Ethereum – Blockchain with Smart Contracts

- Proposed by Vitalik Buterin in 2013 (as a white paper)
- He had been working on various forks of Bitcoin each of which added specialized capabilities
- Idea of 'Smart Contracts' had been put forward by Nick Szabo in 1993
- Designed (with others) a new blockchain that had smart contracts at its core
- Not really intended as a currency – but had a currency(ETH) within i

- Created The Ethereum Foundation – launched coin in July, 2014
- Had a 'pre-sale' of coin (1BTC buys 2,000 ETH); raised 3,700 BTC = $2.3m at the time

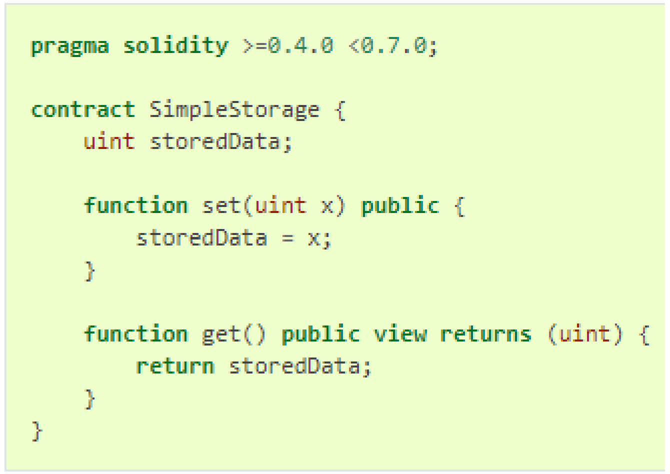
Blockchain Structure

- Ethereum moved away from the UTXO structure
- Blockchain contains a Merkle Tree based Accounts data structure along with a Transaction Tree
- Accounts can be of two types
  - Normal – or Externally Owned Accounts
  - Contract Accounts
- Transactions on Normal Accounts have an effect similar to a Bitcoin transfer
  - Alice sends 5ETC to Bob
  - Works as long asAlice's balance is >= 5ETH – otherwise fails

Bob's ETH Address	Alice's ETH Address	5ETC	Alice's Signature

Smart Contracts

- Contract Accounts are created, using a transaction, and have Code implanted into them at creation time
- The EOA that creates them is the ‘owner’
- The owner creates the contract, compiles it, bundles it into a transaction and sends this to the blockchain
- A contract account is created and the address returned
- Subsequently, any user can invoke any of the contract’s methods by sending a transaction to the contract account – [Coke Machine Analagy]



```
pragma solidity >=0.4.0 <0.7.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Making a contract

- Contracts can potentially be written in any (restricted) language but the most popular is called Solidity and was modelled on Javascript
- Contracts are compiled into Bytecode that can run on the Ethereum Virtual Machine (EVM)
- It is ‘Turing Complete’ – can implement loops, conditionals etc and each contract can establish variables in the state-tree
- You can program bugs!
- An Infinite loop (deliberately or accidentally introduced) could keep the EVM busy forever on one contract
- A contract could SPAM the blockchain with huge useless variables

Gas

- Ethereum introduce the concept of ‘Gas’ to guard against this
- Every operation on the Ethereum blockchain consumes ‘Gas’
  - Every bytecode executed in the EVM
  - Every byte of storage consumed in the state tree
- When a transaction is sent to the Blockchain, it must be accompanied by an amount of ‘Gas’ [the fee]
- If the ‘Gas’ runs out mid-way, the transaction is aborted, state rolled-back and the Gas already used is consumed by the system
- Levies a fee on blockchain resource usage – incetivizes good behaviour

Gas Price

- When a user is preparing a transactions, they include two things:
  - Gas Limit – indicates the maximum gas they are willing to pay to complete the transactions
    - Need to estimate this (there are automatic estimators)
    - If a smart contract does something unexpected, it will terminate once ‘gas limit’ is consumed
  - Gas Price – this indicates how much (in ETH) a user is willing to pay for each gas unit
    - Transaction fee = gas consumed * gas price
    - Miners will pick the transactions with the highest gas price first – can pay more for early execution
- Any excess gas (up to gas limit) is refunded
- When a transaction is aborted, the miner keeps the gas

Contracts used for many applications

- Any service that typically needs a ‘middleman’ that can be coded easily
- Betting
- Dealing in Stocks/Commodities
- Called Decentralized Apps (DAPPS)
  - Catalog of DAPPS here: https://www.stateofthedapps.com/
- One famous contract was to establish a Decentralized Autonomous Organization (DAO)
  - Contact allowed people to assemble a fund made up of donations
  - Donors would then vote to award funding to deserving causes by allocating money to a sub-fund
  - Assembled 12.7m ETH ($150m at the time) – but got hacked
  - Many others have since set up other DAOs (hopefully with no bugs)

Tokens

- Smart Contracts can be used to implement ‘Coloured Coins”
- This is standardized in Ethereum Request for Comments #20 (ERC-20)
- A promoter can create a smart contract that initializes itself with a number of ‘Coins’
- The smart contract keeps a table (Owner: amt) that allows blockchain users to own coins and to transfer them from one person to another.
- The coins (Tokens) can be used to represent real-world assets (e.g. bars of gold, shares in a company)
- Many Blockchain(and other) companies used these to launch Initial Coin Offerings (ICOs) to raise money
- https://www.bloomberg.com/news/articles/2018-12-14/crypto-s-15-biggest-icos-by-the-numbers

ERC-20

- Documents a standard contract interface for tokens

```
//core ERC20 functions
function allowance(address _owner, address _spender) constant returns (uint remaining);
function approve(address _spender, uint _value) returns (bool success);
function balanceOf(address _owner) constant returns (uint balance);
function totalSupply() constant returns (uint totalSupply);
function transfer(address _to, uint _value) returns (bool success);
function transferFrom(address _from, address _to, uint _value) returns (bool success);
```

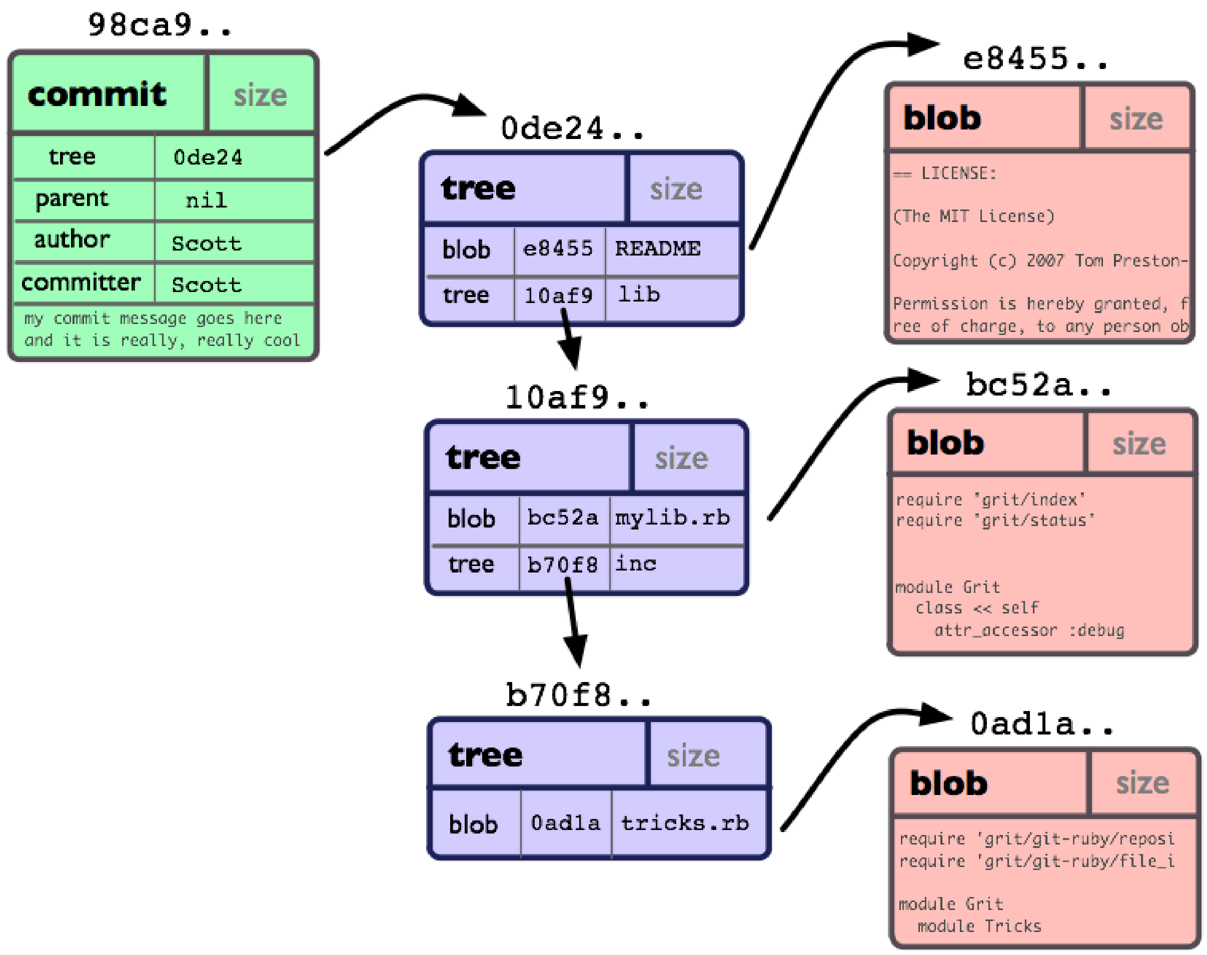
- Totalsupply is initialized when contract is created
- Users can “Transfer” tokens once they have them or
- Allow another user (Broker) to transfer to others on their behalf using “Approve”
- Most Ethereum Wallets supports the transfer of ERC-20 tokens natively

Decentralized Infrastructure – The Interplanetary File System (IPFS)

- A platform for decentralized applications proposed by Juan Benet and Protocol labs (Y-combinator startup)
- Launched Alpha in Feb 2015
- Brings together many different innovations from Tor, S/Kademila and other systems
- Described in a white paper:
- IPFS - Content Addressed, Versioned, P2P File System (DRAFT 3)
- https://ipfs.io/ipfs/QmR7GSQM93Cx5eAg6a6yRzNde1FQv7uL6X1o4k7zrJa3LX/ipfs.draft3.pdf

Inter-Planetary Linked Data (IPLD)

- IPLD allows arbitrary data to be assembled in a linked data structure
- Drew much of its inspiration from Git – the source code control system developed for Linux by Linus Torvals in 2005
- Git wrapped every ‘changed’ file in a ‘Blob’ – identified by its hash
- Gathered these into Trees
- Managed versioning using ‘commit’s



IPLD Data Structures

- IPLD is a common hash-chain format for distributed data structures

```
type IPFSLink struct { Name string // name or alias of this link
                       Hash Multihash // cryptographic hash of target
                       Size int // total size of target }

type IPFSObject struct { links []IPFSLink // array of links
                         data []byte // opaque content data }
```

- These IPFS Objects are intended to be distributed over a content-addressed P2P network – could be used to represent:
  - Files – text, audio,video
  - People
  - Web pages
  - Bindings
  - Arbitrary content
- Since they are identified by their hash – they are immutable – any change means a new object
- An object is only stored once – identical objects are automatically de-duplicated

Files in IPLD

- In order to help represent files and files systems in IPLD – they defined:
- Blob

```
{ "data": "some data here", // blobs have no links }
```

- List

```
{ "data": ["blob", "list", "blob"], // lists have an array of object types as data
  "links": [ { "hash": "XLYkgq61DYaQ8NhkcqyU7rLcnSa7dSHQ16x", "size": 189458 },
             { "hash": "XLHBNmRQ5sJJrdMPuu48pzeyTtRo39tNDR5", "size": 19441 },
             { "hash": "XLWVQDqxo9Km9zLyquoC9gAP8CL1gWnHZ7z", "size": 5286 } // lists have no names in lin
```

- In IPFS, Blobs can be <=256K – larger files are lists of Blobs

Directories

- Tree Object
  - Similar to list but the sub-links are named

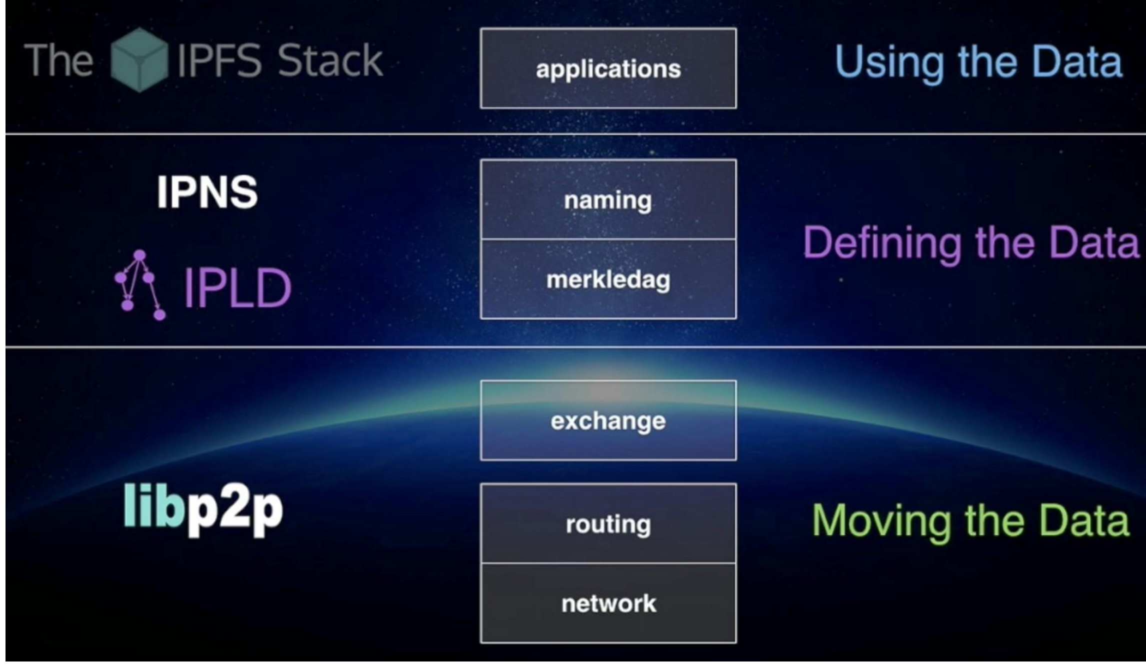
```
{ "data": ["blob", "list", "blob"], // trees have an array of object types as data
"links": [ { "hash": "XLYkgq61DYaQ8NhkcqyU7rLcnSa7dSHQ16x", "name": "less", "size": 189458 },
           { "hash": "XLHBNmRQ5sJJrdMPuu48pzeyTtRo39tNDR5", "name": "script", "size": 19441 },
           { "hash": "XLWVQDqxo9Km9zLyquoC9gAP8CL1gWnHZ7z", "name": "template", "size": 5286 } // trees do have names ]
```

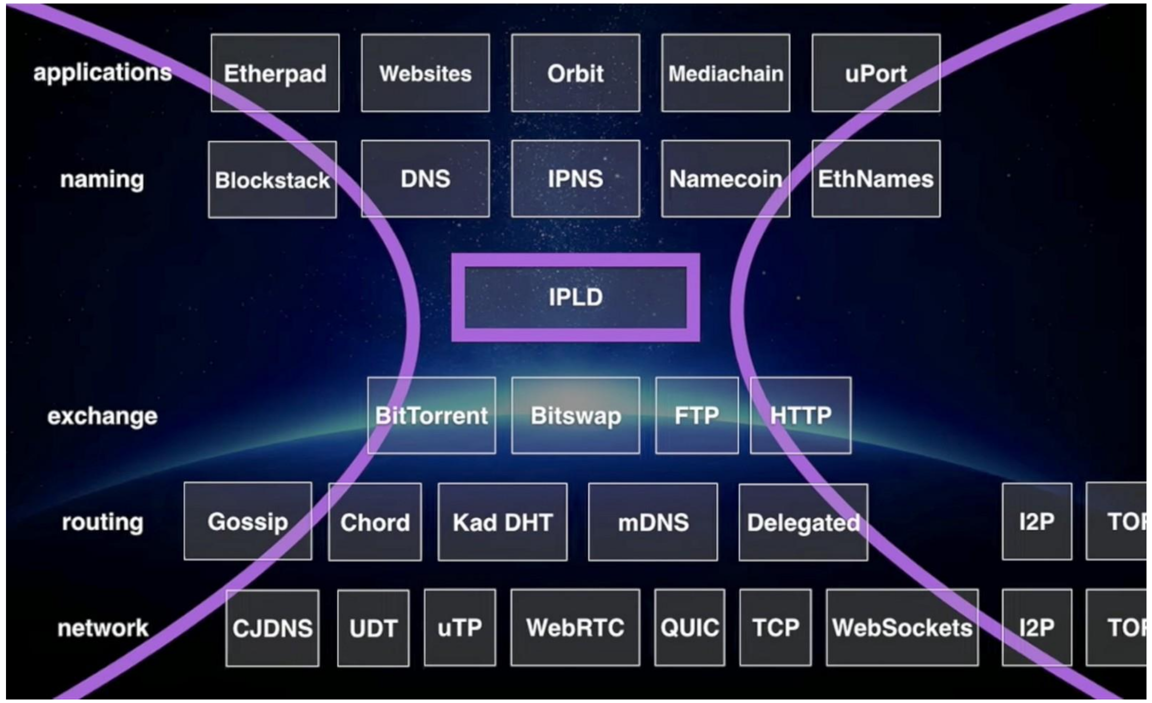
- Can traverse the tree structure using hashes or names

XLWVQDqxo9Km9zLyquoC9gAP8CL1gWnHZ7z/ XLHBNmRQ5sJJrdMPuu48pzeyTtRo39tNDR5
Or
XLWVQDqxo9Km9zLyquoC9gAP8CL1gWnHZ7z/script

IPFS Nodes

- Any node on the internet can join the IPFS network by attaching to the underlying libp2p network (Not from TCD!)
- Libp2p is a collection of modules that supports DHTs, content distribution, delivery of peer-to-peer messages and lots more





Starting an IPFS Node

- When a node initializes, it first generates a NodeID
-

```
type NodeId Multihash type Multihash []byte // self-describing cryptographic hash digest
type PublicKey []byte
type PrivateKey []byte // self-describing keys
type Node struct { NodeId NodeID
                   PubKey PublicKey
                   PriKey PrivateKey }
```

- Does a proof-of-work to generate an acceptable key
- When nodes connect, they exchange public key pairs
- Can decide if new neighbour is a ‘good neighbour’

```
difficulty = <integer parameter>      n = Node{}
 do { n.PubKey, n.PrivKey = PKI.genKeyPair()
        n.NodeId = hash(hash(n.PubKey))
        p = count_preceding_zero_bits(n.NodeId) } while (p < difficulty)
```

Initialization Example

- ipfs init
- initializing ipfs node at /Users/jbenet/.go-ipfs generating 2048-bit RSA keypair...done
- peer identity: Qmcpo2iLBikrdf1d6QU6vXuNb6P7hwrbNPW9kLAH8eG67z
- to get started, enter: ipfs cat /ipfs/QmYwAPJzv5CZsnA625s3Xf2nemtYgPpHdWEz79ojWnPbdG/readme

- ipfs daemon Initializing daemon...
- API server listening on /ip4/127.0.0.1/tcp/5001
- Gateway server listening on /ip4/127.0.0.1/tcp/8080

Local Object Store

- Usually a part of local disk, but could be memory
- Node can ‘add’ objects to the store – they immediately are added to the global tree (forest) of IPFS
    - Instantly Accessible from any other IPFS Node on the Internet

```
$ cat mytextfile.html
<h1>Hello World</h1>

$ ipfs add mytextfile.html
added QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy mytextfile.txt

$ ipfs cat QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy
<h1>Hello World</h1>
```

- Including http://ipfs.io/ipfs/ QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy

Adding directories and whole filesystems

- -w is used to add directory information – notice two objects are added

```
$ ipfs add -w mytextfile.html
added QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy mytextfile.txt
added QmPvaEQFVvuiaYzkSVUp23iHTQeEUpDaJnP8U7C3PqE57w
```

- Can access this with ipfs ls

```
$ ipfs ls -v QmPvaEQFVvuiaYzkSVUp23iHTQeEUpDaJnP8U7C3PqE57w
Hash                                           Size Name
QmZtmD2qt6fJot32nabSP3CUjicnypEBz7bHVDhPQt9aAy 29   mytextfile.html
```

- On Unix can mount IPFS as a filesystem

Content Persistence

- New items go into the local IPFS object store
- They propagate across the web (using libp2p) towards wherever there is demand
- Acts as an automatic Content Distribution Network – content goes to where it is popular
- Objects remain in local stores until they are displaced
- If an object is displaced from all stores, it is no longer accessible
- Node can choose to ‘pin’ content that they want to keep
- Nodes that are ‘add’ed are pinned recursively by default

Mutability and IPNS

- All objects are identified by their hash and are therefore immutable (and automatically de-duplicated)
- If objects (or entire trees) are replaced – need a way to get the latest version of something
- Each IPFS node is assigned a NodeID
- The IP Name Service provides a single writable value associated with this at /ipns/<NodeID>
- The value is self-certified – i.e. it is signed by a public key that hashes to <NodeID>

IPNS Example

- Publish a hash to my NodeID (PeerID)

```
ipfs name publish QmNUhKfcGJyQJnZu3AKn8NoiomDwDCRBicgqPt1YRqJBCz
```

- Returns Published to <NodeID>

```
Published to QmYmmfn68vkcFDeZz1NTZyEXTixjjUnUS6UaPdMSsUBWxs:
 /ipfs/QmNUhKfcGJyQJnZu3AKn8NoiomDwDCRBicgqPt1YRqJBCz
```

- Can now use the NodeID to refer to the published value

  https://ipfs.io/ipns/QmYmmfn68vkcFDeZz1NTZyEXTixjjUnUS6UaPdMSsUBWxs

- The hash that is published might point to a “Commit” node so that the older(displaced) content still be found

Filecoin Dropbox – without Dropbox

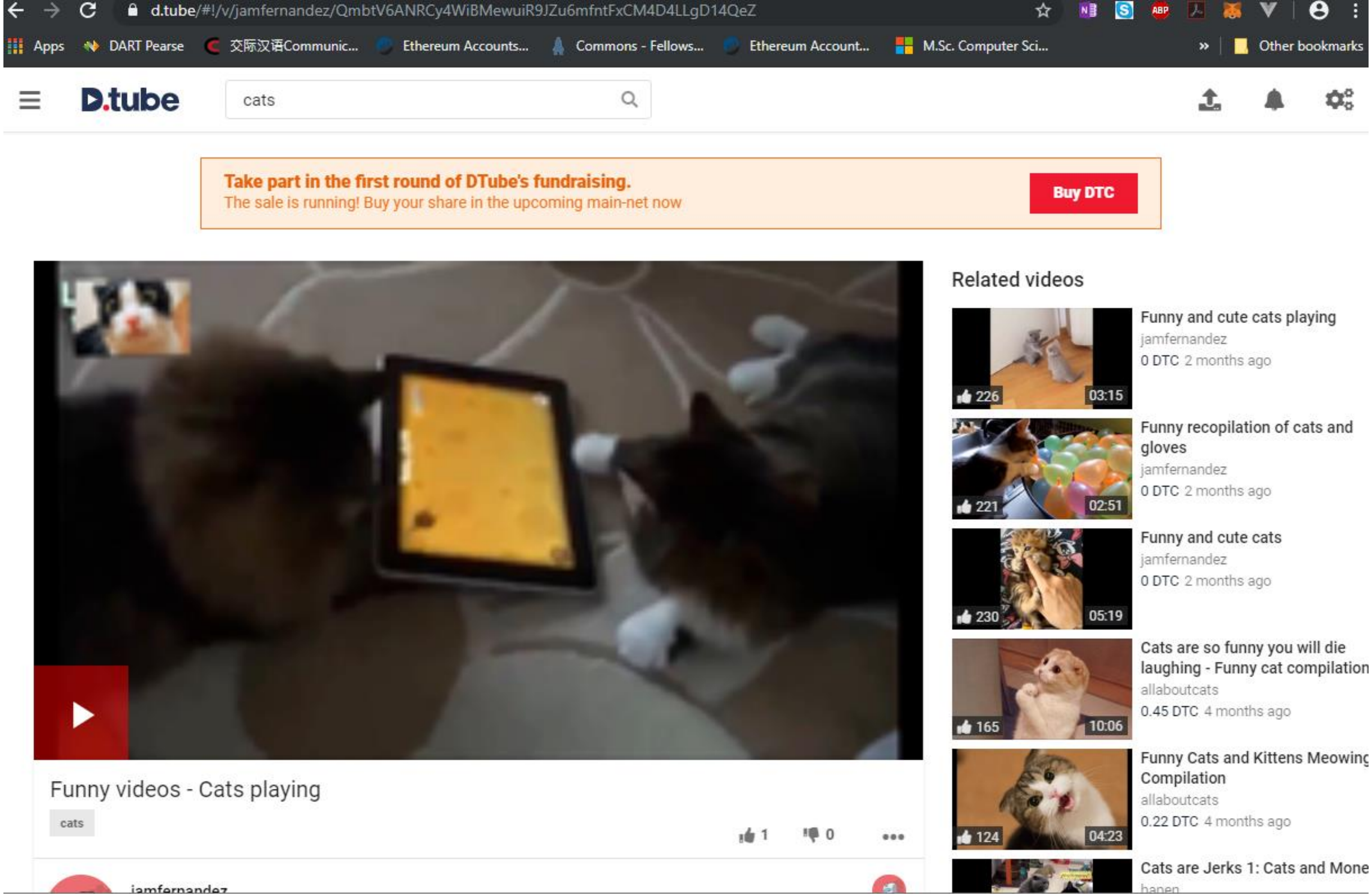
- IPFS is not good for long term storage – no incentive for others to store your content long term
- Filecoin.io – project financed by ICO ($257m) in 2017
- Three types of users:
    - Clients – who want to store and retrieve information
    - Storage Miners – who will store stuff for a fee
    - Retrieval Miners – who get stuff (from clients or storage miners) for a fee
    - Filecoin implements a storage market to match people backed by a suite of smart contracts and a dedicated crypto-Token
- Complex scheme – ~~now in Alpha~~ Launched October 2020
- Good video from 2017 (pre-ICO) of filecoin – high level objectives and concerns
- https://www.youtube.com/watch?v=e02czCnCuCM&list=PLYX7WMBv1JLrH8Uqmoix4k_ahoeEeCfYQ&index=2

Open Index Protocol – Search without Google

- Users who have information to publish will do so through IPFS
- They assemble metadata on it using the Open Index Protocol and write this into a transaction on the FLO blockchain – The ‘Public Space’
- ‘Curators’ scan the FLO blockchain and create websites (possibly ad-driven) that showcase some subset of the published information
- Alexandria.io – covers everything indexed by the Open Index Protocol
- Other Curatators: Caltech Electron Tomography Database, Medici Land Governance project

Examples of Decentralized Systems : Dtube - Youtube without Youtube

- Youtube hosts vast amounts of user video content – funded by ads
- In return for providing ad-targets, Youtube hosts, classifies, controls the connection between users and that content
- D.tube is a decentralized alternative
    - Uses IPFS to host the video content
    - Claims that entire website runs in the browser and accesses IPFS content
    - Uses the Steemit Cryptocurrency to reward content producers – Whitepaper (June 2019) suggests that new currency will be the DTC
    - Signing up creates an entry on the DTC blockchain initialized with some currency


d.tube/#!/v/jamfernandez/QmbtV6ANRCy4WiBMewuiR9JZu6mfntFxCM4D4LLgD14QeZ

Apps DART Pearse 交际汉语Communic... Ethereum Accounts... Commons - Fellows... Ethereum Account... M.Sc. Computer Sci... Other bookmarks

D.tube cats

Take part in the first round of DTube's fundraising.
The sale is running! Buy your share in the upcoming main-net now
Buy DTC

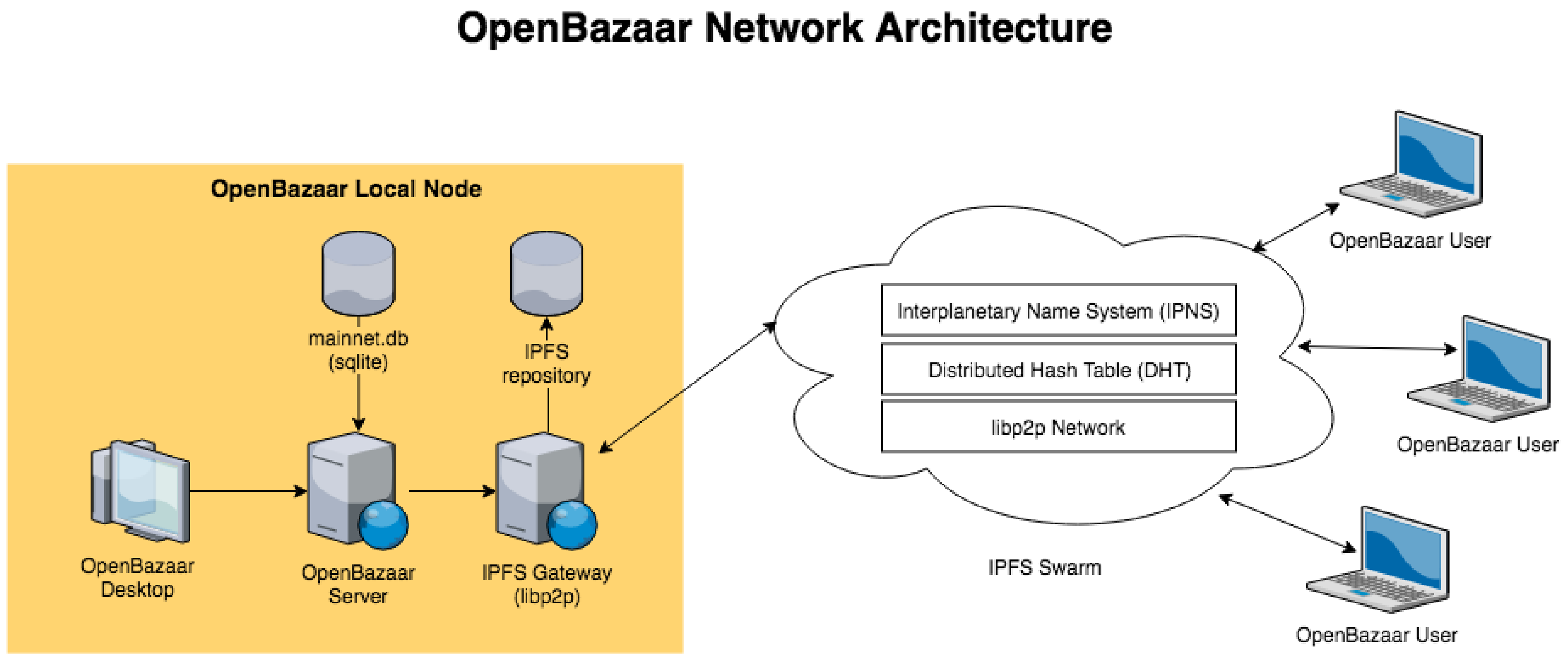
Funny videos - Cats playing
cats
1 0
jamfernandez

Related videos

226 03:15	Funny and cute cats playing jamfernandez 0 DTC 2 months ago
221 02:51	Funny recopilation of cats and gloves jamfernandez 0 DTC 2 months ago
230 05:19	Funny and cute cats jamfernandez 0 DTC 2 months ago
165 10:06	Cats are so funny you will die laughing - Funny cat compilation allaboutcats 0.45 DTC 4 months ago
124 04:23	Funny Cats and Kittens Meowing Compilation allaboutcats 0.22 DTC 4 months ago
	Cats are Jerks 1: Cats and Mone banen

Decentralized E-Commerce – OpenBazaar Amazon without Amazon

- Originally called DarkMarket and developed as a successor to The Silk Road (a marketplace on Tor selling mostly illegal items)
- Got taken up by other developers and renamed OpenBazaar



Openbazaar…

- All users have dedicated Client/Server software which includes an IPFS stack
- Uses IPFS objects and libp2p DHT entries to disseminate offers of goods for sale and orders
- Uses Bitcoin for payments – mutisignature and third parties (any user) as escrow agents

Decentralized Social Media

- Tim Berners-Lee led a project called Social Linked Data (Solid) at MIT
- Slow progress but now being taken forward by Inrupt – an Australian startup
- Each user stores their personal data in a solid ‘Pod’ and then gives other users access to limited portions of this
- Linked Data is a key component of this
- Most effort has gone into syntax of Pods, links and query languages

Other Social Media Projects

- Akasha – uses linked data on IPFS
- Sapien.network – uses dedicated blockchain – claims to use IPFS
- Mastodon – twitter clone built on ActivityPub protocol

Centralized versus Decentralized

Centralized

- Focal Point
- Filestore/Database
- Identity System
- Compute Power
- Profit Motive for FB, Google
- Legal/Governmental Control

Decentralized

The Internet + P2P + DHT

Decentralized storage (IPFS, Filecoin)

- Blockchain or Crypto Identity
- DeC Compute Power
- Incentives in CryptoCurrency
- Community control

Course Outline

- What is an Internet Application and how have these evolved?
- Key Technologies: Javascript, Node Package Managers
- Execution Environment: Client Side (Browser) and Server
- Using Node.JS, NPM and support tools
- Cloud Computing Architectures –SaaS, IaaS, PaaS, Serverless Computing
- Web Frameworks: Angular, React and Vue – The Model-View-Controller paradigm
- A simple Cloud-based Internet Application
- Database Services
- Load Balancing
- Scaling & Monitoring
- Introduction to Containers
- Serverless Computing
- Characteristics & Enabling Technologies for Web 3.0

Thank You for your Attention!