Bitcoin

Funzionamento generale:

- scelta di un leader tramite una challenge crittografica di difficoltà variabile;
- il leader crea e inserisce la nuova pagina;
- il resto della rete verifica la correttezza;
- tutti i nodi aggiornano il libro contabile, aggiungendo una transazione per ricompensare il leader (insieme a una commissione sulle transazioni).

Blockchain: lista di blocchi che compone il libro contabile.

Struttura dei blocchi

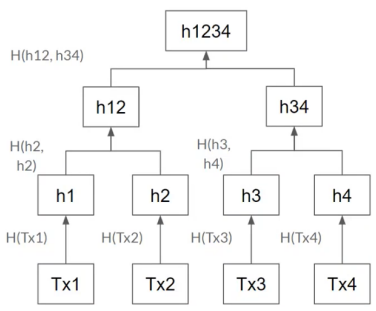
Header:

hash del blocco precedente

timestamp

nonce soluzione della challenge;

Merkle root per la verifica dell'integrità:



> Calcolato dal leader e verificato (ricalcolando indipendentemente) dal resto della rete nella fase di accettazione.

Corpo:

- coinbase transaction (premio per il leader);
- transazioni.

Proof of work

I nodi variano il nonce in modo che SHA-256 dell'header sia minore di un valore fissato dalla rete (più è basso, più è difficile). La difficoltà è calibrata in modo da avere approssimativamente un blocco ogni 10 minuti.

Modificare una transazione

Sarebbe necessario:

- aggiornare la radice del Merkle tree;
- calcolare un nuovo nonce, visto che probabilmente il nuovo header non ha hash accettabile;
- modificare tutti i blocchi successivi aggiornando il campo hash del blocco precedente, e probabilmente anche il nonce.

Tuttavia:

- ricalcolare i nonce è costoso;
- il resto della rete rifiuta la versione alternativa della ledger proposta dall'attaccante perché non coincide con quella memorizzata.

Affinché l'attacco riesca è necessario che l'attaccante controlli almeno il 51% dell'hashrate.

Biforcazioni

Si possono verificare biforcazioni se due leader vengono scelti contemporaneamente. La rete si divide tra i due rami, e dopo 6 blocchi il ramo più corto viene scartato.

Transazioni

Autenticazione, integrità, non ripudio.

Ogni utente ha una coppia di chiavi asimmetriche e un indirizzo derivato dalla chiave pubblica. Per spedire dei bitcoin a B, l'utente A genera una transazione, la firma con la sua chiave privata e la spedisce all'indirizzo di B, che può verificarla con la chiave pubblica di A. Le transazioni sono implementate nel linguaggio (non Turing equivalente) Script.

L'input di una transazione da A è una transazione precedente destinata ad A, il cui valore viene diviso tra un insieme di destinatari (tra cui A stesso se rimane un "resto" – unspent transaction output). La quantità di bitcoin "posseduti" da A è quindi la somma dei valori di tutte le transazioni non ancora spese destinate ad A.

Solo chi conosce la chiave privata del destinatario di una transazione può spendere quella transazione.