Chapter 2

Solana Basics

Solana the Blockchain

Solana is a blockchain like many others you may be familiar with. The blockchain differentiates itself from the others with its low block times (<400ms), and high transaction throughput (~4000 transactions per second as of writing this in April 2023). The blockchain leverages clever engineering at the software layer, as well as high performance machines (fast multicore CPUs, GPUs, and lots and lots of RAM) used for the validators. With transaction confirmation times less than a second, it gets as close to "instantaneous" as any other chain, currently.

Account Model

Like any of the EVM chains you may have used, solana uses an account model (versus a UTXO model like in bitcoin, or other bitcoin-derived networks). Whereas bitcoin has UTXOs (unspent transaction outputs) to denominate coins on the blockchain, for solana, everything is just data written to an "account" somewhere. Each account is "owned" by another program. Balances, coin details, and things like lockups are just data that are stored in the account. The types of information vary by the program to which an account belongs, and the details are essentially limitless (up to the max account size). For coins, solana accounts are owned by the `system` program, stake accounts by the `stake` program, and SPL tokens (solana's ERC-20 equivalent) owned by the `spl-token` program. SPL tokens are specifically very useful, as they can represent everything from margin positions, staked assets, collateral held in a borrow-lend program, tokenized real-world assets (RWA), equities and commodities, to non-fungible tokens (NFT). In addition to the account model's ability to represent assets, it is even generally useful as a vehicle for on-chain storage, to represent data points that can only be written to by cryptographic keys, and therefore can be a trusted source of information. The obvious application here is price and data *oracles*, but the data could represent everything from tweets on a decentralized social network, to a security deposit or bank account balance, as random examples.

Stateless Programs

While both Ethereum and solana share an account model, a big departure from ethereum is the fact that solana programs separate state (data) from execution. What this means is that while a smart contract is executed, it doesn't lock the state for all of the associated program/token accounts. Each program invocation, then, essentially acts as an atomic API call. Accounts that do not have overlapping state changes (requiring sequential execution, where ordering is important) can therefore be executed concurrently. The separation of programs and state allows non-contending transactions to be executed in parallel. As a result, solana users benefit from the assurance that, if they are not interacting

with a popular account (program or dApp), then their transaction should land in a block immediately and be confirmed quickly and affordably.

Proof of History

Solana relies on a clever invention known as "proof of history" or PoH. Co-founder Greg Fitzgerald explains it very nicely in an early Solana podcast as a "water clock." A water clock drips smalll drops of water, which, as the water accumulates, the water level can serve as a measure of the passage of time. In proof of history, the leader or PoH generator repeatedly runs the SHA256 hash function piping its output to its input. Since each hash must be executed fully, and the output cannot be known without completely running the function, each hash represents a discrete tick of time, analogous to each drop in the water clock. While the computation of the hash function has some variability, on average it is predictable, and with each function output depending on the input, we can reliably see the procession of time from start to the end state. PoH is based on the idea of a verifiable delay function, which serves the same purpose: to securely demonstrate the passage of time. Some tech jargon used to describe PoH mechanics are "pre-image and collision resistant [SHA256] hash" [1], which is "is impossible to parallelize without a brute force attack using 2^128 cores" (the tick function cannot be forged).

Great, but why do this? With the clock marching forward, the leader now mixes in user messages to the input of the PoH generator's hash function. With the generator's output dependent on this user message, we can verifiably say that the transaction existed at a certain place in time. This is also analogous to getting your picture taken while holding up a copy of the New York Times–a proof that the picture or event could not have occurred before a certain date. This creates a lower bound on time. And since adding the event to the hash unpredictably changes its output, we also have the upper bound on time–since no history is correct if it doesn't contain the user message used to generate the subsequent hashes. This is the secret sauce that facilitates transaction ordering, provides robust protection against chain reorganization (reorgs), and ultimately enables such a fast time to finality for user transactions.

How is this different than other chains? Bitcoin uses a similar behavior in proof of work, with an algorithm with adjustable difficulty, to target block times of about ten minutes. However, since bitcoin relies on finding a nonce (hash string with a lot of zeroes in it) by repeatedly performing hashes, the time to find one is totally random, and blocktimes can vary from just a few seconds, to sometimes more than an hour. Bitcoin's concept of time is very relaxed in the short term, but assuredly demonstrates a positive passage of time with each new block.

Ethereum, on the other hand, does not rely on cryptographic algorithms to solve the clock problem. Instead, the system just takes the median block time according to many nodes' local clocks. Delays in block propagation (communication) and resultant block arrival times therefore leads to some error in the "true"

measurement, and ultimately puts a lower bound on the fastest block time possible.

1. proof-of-history—a-clock-for-blockchain

Solana 2.0

This is a set of improvements similar in significance to the Ethereum 2.0 milestone, which solana reached just a little bit quicker. These two improvements, delivered middle of 2022, are so important to solana's realization, that they get this title. The improvements are 1) priority fees with "localized" fee markets, and 2) stake-weighted quality of service (QoS), and adoption of Google's QUIC protocol. While the blockchain was already fast, this set of improvements removed the speed bumps that were rendering some services nearly unusable during network congestion, such as dexes during high volatility, or popular NFT mints. The improvements led to an increase in the average *transaction rate* (tx/sec, or *TPS*) to about 4000 TPS, from something that previously ranged 2000-3000 (>33% faster). Many factors affect transaction throughput, but these improvements were huge. The demarkation of solana 2.0 took the network from something that was excellent during the calm times, but buckled under pressure, to a network that could perform under relentless volatility. During the blowup of FTX, solana continued to confirm transactions around 3000 TPS, while its token price experienced a terrifying drawdown, and an annualized volatility >600% for over a week! Not even bitcoin or ethereum have experienced such an adverse event. The chain has proven it can handle volatility with grace now. More recently in April 2023, the popular MadLads NFT mint suffered days of DDoS-scale requests, yet, during the mint, the blockchain continued to process transactions at roughly 4000 TPS, with Anatoly remarking "that was boring." This event provides further confirmation that the chain has made dramatic changes that improve the user experience, and can handle excessive transaction volume. Ultimately, it has been demonstrated that the network is incredibly durable, and with the 2.0 improvements, significantly hardended to congestion, and volatility.

Localized Fee Markets Prior to these 2.0 upgrades, although solana had stateless execution, it had a different problem: a fixed transaction fee. The negligible transaction costs, combined with no means to pay for priority, meant that eager traders and bots would express their urgency by creating thousands of identical transactions, hoping that at least one of them would get observed by the transaction processing unit. This configuration led to probabilistic races to win important transactions like arbitrage and liquidations, to capture MEV. This widespread spamming by bots negatively affected user experience, by overwhelming the nodes and processes responsible for deduplicating and forwarding transactions. Users would send transactions, which would not be processed, nor even acknowledged; it was incredibly frustrating. The network's engineers needed to find a better way to deal with this.

After much debate, which many solana users viewed as antithetical to solana's core principles, it was agreed to implement the feature to allow users to pay an "additional fee" to express their desire for transaction priority. The hope was that bot operators might quit spamming the network with transactions, and thereby reduce the processing load on validators. The solution was to create a fee model not unlike the one prescribed by ethereum's EIP-1559, with base fee plus a tip, or priority fee. There are significant differences, however, in that block size is not adjusted to be approximately half full (as in the case with ethereum), and solana's base fee of 0.000005 (5x10^-6) SOL is neglibigly small, making fee market trades essentially "all tip, no base." We will not provide any judgment on whether this is a good of bad feature–it's just different. Also different is that while ethereum's base fee is 100% burned and the priority fee goes to the validator, for solana, half (50%) of the total transaction fee (base + priority) is burned, and the other half (50%) goes to the validator. This may present an issue where validators and searchers may team up to win priority auctions (the validator being able to rebate a portion of the additional fee to the searcher that outbids its competitors), but this has not yet been demonstrated as a problem.

A positive development in solana's fee markets that is different from ethereum's, is that the "gas" price is local to hotly contested pieces of state, whereas ethereum's gas price is universal and affects all transactions on the chain. This is possible because of solana's parallel execution model. As mentioned before, for transactions that do not touch the same state, blocks are packed in parallel. In this sense, for unassociated transactions, ordering is unimportant. Conversely, for a crowded DEX liquidation, the ordering *is* important, which is determined by a combination of latency, priority fee and finally, stake weight (to be discussed later). Only so many sequential transactions can fit into a block, and blocks are not typically full. This means that a leader can fill the remaining block space with non-priority transactions. Non-priority users' fees are mostly unaffected, and should reach confirmation in an acceptable amount of time.

How does it work in practice? Coin swappers and DEX traders experience smooth and reliable confirmations even with a modest priority fee. Ultimately, the priority or additional fee has provided significant relief to users who want to interact with popular dApps, where transactions may be time-sensitive.

QUIC Prior to the solana 2.0 moment, validators accepted UDP packets sent directly to the transaction processing unit from RPC nodes. UDP is a connectionless protocol, lacking both flow control and receipt acknowledgment [1]. While UDP enables fast, asynchronous communication, it has no controls over traffic, and as we saw with solana, can be abused with astronomical amounts of spam messages. Enter QUIC: a protocol built by Google, which allows asynchronous communication, like UDP, but with sessions and flow control like TCP. Unlike TCP, however, QUIC is consistent with its namesake in that it is extremely fast. QUIC was developed as an alternative to TCP, and is described as a multiplexed transport built on top of UDP, with its transport functionality

encrypted [2]. In addition to being fast, like UDP, it has additional controls for congestion management and feedback, and introduces the possibility to increase the maximum transaction size from its current limit of 1,232 bytes.

Source:

1. Solana Network Upgrades
2. QUIC

Quality of Service (QoS) The final key feature in the solana 2.0 upgrades is the addition of stake-weight quality of service (QoS). Prior to this change, the leader would indiscriminately accept messages on a first-come-first-served basis, without regard to the source of the messages. Leader bandwidth is limited, so, as you might suspect, non-aligned network actors were able to take advantage of this fact, and monopolize the resources that should be available to everybody. Since no consideration was given to the origin of the incoming packets, it was impossible to tell which transactions were legitimate, versus abusive. Because solana is a proof of stake network, with leader positions determined by the amount of stake a validator holds, it is logical to extend stake-weighting to the privilege of transmitting transactions to the leader. In this way, stake-weighted QoS gives every aligned actor (validator) a proportional opportunity to communicate with the leader, while filtering out messages from unqualified transaction spammers with no skin in the game. This arrangement provides some guarantee that if you use an RPC node that is affiliated with a validator with n% of total stake, then that validator will have n% of the bandwidth to the leader, and those transactions will reach the leader's TPU. In order of precedence, stake-weighted QoS is a secondary measure to priority fees, in terms of ordering competing transactions at the leader.

Solana 2.0 Recap The 2022 upgrades comprise a trifecta of capabilities that dramatically enhances the everyday user's experience when interacting with the chain. With priority fees, users can express the urgency of their transactions by their willingness to pay more to get to the front of the line. With QUIC, we have a fast, UDP-like communication protocol that also has the ability to check the origin of packets, throttle traffic, and generally manage connections from the senders. Finally, stake-weighted QoS provides assurances that reputable validators each get their turn to speak to the leader, because they won't be crowded out by redundant, garbage messages. The tangible evidence that these new controls work is that trades get settled and transactions get confirmed even during peak times, and they no longer just disappear, as was commonplace before.

Monolithic versus Modular

An impassioned debate that is not going to be settled soon is the question of whether to use a single, flat, or "monolithic" system that incorporates consensus, censorship resistance, and the execution environment at a single layer (*layer 1*,

or L1), or if another architecture that splits up consensus, data availability, and execution is better. The overarching argument is that, with enough adoption, every block chain will be constrained by block space, and then on-chain fees will need to increase to manage the congestion. The question is about scalability and how best to accomplish it.

Layer 2 Indeed, separating consensus (at the L1) and execution (using rollups or layer 2), which is the path that Ethereum has gone down, shows promise for adding bandwidth and reducing transaction costs, and enabling lower value transactions. This arrangement offloads execution from the L1, and proffers *some* security to the L2. While L2s can provide state compression and thereby reduce gas cost/transaction fees, there is no guarantee that they actually increase TPS. Indeed, without departing from the single-threaded EVM, an L2 or rollup is just ethereum with less traffic and lower fees. Even though optimistic rollups like Optimism and Arbitrum have reduced gas costs for their users, they use closed-source, centralized sequencers that have not increased network bandwidth in a meaningful way; less than 50 TPS for both.

There is also a tradeoff for the user, in that the relationship with the L1 changes based on the design choices of the L2. For instance, while the Ethereum L1 is decentralized, and reasonably censorship resistant, most L2s accomplish their high throughput by using a single transaction sequencer, or in the case of ZK-rollups, a prover, with ambitions to decentralize these components later on. This choice mostly eliminates the censorship resistance of the L2, which may be of consequence for time-sensitive transactions. It also raises a regulatory question which is: if these sequencers are run by a single entity (often a company), are they not just like an exchange, which should have to come in and register, or face legal consequences? Further, a recurring theme in EVM-focused podcasts is "how difficult and how soon can L2s like Arbitrum decentralize their sequencer?" The answer is most often a) very difficult, and b) probably not for several years. And in decentralizing the sequencers, how will that affect block confirmation times? The physical reality is that they will become slower, as they need to heap on communication stacks and consensus protocols. Many ethereum purists hold out that the rollups will deliver the answers, but, in describing an EVM-compatible system, with higher gas limit, including consensus, and marginally decentralized sequencer (validator) set, we are now describing the Binance Smart Chain. The advantages these systems have to offer versus a competing "alt" L1, like BSC, Avalanche, or solana begin to diminish. In short, while there are many benefits to L2s, users of L2s do not have the same assurances that can be found expected when interacting with the main chain directly, and steps to make these systems more censorship resistant will likely undermine their current benefits.

Composability Another pressing issue for the "modular" world is that, with each new L2, *composability*–the capability of smart contracts to share information and interact or "compose" with one another–is significantly impaired. While a main feature of rollups is that you should be able to bypass a sequencer to send

and settle transactions on the main chain, you now also inherit the properties of the L1, like higher transaction costs and slower block times. In the case of optimistic rollups, finality of transactions is contingent on a waiting period of days-weeks, which is required for to allow verifiers time to submit a *fraud proof*. Consequently, activities like bridging from an L1 to the L2 and back takes a very long time for the funds to be honored. This is not really acceptable UX; a user is 99% more likely to just hand carry the funds through an exchange that supports their L2 of choice. What it all means for composability is that interactions between the L1 and L2/rollup is complicated, and takes time.

Of greater significance, though, is how competing L2s compose with each other, or cross-rollup composability. While each rollup may be EVM-equivalent/compatible (they can talk to one another), they each have nuances related to security and finality, and therefore bridging between them you would expect will have an experience quite similar to bridging between alt-L1s, as you do today. The implication is that DeFi and dApps between different L2s cannot share liquidity, they are essentially islands in this respect.

In the case of assets stored on-chain, the use of L2s and disparate execution environments, their subsequent drawbacks related to composability causes a fragmentation of liquidity, because of their inability to interact with other L2s, or even the L1, in a timely manner. During periods of high volatility, it is possible to see departures in the prices of assets on an L2 versus the L1, since they are microcosms of the main chain, with their own dynamics, and limitations that the coins may be unavailable, or temporarily stranded.

If the modular EVM world requires dozens, hundreds, or thousands of L2s in order to scale, this becomes a real problem. Many islands arranged as archipelagos with convoluted paths to get between one another is not consistent with the vision of a dominant blockchain system that is capable of onboarding a billion users.

Given the limitations of L2s to compose with one another, their need for decentralized sequencers, and marginal improvements on block times, it really begs the question: why not do it all on L1?

Monolithic Design This title is a bit misleading because we don't really consider solana to be "monolithic" in its design, since, due to the account model and stateless programs, the system's parallel architecture is itself quite modular. Famous modular proponent Polynya could write a book about what monolithic versus modular means [to him], but I'll offer a simpler definition. Monolithic should just be taken to mean that the L1 is also the execution layer. Solana, BSC, Sui, and many others by that definition are monolithic as their security, consensus, state, and execution are all contained on a single layer.

The Trilemma

At the core of the debate is a concept introduced by Vitalik Buterin, ethereum's founder, the Blockchain Trilemma. The trilemma is depicted as a triangle with vertices labeled security, speed, and decentralization. The argument is that you can have two of these dimensions in a blockchain but not all three. A major flaw in its interpretation by others is in believing that these are discrete properties, but they aren't; these dimensions exist on a spectrum. Of course, a chain could be faster and less decentralized. The bitcoin blockchain is known for being incredibly slow, but with high security and decentralization. EVM-based proponents will say that they can have their cake and it too via rollups, since although the rollups undeniably trade decentralization for speed, the claim is that a rollup "inherits" both decentralization and security from the base layer. Meanwhile, solana detractors will make the argument that solana couldn't possibly be decentralized, if it is to be both fast and secure. Such analyses are fairly unsophisticated, since the art of blockchain design is in the *balance* that its architects strike between the trilemma's components.

Since solana ranks highly for both speed and security within the Trilemma framework, most arguments attempting to invalidate the network start with some muttering about how solana's validators are centralized because common users cannot run them. Yet, we see almost 2,000 validators on the network, run by everyone from grad school hobbyists, to exchanges, to institutional investors and funds. Solana's validator set is remarkably decentralized along industry-accepted metrics, such as the Nakamoto Coefficient (31, as of writing). Next, they say the hardware is too expensive, and must be run in a data center. What those people fail to see is that the people who run validators do so because they derive value from having equitable access to the blockchain's data and services, so "expensive" is quite subjective. They also fail to observe that well over 90% of other blockchain users a) do not run a full node and rely exclusively on RPC services, and b) the ones that do run them are overwhelmingly operated out of data centers, too. Next they talk about data center ASNs and geography. Here, solana also scores quite well, with generous geographic dispersion, and less reliance on Amazon Web Services (AWS) than even ethereum. How can it be that the network can have a good balance of all three?

Hardware is the Answer From solana's beginning, Anatoly hitched his cart to the trend called Moore's Law. The law basically says that the number of transistors in an integrated circuit doubles about every two years. The numbers need not be precise, there is just a general expectation that computer hardware become faster and more affordable over time. This is a reasonable expectation, and although transistor feature size decreases has slowed, a new wave of improvements in multicore processing have continued to all computation to get faster and cheaper. Single-threaded virtual machines like the EVM are not positioned to take advantage of the parallelism available from GPUs and multicore CPUs, but solana is.

Parallel Processing Solana relies on a leader (PoH generator) that is performing repeated, serial computation of the SHA256 hash function, while mixing in user transactions or messages. This is how valid transactions get frozen into the chain. When a series of hashes is complete, all of the message inputs are known, as well as the output(s). So, while the hash function must be executed serially to arrive at the final state, the process of verifying can be done in parallel, by breaking the steps of the series into chunks, combining the intermediate hashes and user messages, to confirm that the state changes are correct. Each verifier (non-leader validator) then merely submits a single transaction signature of the hash they believe is the correct one. This signature is their "vote", and each counts as a confirmation. Once 2/3 of the validator set has voted, then the block is finalized. This rather lengthy example is merely used to show how solana leverages the parallelism of multicore CPUs to reduce blocktimes, and thereby increase its TPS versus non-parallel designs.

If the hash verification process is accelerated by use of multiple processor cores, then it follows that more cores are better. And with more transactions to process, there is more work to be done by the verifiers. Solana will be able to increase its TPS by a) verifying more transactions across more cores and b) reducing block times by more quickly completing the work of verification. At the current time, network upgrades may unlock 200ms block times, with the lower bound of what is physically possible around 100ms (based on the size of the earth, and signal speeds through silicon and fiber networking cables–a problem parallel computing will not solve).

I/O Where there is an expectation for many blockchains to support consumer-grade computers to run nodes, solana does not have this limitation. Whereas with blockchains like bitcoin and ethereum you can probably run a node on a SATA3 SSD, solana nodes store the entire state of the blockchain on RAM, and some finite history on NVMe SSDs. Using RAM for state dramatically reduces I/O latency, and pretty much all accompanying state change operations. Additionally, read/write operations are significantly faster using RAM or NVMe, versus cheaper, commodity drives.

Networking Legacy blockchains typically don't require outrageous networking connections, since they propagate blocks using a gossip protocol. This method of broadcasting blocks is slow, and won't work for a network aiming to be an almost real-time system, or a "blockchain at Nasdaq speed." Verifying a block is contingent upon receiving it, so if it takes considerable time to disseminate information, then your TPS is limited by the delay it takes to receive blocks. Solana's design requires fast internet (speed), and also a big pipe (bandwidth). Solana started on gigabit connections (1 gigabit per second), and has since graduated 10 gbps. While a 10 gbps connection may be unavailable to all retail broadband customers, it is necessary to carry the vast amount of data the network generates in a timely manner. In designing a network that supports millions or billions of people transacting 24/7, support for the fastest connections

available is essential.

Security

Returning to the discussion of the trilemma. This section is a bit of a footnote related to security. A now notorious comment made by Kyle Samami, the prominent VC and hedge fund manager of Multicoin Capital, which has been very active in solana from early days, is that he values "speed over security." Based on clarifying comments he made on an Empire podcast in 2023, it seems he was talking more from an VC point of view, with the expectation that blockchain tecnology is inherently risky, and that he values teams that "move fast and break things." Nevertheless, this comment has caused moderate harm to the ecosystem, because onlookers took this to mean that Solana Labs developers don't care about security, and that they have somehow traded security of the network for faster transaction confirmation times. This is ridiculous, of course, but the misconception is prevalent enough that we should debunk it thoroughly.

Copies of the chain In starting a more nuanced evaluation of security, Anatoly often cites that "security is the number of copies of the blockchain." The primary determinant that can really affect whether you balance is what you think it is, is if the network has access to the blockchain state that says you have a balance. Here, bitcoin has the lead with the most copies of the blockchain, spread across its more than 10,000 nodes. Ethereum probably has #2 position in terms of copies of the ledger. Solana is very respectable here as well, with over 3000 copies of state between RPC nodes and validators.

Wallets Are the wallets any more or less secure on alt L1 chains, versus bitcoin or ethereum? Well, they all use BIP-39 seed phrases, so the keys themselves are not any less secure. Wallet UX and key management may vary, which can have implications for security. And smart-contract-based multisigs may have vulnerabilities or mistakes that can affect security. But all of this is wallet provider or smart contract-specific, and not a quality of the blockchain itself.

Cryptography As mentioned above, most wallets use keys based on elliptic curve cryptography (ECDSA) using libraries that are common to pretty much all blockchains. The public/private key arrangement is common across blockchains, use the same elliptic curves curves and signature scheme (ed25519), and for this reason they are all pretty much the same. Private keys are required for almost everything, so it is of great import, but it is generally not correct to say that one blockchain's keys have more or less "security" than another.

What about the cryptography used in critical functions like hashing, and consensus? Bitcoin and its children (litecoin, dogecoin, bitcoin cash, et cetera) uses SHA256 for their proof of work (PoW) algorithms. Solana also uses SHA256, as part of its novel proof of history (PoH) function. In either case, should a malicious actor have a means to compute these hashes faster than the miners

or validators, this *could* be a problem. In the case of SHA256 proof of work, they could find blocks more quickly, overtake the longest chain, and effectively decide the state of the blockchain. In solana's case, being able to compute hashes faster than your peers may allow you to perform a reversal attack (revert transactions), or a long-range attack, whereby the malicious actor recreates the history of the blockchain using compromised validator keys to forge the ledger. The reversal attack is limited to the period of time after a user message has been mixed into the PoH generator, and both of these attacks have mitigations by design, and needless to say, are incredibly farfetched. Such schemes may be possible with quantum computers, but today, SHA256 is considered quantum resistant, primarily because of the difficulty in factoring its hashes, and it is collision-resistant, or *extremely* unlikely to create the same output from different inputs.

Given the usage of common cryptography libraries and keypair systems across many blockchains, their reliance on SHA256 and ECDSA algorithms, in most cases, it is not proper to say that any blockchain has more "secure" cryptography than another.

3. Solana Whitepaper
4. Here's Why Quantum Computing Will Not Break Cryptocurrencies

Security via PoH As demonstrated above, SHA256 is a reliable algorithm that requires a user to perform the function completely in order to obtain its output, while being collision resistant (not subject to producing the same output from different inputs). SHA256 is the algorithm that underpins solana's conception of what its "state" is. As the proof of history generator advances, sequentially generating hashes as it ticks time, it also mixes in user state transitions (transactions) into the generator. Since the hash could not be generated without the incoming transition information / metadata, it effectively "locks" or binds that transaction to the blockchain at a given point in time. For the reasons above, it is effectively impossible to undo this operation, to forge an alternating history, or to reverse it. Consequently, solana's use of proof of history provides security that is comparable, if not superior, to that offered by other chains.

Weak Subjectivity To explain this subject, we first need a definition, and by far the most accessible one I have seen is provided by Edmund Edgar on StackExchange:

Subjectivity is the idea that there can be multiple potentially correct-looking versions of a particular blockchain ledger, and it's OK if a computer can't automatically choose between them. This is opposed to a blockchain where the correct ledger is knowable by looking at something objective like which chain has the most proof-of-work, or which has the most proof-of-work while conforming to a pre-determined set of validity rules.

Weak subjectivity is the idea that subjectivity is unacceptable for short timespans,

but acceptable for long timespans. Specifically, it is proposed that if you have a node that is continuously online, you will be able to automatically determine the correct ledger, but if your node goes offline for many months, you may need to get information from somebody else (eg a friend with a node that was online, or a block explorer) to determine the correct ledger.

Bitcoin is an example of a blockchain without weak subjectivity. If you have all of the blocks, you can compute the current state of the ledger from the start. In fact some bitcoin wallet like Specter actually allow the user to do this! This is great for a ledger that is concise enough to fit onto a single hard drive, such as bitcoin. However, more data-rich blockchains like ethereum and solana create volumes of data, and over time, computing the state from the historical ledger will be extremely difficult, if not impossible. The thrust of weak subjectivity is that this is OK, presuming there are honest nodes onlines that you can trust to have the correct version of the current state. With weak subjectivity, it is essential that the current state is the correct one, but that how the chain arrived at that state is less important, provided that the rules of consensus were followed in order to get there. The takeaway here is that the bitcoin blockchain has an objectivity, whereas modern proof of stake networks, including both solana *and* ethereum, rely on weak subjectivity.

5. What is Weak Subjectivity
6. Proof of Stake: How I Learned to Love Weak Subjectivity

Security vs. Liveness In most circumstances, solana values security (state "correctness") over liveness (system availability). While it is true that blocks are confirmed optimistically before they are finalized, finality is still achieved incredibly quickly, and is without comparison to an EVM-based L2. In the situations that really matter, such as large-scale partitions of the chain, solana will stop processing non-vote transactions (affecting liveness), to prevent confirming invalid transactions (a security risk). Such safeguards have indeed caused the blockchain to halt, but at no time have coins or funds been "lost" because of some piece of state becoming corrupted, or invalidated. Solana ultimately chooses security over liveness, to ensure users are protected should consensus diverge.

Reorgs For the reasons enumerated above, ad-nauseum, Solana is incredibly robust against chain reorganization, or reorgs. Once a user message is mixed into the PoH generator, the message is essentially sealed in time, and after 2/3 of the validators have voted (verified) that the leader's hash is valid, then the we have achieved finality. Voter delinquency starts at the end of the block time (<= 400ms), which is means that a quorum of verifiers must have responded within the deadline. And since the votes are mixed into the PoH generator's stream, we know that a validator (verifier) voted before the deadline. As a result, Solana provides reliable assurances that the transaction is valid in an exceptionally short amount of time–orders of magnitudes faster than most other chains. Reorgs are confined to the place in time up to when the user message is inserted into the

PoH generator (for reversal or double-spend attacks), and the long-range attack (forging the ledger to alter history) requires substantial resources plus a failure in multiple, high value validator keys to be possible. Both of these flavors of reorg are, again, highly implausible without being able to compute the SHA256 hashes required 100x, 1000x or more faster than the current validator set.

On the subject of blockchain partitions (a disagreement about consensus), Solana chooses consistency over availability according to the CAP Theorem [1]. "Consistency means that all clients see the same data at the same time, no matter which node they connect to. ... Availability means that any client making a request for data gets a response, even if one or more nodes are down." Solana chooses consistency (to the limit of some human-scale timeouts) over availability, because it has a concept of how much time has passed. In order to restore system availability, it can then systematically unstake delinquent validators until the 2/3 voting supermajority is restored. This enables the chain (or its users) to identify and abandon invalid partitions, and reduce the risk of substantial reorg events.

7. CAP Theorem

Summary

We've discussed everything from Solana's proof of history, and how the chain works, to its differentiating features, and design choices. We learned about some vital network upgrades (isolated fee markets, stake-weighted QoS, and QUIC) that dramatically improve the usability of the chain. Finally, we compared Solana's flat blockchain design, with a rollup-centric model, and did considerable work to dispel the numerous misconceptions people have related to Solana's limitations, and security.