KPMG LLP

Consensus – Immutable agreement for the internet of value
Understanding an evolving Blockchain technology landscape of consensus-driven opportunity in financial services

Appendix 3 – Detailed Interview / Questionnaire Responses

Note: This Appendix 3 contains detailed responses to the evaluation questionnaire listed in Appendix 2. Responses are based on a combination of interviews with DLT contacts (where permission has been granted to publish) and KPMG research. In most cases (and where applicable), questionnaire responses from the DLT contacts have been preserved and published verbatim.

Table of Contents

1.	BigChainDB	4
2.	BitShares	13
3.	CASPER	20
4.	Corda	29
5.	DAG (Directed Acyclic Graphs)	34
6.	Derived PBFT (Hyperledger project)	39
7.	Distributed Concurrence	45
8.	Evernym	53
9.	Graphene	67
10.	Juno	76
11.	MultiChain	82
12.	OpenChain	93
13.	PoET by Intel	101
14.	RAFT	107
15.	Ripple	114
16.	Steem	123
17.	Stellar	129
18.	Tangaroa	136

19. Tendermint **143**

Contact Us and Acknowledgements **150**

1. BigChainDB

Source: Interview / Questionnaire

Contact name: Trent McConaghy (gtrent@gmail.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

“Two-level consensus. Bottom level is Raft, to order & store all writes. Top-level is a new algorithm where nodes do a post-write vote.”

How many nodes are need to validate a transaction? (percentage vs. number)

Majority.

Do all nodes need to be online for system to function?

No. # Nodes is up to deployer of network. Upcoming public network will support >100K client nodes, with 20-30 server nodes to start and many more later.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

No

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Participants of network.

What are the different stages involved within the consensus mechanism?

Write (and implicitly: order) a block, vote whether block is valid.

When is a transaction considered "safe" or "live"?

Majority of nodes have voted block as valid.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

One round. Main latency limit is speed of light. In practice, <1 s to <100 ms.

How much time does a node need to reach a decision?

See above.

How much time is actually needed to build the consensus until a new block is added?

See above.

Does system contain synchronous node decision making functionality?

Eventually consistent.

What is the number of current and planned validators?

# *Nodes is up to deployer of network.*

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

1/3 of nodes. FT++. In road map: BFT, for rare situations when the severe performance compromise is justified.

Is there a forking vulnerability?

No. Algorithms in the lineage of Paxos, including Raft, create transaction logs; forking is not in their vocabulary.

How are the incentives defined within a permissioned system for the participating nodes?

Extrinsic: Legal contracts. "If you're malicious, see you in court."

What process does the system follow when it receives data?

It allocates the transaction into a block (set of TX), then writes the block to the DB using the bottom-level consensus algorithm (Raft). Voting on the block occurs afterwards.

How is data currently stored?

JSON-style document store. It's a database.

How does a party take ownership of an asset?

They can register an asset, or have an asset transferred to them. Owner(s) = controls private key. Interledger protocol supported.

Governance, Risks and Control

How is governance / controls enforced?

Up to the ringleader deploying the network. We recommend leveraging existing legal system, e.g., a joint venture or a nonprofit foundation, with standard governance documents.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

An individual node, or the governing body, it depends on the situation. Leverage existing legal system.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Not built into the code itself. It could be built into the governing documents.

How does the consensus mechanism allow access?

Ring of public keys of servers. In road map: more fine-grained support.

How does the consensus mechanism restrict access, concerning malicious activities?

Each node listens to the change feed, malicious actions will be undone after quorum.

What is the permission management process? What is the process for adding or deleting nodes?

Voted on by existing nodes.

How does the protocol assess the trustworthiness of other participants?

Key assesses the permissions. Trustworthiness can be assessed based on, for example, how many times does a node agree with the quorum.

Are there separate admin / administrator privileges? Who manages them?

Currently, no. In road map: admin role proposes changes, which go through upon quorum by nodes.

Are there restriction / privacy rights defined and enforced by node?

Restriction rights: by owner (more fine-grained than by node), via private key based fulfillment conditions. Privacy rights: clients see less than server nodes and any TX can have client-side encryption.

Can a node or a user have only "Read" or only "Write” access? Is specific node access required if only performing one functionality? (e.g., back office outsourcing)

At this point, all federation nodes are equal and have read/write permissions. Clients, however only have read permissions through an API. At this point they can read all the information in the database.

What are the measures in place to reduce risk?

In SW development process, we optimize for risk and other relevant DB criteria. Open-source, third-party audits, etc.

In case of permissioned systems, who manages the “know your client” (KYC)/anti-money laundering (AML) process and where is the data stored?

It's a database. KYC/AML, etc. are at the application level, not the DB level.

How is counterparty risk / settlement risk addressed?

See above.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

<1 s to <100 ms.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

1M writes/s, capacity > 1 PB.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Whatever you like. Doesn't rely on native tokens.

How do you measure scalability?

Throughput, latency, capacity.

Is there a limitation on the number of fields within a transaction?

Keys are JSON documents, so you not only have a list of fields, but they can be hierarchically organized. No limit, except that the JSON document must be <65MB.

Is the speed of the system impacted if the system is made more scalable?

As more nodes are added, throughput and capacity improve, and latency worsens. Just like any modern "big data" distributed DB.

Does synchronization have any impact on scalability?

The current consensus algorithm is mostly asynchronous (except for dependent transactions).

Security

How is transaction activity monitored?

However the system deployer would like. We provide code to help monitoring.

Does the consensus mechanism utilize Digital Signatures?

Yes. bigchaindb.readthedocs.org, bigchaindb.com/whitepaper, GitHub.com/BigChainDB.

How does the consensus mechanism address an assumed industry standard?

No. There will be code and security audits.

Which risk/security issues are currently being worked on?

Shrinking the list of known fault vectors.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

Raft is formally verified. We'll check the certification box if there is sufficient demand.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

It's a database. Obvious how to plug into MVC, LAMP stack, etc.

Briefly describe the security testing performed till date (if any)

Extensive internal benchmarks, ongoing, with FT engineers.

How are you planning to implement/integrate Digital wallets? (Including private key management)

Client side SW libraries.

In case of a breach, what data is at risk?

None, if encrypted. If not, a compromised node can read the data. But assets can't be transferred because protected by owners' private keys; and data won't be lost assuming deployer uses industry-standard DB backup practices (offline storage etc.).

How does the system prevent signature fraud (e.g., stolen keys)?

Just multi signature. “Don't lose your keys.”

Does the consensus mechanism have full documentation in place?

Yes. See http://bigchaindb.readthedocs.org, http://www.bigchaindb.com/whitepaper, http://www.GitHub.com/bigchaindb

How is the system expected to address general server issues?

The same way “sysadmins” currently administer their DBs. No magic needed.

How does the consensus mechanism address the risk of "double spending"?

A block of TXs only passes validation by voting nodes if it doesn't hold any double spend TXs.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

See overall consensus methodology Q's. In short: Raft to order + write; then vote.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Raft has a Lamport-style logical clock.

Under which conditions does a lock/unlock happen? (i.e., what is the proof safety?)

No concept of lock needed. That would greatly impact performance.

What is the process for disaster recovery?

Like any modern DB: use the offline backup.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

It's a database. Fraud etc. are at the application level, not the DB level.

Privacy

How does the system ensure privacy?

Client nodes' visibility limited by what they can query. Server nodes can see all which isn't encrypted. Clients are represented in the database by a public key. The database has no idea of the mapping between the public key and the entity that owns it.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes, signature verification in place.

Do all nodes have visibility into all other transactions?

Server-side nodes have transparency to each other.

How is privacy defined and ensured between applications?

Deploy different networks or use encryption.

How does the data encryption model work?

Client-side.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

This can be done by client. As a client, one may chose, for example, to create a new key pair for each transaction/asset.

Are participants' identities hidden from one another? (e.g., Blackpool)

Clients are represented in the database by a public key. The database has no idea of the mapping between the public key and the entity that owns it.

Cryptography/Strength of Algorithm

How are the keys generated?

The keys are created using the ED25519 (Schnorr) signature scheme. As of April 2016, EdDSA was in "Internet-Draft" status with the IETF but was already widely used. Each user and federation node creates his own set of keys.

What does the key life cycle management look like?

It is advised to rotate them every few months. Keys can be easily imported using CLI commands and/or updating the configuration file.

What is the library approach?

The library used comes is a python wrapper of the SUPERCOP C benchmark suite, using the portable "ref" implementation (not the high-performance assembly code), and is very similar to the copy in the NaCl library.

What is the HSM integration approach?

NA, we do advise users not to share virtual machines on cloud platforms in order to avoid memory leaks.

Does the consensus mechanism require a leader?

The bottom-level Raft consensus has a dynamically chosen leader. The top-level protocol doesn't.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

Currently, majority. Easy to change as code is open source.

How is node behavior measured for errors?

Change feed mechanism: all nodes always hear about behavior of other nodes.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

The content of a transaction contains: ID, time stamp, crypto-fulfilments (inputs), crypto-conditions (output) and a payload (json-data). The ID is a hash of the transaction except for the fulfillment, since a transaction needs an ID before signing and signatures cannot sign itself (ref: transaction malleability).

Which security mechanisms are assigned to the tokens?

Private keys. Interledger protocol supported.

Briefly described the life cycle management process for the tokens

Clients request the issuance of assets. Only federation nodes can create assets. An owner of an asset, i.e., controller(s) of private key, can transfer the asset.

Does the consensus mechanism utilize transaction signing?

Yes, the client needs to sign transactions.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

The use case of consensus is consistency, like any distributed DB. Higher-level use cases: ID, IP, supply chain, financial, energy, certification. Put another way: anywhere you need a DB with no single owner / controller, or anywhere you need a Blockchain at enterprise scale throughput/storage/latency.

What is the implementation cost?

In general, just the cost of cloud resources.

What is the time required to implement?

Up & running in <5 min. It's a database after all. The real time is the time to build the application.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

If you can afford to run a DB, you can afford to use our DB. No magical new concepts necessary. You should only use this DB compared to non-Blockchain DBs if there is benefit from decentralization (no single entity owns or controls), immutability (greater tamper-resistance), or assets (easy mechanisms to issue & transfer assets; assets "live" on DB).

Who are you currently working with? (e.g., Venture Capitalists, Banks, Credit Card companies, etc.)

Organizations that need a database with Blockchain characteristics. This includes the following; we are working with all of these: consortium ringleaders, Blockchain startups in ID/IP/etc., FORTUNE 500 enterprises, financial services, storage vendors, cloud services.

2. BitShares

Source: Interview / Questionnaire

Contact name: Ryan R. Fox (ryan@ryanrfox.com.)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

Delegated Proof of Stake (DPOS) https://bitshares.org/technology/delegated-proof-of-stake-consensus/

How many nodes are need to validate a transaction? (percentage vs. number)

Live stats: http://cryptofresh.com/witnesses Currently 27 witnesses are producing blocks. This is dynamic based on shareholders voting for a witness slate.

Do all nodes need to be online for system to function?

Same as Graphene. See below.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Same as Graphene. See below.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Same as Graphene. See below.

What are the different stages involved within the consensus mechanism?

Same as Graphene. See below.

When is a transaction considered "safe" or "live"?

Same as Graphene. See below.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Same as Graphene. See below.

How much time is actually needed to build the consensus until a new block is added?

Same as Graphene. See below.

Does system contain synchronous node decision making functionality?

Same as Graphene. See below.

What is the number of current and planned validators?

Currently 27, but is dynamic based on approval voting. http://cryptofresh.com/witnesses

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

Same as Graphene. See below.

Is there a forking vulnerability?

Same as Graphene. See below.

How are the incentives defined within a permissioned system for the participating nodes?

Same as Graphene. See below.

What process does the system follow when it receives data?

Same as Graphene. See below.

How is data currently stored?

Same as Graphene. See below.

How does a party take ownership of an asset?

BTS is the token.

Governance, Risks and Controls

How is governance / controls enforced?

Same as Graphene. See below.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

Same as Graphene. See below.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Same as Graphene. See below.

How does the consensus mechanism allow access?

Same as Graphene. See below.

How does the consensus mechanism restrict access, concerning malicious activities?

Same as Graphene. See below.

What is the permission management process? What is the process for adding or deleting nodes?

Same as Graphene. See below.

How does the protocol assess the trustworthiness of other participants?

Same as Graphene. See below.

Are there separate admin / administrator privileges? Who manages them?

Same as Graphene. See below.

Are there restriction / privacy rights defined and enforced by node?

Same as Graphene. See below.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

Same as Graphene. See below.

How is counterparty risk / settlement risk addressed?

Same as Graphene. See below.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Same as Graphene. See below.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., number of trades)

http://coinmarketcap.com/currencies/bitshares/#charts

How do you measure scalability?

http://stats.bitshares.eu/

Is the speed of the system impacted if the system is made more scalable?

Same as Graphene. See below.

Security

In case of a breach, what data is at risk?

Same as Graphene. See below.

Does the consensus mechanism have full documentation in place?

Same as Graphene. See below.

How does the consensus mechanism address the risk of "double spending"?

Same as Graphene. See below.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Same as Graphene. See below.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Same as Graphene. See below.

Under which conditions does a lock/unlock happen? (i.e., what is the proof safety?)

Same as Graphene. See below.

What is the process for disaster recovery?

Same as Graphene. See below.

What is the threat model being tested? What has been defined as “normal’? How do you monitor fraud?

Same as Graphene. See below.

Privacy

How does the system ensure privacy?

Same as Graphene. See below.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Same as Graphene. See below.

Do all nodes have visibility into all other transactions?

Same as Graphene. See below.

How is privacy defined and ensured between applications?

Same as Graphene. See below.

How does the data encryption model work?

Same as Graphene. See below.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

Same as Graphene. See below.

Are participants' identities hidden from one another? (e.g., Blackpool)

Same as Graphene. See below.

Cryptography / Strength of Algorithm

What does the key life cycle management look like?

Address = ‘BTS’+base58(ripemd(sha512(compressed_pub))) (checksum obviated) But addresses are not used directly, instead you have an account (that can be controlled by one or more address, pubkey or another account).
https://bitshares.org/technology/dynamic-account-permissions/

What is the library approach?

Same as Graphene. See below.

Does the consensus mechanism require a leader?

Same as Graphene. See below.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

Same as Graphene. See below.

How is node behavior measured for errors?

Same as Graphene. See below.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

The token is BTS.

Which security mechanisms are assigned to the tokens?

Same as Graphene. See below.

Briefly described the life cycle management process for the tokens

Same as Graphene. See below.

Does the consensus mechanism utilize transaction signing?

Same as Graphene. See below.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Same as Graphene. See below.

What is the implementation cost?

Free, License is MIT.

What is the time required to implement?

A witness node can be spun up on the BitShares network using the Microsoft Azure Blockchain as a Service platform (https://GitHub.com/Azure/azure-quickstart-templates/tree/master/bitshares-ubuntu-vm)

Who are you currently working with? (e.g., venture capitalists, banks, credit card companies, etc.)

CCEDK, OpenLedger, BlockTrades

3. CASPER

Source: Interview / Questionnaire

Contact name: Vlad Zamfir (vldzmfr@gmail.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

“consensus-by-bet” - nodes expose themselves to loss in concert, in order to commit to decisions.

How many nodes are need to validate a transaction? (percentage vs. number)

If a node commits at all to an invalid transaction execution they lose their entire security deposit. All consensus-forming nodes are expected to validate every TX.

Do all nodes need to be online for system to function?

The system functions so long as one consensus-forming node remains online (Casper very strongly favors availability).

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

At any time the participants are known to all clients - current participants finalize changes to the set of consensus-forming nodes.

What are the different stages involved within the consensus mechanism?

Casper involves betting in concurrent “betting cycles”, each of which only has one stage. Betting cycles never restart from ground zero, they “just continue” until convergence or until they are combined.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

There is a (in-consensus) process that manages the betting cycles, they terminate on convergence and when they are composed.

If applicable, what is the voting process after the "propose" stage?

Each betting cycle chooses between mutually exclusive proposals.

When is a transaction considered "safe" or "live"?

The "safety" of a given transaction receipt R is the amount of security deposits that would be lost in all consensus states where the TX receipt is not R.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Betting cycles aren't guaranteed to terminate in a given amount of time, and does not progress in rounds. Bets form a DAG, and some DAGs correspond to a convergent betting cycle. Cycles are used to choose between competing blocks at every height.

How much time does a node need to reach a decision?

Decisions are not guaranteed to happen in a finite amount of time in an asynchronous network (FLP impossibility).

Does system contain synchronous node decision making functionality?

No – although some strategy choices involve timeouts, these choices are never decisions in the traditional sense of the word.

What is the number of current and planned validators?

We like to say 250, but the number is not set in stone.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

>50% can censor transactions (although not without being punished), b% can revert blocks/state created by a% < b% of nodes, unless a% > finality threshold F in which case no amount of faults can revert the block/state, the proportion of nodes required to prevent convergence depends on network condition (one faulty node is enough under asynchronous networks, a la FLP), and 100% of nodes must be faulty to secretly create an invalid block, but only the finality threshold of faults are required for nodes to finalize invalid blocks, and 2*(F – 50%) faults can cause permanent consensus failure in an asynchronous network (requires hard-fork to fix) , but F faults are required to cause consensus failure in a synchronous network (weird network conditions will require some number of faults 2*(F – 50%) < x < F for consensus failure).

Is there a forking vulnerability?

Non finalized state can be reverted, consensus failure is possible due to finality conditions.

How are the incentives defined within a permissioned system for the participating nodes?

Precisely the same way they are defined for a public economic system – super weird question because incentives normally are not required for permissioned systems.

What process does the system follow when it receives data?

The I of I/O works only through transactions. Transactions have payload. O happens through transaction receipts.

How is data currently stored?

A Patricia Merkle tree that stores the state of all the distributed virtual machine.

How does a party take ownership of an asset?

That depends on the asset – Ethereum supports arbitrary access policies.

Governance, Risks and Control

How is governance / controls enforced?

Hard forks happen completely outside of the protocol, and without a clear process. This mitigates people's ability to game upgrades.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

Not clear that any legal action will be taken, we want to invite attack, not discourage it.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

The protocol aims to guarantee the forfeiture of security deposits of nodes exhibiting byzantine behavior.

How does the consensus mechanism allow access?

As long as there is no censorship anyone can place a bonding transaction and become a consensus-forming node.

How does the consensus mechanism restrict access, concerning malicious activities?

It revokes access when it sees clearly malicious behavior, and institutes punishment when it seems shady behavior.

What is the permission management process? What is the process for adding or deleting nodes?

All changes to the set of nodes are validated and finalized by the consensus - there is a queue for bonding and a waiting time before deposits are returned, some details vary between versions of Casper.

How does the protocol assess the trustworthiness of other participants?

Security deposits are taken as performance bonds – nodes that perform poorly will not be left with as much of their deposit.

Are there separate admin / administrator privileges? Who manages them?

No – Casper is designed specifically not to have admins.

Are there restriction / privacy rights defined and enforced by node?

No – Casper is a public protocol.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., Back Office outsourcing)

Only bonded nodes can write, anyone can read.

What are the measures in place to reduce risk?

A lot of effort is being placed in formal verification, and more and more peer review is being done.

How is counterparty risk / settlement risk addressed?

Finalized blocks are used to mitigate settlement risk, economic consensus punishes counterparties who behave badly.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

We're still in early prototype phases, no real measurements are available.

How do you measure scalability?

Not much more scalable than a normal Blockchain.

Is the speed of the system impacted if the system is made more scalable?

Adding nodes increases overhead and does not scale capacity – increasing the capacity of nodes does scale capacity. We plan on moving to Blockchain sharing ASAP.

Does synchronization have any impact on scalability?

Network weirdness increase the protocol's overhead and reduces capacity.

Security

How is transaction activity monitored?

By validators who verify signatures and transaction fees.

Does the consensus mechanism utilize Digital Signatures?

Yes – the bitcoin curve.

How does the consensus mechanism address an assumed industry standard?

For any standard consensus protocol, one can find a network condition where Casper converges but that consensus protocol does not. Casper is more flexible than PBFT, RAFT, Tendermint, etc.

Which risk/security issues are currently being worked on?

"Greifing" attacks where nodes go offline to punish online nodes, who the protocol believes are censoring the offline nodes.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

No – haven't thought about it – still in early peer review stages.

Briefly describe the security testing performed till date (if any)

Small amounts of simulation, mostly analysis.

How are you planning to implement/integrate Digital wallets? (Including private key management)

This question is independent of the consensus protocol, and is an Ethereum-wide thing.

In case of a breach, what data is at risk?

Data on the Blockchain is not private.

How does the system prevent signature fraud (e.g., stolen keys)?

Systems for revocation and recovery of credentials can be built into smart contracts.

Does the consensus mechanism have full documentation in place?

Not yet – the protocol is still in flux.

How is the system expected to address general server issues?

The validators are responsible for these issues, if they don't fix them they will be operating at a loss.

How does the consensus mechanism address the risk of "double spending"?

We provide a consensus protocol. There is no separate mechanism for double-spends.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Security-deposit proof-of-stake allows for super-fast synchronization.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Clocks need to be synchronized, but network messages don't need to propagate in predictable times.

Under which conditions does a lock/un-lock happen? (i.e., what is the proof safety?)

A finality threshold of nodes are required to make finalized decisions.

What is the process for disaster recovery?

Hard forks are used to recover from consensus failure, long (order of months) timeouts are used to recover from mass crash-faults.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

We assume that the market of bonded validators is highly concentrated, and we want to show that the protocol guarantees are not undermined under oligopolistic market models, unless there is some quantifiable extra-protocol incentive to undermine the protocols.

Privacy

How does the system ensure privacy?

It doesn't

Does the system require verifiable authenticity of the messages delivered between the nodes?

All blocks and transactions and bets have signatures.

Do all nodes have visibility into all other transactions?

Yes – it's a public Blockchain.

How is privacy defined and ensured between applications?

It's not

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

Credentials should be issues to each node.

Are participants' identities hidden from one another? (e.g., Blackpool)

No, all members know the list of all members.

Cryptography/Strength of Algorithm

How are the keys generated?

We use the Bitcoin curve for transactions, but contracts can verify arbitrary credentials.

What does the key life cycle management look like?

Still too immature to tell – very flexible credential management systems can be built on top of Ethereum.

Does the consensus mechanism require a leader?

Some versions of Casper have a round-robin, none have leaders who produce blocks until they timeout, however.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

We do not support changing the rules of the protocol from inside the protocol as it would be a vulnerability.

How is node behavior currently measured for errors?

The betting cycle's trace evidences the node's performance, and is used to determine payoffs.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Ether is used for deposits, possibly fees, contracts can be used for arbitrary token stuff.

Which security mechanisms are assigned to the tokens?

Security deposits are taken as performance bonds – nodes that perform poorly will not be left with as much of their deposit.

Briefly described the life cycle management process for the tokens

Security deposits have a lifetime on the order of months that the protocol sets.

Does the consensus mechanism utilize transaction signing?

Yes – for now the bitcoin curve – eventually we're moving to use arbitrary stateless credential verification.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

The public Ethereum Blockchain.

What is the implementation cost?

So far we've probably spent around $100K, probably we require a lot more.

What is the time required to implement?

Unknown, although progress seems to be steady.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Not a business case, but we do compare Casper to other public economic consensus protocols.

Who are you currently working with? (e.g., Venture Capitalists, Banks, Credit Card companies, etc.)

This work is being done by academics and individual professional researchers and hobbyists.

4. Corda

Source: KPMG Research

Contact name: See Contact Us below

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

Corda achieves consensus between firms at the level of individual deals, not the level of the system.

How many nodes are need to validate a transaction? (percentage vs. number)

n2n Corda transactions are validated by parties to the transaction rather than a broader pool of unrelated validators (N2N/2).

Do all nodes need to be online for system to function?

N2N so only 2 parties involved need to be online.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Participants

When is a transaction considered "safe" or "live"?

Deterministically final settled when the counterparty ledger is updated.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No

How much time does a node need to reach a decision?

Instantaneous as soon as nodes agree.

What is the number of current and planned validators?

n2n

Is there a forking vulnerability?

No

What process does the system follow when it receives data?

Each counterparty has their own view. They hash their version of the data and present it to other party to view for comparison. They transmit their version of the hash to counterparty and vice versa. These should match.

How is data currently stored?

Corda has no unnecessary global sharing of data: only those parties with a legitimate need to know can see the data within an agreement. Gets stored in the individual nodes.

How does a party take ownership of an asset?

Off-chain.

Governance, Risks and Control

How is governance / controls enforced?

Corda's design directly enables regulatory and supervisory observer nodes.

How does the consensus mechanism allow access?

Access is granted for nodes involved in transaction.

What is the permission management process? What is the process for adding or deleting nodes?

Implementation dependent. Can have an opt-in for other nodes to be part of a transaction.

How does the protocol assess the trustworthiness of other participants?

By allowing only the counterparties who want to transact be involved assumptions can be can be made around trust in that transaction.

Are there separate admin / administrator privileges? Who manages them?

No

Are there restriction / privacy rights defined and enforced by node?

Yes

What are the measures in place to reduce risk?

The system is closed to only the counterparties so that reduces a lot of risk.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Instantaneous as soon as nodes agree.

Is the speed of the system impacted if the system is made more scalable?

No

Does synchronization have any impact on scalability?

No synchronization N2N.

Security

How is transaction activity monitored?

At each counterparty with full logging of all transactions both successful and attempted/failed/canceled transactions.

Does the consensus mechanism utilize Digital Signatures?

Yes

Which risk/security issues are currently being worked on?

Corda's design directly enables regulatory and supervisory observer nodes.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Not applicable as the counterparties transactions are handled serially.

Privacy

How does the system ensure privacy?

Transactions are N2N and available to the counterparty with which it is shared. No external parties are needed but there is an opt-in for regulators.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes, messages are data encrypted and node to node.

Do all nodes have visibility into all other transactions?

No

How does the data encryption model work?

Node to node.

Are participants' identities hidden from one another? (e.g., Blackpool)

Yes

Cryptography / Strength of Algorithm

What does the key life cycle management look like?

Hash's digital signature cryptographic support, keys generated between counterparties. Multiple different keys b/c each counterparty will have different keys and create new ledger with new set of keys.

Does the consensus mechanism require a leader?

No

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

N2N/2

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Corda has no native crypto currency.

Which security mechanisms are assigned to the tokens?

Corda has no native crypto currency.

Briefly described the life cycle management process for the tokens

Corda has no native crypto currency.

Does the consensus mechanism utilize transaction signing?

Digital Signatures.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Corda used with Barclays for smart contract usage (ISDA) derivatives swaps.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Banks subscribe to service.

Who are you currently working with? (e.g., venture capitalists, banks, credit card companies, etc.)

A consortium of banks.

5. DAG (Directed Acyclic Graphs)

Source: Interview / Questionnaire

Contact name: Aviv Zohar (avivz@cs.huji.ac.il)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

Proof of Work.

Do all nodes need to be online for system to function?

No

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

P2P

How much time is actually needed to build the consensus until a new block is added?

We expect block to be added at a rate of 1 per sec or so. Waiting time for irreversibility is then on the order of several seconds.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

51 percent Attacks.

How are the incentives defined within a permissioned system for the participating nodes?

When we talked last you talked about how the white paper is all about changing incentives for the miners and can you explain how smaller miners benefit too. Rewards are given through: 1. Minting: to everyone who creates a block (not a problem - incentives are well aligned as each block adds to the security of the ledger, and no one can create blocks at will due to the Proof of Work. 2. Tax fees: only to the miner that included the tax in his block. If there are several miners that attempt to include the tax (in conflicting blocks) we can pay only one of them - the one whose block is earliest. Re the transaction

fees: if you now know that your blocks may not be earlier than other conflicting blocks that appear, you have the option of including transactions that pay slightly less but are unlikely to be included by others, thus guaranteeing a higher probability that you will collect payment for them. In contrast, in the vanilla bitcoin implementation, you are incentivized to always include the highest paying taxes, and you lose all rewards if your block is in conflict and you did not eventually win. This behavior can be used by smaller miners (who often lose block races) to mitigate their losses and to still earn more than they would under "vanilla bitcoin". The other implications are that they also include additional transactions in the chain and that conflicting blocks are not very similar (again, in contrast to vanilla bitcoin).

Governance, Risks and Control

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

No. There is none. It is just shown to be hard to do.

How does a party take ownership of an asset?

Same as Proof of Work.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

No. There is none. It is just shown to be hard to do.

How does the consensus mechanism allow access?

Same as Proof of Work.

What is the permission management process? What is the process for adding or deleting nodes?

Same as Proof of Work.

How does the protocol assess the trustworthiness of other participants?

Same as bitcoin.

Are there separate admin / administrator privileges? Who manages them?

No

Are there restriction / privacy rights defined and enforced by node?

Same as Proof of Work.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

1 MB per second of block adding as opposed to every 10 minutes in Proof of Work now confirmation time several seconds.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

2000 TX per second.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Unlimited.

How do you measure scalability?

Blocks are created every second in the system and this allows for double spend to be mitigated.

Is there a limitations on the number of fields within a transaction?

No

Is the speed of the system impacted if the system is made more scalable?

No

Security

Does the consensus mechanism utilize Digital Signatures?

Yes, Same as Proof of Work.

Which risk/security issues are currently being worked on?

Double spending attacks, selfish mining attacks, network connectivity.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

We are indeed using the same infrastructure, and based on Proof of Work, though we are also working on non-Proof of Work based variants (permissioned chains).

Briefly describe the security testing performed till date (if any)

We are proving theorems regarding the security of the double spend attack, and also have begun implementing simulations to verify these claims independently.

How are you planning to implement/integrate Digital wallets? (Including private key management)

We're not dealing with this directly. We plan to adopt Bitcoin's code here.

In case of a breach, what data is at risk?

Generally a breach can only harm the security of one's own wallet.

How does the system prevent signature fraud (e.g., stolen keys)?

Just as in Bitcoin.

Does the consensus mechanism have full documentation in place?

We are not building a complete system, but are working on a paper to explain how to change the consensus core.

How does the consensus mechanism address the risk of "double spending"?

We are using the DAG to order transactions. Double spends receive lower priority and are thus discarded.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

As in Bitcoin, all nodes need to be connected to the main network. We assume a robust P2P infrastructure, through which blocks are downloaded. A full node needs to download all past blocks, while a light node is expected to need to download only block headers (80 bytes per block or so).

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

We do not assume clocks held by nodes are synchronized/accurate.

What is the process for disaster recovery?

Same as Proof of Work.

Cryptography/Strength of Algorithm

How are the keys generated?

Same as Proof of Work.

Does the consensus mechanism require a leader?

No

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

We are token agnostic. Smart contracts/tokens/anything else that would go in the chain does not matter much to us.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Same as bitcoin.

What is the time required to implement?

We have a running version of the code for the consensus core, but a full blown implementation as a currency will probably take on the order of six months to a year to get.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Same as bitcoin and working on solutions for Proof of Work.

6. Derived PBFT (Hyperledger project)

Source: KPMG Research

Contact name: See Contact Us below.

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

PBFT Derived.

How many nodes are need to validate a transaction? (percentage vs. number)

Minimum mathematical one but then you couldn't really leverage PBFT which means you have practical four.

Do all nodes need to be online for system to function?

The mathematical minimum is four due to one-third faulty nodes due to PBFT mechanism; practical beyond four, currently testing with 12 - 15 nodes (depending on use cases).

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes, permissioned system as assumption.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

To really use the Blockchain infrastructure the owner of the network shall own them. Validating nodes = owned by transaction participants, other (reading nodes + regulators) will help to improve the resiliency of the network.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Three rounds (but depends on use case).

How much time does a node need to reach a decision?

That depends on the scenario to be validated, three rounds of network communication and in best case scenario it also depends on latency, but then the validation is happening in milliseconds.

How much time is actually needed to build the consensus until a new block is added?

Depends on use case.

Does system contain synchronous node decision making functionality?

The time-out window depends on the business scenario (today five seconds).

What is the number of current and planned validators?

Current testing with approx. ten but can be more.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

PBFT one third

Is there a forking vulnerability?

Theoretically, yes.

How are the incentives defined within a permissioned system for the participating nodes?

No incentives defined, only cost of participating in the network.

What process does the system follow when it receives data?

Depends on the use case.

How is data currently stored?

Depends on the use case.

How does a party take ownership of an asset?

Depends on the use case.

Governance, Risks and Control

How is governance / controls enforced?

Depends on the use case.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

Depends on the use case.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Depends on the use case.

How does the consensus mechanism allow access?

Configurable.

What is the permission management process? What is the process for adding or deleting nodes?

Since the consensus is a defined by a majority of nodes, it is a prerequisites to know what the network consists of. By removing nodes you could change the consensus and potentially see a split. The membership of the network has to be a meta-layer or overlay of the overall consensus building. Adding a new participant has to be a transaction which is propagated in the network and validated by the other nodes. Removing a node is a bit more complex: because if a node is not replying anymore it cannot just simply be removed. The actual removal of a node would need to be proposed as a transaction to the network and to be agreed upon. Once the new status is confirmed (the nodes come to the agreement that the new status doesn't include the node) you can move to a new system status and the node is removed.

Are there separate admin / administrator privileges? Who manages them?

That is use case dependent, technically it is configurable; ideally that is a central authority or potential other appropriate governance structure (maybe even a regulator).

Are there restriction / privacy rights defined and enforced by node?

We rely on various cryptographic concepts, but that is configurable by use case. We can establish a bilateral exchange of information; we have got a few options, we can replicate and encrypt data in various ways and so that is case by case dependent, but it needs to have a mechanism to also ensuring the nuances in the industry are addressed by various layers (announcements will be made third week of April).

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back office outsourcing)

Yes, rings of access management: outer = reading access and inner layers = validating and writing access where inner layers have the greater degree of access.

A reading node would get a copy of the event, but they are not validating or processing but have the capability still to communicate with other nodes.

How is counterparty risk / settlement risk addressed?

That is separate from the consensus mechanism that would need to be checked as part of authorization before you go to the consensus stage.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Depends on use case, but milliseconds.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

Depends on use case.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Depends on use case.

How do you measure scalability?

Transaction throughput in seconds, and number of nodes able to participate in validation.

Is there a limitations on the number of fields within a transaction?

Depends on use case.

Security

How is transaction activity monitored?

Depends on use case.

Does the consensus mechanism utilize Digital Signatures?

Yes, distributed signing as second step. That service will receive transaction, check and determine if it can authorize and that can check in with existing BO systems. Technically, it is not part of the actual consensus mechanism; it is a second step to distributed signing, transactions once confirmed will then be added. Steps (increasing complexity) distribution of data, distribution of signature, distribute consensus, distribute of business logic.

How does the consensus mechanism address an assumed industry standard?

Hyperledger project.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

Hyperledger project, aiming for project participants' review; are aiming for certification by Hyperledger foundation.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

Depends on use case.

How are you planning to implement/integrate Digital wallets? (Including private key management)

Depends on use case.

In case of a breach, what data is at risk?

Depends on use case.

Does the consensus mechanism have full documentation in place?

Yes

What is the process for disaster recovery?

Reproduce the key from the archive, going forward you use rolling keys.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

It creates an entirely new class of threat denial of service, like a node is not playing the rules.

Cryptography / Strength of Algorithm

How are the keys generated?

Generally, we limit the chances of keys getting lost and also limiting the potential for strong keys due to the potential vulnerability. Keys will be generated out of master keys in a hierarchy.

What does the key life cycle management look like?

Via the hierarchy, the master of a higher key can create a new key a lower level down; moving down the hierarchy is simple, moving up is impossible. The master key is owned and will be commonly shared via fractural pieces and will be reconstructed if lower hierarchy keys will need to be recreated. If we would use random keys it increases the chances we would lose some of them or they would be corrupted. A hierarchy system of creating keys which are used for a single transaction is safer. The hierarchy can also be used to roll daily and monthly keys. At some point, we have to store some root keys and they are stored not in one piece. That makes it more flexible.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

Supermajority as PBFT characteristics, one third

The system is flexible; phase 1 (Hyperledger project): central authority certificate determines who participates like DTTC; phase two: decentralized authority on the basis of democratic votes.

Evaluating the adjustment of the strictness is a theoretical boundary, we cannot be less strict than the one third otherwise we lose the properties, practically the one third is the upper limit in practice within the industry would require stronger mechanism, in reality if only one bank behaves suspicious, intervention manually first then later system defined will happen, system continuity has priority.

How is node behavior measured for errors?

An administrator monitoring currently build in, every node is / audit log and audit trail.

Tokenization (if used)

Does the consensus mechanism utilize transaction signing?

Yes, but depends on use case.

7. Distributed Concurrence

Source: Interview / Questionnaire

Contact name: Dan Conner (dan.conner@disledger.com)

Questionnaire responses

Consensus Methodology

How many nodes are need to validate a transaction? (percentage vs. number)

Nothing happens outside of the two parties. two counterparties no independent validator and no independent auditor. Regulators data store.

Do all nodes need to be online for system to function?

Bo just the two nodes involved in the transaction.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Participants on the nodes, they implement in best manner for themselves. Cloud service and data center.

What are the different stages involved within the consensus mechanism?

Three stages: 1) Transaction Concurrence 2) Chain Concurrence Repeated Periodic Polling of hash.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

See process flowchart for transaction and chain concurrence flows.

If applicable, what is the voting process after the "propose" stage?

See process flowchart for transaction and chain concurrence flows.

When is a transaction considered "safe" or "live"?

One of the key differentiators of concurrence ledgers is that the transactions are definitively and deterministically final settled when the counterparty ledger is updated. This feature is something that consensus Blockchain cannot offer and is critical in securities transactions.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No consensus.

How much time does a node need to reach a decision?

Time to reach a decision is determined by physical distance between the servers.

How much time is actually needed to build the consensus until a new block is added?

Instantaneous node to node.

What is the number of current and planned validators?

No external parties have a say in the concurrence model so no one can corrupt, delay, or refuse to provide consensus (e.g., a denial of service via denial of consensus voting).

Is there a forking vulnerability?

No, there is no Blockchain so no possibility for a fork. The maximum deviation is one transaction could be proposed that is not accepted by the other counterparty. It would be caught by either the next periodic hash check, or when the next transaction is proposed whichever comes first. But no other transaction can be entered into until the discrepancy is resolved and the ledger in error is brought into synch.

How are the incentives defined within a permissioned system for the participating nodes?

Node to node so incentive is private trades.

What process does the system follow when it receives data?

Each counterparty has their own view. They hash their version of the data and present it to other party to view for comparison. They transmit their version of the hash to counterparty and vice versa. These should match.

How is data currently stored?

Data is stored in the nodes that do the transaction. They own the data.

How does a party take ownership of an asset?

Ownership of assets: work w/off chain assets, on ramp how do you get your physical assets, needs to be trust there that asset will be linked-trust created in audit process, trust gov entities, account firms, internal/external auditors are doing their jobs to verify in good faith that counterparties and assets are real.

Governance, Risks and Control

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

Settled off-chain. Legal recourse.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

No external parties have a say in the concurrence model so no one can corrupt, delay, or refuse to provide consensus (e.g. a denial of service via denial of consensus voting).

How does the consensus mechanism allow access?

Access implementation: run some code base create counterparty ledger for new person. N-1 number of ledgers, actual number of counterparties: ledgers created are 1-1 so n-1 for you to create with all counterparties.

How does the consensus mechanism restrict access, concerning malicious activities?

One key feature is that the concurrence ledger will lock because the ledgers don't hash equally. So malicious transactions won't be able to be applied no matter how many times they are attempted. Malicious activity detection is automated with the periodic ledger hash checks. And the bad transaction is always at the top of the chain on only one side with log entries recording the activity so rollback is very easy to clean up the system post-attack.

What is the permission management process? What is the process for adding or deleting nodes?

It can be implemented in various ways depending on the use case, but yes N2N makes it simpler.

How does the protocol assess the trustworthiness of other participants?

Trust is derived from the regulated members' compliance departments, auditors, examiners and regulators oversight.

Are there separate admin. / administrator privileges? Who manages them?

No consensus protocol but the concurrence protocol is patent pending and may become industry standard.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back office outsourcing)

The two counterparties have complete access to data no one else on chain does unless you want them to.

If the functional area doesn't need access to live data they can operate off read only, backup data.

What are the measures in place to reduce risk?

The system is closed to only the counterparties so that reduces a lot of risk vectors. Additionally, at each level steps can be implemented to protect the system from physical isolation to active network and transaction monitoring.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

Handled by the members directly – assumes entity performs its KYC/AML checks prior to establishing a counterparty ledger.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Instantaneous.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

Not a monolithic Blockchain - 100,000s TX per sec simultaneously on individual ledgers as well.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Structure less data no limit on the amount of data.

How do you measure scalability?

Because the counterparty ledgers can all be separately processed the system is n-times more scalable than a consensus network (where n is the number of nodes in the

network). Additionally, since all of the delay in seeking consensus is eliminated each transaction is much faster as well.

Is there a limitations on the number of fields within a transaction?

No

Is the speed of the system impacted if the system is made more scalable?

No

Does synchronization have any impact on scalability?

No synchronization N2N.

Security

How is transaction activity monitored?

At each counterparty with full logging of all transactions both successful and attempted/failed/canceled transactions.

Does the consensus mechanism utilize Digital Signatures?

Yes

How does the consensus mechanism address an assumed industry standard?

No

Which risk/security issues are currently being worked on?

A holistic approach evaluating from the physical layer (what fiber bundle will interconnect counterparties) all the way to feasibility studies of AI at the members' node for deep learning of usual/unusual activity.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

Different routes are possible for different industries and use cases (e.g., ISO 20022 messages for payments but aren't needed by exchanges/clearinghouses) so it will depend on customer requirements.

What are the infrastructure hosting options? (e.g., cloud, hosted in a data center, etc.)

System can be cloud, data center or individual server based depending on scale, number of processers/threads active, and counterparty proximity requirements.

Briefly describe the security testing performed till date (if any)

Haven't deployed and tested yet.

How are you planning to implement/integrate Digital wallets? (Including private key management)

No wallets planned at this time as it is a B2B system.

In case of a breach, what data is at risk?

Depends on the severity of the breach.

How does the system prevent signature fraud (e.g., stolen keys)?

Key management is critical with any system. In this case, we have the ability to terminate and create new counterparty ledgers easily since only the specific participants are affected. Additionally, as each counterparty ledger can be controlled by a separate key, loss of a single key would be limited to that single ledger and not affect the entire system.

Does the consensus mechanism have full documentation in place?

Currently in development.

How is the system expected to address general server issues?

Can be hosted in cloud, data center, or individual servers for small enterprises. Expect the usual redundancies and backups to be in place. Additional benefit is that since this is for transactional systems old data can be moved to secondary servers so the data is still accessible but not clogging the operational servers.

How does the consensus mechanism address the risk of "double spending"?

There is no inherent crypto currency required so no chance for a double spend.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Not applicable as the counterparties transactions are handled serially.

What is the process for disaster recovery?

Backup server.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

Currently working the details on a reputation score for counterparties – e.g., if a counterparty ledger fails a periodic hash, notify/poll other members of the network to determine if it’s a larger issue or just that one transaction that is an issue. Since only one transaction can be improperly added to a counterparty ledger without concurrence the total risk is capped.

Privacy

How does the system ensure privacy?

The ledgers are only available to the counterparty with which it is shared – no external entities have access.

Do all nodes have visibility into all other transactions?

No

How is privacy defined and ensured between applications?

N2N

How does the data encryption model work?

Encryption can be applied on each counterparty ledger and is easily managed because only the parties need to agree to change encryption methods, etc., they don’t have to wait for the entire network to make system-wide changes.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

No

Are participants' identities hidden from one another? (e.g., Blackpool)

No for KYC/AML counterparties are completely known. (Pseudonymity fails after the first transaction with a counterparty in a permissioned network so this may not be relevant.)

Cryptography / Strength of Algorithm

What does the key life cycle management look like?

Hash’s digital signature cryptographic support, keys generated between counterparties. Multiple different keys b/c each counterparty will have different keys and create new ledger with new set of keys.

Does the consensus mechanism require a leader?

No

How is node behavior measured for errors?

Immediate dispute resolution node behavior. Need to only look for errors in last transaction to see the difference. Authentication at the individual level.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Everything is representation of an off chain asset, validated by the auditors, examiners, etc. There is no inherent cryptocurrency that adds risk for the participants due to value fluctuation or during the state transition.

Does the consensus mechanism utilize transaction signing?

Digital Signatures.

8. Evernym

Source: Interview / Questionnaire

Contact name: Jason Law / Timothy Ruff / Drummon Reed (jason@evernym.us / timothy@evernym.us / drummond@respect.network)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

We have implemented a BFT protocol called Plenum which is a variation of the Redundant Byzantine Fault Tolerant protocol (http://pakupaku.me/plaublin/rbft/report.pdf). Major differences include no reliance on MAC authenticators. We are using Ed25519 digital signatures throughout. Also, RBFT doesn't specify a method for election, so we've implemented one consistent with Byzantine Agreement (BA) protocol itself.

How many nodes are need to validate a transaction? (percentage vs. number)

Like other BFT protocols, RBFT relies on 2f+1 nodes to come to agreement. There is one important variation: double-spend-proof (DSP) transactions, such as the creation of an identifier, must be executed across all nodes. However, where DSP is not required, say in the case where attributes are added to an existing identifier, then a subset of nodes are used to validate and execute the transaction. This is one of the extensions to RBFT we have that allows us to scale BFT, a protocol that is traditionally hard to scale.

Do all nodes need to be online for system to function?

No, a little more than 2/3 of the nodes need to be online for the system to function. In fact in a system of n nodes, floor (n-1/3) can go down and the system will still function.

A pilot is underway now. Several financial institutions, life management platforms, universities, and other interested parties, including a large data center are participating. These exist in Europe, Asia, and North America. They span multiple operating systems (Windows and Linux), virtual and dedicated machines, cloud and data centers, and administrative zones. In this phase, load testing and PoC for use cases are underway.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes and No. Yes, some nodes and their Stewards (operators of Validator nodes) are bootstrapped in a set of "genesis transactions". No, in the sense that those nodes and stewards can change over time. There is a special Pool Management ledger—separate

from the main transaction ledger—for governance of how Validator nodes are added to the network. New Stewards and the requirements for their nodes are also managed by this governance process, as is revocation of Stewards. Various algorithmic and time-based mechanisms will be employed, but there are cases where node membership will require votes from existing membership, perhaps even leveraging some kind of limited proof-of-stake for pool membership. This aspect of the governance model is still under development (see below).

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Validator nodes are controlled by stewards, which are selected per the governance mentioned above. There is a second type of node called an "Observer", who does not participate in the consensus process but one keeps track of all the transactions that the consensus pool has successfully processed. The purpose of Observer nodes is to provide a service to any party who is concerned with only reading "safe/live" transactions. This type of node reduces the load on the Validator node so the Validator node can only focus on providing consensus.

What are the different stages involved within the consensus mechanism?

Clients submit requests to at least one node, and the nodes verify the client signatures and propagate the client requests to the other nodes. When enough requests have been propagated, the primary (leader) starts the three-phase commit.

The consensus mechanism currently uses a three phase commit which can tolerate Byzantine faults. The three phases that any transaction goes through for consensus are: Phase 1. The first phase is PRE-PREPARE phase (read can Commit according to standard three phase commit terminology) in which the primary sends a PRE-PREPARE message to all non-primaries to tell them about a new transaction which is coming for consensus.

Phase 2. The non leaders respond with a PREPARE message to the primary demonstrating their willingness to accept this transaction in this phase. Phase 3. In the third phase, all participants (primary and non-primaries) give their consent to this transaction in the form of a COMMIT message. If a Byzantine quorum (2f+1) of COMMITs is achieved, then the transaction is executed.

One distinction, between RBFT (the basis for Plenum) and PBFT is that there are f+1 parallel RBFT “instance”' that execute the same three-phase-commit with a different primary. There is one Master and multiple Backups. If the Backups detect performance anomalies or favoritism or other abnormalities with the Master, then through consensus, a new Primary is elected for the Master instance.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

Precondition for PREPARE. Any non leader who gets a PRE-PREPARE from a leader will send a PREPARE if it has enough confirmations from other nodes that the request actually did come from a client in the system and not by some malicious leader who sent a fake PRE-PREPARE. The threshold is f+1, f being the maximum failures the system can

tolerate. Precondition for COMMIT. Any node (leader and non leader) will send a COMMIT if it sees quorum of PRE-PREPARE and PREPARE (2f+1, 1 PRE-PREPARE and 2f PREPAREs) from different nodes in the system.

If applicable, what is the voting process after the "propose" stage?

If propose phase refers to the can Commit phase (PRE-PREPARE in RBFT) then there are two more phases after the can Commit phase (PRE-PREPARE). Refer to the answer of the "current stages" question above.

When is a transaction considered "safe" or "live"?

As soon as the three-phase-commit is completed on a request, it is executed. Execution involves writing to a physical ledger on disk. As soon as this is done, it is considered safe or live. This is considerably faster than Bitcoin transactions.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No. Although there are three phases, we don't have multiple rounds of consensus like Ripple.

How much time does a node need to reach a decision?

Less than one second with network latency.

How much time is actually needed to build the consensus until a new block is added?

We do not put transactions in blocks. We split transactions into multiple files, but that is not material to the protocol. We have implemented a perpetual Merkle tree, much like the one employed in Google Certificate Transparency. This tree grows predictably, which allows for fast audit (inclusion) proofs and consistency proofs and verification of those proofs. We do employ batching in several places for performance; for instance, the three-phase-commit can process multiple ordered and validated transactions in a single batch.

Does system contain synchronous node decision making functionality?

Yes. Also say 2 transactions T1 and T2 were submitted to the same sub pool at times t1 and t2 where t1 < t2. The system guarantees that T1 will complete before T2 completes.

What is the number of current and planned validators?

Current is 10 with a set of pilot validators mentioned above. The expectation is we'll have from 60 to 120 world-wide. Testing and usage will reveal the right number.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

More than one-third of the nodes need to be compromised before the system is considered unreliable (non functional).

Is there a forking vulnerability?

No.

How are the incentives defined within a permissioned system for the participating nodes?

This is not in place today; we are in discussions about how the incentive system will work. We do not have a crypto-currency, but usage of credit or coupons and some level of accounting are part of the discussions.

What process does the system follow when it receives data?

The process is: 1) validation of the request, 2) verification of the signature, 3) authorization of the request, 4) propagation of the request to other nodes. Then it starts the three-phase-commit. See the answer to the 'current stages of mechanism' question for more details.

How is data currently stored?

There are five data structures:

1. A ledger for Pool Management transactions.

2. A ledger for Domain transactions (in the case of Sovrin, these would be identity transactions). The domain transaction ledger can be broken down further. It will have one ledger for double-spend-proof transactions (like identifier creation), and then each sub-pool would have its own ledger that would be propagated to other sub-pools after execution.

3. Indexes of ledger transactions. These leverage the same database as the attribute graph (below), but can be broken out.

4. Attribute graph stored in a mutable database. Today, this is OrientDB, but it is not implementation-dependent.

5. Any external feeds and data stores to which links are made.

How does a party take ownership of an asset?

No private keys are ever created by a third party and shared with another. However, there is the concept of an identity "sponsor", where a sponsor bootstraps an individual. The individual may provide a public key at the time of bootstrapping, or it may happen at a

later time. If it happens at a later time, there is an equality transaction that assigns an individual's public key to an existing identifier.

Governance, Risks and Control

How is governance / controls enforced?

We are currently fleshing out governance in the form of a Sovrin trust framework being developed with the help of seasoned industry specialists. The Sovrin ledger divides responsibilities among two kinds of nodes, Validators and Observers. Validators are the nodes that take part in the consensus process. Observers nodes only provide the results of already executed transactions (i.e., they are a "read-only" layer that supports caching/scaling). Validators and Observer nodes are operated by Stewards.

Sovrin governance will specify criteria for adding and removing Stewards, and for Stewards to add or remove Validator and Observer nodes. It will define the requirements for diffusion of Stewards so it is extremely difficult for one or a small set of colluding organizations to take control of the whole system. The governance model will also define the fees to be levied on different transactions to provide the right balance of incentives. Lastly, the governance model will also define the criteria for releasing new code for the nodes.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

The Sovrin trust framework will define the parameters for legal liability in case of malicious actions. It will also define how blacklisting will work upon detection of malicious behaviors by nodes and clients.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

When attempts to introduce errors in consensus process are detected, the suspected parties can be blacklisted for short or long term depending on the severity of the suspicion.

How does the consensus mechanism allow access?

Access at the Steward level will be governed by the Sovrin trust framework and the special governance consensus pool. Access at the client level is available to registered clients by virtue of their control of the private keys necessary to sign transaction requests to the consensus pool.

How does the consensus mechanism restrict access, concerning malicious activities?

Nodes and clients can be blacklisted (different levels of blacklisting can come into action depending on the type and persistence of maliciousness).

What is the permission management process? What is the process for adding or deleting nodes?

See the Sovrin governance process described in C30 above.

How does the protocol assess the trustworthiness of other participants?

Participants are deemed trustworthy unless some suspicion is reported against them. Participants can be suspected to be malicious based on various behaviors including slow response times, inappropriate messages attempting to disrupt the consensus process, flooding, etc. Every suspicious activity has a severity attached to it and that severity forms the basis of the action taken suspect.

Are there separate admin. / administrator privileges? Who manages them?

The provider of a node reserves the administrator rights of the node like starting, stopping, restarting, upgrading (hardware and software).

Are there restriction / privacy rights defined and enforced by node?

Only those transactions that need to be double spend proof need all nodes to be part of the consensus process. Other transactions involve specific subset of all the nodes for the consensus process.

Can a node or a user have only "Read" or only "Write” access? Is specific node access required if only performing one functionality? (e.g., back-office outsourcing)

Currently, all Validator nodes function in Read/Write mode. Observer nodes function as read-only nodes since their job is alleviate the load on the Validator nodes. Observer nodes can be used by any clients of the system who want to query for the transactions already processed using consensus.

What are the measures in place to reduce risk?

We avoid correlation by allowing a different key for each transaction a client makes. Our strong encryption is much harder to break than many existing encryption schemes like RSA, DES, etc. We use digital signatures for node-to-node and node-to-client communication instead of MAC authenticators. Finally, the Stewards themselves form a trust network who share the incentives to reduce risk across the system.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

We will let financial institutions drive these requirements, including where the data is stored, how it is disclosed, and to whom. This can be an excellent application for the work we're doing with Zero Knowledge Proofs.

How is counterparty risk / settlement risk addressed?

Sovrin does not handle settlement. However, the parties involved in a transaction can use strongly proofed identity, and Sovrin/Plenum can provide triple-signed-receipts which provide all parties with proof of transaction, even if that transaction is not stored in a distributed ledger.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

- The client request is usually validated in a few milliseconds.

- It takes around one second for the pool to reach consensus.

So transactions take in order of seconds as compared to minutes for Bitcoin-based or other proof of work related Blockchain.

How do you measure scalability?

Scalability for a Plenum system may be defined as the ability of the system to handle an increasing number of client requests in the same amount of time.

Any reasonably large application built using Plenum must have multiple consensus pools. The reason is that, unlike traditional Web applications, adding more nodes to an RBFT consensus pool increase the fault tolerance but slows down the pool due to increased network traffic.

So the way to scale up the system to handle more transactions per minute is to add more consensus pools. Each pool gets its own ledger. Ledger transactions are synchronized independently and asynchronously.

Is there a limitations on the number of fields within a transaction?

No. But there is a cost associated with additional attributes.

Is the speed of the system impacted if the system is made more scalable?

Scalability of the system is primarily achieved by adding more consensus pools to the system as mentioned above. The effect of adding more consensus pools on speed should be negligibly small.

Does synchronization have any impact on scalability?

No, as soon as a node has caught up on any transactions it has missed, it starts working like any other node in the system.

Security

How is transaction activity monitored?

A monitoring system tracks the throughput of nodes and their latencies in processing transactions across all clients as well as per individual client.

Does the consensus mechanism utilize Digital Signatures?

Yes, all communication between nodes is digitally signed by the nodes.

How does the consensus mechanism address an assumed industry standard?

Plenum is based on RBFT which is a more robust version of PBFT.

PBFT and RBFT share the same three-phase commit protocol along with the view change mechanism. However PBFT is not as robust as RBFT.

First, as described in the Aardvark paper (Clement et al., Making Byzantine fault tolerant systems tolerate Byzantine faults, NSDI '09), a malicious client can trigger view changes at will that will stop the progression of the protocol.

Second, from an implementation point of view it does not separate the logic of accepting client requests and ordering them, which leads to possible DoS attacks from the client.

Third, a malicious primary can order requests at an arbitrary speed without being detected. These problems are fixed in Aardvark. RBFT improves Aardvark by executing several protocols in parallel to detect any performance problem in real-time, without assuming anything about the previous or future performance/condition of the system.

Finally, Plenum further improves upon RBFT. For instance, it uses Ed25519 instead of MAC authentication. Also, Plenum is more scalable than plain RBFT. See the questions on Scalability for more details.

Which risk/security issues are currently being worked on?

One of the biggest risks with a public ledger focused on identity is correlation. We're working on anonymous authentication and credentials to reduce correlation risk for the system as a whole.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

We would advise not to run Plenum or Sovrin nodes on any machine which is also serving some other application. Hosting a node on any cloud environment is fine.

Briefly describe the security testing performed till date (if any)

Plenum's source code (https://GitHub.com/evernym/plenum) includes a plethora of tests that describe various faults (malicious and nonmalicious) and Plenum's resistance to them. We are also working with the authors of RBFT and other cryptographers for ideas and extensions to the protocol to increase security, privacy, reliability, and robustness.

How are you planning to implement/integrate Digital wallets? (Including private key management)

We are exploring integration opportunities with existing digital wallets.

In case of a breach, what data is at risk?

A breach is possible but wouldn't compromise the system as the data that is meant to visible to only a selected number of parties is encrypted using symmetric key encryption and the keys are disclosed to only those parties for whom the transaction is meant. Also the disclosure of keys is made using asymmetric encryption. So the system does not provide any incentive for a breach.

How does the system prevent signature fraud (e.g., stolen keys)?

If a key theft is reported then those keys would be blacklisted in the system so no further transactions made using those keys will be successful.

Does the consensus mechanism have full documentation in place?

Not complete but quite extensive. Plenum has a wiki and also documentation for the source code.

How is the system expected to address general server issues?

The Sovrin trust framework will defines SLAs including minimum hardware and software requirements that Validator and Observer nodes must satisfy to be a part of the network. The protocol tolerates low-performance Validators, but if there are too many, it can have a negative impact on system performance as a whole. Node performance is recorded on the ledger and can become a criteria for the node to remove from the network.

How does the consensus mechanism address the risk of "double spending"?

The system is double spend proof as more than two thirdsof the nodes need to agree on a particular transaction for it to be written to the distributed ledger.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Each node maintains a Merkle tree which records all transactions that have occurred up to the present. When a node has been unable to receive transactions (because it crashed or is new in the network), it communicates its status to other nodes and receives the updates and Merkle proofs of those updates from every node. When it has received

sufficient Merkle proofs that are consistent, it applies those updates. While the node is in this process of getting updates, it does not participate (give its votes) in the consensus process. However, it will still receive and record every transaction on which consensus has been established. When it is fully caught up it will become a fully participating node.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

The nodes do not depend on clock synchronization to function as of now (although some domain implementations might require the same view of time for each node) since transactions use a mechanism similar to Lamport timestamps.

Under which conditions does a lock/un-lock happen? (i.e., what is the proof safety?)

Nodes and clients can be blacklisted upon suspicious activity. They can be again whitelisted in certain cases where the severity of their suspicious action is not high or their past record has been good.

What is the process for disaster recovery?

Because nodes are geographically distributed, it is difficult for natural disasters, large Proof of Worker/Internet outages (like the ones over entire continents), or other acts of God to bring the system down since the system can function even if one thirdsof the nodes are down. After coming back up, the nodes go through a recovery process in which they are able to get data for the transactions they have already missed (see C65). Also in case of permanent loss of a node, a new node can be added in the system according to the governance model. This new node then starts synchronization with other nodes.

Privacy

How does the system ensure privacy?

Every transaction contains encrypted data and it can only be decrypted by the intended party (the one(s) who have the keys).

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes, all messages between nodes and between nodes and clients are digitally signed using elliptic curve cryptography (Ed25519).

Do all nodes have visibility into all other transactions?

Yes.

How is privacy defined and ensured between applications?

Since transaction data is encrypted, one application can see the data of another application but cannot make sense of it.

How does the data encryption model work?

Symmetric keys used to encrypt data of transaction is stored in the client's wallet and is shared using elliptic curve Diffie-Hellman key exchange, Curve25519. Symmetric keys use XSalsa20 stream cipher (https://en.wikipedia.org/wiki/Salsa20).

If consensus happens in a permissioned network, are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

The client making the transaction can use a different public key for each transaction but the network needs to know beforehand which public keys the client can use.

Are participants' identities hidden from one another? (e.g., Blackpool)

No. Pool membership is highly visible, because they need to be accountable. The membership is managed through governance as defined by the Sovrin trust framework.

Cryptography / Strength of Algorithm

How are the keys generated?

The Steward while setting up the node generates keys for that particular node using a script provided by Plenum. These secret keys are stored on the node and they never leave the node. The public keys of the node are told to the other nodes through a new transaction.

What does the key life cycle management look like?

There are transaction types for key creation, rotation, revocation, and delegation.

Creating a new key is a transaction in the system. If that transaction is successful (consensus is reached), then the nodes begin recognizing that key as a valid key and new transactions can be made in the system using that key. The system also supports transferring control of key(s) from one user to another via a key delegation transaction. In case of loss of keys or accidental disclosure of keys (out of band disclosure), a user can make a transaction to revoke the keys. This will invalidate any future transaction with that key.

What is the library approach?

The library uses elliptic curve cryptography. Curve25519 Diffie–Hellman key-exchange function is used for public key encryption. Curve25519 can compute a 32 byte public key using a 32 byte private key. Given one user's 32-byte private key and another user's 32-byte public key, Curve25519 computes a 32-byte secret shared by the two users. This secret can then be used to authenticate and encrypt messages between the two users. Digital signatures use Ed25519 which is a very fast key generation and signing public key signature system. With Ed25519 signing and verification are both 32 bytes and the signature is just 64 bytes.

What is the HSM integration approach?

We've had some discussion about using HSMs, but Plenum does not support them yet.

Does the consensus mechanism require a leader?

Yes, the algorithm does create leader (called a Primary in RBFT). However, in RBFT a leader can change if it is found to be performing badly without halting the system.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

The number of faulty nodes the protocol can tolerate depends on the number of nodes. Plenum is engineered so that once a satisfactory f value is achieved, the nodes can serve as standbys, much like that of a RAID array.

Reducing the number of required participants in the consensus is definitely possible, and would be a trivial code change; however, we would strongly caution against that since it would affect the robustness of the system.

How is node behavior measured for errors?

All errors, warnings and exceptions are logged to files. Nodes monitor other nodes for their performance as well as other actions. If a node discovers a suspicious behavior from another node, it can blacklist the other node temporarily or permanently depending on the severity of the suspicious action. The discovering node can also communicate to other nodes about the suspicious activity by an offending node. Any node that receives a quorum of suspicious activity messages for a particular offending node will blacklist that node. In this case, even if a malicious node is being selectively malicious, it can still be blacklisted by the whole network.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Plenum does not use tokens; it uses signed transactions. See C43.

Which security mechanisms are assigned to the tokens?

Plenum does not use tokens; it uses signed transactions. See C43.

Briefly described the life cycle management process for the tokens

Plenum does not use tokens; it uses signed transactions. See C43.

Does the consensus mechanism utilize transaction signing?

A client before making a transaction, digitally signs it. Also a Validator node after executing a transaction will sign the transaction. For details of digital signing refer to C25.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

- In Bitcoin, the "longest chain" – the chain with the most proof-of-work – is considered to be the valid ledger

- Crypto-economics –Distributed economic consensus methods

- Central and commercial banks – Ripple, Bitcoin

- Custodian banks – Hyperledger, Ripple Gateways

- Clearing houses – Eris, Ethereum

In banks it can be used for:

- Cross-border Settlement / B2B international transfers

- Improving the SWIFT and correspondent banking network

- Central clearing (e.g., derivative clearing)

- Mortgages

- Designated settlement systems

- Financial and fiduciary computations

What is the implementation cost?

Assuming each system has dual core CPU, 4GB RAM and 30 GB disk, cost on AWS can be $4000-$5000 for one year for a 16 node system.

What is the time required to implement?

Integration time would be 2 hours for each system.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Distributed consensus ledgers (DCLs) can make a difference compared to existing technology. They enable distributed, balanced control to situations where it's currently not possible or easy. This is particularly true where monitoring by a central authority is not feasible or where a centralized control point create unnecessary inefficiencies, costs and barriers. Examples includes international payments, correspondent banking payments, card transactions and international remittances.

DCLs also go beyond the capabilities of existing technology by providing transparency where it has previously been impossible or difficult to achieve. Examples include in anti-money laundering (AML)—an area where DCL's potential is attracting growing interest and investment. For example, the London-based startup Elliptic has harnessed the underlying technology supporting its visualization of the Bitcoin ecosystem to develop a suite of AML services.

Source: https://www.accenture.com/t20151002T010405__w__/us-en/_acnmedia/Accenture/Conversion-Assets/DotCom/Documents/Global/PDF/Dualpub_22/Accenture-Banking-Distributed-consensus-ledgers-payment.pdf

The transactions in Plenum do not involve the expensive process of mining. Also, the time taken to reach consensus is in seconds as compared to nearly 10 minutes on bitcoin-based Blockchain that use proof-of-work. Plenum is cheaper by orders of magnitude as compared to bitcoin (i.e., if an application like bitcoin is built using Plenum).

9. Graphene

Source: Interview / Questionnaire

Contact name: Ryan R. Fox (ryan@ryanrfox.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

Delegated Proof of Stake (DPOS) https://bitshares.org/technology/delegated-proof-of-stake-consensus/

How many nodes are need to validate a transaction? (percentage vs. number)

11 witness nodes is the recommended minimum by the authors. The number of nodes is configurable by a vote of the committee account. An odd number is required to ensure forks are resolved by a majority, not a 50/50 split.

Do all nodes need to be online for system to function?

A single node could sustain the Blockchain, but transaction throughput would be impacted as the non-producing witness time slots when unfulfilled. Communication is between nodes is key. Offline due to network issues may lead to forking, requiring the majority of nodes to validate the longest approved chain.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes, the algorithm is deterministic in its block producer ordering per round. A block producer is a witness, as in it witnesses the transactions on the wire, validates them, and produces a block containing unpublished transaction within its timeslot window (three seconds). A witness set is the group of elected witnesses, as voted on by stakeholders that are able to produce blocks. A round is the production of one block by each witness in the witness set. At the end of a round, the algorithm determines the order of witnesses within the witness set. A witness cannot produce a block back to back, so the algorithm ensures this between rounds.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Witness nodes are operated by individuals. Anyone can run the witness software, but only those with the threshold number of votes approving them to be an active witness are allowed to sign blocks to be accepted by the network.

What are the different stages involved within the consensus mechanism?

Validation of transaction and blocks is done by witness nodes. There is not a proposal of blocks, rather validation by future block producers to extend that chain. There is the concept of a proposed transaction on the Blockchain, whereby a user may submit a transaction proposal to the network looking for a threshold of valid signatures to later reference the transaction and sign it. When the threshold is met, the transaction may be validated. This all happens on chain, rather than needing to send around a partial transaction offline and submit a fully signed transaction to the network. The transaction may have a time lock and expire prior to threshold signatures reached. Also, a signer may update the transaction to remove their signature at any time prior to the threshold being met.

When is a transaction considered "safe" or "live"?

A transaction is considered committed when at least half +1 of active witnesses have appended to the block in question. As all witnesses are validating all transactions, a witness may choose to fork out a given transaction, and subsequent witness nodes will determine which fork to follow by signing their preferred fork.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

The block interval is three seconds. A witness is collecting and validating all transactions it receives, then appends a block containing only the unique transactions within their block at the appointed time slot.

How much time does a node need to reach a decision?

Transactions are received, validated and broadcast to peers. I don't have calculation timing of the validate transaction operation, but it's fast given we tested 3,000+ transactions per second on a Testnet (https://bitsharestalk.org/index.php/topic,18684.msg241050.html#msg241050)

How much time is actually needed to build the consensus until a new block is added?

Each witness is validating each transaction, so they each have their view of state. Consensus is finalized when half + 1 of active witnesses build upon the target block.

Does system contain synchronous node decision making functionality?

I'm not sure how it could be distributed and synchronous at global scale with multiple nodes. Chain state needs to be synchronized thru consensus. Each node independently validates the transactions and blocks presented. Block production will require a block producer to by in sync with their peers to have their block included.

What is the number of current and planned validators?

Default is 11 initial witnesses. Stakeholders elect a witness set (always an odd number) by voting for a slate of witnesses (any number they want to produce blocks for the network). The number of witnesses is based on the stake based weight of the number of witnesses on all slates. The witness set is the highest vote receivers within the maintenance cycle (default is one hour, configurable by committee vote). At the

beginning of a maintenance cycle the votes are tallied and the witnesses are set until the next election.

What is the Fault Tolerance?

How many nodes need to be compromised before everything is shut down? A single node could sustain the Blockchain, but transaction throughput would be impacted as the nonproducing witness time slots when unfulfilled. If by compromised you mean hacked, well the witnesses nodes could continue to produce empty blocks as a DDOS.

Is there a forking vulnerability?

The Blockchain state is deterministic after half +1 of witnesses sign a given fork. Going below a three-second-block interval is not recommended on a global scale due to speed of light latency and transaction broad casts and block replication.

How are the incentives defined within a permissioned system for the participating nodes?

Each witness node is rewarded with the CORE asset from the reserve pool. Each operation performed has a given fee associated. These operation fees are returned to the reserve pool to continue funding the Blockchain.

What process does the system follow when it receives data?

Receive a transaction, if new transaction, validate the transaction, if valid, broadcast to peers. Receive block, validate transaction within with mempool, if valid transaction, remove from mempool, if block contains completely valid transactions, build on, else create fork; wait for block production time slot, add transactions from mempool to new block and publish block to peers.

How is data currently stored?

LevelDB is the database. Only valid blocks are logged to the database to track Blockchain state. All accounts, balances and transactions are held in memory and used to evaluate transaction validity.

How does a party take ownership of an asset?

The CORE asset is used to pay operation fees. The user presents a transaction containing: signed transaction operation [transfer operation (fee, from, to, asset, amount, memo)].

Governance, Risks and Control

How is governance / controls enforced?

Ultimately the active witnesses are in control of the network, because they are producing the blocks. They must be coordinated to ensure they are running compatible code across the network, so as not to produce forks and reach consensus. The Committee can modify Blockchain parameters by voting on them. The witnesses automatically read these values

from the Blockchain at each maintenance window. This allows dynamic Blockchain behavior without witness nodes changing anything in their codebase or running configuration.

Who is responsible and what are they responsible for in case of malicious actions within the network?

The stakeholders elect the witnesses to produce blocks at their appointed timeslots. Failure to do so will get them voted out. A bug in the code will require developer intervention to resolve, post and witness nodes to implement. A single node running patched software can further the chain.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Incentive to maintain the Blockchain and one's reputation have thus far prevented corruption of the block producers and the Blockchain. A single witness behaving badly will be spotted and voted out. Block producers following that misbehaving node will not build on their chain, thus nullifying their attempts. This is DPOS, not Proof of Work, so one must corrupt multiple individuals, not just acquire hash Proof of Worker to control the network.

How does the consensus mechanism allow access?

Anyone can run the witness node software, but only achieving a threshold of votes from the stakeholders can one produce a block.

How does the consensus mechanism restrict access, concerning malicious activities?

Block producers that issue blocks on multiple forks will be voted out. This resolves the nothing at stake problem found in many POS implementations. One cannot create a longer chain in secret and present it, as the witness timeslots are fixed and all nodes validate all transactions and blocks, then add their signature.

What is the permission management process? What is the process for adding or deleting nodes?

This is dynamic and handled on chain using a voting process. A stake holder may vote for any number of witnesses. Their vote is cast based on their total CORE asset holdings. The vote is for the entire slate of witnesses, not spitting the stake across them. So holding 1000 CORE and voting for 11 witnesses allocates 1000 votes to those 11 witnesses. Changing the vote to 22 different witnesses later then moves those 1000 CORE votes as defined. The user retains their CORE asset and does not lock them up in this vote process. If the number of CORE changes in the users account, their votes are automatically adjusted to match their CORE holdings. Votes for the Committee are handled similarly.

How does the protocol assess the trustworthiness of other participants?

Trust for stakeholders is primarily with the witnesses, as they produce the blocks. Less so, with the Committee, as they set the Blockchain parameters.

Are there separate admin. / administrator privileges? Who manages them?

Separation of duties is a design goal: Developers write code. Witnesses watch network, validate transactions and blocks, Committee adjusts Blockchain parameters. Stakeholders elect who they want to perform the tasks.

Are there restriction / privacy rights defined and enforced by node?

Anyone can run a witness node. Only the elected witnesses may produce a valid block at their appointed time slot.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

Not part of the code base.

How is counterparty risk / settlement risk addressed?

Witness nodes validating the Blockchain state.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

The default Block interval is three seconds. This can be adjusted by the committee at any time, or defined in the genesis if starting anew. Three seconds is close to the global minimum given latency replication constraints.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., #number of trades)

See BitShares and Steem for this information, as Graphene is just a toolkit, not a Blockchain implementation.

How do you measure scalability?

See peek stats: http://bitsharestalk.org/index.php?topic=18684.0

Is the speed of the system impacted if the system is made more scalable?

"Graphene Design Goals:

Limit Database Disk I/O (only log new blocks to DB)

Keep State Objects in Memory (name, key, balances < 1KB)

Minimize Hash Operations (Favor deterministic objects)

Single Thread (queue based preprocessor)

Separate Validation from State Changes

Keep Transaction Interpretation Deterministic and Explicit (end state determined by TX inputs)

Separated Permissions from Identity (Hierarchical Threshold Multi-Sig)"

Does synchronization have any impact on scalability?

Because Graphene is designed to hold the Blockchain state within memory, the validation of transaction is as fast as validating the signature of the transaction, evaluating the operations within and updating the balances referenced. Very few operations are signature validation, as the objects within Graphene are represented as an objectID (see above) not a hash. The objectID are linked to a key and this does get validated, but the objectID for an account is only 48-bits rather than 20 or 32-bytes in Bitcoin and Ethereum. Huge savings on the wire and in computational resources.

Security

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

Cross platform for Linux, Mac and Windows.

How are you planning to implement/integrate Digital wallets? (Including private key management)

Account security is a key design goal within Graphene, and thereby the tokens each holds. Accounts should be setup using hierarchical threshold accounts. This novel technology allows an arbitrary depth of accounts with approval weighting required to spend given balances within the account (https://bitshares.org/technology/dynamic-account-permissions/). Continue reading under Steem, as this applies to Graphene, Steem and BitShares.

In case of a breach, what data is at risk?

If the Owner key is breached, all assets held by that account are at risk, as the attacker may produce transactions using the Owner key. If the Active key is breached, the attacker may produce transaction, but will lose that ability when the Owner key is used to assign a new Owner key to the Named Account. Similar for memo, reading this field within previous transactions.

Does the consensus mechanism have full documentation in place?

Yes. http://docs.bitshares.eu///index.html

How does the consensus mechanism address the risk of "double spending"?

Transaction and block validation by witnesses.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Nodes use NTP for time sync and produce blocks at their deterministic timeslot.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

NTP

Under which conditions does a lock/un-lock happen? (i.e., what is the proof safety?)

Only validated blocks are written to disk, all other chain state is held in memory for validation prior to the disk write.

What is the process for disaster recovery?

A failed node can download the entire chain from peers if their database is corrupt or bootstrap. Better is to validate existing database, then catch up the delta blocks by validating those sent by peers.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

Witnesses are elected by the stakeholders. Each is assumed to act to protect the network, else will be voted out of this role.

Privacy

How does the system ensure privacy?

Largely, privacy is an illusion. All public Blockchain suffer from heuristic scanning to infer transaction outcomes. It is incumbent upon the end user to maintain the level of privacy they are capable of. Telling a user their transactions are securely private is a fallacy that Graphene would rather call out and demonstrate that accounts and balances are public. Using unlinked accounts in an attempt to produce private transaction is possible on any Blockchain, it just requires end user management. The memo field of a transfer operation can be encrypted and readable only by the parties to the transaction. See BitShares for an implementation of Stealth Transactions.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes, witness nodes publish their publishing key to the Blockchain. The network uses this publishing key to verify it was signed by the proper witness at the given time slot. A witness may update their signing key at any time, the peers will see the update transaction and update their chain state to recognize the new signing key (held in memory).

Do all nodes have visibility into all other transactions?

Yes, all nodes can validate any transaction on the wire. Only block producers can submit them within a block.

How is privacy defined and ensured between applications?

OpenSSL for RPC and WebSockets between API server and wallet.

How does the data encryption model work?

Same as Bitcoin, secp256k1.

Are participants' identities hidden from one another? (e.g., Blackpool)

No, named accounts and values are contained within transactions, public knowledge.

Cryptography / Strength of Algorithm

How are the keys generated?

Same as Bitcoin, secp256k1.

What does the key life cycle management look like? See BitShares column.

Substitute CORE for BTS.

What is the library approach?

The code is C++ building atop Boost and FC (Larimer, et al.). The command line client wallet is a separate executable from the witness node Blockchain executable. The witness node allows an API endpoint to be exposed through web sockets to the client. A Node.JS light client is also available, where by the keys remain within the browser and the chain state is from a trusted API server running the witness node executable.

Does the consensus mechanism require a leader?

Original author is Danial Larimer of Cryptonomex, Inc. There are a handful of active developers.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

A key concept of Graphene is flexibility of Blockchain parameters. Fees, number of witnesses, block interval, block rewards, etc. are all configurable by the committee, which is separate group of elected stakeholders from the witnesses, but they do not receive any rewards, but the ability to manipulate the global Blockchain parameters by vote and applied in a maintenance window.

How is node behavior measured for errors?

Witness nodes that sign on multiple forks will be voted out of the witness role.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Tokens are "Assets" within Graphene, a special objectID defined within the Blockchain. The default asset is "CORE" and is objectID 1.3.0 and each new asset introduced by the committee will increment the instance ID (third part of the dotted notation). See the reference for objectID in Graphene that describes this format.

Which security mechanisms are assigned to the tokens?

The CORE asset is defined in the genesis block. All other assets are defined by the issuer. The issuer may configure the assets parameters as they wish: asset symbol, description, number of tokens, precision, fees, etc.

Briefly described the life cycle management process for the tokens

For all assets OTHER THAN CORE, the issuer has great control over the asset throughout the life cycle. White lists / black lists may be defined, revocation of tokens, issuance of additional tokens, and much more.

Does the consensus mechanism utilize transaction signing?

Yes, transaction may be proposed and signed on chain.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

BitShares, Steem, soon PeerPlays.

What is the implementation cost?

Free, License is MIT.

What is the time required to implement?

Depends on project goals.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

See BitShares vs. many other decentralized exchanges. See Steem vs. Reddit.

10. Juno

Source: KPMG Research

Contact name: See Contact Us below

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

BFT "Hardened" Tangaroa variant.

How many nodes are need to validate a transaction? (percentage vs. number)

A simple majority is all that is needed to make progress, but the number of nodes that can vote is something that can change while the system is running (i.e. adding/removing nodes).

Do all nodes need to be online for system to function?

No, but a majority need to be running to make progress. We've tested out clusters of 50 nodes previously.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Participants – you're not much of a distributed system if the provider needs to hold the nodes… why bother with Blockchain at all, just centralize the solution.

What are the different stages involved within the consensus mechanism?

In BFT Raft, each node is in one of the three states: leader, follower, or candidate. Similar to Raft, BFT Raft divides time into terms, which start with an election. The winner of the election serves as the leader for the rest of the term. Sometimes, an election will result in a split vote, and the term will end with no leader and a new election will be held.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

To become Leader, a node must gather a majority vote from the cluster. To become a Follower, a node must receive a set of votes (representing a majority) from a node that is declaring itself leader. As all messages are signed, the Leader converts nodes to follow it by retransmitting the votes it received. To become candidate, a follower must encounter an election timeout (not heard from leader in X time) and not have received a request vote from another node (in this case the follower issues its vote to the candidate and stays a follower).

If applicable, what is the voting process after the "propose" stage?

After a leader is elected, every node transmits its Append Entry Response to every other node and every node awaits such evidence to decide when to increase its commit index. The answer to this question is technically "N/A" but this second process is somewhat similar to what is being asked.

When is a transaction considered "safe" or "live"?

When it is committed to the log (fully replicated). Each node independently decides when to commit a log entry based on evidence it receives from other nodes.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No

How much time does a node need to reach a decision?

~5ms for full consensus + network latency.

How much time is actually needed to build the consensus until a new block is added?

Every log entry is individually committed, incrementally hashed against the previous entry. ~5ms for a single log entry to go from leader receiving the entry to full consensus being reached + network latency.

Does system contain synchronous node decision making functionality?

No

What is the number of current and planned validators?

The leader decides the order of commands, every node validates.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

This isn't wrong "In standard Raft, you need to replicate a log entry to a majority of nodes in the cluster before committing it. For BFT consensus algorithms, including Tangaroa, the required quorum size is 2f + 1, where f is the number of failures you want to tolerate (including both crashed nodes and compromised nodes)." But the brief answer is Juno handles up to half the cluster being byzantine.

Is there a forking vulnerability?

Hardened Raft establishes linearity: no forking.

How are the incentives defined within a permissioned system for the participating nodes?

As Juno targets this, there are no incentives. This isn't wrong "Each replica in BFT Raft computes a cryptographic hash every time it appends a new entry to its log. The hash is computed over the previous hash and the newly appended log entry. A node can sign its last hash to prove that it has replicated the entirety of a log, and other servers can verify this quickly using the signature and the hash."

What process does the system follow when it receives data?

See Juno's readme performance section for an example. Original: The data comes from clients of the Raft cluster, who send requests to the leader. The leader replicates these requests to the cluster, and responds to the client when a quorum is reached in the cluster on that request. What constitutes a "request" is system-dependent.

How is data currently stored?

In memory.

Governance, Risks and Control

How is governance / controls enforced?

Juno seeks to provide immutable evidence for dealing with malicious behavior in the courts after an event has occurred. For the domain that we're targeting, this will happen anyway. Orig: Like Raft, BFT Raft uses randomized timeouts to trigger leader elections. The leader of each term periodically sends heartbeat messages (empty AppendEntries RPCs) to maintain its authority. If a follower receives no communication from a leader over a randomly chosen period of time, the election timeout, then it becomes a candidate and initiates a new election. In addition to the spontaneous follower-triggered elections BFT Raft also allows client intervention: when a client observes no progress with a leader for a period of time called the progress timeout, it broadcasts UpdateLeader RPCs to all nodes, telling them to ignore future heartbeats from what the client believes to be the current leader in the current term These followers will ignore heartbeat messages in the current term and time out as though the current leader had failed, starting a new election.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

There are many thousands of pages of regulations stipulating how this is to take place. Running a different system will never change this. We seek only to have known participants and immutable evidence.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

No

How does the consensus mechanism allow access?

K-way admin. key signing.

How does the consensus mechanism restrict access, concerning malicious activities?

All messages must be signed and the pubkey + node id known to all nodes before any message is accepted.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., Back Office outsourcing)

Yes, no, yes, no

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

See Juno's readme performance section for an example. We run at 5k/s now.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

See Juno's readme performance section for an example. We run at 5k/s now.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Unlimited.

How do you measure scalability?

See Juno readme's performance section for an example. We run at 5k/s now.

Is there a limitations on the number of fields within a transaction?

No

Is the speed of the system impacted if the system is made more scalable?

No

Does synchronization have any impact on scalability?

Minor, mostly due to required bandwidth.

Privacy

How does the system ensure privacy?

Juno doesn't care about the body of the log entry. As such, every message can be encrypted if need be in whatever method the user prefers.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes

Do all nodes have visibility into all other transactions?

Juno doesn't care about the body of the log entry. As such, every message can be encrypted if need be in whatever method the user prefers.

How is privacy defined and ensured between applications?

Juno doesn't care about the body of the log entry. As such, every message can be encrypted if need be in whatever method the user prefers.

How does the data encryption model work?

Every message can be encrypted if need be in whatever method the user prefers.

Cryptography/Strength of Algorithm

How are the keys generated?

Ed25519 is used to create key pairs prior to startup.

Does the consensus mechanism require a leader?

Consensus has a leader.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

Flexible. Adding new supported languages is trivial. Juno has previously run Hopper, Transmatic, Scheme. Switching between them takes ~30min.

Tokenization (if used)

Does the consensus mechanism utilize transaction signing?

Blockchain solves this problem by Public-Private key pairs for signatures on and verification of transactions. Tangaroa's protocol specifies using a similar system, but at the consensus level as well. This provides a means for one node to validate that a message came from another node (so long as keys haven't been compromised).

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Private networks in enterprise environments, either intra or inter firm.

What is the implementation cost?

Depends on the domain.

What is the time required to implement?

Depends on the domain, but pretty fast at this point.

11. MultiChain

Source: Interview / Questionnaire

Contact name: Gideon Greenspan / Maya Zehavi (gideon@coinsciences.com / maya@coinsciences.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

"Distributed consensus between a configurable group of validators, in which the validators confirm blocks in a round-robin pattern, with some (configurable) leniency in that pattern to allow for non-functioning nodes. See the "mining diversity" explanation in the MultiChain white paper.

How many nodes are need to validate a transaction? (percentage vs. number)

There are two aspects to validation. First, that the transaction is a legitimate one in and of itself. This is validated by every single node independently, and there is no room for opinion. Second, that the transaction is not performing a double spend against another transaction. This is the role of the Blockchain, and is validated by whichever individual node generates the next block. Every block as a whole is still evaluated independently by every node, and again there is no room for opinion.

Do all nodes need to be online for system to function?

No, not at all. Any number of regular nodes can go down, and the system functions fine. There will only be a problem is too many mining nodes have gone down, in which case the distributed consensus scheme will freeze up. In this case new transactions can still be processed, but not achieve final confirmation in the Blockchain. (This is a key advantage of bitcoin-style Blockchain over Ethereum-style Blockchain - that unconfirmed transactions can be meaningfully processed and chained together.)

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

The only way to join a MultiChain Blockchain is if one or the admins node granted that node permission. But these permissions are dynamic and nodes can be added or removed at any time.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Each participant owns its own node, completely.

What are the different stages involved within the consensus mechanism?

Nodes which are able to create the next block (because they have mining permission and because they have not mining a previous block too recently) generate and broadcast a block at a random time interval, whose average is configurable in the Blockchain parameters (default: 15 seconds). Other nodes then build on this block in exactly the same way as the bitcoin network. If two mining nodes happen to generate a block at the same time, then a fork occurs in the consensus and this resolves itself when the next block is generated.

When is a transaction considered "safe" or "live"?

Immediately after a transaction is received, it is considered live, and conflicting transactions are not accepted by a node. This means that unless a node is deliberately broadcasting double spend transactions, the network effectively reaches consensus in a second or less. In a trusted environment, double spends can lead to people being sued, so the Blockchain's purpose is to act as a final confirmation that there was no double spend (and to help any nodes who missed transactions due to downtime). For day-to-day practical purposes, transactions can be trusted before they are even confirmed in the Blockchain. Again, this is a key advantage of using a bitcoin-style transaction model, in which transactions are directly chained together from outputs to inputs.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No

How much time does a node need to reach a decision?

This depends on the number of transactions in a block, but for most purposes a split second.

How much time is actually needed to build the consensus until a new block is added?

Because there is no back-and-forth voting, it's simply the network propagation time for a block. This depends more on the geographic separation of the nodes than anything else.

Does system contain synchronous node decision making functionality?

This is not a feature of MultiChain but can be easily added on top by an external process which queries each node for its last X block hashes, and therefore can ascertain that consensus has been reached.

What is the number of current and planned validators?

This is completely configurable in the product.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

Again, this is based on the configuration of the product, but the parameters can be set so that the system can tolerate 49 percent of compromised nodes, exactly like bitcoin.

Is there a forking vulnerability?

If the Blockchain is configured to allow some leniency for nonfunctioning mining nodes (via the mining diversity parameter) then a fork can occur but this will be resolved when the next block is generated. Note also my previous comments about the transaction model, in which only deliberate (i.e., contract violating) double spends can cause network disagreement. Because of this a temporary fork in the Blockchain itself does not affect the ability to continue transacting safely, unless somebody has done something illegal.

How are the incentives defined within a permissioned system for the participating nodes?

This is configurable in the Blockchain parameters, either purely external incentives, or else rewards can be configured for the validating nodes, and the system can even be made proof-of-work if the users wish.

What process does the system follow when it receives data?

"Transactions are first added to the memory pool, then when a block comes in the disk-based storage is updated?

How is data currently stored?

Raw binary files for the blocks, LevelDB for many other aspects of the system's state.

How does a party take ownership of an asset?

By that asset being sent to their public address, either in the initial issuance transactions, or by being transferred from another party. Only addresses with receive permissions are able to take ownership.

Governance, Risks and Control

How is governance / controls enforced?

MultiChain has an integrated permissions model in which one or more administrators control permissioning (per address) for seven types of operation - connect, send, receive, issue, mine, activate and admin. The latter four of these permissions are subject to configurable admin consensus, i.e., a certain proportion of admins have to agree on a permissions change before it becomes active.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

MultiChain is a stand-alone platform, like Oracle or MySQL, so we do not see or have any influence over the transactions taking place in a particular Blockchain. Therefore, this is entirely up to the users of a particular Blockchain to decide for themselves.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

There is no intrinsic negative penalty for attempted corruption but there can be a positive reward for those who successfully validate blocks.

How does the consensus mechanism allow access?

Through the per-address permissioning system.

How does the consensus mechanism restrict access, concerning malicious activities?

Each node throttles or disconnects peer nodes which are behaving antisocially.

What is the permission management process? What is the process for adding or deleting nodes?

To add a new node: first, any existing node on the network shares its "node address" (e.g., chain1@banknet.org:5678) with the new node. This gives the new node an initial entry point for connecting to the Blockchain. The new node makes an initial connection to this node address, obtains a minimal number of Blockchain parameters, is then disconnected, and self generates and displays its first private key and public address. This public address is sent to an administrator who grants it (at least) the right to connect to the Blockchain, and the transaction granting this right propagates to all nodes rapidly. The new node then reconnects to its initial entry point, self-identifying using its public address and signing a challenge message (in the peer-to-peer protocol) to prove it owns the corresponding private key. So this time it is granted permission to remain connected and begins to act like any other node, and it can start discovering and connecting to other nodes. To remove a new node: this connect permission is revoked by an administrator, and the transaction enacting this propagates to all nodes rapidly. Every node immediately disconnects any peers who were connected using a revoked address.

How does the protocol assess the trustworthiness of other participants?

Through permissioning, and by throttling or disconnecting antisocial peers.

Are there separate admin / administrator privileges? Who manages them?

Yes, this is one of the permissions in the system, and it is up to the participants in a Blockchain to decide who has these permissions and what level of consensus is required between the administrator for certain high-risk actions.

Are there restriction / privacy rights defined and enforced by node?

There are restrictions through the permissioning system, but these apply to write rather than read actions.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back-office outsourcing)

MultiChain's permissions management system consists of seven types of permissions - connect, send, receive, issue, mine, activate and admin. It is possible for a node to have read-only access, if it has connect permissions only. It is possible for a node to have read/write but no validation access, if it does not have mining permission. There is less value in a node having write but no read access, because it cannot build transactions if it does not know where it is receiving its assets from.

Please define which sorts of risk you are referring to?

All permission changes are recorded in the Blockchain itself, so a full audit trail is available of who allowed whom to do what, and when. It is up to the developers building on MultiChain to decide who manages this process and whether to store the KYC/AML data inside the chain or outside it.

How is counterparty risk / settlement risk addressed?

There is full and easy-to-use support in the MultiChain transaction model and API for DvP exchange transactions, between any number of parties and involving any number of assets. This means a single Blockchain transaction can represent any exchange of value, and if for some reason any part of the exchange fails, then every other part of the exchange will fail as well.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

This completely depends on the Blockchain parameters, the geographical spread of the nodes, and how many permitted miners there are. The initial propagation of transactions can be a split second.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., number of trades)

On regular (cloud) servers, around 200 TX/second. On top end, 500-1000 TX/second.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Irrelevant, the quantities involved have no influence on the Blockchain's performance.

How do you measure scalability?

The number of nodes is not relevant because this is a peer-to-peer protocol and there is no need for every peer to be connected to every other. The scalability is limited rather by the volume of transactions per second that can be processed by each node.

Is there a limitations on the number of fields within a transaction?

Up to 8MB of metadata can be added to any transaction, either as a single blob of data, in JSON format, or whatever the application level requires.

Is the speed of the system impacted if the system is made more scalable?

The number of nodes does not have a bearing on the experience of each node in itself. And as we scale the total support transaction throughput higher, this will necessarily mean that each transaction is processed faster.

Does synchronization have any impact on scalability?

No, because each node does not need to wait to hear back from every other node in order to process a block.

Security

How is transaction activity monitored?

This is up to the application developer, but we offer the free open source MultiChain Explorer which can make this easy.

Does the consensus mechanism utilize Digital Signatures?

Yes, ECDSA.

How does the consensus mechanism address an assumed industry standard?

N/A (and I think it may be too early to talk of industry standards in this case!).

Which risk/security issues are currently being worked on?

We have one outstanding security issue we are aware of, which is that a malicious node can cause a slowdown by transmitting an old block that generates a long fork. The solution is easy, by verifying the miner permissions of a fork-generating block before it is processed, based on the end state of the immediately preceding block.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

We've had some requests for a security audit to be done in future, and expect this will happen, but it is too early for now.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

The node should be installed either on premise or on a remote server (dedicated or cloud) that is under the end user's control, through a trusted provider. It is compatible with any modern 64-bit Linux.

Briefly describe the security testing performed till date (if any)

MultiChain is a fork of Bitcoin Core, which continues to safely steward billions of dollars in crypto currency value, and we follow any vulnerabilities that are reported in Bitcoin Core. The vast majority of our modifications do not touch security sensitive areas (e.g., API interface, wallet management, peer-to-peer transactions). Those changes that do touch these areas have been repeatedly reviewed internally, and will be externally reviewed at the appropriate time.

How are you planning to implement/integrate Digital wallets? (Including private key management)

MultiChain has an integrated digital wallet, with encrypted on-disk key storage, and it is also highly compatible with the ecosystem of digital wallets developed for the bitcoin network.

In case of a breach, what data is at risk?

The biggest risk is the private keys, and if desired, MultiChain can be used without storing any private keys within the node itself, via the "raw transactions" interface.

How does the system prevent signature fraud (e.g., stolen keys)?

On-disk encryption of private keys, or else don't store private keys in the node at all.

Does the consensus mechanism have full documentation in place?

Extensive documentation is available at http://www.multichain.com/developers/ and this also references the bitcoin documentation where appropriate.

How is the system expected to address general server issues?

Nodes perform a full sanity check when launched, and if a data discrepancy is detected, roll back 128 blocks and attempt to recover the database by replaying from that point. (The current alpha of MultiChain does not implement this completely, but we're working on it at this exact moment.). There is also the option of "re-indexing", i.e. fully rebuilding the database state from scratch, using the Blockchain stored on disk.

How does the consensus mechanism address the risk of "double spending"?

MultiChain uses the unspent transaction output (UTXO) model, in which assets are passed directly from one transaction's outputs to the next one’s inputs. Each output can only be spent once.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

This entirely depends on the number of transactions that have taken place since a node last connected. The initial synchronization is very fast (1000s of TX/sec) because signature checking is performed in parallel on multiple CPU cores.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

System time (for which NTP is a good sync mechanism) is sufficient and the consensus protocol allows for significant time skew.

What is the process for disaster recovery?

Blockchain by nature is highly disaster resistant since the transaction log is stored in full on multiple nodes. So multiple nodes can simply be run by each participant, on different servers in different locations. If necessary private keys can be shared between these nodes, or funds split across the addresses belonging to different nodes. Private keys can also be easily externally backed up.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

The biggest threat would be in a hybrid Blockchain where certain key actions (including mining and administration) were permissioned, but the chain is open to the wider world for connecting, sending and receiving. (A Blockchain can be configured in this way using its parameters.) We assume that these external actors could be highly malicious. However we do not monitor fraud directly because we do not see the activities that take place on a particular MultiChain Blockchain.

Privacy

How does the system ensure privacy?

The Blockchain itself can be permissioned in terms of the right to connect, however there is not currently a notion of finer-grained privacy between nodes.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes, completely.

Do all nodes have visibility into all other transactions?

For now, yes.

How is privacy defined and ensured between applications?

MultiChain currently assumes that all full nodes can see all transactions.

How does the data encryption model work?

Any binary data can be stored in transaction metadata, so any encryption method can be used to store information in a Blockchain, which should not be visible to all of its participants.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

This is the choice of the system's users. Randomizing public addresses for every transaction is not compatible with address-based permissioning for send and receive actions. But a chain could be defined with closed connect permissions only, and anyone can send or receive, in which case each participant would be free to use as many self-generated addresses as they wished. It would still be private from the perspective of the outside world.

Cryptography/Strength of Algorithm

How are the keys generated?

"Chain will also collaborate with other open source Blockchain, cryptography and distributed systems projects to ensure interoperability and harmonization across industry efforts."

Asset Issuer creates unlimited number of cryptographically unique Asset IDs.

Rotating keys - every two to three weeks. May need 2 out of 3 keys to signing. Advised to keep each key in different datacenter. If one gets compromised then use other to generate backup keys and transfer over all assets to new key."

What does the key life cycle management look like?

"Blockchain depend on proper management and rotation of key material to secure digital assets. Chain Core integrates with industry-standard hardware security module (HSM) technology. All block and transaction signing takes place within hardened HSM firmware. Multi signature accounts using independent HSMs can further increase Blockchain security.

HSM firmware that secures all transactions and blocks

Multi signature accounts to eliminate single points of failure."

What is the HSM integration approach?

All block and transaction signing takes place within hardened HSM firmware. Chain parts and plans to partner with industry leaders for HSM for production environments.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

The Chain OS is focused on networks that can digitize the worlds existing assets (not a new currently like bitcoin), whether commonplace ones like gift cards of more obscure ones like syndicated loans and was developed by applying the technology to real objects in areas such as banking, payments, capital markets and insurance.

Briefly described the life cycle management process for the tokens

NO Natural bitcoin. No ripple. Partners working with chain prefer that.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

"Asset Issuance – Digitize existing assets for transacting on a Blockchain network.

Simple Payment – Transfer assets from one account to another.

Bilateral Trade – Swap one asset for another with no counterparty risk.

Order book – Name your sale price and let a buyer find you.

Collateralized Loan – Lend assets, guaranteed by locked collateral

Auction – Set a minimum price and sell your assets to the highest bidder"

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Nasdaq, in partnership with Chain, was the first to launch a Blockchain product, Linq, a platform for trading shares in pre-IPO companies. In late December, the first private securities transaction was recorded on a Blockchain in Linq and today it remains the only live privately run Blockchain network.

Who are you currently working with? (e.g., Venture Capitalists, Banks, Credit Card companies, etc.)

Visa, Nasdaq, FirstData, Citi, Capital One, MUFG, State Street, Fidelity, Orange, Fiserv.

12. OpenChain

Source: Interview / Questionnaire

Contact name: Flavien Charlon (flavien.charlon@coinprism.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

OpenChain is private chain software, and every organization would deploy their own chain and become administrator for that chain. OpenChain relies on the underlying database for consensus. Databases such as Cassandra offer a very efficient and scalable consensus model, and OpenChain leverages this. OpenChain offers a pluggable architecture that allows the organization deploying it to select which storage engine to use, and therefore which consensus to use. On top of that, OpenChain uses a publisher/subscriber model across trust boundaries. There is generally one validating cluster, and many subscribers that act as non–validating full nodes.

How many nodes are need to validate a transaction? (percentage vs. number)

Because transaction validation is under the control of the organization deploying the OpenChain instance, generally one node is sufficient to validate the transaction (but that has to be a node controlled by the administrator).

Do all nodes need to be online for system to function?

No, only the validating cluster (controlled by the administrator of the instance) has to be online for the system to be writable. Only one node on the given chain has to be online for the system to be readable.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

The validating organization has to be known ahead of time.

If applicable, what is the voting process after the "propose" stage?

The voting mechanism is outsourced to the underlying database (SQL Lite, SQL Server currently supported, MongoDB, and Cassandra on the roadmap).

When is a transaction considered "safe" or "live"?

As soon as the validator receive the transaction, it will decide whether the transaction is live (few milliseconds).

How much time does a node need to reach a decision?

Generally around 10 milliseconds.

How much time is actually needed to build the consensus until a new block is added?

Since the validating cluster for a given chain is the one and only source of truth for that chain, it is instant.

Does system contain synchronous node decision making functionality?

Yes, transaction validation is synchronous.

What is the number of current and planned validators?

At least one per chain.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

In a Cassandra type deployment, the system will stop accepting writes if quorum (n/2 + 1) cannot be reached.

Is there a forking vulnerability?

The organization that deployed a given chain can fork it if needed, external participants can't.

How are the incentives defined within a permissioned system for the participating nodes?

The organization that deployed the chain is incentivized to keep the chain running and validating transaction because their own service generally relies on it.

What process does the system follow when it receives data?

New transactions are relayed to the validating node, then the validating node will validate it and broadcast to subscriber participants assuming it passed validation.

How is data currently stored?

Multiple storage engines are available (SQL Lite, SQL Server currently supported, MongoDB, and Cassandra on the roadmap). A chain can have nodes with mixed storage engines. Each node has its own copy of the chain.

How does a party take ownership of an asset?

Generally, the asset is sent to their address.

Governance, Risks and Control

How is governance / controls enforced?

Every chain is fully governed by the organization that deployed that given chain.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

The administrator of the chain can act as a third party trusted by all participants and can therefore mediate any dispute between different participants on the chain. If the administrator of the chain is the source of the malicious actions, legal action must be taken against them, and the chain itself can be used as evidence in court as it will contain the signatures from the administrator for these illegitimate actions.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

The consensus is fully under control of the administrator. Penalty against the administrator involves taking legal actions against them. The chain itself can be used to incriminate them, and prove a breach of contract for instance.

How does the consensus mechanism restrict access, concerning malicious activities?

OpenChain has an elaborate access control permissioning system. The administrator is the root user, and can delegate permissions down to other users.

What is the permission management process? What is the process for adding or deleting nodes?

Anybody can join the network as a non–validating full node as long as they have access to another node on the network.

How does the protocol assess the trustworthiness of other participants?

By default, no participant is trusted, except the administrator. The administrator can delegate permissions to normal users.

Are there separate admin / administrator privileges? Who manages them?

The administrator is defined during the deployment of an instance of OpenChain. That can be changed later on assuming physical access to the validating node.

Are there restriction / privacy rights defined and enforced by node?

This can be implemented on top of OpenChain.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back–office outsourcing)

The permission system on OpenChain is very flexible and granular and any combination is possible.

What are the measures in place to reduce risk?

The administrator of the chain has full control on the chain, and can manage risks as needed.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

The administrator would generally have an onboarding process for AML/KYC. The data can either be stored on-chain but more likely off-chain for privacy.

How is counterparty risk / settlement risk addressed?

The asset issuer is a counterparty in every transaction where that given asset is involved.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Transactions validation (and consensus) takes generally about 10 milliseconds.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

A standard laptop used as a validating node can achieve several thousands of transactions per second. A Cassandra-type cluster should in theory be able to achieve hundreds of thousands transactions per second.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

There is no limit to the monetary amount of transactions.

How do you measure scalability?

Transactions per second (several thousand).

Is there a limitations on the number of fields within a transaction?

No limitation, the data model is extremely extensible and can be modeled for any use case.

Is the speed of the system impacted if the system is made more scalable?

No.

Does synchronization have any impact on scalability?

No as OpenChain uses a publisher/subscriber model, and subscribers don't have to be perfectly in sync for the system to function.

Security

How is transaction activity monitored?

Every node monitors all the transactions.

Does the consensus mechanism utilize Digital Signatures?

Yes.

How does the consensus mechanism address an assumed industry standard?

OpenChain is self-sufficient, and can and will continue functioning regardless of an industry standard.

Which risk/security issues are currently being worked on?

Better tooling support for mixed transparency chains (i.e., chains only available on protected VPNs)

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

No

How are you planning to implement/integrate Digital wallets? (Including private key management)

We have a developer wallet, but wallet infrastructure should be implemented as part of client-side applications for OpenChain.

In case of a breach, what data is at risk?

The data that could be leaked is the data that is stored on the chain; however, all the participants already have full access to that data. If private information should be stored on the chain, it should be encrypted by participants.

How does the system prevent signature fraud (e.g., stolen keys)?

Keys can be reset by the administrator of the chain if needed.

Does the consensus mechanism have full documentation in place?

Yes (https://docs.openchain.org)

How does the consensus mechanism address the risk of "double spending"?

The state of the chain prevents double spending.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

The system doesn't make assumptions on synchronicity.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

The clock of the validator is the authoritative time.

What is the process for disaster recovery?

Provided by the underlying storage engine. For instance, SQL Server has decades of production use and well defined processes for disaster recovery.

Privacy

How does the system ensure privacy?

Addresses can be pseudonymous.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes all messages must be signed.

Do all nodes have visibility into all other transactions?

Yes

How does the data encryption model work?

End to end encryption can be used.

If consensus happens in a permissioned network, are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

Public keys can either be regenerated on every transaction, or can be reused if needed. This depends on the business requirements of the organization deploying the chain.

Are participants' identities hidden from one another? (e.g., Blackpool)

Yes, they can be. Again this depends on how the administrator decides to configure their OpenChain instance.

Cryptography/Strength of Algorithm

How are the keys generated?

Currently, OpenChain only supports ECDSA. The range of valid private keys is governed by the secp256k1 ECDSA standard. We might support RSA cryptography in the future.

What does the key life cycle management look like?

Private keys are used to sign transactions and must be kept securely. Private keys should be rolled over regularly.

What is the library approach?

Any cryptographic library supporting ECDSA will work with OpenChain (that includes any library made for bitcoin-type cryptography as OpenChain uses the same elliptic curve and key size).

What is the HSM integration approach?

Any HSM supporting ECDSA will work with OpenChain (that includes any HSM that works with bitcoin/Ethereum, since OpenChain uses the same cryptography).

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

Asset types can be dynamically created, and are identified by a path. It is possible to create sub assets, for example a T-Bonds could be a generic asset types, with subtypes for each maturity date.

Which security mechanisms are assigned to the tokens?

A participant must have issuance rights to issue a given asset type. Participants must also have a specific right to either receive it or send it. The ledger can be setup in many different ways, with different level of users, depending on the business requirements.

Briefly described the life cycle management process for the tokens

Tokens are generally first issued by an issuer (which may or may not be the administrator of the chain), exchanged by different participants, and finally redeemed back at the issuer.

Does the consensus mechanism utilize transaction signing?

Transactions are all signed using EDCSA signatures.

Implementation Approach

What is the implementation cost?

Depends largely on the scale of the project.

What is the time required to implement?

Some companies have implemented OpenChain in as little as two days.

Who are you currently working with? (e.g., venture capitalists, banks, credit card companies, etc.)

Banks, accounting companies, startups in energy, supply chain, wealth management, document archiving.

13. PoET by Intel

Source: KPMG Research

Contact name: See Contact Us below

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

Sawtooth Lake is architected to be a modular solution. The consensus model is explicitly pluggable. It is distributed with a voting consensus as well as our Proof of Elapsed Time. Sawtooth Lake abstracts the core concepts of consensus, isolates consensus from transaction semantics, and provides two consensus protocols with different performance trade-offs. The first, called PoET for "Proof of Elapsed Time", is a lottery protocol that builds on trusted execution environments (TEEs) provided by Intel's SGX to address the needs of large populations of participants. The second, Quorum Voting, is an adaptation of the Ripple and Stellar consensus protocols and serves to address the needs of applications that require immediate transaction finality. Information on POET are here:http://intelledger.GitHub.io/introduction.html#proof-of-elapsed-time-poet.

How many nodes are need to validate a transaction? (percentage vs. number)

PoET consensus is a 51 percentage model.

The architecture is designed to support pluggable consensus so other models can be used in lieu of PoET (for example we include a voting consensus as well.)

Do all nodes need to be online for system to function?

No. It's byzantine fault tolerant.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

No. That depends on which consensus you plug in, but that is not a requirement of PoET.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Participants, though again you can choose to plug in a permissioned consensus module with that requirement.

What are the different stages involved within the consensus mechanism?

It randomly distributes leadership election across the entire population of validators with distribution that is similar to what is provided by other lottery algorithms. The probability of election is proportional to the resources contributed (in this case, resources are general purpose processors with a trusted execution environment). An attestation of execution provides information for verifying that the certificate was created within the TEE (and that the validator waited the allotted time). Further, the low cost of participation increases the likelihood that the population of validators will be large, increasing the robustness of the consensus algorithm.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

For the purpose of achieving distributed consensus efficiently, a good lottery function has several characteristics:

Fairness: The function should distribute leader election across the broadest possible population of participants.

Investment: The cost of controlling the leader election process should be proportional to the value gained from it.

Verification: It should be relatively simple for all participants to verify that the leader was legitimately selected.

If applicable, what is the voting process after the "propose" stage?

Basically, every validator requests a wait time from a trusted function. The validator with the shortest wait time for a particular transaction block is elected the leader. One function, say "CreateTimer" creates a timer for a transaction block that is guaranteed to have been created by the TEE. Another function, say "CheckTimer" verifies that the timer was created by the TEE and, if it has expired, creates an attestation that can be used to verify that validator did, in fact, wait the allotted time before claiming the leadership role.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No

How much time is actually needed to build the consensus until a new block is added?

Configurable see PoET description.

What is the number of current and planned validators?

PoET is designed to support a large number of nodes, well beyond what PBFT mechanisms are designed to support. PBFT is designed to support a dozen to 10s of validators. PoET is designed to support 100x that.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

0.51

Is there a forking vulnerability?

Two different deployments. You can deploy as a public system or deploy as a restricted system with permissioned hosts.

Governance, Risks, and Controls

How does the protocol assess the trustworthiness of other participants?

Facilitated by the consensus protocol.

Are there separate admin. / administrator privileges? Who manages them?

Sawtooth has an optional administration key for certain messages. For example a test network can be shutdown with an administration key. This is an optional feature.

Can a node or a user have only "Read" or only "Write–access? Is specific node access required if only performing one functionality? (e.g., Back Office outsourcing)

Sawtooth supports different actor types such as validator, transactor and observer.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

Questions on stuff like KYC/AML, fraud, privacy all relate to a specific deployment of the system. The system itself is designed to provide underlying functionality for a range of deployments (financial, IOT, etc.)

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Interblock time is configurable from seconds to minutes or whatever is desirable for the usage area.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

Volume depends on transaction complexity.

How do you measure scalability?

Number of validator nodes and transactions per second under some normalized transaction complexity are relevant metrics.

Security

Does the consensus mechanism utilize Digital Signatures?

ECDSA signatures are used extensively in the system.

What are the infrastructure hosting options? (e.g., cloud, hosted in a datacenter, etc.)

Designed for public and permissioned systems.

How are you planning to implement/integrate Digital wallets? (Including private key management)

Intel has hardware security mechanisms that can be leveraged for client security.

In case of a breach, what data is at risk?

Validity controlled by consensus. A compromised host can't inject bad data because other hosts will reject it.

How does the system prevent signature fraud (e.g., stolen keys)?

Some keys protected in hardware. Customer (end user in public, or enterprise user/system in private) application will likely have keys specific to adjacent applications that need to be properly controlled.

Does the consensus mechanism have full documentation in place?

Documentation at GitHub link above.

How does the consensus mechanism address the risk of "double spending"?

Doubles pending controlled by consensus. The validators validate the transactions so they can't create invalid operations like spending the same value twice, and that's enforced by consensus that the validators all agree on some transaction ordering...

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

For synchrony the network can be tuned to different Interblock times. In the tutorial, consensus set to a matter of seconds.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Mechanism isn't time dependent.

Privacy

Does the system require verifiable authenticity of the messages delivered between the nodes?

Messages are all signed and must be verified before use.

Do all nodes have visibility into all other transactions?

Transaction transparency is default. Transaction privacy isn't disclosed yet.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

System has pluggable transaction model. The open source release includes an example "marketplace" transaction family. In that example, transaction family participants are provisioned with a 1:1 participant: key. That could be adapted to generate a different key or participant for every transaction. Probably in a private deployment, that would not be desirable.

Cryptography/Strength of Algorithm

How are the keys generated?

The PoET keys are generated in the SGX Enclave

Participant keys are generated using a keygen method in the python. The latter is a convenience method so can be replaced with other key provisioning systems in an enterprise.

Does the consensus mechanism require a leader?

Yes

Implementation Approach

What is the implementation cost?

Cost/time depends on complexity of deployment, e.g. for adding domain specific logic. However the existing open source system has been tested with complex deployments requiring only nominal customization time.

What is the time required to implement?

Cost/time depends on complexity of deployment, e.g. for adding domain specific logic. However the existing open source system has been tested with complex deployments requiring only nominal customization time.

14. RAFT

Source: KPMG Research

Contact name: See Contact Us below

Questionnaire responses

Consensus Methodology

How many nodes are need to validate a transaction? (percentage vs. number)

Logs need to be propagated to an absolute majority of the network (including the leader itself), i.e., strictly more than a half, i.e., >50 percentage. 50 percentage itself is insufficient.

Do all nodes need to be online for system to function?

No a majority of the nodes. Typically 5.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

There are extensions to the Raft algorithm that allow for adding or removing nodes from a cluster. This is described in the PhD thesis on Raft.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Raft is a centralized transaction processor contrasting distributed.

What are the different stages involved within the consensus mechanism?

Each server stores a log containing commands. Consensus algorithm ensures that all logs contain same commands in the same order. State machines always execute commands in the same log order. They will remain consistent as long as command executions have deterministic results. Client sends a command to one of the servers. Server adds the command to its log. Server forwards new log entry to the other servers. Once a consensus has been reached, each server state machine process the command and send its replies to the client.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

Followers only respond to requests from other servers. If a follower receives no communication, it becomes a candidate and initiates an election. A candidate that receives votes from a majority of the full cluster becomes the new leader. Leaders

typically operate until they fail. Heartbeats sent out to make sure leader is still there, if nothing received a new election takes place.

If applicable, what is the voting process after the "propose" stage?

Leader election: Raft uses randomized timers to elect leaders. This adds only a small amount of mechanism to the heartbeats already required for any consensus algorithm, while resolving conflicts simply and rapidly.

When is a transaction considered "safe" or "live"?

When a majority has been achieved.

How much time does a node need to reach a decision?

You'll need to be clearer on what decision you mean. In normal operation, the node will apply items to its FSM and read from its FSM. You could (with appropriate timeouts) have each of those take one second. This would, however, involve setting an election timeout an order of magnitude higher.

How much time is actually needed to build the consensus until a new block is added?

Blocks are not a raft concept. It takes about 10–80 microseconds to form a consensus on optimized networks, but you can add multiple records processed concurrently.

Does system contain synchronous node decision–making functionality?

Yes

What is the number of current and planned validators?

1-Strong leader does most of the work issues all log updates.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

If half or more of the nodes are down, the cluster will no longer be able to progress new log writes. Reads to the FSM may work if stale reads are permitted. If you really meant "compromised" in the sense of "hacked", A single node being compromised could be sufficient to DoS the system without further protection.

Is there a forking vulnerability?

No

What process does the system follow when it receives data?

Replicated state machine architecture. The data comes from clients of the Raft cluster, who send requests to the leader. The leader replicates these requests to the cluster, and responds to the client when a quorum is reached in the cluster on that request. What constitutes a "request" is system-dependent.

How is data currently stored?

How data is stored is system-dependent. It's important for state to persist to disk so that nodes can recover and remember information that they have committed to (which nodes they voted for, what log entries they have committed, etc.). The protocol can't work without this.

How does a party take ownership of an asset?

I think this depends on asset (digital or physical).

Governance, Risks and Control

How is governance / controls enforced?

"This is outside the scope of Raft.

Like Raft, BFT Raft uses randomized timeouts to trigger leader elections. The leader of each term periodically sends heartbeat messages (empty AppendEntries RPCs) to maintain its authority. If a follower receives no communication from a leader over a randomly chosen period of time, the election timeout, then it becomes a candidate and initiates a new election."

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Raft doesn't cater for Byzantine failure – it assumes perimeter based security.

How does the consensus mechanism allow access?

Complete Raft supports on dynamic reconfiguration. In practice we support some downtime to reconfigure the number of nodes.

How does the consensus mechanism restrict access, concerning malicious activities?

Raft does not specify a transport layer, and authentication / authorization is a matter for the transport layer. Hashicorp/raft from memory uses TLS as can Etcd/Raft. However, this is outside the spec of raft. When raft gets the messages, it is assumed they have been authenticated / authorized.

What is the permission management process? What is the process for adding or deleting nodes?

"Epochs of arbitrary length. Starts with a leader being elected ends when no leader can be selected (split vote) or leader becomes unavailable (offline).

Raft's mechanism for changing the set of servers in the cluster uses a new joint consensus approach where the majorities of two different configurations overlap during transitions. This allows the cluster to continue operating normally during configuration changes."

Are there separate admin. / administrator privileges? Who manages them?

Built into Raft.

Are there restriction (/) privacy rights defined and enforced by node?

This is outside the scope of Raft – Raft is purely a consensus system.

Can a node or a user have only "Read" or only "Write access"? Is specific node access required if only performing one functionality? (e.g., back–office outsourcing)

Use RPC (Remote procedure Call) to communicate Leader initiates AppendEntry to replicate log entries and leaders provide a heartbeat (lets other know they are still online and updating log) remain as long as they send valid RPC's, after a period of time if no "heartbeat", election process starts.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

"Raft communication is unicast. Under normal operation the leader makes RPC calls to the followers, which means communication is O(n). As the cluster gets larger, there is more chance of an individual node failing in a given time interval; however, it takes more individual nodes to fail to break a quorum. When individual nodes fail and come back online (assuming no reelection) their logs need to be "caught up". The failure of the network fabric is also going to have some relationship with the size of nodes. There are some studies on raft scaling in both papers – particularly in as it affects election timeouts. In general, the number of nodes is not going to be the key determinant of transactions per second. The practical limit in most scenarios is likely "a few hundred" though doubtless the algorithm could be tuned. Many use cases with more "nodes" have a small subset participating in raft (and sharing the FSM) and a larger subset that are nodes but not raft nodes (e.g., consul). The number of transactions per second is going to depend more on network latency and time to apply to the FSM than number of nodes. Remember that whilst each follower processes every log entry, they can do so in parallel.

1000sper second

In a three node cluster ~ 80 micro seconds + 2 x record write time + latency long tail overhead."

Is the speed of the system impacted if the system is made more scalable?

Yes. More transactions = more processing / writes. Three, five or seven Nodes provides different reliability constraints and different read (/) write behavior. You use more nodes for greater availability so you get a slight write slow down as the quorum requirement goes from three-five-seven nodes.

Does synchronization have any impact on scalability?

Yes – you have to go through a strong leader and you get queuing behavior.

Security

How is transaction activity monitored?

We built a TPM (Transaction processing Monitor).

Does the consensus mechanism utilize Digital Signatures?

No

Which risk/security issues are currently being worked on?

Out of scope of raft implementation dependent.

In case of a breach, what data is at risk?

Implementation dependent.

How is the system expected to address general server issues?

No

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

It is the leader who synchronizes and sends out a replicated log.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Clock is managed by the central node. Raft's notion of a term does away with having to keep centralized clocks.

Under which conditions does a lock/unlock happen? (i.e., what is the proof safety?)

Raft uses a replicated finite state approach to transactions rather than locking. We model our transactions on this basis.

What is the process for disaster recovery?

Read the raft papers searching for “stable store” or similar. The whole point is that the raft log items, once accepted, are guaranteed to be on stable storage. So, in the event of a total outage, the cluster should just come back alive. Similarly you only need one copy of the FSM to restore the whole lot. The details of the process will depend both on the implementation and on operational procedures by the end user (e.g., how backups are taken). Note the fact that Raft is designed to operate between data centers makes this easier. One minor challenge is that the state of the peers is itself in the FSM, so if you lose a lot of peers, adding back peers to form a quorum needs thought. From memory this is covered in chapter 4.

Privacy

Does the system require verifiable authenticity of the messages delivered between the nodes?

Note that TCP itself will prevent most corruption and ensure retransmission, and most users use TCP (with or without encryption on top). Note that Raft does not have to run over TCP (I submitted a UDP version for instance, and running over say ZeroMQ would be quite feasible).

How does the data encryption model work?

While encryption is out of Raft’s scope – yes you can encrypt.

Are participants' identities hidden from one another? (e.g., Blackpool)

No

Cryptography/Strength of Algorithm

Does the consensus mechanism require a leader?

Yes

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

"Not 100% on what you mean by ’strictness‘. Raft consensus is achieved by majority for a transactional write and is designed to provide hard guarantees.

Implementation dependent. Many implementations are inherently flexible (see Etcd/Raft / Hashicorp/Raft). I'm sure it would be possible to design something gross that is hard coded."

How is node behavior measured for errors?

The raft specs do not specify measurement. Most implementations measure some aspects (e.g., with monotonic counters), and often send them off to a stats server. Grep for "stats" in Hashicorp/Raft for instance. If you mean "How does the algorithm tell whether a node has errored?" it simply looks for either the absence of a valid reply or heartbeat within a timeout, or the presence of an invalid reply or heartbeat (where "valid" and "invalid" are determined by the spec).

Tokenization (if used)

Does the consensus mechanism utilize transaction signing?

No

Implementation Approach

What is the implementation cost?

Transport Dependent.

What is the time required to implement?

Transport Dependent.

15. Ripple

Source: Interview / Questionnaire

Contact name: Bob Way (bob@ripple.com)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

The Ripple consensus algorithm (RCA), is applied every few seconds by all nodes, in order to maintain the correctness and agreement of the network. Once consensus is reached, the current ledger is considered "closed" and becomes the last-closed ledger. Assuming that the consensus algorithm is successful, and that there is no fork in the network, the last-closed ledger maintained by all nodes in the network will be identical.

How many nodes are need to validate a transaction? (percentage vs. number)

A supermajority which is 80percent.

Do all nodes need to be online for system to function?

No

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

No

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Ripple is an open network. Anyone can run a node. Anyone can participate in the consensus process.

When is a transaction considered "safe" or "live"?

A transaction is "safe" after it has been validated by your rippled server. This process normally takes around five seconds.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Yes

How much time does a node need to reach a decision?

This process normally takes around five seconds.

How much time is actually needed to build the consensus until a new block is added?

Consensus is reached about every five seconds. Ripple uses a process called "continuous ledger close". That means once consensus has been reached, if any server has received new transactions, they can start the next ledger close process immediately. There is no need to wait for a pre–specified timeout.

Does system contain synchronous node decision making functionality?

Yes, all nodes reach consensus at the same time.

What is the number of current and planned validators?

This is flexible and open. Participants determine individually if they want to act as a validator.

Currently there are around 40 known validators on the network.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

0.2

Is there a forking vulnerability?

If there are no restrictions on the membership of the UNL, and the size of the UNL is not larger than 0.2 ∗ ntotal where ntotal is the number of nodes in the entire network, then a fork is possible.

How are the incentives defined within a permissioned system for the participating nodes?

There are no incentives for validation the ledger. Profitable entities relying on the ledger for business will choose to validate the ledger in order to guarantee its continued operation.

What process does the system follow when it receives data?

Rippled servers only receive "transactions". They do not provide a generalized messaging capability.

When a transaction is receive, rippled:

1) Checks the signature of the transaction. (Authentication)

2) Verifies the transaction by attempting to apply the transaction to its local ledger. (Authorization)

3) If the transaction is authentic and correct, the rippled server broadcasts it to its peers for consensus. (Validation)

How is data currently stored?

All Ripple data is stored in an internal database that is managed by and accessed through the rippled server.

How does a party take ownership of an asset?

The Ripple ledger manages account balances between parties. If a party wants to settle their account (return the balance to zero) then those arrangements must be agreed upon prior to establishing the account relationship.

Governance, Risks and Control

What is the permission management process? What is the process for adding or deleting nodes?

Anyone can add a new node to the RCL network. All you need to do is install the rippled package and run it.

Anyone can configure their rippled server to act as a validating node. No permission is required to do so. Validating nodes broadcast "proposals" as part of the consensus process.

It is up to the other nodes whether or not they wish to consider your proposals as part of their local decision–making process. Rippled operators individually configure the nodes whose validation proposals they will consider. This configuration is referred to as the "Unique Node List" (UNL).

An operator's decision which validation nodes to configure on the UNL is determined based on three key factors:

1) Do I know who is operating the validator? (Validators cannot be anonymous)

2) Is the validator reliable? (Is it operated 24/7 and upgraded consistently)

3) Does the validator stay synchronized? (Are the validators' proposals regularly adopted by the rest of the network?)

Ripple (the company) is building tools to help others monitor the above so they can make intelligent decisions.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

“Permissioned system” is a term generally used when talking about private Blockchain. KYC/AML (anti–money laundering) seems at a different level.

In general, the Ripple Consensus Ledger (RCL) is not a permission–based or private system. Each validator individually determines what other validators they wish to stay in consensus with. Overlapping relationship decisions assure the federated ecosystem stays in consensus as a whole.

How is counterparty risk (/) settlement risk addressed?

The Ripple ledger manages both the entities and the accounting relationships between those entities. This allows the system to support multi–party atomic payments. (Rippling payments).

The accounting relationships can be arranged to provide instant settlement. Or the relationships can be arranged to account for deferred net settlement. The choice is up to the parties who wish to do business.

As a rule, each entity must evaluate their direct counterparties before establishing an accounting arrangement. This is just a generalized case of “evaluate your bank before you open an account."

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

The ledger close interval looks like about 3.5 seconds per ledger at the moment. Depending on timing, a transaction will normally make it into the current or next ledger. Normally average it to “within about 5 seconds”.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., # of trades)

Unlimited.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Unlimited.

Is the speed of the system impacted if the system is made more scalable?

No

Security

How is transaction activity monitored?

We have a data team that keeps watch over the network for our own interests and needs. Anyone else can do the same.

You can watch transactions scroll by here. https://www.ripplecharts.com/#/transactions

Does the consensus mechanism utilize Digital Signatures?

Yes, all transactions must be signed to make it into a ledger. We use the same signature scheme as bitcoin, plus we have added an ED25519 signature mechanism.

Which risk/security issues are currently being worked on?

The system has been stable and operational for almost four years. We monitor for potential security issues and submit open source patches as when prudent.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

Working on some system and corporate certifications, but it is not consensus mechanism specific.

Briefly describe the security testing performed till date (if any)

Ripple's open source code and running network has been reviews by NCC Group.

https://ripple.com/insights/ripple-eff-open-approach-to-security-will-define-next-chapter-of-finance/.

How are you planning to implement/integrate Digital wallets? (Including private key management)

Private Key management is done using our suite of banking products.

We don't currently support a consumer wallet.

In case of a breach, what data is at risk?

There is no data to breach. Anyone can connect a rippled server to the network and view all transaction data.

Bank customer information is never included in ledger transactions.

How does the system prevent signature fraud (e.g., stolen keys)?

We have certain operational practices that we recommend to mitigate the loss of a private key. (Issuing vs operational addresses) We are also currently adding a multisign capability to each Ripple address.

However, in general, users of the Ripple network are responsible for protecting their own private keys.

Does the consensus mechanism have full documentation in place?

Yes

https://ripple.com/build/

How is the system expected to address general server issues?

The Ripple consensus ledger is a distributed peer to peer network. Each peer is responsible for the operation of their own rippled server.

Open source patches are made as GitHub pull requests. New release builds are made available regularly. Each peer is responsible for updating their own servers.

How does the consensus mechanism address the risk of "double spending"?

"Double spending" is a bitcoin–specific concept. In the banking world it would be called "over-drafting your account".

The Ripple ledger allows each participating entity to set specific limits and rules for each of their account relationships. The entire Ripple system is dedicated to validating these rules prior to applying any transaction.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Nodes sync the most recent ledger first, then they work backwards to download as much history as they wish. There is no requirement to download the entire "Blockchain" as there is with bitcoin.

It generally only takes a few minutes for a newly spun up rippled server to reach synchronization with the current ledger and to begin processing transactions.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Each node maintains their own internal clock. The RCL as a whole reaches consensus on the time as it reaches consensus on each ledger.

Under which conditions does a lock/un–lock happen? (i.e., what is the proof safety?)

I don't know what a lock/un-lock is. We don't have this concept in Ripple.

What is the process for disaster recovery?

The Ripple RCL is a peer to peer distributed system. Each operator is responsible for their own disaster recovery process.

Ripple (the company) has our own disaster recovery process documented internally for the servers we operate.

What is the threat model being tested? What has been defined as "normal"? How do you monitor fraud?

Like with bitcoin, Ripple is an open distributed system that anyone is free to attack as they see fit. We consider this "normal".

Ripple (the company) has data and compliance teams that monitor the network. However, this is for our Ripple (the company's) own interests. Anyone else is welcome monitor the network for fraud or anything else they find interesting.

Privacy

How does the system ensure privacy?

At the ledger (Blockchain) level all transactions are publicly visible. The Ripple payment solution provided to banks handles customer privacy at another communication level, without requiring personal information to move across the Blockchain.

Does the system require verifiable authenticity of the messages delivered between the nodes?

All transactions are authenticated by digital signature.

Do all nodes have visibility into all other transactions?

Yes

How is privacy defined and ensured between applications?

See below.

How does the data encryption model work?

Communication between nodes is encrypted using SSL. However, this is only to assure data is not tampered with in transit.

No encryption is provided with the intention of data privacy. All transactions applied to the ledger are public.

Ripple transaction provide a "memos" field for user determined data. If necessary, this field can be pre-encrypted by the user before a transaction is sent.

Cryptography/Strength of Algorithm

How are the keys generated?

Each entity on Ripple is represented by a Ripple address. This address is identical in concept to a bitcoin address. It is simply an encoding of a public key. Of course being a public key, it must also have a corresponding private key.

Cryptographic keys must be generated in private by the owner. Doing so doesn't require a ledger, nor does it touch the ledger during generation. Key generation can be done by calling an API on your own local rippled server. Or it can be done by running the key generation code in our Ripple JavaScript library. There are other known implementations for the key generator as well. Basically the process starts with a large random number (the seed) that is then algorithmically transformed into a point on the elliptic curve.

What does the key life cycle management look like?

A Ripple address only becomes known to the ledger when it is funded with XRP. At this point it becomes an "entity" entry in the Ripple ledger. Each entity entry can also manages some additional meta-data. This includes a secondary, rotatable, signing key.

To rotate a signing key, the user first generates a new key offline using one of the above methods. Then the pass the new key to the Ripple ledger in a transaction signed with the old key. The rippled server takes care of the rest.

Private keys must always be managed by their owners off of the Ripple ledger. The ledger never touches private keys.

What is the library approach?

Rippled is accessed via RPC or Web socket APIs. We also provide a JavaScript library to assist those writing client software which calls rippled.

What is the HSM integration approach?

Ripple doesn't provide a hardware security module, but any HSM that can manage bitcoin keys can also be used to manage Ripple keys.

Does the consensus mechanism require a leader?

No

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

Hardcoded

How is node behavior measured for errors?

Nodes are monitored to see how successfully they stay in sync with the network over time.

They are also monitored to see if their validation proposals are adopted by the rest of the network.

These two parameters provide a metric to help others decide if a node is worth reaching consensus with.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Payments, correspondent banking.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

Information is available on the http://ripple.com website.

Who are you currently working with? (e.g., Venture Capitalists, Banks, Credit Card companies, etc.)

Yes

16. Steem

Source: Interview / Questionnaire

Contact name: Ryan R. Fox (ryan@ryanrfox.com.)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

A hybrid DPOS that adds a Proof of Work queue to fill one of the witness nodes of each round. Page 22 @ https://steem.io/SteemWhitePaper.pdf

How many nodes are need to validate a transaction? (percentage vs. number)

21 total witness, fixed: 19 elected witness nodes plus the next 1 Proof of Work miner from the top of the queue plus one random unelected witness. This hybrid approach allows blocks to be produced at a predictable timeslot and is open to Proof of Work miners competing to solve enough separate Proof of Work puzzles to be added to the queue of future block producers. The difficulty of the puzzle is relative to the length of the queue, to ensure a constant production of blocks at a given timeslot. Further, anyone may be selected at random, weighted by their approval ranking.

Do all nodes need to be online for system to function?

Same as Graphene. See above.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Same as Graphene. See above.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Same as Graphene. See above.

What are the different stages involved within the consensus mechanism?

Same as Graphene. See above.

When is a transaction considered "safe" or "live"?

Same as Graphene. See above.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Same as Graphene. See above.

How much time is actually needed to build the consensus until a new block is added?

Same as Graphene. See above.

Does system contain synchronous node decision making functionality?

Same as Graphene. See above.

What is the number of current and planned validators?

21

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

Same as Graphene. See above.

Is there a forking vulnerability?

Same as Graphene. See above.

How are the incentives defined within a permissioned system for the participating nodes?

Same as Graphene. See above.

What process does the system follow when it receives data?

Same as Graphene. See above.

How is data currently stored?

Same as Graphene. See above.

How does a party take ownership of an asset?

STEEM is the token.

Governance, Risks and Control

How is governance/controls enforced?

Same as Graphene. See above.

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

Same as Graphene. See above.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Same as Graphene. See above.

How does the consensus mechanism allow access?

Same, but adds the Proof of Work queue for one slot as well.

How does the consensus mechanism restrict access, concerning malicious activities?

Same as Graphene. See above.

What is the permission management process? What is the process for adding or deleting nodes?

Same as Graphene. See above.

How does the protocol assess the trustworthiness of other participants?

Same as Graphene. See above.

Are there separate admin. (/) administrator privileges? Who manages them?

Same as Graphene. See above.

Are there restriction (/) privacy rights defined and enforced by node?

Same as Graphene. See above.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

Same as Graphene. See above.

How is counterparty risk (/) settlement risk addressed?

Same as Graphene. See above.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Same as Graphene. See above.

How do you measure scalability?

https://steemle.com/stats.php

Is the speed of the system impacted if the system is made more scalable?

Same as Graphene. See above.

Security

How are you planning to implement/integrate Digital wallets? (Including private key management)

Named Accounts contain a hierarchy of keys: Owner, Active, and Memo. Witness Accounts also have Signing key. The Owner key is at the top and modify the associated keys below. This is the most important key to maintain safely offline. Users can sign transactions using their Owner key. This is the key that will be used within the wallet software. Witnesses need only their Signing key to produce blocks.

In case of a breach, what data is at risk?

Same as Graphene. See above.

How does the consensus mechanism address the risk of "double spending"?

Same as Graphene. See above.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Same as Graphene. See above.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Same as Graphene. See above.

Under which conditions does a lock/un–lock happen? (i.e., what is the proof safety?)

Same as Graphene. See above.

What is the process for disaster recovery?

Same as Graphene. See above.

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

Same as Graphene. See above.

Privacy

How does the system ensure privacy?

Same as Graphene. See above.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Same as Graphene. See above.

Do all nodes have visibility into all other transactions?

Same as Graphene. See above.

How is privacy defined and ensured between applications?

Same as Graphene. See above.

How does the data encryption model work?

Same as Graphene. See above.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

Same as Graphene. See above.

Are participants' identities hidden from one another? (e.g., Blackpool)

Same as Graphene. See above.

Cryptography (/) Strength of Algorithm

How are the keys generated?

Same as Graphene. See above.

What does the key life cycle management look like?

See BitShares column. Substitute STEEM for BTS.

What is the library approach?

Same as Graphene. See above.

Does the consensus mechanism require a leader?

Larimer is the lead Dev for Steemit, Inc.

How is node behavior measured for errors?

Same as Graphene. See above.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

The token is STEEM.

Which security mechanisms are assigned to the tokens?

Same as Graphene. See above.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Same as Graphene. See above.

What is the implementation cost?

Closed license held by Steemit, Inc.

What is the time required to implement?

Active

Who are you currently working with? (e.g., Venture Capitalists, Banks, Credit Card companies, etc.)

Ask Steemit, Inc.

17. Stellar

Source: Interview / Questionnaire

Contact name: Jed McCaleb / Joyce Kim (jed@stellar.org / joyce@stellar.org)

Questionnaire responses

Consensus Methodology

What is the underlying methodology used by the consensus mechanism?

The Stellar Consensus Protocol (SCP) provides a way to reach consensus without relying on a closed system to accurately record financial transactions. SCP has a set of provable safety properties that optimize for safety over liveness—in the event of partition or misbehaving nodes, it halts progress of the network until consensus can be reached. SCP simultaneously enjoys four key properties: decentralized control, low latency, flexible trust, and asymptotic security. https://www.stellar.org/papers/stellar-consensus-protocol.pdf.

How many nodes are need to validate a transaction? (percentage vs. number)

It depends on the network topology but likely around 67% for a network with untrusted nodes or 51percent for trusted nodes.

Do all nodes need to be online for system to function?

Not all but at least the percentages discussed above.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

No

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Flexible.

What are the different stages involved within the consensus mechanism?

SCP depends on each node in the network having a list of other nodes in the network that it listens to. It doesn't have to trust these nodes, it just has to believe these nodes won't collude to cheat it. These other nodes form what is called the quorum set of the node. Each node can have its own quorum set. SCP shows that as long as there is some degree of overlap that the network will converge on the same value during consensus.

Phases of the protocol: Nomination, Balloting (Prepare, Confirm, Externalize).

Nomination: Because slots need only be partially ordered, some applications of SCP will have only

One plausible ballot per slot. For example, in certificate transparency, each CA may

have its own series of slots and sign exactly one certificate tree per slot. However,

other applications admit many plausible values per slot, in which case it is helpful to

narrow down the possible input values. Our strategy is to begin with a synchronous

nomination protocol that achieves consensus under certain timing assumptions, and

feed the output of the nomination protocol into an asynchronous ballot protocol whose

Safety does not depend on timing.

Balloting: Once nodes have a composite value, they engage in the ballot protocol, though nomination

May continue to update the composite value in parallel. Balloting involves three phases for a given value. First a node "prepares" the value. This means it essentially announces to the world that it will go with that value if a quorum slice of its also is willing to accept that value. If that happens, then in "confirms" the value. If a quorum slice does in fact also confirm the value then the node can externalize the value and it is now considered valid. There are more details of course in the paper.

When is a transaction considered "safe" or "live"?

It is considered valid after a quorum slice has all confirmed the transaction. At this point you externalize the transaction.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

All well-formed transactions are included in the set of transactions.

How much time is actually needed to build the consensus until a new block is added?

Varies depending on the latency between nodes in the network. On the open Internet, it should be < 2 sec.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

Maximum theoretically possible N=3f+1 where f is the number of faults you want to allow and N is the number of nodes in the network.

Is there a forking vulnerability?

No

How is data currently stored?

SQL DB and in flat files that can be stored wherever for easy archival of history.

Governance, Risks and Control

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

The network is really just a messaging standard so legal action can and would still be taken in the same way it is done today.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Node operators would simply stop listening to the SCP messages of the offending nodes.

What is the permission management process? What is the process for adding or deleting nodes?

It is an open network; anyone can join or leave. However, it is possible to construct private networks using the stellar software that need some process for joining.

Are there separate admin. (/) administrator privileges? Who manages them?

Not in the public network. But this could be added to a private one.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back–office outsourcing)

Nodes can be either validating or not. So yes nodes can be read only.

In case of permissioned systems, who manages the KYC/AML process and where is the data stored?

This is managed by the institutions connected to Stellar, but we have created a compliance protocol to allow them to easily communicate.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

Consensus is reached every five seconds. This leads to three to five seconds to validate a transaction depending on where in the ledger close cycle the transaction is introduced to the network.

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., number of trades)

Highly dependent on the hardware you are running the nodes on, but we have achieved 3000/second on modest hardware. There is still a lot of room for optimization and the upper bound is still much higher than this number.

How do you measure scalability?

The number of transactions per second depends only marginally on the number of accounts in the system. The system scales to 500 million accounts without much effect on maximum transactions/second.

Security

Does the consensus mechanism utilize Digital Signatures?

Yes all the transaction are signed and all the SCP messages are also signed by the validator.

How are you planning to implement/integrate Digital wallets? (Including private key management)

This is something that is being done by thirdrd parties. We are building the core protocol and the more user–facing features are left for integrators and people connecting to the platform.

In case of a breach, what data is at risk?

There are no breaches at the Stellar protocol level since everything is public. There can certainly be encrypted pieces built on top of Stellar but that will be implementation specific.

Does the consensus mechanism have full documentation in place?

Yes

How does the consensus mechanism address the risk of "double spending"?

The consensus mechanism is designed to prevent double spending.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

This varies depending on how long the node has been offline and the amount of data in the current ledger. From instant to a few hours.

Privacy

How does the system ensure privacy?

This is handled in a layer above the stellar protocol through systems like the lightning network.

Does the system require verifiable authenticity of the messages delivered between the nodes?

Yes

Do all nodes have visibility into all other transactions?

Yes, all transactions on the network are public but there can be encrypted metadata attached to transactions which would be private. And there are also protocols for keeping the bulk of the transactions off the public ledger so they can remain private.

Are participants' identities hidden from one another? (e.g., Blackpool)

There is no inherent connection between names and accounts.

Cryptography / Strength of Algorithm

How are the keys generated?

The keys are ed25519 keys so can be generated with standard libraries or by the stellar software.

What does the key life cycle management look like?

This isn't dictated by Stellar, but stellar is very flexible on how you want to handle this. You can have multiple signers on an account. You are able to change the keys on your account.

Does the consensus mechanism require a leader?

No

How is node behavior measured for errors?

You can easily detect if a node is misbehaving since all SCP messages are broadcast to everyone in the network you would be able to see conflicting or invalid messages and the signature would tell you who sent them.

Tokenization (if used)

How are the asset tokenized (if applicable)? Briefly describe the tokenization concept and terminology

The Stellar distributed network can be used to track, hold, and transfer any type of asset: dollars, euros, bitcoin, stocks, gold, and other tokens of value. Any asset on the network can be traded and exchanged with any other.

When you hold assets in Stellar, you're actually holding credit from a particular issuer. The issuer has agreed that it will trade you its credit on the Stellar network for the corresponding asset–e.g., fiat currency, precious metal–outside of Stellar. Let's say that Scott issues oranges as credit on the network. If you hold orange credits, you and Scott have an agreement based on trust, or a trust line: you both agree that when you give Scott an orange credit, he gives you an orange.

When you hold an asset, you must trust the issuer to properly redeem its credit. Since users of Stellar will not want to trust just any issuer, accounts must explicitly trust an issuing account before they're able to hold the issuer's credit. In the example above, you must explicitly trust Scott before you can hold orange credits.

To trust an issuing account, you create a trust line. Trust lines are entries that persist in the Stellar ledger. They track the limit for which your account trusts the issuing account and the amount of credit from the issuing account that your account currently holds.

Does the consensus mechanism utilize transaction signing?

Asset transfer and exchange works like other transactions in Stellar. They must be signed by the public key or keys that own the asset.

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Open payment standard. But the consensus mechanism could be used for lots of things.

What is the implementation cost?

~one to two weeks of developer time.

Is there a reviewed business case to compare the implementation costs (including cost of the solution) to the current as-is process?

(No response).

Who are you currently working with? (e.g., venture capitalists, banks, credit card companies, etc.)

Deloitte, Mifos, Oradian, Tempo, Parkway, several other banks.

18. Tangaroa

Source: KPMG Research

Contact name: See Contact Us below

Questionnaire responses

Consensus Methodology

How many nodes are need to validate a transaction? (percentage vs. number)

In standard Raft, you need to replicate a log entry to a majority of nodes in the cluster before committing it. For BFT consensus algorithms, including Tangaroa, the required quorum size is 2f + 1, where f is the number of failures you want to tolerate (including both crashed nodes and compromised nodes).

Do all nodes need to be online for system to function?

No. Implementation Dependent.

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

There are extensions to the Raft algorithm that allow for adding or removing nodes from a cluster. This is described in the PhD thesis on Raft.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Implementation dependent.

What are the different stages involved within the consensus mechanism?

In BFT Raft, each node is in one of the three states: leader, follower, or candidate. Similar to Raft, BFT Raft divides time into terms, which start with an election. The winner of the election serves as the leader for the rest of the term. Sometimes, an election will result in a split vote, and the term will end with no leader.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

The candidate continues in the candidate state until one of the three things happens: (a) it wins the election, (b) another node establishes itself as a leader, or (c) a period of time goes by with no winner (i.e., it experiences another election timeout). A candidate wins an election if it receives votes from a quorum of the nodes. The candidate then promotes itself to the leader state and sends heartbeat messages with the votes and the updated

term number to establish its authority and prevent new elections. The signed votes effectively prevents a byzantine node from arbitrarily promoting itself as the leader of a higher term. Followers that receive this heartbeat message will update their leader ID and term if the leader presented enough signed votes for the matching term.

If applicable, what is the voting process after the "propose" stage?

To begin an election, a follower increments its current term and sends RequestVote RPCs in parallel to each of the other nodes in the cluster asking for their vote. RequestVote RPCs themselves work similarly to Raft. The modifications come primarily in the recipient of a RequestVote RPC. When a node receives a RequestVote RPC with a valid signature, it grants a vote only if all five conditions are true: (a) the node has not handled a heartbeat from its current leader within its own timeout (b) the new term is between its current term + 1 and current term + H, (c) the request sender is an eligible candidate, (d) the node has not voted for another leader for the proposed term, and (e) the candidate shares a log prefix with the node that contains all committed entries. A node always rejects the request if it is still receiving heartbeat messages from the current leader, and it ignores the RequestVote RPC if the proposed term has already begun.

When is a transaction considered "safe" or "live"?

After leader replicates log to all the nodes and they agree, a commit to the log happens.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

No

How much time does a node need to reach a decision?

It depends on many factors, but it's a little unclear on what you mean by "reach a decision". The Raft paper has an analysis on how quickly leader election can occur once the current leader is found to be unresponsive.

What is the number of current and planned validators?

The Leader.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

"In standard Raft, you need to replicate a log entry to a majority of nodes in the cluster before committing it. For BFT consensus algorithms, including Tangaroa, the required quorum size is 2f + 1, where f is the number of failures you want to tolerate (including both crashed nodes and compromised nodes).

Yes. A BFT Raft cluster that tolerates f Byzantine failures must contain at least n ≥ 3f + 1 nodes, where n − f nodes form a quorum."

Is there a forking vulnerability?

No

What process does the system follow when it receives data?

Each replica in BFT Raft computes a cryptographic hash every time it appends a new entry to its log. The hash is computed over the previous hash and the newly appended log entry. A node can sign its last hash to prove that it has replicated the entirety of a log, and other servers can verify this quickly using the signature and the hash.

What process does the system follow when it receives data?

The data comes from clients of the Raft cluster, who send requests to the leader. The leader replicates these requests to the cluster, and responds to the client when a quorum is reached in the cluster on that request. What constitutes a "request" is system-dependent.

How is data currently stored?

How data is stored is system-dependent. It's important for state to persist to disk so that nodes can recover and remember information that they have committed to (which nodes they voted for, what log entries they have committed, etc.). The protocol can't work without this.

Governance, Risks and Control

How is governance / controls enforced?

"BFT Raft allows clients to interrupt the current leadership if it fails to make progress. This allows BFT Raft to prevent Byzantine leaders from starving the system.

Like Raft, BFT Raft uses randomized timeouts to trigger leader elections. The leader of each term periodically sends heartbeat messages (empty AppendEntries RPCs) to maintain its authority. If a follower receives no communication from a leader over a randomly chosen period of time, the election timeout, then it becomes a candidate and initiates a new election. In addition to the spontaneous follower-triggered elections BFT Raft also allows client intervention: when a client observes no progress with a leader for a period of time called the progress timeout, it broadcasts UpdateLeader RPCs to all nodes, telling them to ignore future heartbeats from what the client believes to be the current leader in the current term. These followers will ignore heartbeat messages in the current term and time out as though the current leader had failed, starting a new election."

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Heartbeats

How does the consensus mechanism allow access?

A Byzantine node can decide to arbitrarily increase the commit index of other nodes before log entries have been sufficiently replicated, thus causing safety violations when nodes fail later on. BFT Raft shifts the commit responsibility away from the leader, and every node can verify for itself that a log entry has been safely replicated to a quorum of nodes and that this quorum agrees on an ordering.

How does the consensus mechanism restrict access, concerning malicious activities?

BFT Raft allows clients to interrupt the current leadership if it fails to make progress. This allows BFT Raft to prevent Byzantine leaders from starving the system.

What is the permission management process? What is the process for adding or deleting nodes?

Raft's mechanism for changing the set of servers in the cluster uses a new joint consensus approach where the majorities of two different configurations overlap during transitions. This allows the cluster to continue operating normally during configuration changes.

How does the protocol assess the trustworthiness of other participants?

Election leader timeout.

Are there separate admin / administrator privileges? Who manages them?

No

Are there restriction (/) privacy rights defined and enforced by node?

Nodes can stop byzantine leaders from committing to the log.

Can a node or a user have only "Read" or only "Write access? Is specific node access required if only performing one functionality? (e.g., back–office outsourcing)

Leader Append-Only everyone can write but leader approves log, replicates log and sends it out to all nodes.

What are the measures in place to reduce risk?

Tangaroa has used BFT with raft to stop byzantine leaders from starving the system.

Performance

How do you measure scalability?

This depends a lot on the system's characteristics. In general though, the more nodes you have in a cluster, the more work the leader has to do to commit each log entry, because it must always be replicated to a majority of the cluster.

Is there a limitations on the number of fields within a transaction?

No

Is the speed of the system impacted if the system is made more scalable?

Slow nodes do not slow the system.

Security

Does the consensus mechanism utilize Digital Signatures?

Yes BFT Raft uses digital signatures extensively to authenticate messages and verify their integrity. This prevents a Byzantine leader from modifying the message contents or forging messages.

In case of a breach, what data is at risk?

Implementation dependent.

How is the system expected to address general server issues?

No

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

It is the leader who synchronizes and sends out a replicated log.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Clock is managed by the central node. Raft's notion of a term does away with having to keep centralized clocks.

Under which conditions does a lock/un–lock happen? (i.e., what is the proof safety?)

Raft uses a replicated finite state approach to transactions rather than locking. We model our transactions on this basis.

What is the process for disaster recovery?

Read the raft papers searching for "stable store" or similar. The whole point is that the raft log items, once accepted, are guaranteed to be on stable storage. So, in the event of a total outage, the cluster should just come back alive. Similarly, you only need one copy of the FSM to restore the whole lot. The details of the process will depend both on the implementation and on operational procedures by the end user (e.g., how backups are taken). Note the fact that raft is designed to operate between data centers makes this easier. One minor challenge is that the state of the peers is itself in the FSM, so if you lose a lot of peers, adding back peers to form a quorum needs thought. From memory this is covered in chapter 4.

Privacy

Does the system require verifiable authenticity of the messages delivered between the nodes?

Note that TCP itself will prevent most corruption and ensure retransmission, and most users use TCP (with or without encryption on top). Note that Raft does not have to run over TCP (I submitted a UDP version for instance, and running over say ZeroMQ would be quite feasible).

How does the data encryption model work?

Whilst encryption is out of Raft's scope – yes you can encrypt.

Are participants' identities hidden from one another? (e.g., Blackpool)

No

Cryptography/Strength of Algorithm

Does the consensus mechanism require a leader?

Yes

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

"Not 100% on what you mean by 'strictness'. Raft consensus is achieved by majority for a transactional write and is designed to provide hard guarantees.

Implementation dependent. Many implementations are inherently flexible (See Etcd/Raft / Hashicorp/Raft). I'm sure it would be possible to design something gross that is hard coded."

How is node behavior measured for errors?

The raft specs do not specify measurement. Most implementations measure some aspects (e.g., with monotonic counters), and often send them off to a stats server. Grep for “stats” in Hashicorp/Raft for instance. If you mean “How does the algorithm tell whether a node has errored?“ it simply looks for either the absence of a valid reply or heartbeat within a timeout, or the presence of an invalid reply or heartbeat (where “valid” and “invalid” are determined by the spec).

Tokenization (if used)

Does the consensus mechanism utilize transaction signing?

No

Implementation Approach

What is the implementation cost?

Transport Dependent.

What is the time required to implement?

Transport Dependent.

19. Tendermint

Source: Interview / Questionnaire

Contact name: Jae Kwon (jae@tendermint.com)

Questionnaire responses

Consensus Methodology

How many nodes are need to validate a transaction? (percentage vs. number)

More than two thirds of the validators need to be online.

Do all nodes need to be online for system to function?

More than two thirds of the validators need to be online. (Same as above)

Does the algorithm have the underlying assumption that the participants in the network are known ahead of time?

Yes and no. Validator sets can change according to TMSP application logic.

Who has ownership of the nodes? (e.g., consensus provider or participants of network)

Implementation dependent.

If applicable, what conditions are needed to be met to enter and exit each stage of the consensus mechanism?

See spec on GitHub. https://GitHub.com/tendermint/tendermint/wiki/Byzantine-Consensus-Algorithm 1) Propose 2) PreVote 3) PreCommit 4) Commit.

If applicable, what is the voting process after the "propose" stage?

In the ideal case, two phases of signatures (two phase commit).

When is a transaction considered "safe" or "live"?

A system is considered safe or live… not transactions. A transaction is considered committed when a block includes that transaction, and the block's Merkle hash root gets signed by > two thirds of validators with a pre-commit vote, at the same height and round number.

Are there multiple rounds of vetting to decide which set of transactions are going to make it into the next round of consensus?

Ideally happens in one round. If the primary proposer for a block isn't online, will need to move onto next round, etc. Otherwise no, not sure what multiple rounds is referring to.

How much time does a node need to reach a decision?

Blocks commit on the order of one second. Depends on network size, physical distance (global vs. localized), etc.

How much time is actually needed to build the consensus until a new block is added?

Depends on parameters, but on the order of one second for typical applications.

Does system contain synchronous node decision making functionality?

The system is partially synchronous. It's mostly asynchronous, with some sensible timeouts built in.

What is the number of current and planned validators?

One to thousands, depending on requirements. A 50-node global network can still commit blocks on the order of a second.

What is the Fault Tolerance? How many nodes need to be compromised before everything is shut down?

"Can tolerate up to one thirds of Byzantine nodes."

Is there a forking vulnerability?

Only if BFT threshold is exceeded.

How are the incentives defined within a permissioned system for the participating nodes?

It's undefined. It can be defined in the TMSP application (similar to Hyperledger's chaincode).

What process does the system follow when it receives data?

Depends. There's a mempool module for transaction sharing prior to consensus, as well as a consensus module. These modules are multiplexed on the same TCP connection between peers. The consensus module runs on its own consensus sub–channel. Details can be found here: https://GitHub.com/tendermint/tendermint/wiki/Byzantine-Consensus-Algorithm.

How does a party take ownership of an asset?

Depends on the application.

Governance, Risks and Control

How is governance / controls enforced?

"We're creating a governance system called Governmint. We're creating a governance system called Governmint. http://GitHub.com/tendermint/governmint TMSP Governance Layer A simple voting system that enables itself to evolve over time. Entities are identified by a pubkey. Members are entities associated with a group; can vote on proposals for that group. Groups are collections of members Votes are cast on proposals by members proposal types: GroupUpdateProposal: change the group membership, etc. GroupCreateProposal: create a new group VariableSetProposal: set a variable value TextProposal: create a human readable proposal SoftwareUpgradeProposal: upgrade software TX types ProposeTx to propose something for a group to vote on CastTx to vote on a proposal"

Who is responsible and what are they responsible for in case of malicious actions within the network? How does legal action take place?

There is a method of determining liability in the case of a fork. BFT algorithms don't typically guarantee this.

Is there an intrinsic penalty mechanism in place for an attempted corruption of the consensus?

Yes, for a successful corruption.

How does the consensus mechanism allow access?

Currently each node is configured manually, but later we'll re–introduce peer-exchange functionality. It's based on gossip, like Bitcoin. Nodes don't have to be (active signing) validators, they can still help with block/vote propagation.

How does the consensus mechanism restrict access, concerning malicious activities?

"Several mechanisms.

In the p2p networking layer, a system of multiplexing many connections with fairness. In the mempool layer, CheckTx TMSP messages to check for transaction validity before propagation to peers. In the consensus layer, public key infrastructure to ensure that correct validators sign blocks."

What is the permission management process? What is the process for adding or deleting nodes?

TMSP has basic notion of validator set changes, but other permissions are handled by the app. Similar to adding a Bitcoin node, except every validator needs its public key added to the validator-set of the Blockchain from the TMSP app.

How does the protocol assess the trustworthiness of other participants?

Pubkey infrastructure.

Are there separate admin. (/) administrator privileges? Who manages them?

No, depends on the application.

Are there restriction (/) privacy rights defined and enforced by node?

Depends on the application. Tendermint Core has no assets besides validator voting Proof of Worker, which isn't natively fungible.

Performance

How long does it take for transactions to be validated and/or consensus to be achieved?

"- Transactions are pushed asynchronously via TMSP to the app. Unix sockets are very fast, so the bottleneck is mostly in the application. Tendermint Core can handle around 10k TXs/sec for 250byte size transactions."

Provide some general measures of volume that the consensus mechanism can or will handle (e.g., number of trades)

Depends on the app. Tendermint Core can handle around 10k txs/sec for 250byte size transactions.

Provide some general measures of the value that the consensus mechanism can or will handle (e.g., $ value of trades)

Unlimited.

How do you measure scalability?

"● Number of validators

● Compute/network limitations of validators

● Number of txs/sec on an empty TMSP application

● Average size of txs"

Is the speed of the system impacted if the system is made more scalable?

No. Tendermint is ideal for scaling. Have as many shards as you want.

Security

How is transaction activity monitored?

RPC APIs

Does the consensus mechanism utilize Digital Signatures?

Yes. Consensus relies on digital signatures.

How does the consensus mechanism address an assumed industry standard?

"It's an optimal BFT algorithm that can also provide monetary guarantees on the security of the Blockchain.

●For the Blockchain to fork, > one third need to be Byzantine

●When a fork occurs, you can determine liability

●Validators can have collateral posted on a Blockchain or elsewhere (e.g. distributed)"

Which risk/security issues are currently being worked on?

Various testing.

Are there any plans for getting the application/consensus mechanism certified (e.g., ISO, SOC, etc.)?

No, there aren't many people that can certify consensus algorithms. We will always be working toward formal proof systems. More academic peer review.

Briefly describe the security testing performed till date (if any)

We just finished a round of network integration tests. The last test found a subtle bug in the gossip logic in the case of malicious double-signing.

In case of a breach, what data is at risk?

The private key of a validator. It's assumed that the Blockchain information is public (e.g. no privacy except what is provided by the application)

How does the system prevent signature fraud (e.g., stolen keys)?

Outside our scope. Complemented with trusted computing (e.g., Intel Sawtooth)

Does the consensus mechanism have full documentation in place?

No.

How does the consensus mechanism address the risk of "double spending"?

Total ordering of blocks, as in Bitcoin, except without the need for confirmation blocks.

How does system ensure network synchronization? And, what is time needed for the nodes to sync up with the network?

Depends on the parameters of the application, e.g., application compute complexity, txs (/) sec etc.

Do the nodes have access to an internal clock/time mechanism to stay sufficiently accurate?

Typical computer. An accurate clock is not required.

Under which conditions does a lock/un–lock happen? (i.e., what is the proof safety?)

"See spec on GitHub. https://GitHub.com/tendermint/tendermint/wiki/Byzantine-Consensus-Algorithm Proof of Safety

What is the threat model being tested? What has been defined as “normal”? How do you monitor fraud?

Highest threat level.

Privacy

How does the system ensure privacy?

Up to the application. Blockchain txs and application state should be public. Use various encryption techniques, e.g., homomorphic encryption.

Does the system require verifiable authenticity of the messages delivered between the nodes?

No, messages are signed, signatures are included.

Do all nodes have visibility into all other transactions?

Yes.

If consensus happens in a permissioned network are random public keys issued for every single transaction to increase the privacy? Or does randomized CUSIP translation factors take place?

Depends on the application. A TMSP application can be like UTXO, or not.

Cryptography/Strength of Algorithm

What is the library approach?

Best open-source practices.

Does the consensus mechanism require a leader?

Round robin leaders, for every block height and every round. (Ideally a block is found in one round). A single leader (proposer) at a time. But everyone is supposed to vote. The leader is not required for aggregating signatures.

How strict is the consensus mechanism? (Is the system strictness hard coded, or built with code flexibility?)

TMSP is a socket protocol. It provides ultimate flexibility. We're also developing a few TMSP applications ourselves, but this survey doesn't address those.

Does the consensus mechanism require a leader?

Consensus has a leader.

Tokenization (if used)

Does the consensus mechanism utilize transaction signing?

Blockchain solves this problem by Public-Private key pairs for signatures on and verification of transactions. Tangaroa's protocol specifies using a similar system, but at the consensus level as well. This provides a means for one node to validate that a message came from another node (so long as keys haven't been compromised).

Implementation Approach

What are the current uses cases being explored, tested or implemented?

Private networks in enterprise environments, either intra– or inter–firm.

What is the implementation cost?

Depends on the domain.

What is the time required to implement?

Depends on the domain, but pretty fast at this point.

Contact Us and Acknowledgements

Bill Cline
KPMG LLP
Principal - Advisory,
Financial Services Strategic Capabilities & Alliances Lead
704-335-5552
wcline@kpmg.com

Sigrid Seibold
KPMG LLP
Principal – Advisory
Capital Markets
917-971-5880
sigridseibold@kpmg.com

George Samman
twitter: @sammantic
blog: sammantics.com

Acknowledgements: We'd like to acknowledge the contributions of numerous people in the Blockchain network, many of whom reviewed and verified portions of this paper:

KPMG LLP: Bob Hayward, Kiran Nagaraj, Walter Murphy, Mihai Liptak, Roshan Rao, Burak Karvan and Francis Sam Yesurathinam

BigChainDB: Trent McConaghy – gtrent@gmail.com
BitShares 2.0: Ryan R. Fox – ryan@ryanrfox.com
CASPER: Vlad Zamfir – vldzmfr@gmail.com
Directed Acyclic Graphs: Aviv Zohar – avivz@cs.huji.ac.il
Distributed Concurrence: Dan Conner – dan.conner@disledger.com
Evernym: Jason Law – jason@evernym.us; Timothy Ruff – timothy@evernym.us; Drummon Reed – drummond@respect.network
Graphene: Ryan R. Fox – ryan@ryanrfox.com
MultiChain: Gideon Greenspan – gideon@coinsciences.com; Maya Zehavi – mayazi@gmail.com
OpenChain: Flavien Charlon – flavien.charlon@coinprism.com
Ripple: Bob Way – bob@ripple.com
Steem: Ryan R. Fox – ryan@ryanrfox.com
Stellar: Jed McCaleb – jed@stellar.org; Joyce Kim – joyce@stellar.org
Tendermint: Jae Kwon – jae@tendermint.com

Note: Ryan R. Fox does not represent Cryptonomex, Inc., Steemit, Inc., nor any other entity within the Blockchain space. His responses are his own informed opinions based upon independent research.