VISION HILL
RESEARCH

A Distributed Consensus Case Study

July 2, 2019

A Basic, Universal Framework to Understand Distributed Consensus

In November 2018, Preethi Kasireddy introduced a terrific write-up on distributed consensus. It offers robust coverage on the basics of distributed systems and their history.

At a high-level, distributed systems are about tradeoffs. Every consensus algorithm can generally be broken down into a three-step process: 1) elect, 2) vote and 3) decide. There are various ways to execute this process.

Historically, in distributed systems engineering, we have not been able to reach consensus in an asynchronous and byzantine fault tolerant ("BFT") manner. Let's break these terminologies down:

"Asynchronous"

The term "synchronous" refers to simultaneous action or occurrence. Therefore, "asynchronous" is the opposite of synchrony; it refers to the absence or lack of concurrence in time. Synchronous systems assume a perfect network where nodes are organized and deliver messages within a defined time bound. This doesn't match reality because distributed systems lack a global clock. Instead, they need a way of determining the order or sequence or events happening across all computers in the network. There are ways to resolve this, most notably: 1) partial synchrony and 2) asynchrony.

Partial synchrony lies somewhere between synchrony and asynchrony. It introduces the ability to make certain time-bound assumptions but limits their impact. Consensus can be reached regardless of whether the time bounds are known.

With asynchrony, consensus is not reached in a fixed time (no fixed upper bounds exist). It is assumed that a network may delay messages indefinitely, duplicate them, or deliver them out of order. This is often closer to reality for real-world systems.

"Byzantine Fault Tolerance"

For those unfamiliar with the Byzantine Generals' Problem, it is an age-old computer science problem whereby components of a system may fail and there is imperfect information on whether a component has failed. The term takes its name from an allegory in which a group of Byzantine army generals formulate a plan for attacking a city. In its simplest form, the generals must decide only whether to attack or retreat. Some generals may prefer to attack, while others prefer to retreat. The important thing is that every general agrees on a common decision, for a halfhearted attack by a few generals would become a rout, and would be worse than either a coordinated attack or a coordinated retreat.

The problem is complicated by the presence of treacherous generals who many not only cast a vote for a sub-optimal strategy, they may do so selectively. For instance, if five generals are voting, two of whom support attacking while two others are in favor of retreat, the fifth general may send a vote of retreat to those generals in favor of retreat, and a vote of attack to the rest. Those who received a retreat vote from the fifth general will retreat, while the rest will attack (which may not go well for the attackers). The problem is complicated further by the generals being physically separated and having to send their votes via messengers who may fail to deliver votes or may forge false votes. The incentive for the fifth general to carry out malicious behavior in this example is the possibility that post-rout, all armies have effectively been weakened to the point of vulnerability , such that the fifth general and his army could perhaps invade the city and claim full power (without having to share such power with the other generals). In conclusion, the Byzantine Generals Problem pertains to

the fact that actors must agree on a concerted strategy to avoid catastrophic system failure, but some of the actors are unreliable.

Byzantine failure inherently implies no restrictions, and makes no assumptions about the kind of behavior a node in a distributed network may pursue. In a Byzantine system, nodes have different incentives and can lie, coordinate, or act arbitrarily, and one cannot assume a simple majority is enough to reach consensus. The premise of a Byzantine fault tolerant ("BFT") system design is that the nodes in the network should not be dependent on the actions of a single node in any relevant way whatsoever.

Why is it so challenging to design asynchronous BFT networks?

Building a consensus algorithm that is both BFT and asynchronous has been incredibly challenging historically. Two primary concerns for reaching BFT and asynchronous consensus emerged: liveness vs. safety.

At a very basic level, "liveness" refers to a situation where a lack of consensus will cause a network partition (aka, a fork). As such, a fork-choice rule is implemented in these systems (more on that below). Safety, on the other hand, refers to halting a system until consensus is reached.

Enter bitcoin.

The brilliance of bitcoin is not exactly the underlying technology (it's a C++ coded database after all), but the game theory that cracked the BFT and asynchronous challenge (favoring liveness). In proof-of-work systems, all nodes don't have to communicate to achieve termination (e.g., agree on a final state) and transition to the next state. Rather, they agree on the *probability of the state being correct,* as determined by the node that can solve the mathematical proof-of-work puzzle the fastest.

Since bitcoin's birth, developers have been pushing the boundaries of base layer ("layer 1") innovation to understand how different tradeoffs might impact what can ultimately be achieved not just at the base layer, but also with any other use cases and applications built atop them.

Proof-of-work is arguably the most secure mechanism design known to date. For a use case to potentially capture trillions of dollars in value, high security is vital. But at what point can one draw the line between "extremely secure" and "secure enough? What is "sufficiently secure"? And could proof-of-stake systems be sufficiently secure?

To understand the current state of proof-of-stake innovation, it's important to understand the limitations of proof-of-work. Proof-of-work security is a function of two scarce expenditures (contributions): 1) energy and 2) time. Proof-of-work proponents argue the act of hashing in proof-of-work provides unforgeable costliness, something that is not available in proof-of-stake systems.

On the contrary, proof-of-stake proponents argue this view is limited as there's another scarce contribution captured in proof-of-stake: the opportunity cost of capital. The opportunity cost of capital is a foundation of financial theory. Locked-up (staked) capital is subjected to the opportunity cost of investing that capital elsewhere. If we want to better understand whether proof-of-stake is "secure enough", then we should learn about the attack vectors and some solutions modern proof-of-stake networks can use to combat them and how that may impact their ability to reach distributed consensus.

First up: Sybil attacks.

This term is generally tossed around loosely by many in the industry. This is a type of attack seen in peer-to-peer networks in which a node in a network operates multiple identities actively at once and undermines and/or subverts the authority in reputation systems by flooding the networks with duplicate votes.

At quick glance, it's easy to think Sybil attacks aren't an issue with proof-of-stake systems. After all, staking is inherently something that takes 'stake' (e.g., skin in the game), which is what should prevent a Sybil possibility in the first place.

Assuming proper design, this may be the case, but in the absence of a true cost of staking or reputation system, Sybil attacks can and do occur. There is not necessarily always an external cost to staking (e.g., borrowing, forks, delegation). Many view staking off a basis of an inherently limited resource, where as long as weight is proportional to that resource, it doesn't matter if its split across a bunch of Sybils or placed in one entity.

This isn't always the case. The EOS blockchain, for instance, doesn't have this: block producers are elected by token holders, so Sybil attacks for combatted via a reputation system – they can be voted out if they are seen to be malicious in any way.

In conclusion, Sybil attacks can be combatted via (external) costs to stake or a reputation system. Some popular projects implementing the cost methods are Ethereum (in the Ethereum 2.0 roadmap), Casper Labs, Solana Labs, Tezos, DFINITY, Near Protocol, Oasis Labs, Thunder Protocol and the Polkadot Network.

Some projects implementing the reputation method are EOS, Cosmos, Tendermint, and Terra.

Second: Distributed Denial of Service (DDoS) attacks.

DDoS attacks are cyber attacks in which a perpetrator seeks to make a network resource unavailable to its intended users by temporarily or indefinitely disrupting services.

As we understand it, several methods can combat this starting with the introduction of unbiasable randomness (entropy) to the consensus election process (recall from earlier that every consensus algorithm can generally be broken down into a three-step process: 1) elect, 2) vote and 3) decide). This takes the form of a verifiable random function ("VRF").

In simplest terms, randomness serves as a defense mechanism against front-running the election process in distributed consensus. There are three common approaches to constructing VRFs: 1) a verifiable delay function ("VDF"), 2) a commit-reveal scheme, and 3) BLS signatures.

1. A verifiable delay function ("VDF") is a function that enables a time delay imposed on the output of some pseudo-random generator. VDFs are extremely computationally expensive functions that cannot be parallelized (an attractive security feature).

   The delay prevents malicious actors from influencing the output of the pseudo-random generator, since all inputs will be finalized before anyone can finish computing the VDF. Ethereum 2.0, Casper Labs, Solana Labs, and Tezos are some popular proof-of-stake projects using VRFs via VDFs.

2. In a commit-reveal scheme, a validator commits to a chosen value while keeping it hidden from others, and then reveals the committed value later after hashing. These schemes are

designed so committed values cannot be changed and randomness cannot be manipulated. Algorand and Ouroboros Genesis consensus (Coda Protocol, Cardano) are proof-of-stake networks using VRF and commit-reveal schemes.

3. BLS Signatures are a form of "threshold" signatures whereby bi-linear pairing (e.g., defined M of N multisig) is used for verification. Long post here.

These are dense topics, so let's digest. Randomness helps combat DDoS vulnerabilities because it challenges the ability for a perpetrator to front-run the election process in distributed consensus. In this section, we discussed a verifiable random function and three common approaches to construct it.

Another way to potentially combat against DDoS vulnerabilities is to have an identifiable, fixed committee of validators that are old-fashioned trusted as each one injects security deposit-based collateral into a network that can be slashed (lost) in the event of malicious behavior. Cosmos, Tendermint, EOS, Near Protocol, Thunder Protocol, Polkadot Network and Terra are some examples of popular proof-of-stake projects that implement fixed committees with security deposit-based collateral.

Third: Stake grinding attacks.

Stake grinding is an attack in which an attacker manipulates the blockchain in a way to maximize the probability of being elected as the leader for the ensuring block(s). An obvious example of a stake grinding attack would be when an attacker has a small amount of stake and goes through a blockchain's history to find places where his/her stake wins a block. In order to consecutively win, the attacker modifies the next block header until he/she wins again.

Stake grinding can generally be combatted somewhat similarly to how DDoS is combatted, with unbiasable randomness implemented in the election process or fixed committed that have injected security deposit-based collateral.

Fourth: Nothing-at-stake attacks.

Nothing-at-stake attacks are ones in which a validator in a committee votes (without penalty) for two blocks at the same block height. Simply put, the nothing-at-stake problem allows someone to misbehave – and get away with it.

There are generally two ways to combat nothing-at-stake attacks: 1) bond deposits (e.g., security deposit-based collateral that can be slashed in the event of malicious action), or 2) implement a fork-choice rule. This goes back to liveness vs. safety. Recall that liveness refers to a situation where if there is no consensus, a network partition (fork) will occur and a fork-choice rule is implemented. On the contrary, safety refers to halting a system until consensus is reached.

Slashing is generally straightforward - if one behaves badly, he/she is financially penalized. This is common with networks that favor safety over liveness (see Cosmos, Tendermint, Near Protocol, Oasis Labs, Polkadot Network, Tezos, Celo and Terra).

Fork-choice rules are slightly more complex and are inherent in networks favoring liveness over safety. Some popular proof-of-stake projects favoring liveness are Ethereum 2.0, Casper Labs, Solana Labs, Algorand, DFINITY, Thunder Protocol and Ouroboros Genesis consensus (Coda Protocol, Cardano). More on this later.

Fifth: Network partitions.

The technical definition of a network partition is one in which a network splits between nodes due to the failure of network devices. Simply put, if consensus is unable to be reached, and a network favors liveness over safety, the network will fork and two chains will exist alongside each other (hence the term "fork-choice rule").

To understand the premise behind the fork-choice rule, recall that in proof-of-work networks, the probability of miners winning the next block is proportional to the percentage of total hashpower contributed.

Bitcoin combines the fork-choice rule with the requirement that miners produce proofs of work (solving algorithmically complex mathematical puzzles), which are costly to generate because each miner has limited hashpower that is expensive to acquire. Any hashpower improperly spent on the wrong chain is wasted.

This is why in the event of network partitions (forks), little value is justified for minority chains by the market (the market makes a choice), as we have evidently seen (bitcoin vs. bitcoin cash is a famous example).

We've now covered ways to combat Sybil attacks, DDoS attacks, stake grinding attacks, nothing-at-stake attacks and network partitions, five commonly known attacks in blockchain land. Readers should notice, with the exception of bitcoin, all examples of networks used were proof-of-stake based. Why was that?

It's simple. This case study was an effort to explore if a line can be drawn between "extremely secure" and "secure enough". Can proof-of-stake networks be "secure enough" (vs. proof-of-work) to still capture billions, if not trillions, of dollars in value depending on the use case?

Some investors in the blockchain industry are already questioning if security premiums can develop irrespective of monetary premiums. We hope more work gets done on this subject.

Our Conclusion

To conclude, we share our key takeaways from this exercise and examine them in the context of a risk/reward framework. We believe everybody working in the blockchain industry should have a basic foundational understanding of these technologies and how they work, especially those managing investor money with a fundamental focus.

It boils down to risk management. These are technological assets that are in their early stages and are highly experimental in nature. Without understanding the technicals, it's not hard to chase high reward potential and greatly misprice the risks while at it.

This is our first key takeaway – the need for technical diligence. Investors should understand design vulnerabilities to anticipate cracks that could potentially form in a network in the future. Investors should understand what solutions can be implemented and whether such solutions are backward compatible.

Our second key takeaway from this case study is the importance of understanding *why* tradeoff decisions are made. Developers ultimately want to build atop a protocol they can trust. Self-sovereign or not, it's important to know what these builders are attracted to. They're shaping the direction of our future after all.

The same can be said for investors. Identify the patterns or trends that are forming as a result of alignment between investors and engineers. The stronger and more widespread the alignment, the greater the systemic de-risking of a given network. Decentralized networks are a social coordination game; the greater the social endorsement, the larger the "stickiness", and the higher the justified network value for a given network.

Lastly, our final takeaway – security is a spectrum. Just as too much working capital can be inefficient for a company (because it means the amount of money available within the company is much more than what's needed for operations), too much security can be inefficient as well.

We question whether a security premium can develop irrespective of a monetary premium for base layer networks. A counter to this is that all base layer protocols are competing to be some form of money. But not all data or information is monetary or financial in nature (e.g., identity, medical records, etc.).

We believe one of the most significant experiments that will take place over the next decade will be whether decentralized information networks can accrue substantial value without necessarily representing financial assets. Time will tell.

Grateful thanks to Preethi Kasireddy, Kyle Samani, Jason Choi, Spencer Noon and so many others that inspired us to push this forward.

Disclaimers

The information in this report and the presentations therein is for informational purposes only, was prepared by Vision Hill Research, is believed by Vision Hill Research to be reliable and, wherever appropriate, has been obtained from sources believed to be reliable. The content provided herein should not be considered investment advice, and is not a recommendation of, or an offer to sell or solicitation of an offer to buy, any particular security, strategy, or investment product. Vision Hill Research makes no representation as to the accuracy or completeness of such information. Opinions, estimates, and projections in this report and the presentations therein constitute the current judgement of Vision Hill Research and are subject to change without notice.

Vision Hill Research has no obligation to update, modify or amend this report and the presentations therein, or to otherwise notify a reader thereof in the event that any matter stated herein, or any opinion, projection, forecast or estimate set forth herein, changes or subsequently becomes inaccurate for whatever unbeknown or unforeseen reason.

Vision Hill Group is a full-service digital asset management and solutions firm that aims to lead investors into the future of digital assets. Vision Hill Group brings together a team with extensive experience in traditional financial markets, a deep passion and understanding of crypto and digital asset markets, and a history of risk and portfolio management. Vision Hill Group has three primary divisions: Vision Hill Asset Management, Vision Hill Research, and Vision Hill Advisors. Vision Hill Asset Management is the multi-strategy digital asset investment management division of Vision Hill Group. Our platform offerings enable us to invest across multiple strategies, investment theses, and time horizons to provide broad-based and diversified exposure to the evolving digital asset class. Vision Hill Research curates and distributes in-depth research and analytical insights for institutional clients and accredited investors that seek trustworthy, unbiased market intelligence and guidance in the nascent digital asset industry. The Advisory division of Vision Hill Group provides investment portfolio management and advisory services to institutional clients and accredited investors seeking bespoke solutions and consultation to assist them in achieving their investment objectives.

Vision Hill is indirectly invested in some of the networks mentioned in this report. The content provided herein should not be considered investment advice, and is not a recommendation of, or an offer to sell or solicitation of an offer to buy, any particular security, strategy, or investment product. Vision Hill has not received and will not be receiving any compensation as a result of this report.