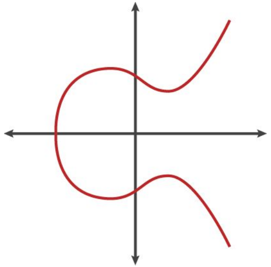
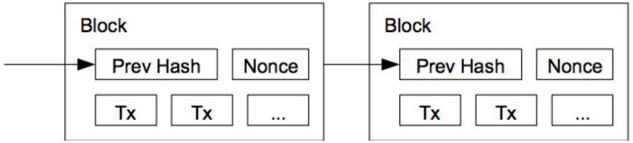
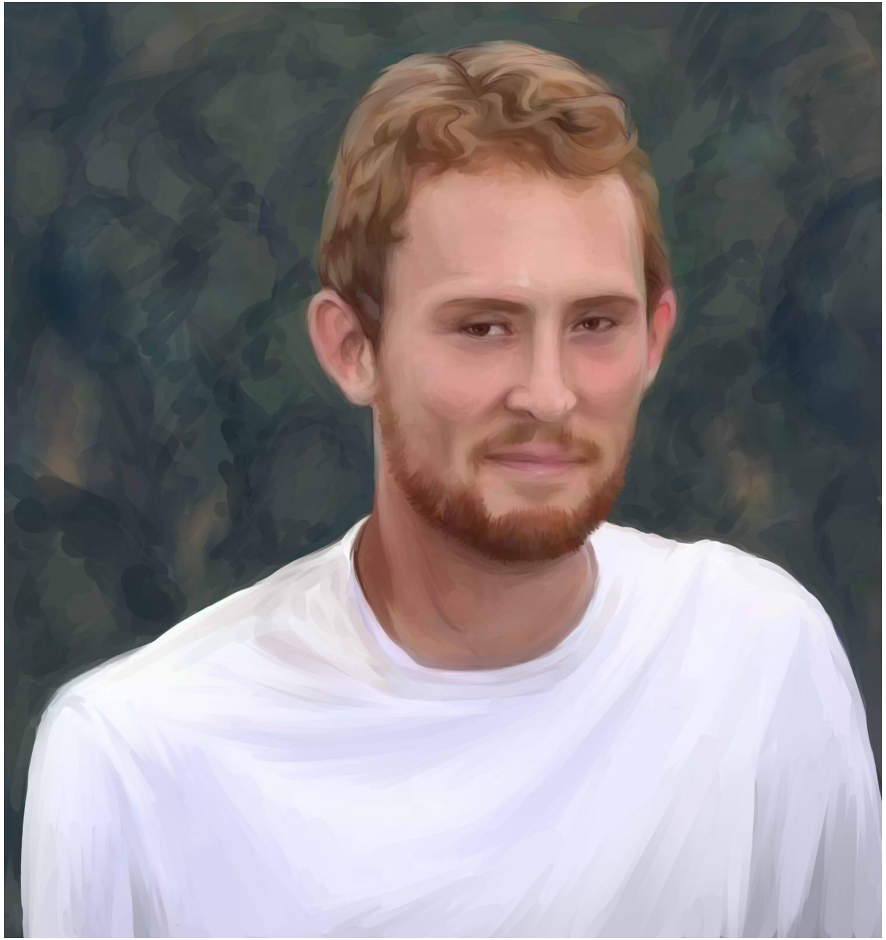
Technical Intro to Bitcoin Montreal





Alex Melville

5F11 78CD D43A 49E9 10D6 D27C 773A E36E 3704 569C



Alex Melville

Software Engineer @BitGo

World Traveler

github.com/Melvillian/talks

Technical Intro to Bitcoin

- Sending a Transaction
  - Developers
  - Miners (Secure the Network)
    - Cryptographic Hash Functions
  - Addresses
  - Digital Signatures
  - Transaction Structure and Signing
  - Gossip Protocol
  - Bitcoin Script (if we have time!)

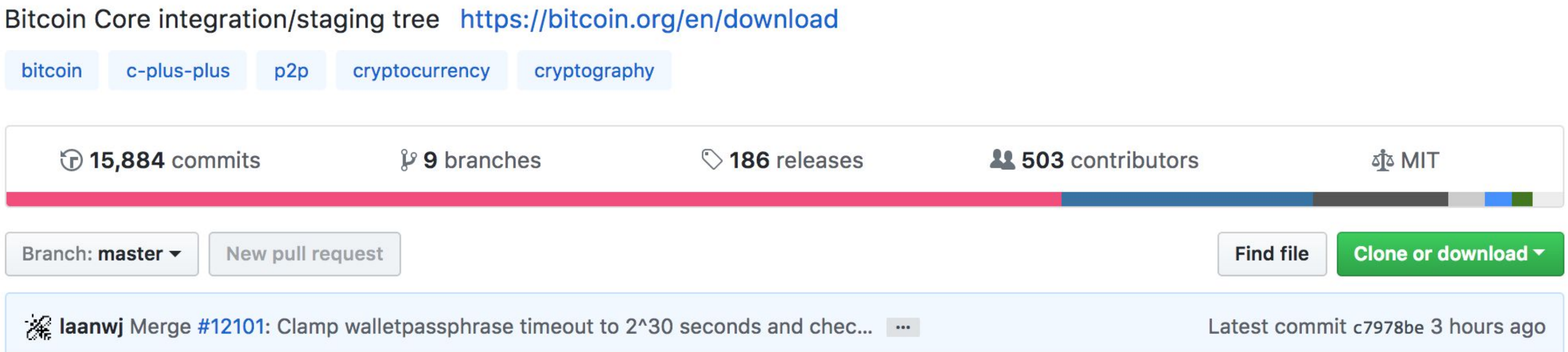
Read the Satoshi Whitepaper!

https://bitcoin.org/bitcoin.pdf

Only 9 pages!

Developers

- Bitcoin Core (open source software)
  - Over 500 unique contributors around the planet
  - Generates, communicates, and validates blocks and transactions
  - One of many bitcoin clients (but certainly the most popular!)
    - Btcd (Golang)
    - Nbitcoin (.Net)
    - Bcoin (Javascript)



Bitcoin Core integration/staging tree https://bitcoin.org/en/download

15,884 commits

9 branches

186 releases

503 contributors

MIT

laanwj Merge #12101: Clamp walletpassphrase timeout to 2^30 seconds and chec... ···

Latest commit c7978be 3 hours ago

Sending a Transaction

- Need to spend bitcoin you already own
- But then… where does the bitcoin you own originally come from?

Mining

- Roughly every 10 minutes, 12.5 bitcoin ($125,000) are generated out of nothing
- This is the miner’s incentive/reward for securing the network

Mining

- Roughly every 10 minutes, 12.5 bitcoin ($125,000) are generated out of nothing
- This is the miner’s incentive/reward for securing the network
- What does “securing the network” mean?

Mining

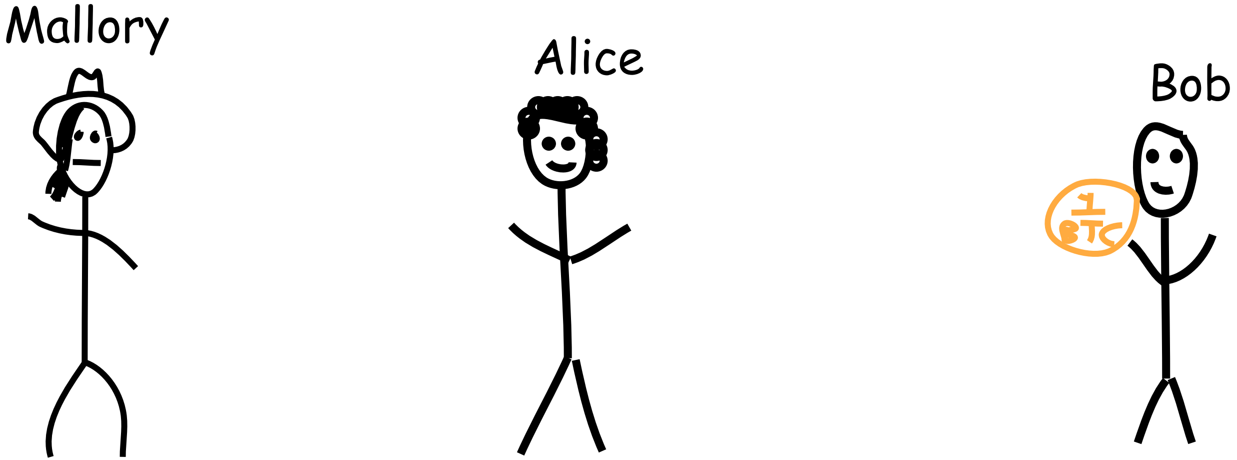
- Solves 2 problems
    - How to generate bitcoin without a third party (bank)

Mining

- Solves 2 problems
  - How to generate bitcoin without a third party (bank)
  - How to prevent double spend attack

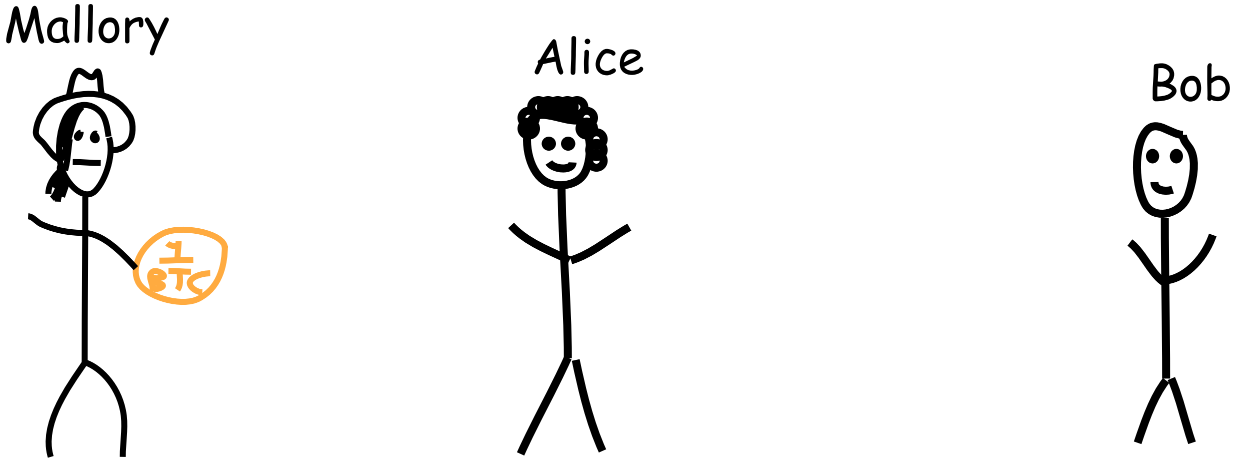
Mining

- Solves 2 problems
  - How to generate bitcoin without a third party (bank)
  - How to prevent double spend attack



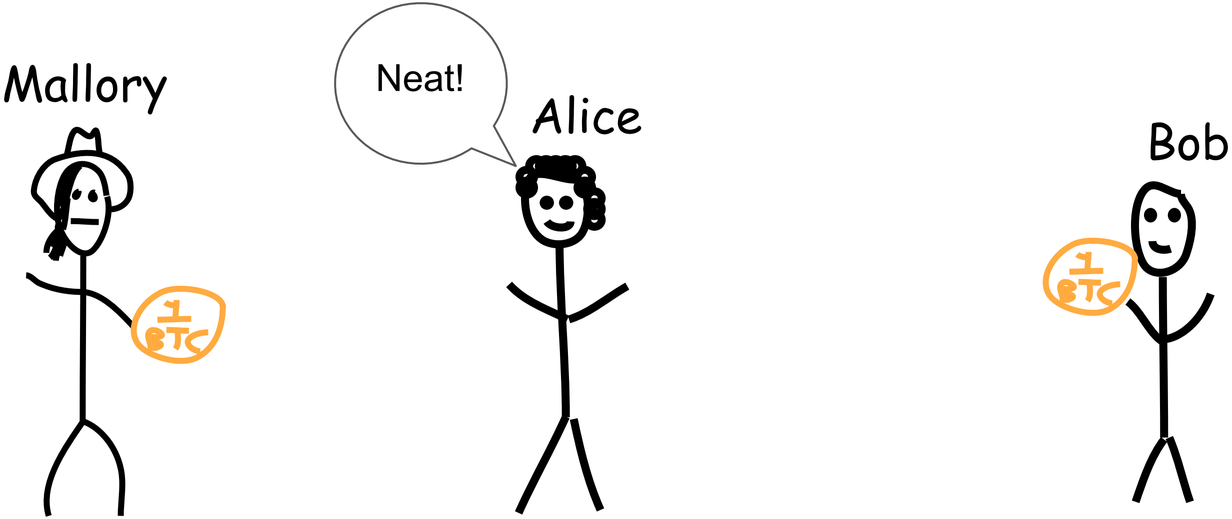
Mining

- Solves 2 problems
  - How to generate bitcoin without a third party (bank)
  - How to prevent double spend attack



Mining

- Solves 2 problems
  - How to generate bitcoin without a third party (bank)
  - How to prevent double spend attack

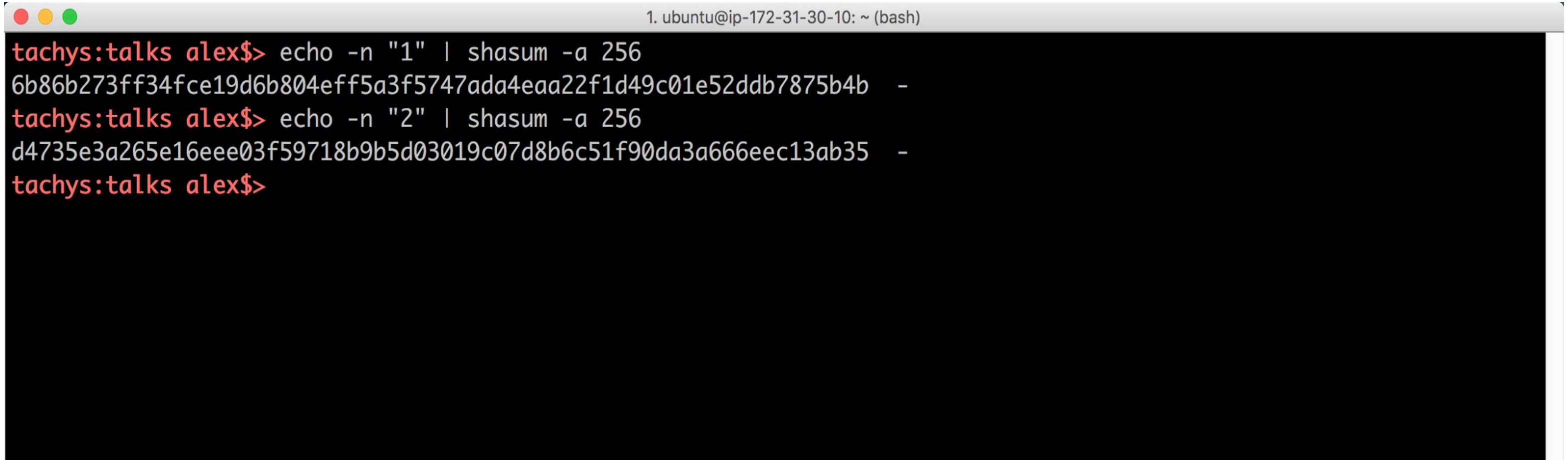


Mining

So what does mining actually involve?

Cryptographic Hash Functions

- “Hash” as in Hashmap (with keys)
- Given some input data, map it to a random output data
- Even a single bit difference will change roughly half of the bits in the output data
- Given a hash, you should not be able to guess the data that hashed to it



```
tachys:talks alex$> echo -n "1" | shasum -a 256
6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b  -
tachys:talks alex$> echo -n "2" | shasum -a 256
d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35  -
tachys:talks alex$>
```

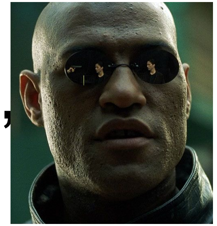
Mining

- The random nature means mining is a random process, and the target value tunes how long it takes to mine a block

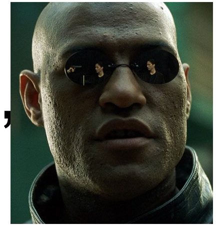
SHA256(SHA256(Block Data + nonce)) =
0000000000000000005b5691dbe96364074f0e066631e1f8ae45e84ae495a89b

(Bitcoin block from today, January 17th, 11:25am EST)

Mining: “There are fields Neo, endless fields…”



Mining: “There are fields Neo, endless fields…”



Sending a Transaction

- OK, we’ve now got our own coin
- How do we send it to someone?

Addresses

- Long strings of alphanumeric characters with different versions
  - 1D3mnTriicrjdcKhucHm6CAfqy7gNfGcyt (P2PKH)
  - 3JN9RvhN9TMM4q1Hx6Zta3vvBP9Ps5AmFo (P2SH)

Addresses

- Long strings of alphanumeric characters with different versions
  - 1D3mnTriicrjdcKhucHm6CAfqy7gNfGcyt (P2PKH)
  - 3JN9RvhN9TMM4q1Hx6Zta3vvBP9Ps5AmFo (P2SH)
- Generated from a public/private ECDSA key pair
  - xpub661MyMwAqRbcFtXgS5sYJABqqG9YLmC4Q1Rdap9gSE8NqtwybGhePY2gZ29ESFjqJoCu1Rupje8YtGqsefD265TMg7usUDFdp6W1EGMcet8
  - xprv9s21ZrQH143K3QTDL4LXw2F7HEK3wJUD2nW2nRk4stbPy6cq3jPPqjiChkVvvNKmPGJxWUtg6LnF5kejMRNNU3TGtRBeJgk33yuGBxrMPHi
- Like email addresses, but cryptographic!

What is a Digital Signature?

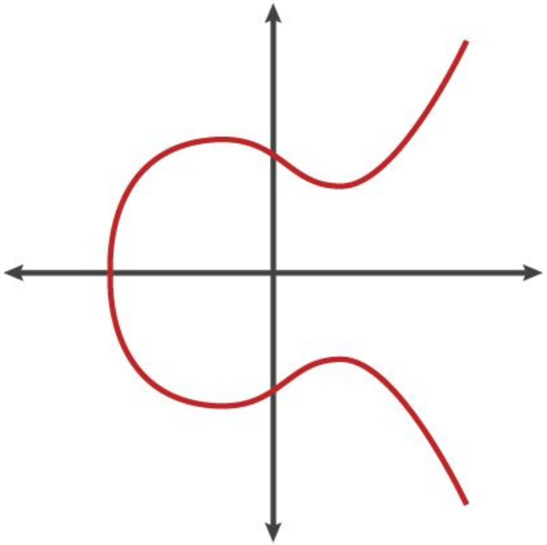
- Like a handwritten signature, allows you to attest to a piece of data
  - Cheques
  - Contracts

- Looks like:

```
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iJwEAQEKAAYFAlRGklcACgkQU805K
63BbbvgkQP/cJktaCbNQtxCfV/ZXliwn
6Mv
tVELtCdcF/JWKD/1BPGaKXT6BiVa6vr
B6dOwRWqUGiZbV1VWkj/LglaMqPa1Z
EnZ
Bwpux8hyUYRNbjnyVSDYCyyBH/qvh
E/9wGgeLRJ5eK/Na6QoKw4XDAo2RH
oiBF3o
wwm6vk4PZF8DacCv64o=
=SadA
-----END PGP SIGNATURE-----
```

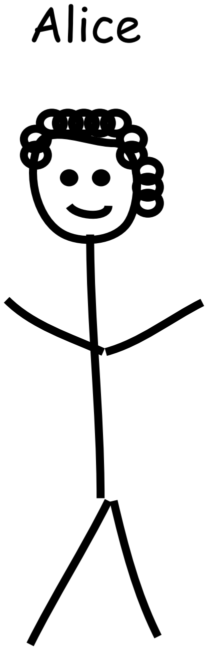
What is a Digital Signature?

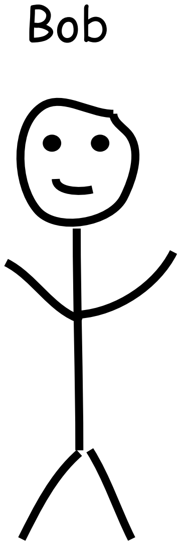


- Like a handwritten signature, allows you to attest to a piece of data
  - Cheques
  - Contracts

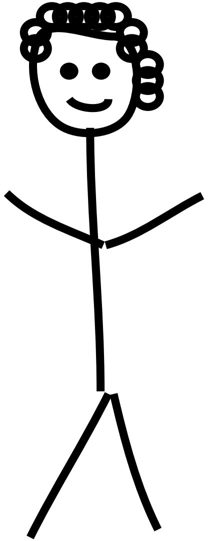
- Looks like:
- Bitcoin uses Elliptic Curves to generate secure public private keys
  - **xpub**661MyMwAqRbcFtXgS5sYJABqqG9YLmC4Q1Rdap9gSE8NqtwybGhePY2gZ29ESFjqJoCu1Rupje8YtGqsefD265TMg7usUDFdp6W1EGMcet8
  - **xprv**9s21ZrQH143K3QTDL4LXw2F7HEK3wJUD2nW2nRk4stbPy6cq3jPPqjiChkVvvNKmPGJxWUtg6LnF5kejMRNNU3TGtRBeJgk33yuGBxrMPHi
- Public keys are used to validate the digital signatures on every transaction

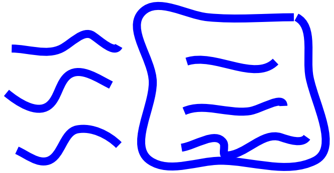
Alice & Bob



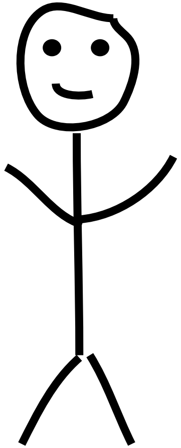


Alice wants to send an invoice to Bob



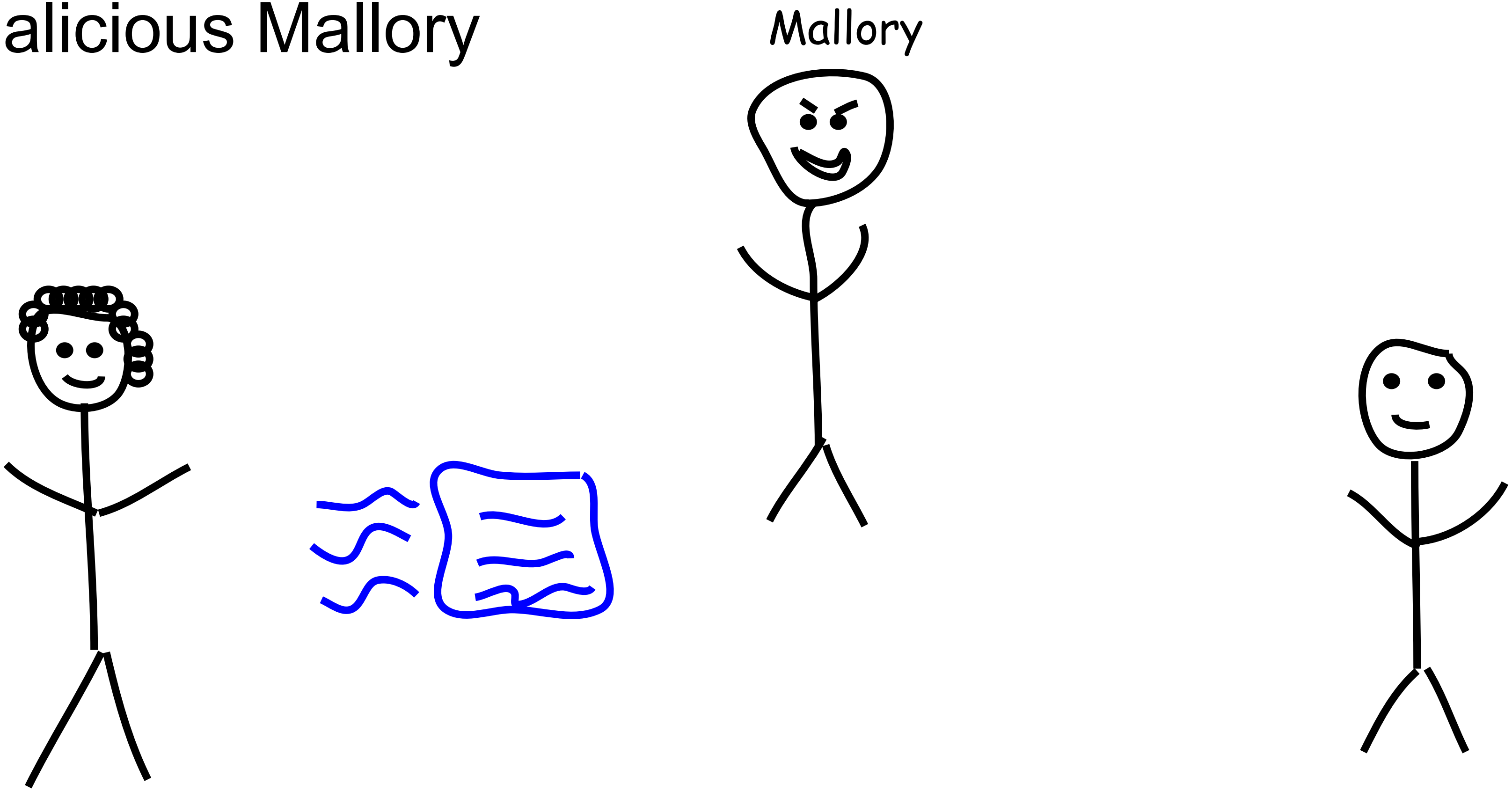


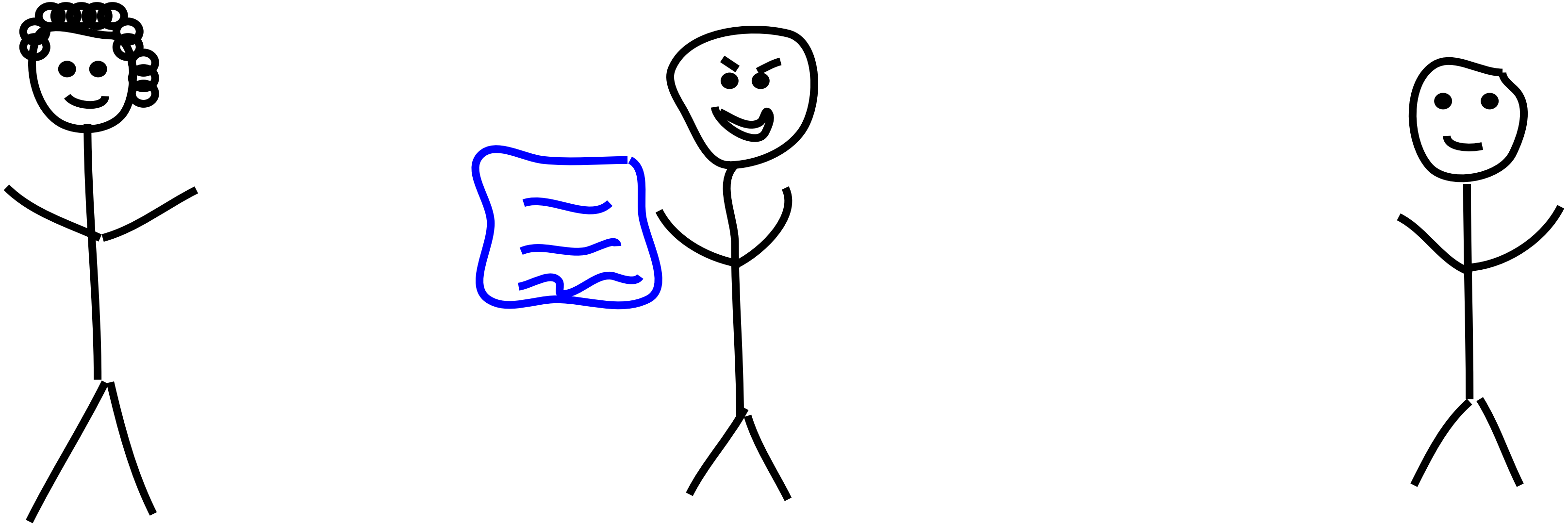
Pay rent to
account
#79BE667E



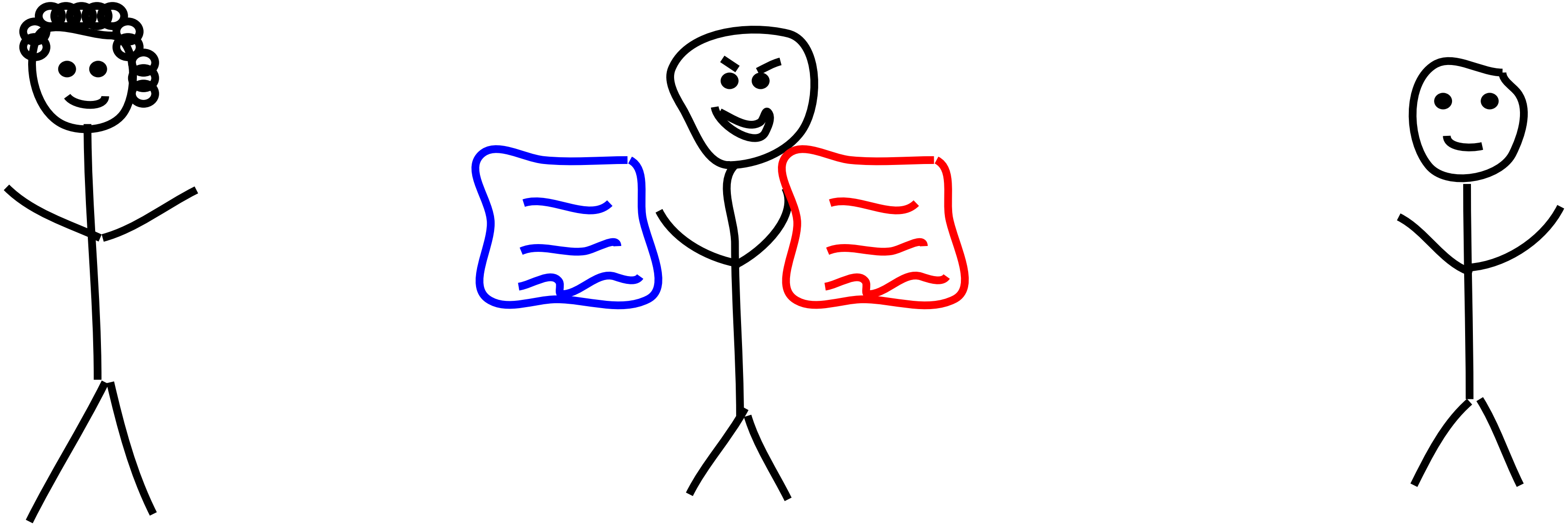
OK!

Malicious Mallory

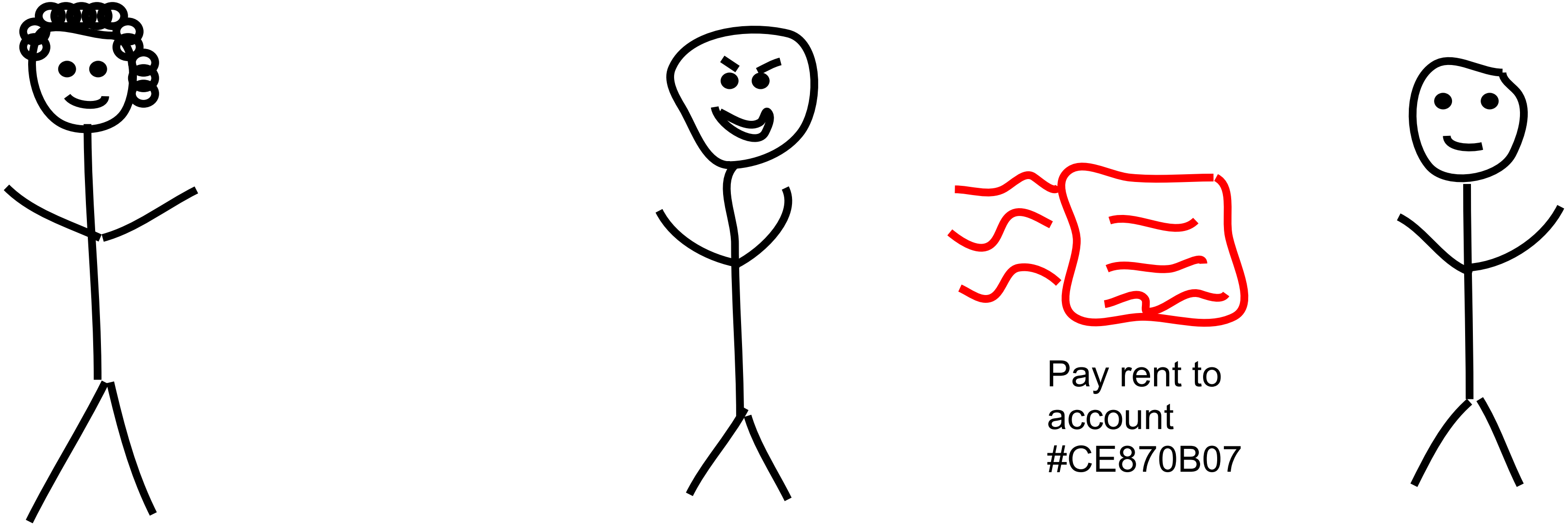




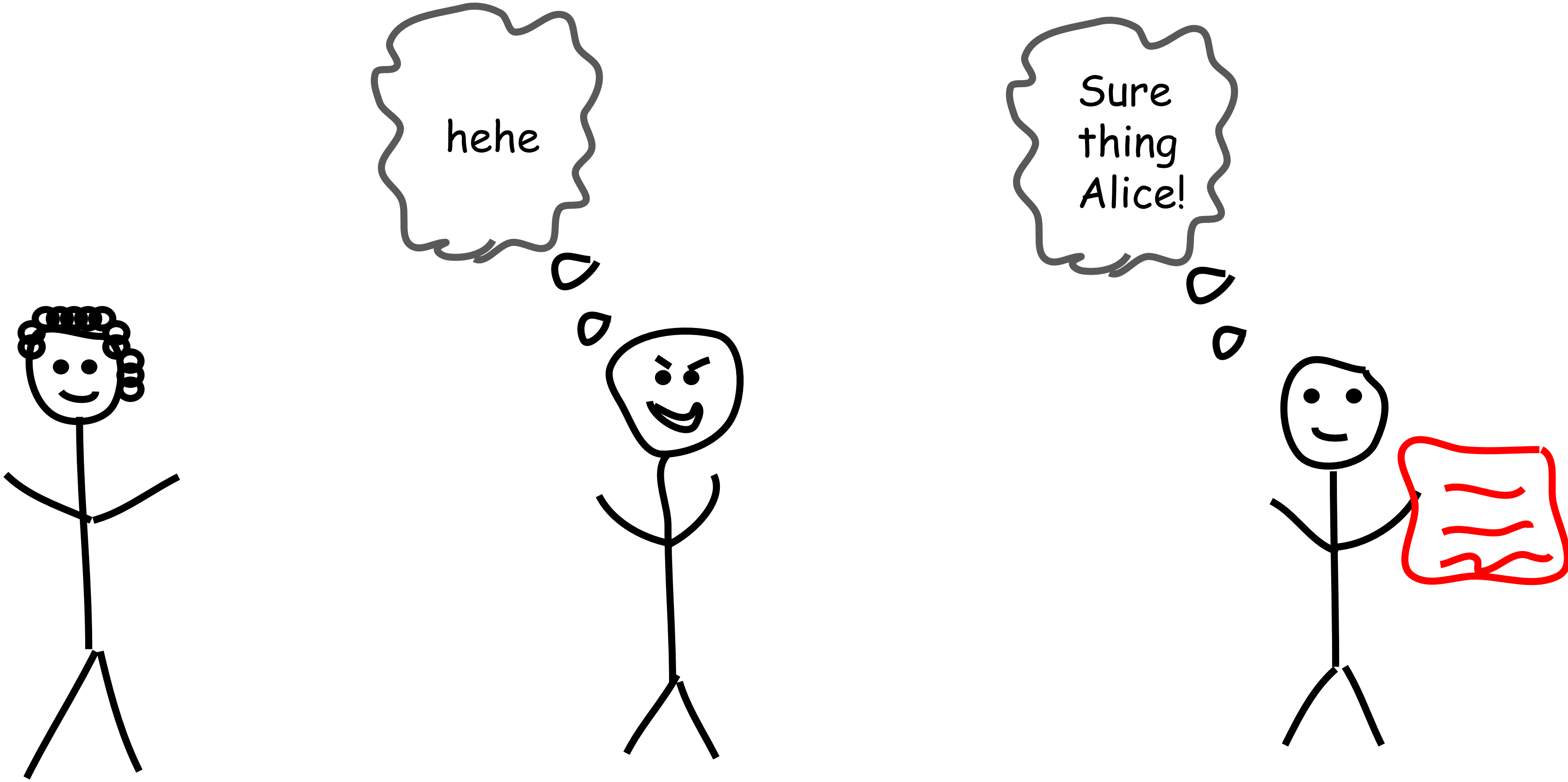
Mallory replace Alice’s message with her own



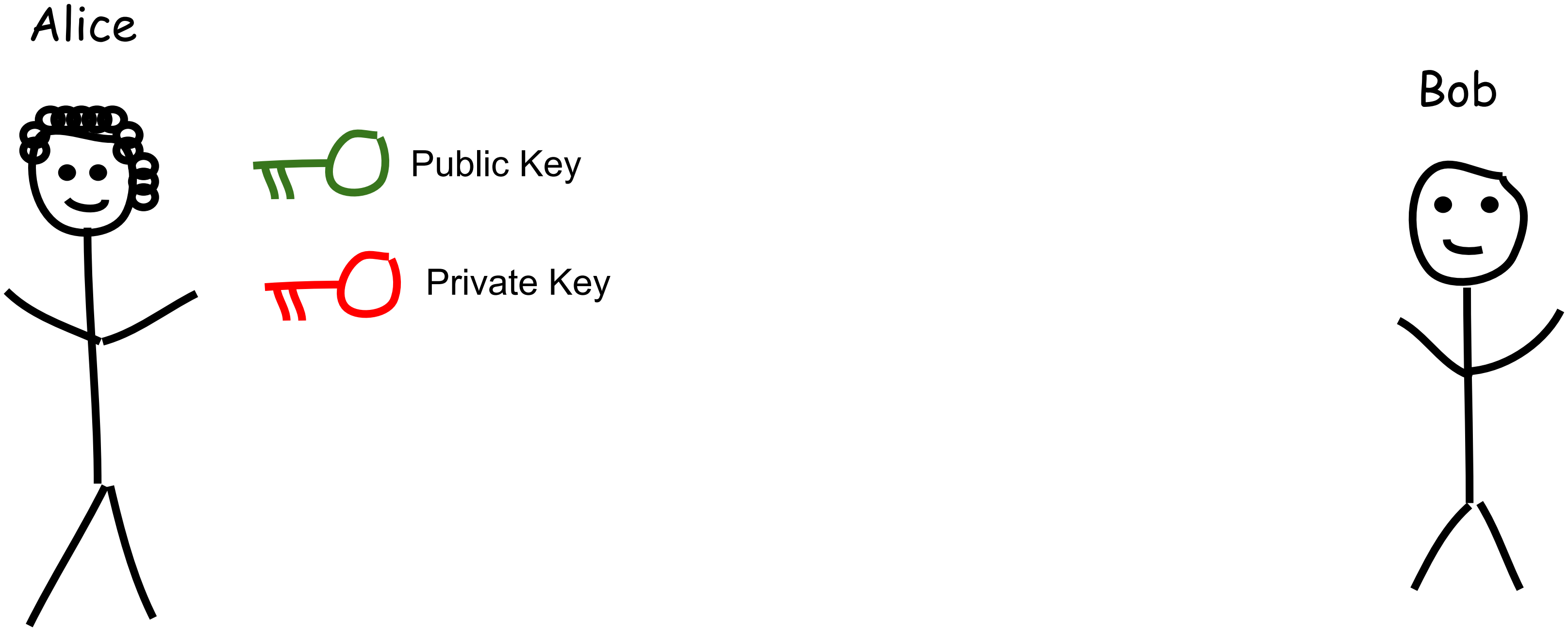
Mallory replace Alice’s message with her own



Pay rent to
account
#CE870B07


hehe
Sure thing Alice!

Public Private Key Cryptography

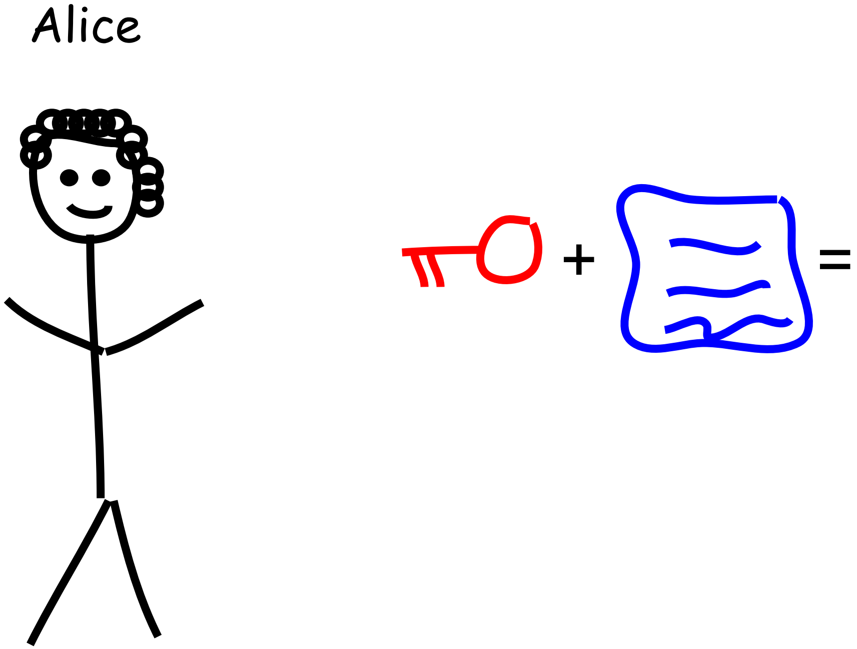


Public Key

Private Key

Public Private Key Cryptography

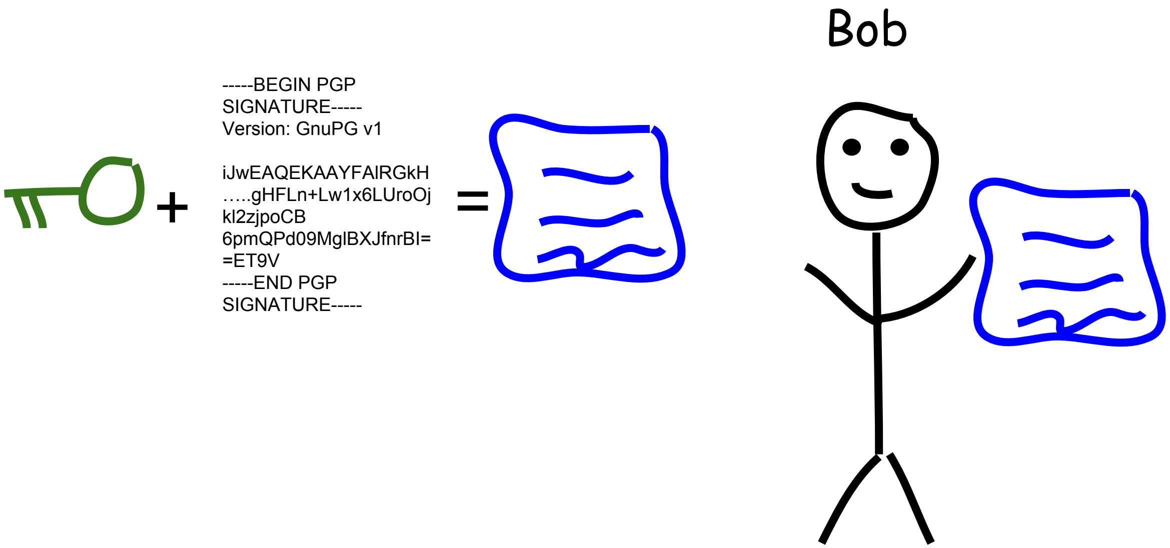
Public Private Key Cryptography



```
-----BEGIN PGP
SIGNATURE-----
Version: GnuPG v1

iJwEAQEKAAYFAlRGkH
…..gHFLn+Lw1x6LUroOj
kl2zjpoCB
6pmQPd09MglBXJfnrBI=
=ET9V
-----END PGP
SIGNATURE-----
```

Use the public key to verify the message



```
-----BEGIN PGP
SIGNATURE-----
Version: GnuPG v1

iJwEAQEKAAYFAlRGkH
…..gHFLn+Lw1x6LUroOj
kl2zjpoCB
6pmQPd09MgIBXJfnrBI=
=ET9V
-----END PGP
SIGNATURE-----
```

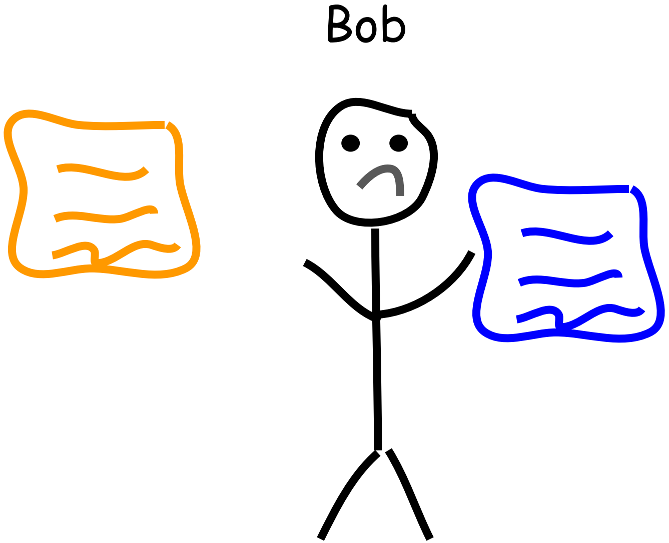
Different public key *will not* verify the message

+

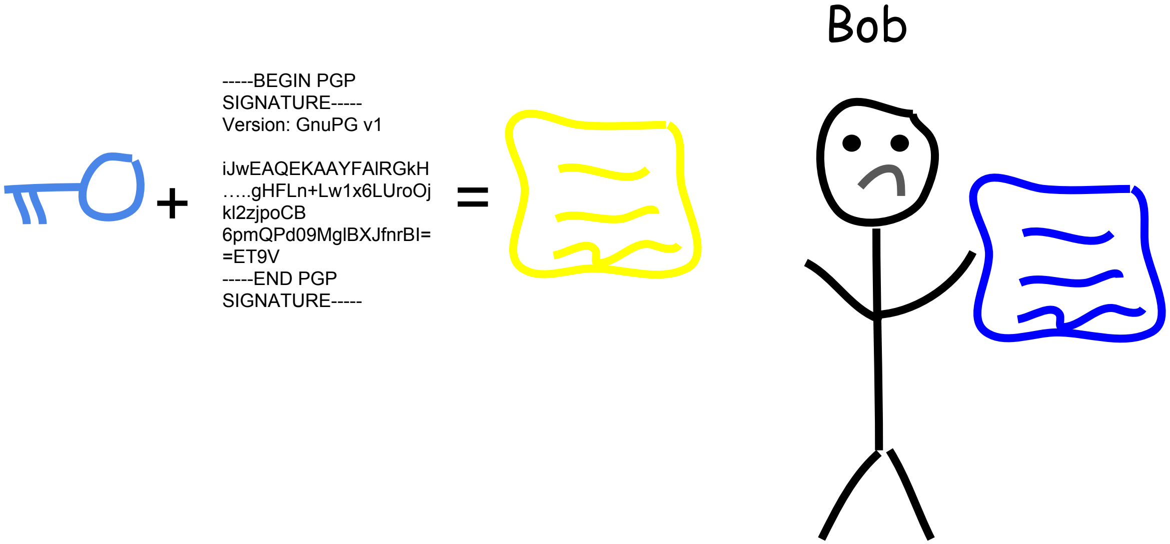
```
-----BEGIN PGP PUBLIC
KEY BLOCK-----
Version: 2.6.i

mQCNAi+UeBsAAAEEA
MP0kXU75GQdzwwlMiw
….kYboAFx
xHg43Cnj60OeZG2PKp/k
U91ipOJP1cs8/xYOGkeo
AMqDfwPeFlkBiA==
=ddBN
-----END PGP PUBLIC
KEY BLOCK-----
```

=



Different public key *will not* verify the message



```
-----BEGIN PGP
SIGNATURE-----
Version: GnuPG v1

iJwEAQEKAAYFAlRGkH
…..gHFLn+Lw1x6LUroOj
kl2zjpoCB
6pmQPd09MglBXJfnrBl=
=ET9V
-----END PGP
SIGNATURE-----
```

Sending a Transaction

- OK, we’ve now got our own coin
- And we’ve got an address (1D3mnTriicrjdcKhucHm6CAfqy7gNfGcyt)

Sending a Transaction

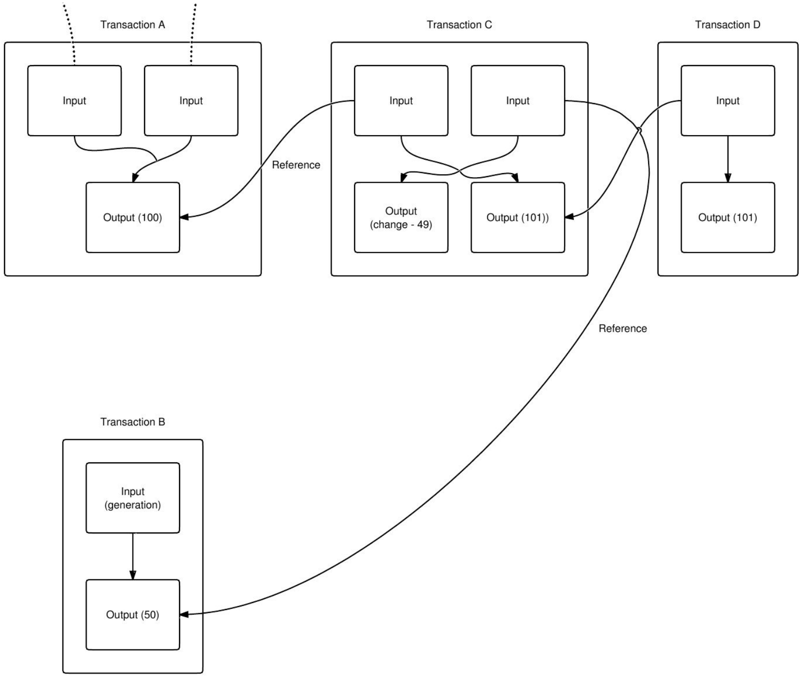
- OK, we’ve now got our own coin
- And we’ve got an address (1D3mnTriicrjdcKhucHm6CAfqy7gNfGcyt)
- What does a valid transaction actually look like?

Valid Transaction Structure

- Transaction Hash (ID)
    - 0fecf9c3b408f87d5fce986e06c78215ea0e1d869568e5517c789174c3a997dd
- Version (4 bytes, usually equal to 1)
- List of Inputs (coins you’re spending from previous transactions)
- List of Outputs (receiving address + amount)

Inputs

- Kind of like buying something with different coins from your wallet
- Each input references the output from a previous transaction
- Contains a **signature** corresponding to the previous output's public key
- Can be multiple inputs per transaction



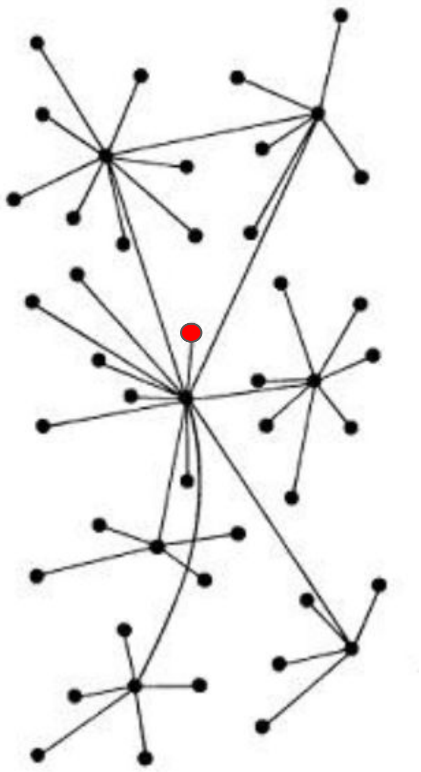
Outputs

- Much simpler than inputs, simply contains the address to send to, and the amount to send (in satoshis, 1e8 satoshis in 1 BTC
- There can be multiple outputs per transactions (batched transactions)

Address:
1D3mnTriicrjdcKhucHm6CAfqy7gNfGcyt

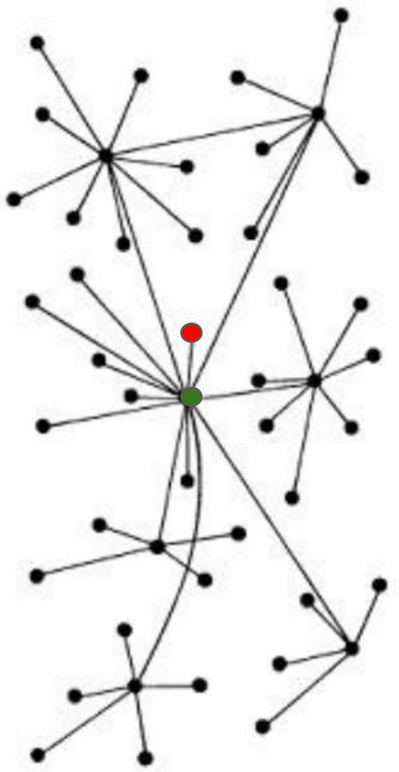
Amount: 10,000,000 satoshi

Transaction Broadcasting (Gossip Protocol)



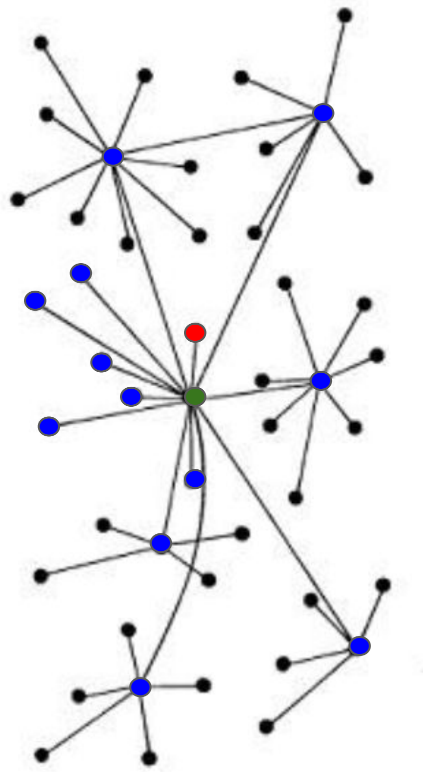
1st
Broadcast

Transaction Broadcasting (Gossip Protocol)



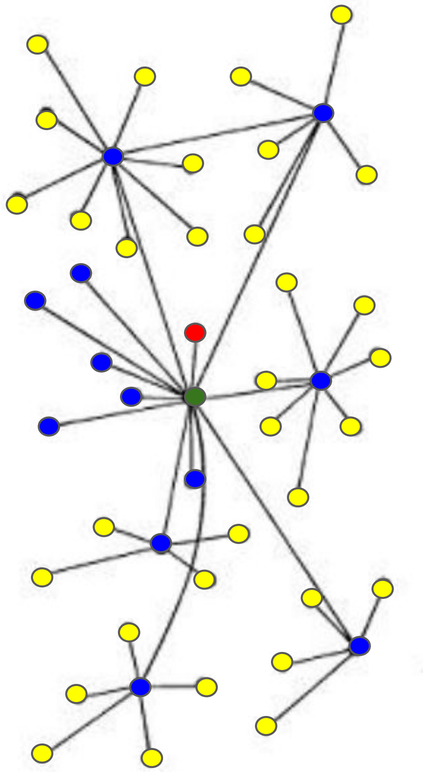
2nd
Broadcast

Transaction Broadcasting (Gossip Protocol)



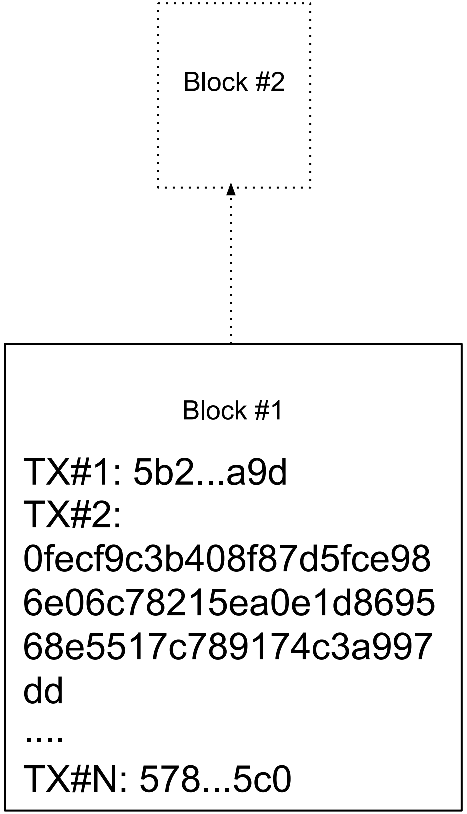
3rd
Broadcast

Transaction Broadcasting (Gossip Protocol)



4th
Broadcast

Confirmed Transaction (we’re done!)



Block #2

Block #1

TX#1: 5b2...a9d
TX#2:
0fecf9c3b408f87d5fce98
6e06c78215ea0e1d8695
68e5517c789174c3a997
dd
....
TX#N: 578...5c0

Thanks!

Questions?