Digital Signatures

Public & Private Key pairs

- Private key: needed to sign a message
- Public key: used to verify a message & signature

3 protocols:

1. KeyGen() → s, v (s: secret signing key; v: (public) verif. key)
2. Sign(m, s) → t
3. Verif(v, t, m) → pass or fail

What are wallet ids?

- Public (verif) keys
  1. Pubkey of block miner
  2. Recipients of transactions

What do we sign?

Whoever's paying signs a transaction.

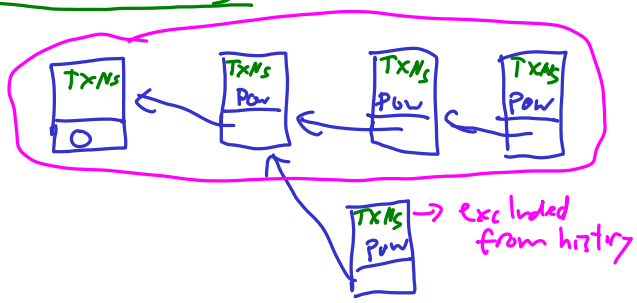
Ed25519 Signatures
(nacl.signing) pynacl

Private, pub keys: 32 bytes each

Signature: 64 bytes

ECDSA, RSA
- works both ways
- huge key sizes

Transactions



Longest chain: send_chat (arbitrary)

server /push
(prefer current head)

Bitcoin solved:

1) Consensus (distributed)
   - Previous block hash
     linear history)
   - Longest chain rule
   - Proof of work
     (10 mins per block)

Bitcoin Paper solves:

(2) "Double spend problem"

- Prevent a valid, signed transaction from being duplicated.

Transaction contents

- Signature of payer
- Acct no. of payer → (pub key)
- Pub key of recipient
- Amount
- Unique id, avoid cashing twice
- Date (we won't focus on)
  - ↳ earliest that txn may be included

Validity checks

- Hasn't been spent in same chain
- Date not in future
- Signature is valid
- Amount ≤ amount available

Bitcoin Transaction

Version num. (features & improvements)

Inputs (list)
- UTXO → unspent transaction output
- Signature

Outputs (list)
- Public Key (who gets paid)
- Amount

Single transaction has multiple ins + outs.

Date (locktime)

Why multiple inputs?

May have multiple smaller amounts to combine

Why multiple outputs?

→ Pay yourself the "change"

Fees

Must have

$\sum$ input amts $\geq$ $\sum$ output amts

↓

difference is a "fee" paid to block miner

Coinbase transaction

- first txn in a block
- No inputs
  (Implicitly: Block reward + all fees)
- Outputs as usual

How to refer to transaction?

txid : Hash of entire transaction
index : which output

~~block id~~ (can't know in advance)

→ UTXO