sharding / p2p

Felix Lange, Péter Szilágyi

March 20, 2018

Tools We Have

- Discovery: node metadata dissemination
- Unstructured network: random topology
- Structured multicast network: swarm/pss

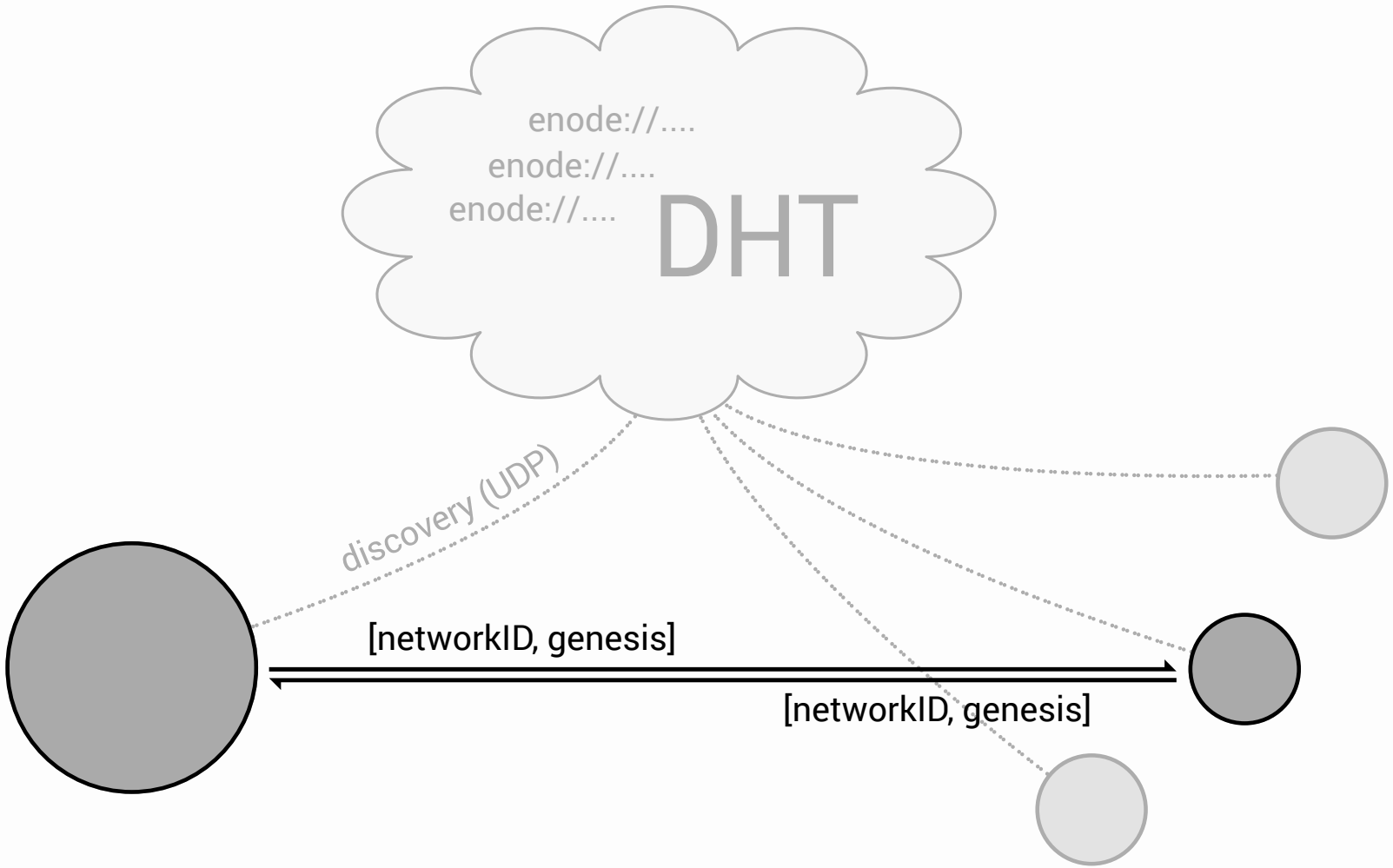
devp2p

- Part of all known Ethereum implementations
- Very few protocol changes since 2014
- Node Discovery, TCP Transport, Application Layer

eth Protocol

- Uses random topology
- Block relay, tx relay
- Downloading of headers, bodies, receipts, state

eth Protocol Handshake



Block Relay

- Propagate on valid PoW before execution
- Full blocks sent to sqrt(n)
- New block 'notification' to everyone else
- Relayed block hashes stored in capped per-peer set

Transaction Relay

- Local/remote transaction pools synced by dribbling full content of pool
- New transactions relayed to all peers
- Relayed/announced transactions stored in per-peer set

Blockchain Sync

Sync modes:

- Full sync – fetch headers, bodies and re-execute
- Fast sync – fetch headers, bodies, state
- Light sync – fetch headers upfront, rest on demand

Full Sync Challenges

From which peer? (fresh, stale, synced, forked, malicious)

Advertised total difficulty is the best we have, cannot be verified

From which block? (short reorg, large reorg, attack reorg)

Find the common ancestor, avoid chain head lies

When to swap peer? (faster network, higher difficulty)

Never, the network is untrusted (crappy peer $\gg$ attacker)

Full Sync Challenges contd.

How to max bandwidth? (slow peers, fast peers, stalling peers)

Download bodies concurrently, estimate capacity, stream priority

How to avoid header stalls? (crappy or malicious master peer)

Download skeleton from master, backfill concurrently from everyone

How to avoid leechers? (crappy remote connection vs. crappy local connection)

Enforce a dynamic an expected trip time, timeouts fail on high latency

Fast Sync Challenges

How to save on PoW? (network bandwidth $\gg$ ethash bandwidth)

Only sparsely PoW-verify the chain, later ones secure the earlier ones

How to save on Ethash generation? (network bandwidth > mobile ethash)

Pre-generate future and memory-map all Ethash caches

Fast Sync challenges contd.

How to speed up state sync? (state size $\ll$ chain size, state count $\gg$ chain count)

Fetch head state concurrently with chain, switch over to pivot

How to cater for pruning? (trie unavailable within 16 minutes)

Constantly progress header chain, keep pivot between HEAD-128..HEAD.

What's wrong with fast sync?

Each block -1000/+2000 trie nodes, thrashes the database

105+ million trie nodes, network latency kills performance

State trie size unknown in advance, nasty user experience

Sharding Network

- Sharding ideas still under heavy research
- Needs custom network topology, application protocols

(Reverse) Q/A

- Which role needs connection to which other role?
- What information is broadcasted?
- Who stays hidden?