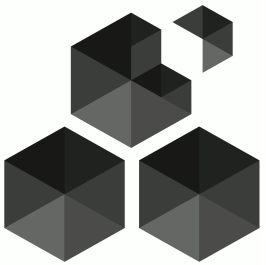
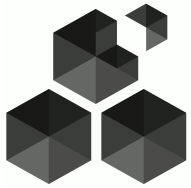
swarm: Distributed storage for Ethereum, the Turing-complete blockchain



swarm

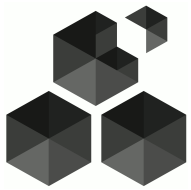
A LinuxPiter presentation by
Daniel A. Nagy



swarm: purpose

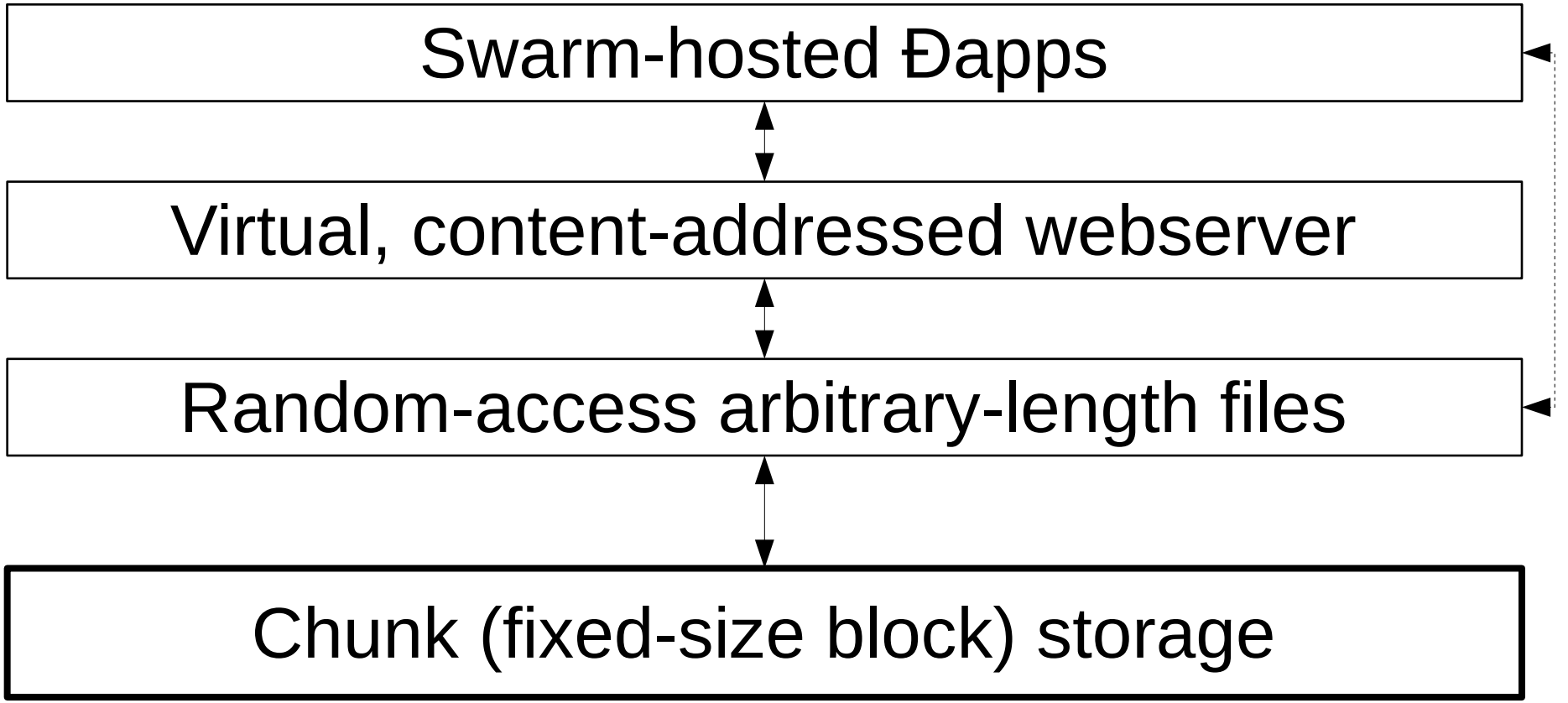
- Keeping the permanent record*
  safe and accessible
- Fair allocation of storage and bandwidth
- Reasonable redundancy
- Guaranteed integrity

* Đapp content, blockchain & state archive, contract source, NatSpec, registry indexes …



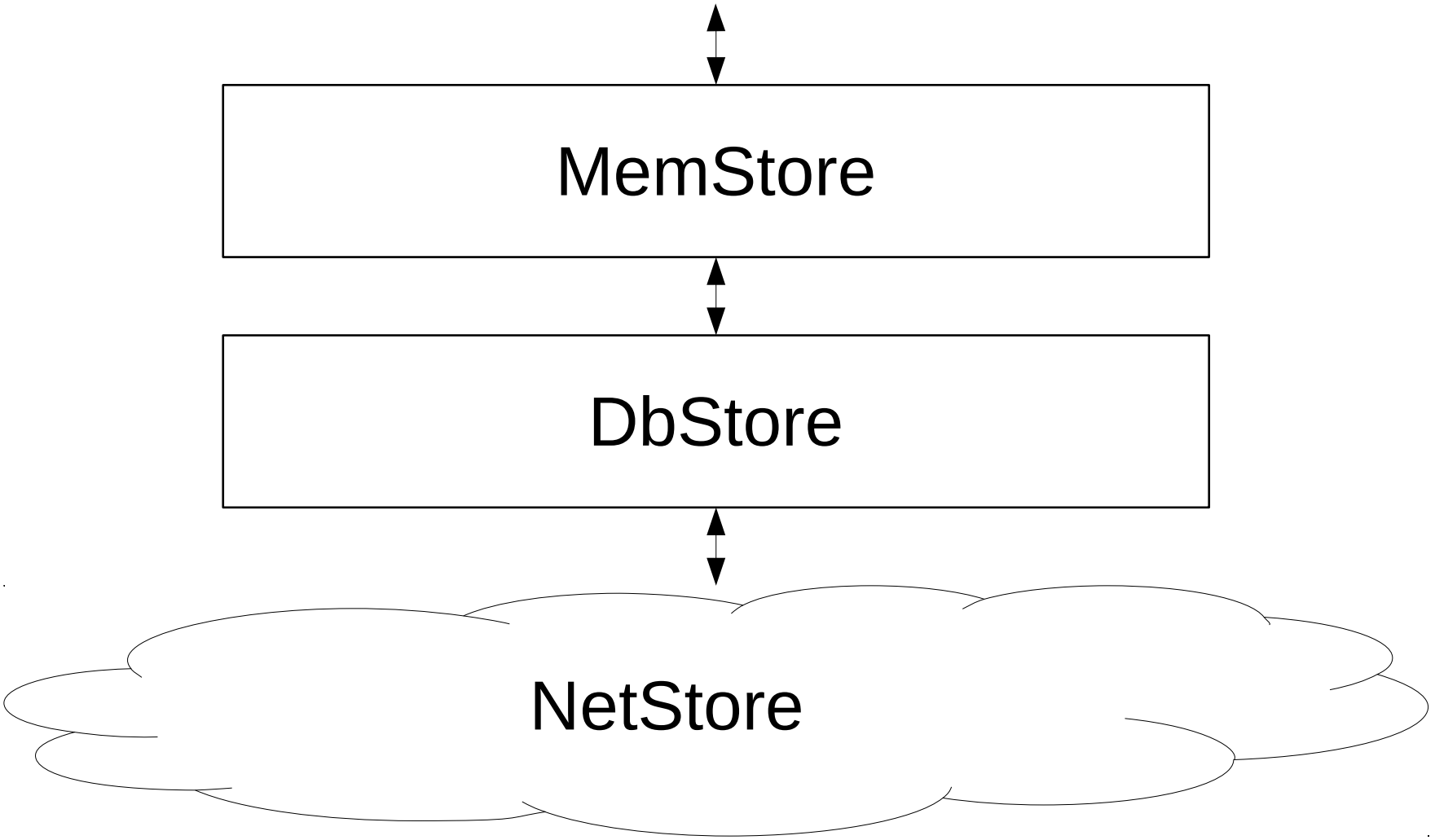
swarm: architecture

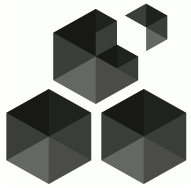
Well-separated layers connected by simple APIs:



swarm: chunk store layer

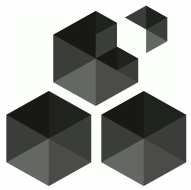
Underlying storage mechanism:





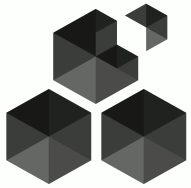
swarm: network store layer

- Ethereum devp2p multiprotocol suite
  - Reliability and security guarantees
  - Semi-permanent peerpool
- Kademlia topology and routing
  - Request forwarding
  - Smart syncronisation, content distribution
- What to store?
  - Proximity vs popularity
  - Maximum resource utilization, auto-scaling elastic cloud



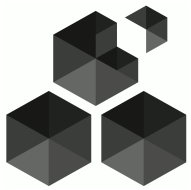
swarm: incentives

- Swarm Accounting Protocol
  - balance with service or pay
  - p2p micropayment scheme
  - Bzz: retrieval/bandwidth w chequebook
  - Strategies of withheld auto-payments
  - min risk & tx cost, max liquidity
- Availability insurance
  - (auto)litigation by vm-verifiable challenge
  - Registration & deposit



swarm: bzz and APIs

- BZZ Protocol
  - Peer forwarding protocol
  - Retrieval, request forwarding protocol
  - Syncronisation protocol
  - SWAP payment protocol
- APIs
  - JS (console, json-rpc, ipc, web3.js)
  - HTTP proxy
  - Command line tools

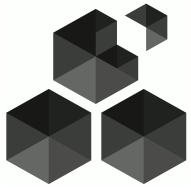


swarm: upper layers

HTTP-based API, like a locally running web server with GET, PUT, POST, DELETE methods

URL examples:

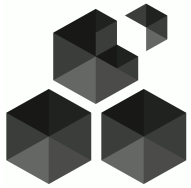
```
bzz://raw/9b4147a...9abc6c
bzz://9b4147a...9abc6c/
bzz://9b4147a...9abc6c/#4
bzz://9b4147a...9abc6c/imgs/apron.jpg
bzz://clippedwings/#4
```



swarm: upper layers

Basic Đapp example: personal photo album

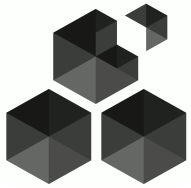
- One producer – many consumers
- No concurrent modification
- Infrequent changes
- Latency acceptable



swarm: upper layers

Complex Đapp example: distributed “dropbox”

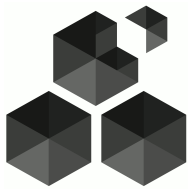
- Concurrency
- Payment and rewards
- Security
  - Confidentiality
  - Plausible denyability



swarm: upper layers

From Web 2.0 to Web3;
complex Đapp architecture

- Facebook
- Google
- Wikipedia
- OpenStreetMap



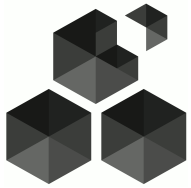
swarm: status

Project status:

- Reference implemention: swarm-capable geth client, vanilla proof of concept
- Funding, community support
- Roadmap

Source repository:

`https://github.com/ethersphere/go-ethereum/tree/bzz`



swarm: **Thank you!**

Questions, comments?