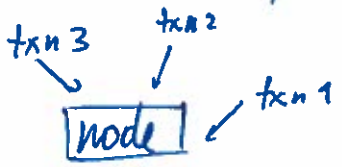
(F) Proof of work: (1) Make computationally costly for network users to validate txns

(2) Reward them for helping to validate txns (incentive)



Block B

(1) Check txn_i valid

(2) Solve crypto puzzle (proof of work)

h = sha-256 hashing fn.

Find nonce s.t. $h(\text{Block}) \leq \text{target}$

or, think of as # of leading zeroes = target

i.e., get $h(\text{Block}) =$ 000...0 3af 42fc... (000...0 = target)

Difficult to find nonce but easy to check.

size of space of hashes to explore → [16x16x16...]

Validation process = "Mining"

Reward (generating BT)

Until ~2140

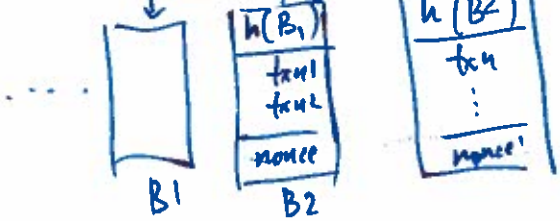
After 2140 txn fees as incentive (specified by txn party)

Chance of finding nonce ~ Real Compute power => Sybils ineffective!

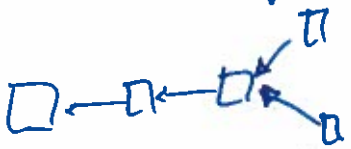
(random)

(G) Missing: ordering of txn ($txn_1 \leq_{time} txn_2$)?

Include hash of previous block



(H) But, can create a fork in this chain



Rule: (1) Only work to extend longest fork

(2) Keep track of all forks

txn not "confirmed"

unless (1) It's part of longest chain

(2) It has 5 blocks follow it ("6 confirmations")

Key Note: Difficult to create a fork + recompute all of txns.

(Requires controlling >50% of compute

BitCoin: Satoshi Nakamoto / Digital currency

Key ideas:
- proof of work
- blockchain
- P2P transactions ledger

Key Challenges:
- Double spending] Proof of work
- Trust in the network] P2P
- Incentives

(A) Alice → Bob
msg
"Tim giving Bob 1 BitCoin"

(A) forgeable ⇒ Sign the message with private key from Alice
easy to verify
$Sign_A(msg)$
difficult to forge

(B) A → B
$S_k(msg)$

(B) replay attack ⇒ give each BTcn identity w/ unique # + BANK central
B could claim 10 BT

Trusted source of serial #s + verify legitimacy

(C)
Bank
1. get 1 BT
2. S44...
3. $S_k(msg)$ BT S44
4. check S44.. Alice's + not spent
5. ACK
A → B

(C) Centralized
Puts all trust and control in one entity
⇒ (P2P) Make everyone the Bank

(D) Have everyone keep track of who owns which BT.
i.e., shared ledger (public) with all txns.
Block Chain

Double Spend by A

C
network
2. OK
3. check
4. accept
$S_k(msg)$
A → B
1. $S_k(msg)$
4. Accept

(D) Distributed, but ∃ race condition w/ Double spending
⇒ 2PC like verification of txn. (Stateful network)
Can be detected later, but BAD to have

(E)
network
txn txn
OK OK
txn
tx commit
A → B
OK

(E) ⇒ Proof of Work
Big problem: A could still double spend by redrilling work & return

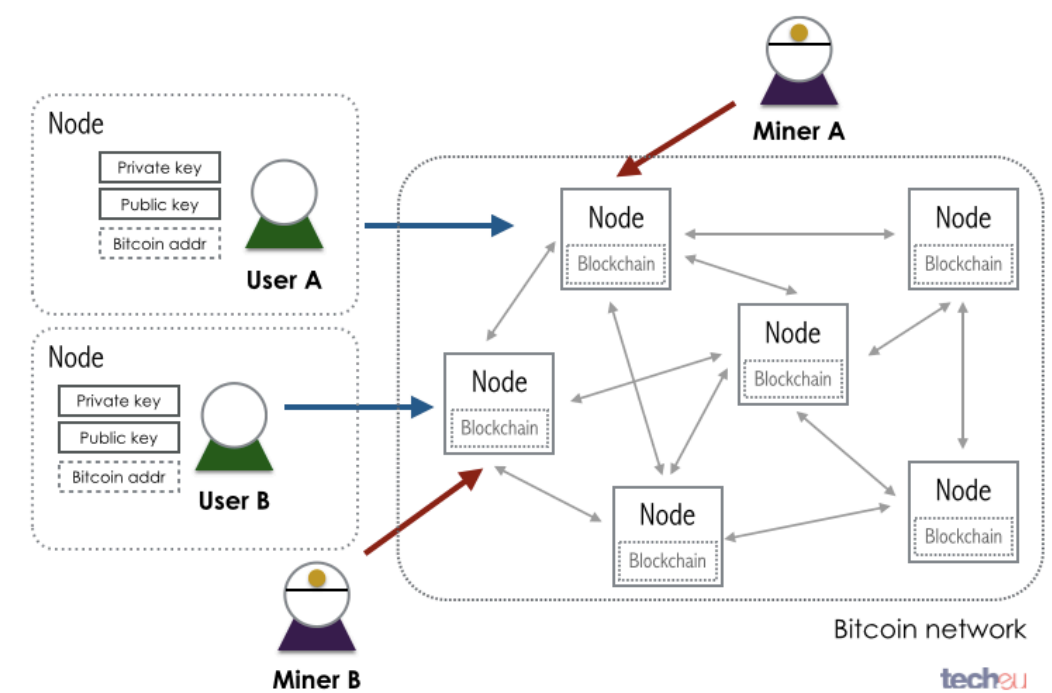
sock puppet = sybil attack

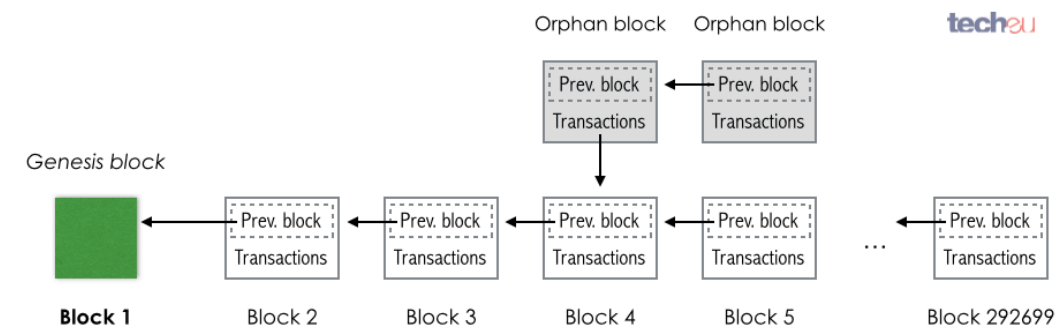
More details: * Multiple inputs/outputs / All inputs "spent", w/ change given to outputs
* joint ledger of txns

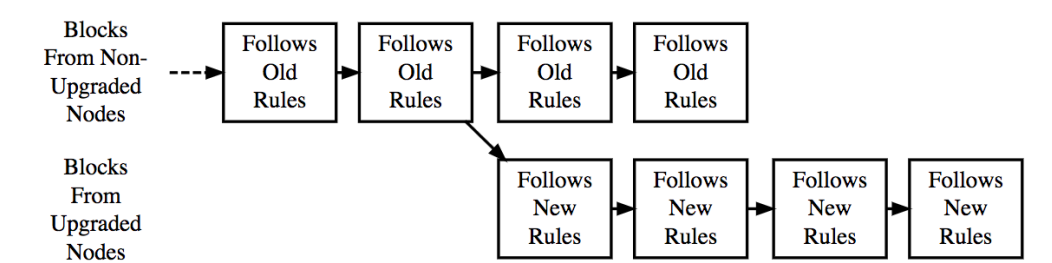
Following Chain of txns

Going Backwards ① → * Genesis block: no inputs, 50 BT output
② → Coinbase txn: Reward to a miner

* Merkle tree data structure compact representation
* Network — join/leave/protocols
* BT Scripting Lang: each txn has a script







A Hard Fork: Non-Upgraded Nodes Reject The New Rules, Diverging The Chain