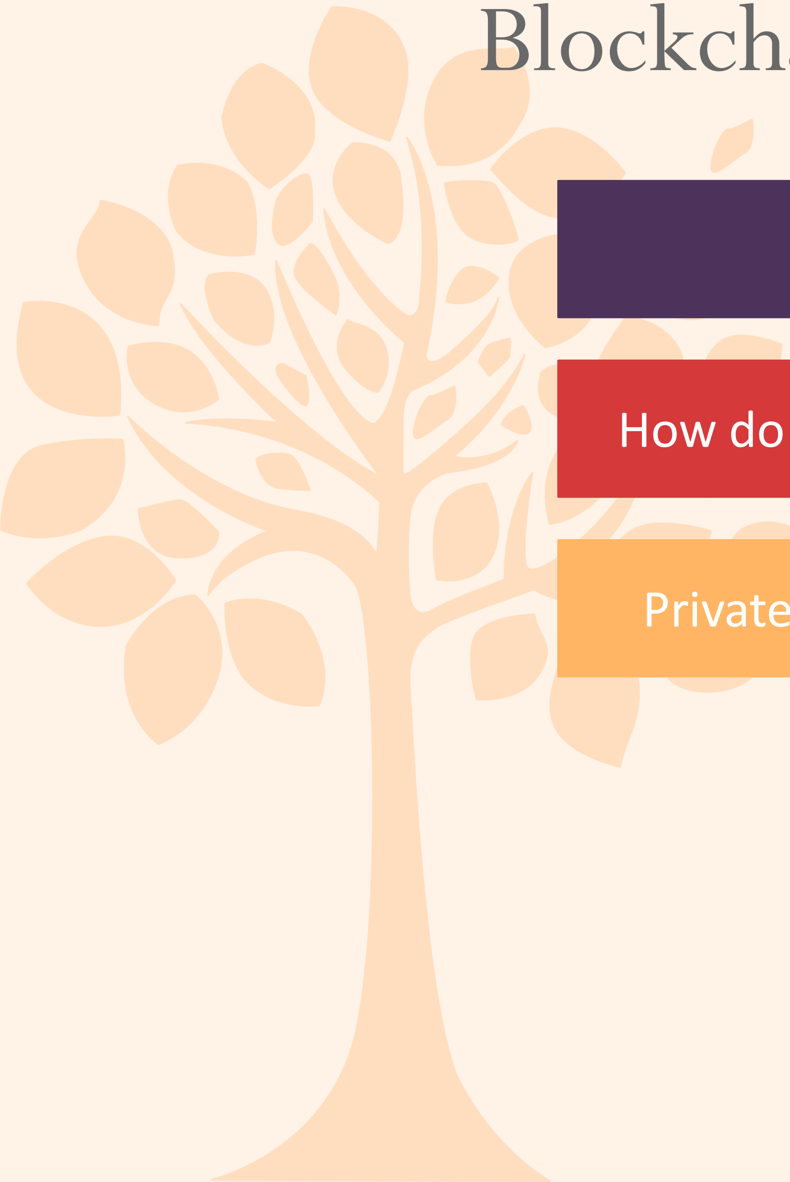
Blockchain Demystified



Why the Need 01

How do some Blockchains work? 02

Private versus Public networks 03

A source of misunderstandings…

Who is the public? In what sense is this a ledger?

“The block chain provides Bitcoin’s public ledger, an ordered and timestamped record of transactions. This system is used to protect against double spending and modification of previous transaction records.” – Bitcoin.org

Does this conform to our normal intuitions of time and stamps?

“Protected” is not synonymous with “Guaranteed” – recall Capital Guaranteed vs Capital Protected products

A more general definition

“the term is used to describe a process of adding blocks of cryptographically signed data to form **perpetual and immutable** records”

– Oliver Wyman

Decoding Blockchain Buzzwords

Distributed consensus

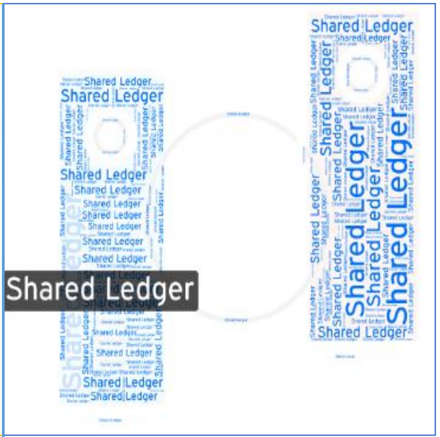
- A fault tolerant way for multiple computers to maintain consistency on some data

Non-repudiation

- Using cryptography to verify identity and secure transmissions – so transactions are tamper-resistant and not deniable

Smart contracts

- Stored logic to automate and limit one’s actions after agreement is reached



Shared ledger

- All parties see the same information

The Need

“A distributed system is one in which the failure of a computer you didn’t even know existed can render your own computer unusable” – Leslie Lamport*

*Email message sent to a DEC SRC bulletin board at 12:23:29 PDT on 28 May 87.

Distributed Systems have issues

Halting nodes

- Nodes stop, nodes go into infinite loops

Network fragility

- Connections break

Omission

- Messages get lost

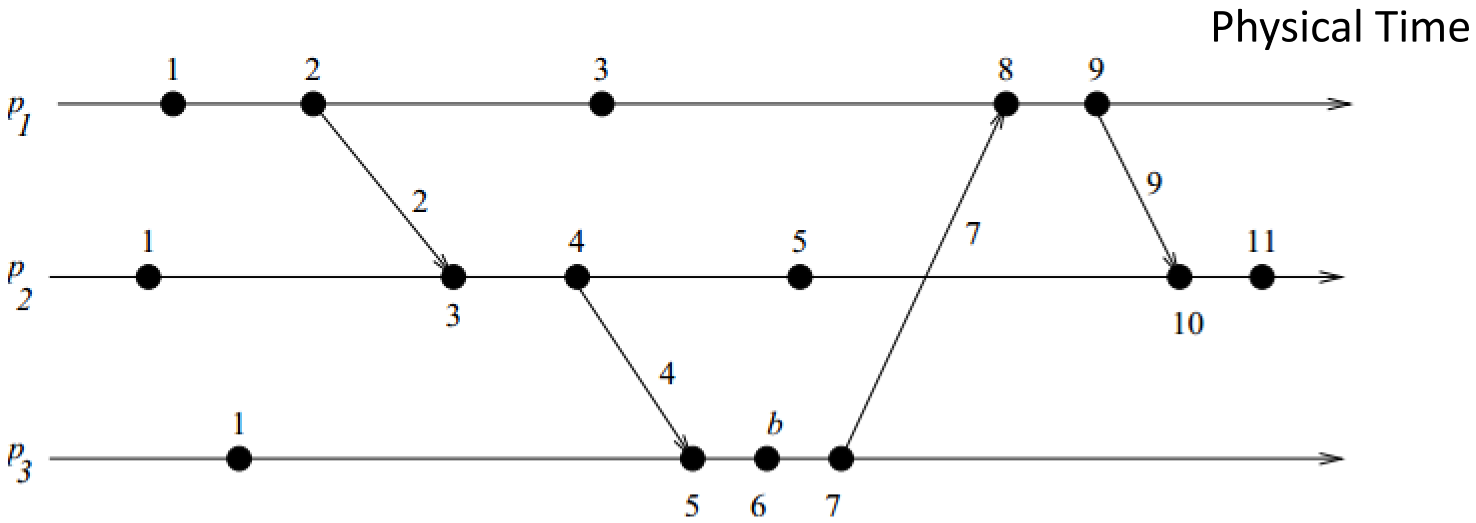
Timing failures

- Clock skew

Byzantine failures

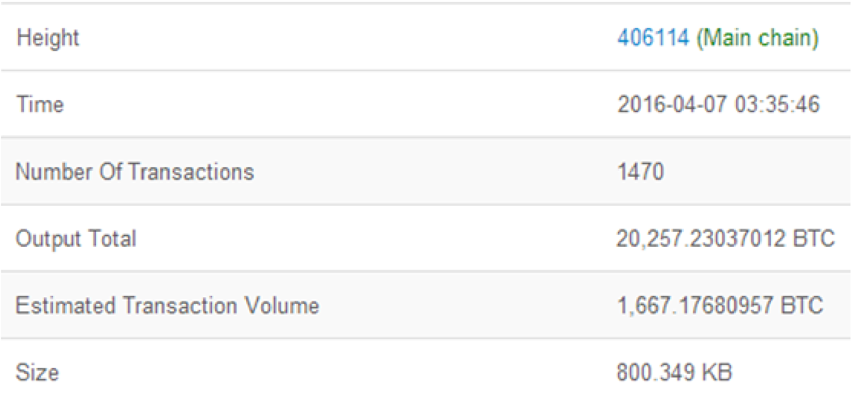
- Arbitrary corruption

Simplest consensus problem: Can we agree on the time?

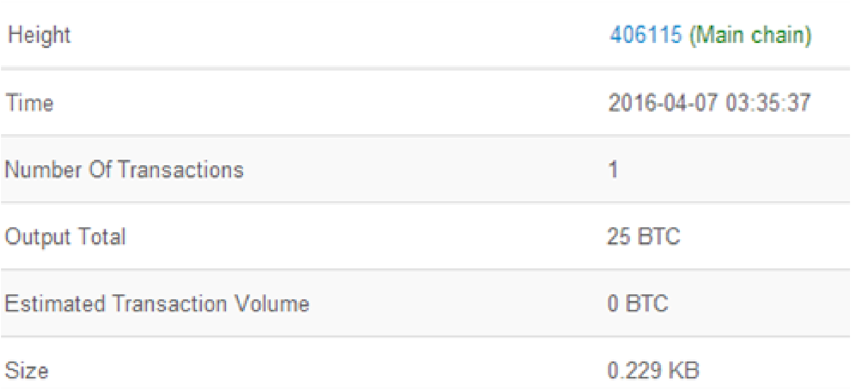


- If all the parties are updating a common resource…
  - P_1 sends a message to P_2 when its clock strikes 2
  - P_2 receives the message when its clock strikes 3
  - To keep time consistent, the recipient of messages adjusts its clock such that
    [time of receipt > latest time stamp on received messages]
- The time each party sees on its own physical clock is different and they are none the wiser

Logical versus Physical Time



Height	406114 (Main chain)
Time	2016-04-07 03:35:46
Number Of Transactions	1470
Output Total	20,257.23037012 BTC
Estimated Transaction Volume	1,667.17680957 BTC
Size	800.349 KB



Height	406115 (Main chain)
Time	2016-04-07 03:35:37
Number Of Transactions	1
Output Total	25 BTC
Estimated Transaction Volume	0 BTC
Size	0.229 KB

- Real life implications:
  - Block 406114 is time stamped 03:35:46
  - The following Block 406115 is time stamped 03:35:37
- Blocks of transactions do not enter the record in the order they are time stamped

Brewer’s CAP Theorem & Wedding Analogy

Consistency

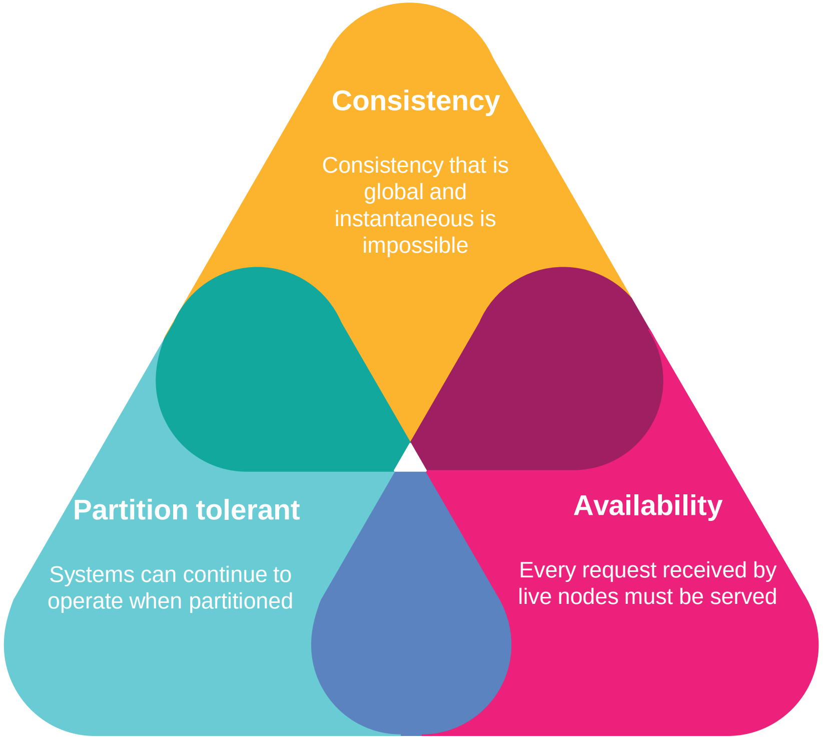
All clients see the same view even in the presence of updates. Requires that total ordering exists and updates appear atomic

Availability

All clients can find some replica of data even in presence of failure

Partition Tolerance

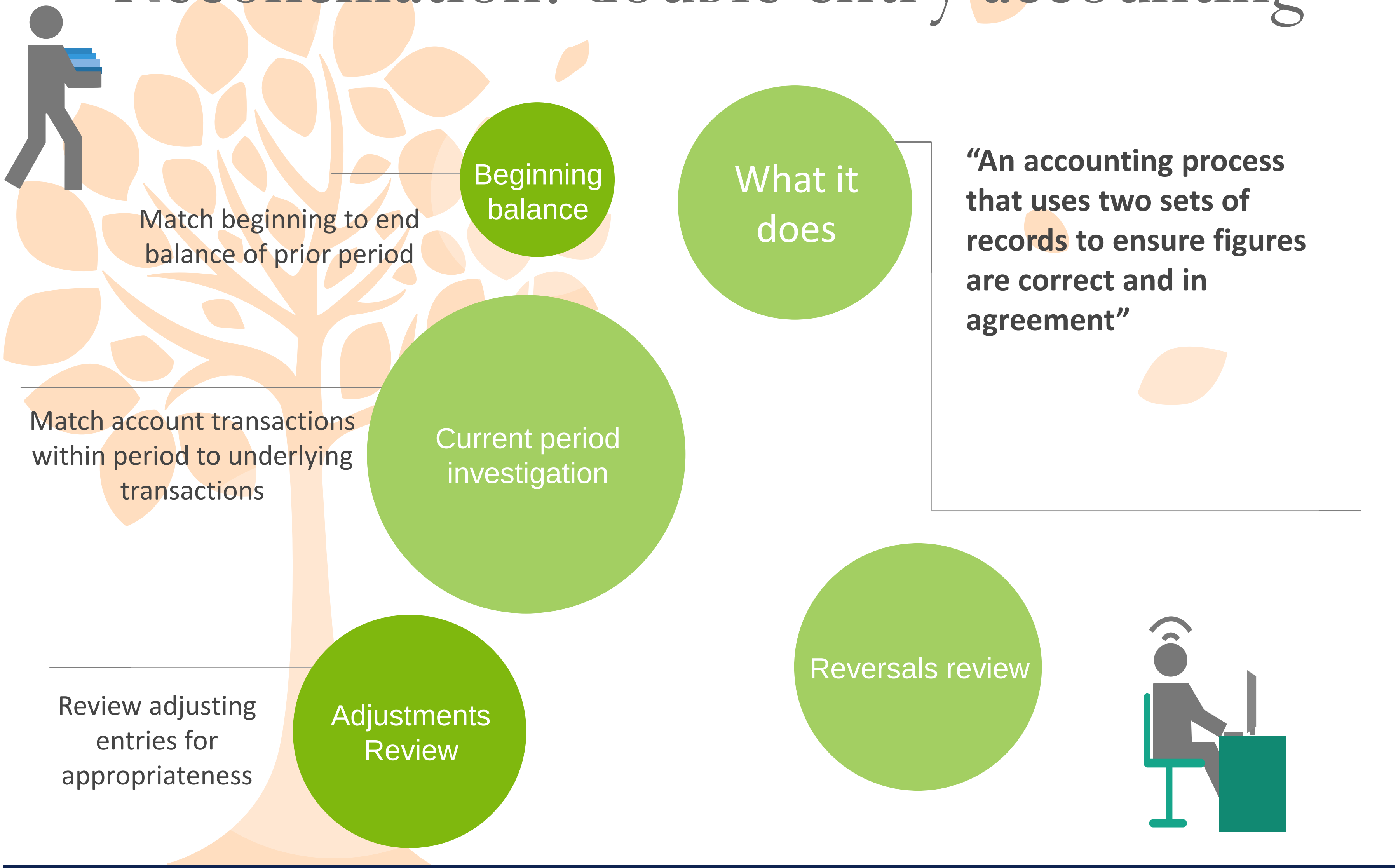
If the network stops delivering messages between two sets of servers, will it work correctly?



Consistency
Consistency that is global and instantaneous is impossible
Partition tolerant
Systems can continue to operate when partitioned
Availability
Every request received by live nodes must be served

If we allow the network to drop messages, then one has to choose to either allow updates to both sides of the partition (for availability) and lose consistency, or shut the system down until the errors are resolved to prioritize consistency

Reconciliation: double entry accounting



What it does

“An accounting process that uses two sets of records to ensure figures are correct and in agreement”

Beginning balance

Match beginning to end balance of prior period

Current period investigation

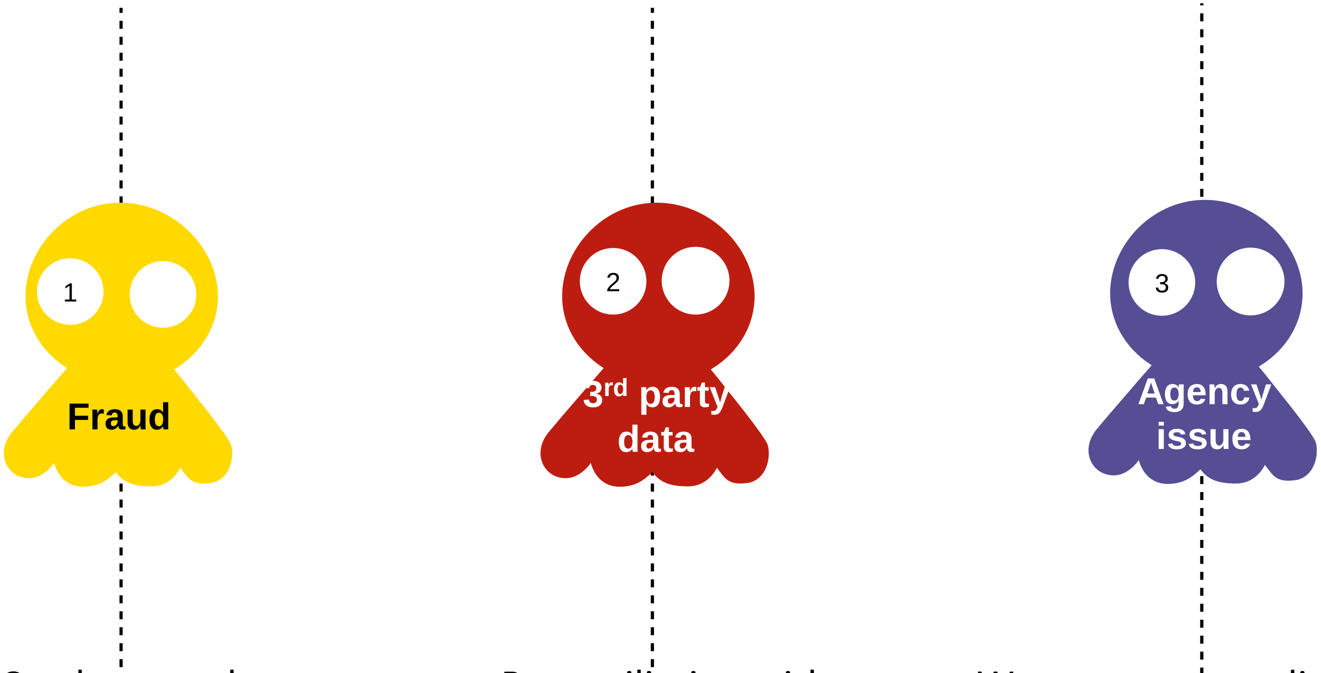
Match account transactions within period to underlying transactions

Adjustments Review

Review adjusting entries for appropriateness

Reversals review

Double-Entry accounting’s problems



1 **Fraud**

Costly controls needed to prevent tampering with internal ledgers

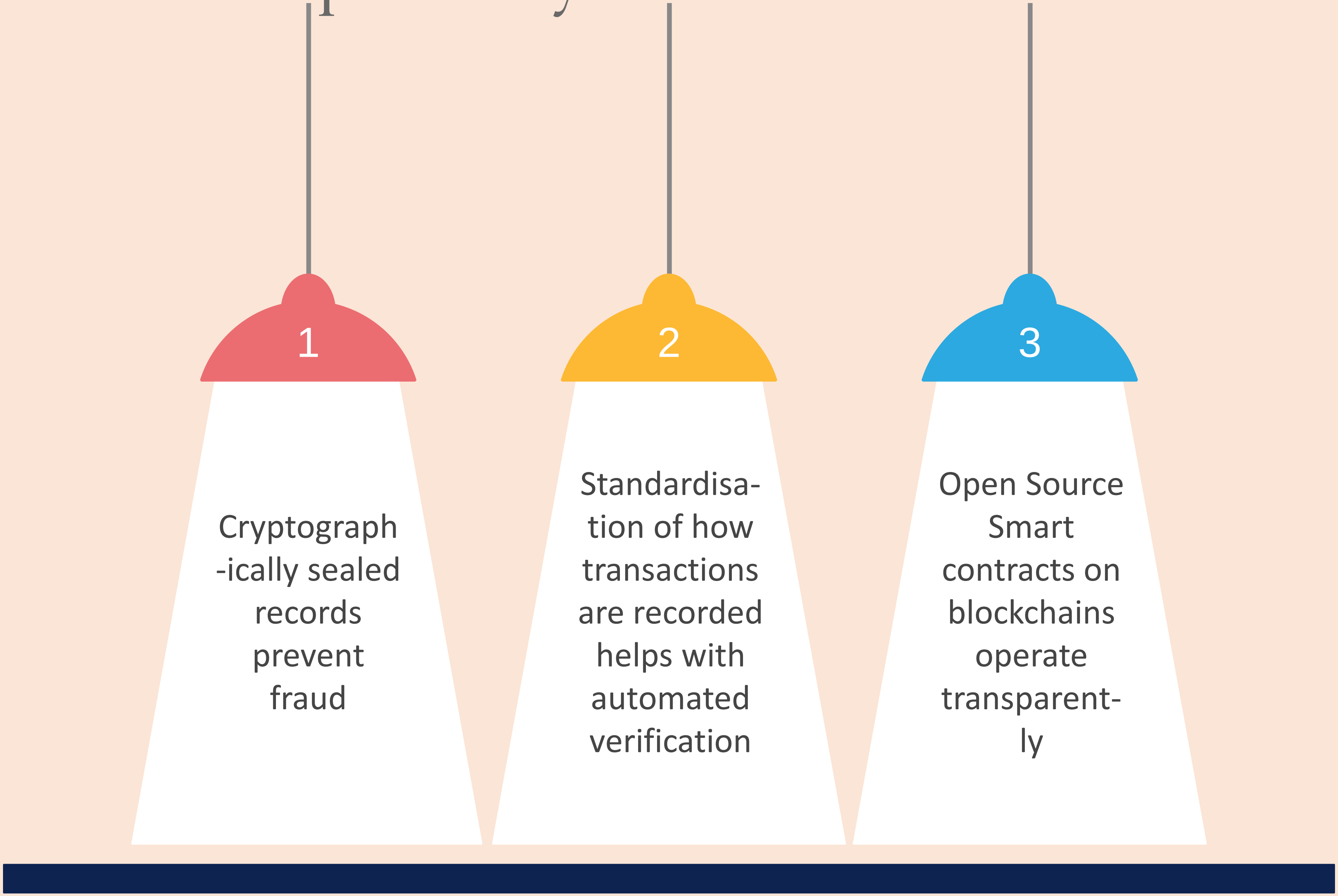
2 **3rd party data**

Reconciliation with third party data such as bank accounts, cards is time consuming

3 **Agency issue**

We must trust auditors who are hired and paid by managers to check on the same management’s integrity

How Triple Entry works with Blockchains



1. Cryptographically sealed records prevent fraud

2. Standardisation of how transactions are recorded helps with automated verification

3. Open Source Smart contracts on blockchains operate transparently

Triple Entry Accounting*

Alice

Debit	Credit
5	
	2
	9
10	

Bob

Debit	Credit
5	
	2
	9
10	

Public

Debit	Credit
-5	5
2	-2
9	-9
-10	10

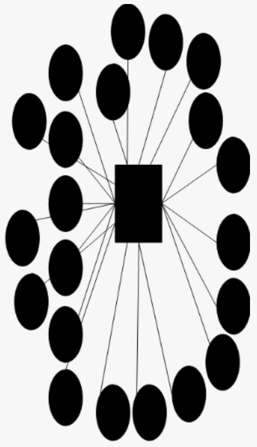
What if the Public were not just a third party notary, but a large set of non-colluding third parties?

*Example thanks to http://villagemall-ceo.blogspot.sg/2015/06/triple-entry-accounting-and-block-chain.html

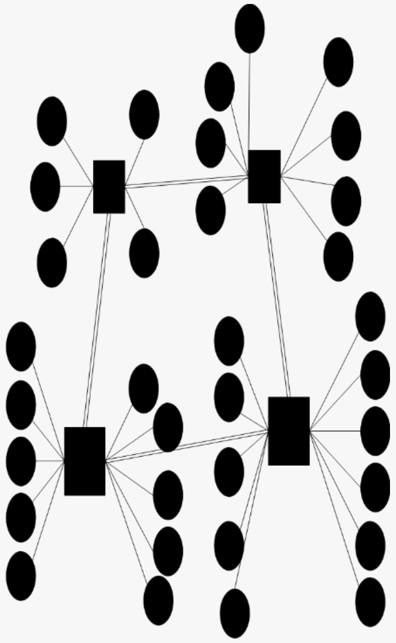
Decentralized and Distributed – dispersal of risk

network

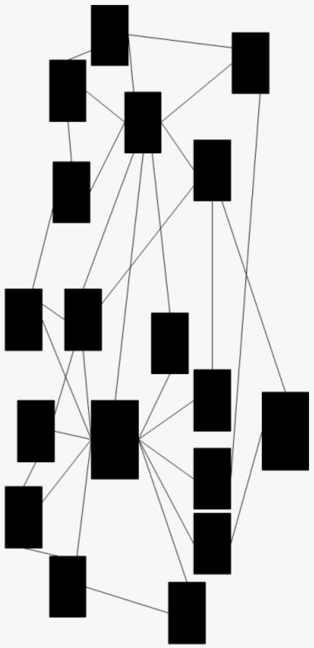
Centralized
(Bicycle Wheel)



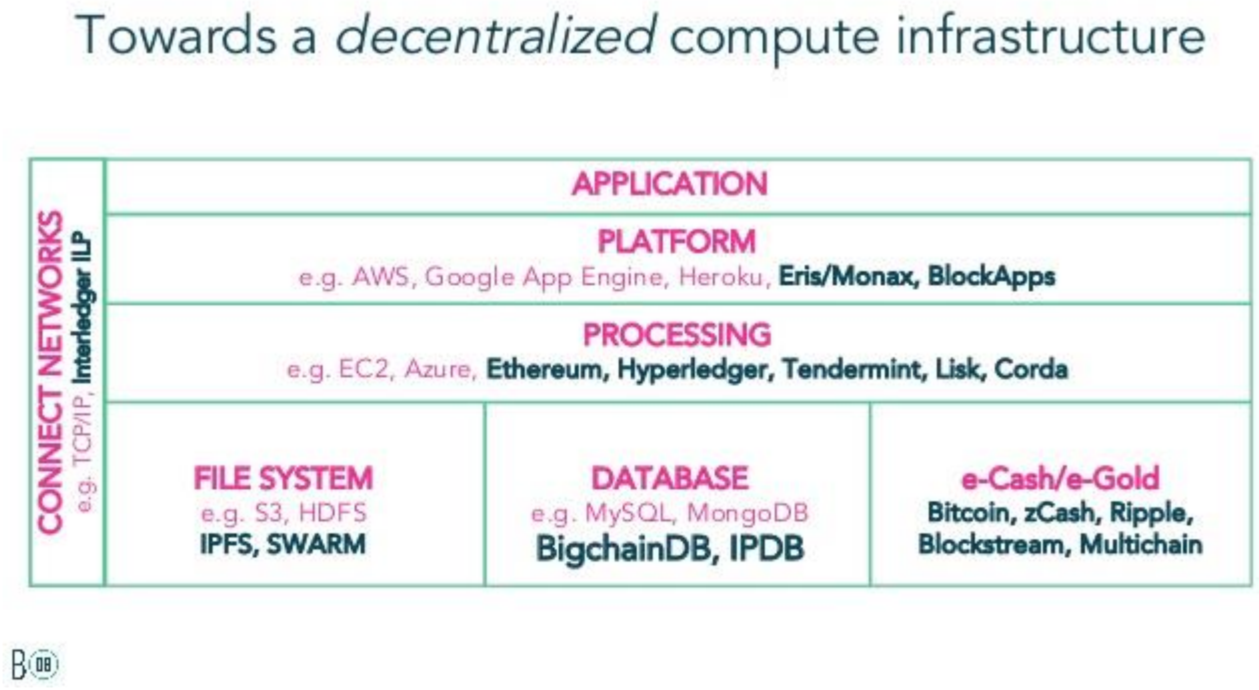
Decentralized
(Big Hubs and many spokes)



Distributed
(No hierarchy, strongly connected)



BigChain DB’s view of the future



Towards a *decentralized* compute infrastructure

CONNECT NETWORKS
e.g. TCP/IP, **Interledger ILP**

APPLICATION

PLATFORM
e.g. AWS, Google App Engine, Heroku, **Eris/Monax, BlockApps**

PROCESSING
e.g. EC2, Azure, **Ethereum, Hyperledger, Tendermint, Lisk, Corda**

FILE SYSTEM
e.g. S3, HDFS
IPFS, SWARM

DATABASE
e.g. MySQL, MongoDB
BigchainDB, IPDB

e-Cash/e-Gold
Bitcoin, zCash, Ripple, Blockstream, Multichain

Triple Entry Accounting Companies

- Balanc3 – A Consensys company, uses Ethereum and Bitcoin chains to trace exchange of value and provides bookkeeping using smart contracts
- Factom: Timestamped data hashing to the blockchain

How some blockchains work

From Bitcoin to Ethereum

A Classic Diagram

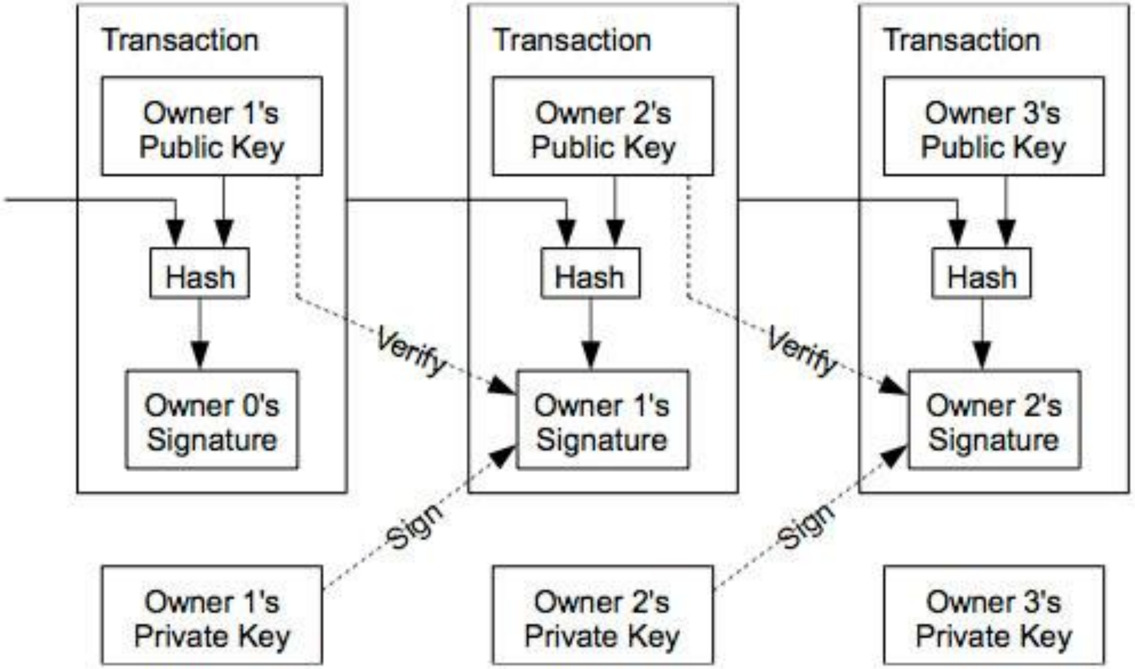
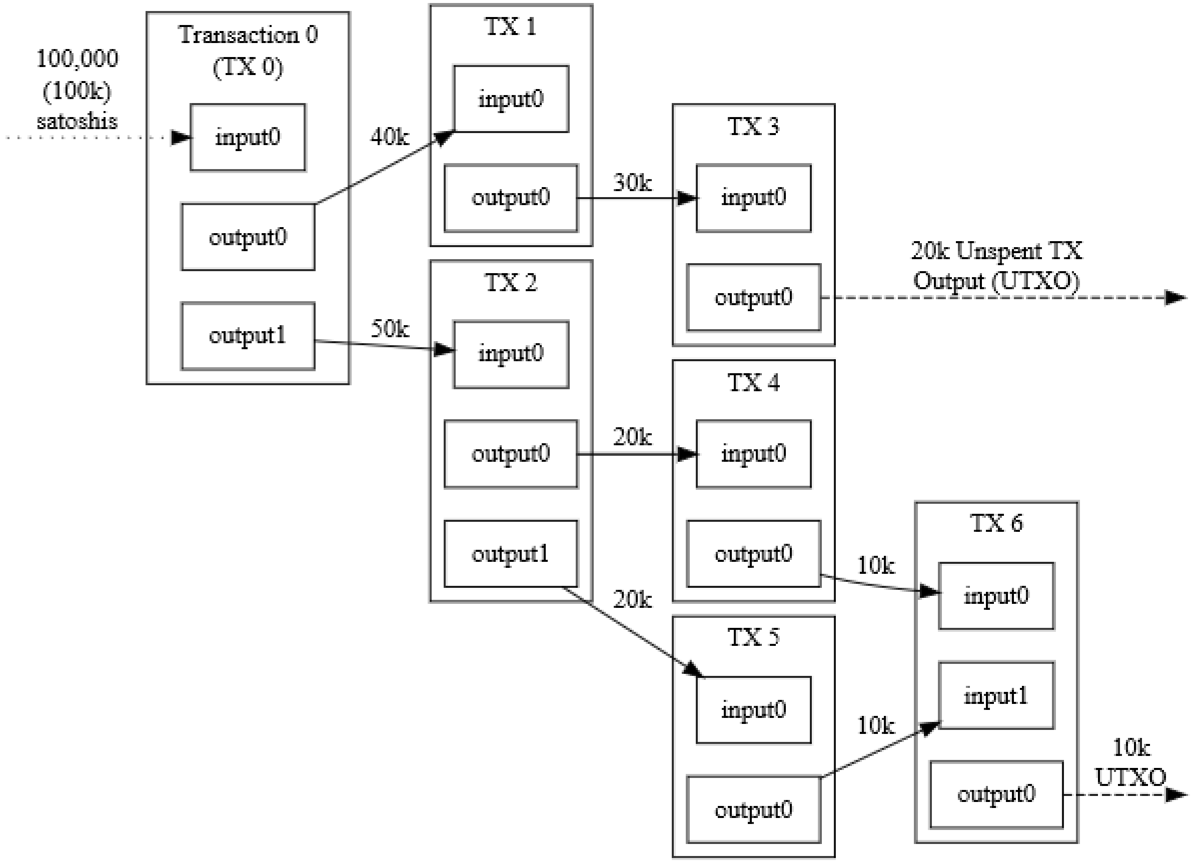


Diagram of a Bitcoin
from *Bitcoin: A Peer-to-Peer Electronic Cash System*,
published in 2008 by "Satoshi Nakamoto".

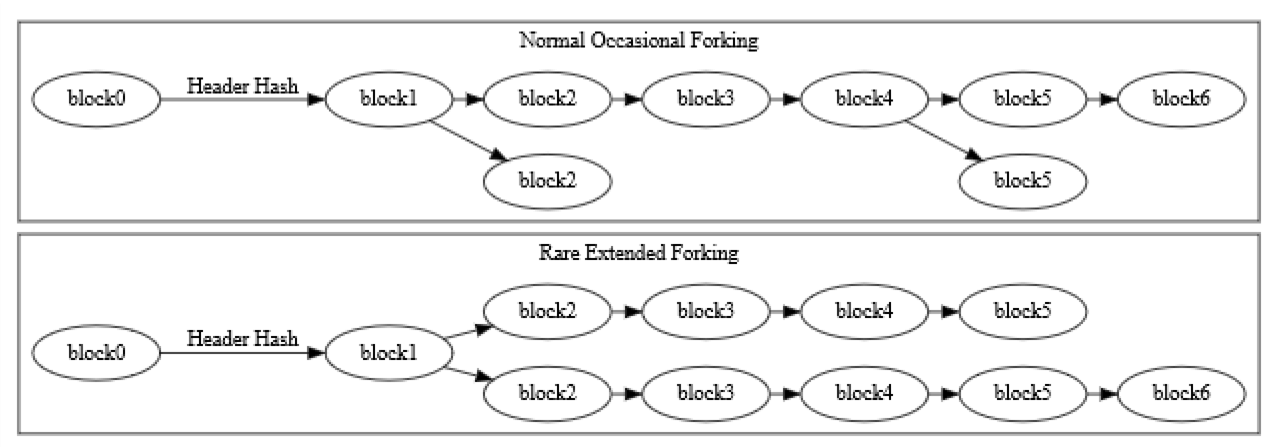
Unspent Transaction Outputs

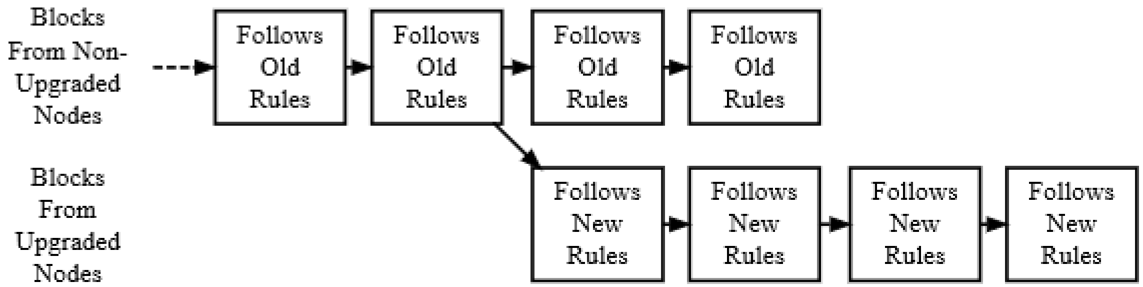


Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

Source: https://bitcoin.org/en/developer-guide#block-chain-overview

Forks in the chain – normal vs hard





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

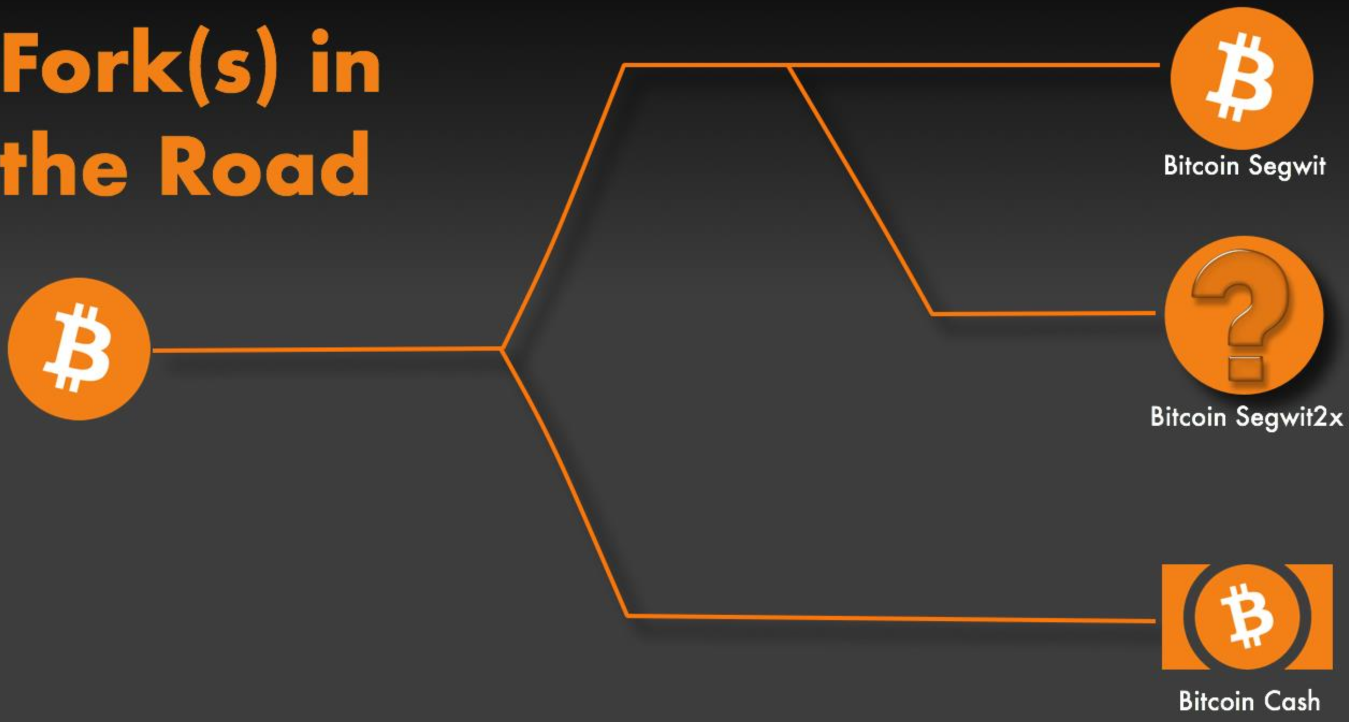
Bitcoin has been upgraded.
New features are available on Bitcoin Cash.

If you owned bitcoin on August 1st, you already have Bitcoin Cash.

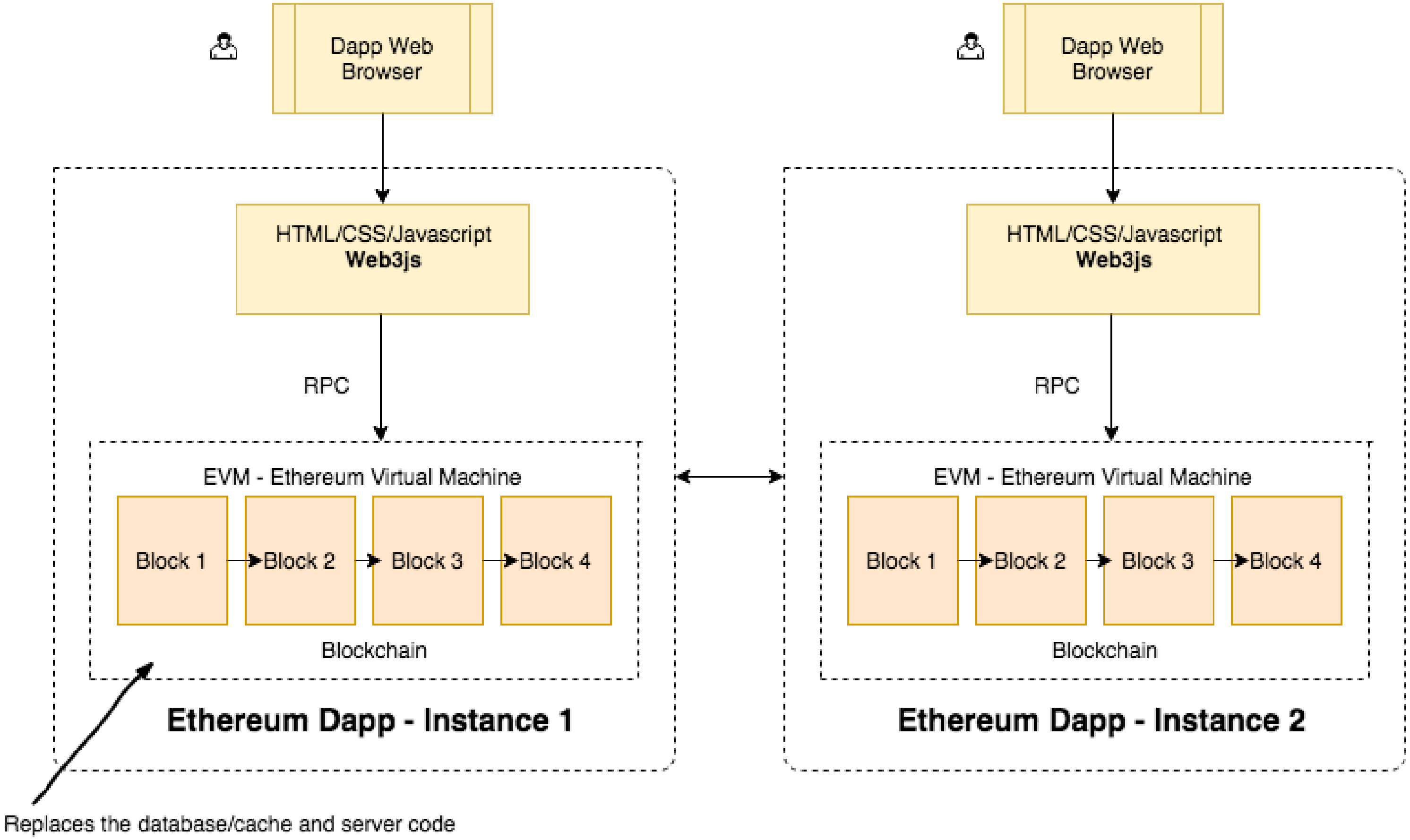
bitcoin		Bitcoin Cash
Standard Block Size: 1MB Maximum.	→	**PowerBlocks**: 8MB Maximum.
SegWit: Transaction signatures can be discarded from the blockchain.	→	**SecureSigs**: All transaction signatures must be validated and secured on the blockchain.
Single centralized development team and client implementation: Bitcoin Core.	→	**Multiple independent development teams** and client implementations including: Bitcoin Unlimited, Bitcoin ABC, Bitcoin XT, and Bitcoin Classic.
Scaling plan: Off-chain payment channels.	→	**Scaling Plan**: On-chain transactions and market driven blocksize increases.

Find out more at: www.bitcoincash.org | www.bitcoinabc.org | www.reddit.com/r/btc

The Bitcoin Fork(s) in the Road

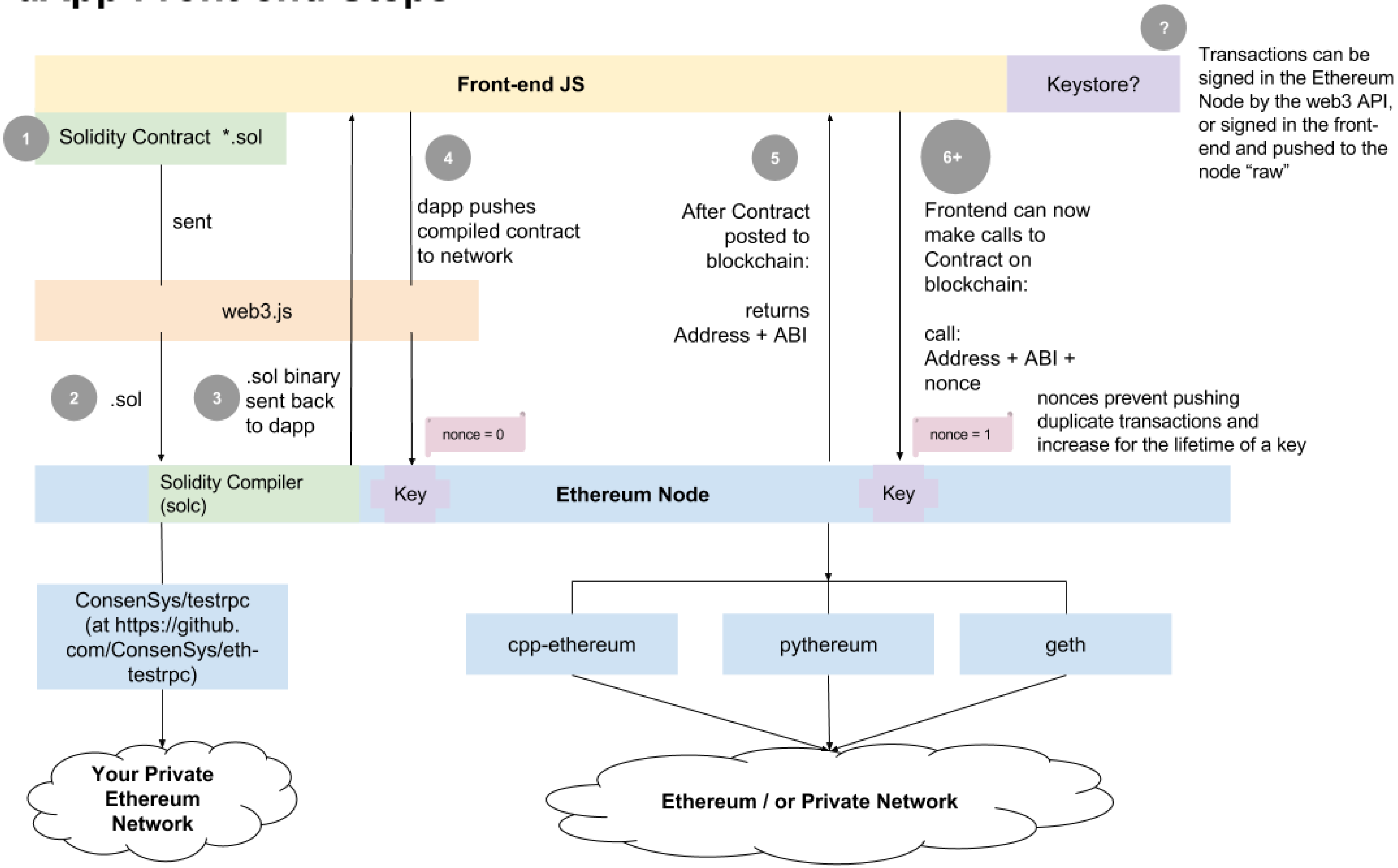


Ethereum



Source: https://medium.com/@mvmurthy/ethereum-for-web-developers-890be23d1d0c

dApp Front-end Steps



Front-end JS

Keystore?

?

Transactions can be signed in the Ethereum Node by the web3 API, or signed in the front-end and pushed to the node “raw”

1 Solidity Contract *.sol

sent

web3.js

2 .sol

3 .sol binary sent back to dapp

4 dapp pushes compiled contract to network

nonce = 0

5 After Contract posted to blockchain:

returns Address + ABI

6+ Frontend can now make calls to Contract on blockchain:

call: Address + ABI + nonce

nonce = 1

nonces prevent pushing duplicate transactions and increase for the lifetime of a key

Solidity Compiler (solc)

Key

Ethereum Node

Key

ConsenSys/testrpc (at https://github.com/ConsenSys/eth-testrpc)

cpp-ethereum

pythereum

geth

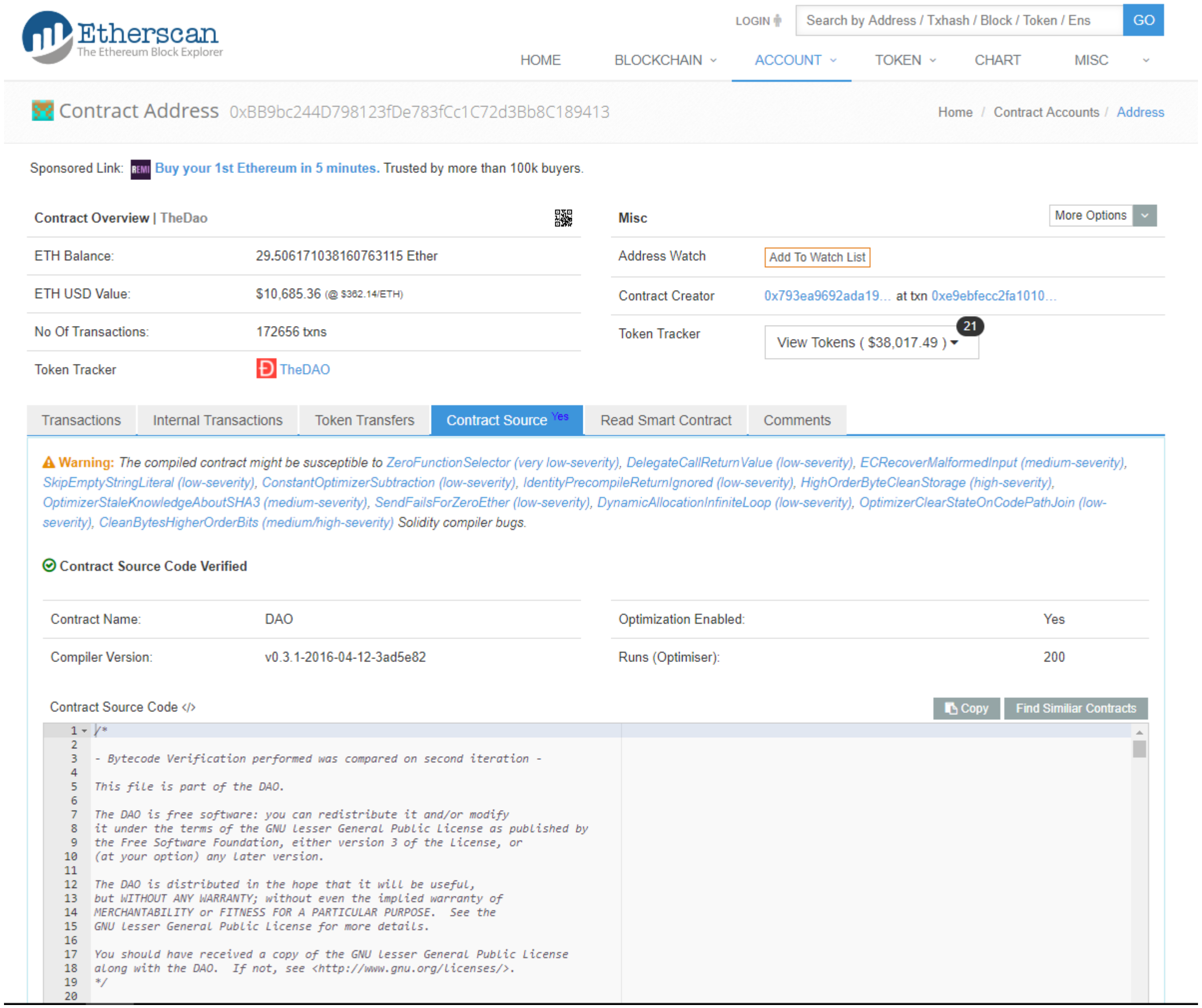
Your Private Ethereum Network

Ethereum / or Private Network

A **Contract Creation Transaction** is shown in steps 1-5 at above.

An **Ether Transfer** or **Function Call Transaction** is assumed in step 6.

Smart Contracts - Ethereum


Etherscan
The Ethereum Block Explorer
LOGIN
Search by Address / Txhash / Block / Token / Ens
GO
HOME
BLOCKCHAIN
ACCOUNT
TOKEN
CHART
MISC

Contract Address 0xBB9bc244D798123fDe783fCc1C72d3Bb8C189413

Home / Contract Accounts / Address

Contract Overview | TheDao	
ETH Balance:	29.506171038160763115 Ether
ETH USD Value:	$10,685.36 (@ $362.14/ETH)
No Of Transactions:	172656 txns
Token Tracker	TheDAO

Misc	More Options
Address Watch	Add To Watch List
Contract Creator	0x793ea9692ada19... at txn 0xe9ebfecc2fa1010...
Token Tracker	View Tokens ($38,017.49) 21

⚠ **Warning:** *The compiled contract might be susceptible to ZeroFunctionSelector (very low-severity), DelegateCallReturnValue (low-severity), ECRecoverMalformedInput (medium-severity), SkipEmptyStringLiteral (low-severity), ConstantOptimizerSubtraction (low-severity), IdentityPrecompileReturnIgnored (low-severity), HighOrderByteCleanStorage (high-severity), OptimizerStaleKnowledgeAboutSHA3 (medium-severity), SendFailsForZeroEther (low-severity), DynamicAllocationInfiniteLoop (low-severity), OptimizerClearStateOnCodePathJoin (low-severity), CleanBytesHigherOrderBits (medium/high-severity) Solidity compiler bugs.*

✓ **Contract Source Code Verified**

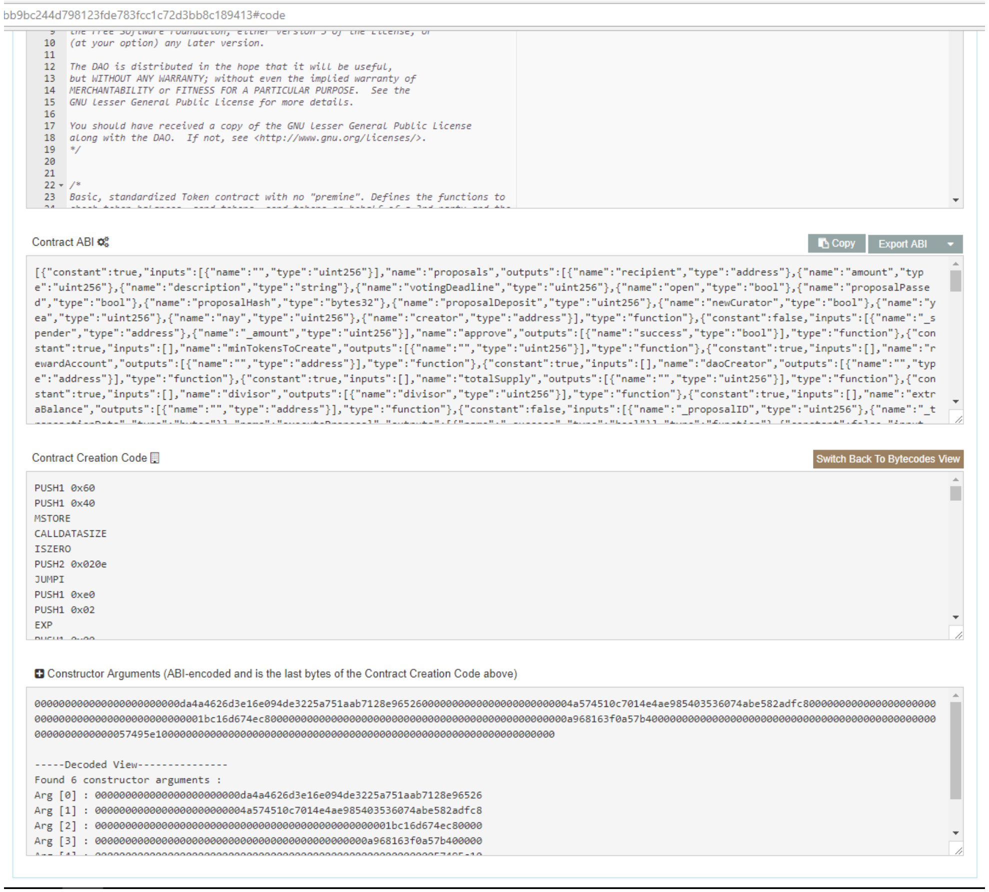
Contract Name:	DAO	Optimization Enabled:	Yes
Compiler Version:	v0.3.1-2016-04-12-3ad5e82	Runs (Optimiser):	200

Contract Source Code </>

```
/*

- Bytecode Verification performed was compared on second iteration -

This file is part of the DAO.

The DAO is free software: you can redistribute it and/or modify
it under the terms of the GNU lesser General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

The DAO is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU lesser General Public License for more details.

You should have received a copy of the GNU lesser General Public License
along with the DAO.  If not, see <http://www.gnu.org/licenses/>.
*/

```

Address: 0xBB9bc244D798123fDe783fCc1C72d3Bb8C189413

Compiled & Deployed Contracts



bb9bc244d798123fde783fcc1c72d3bb8c189413#code

```
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

The DAO is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU lesser General Public License for more details.

You should have received a copy of the GNU lesser General Public License
along with the DAO.  If not, see <http://www.gnu.org/licenses/>.
*/


/*
Basic, standardized Token contract with no "premine". Defines the functions to
```

Contract ABI

```
[{"constant":true,"inputs":[{"name":"","type":"uint256"}],"name":"proposals","outputs":[{"name":"recipient","type":"address"},{"name":"amount","typ
e":"uint256"},{"name":"description","type":"string"},{"name":"votingDeadline","type":"uint256"},{"name":"open","type":"bool"},{"name":"proposalPasse
d","type":"bool"},{"name":"proposalHash","type":"bytes32"},{"name":"proposalDeposit","type":"uint256"},{"name":"newCurator","type":"bool"},{"name":"y
ea","type":"uint256"},{"name":"nay","type":"uint256"},{"name":"creator","type":"address"}],"type":"function"},{"constant":false,"inputs":[{"name":"_s
pender","type":"address"},{"name":"_amount","type":"uint256"}],"name":"approve","outputs":[{"name":"success","type":"bool"}],"type":"function"},{"con
stant":true,"inputs":[],"name":"minTokensToCreate","outputs":[{"name":"","type":"uint256"}],"type":"function"},{"constant":true,"inputs":[],"name":"r
ewardAccount","outputs":[{"name":"","type":"address"}],"type":"function"},{"constant":true,"inputs":[],"name":"daoCreator","outputs":[{"name":"","typ
e":"address"}],"type":"function"},{"constant":true,"inputs":[],"name":"totalSupply","outputs":[{"name":"","type":"uint256"}],"type":"function"},{"con
stant":true,"inputs":[],"name":"divisor","outputs":[{"name":"divisor","type":"uint256"}],"type":"function"},{"constant":true,"inputs":[],"name":"extr
aBalance","outputs":[{"name":"","type":"address"}],"type":"function"},{"constant":false,"inputs":[{"name":"_proposalID","type":"uint256"},{"name":"_t
```

Contract Creation Code

```
PUSH1 0x60
PUSH1 0x40
MSTORE
CALLDATASIZE
ISZERO
PUSH2 0x020e
JUMPI
PUSH1 0xe0
PUSH1 0x02
EXP
```

Constructor Arguments (ABI-encoded and is the last bytes of the Contract Creation Code above)

```
000000000000000000000000da4a4626d3e16e094de3225a751aab7128e965260000000000000000000000004a574510c7014e4ae985403536074abe582adfc8000000000000000000000
0000000000000000000000000001bc16d674ec80000000000000000000000000000000000000000000000000000a968163f0a57b400000000000000000000000000000000000000000000000
0000000000000057495e100000000000000000000000000000000000000000000000000000000000000000

-----Decoded View---------------
Found 6 constructor arguments :
Arg [0] : 000000000000000000000000da4a4626d3e16e094de3225a751aab7128e96526
Arg [1] : 0000000000000000000000004a574510c7014e4ae985403536074abe582adfc8
Arg [2] : 0000000000000000000000000000000000000000000000001bc16d674ec80000
Arg [3] : 000000000000000000000000000000000000000000000a968163f0a57b400000
```

Public Versus Private

Comparing the chains

A spectrum between openness and private control

- Public chains: Open

Writers: Anyone

Trust base: Global validation & consensus

Applications:

1. Dapps
2. Cryptocurrency (ICO)

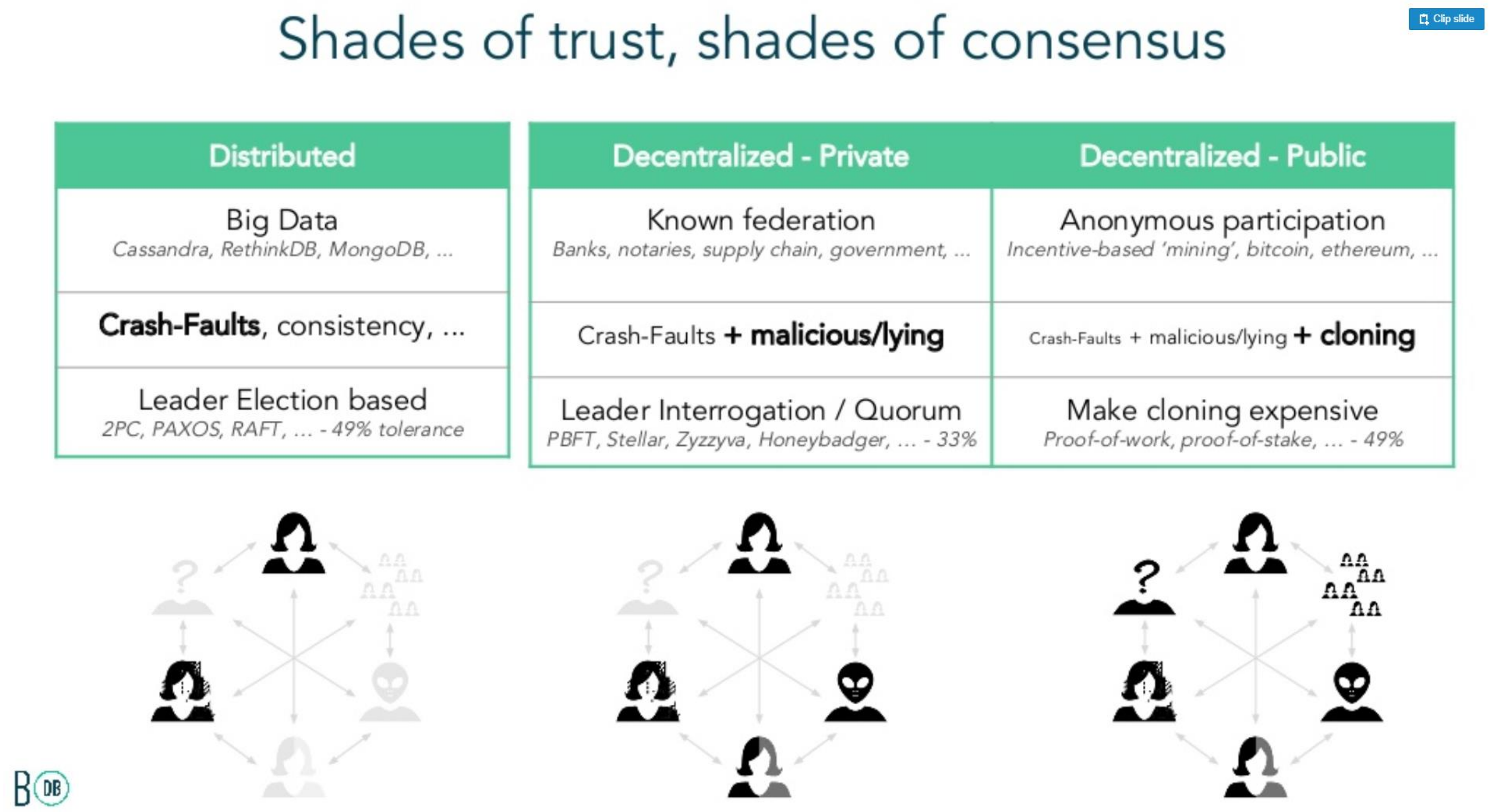
- Consortia chains: Closed, private membership

Writers: Known participants

Trust base: Voting, dictatorships

Applications:

1. Enterprise apps
2. Clearing & Settlement
3. Provenance chains
4. Asset Registries with partial trust



Shades of trust, shades of consensus

Distributed	Decentralized - Private	Decentralized - Public
Big Data *Cassandra, RethinkDB, MongoDB, ...*	Known federation *Banks, notaries, supply chain, government, ...*	Anonymous participation *Incentive-based 'mining', bitcoin, ethereum, ...*
Crash-Faults, consistency, ...	Crash-Faults **+ malicious/lying**	Crash-Faults + malicious/lying **+ cloning**
Leader Election based *2PC, PAXOS, RAFT, ... - 49% tolerance*	Leader Interrogation / Quorum *PBFT, Stellar, Zyzzyva, Honeybadger, ... - 33%*	Make cloning expensive *Proof-of-work, proof-of-stake, ... - 49%*

Source: BigChain DB