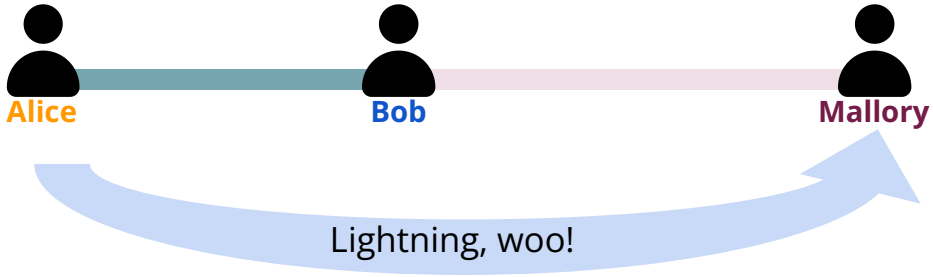


Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.



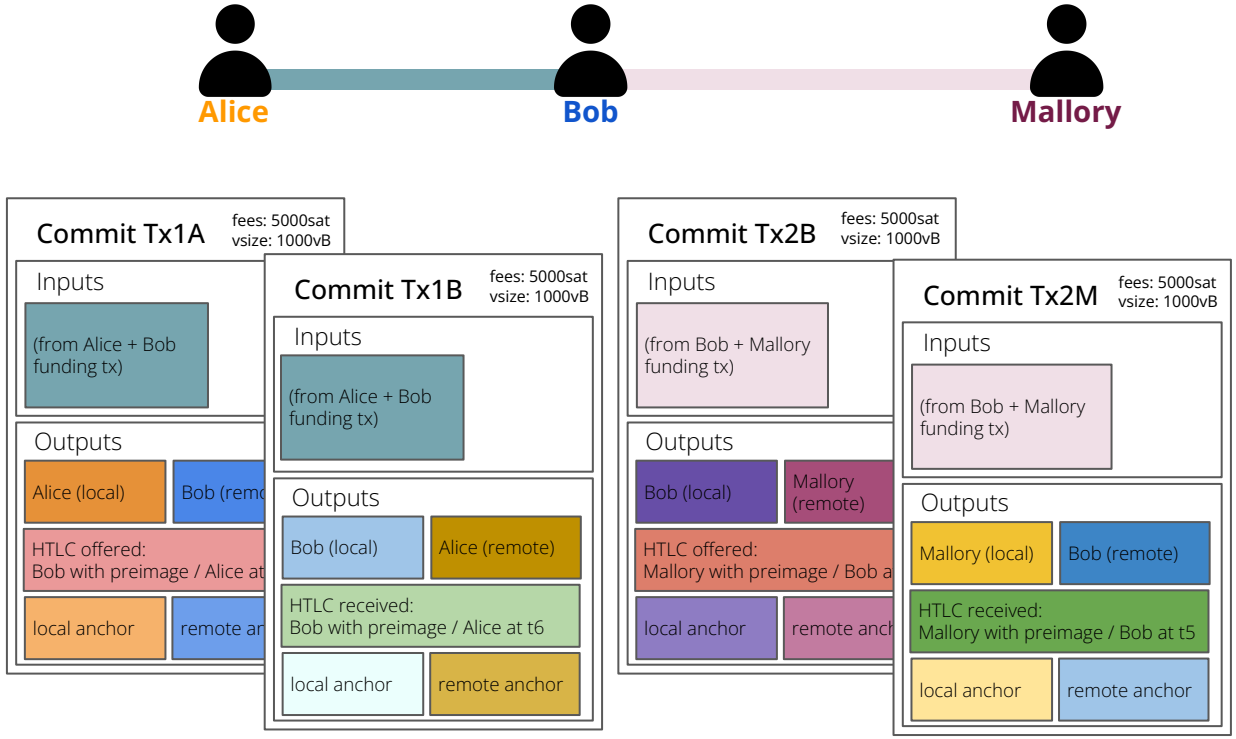
Expected outcomes:
- Both get paid
- Nobody gets paid

Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6



Known Policy Issues

Lightning Attacks

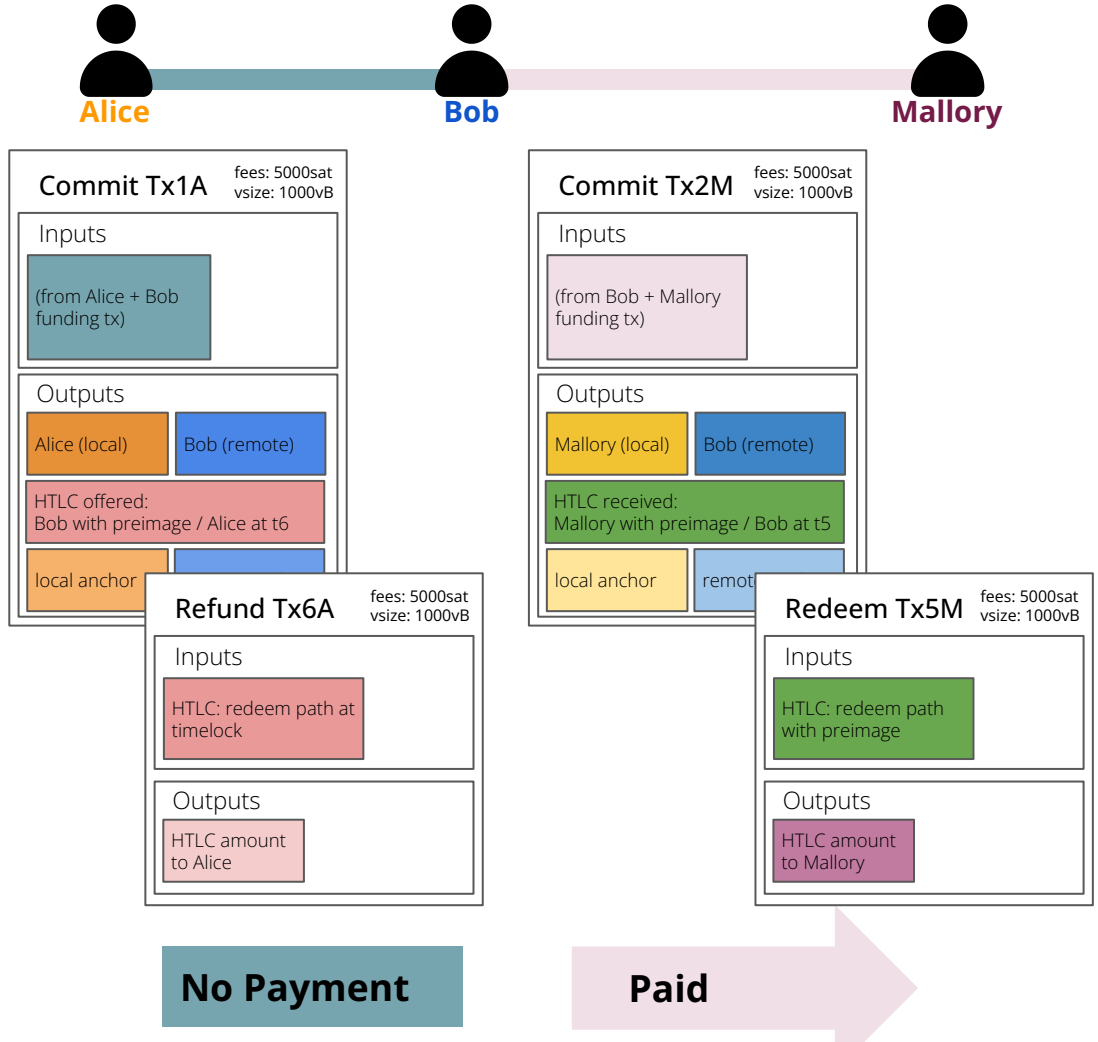
t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

(spoilers!)

t6. Alice gets refund when timelock expires

t7. Mallory redeems HTLC with preimage.
Bob loses HTLC amount.



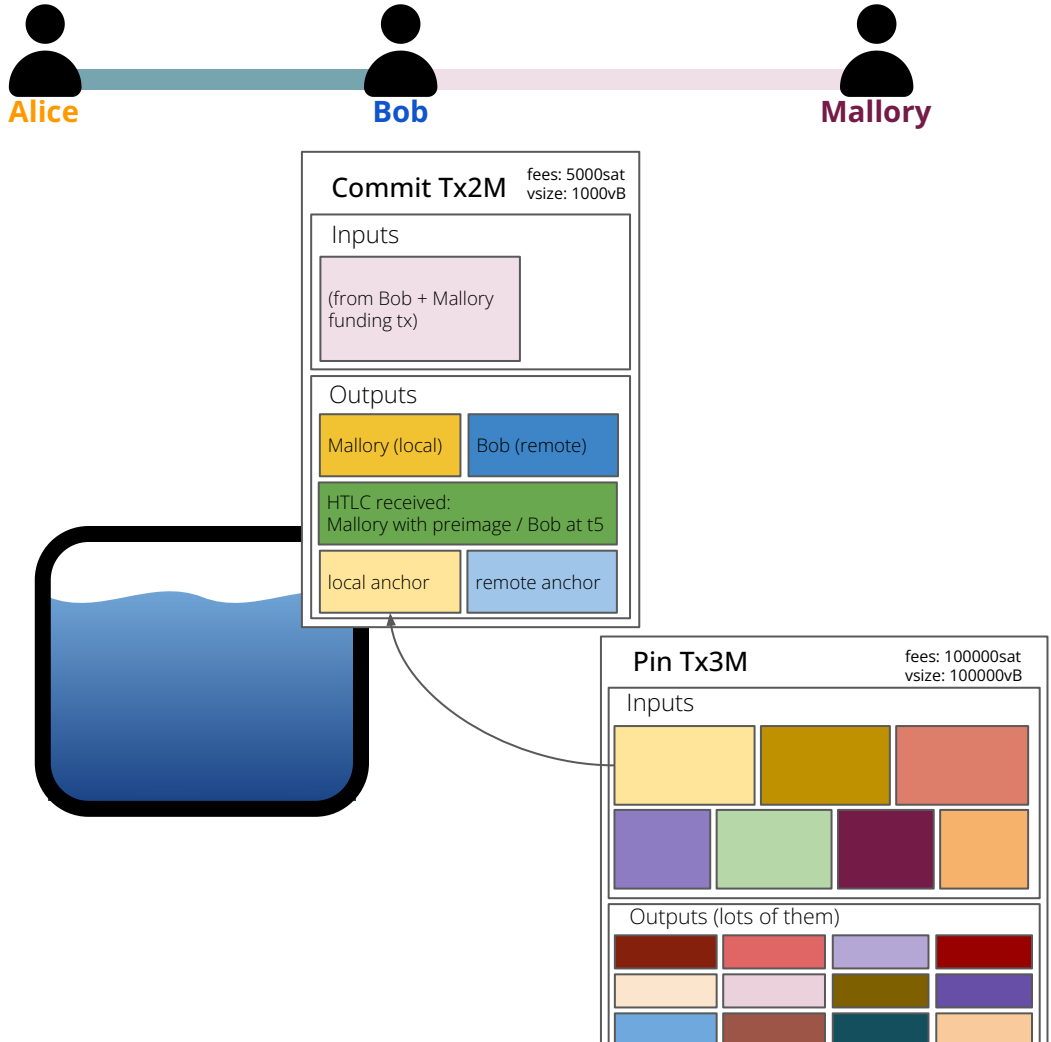
Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

t4. Mallory broadcasts Tx2M + Tx3M



Known Policy Issues

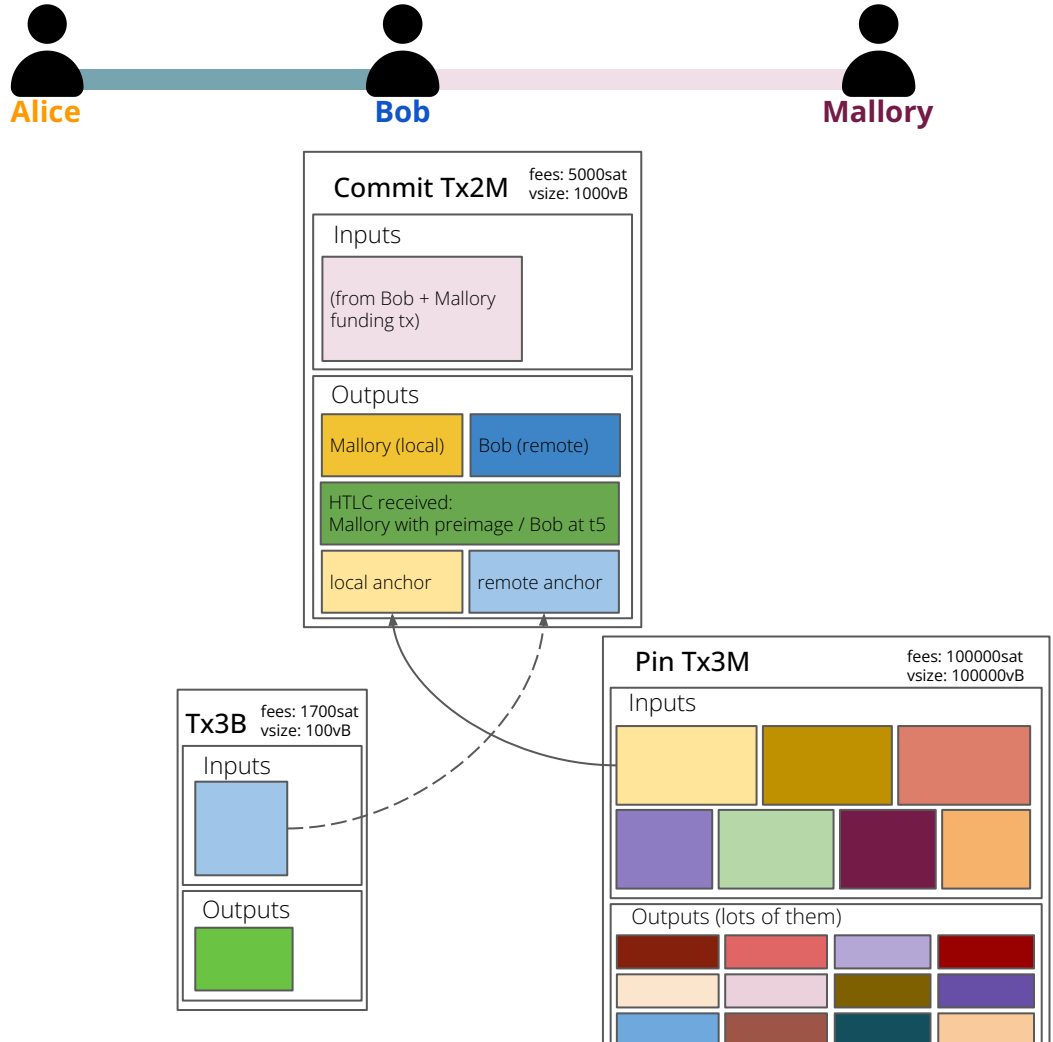
Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

t4. Mallory broadcasts Tx2M + Tx3M.

t5. Case #1: Bob has mempool
Need CPFP Carve-Out



Known Policy Issues

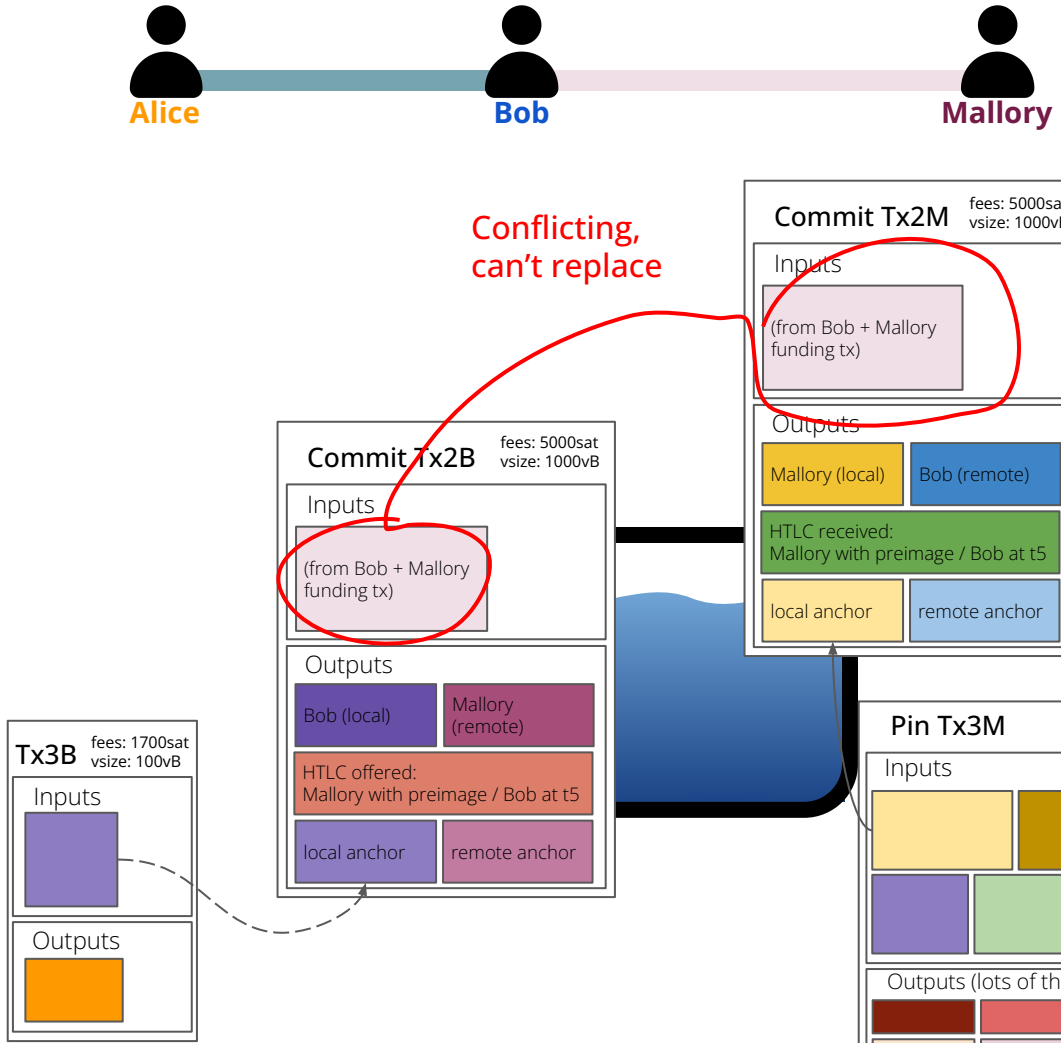
Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

t4. Mallory broadcasts Tx2M + Tx3M.

t5: Case #2: Bob doesn't have mempool,
tries to close by broadcasting Tx2B + Tx3B
Need Package RBF + Package Relay



Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

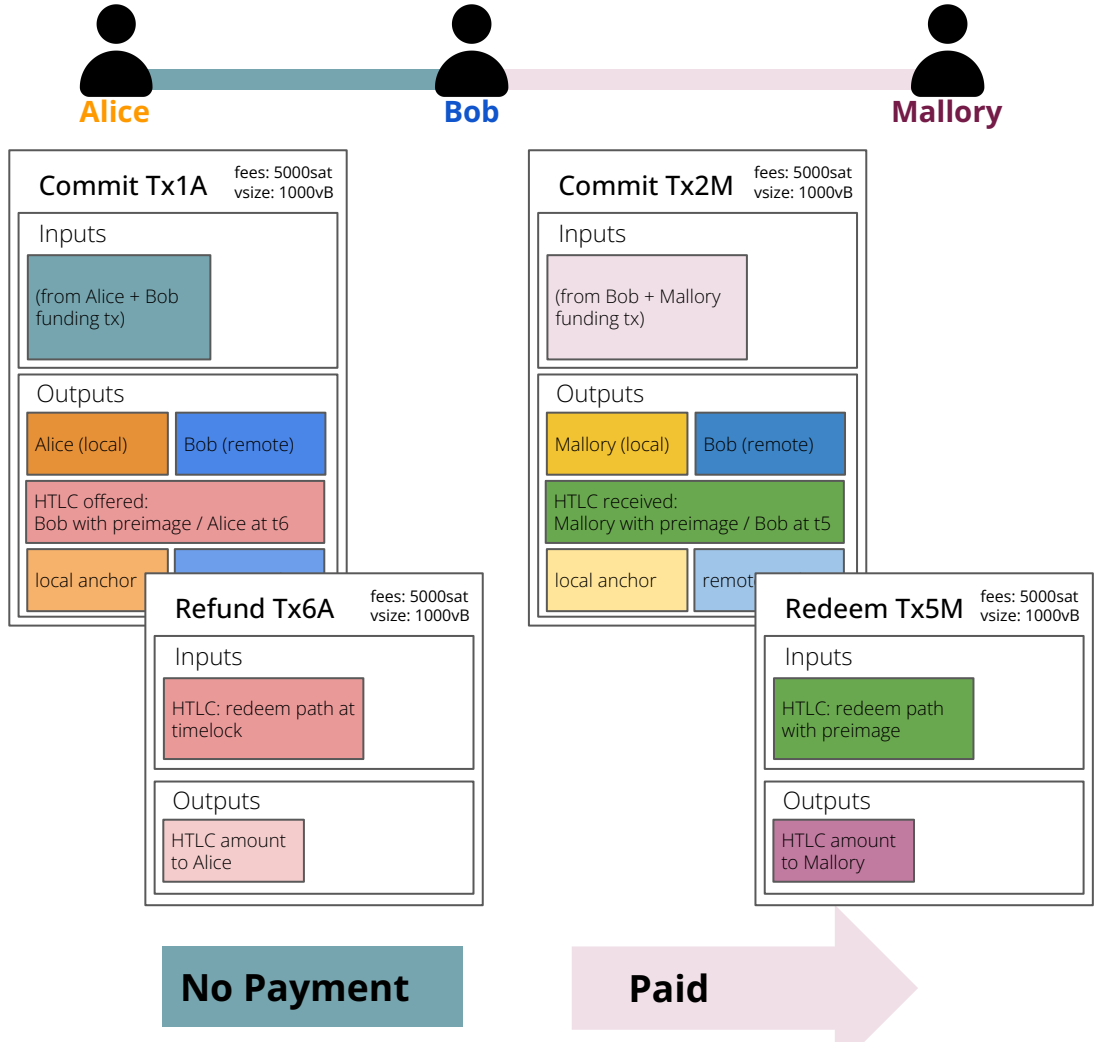
t4. Mallory broadcasts Tx2M + Tx3M

t5. Case #1: Bob has mempool
Need CPFP Carve-Out

Case #2: Bob doesn't have mempool
Need Package RBF + Package Relay

t6. Alice gets refund when timelock expires

t7. Mallory redeems HTLC with preimage.
Bob loses HTLC amount.



Known Policy Issues

Lightning Attacks

t0. Alice + Bob channel
Bob + Mallory channel

t1. Alice pays Mallory through Bob.
Bob can get refund at t5
Alice can get refund at t6

t4. Mallory broadcasts Tx2M + Tx3M

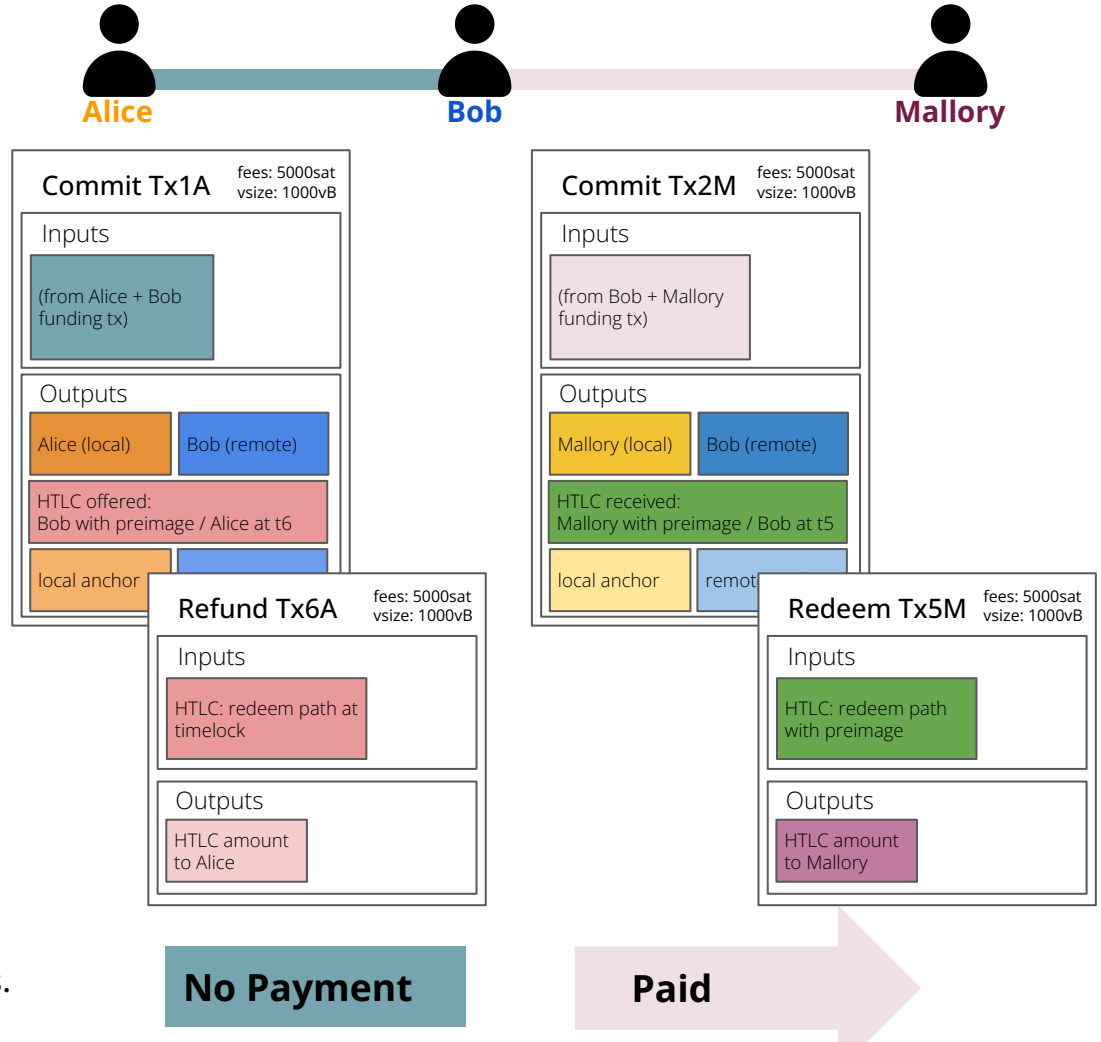
t5. Case #1: Bob has mempool
Need CPFP Carve-Out

Case #2: Bob doesn't have mempool
Need Package RBF + Package Relay

t6. Alice gets refund when timelock expires

t7. Mallory redeems HTLC with preimage.
Bob loses HTLC amount.

Disclaimer: Easily avoided with generous feerates.



Let’s try to be friends?

L1 devs want transactions to propagate

- Documentation and testing interface
- Improvements & Simplification
- Don’t restrict before notifying bitcoin-dev

L2 devs want transactions to propagate

- Never rely on zero-conf tx you don’t control
- Lean towards overestimating fees
- Test assumptions with `testmempoolaccept`
- Communicate grievances
- Feedback on proposals?

Thanks!

@glozow