Blockchain 2

Transactions and UTXO

DAT650 Blockchain technology

Leander Jehl

Signatures and Transactions

Transactions

How can we create an application/cryptocurrency on a blockchain?

- What is in the blocks?
- How to build a meaningful application from it?
- Assume anyone can submit data to the blockchain.

Transactions

Digital Signatures

$$pk, sk \leftarrow setup(\kappa)$$

$$\sigma \leftarrow sign(sk, msg)$$

$$bool \leftarrow verify(\sigma, msg, pk)$$

Ideas:

- Use public key as identity.
- Put signed messages on the blockchain. $\langle msg \rangle_\sigma$
- Signed messages are called *transactions.*

Transactions

Accounts

Transactions are: $\langle pk_{from}, pk_{to}, value \rangle_{\sigma}$

State is: balance for each public-key

Checks:

- Is signature correct?
- Does pk_{from} have enough money?

Transactions

Accounts

Transactions are: $\langle pk_{from}, pk_{to}, value \rangle_{\sigma}$

Algorithm 1 Account transactions

```
balances := [pk]uint
for block in chain do
    for ⟨pk_from, pk_to, value⟩_σ in block.data do
        if !verify(pk_to||value, pk_from, σ) then
            continue
        if balances[pk_from] < value then
            continue
        balances[pk_from] −= value
        balances[pk_to] += value
```

Transactions

Accounts

Transactions are: $\langle pk_{from}, pk_{to}, value \rangle_{\sigma}$

State is: balance for each public-key

Checks:

- Is signature correct?
- Does pk_{from} have enough money?

Problems:
1. **How to deposit money?**
2. **Replay attack!**

Transactions

Accounts

Transactions are: $\langle pk_{from}, pk_{to}, value \rangle_{\sigma}$

Deposit:

- Give out some money
- Deposit with someone who has money

Replay attack:

- A signed transaction can be submitted multiple times.
- Sequence numbers!

Transactions

Accounts

Algorithm 2 Account transactions

```
1:  balances := [pk]uint
2:  sqNrs := [pk]uint
3:  for block in chain do
4:      for ⟨pk_from, pk_to, value, sqNr⟩_σ in block.data do
5:          if !verify(pk_to||value||sqNr, pk_from, σ) then
6:              continue
7:          if balances[pk_from] < value then
8:              continue
9:          if sqNrs[pk_from] = sqNr then
10:             balances[pk_from] −= value
11:             balances[pk_to] += value
12:             sqNrs[pk_from]++
```

Idea: do checks when adding transaction to chain.

Transactions

UTXO

UTXO: Unspent transaction output

Idea: *No balances but coins*

- For each coin store pk of owner and unique id
- Transaction spends some coints and creates new ones.

Transactions

UTXO

Transactions:

$$tx = \langle [(id_1, \sigma_1), (id_2, \sigma_2)], [(pk_a, value_a), (pk_b, value_b)] \rangle$$

Inputs Outputs

State is unspent outpus $map[id](pk, value)$

Transactions

UTXO

Transactions:

$$tx = \langle [(id_1, \sigma_1), (id_2, \sigma_2)], [(pk_a, value_a), (pk_b, value_b)] \rangle$$

Inputs

Outputs

Valid if:

- Inputs refer to unspent outputs.
- Signatures are correct (with outputs public key)
- Value of all inputs larger or equal than all output values.

Transactions

UTXO

Algorithm 3 Transaction validation and maintenance of UTXO

$UTXO := map[id](value, pk)$
for $tx = \langle inputs, outputs \rangle$ **do**
  for $(id, \sigma) \in inputs$ **do**
    if $UTXO[id]$ does not exist **then**
      abort ▷ invalid transaction
    if verify$(tx, \sigma, UTXO[id].pk) ==$ **false then**
      abort ▷ invalid transaction
  if sum of values of inputs $<$ sum of values of new outputs **then**
    abort ▷ invalid transaction
  for $((id, \sigma) \in inputs$ **do**
    remove$(UTXO[id])$ ▷ output spent
  for $((pk, value) \in inputs$ **do**
    $UTXO[newid] = (pk, value)$ ▷ add new outputs

Transactions

UTXO

Transactions:

$$tx = \langle [(id_1, \sigma_1), (id_2, \sigma_2)], [(pk_a, value_a), (pk_b, value_b)] \rangle$$

Inputs Outputs

- No replay attack
- What to sign: $\langle [id_1, id_2], [(pk_a, value_a), (pk_b, value_b)] \rangle$

Transactions

Accounts vs. UTXO

Assuming only valid transactions on chain,
how to verify that a pk has money.

Accounts: Check all received and sent transactions.

UTXO: Check received output and that it is unspent.

Transactions

Accounts vs. UTXO

Assuming only valid transactions on chain,
how to verify that a pk has money.

Accounts: Check all received and sent transactions.

UTXO: Check received output and that it is unspent.

Does UTXO provide anonymity/prevent tracing?

Transactions

Accounts vs. UTXO

Assuming only valid transactions on chain,
how to verify that a pk has money.

Accounts: Check all received and sent transactions.

UTXO: Check received output and that it is unspent.

Does UTXO provide anonymity/prevent tracing?

- Also in UTXO transactions from one sender can be traced.
- But most untracable solutions build on UTXO

Take away

A blockchain is an append only log secured against changed.

Transactions/state changes are recorded in the blockchain.

Application state can be recreated by applying all transactions.