Logos Network Primer

02.05.2018

Michael Zochowski
Promethean Labs
New York, NY

Version 1.0

Contents

Introducing Logos

Logos is a decentralized, cryptographic transaction network that solves the scalability problem faced by legacy cryptocurrencies. Logos fundamentally redefines the concept of a cryptocurrency through a new architecture and consensus algorithm, which allows it to shed the constraints imposed by the blockchain model. Logos empowers users on a global scale to send rapid, secure, cheap, and smart transactions of all sizes, a prospect that can revolutionize both existing and future payment applications. Logos has a singular focus on providing the optimal transaction infrastructure, which ensures that it is unfettered by the overhead that is inherent in more complex networks.

The Problem

The future of money is digital, as seen in the huge growth of peer-to-peer payments, the massive potential of Internet of Things networks, and the revolutionary leaps in decentralized technology. In the coming decades, cash and other physical forms of payment will disappear as the world fully transitions to electronic transfers. Much like the introduction of the internet fundamentally changed how people interact, so too will the digitization of money redefine the basic economic relationships between people, devices, and companies.

However, existing solutions have thus far proven unable to fulfill this looming demand. While almost everyone in the world has access to the internet, billions of people globally remain unbanked and without access to basic financial services such as a bank account. Centralized solutions remain slow, costly, and reliant on a trust model that frequently breaks down. FinTech solutions within the existing banking infrastructure have had some success but remain fettered by bureaucratic constraints that result in high fees, slow confirmation times, fraud, and lack of control over one's own funds.

Starting with Bitcoin, cryptocurrencies recognized this issue and provided trustless (i.e. security without requiring trust), decentralized, and open transaction networks that anyone, anywhere can use. Without the constraints of the existing financial system, the potential applications for cryptocurrencies soon grew beyond simple payments to a massively diverse range of use cases. Despite this huge promise, existing cryptocurrencies such as Bitcoin and Ethereum have rapidly hit their capacity[1] limits, resulting in slow and expensive transactions that are inferior to centralized alternatives.

The problem of scalability is existential for cryptocurrencies. Almost all compelling applications of distributed ledger technology require massive capacity to be successful, which necessarily mandates a global scale. While addressing the scalability issue has been at the forefront of crypto development for years, existing networks have failed to produce practical solutions.

[1] In the context of cryptocurrencies, capacity and throughput typically refer to how many transactions per second (tps) the network can process.

Many scaling solutions have been proposed, but with few successes. Mainstream solutions like Lightning or Casper are restricted by the constraints of legacy crypto paradigms and models. More unconventional approaches like directed acyclic graphs (DAGs) often lack theoretical guarantees that are necessary for global adoption. These networks have consequently struggled with decentralization and practical functionality. Other new crypto networks that try to solve scaling are hampered by a "maximalist" mentality that seeks to provide both unbounded computation and limitless scaling. Since these goals are inherently opposed, they will likely achieve neither.

If crypto is to move beyond early adopters and niche use cases, it must fundamentally change its structure. Logos does just that.

The Solution

Logos achieves scalability through three major innovative designs. We intuitively explain each of these components in this primer.

First, Logos introduces a new consensus model called *Axios*. Instead of mining, Axios is based on delegated proof-of-stake validator selection models, which ensure safety while allowing rapid transaction validation. The underlying consensus algorithm is built on Practical Byzantine Fault Tolerance (PBFT), the gold standard for achieving efficient agreement that is theoretically secure.

Second, Logos uses a completely new hybrid base structure that moves beyond blockchain. At the heart of this structure is the *chain mesh*, a set of accounts each with an independently ordered transaction chain that jointly form a directed acyclic graph (DAG). The key insight of this structure is the ability to process independent transactions in parallel, rather than the serial model of blockchain. A global blockchain called the *archive* serves as a checkpoint system that ensures global synchronicity and safety. This means that Logos has the throughput of a DAG and the proven safety of a blockchain.

Third, Logos implements state sharding via *Polis*. Any network run by a single validator group is inherently constrained by hardware limits since the majority of validators need to see every transaction. Sharding bypasses these limits by dividing the network into several groups, each of which validates its own transactions. This adds a second dimension of parallelism to transaction processing, which allows Logos to scale with the number of nodes while preserving overall security.

This architecture will initially support thousands of transactions per second without sharding and hundreds of thousands soon thereafter with sharding. Underlying all these design decisions is a philosophy emphasizing theoretical rigor, opportunistic adoption, and practicality. These tenets continue to guide development of new features and the broader ecosystem.

Authors Note

This primer is intended as a high-level overview of Logos and its functionality. In furtherance of this goal, we make several simplifications and abstractions. We direct readers to the Logos white paper for a more detailed theoretical treatment.

Overview of Existing Cryptocurrencies

Before turning to Logos, we first summarize existing cryptocurrency technology and the history of the blockchain architecture to give context to Logos's design and shed light on its innovations.

What are cryptocurrencies?

A cryptocurrency can be distilled down to a distributed ledger that solves consensus in the presence of Byzantine faults.

We can unpack this definition by explaining each of these individual terms:

- *Distributed ledger*: a database that is stored on multiple computers. Since it is not stored solely by any single party, the database is decentralized. This database can theoretically record anything, but in the case of cryptocurrencies, it usually tracks a set of accounts with balances.
- *Consensus*: the process by which network nodes come to agreement on an update to the database. This includes verifying that the update is valid and ensuring that nodes are synchronized.
- *Byzantine faults*: when a node behaves in an arbitrary and possibly malicious way. This is in contrast with a fail-stop fault, where a bad node simply shuts down.[2]

So the definition can be expressed as: A cryptocurrency is a decentralized database that tracks user account balances (or other data) and is secure even if many users on the network behave maliciously. The difficulty and the nuance in designing a cryptocurrency comes from capping the number of users that can be malicious while minimizing the time it takes for the aggregate system to come to agreement.

With this definition in mind, some key properties of (public) cryptocurrencies are:

- *Immutability*: once an update is made to the database (e.g. Alice sends tokens to Bob in exchange for coffee), that transaction will never be changed.
- *Trustlessness*: any two users can transact with each other without trusting each other. Instead, they can mathematically and computationally trust the system.
- *Openness*: anyone, anywhere should be able to access and use the network without the need for verification or trust.

These properties are the major advantage that cryptocurrencies have over centralized alternatives, such as banks or credit card networks. Traditional financial networks solve the issue of trust by introducing a trusted intermediary between unknown parties, such as a bank. However, this requires that both parties trust the bank, and also bars participation by a significant portion of the population that the bank does not trust (for example, they don't have

[2] For a simplified example of how this could be problematic, please see the Two General's Problem.

a long enough credit history). Furthermore, the transactions facilitated by that bank can potentially be reversed or changed by that bank, whether intentionally or unintentionally. Frequently, this trust relationship breaks down, and people had few alternatives before cryptocurrencies.

A practically useful cryptocurrency also needs to be faster, cheaper, and more efficient than centralized networks. For example, credit card processors often charge merchants 3% of each purchase and banks charge $25 for wire transfers that take hours to be processed. While cryptocurrencies have a significant advantage since they lack the bureaucracy and overhead of traditional financial intermediaries, in practice they have suffered from limited capacity, long confirmation times, and expensive fees. Logos solves these issues.

Cryptography

Cryptography is the set of techniques that enable secure communication in the presence of adversaries. As the name suggests, cryptography is at the heart of any cryptocurrency.

While cryptography is a complex and deep field, there are two specific cryptographic techniques that are of particular relevance. First are hash functions, which map data of arbitrary size to data of fixed size. The key property is that it is easy to verify if a particular input maps to a given hash value, but given only the hash value it is extremely difficult (practically impossible) to reconstruct the input value. That is, for hash function H and input x, it is easy to compute $H(x)$ if you know x, but it is infeasible to find x if you only know $H(x)$. Hash functions are an example of one-way functions, which have this property.

Hash functions are useful since they effectively uniquely summarize data in a fixed size “hash”, without giving any information about the original data. This means that you can store the hash of sensitive or large data and, when presented with new data, can easily see if it matches the original data by comparing the hash of the new data to your stored hash. Logos uses the Blake2b hash function.

Second is public key cryptography, which is a type of cryptographic system that allows any two people to securely communicate without ever exchanging private information. Such a system involves two encryption keys for each user: a public key that is widely known, and a private key that is kept secret. Alice can securely send a file to Bob by encrypting it with Bob’s public key, and the resulting ciphertext can only be decoded by Bob’s private key. Bob can also digitally sign a message, which means that, given the message, signature, and Bob’s public key, any other user can verify that the message was sent by Bob. An important extension is the multisignature, which is a way for users to verify if a message was approved by a specific group of users (as opposed to just one user for basic signatures).

Public key cryptography relies on one-way functions that make decrypting or signing messages associated with a particular public key easy for someone who knows the corresponding private

key but virtually impossible for someone who doesn't. Logos uses public key cryptography based on a certain type of one-way functions involving elliptic curves (EC). Logos also uses a signature scheme called Schnorr signatures, which allow you to store multisignatures in a fixed amount of data regardless of how many users are included in the signature. The combined scheme is called EC Schnorr signatures.

Byzantine Fault Tolerance

Byzantine fault tolerance (BFT) is a property of a network that is able to continue to function despite nodes that fail in Byzantine (i.e. possibly malicious) ways. This is a key property of distributed systems, particularly cryptocurrencies, that are open to users that may not be trustworthy.

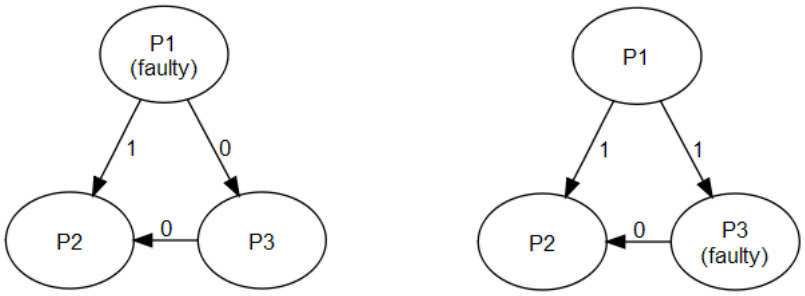
Specifically, we desire that a cryptocurrency maintain *safety* and *liveness* even if some users are malicious. Safety means that system integrity is maintained across good nodes: no one's balances can be moved without their permission and no previous transactions can be reversed. Liveness means that the system will continue to process new transactions and bad nodes cannot halt the continued operation of the system for an indefinite amount of time.

Bitcoin was the first (probabilistically) Byzantine fault tolerant payment system, but BFT is actually a well-studied problem in distributed systems with decades of research exploring various solutions. The issue was originally stated as the Byzantine Generals Problem. This problem asks how a group of generals that includes several traitors should come to agreement (consensus) on whether to attack or retreat. The problem is solved if all loyal generals agree on the same plan of action (Byzantine agreement).

It turns out there are many solutions to the Byzantine Generals Problem. However, they all require a similar set of assumptions. Most importantly, if there are f faulty nodes, there must be at least $3f + 1$ total nodes, meaning that fewer than 1/3 of nodes are Byzantine.[3]

To illustrate this bound, consider the following two scenarios in a 3 node system.[4] In both cases, node *P1* is the leader that proposes either option 0 or option 1. The non-leader nodes then try to reach consensus on the correct option by reporting to each other what they heard from the leader.

[3] This assumes partial synchronicity between nodes. There are solutions that allow for only $2f + 1$ nodes, but they require full synchronicity and digital signatures.
[4] Source: http://marknelson.us/2007/07/23/byzantine/



In the first scenario, *P1* is faulty and sends *P2* and *P3* separate messages. In the second scenario, *P3* is faulty and sends *P2* a 0 even though it received a 1 from *P1*. In both scenarios, *P2* receives a 1 from *P1* and a 0 from *P3*, which means it cannot distinguish between the two scenarios. This means *P2* cannot determine who is the faulty node, and therefore cannot reach consensus.

The same basic principles of the Byzantine Generals Problem (BGP) can be translated to the cryptocurrency context. A single node proposes a change to the database (e.g. Alice sends tokens to Bob), and then all the nodes need to reach consensus on whether or not to commit that change. Ideally, if the change is valid, the nodes should commit it, but the most important thing is all honest nodes decide on the same action.

While there are many ways to reach consensus, many of them are inefficient and require many rounds of messages between nodes. Logos's consensus scheme is based on Practical Byzantine Fault Tolerance (PBFT), an efficient consensus algorithm that requires only 3 rounds and realistic assumptions. It is well studied and offers strong security guarantees. We explain Logos's version of PBFT in more detail later in the primer.

A Modular Model of Cryptocurrency

Given this context, we can characterize base cryptocurrency protocols (such as Bitcoin, Ethereum, or Logos) as architectures with several modular components. Each individual cryptocurrency is defined by the design choices for each component.

These components include:

1. *Base structure*: the fundamental unit tracked by the cryptocurrency database. This typically is some means of tracking account balances but can also include more complex data.
2. *Validator selection*: how the network decides which nodes will validate transactions.
3. *Validator consensus*: given the set of validator nodes, how those nodes reach agreement on changes to the database.

4. *Request processing*: how the network processes transactions. In most legacy systems, this takes the form of a blockchain, where transactions are batched together into blocks and committed together.
5. *Cryptography*: the choice of hash function, signature scheme, and other cryptographic protocols.

There are other components, such as the choice of randomization algorithm or smart contract execution, but those can generally be easily substituted without changing the fundamental design paradigm of the network. The ones enumerated are basic choices that are required for any cryptocurrency protocol. As explained in the following sections, most current cryptocurrencies conflate two or more of these components. This typically leads to suboptimal structure due to the presence of an inefficient component. Logos is able to achieve far better performance than legacy networks by breaking down the crypto model and optimizing at the component level.

The Legacy Crypto Paradigm

How Proof-of-Work Blockchains Work

Starting with Bitcoin, almost all cryptocurrencies have followed a proof-of-work blockchain architecture. Before explaining why this legacy design came about and its limitations, we first explore how it functions. We use Bitcoin (BTC) as a canonical example, but this explanation applies generally to most blockchains.

Bitcoin's base database tracks so-called unspent transaction outputs (UTXOs). As the name suggests, these are token balances that result from previous transactions that can be spent by the holder of the corresponding account. A given account (basically a public key – private key pair) can have many UTXOs, and each transaction takes as input one or more UTXOs and outputs one or more UTXOs. For example, if Alice has 2 UTXOs, one with 4 BTC and one with 6 BTC, and wants to send 8 BTC to Bob, then she creates a transaction that takes in both of her UTXOs as input and outputs a UTXO with 8 BTC to Bob and a UTXO with 2 BTC that goes back to her.

Bitcoin processes transactions via a proof-of-work (mining) blockchain. In this architecture, transactions are batched together into blocks, which have limited capacity. The subset of pending transactions that should go in each block is completely subjective, and two different miners can propose two totally different blocks of transactions. Each block must reference the previously approved block, so the blocks end up forming a chain (hence *blockchain*).

In proof-of-work, the set of uncoordinated miners race to solve a computationally difficult problem that is set by the network. This involves finding a hash of the block (which is chosen by the miner and includes a random value adjusted by the miner) that is less than a certain value. As discussed previously, this is easy to verify but hard to solve since it is practically impossible

to invert a hash function. Miners continue to change the random value until one finds a solution. The first miner that solves this problem publishes proof to the other miners, who verify its validity, add that miner's block to the blockchain, and then move on to mining the next block. The successful miner is rewarded with newly minted coins plus transaction fees from the transactions in its block. The difficulty of the mining problem changes periodically based approximately on the total computational power mining on the network to ensure the average time between blocks remains constant.

Since the block contents are subjective, a malicious actor could attempt a *double spend*, which involves sending the same UTXO to two different accounts. For example, Eve could purchase a coffee using BTC in one UTXO and also send the same BTC back to herself, in the hope of keeping both the coffee and the BTC. Such a scenario can produce a fork in the blockchain, where there are two possible but conflicting chains. These forks are resolved by accepting the longest chain as the correct one. As long as 51% of the network consists of honest miners, the honest miners should probabilistically add the next block quicker than the attackers, thereby creating the longest chain.

An attacker that wishes to successfully double spend must therefore produce the longest chain. The probability of doing so decreases exponentially with each added block and the computing power of honest miners. To guarantee probabilistic success, an attacker requires 51% of mining power. Note that otherwise spending someone else's funds is impossible since transactions require the digital signature of the spender.

While this is a relatively simple architecture, it has many drawbacks and is largely obsolete, as we will explain subsequently.

Why were these design choices made?

In order to understand these design choices, we must contextualize them in the origin of cryptocurrencies. At the genesis of Bitcoin, very few people assigned the coins any value. In order to bootstrap the system to a state with value, Bitcoin had to overcome several problems that are not faced by cryptocurrencies today.

A key insight of Bitcoin is the use of computer power to prevent *Sybil attacks*, which involve an attacker setting up many accounts to game the network. This is very difficult to prevent in most network validator selection and consensus schemes. For example, if the network decided on whether to commit a transaction using a simple vote, an attacker could create many fake profiles to gain additional votes.

Note that that this problem must be dealt with before the Byzantine Generals Problem can be solved. BGP assumes the set of validators is predefined, but in a cryptocurrency we want to allow validators to enter and exit the network dynamically. Furthermore, we want to make sure that validators have a barrier to entry or something at stake to prevent a single person from undermining the network at very little cost.

Bitcoin neatly solved this via proof-of-work, which ties together validator selection and consensus. Any attacker's computing power would be spread among his various identities, and he would have no additional power from creating more accounts. Any attacker therefore has to make an economic investment in computing power, which deters attacks. Proof-of-work has several additional desirable qualities, such as allowing for the set of validators to change over time and not requiring sophisticated coordination between miners.

Blockchain is a natural consequence of proof-of-work. The chain structure is key to ordering transactions and is required for any cryptocurrency in some form. Since the miners are racing against each other to produce the next block, they lack coordination and must wait to receive the proof of a successfully mined block over the network. This necessitates a single chain to synchronize the state between nodes. This process takes time, and it is important that most nodes see the most recent block in order to know which slot in the chain they should be mining. If they are mining the wrong slot in the chain, the network computing power is effectively reduced. As a result, there is a minimum time required between updates to the database if the system is to remain safe. Due to this minimum time, it makes sense to batch transactions together rather than process them one-by-one.

Problems with the legacy architecture

While the legacy architecture makes sense in the context of its genesis, it has many drawbacks. Now that cryptocurrency value has been bootstrapped to hundreds of billions of dollars, the problems that mandated this structure can be solved in better ways.

By the same logic that mandates a minimum time between blocks, there is an additional limit on the size of each block. If blocks are too large, they cannot propagate through the network fast enough. The combined limits on block time and block size place an inherent limit on network capacity. While this architecture can likely achieve better transactions per second (tps) throughput than the low double digit transactions per second provided by Bitcoin and Ethereum, it likely cannot exceed the low hundreds of transactions per second.

An additional issue is the massive energy consumption required by proof-of-work networks. It is estimated that Bitcoin alone exceeds the energy consumption of Ireland. While there is social benefit to cryptocurrencies that outweighs the deadweight loss, there are far more efficient consensus mechanisms if validators can more intelligently coordinate.

Forks also can present a large threat to blockchains. If a single transaction is successfully changed, it invalidates its entire block as well as every subsequent block. Since the contents of each block are largely subjective, there is no deterministically correct history. As a result, forks are complex and difficult to deal with under the blockchain paradigm, particularly in conjunction with more sophisticated choices of validator selection and consensus algorithms.

A related issue is one of finality. In a proof-of-work blockchain system, there is always a chance that a double spend or chain reorganization can result in the longest chain and therefore

become the accepted reality. Therefore, such blockchains only provide *probabilistic* safety. This results in longer confirmation times, as someone accepting payment must wait for several blocks to be added, on top of the one that includes the desired transaction, to be confident that the transaction will not be reversed.

The Logos Architecture

Logos abandons the legacy cryptocurrency conventions that have proven to be inefficient and instead builds its architecture from first principles around achieving optimal capacity. At a high level, there are three primary sources of scaling in the Logos system. First, a unique, non-blockchain base structure called the *chain mesh* enables parallel processing of transactions and more efficient data storage. Second, a new consensus scheme called *Axios* based on delegated proof-of-stake and PBFT optimizes confirmation times while guaranteeing safety. Third, a state sharding mechanism called *Polis* allows only a subset of validators to approve each transaction, resulting in a second dimension of parallel processing. The remainder of this section explores each of these designs in detail.

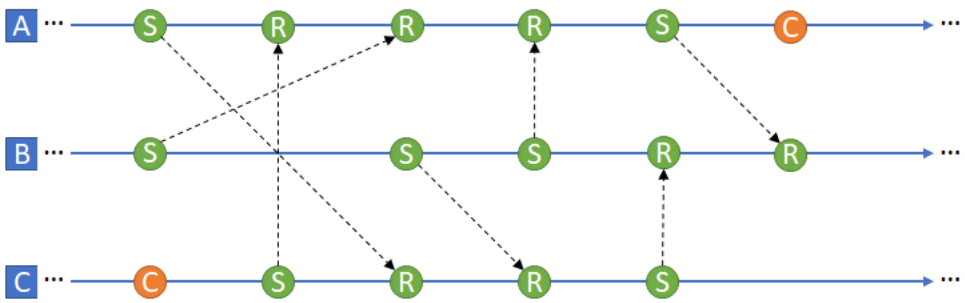
Base Structure: Chain Mesh + Archive

Logos is a network of nodes that track a set of accounts, each with a balance of the native network token. Unlike the UTXO model used by most cryptocurrencies that tracks each individual transaction separately, this model tracks the running balance for each account. This is similar to a traditional bank database, which records your balance at the beginning of the month and the transactions for that month. Transactions that happened two years ago are largely irrelevant as long as you know the current balance.

Each account is identified by a public key, and the corresponding private key can be used to approve transactions (called requests) from that account. Instead of batching together requests into blocks and recording them in a blockchain, Logos nodes record individual chains of requests for each account (see the below figure for a visual representation). Each request contains a reference to the previous request in that account's chain, so each account uniquely defines their own chain.

Since transactions move money between multiple accounts (every send has at least one recipient), transactions link together different account chains. Send requests and receive requests are separate, which allows a receiving account to choose when to add an incoming transaction to its chain.[5] If one views each request as a vertex and each reference to a request as an edge (each request refers to the previous request in the account chain, and each receive request refers to the corresponding send on another account chain), then the resulting structure is what is called a directed acyclic graph (DAG). This means that there is at least one total ordering of all transactions, which is a key security property. We call this structure the *chain mesh*.

[5] Separating sends and receives ensures that each account has total control over its chain. Among other benefits, this is important to prevent spam attacks on an individual account. It guarantees that accounts are independent of other accounts, at least locally. If they were not separated, a disputed incoming send would effectively freeze the recipient account, since it would not know which is the most recent request in its chain.



The chain mesh: each account chain is comprised of an account's requests. Transactions link together different chains as one account sends tokens to another account, forming a DAG.
S = send; R = receive; C = change representative (explained later)

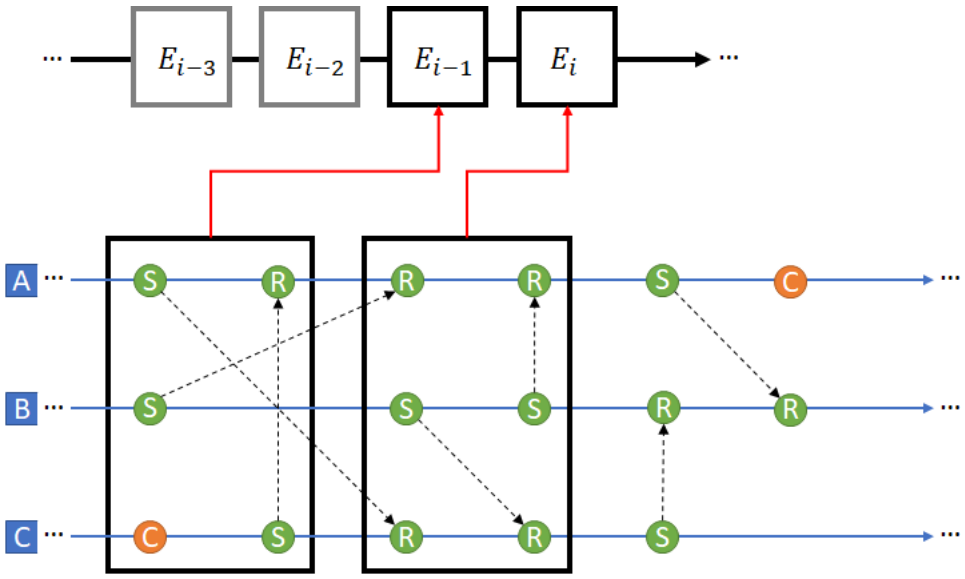
The chain mesh structure has many advantages over blockchain, including:

1. *Parallel processing*: Independent requests are processed in parallel, while blockchain processes all requests serially. This means that if there is a disputed transaction for a single account, all other accounts continue to function normally.
2. *Faster processing*: Since all chains are independently determined by their account, validators can process transactions as fast as their hardware allows. This is a huge improvement over blockchain, which has strict limits on the combination of block time and block size.
3. *No subjective forks*: Since each account defines the order of its request chain, forks can only occur if an account intentionally attempts to double spend. This means the base structure is entirely objective. On the other hand, blockchain blocks are entirely subjective, and forks can happen for many reasons.
4. *Forks have limited impact and are easy to resolve*: Since forks can only happen intentionally, they are simple to resolve. For example, the offending account can simply be frozen for a period of time, and the rest of the network will continue to function normally. Furthermore, since each account chain is independent, a fork only affects the offending account and the recipient accounts. Resolving a fork only impacts those accounts, or just the sender if the recipients did not yet add the transaction to their chain. Conversely, a fork in blockchain impacts all transactions that occurred after the fork in the chain, and it can easily render the system practically unusable while the fork is being resolved.
5. *Zero reward variance*: Since blockchains batch many transactions together and typically give a single validator a reward for proposing the block, validator rewards have high variance. For example, a small miner may go for days without solving a Bitcoin block. This can create perverse incentives, and miners demand more reward in exchange for accepting volatility. By unbundling transactions (meaning there are many more independent validations), this variance drops significantly. In fact, Logos validator rewards have zero variance (see consensus section for more details).

In addition to the base chain mesh structure, Logos records all transactions that happened in a set period of time in a blockchain called the *archive*. The archive also contains results of votes and other system events. The period of time covered by a single archive block is called an

epoch. Each epoch is very long (on the order of 12 hours), so the limits imposed by the blockchain structure do not interfere with network speed. Rather than handling ongoing transactions (which is done by the chain mesh), the archive instead serves as a periodic checkpoint that summarizes what has changed in the system. The archive confers many benefits, including forcing the network nodes to synchronize, helping new nodes bootstrap to the present state, and enabling a decentralized governance system.

This hybrid DAG/blockchain architecture captures the best of both structures. DAGs like the chain mesh are best for rapid, parallel transaction processing and have fewer forking problems than blockchain, while blockchains like the archive are good at recording universal state changes and governance modifications at a lower frequency.



The archive: each epoch block records the transactions that occurred on the chain mesh during that epoch along with governance votes. Since epochs are long, the archive's blockchain structure does not degrade system performance.

Consensus: Axios

A consensus scheme consists of two components: first, a mechanism for selecting a set of validators; and second, a mechanism for those validators to come to agreement on changes to the account database. Logos's consensus scheme is called Axios. Axios selects validators via a delegated proof-of-stake mechanism, and those validators reach consensus using a PBFT-based algorithm.

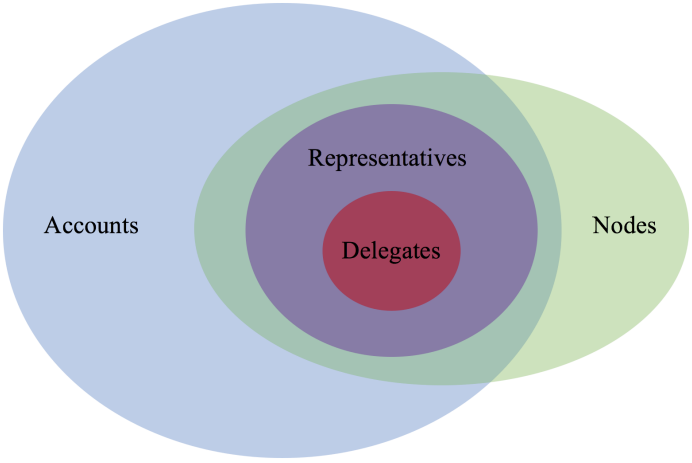
Breaking down the jargon, proof-of-stake is an alternative to proof-of-work (mining) where power in the system is derived from buying and holding network tokens (a stake) rather than from computer power. As noted previously, proof-of-work was necessary to bootstrap the

cryptocurrencies from zero value. Proof-of-stake would not have been a viable hurdle to becoming a validator since there was no economic cost to buying something with zero value.

However, now that cryptocurrencies have substantial value, using proof-of-stake to establish validator identity is far better than proof-of-work for two primary reasons. First, it is significantly more energy efficient since validators do not need to mine. Second, it requires coordination between validators, which allows transactions to be processed much faster. There typically is one major drawback in standard implementations of proof-of-stake: since a majority of validators need to see each transaction (a basic requirement for BFT), allowing anyone who stakes tokens to become a validator will result in a large number of validators, which slows the system down. This is effectively anti-scaling in the number of validators, while proof-of-work has constant scaling (mining difficulty is adjusted to ensure average block time is constant).

To solve this issue, Logos has a delegated validation system with several levels. The idea is that stakeholders at each level elect a smaller number of agents to represent them, which minimizes the required number of messages to run the system.

The bottom level is the set of accounts, the majority of which are typically offline and not paying attention to the ongoing validation. Accounts designate other accounts that they expect to be online and trust as their *representatives*, which can be changed at any time. These representatives vote on governance proposals and elect the validators, called *delegates*, that run the actual consensus algorithm to process transactions. Since consensus is an expensive process that requires many messages, the number of delegates is fixed at a small number (typically less than 100). This means that transaction processing time has constant scaling in the number of nodes.

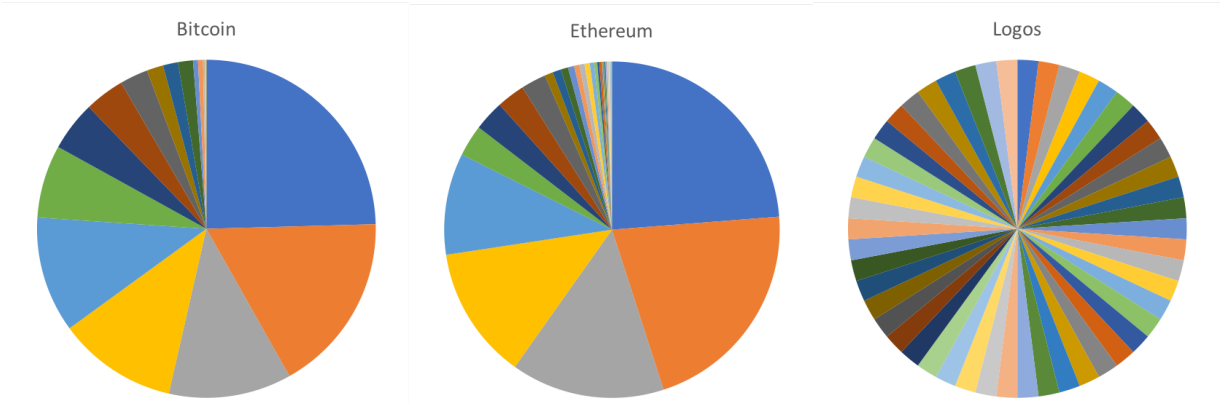


Since the entire network benefits from maximizing transaction throughput, both directly in the form of transaction fees and indirectly in increased network utility, there is a natural economic incentive for the representatives to elect delegates with the best hardware. This inherent specialization further improves potential network capacity. Elections are held every epoch, which ensures that the network has ample opportunity to find the best delegates.

While at first glance this delegate system seems to potentially concentrate power in the hands of only a few stakeholders, thereby potentially threatening the desired safety and liveness properties, it actually is robust to bad delegates.

These delegates are accountable to the representatives, and bad delegates (whether intentional or unintentional) can be voted out of their positions or recalled. Each transaction approved by the delegates is verified by the other network nodes (which is a much faster process than consensus itself), so an invalid or conflicting transaction will never be approved by the network overall. The recall system also ensures that delegates can only halt the network for a bounded period of time. So both safety and liveness guarantees are preserved despite the concentration of validation power in the delegates.

Additionally, the delegate system actually results in a far less centralized validation process than most other cryptocurrencies, such as Bitcoin or Ethereum. As shown in the graphs below, large mining pools hold a disproportionate amount of power in both cryptocurrencies, while a delegate system spreads power more uniformly over delegates.



Share of total validator power of top validators for Bitcoin, Ethereum, and Logos. [6]

Another major guarantee of validator behavior are slashing conditions, which apply to representatives and delegates alike. A slashing condition is when a certain account's balance can be penalized if they have performed a certain bad action. The network defines the set of these "bad" conditions that can only be violated intentionally and maliciously. For instance, only a malicious delegate would approve two conflicting transactions to be committed. When these conditions are violated, other nodes in the network can submit a slashing request, which results in the violator losing its entire stake. To ensure the threat of being slashed is a deterrent, a validator's stake is locked up in the form of a deposit from the time he becomes a

[6] Under an equally weighted delegate scheme. Weighting by number of votes will result in a less uniform distribution. Sources:
https://blockchain.info/pools
https://www.etherchain.org/charts/topMiners

validator until a long time after he reverts to a normal account. While his stake is locked, the validator cannot spend or move his balance, but it can be confiscated if he behaves maliciously.

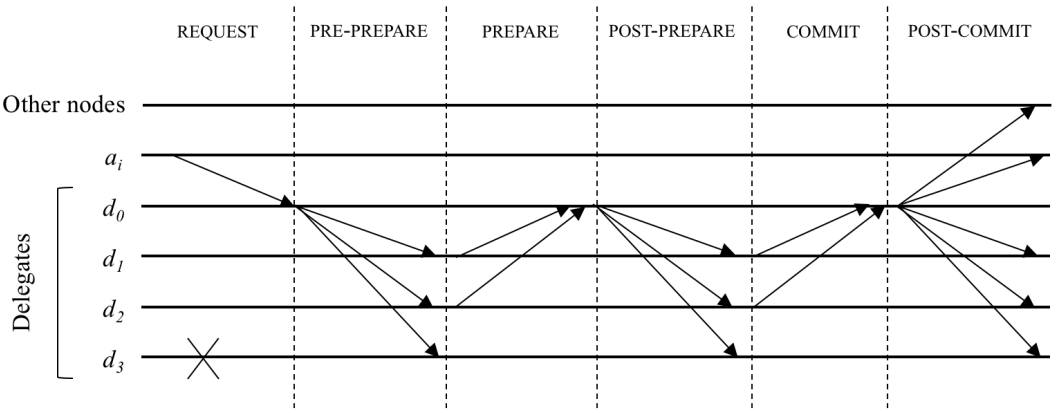
Once delegates are determined for a given epoch, they can start validating transactions. This involves using Logos's consensus algorithm, which is based on the Practical Byzantine Fault Tolerance (PBFT) algorithm described previously. Logos makes several improvements on standard PBFT, including a special type of digital signature that reduces data size and optimized communication trees to reduce message transmission overhead.

In Logos, an account sends a transaction to a single delegate of its choice. That leader delegate then initiates PBFT, which has three phases to reach consensus. PBFT and its derivatives require that more than $^2/_3$ of nodes are honest, so each step attempts to get more than $^2/_3$ of delegates to agree to advance to the next phase. Note that this is the optimal bound for non-synchronous consensus.

The phases are:

1. *Pre-prepare*: the leader confirms that the transaction is valid and sends a signed message to the other delegates to inform them of the transaction.
2. *Prepare*: the other delegates all check to see if the transaction is valid and send a signed "prepare" message to the leader, provided that they have not seen any conflicting transactions. The leader waits until it has received "prepare" messages from more than $^2/_3$ of delegates (including itself), at which point is sends out a "post-prepare" message as proof to the other delegates.
3. *Commit*: upon receiving a valid "post-prepare" message for the transaction, the other delegates send a signed "commit" message to the leader. The leader again waits until it has received "commit" messages from more than $^2/_3$ of delegates (including itself), at which point it sends out a "post-commit" message to the broad network as proof. When nodes see a valid "post-commit" message, they update their databases with the approved transaction.

Logos's consensus algorithm is summarized in the chart below.



Various message types sent during Logos's Axios consensus algorithm. Account a_i sends a request to delegate d_0, who acts as leader. The request is validated iteratively according to the algorithm, and consensus is achieved despite the failure of delegate d_3. After the request is committed, it is disseminated to all network nodes.

Once a transaction is committed by the delegates and accepted by the network, it is economically finalized. This means that it cannot be reversed unless a substantial portion of the validators lose their stakes from slashing. After the transaction has propagated to the entire network, it can never be reversed. By locking in the validator stakes for an extended period of time, we ensure that the network has sufficient time to reach this finality.

Logos chose PBFT as the base consensus algorithm since its proven security has been vetted extensively and recognized widely for its efficiency. While other consensus algorithms assume only more than $^1/_2$ of delegates are honest, they require more phases and stronger synchronicity assumptions. PBFT ensures strong safety and liveness guarantees while offering speed, all of which are key components of a practically useful transaction network.

We estimate that Axios consensus in conjunction with the chain mesh structure will give Logos a maximum capacity of 2,500 to 15,000 transactions per second, depending on delegate hardware and internet speed. This is a substantial improvement over the 7 tps offered by Bitcoin or the 14 tps offered by Ethereum.

State Sharding: Polis

The third major scalability component is Logos's state sharding system, *Polis*. Sharding is an established concept in distributed databases that involves dividing the database into parts that are stored by disjoint groups of nodes. While an unsharded consensus system requires that a supermajority of validator nodes see every transaction, a sharded system only requires that a fraction of nodes see each transaction. This means that different shards can process different transactions in parallel, which roughly multiplies the capacity of the network by the number of shards. Sharding therefore allows Logos to achieve scaling in the number of validators.

Implementing sharding is a non-trivial task, which is why few cryptocurrencies have implemented this critical component. The key concern with sharding is that each shard must not break the Byzantine assumption (that is, more than $^2/_3$ of nodes are honest). For instance, if malicious accounts controlled 10% of the total stake in a 5 shard network, then they could potentially control up to 50% of total shard stake if they were all placed in the same shard. This would jeopardize safety for accounts in that shard.

There are several mitigating measures that can reduce the risk of a Byzantine shard. Most important is a source of practical randomness that cannot be gamed. If there are a sufficiently large number of nodes (1000 or more in a 10 shard network) and more than $^2/_3$ of them are

honest, then randomly assigning nodes to shards will result in all shards having more than $^2/_3$ honest nodes with a very high probability.[7]

We estimate that adding sharding on top of Axios consensus and chain mesh will give Logos a capacity of more than 100,000 transactions per second.

Economic Incentive System

One of the most critical parts of a healthy crypto network is the economic incentive system, which is embedded throughout the system components. Logos takes care to ensure that the network has the proper incentives to promote growth, encourage security, and mitigate attacks.

The network is paid for via transaction fees, which are part of each request. Due to the huge scalability of the network, these fees can be quite low per transaction but large in aggregate. We estimate that the average fee will remain below $0.0001, which ensures that the network remains accessible for a full range of microtransaction applications. These transaction fees are paid to the delegates and representatives, and they are a key feature for Logos.

While the concept of a zero-fee network is appealing, it fails upon closer scrutiny. There are real economic costs to validating a network, particularly hardware, bandwidth, and energy costs. While enthusiasts and early adopters can afford to support the network when it is only processing double digit transactions per second, the cost will become prohibitive at the scale of hundreds of thousands of transactions per second. To prevent a classic "tragedy of the commons", it is necessary to reward nodes to validate transactions on advanced hardware.

There are two plausible compensation schemes to incentivize validators. Bitcoin and many other cryptocurrencies subsidize validators with newly minted tokens, which effectively dilutes all token holders via inflation. The alternative is direct transaction fees, which charge users based on their usage. The transaction fee model is economically optimal since it allocates capacity to those who value it the most, while the inflation model results in deadweight loss. As a result, Logos is committed to the transaction fee model in the long term. While we expect that the transaction fees will never be prohibitive, Logos will nevertheless subsidize validation initially to bootstrap the network to maturity.

Economic incentives are also important to protect the network. Validators are incentivized to act properly with both "carrot" and "stick" incentives. Fees are locked up until validators finish an epoch properly, and all validators benefit from processing as many transactions as possible. Conversely, if validators fail to stay connected to the network or properly process transactions, they lose their fees. The most extreme negative incentive is the slashing mechanism, which confiscates the entirety of a validator's stake in cases of provably malicious behavior.

[7] The math behind this conclusion is beyond the scope of this primer but is explained in the white paper.

Accounts are also incentivized to act benignly. Transaction fees act as an anti-spam measure. Similarly, each request requires a small proof-of-work to prevent spam (note that this is entirely separate from the proof-of-stake validation scheme). Finally, to prevent account database bloat and encourage pruning, each new account requires a small deposit that is released when the account is closed. This also makes it easy to blacklist spamming accounts since it is not viable to open many accounts.

Logos's robust economic incentive structure ensures that network capacity is allocated to those who value it most, keeps the network free from spam or frivolous transactions, provides strong security, and encourages hardware investment that facilitates scaling.

Tradeoff between functionality and capacity

There is a "No Free Lunch" principle for cryptocurrencies that states: given finite computational resources and the need for universal consensus, a single network cannot provide an arbitrarily broad and complex range of services for an arbitrarily low cost and high performance.

This means that a network that attempts to be a "silver bullet" for all cryptocurrency applications, from decentralized storage and computation to payments, is bound to be inferior to networks that specialize on a specific application. This is because there is an inherent tradeoff between the complexity of features supported by the network and throughput. For example, any unbounded smart contract system will require significantly more computation by each validator than a pure payment system, and so it will necessarily be slower, all else equal.

Many of the networks that aim to solve the scaling problem in cryptocurrencies make unrealistic promises that fail to account for this tradeoff. On the other hand, the Logos team thinks critically about each potential feature to weigh its accretive value against its increased overhead. All functionality, such as Logos's bounded smart contract system, is designed to serve the goal of providing the premier transactional network.

Life Cycle of a Transaction

To illustrate how Logos works, we will walk through the life cycle of a transaction of 10 tokens from Alice to Bob.

When the transaction is issued, there are four current delegates: Dan, Dave, Delilah, and Daphne. Unbeknownst to the other characters, Daphne is an attacker who wants to disrupt the network.

The transaction progresses from issuance to commitment in several steps:

1. Alice runs a random delegate selection function, which chooses Dan. She designates Dan as the leader in her send request.

2. Alice digitally signs her send request and transmits it to Dan. Since a valid request from Alice's account must contain her signature, it is impossible for anyone else to fraudulently issue requests on her behalf.[8]

3. Upon receiving the send request, Dan checks to confirm it is valid. This includes checking the signature, ensuring that Alice has sufficient funds, and confirming that there is not a conflicting transaction in her account chain. Dan concludes that the send is indeed valid and generates a signed "pre-prepare" message that he sends to Dave, Delilah, and Daphne.[9]

4. Dave and Delilah also check that the transaction is valid. After doing so, they send signed "prepare" messages back to Dan. Daphne may do anything at this point, including sending a prepare message or not responding, but she is careful not to violate any slashing conditions since she does not want to lose all of her funds. Note, even if she sends conflicting messages and risks her funds, the end result to Dan is the same.

5. Dan receives the "prepare" messages back from Dave and Delilah, at which point he has valid "prepares" from more than two-thirds of delegates (including his original "pre-prepare"). He constructs a "post-prepare" message that contains proof of these "prepares" using a special Schnorr signature and sends it to Dave, Delilah, and Daphne.

6. Dave and Delilah confirm that the "post-prepare" message does indeed contain valid "prepare" signatures from more than two-thirds of delegates. After doing so, they send signed "commit" messages back to Dan. Daphne continues to act arbitrarily but can do

[8] If Alice selected Daphne, the transaction would not go through since Daphne will probably not initiate consensus (since she is malicious). Alice deals with this situation by waiting until a timer has expired, at which point she can resubmit the transaction to another delegate.

[9] If the transaction is invalid, Dan typically will just ignore it.

nothing to halt progress.

7. Dan receives the "commit" messages back from Dave and Delilah, at which point he has valid "commit" from more than two-thirds of delegates (including his original "post-prepare"). He constructs a "post-commit" message that contains proof of these "commits" using a special Schnorr signature. This “post-commit” message is sent to Alice, the other delegates, and other nodes that Dan knows.

8. Upon receiving a valid “post-commit” message, a node will forward it to the nodes it knows. In this way, the message is transmitted throughout the network. Each honest node that sees the message will add the send transaction to Alice’s account chain and deduct 10 tokens from her balance. They will also add 10 tokens to the unsettled pool and remember that only Bob can claim them.[10]

9. When Bob sees the “post-commit” message, he has proof that the network has irreversibly recorded Alice’s send transaction to him. He then can create a receive request that contains the “post-commit” proof. This is then sent to a random delegate, and the process repeats. Provided that the delegates have not changed, this can be executed very quickly since the delegates that committed Alice’s send are known.

10. Upon receiving a valid “post-commit” message for Bob’s receive request, network nodes will remove Alice’s sent tokens from the unsettled pool and credit them to Bob’s account. The receive request is concurrently added to Bob’s account chain.

11. Once the transaction has propagated through the network, all honest nodes are guaranteed to have the same view of Alice and Bob’s account chains. The delegates include the transaction in the epoch block, which is then added to the archive chain. At this point, most network nodes can prune the transaction from their storage since they only need to remember Alice and Bob’s balances.

[10] Typically, Alice will also specify a transaction fee, which is added to the transaction fee pool and can be claimed by the delegates after the current epoch concludes. Since Daphne never sent a “commit” message for the transaction, she loses her portion of the transaction fee.

Use Cases

Logos is a practical solution for all applications requiring efficient and secure transfer of value.

IoT Microtransactions

The Internet of Things market is projected to generate $450 billion in annual revenues by 2020. To achieve their full potential, IoT devices will require the ability to interact with and transfer value to other devices on a trustless basis. Logos provides an ideal solution to economically connect the IoT. It offers unbounded scalability, rich transactions, and minimal transaction fees, enabling frictionless and intelligent transfers. Furthermore, Logos is carefully designed to accommodate low footprint light clients that can run on limited hardware without having to continually trust a third party full node. Finally, its programmability allows for interoperability with other networks used by IoT devices, both public and private.

Merchant Payments

Merchants require secure payment networks that provide both high transaction throughput and low confirmation latency. Logos is designed around these qualities, and the Logos ecosystem will develop applications that enable seamless network integration with merchant systems. Logos offers additional benefits of global usability (particularly valuable for internet commerce), low fees (orders of magnitude cheaper than the 2.5% charged by credit card networks), and transaction finality (which will reduce losses due to fraud and charge backs). Logos is able to finalize transactions faster than other cryptocurrencies due to the chain mesh structure, allowing merchants to be confident that transactions will not be reversed as soon as they are approved. This drastically reduces fraud such as improper charge backs, and savings can be passed on to consumers. At the same time, Logos's cryptographically secure account system reduces the risk of theft, protecting users in the event of data breaches and removing the need for trust in online payments.

Peer-to-Peer Transfers

Logos is entirely open and trustless, enabling anyone anywhere in the world to join the network and transact with other users in a decentralized manner. Compared to centralized peer-to-peer payment solutions like PayPal and Venmo, users always control their funds, so they cannot be frozen or locked up arbitrarily. The Logos ecosystem will include frameworks that will empower secure and simple peer-to-peer transactions. This functionality can be extended to a diverse range of applications, including game economies, international remittances, and alternative payment systems for countries with unstable currencies.

International Trade and Reserves

Many countries will welcome the chance to settle corporate accounts in a currency independent from any single country or central authority. Currently, most countries need to hold large reserves of international currency to facilitate trade that occurs in dominant currencies like USD and EUR. Countries like China (exports priced in USD) and Russia (commodities priced in USD) are deeply uncomfortable with this status quo, as it ties a

substantial portion of their economy to a foreign government and central bank. A non-sovereign, trustless crypto network like Logos is far preferable as a reserve. Furthermore, Logos's highly secure and fast transaction capabilities are an attractive option for companies transacting with foreign entities, as it substantially reduces risk and settlement times while being easy to integrate.

Store of Value

In many ways, cryptoassets are ideal stores of value. They are open, decentralized, cryptographically secure, censorship-resistant, mostly non-inflationary, cheap to store, and increasingly easy to use. The main barrier to this use case is the high volatility of crypto prices; however, as successful cryptoassets mature and adoption increases, volatility will dampen significantly. Logos, as a scalable, highly secure, and minimally frictional payment system with interoperability with other systems, is an ideal cryptoasset for store of value. A few dominant, transaction-focused cryptoassets will likely win out as stores of value and have a high chance of displacing much of the $3 trillion of gold used for store of value and the $12 trillion of national fiat currencies held as reserves. Store of value is thus the highest potential source of cryptoasset value, and Logos is well positioned to capture a portion of that value.

Legal Notice

The information contained in this presentation is confidential and may constitute trade secrets and information that is otherwise protected by applicable law. This presentation is only for informational purposes, and may not be copied or disclosed to anyone without the express written permission of Promethean Labs LLC ("Promethean").

The information included in this presentation is based on information reasonably available to Promethean as of the date hereof, and does not purport to be complete. Promethean does not undertake any duty to update the information set forth in this presentation. Furthermore, the information included in this presentation has been obtained from sources that Promethean believes to be reliable. However, these sources cannot be guaranteed as to their accuracy or completeness. No representation, warranty or undertaking, express or implied, is given as to the accuracy or completeness of the information contained herein by Promethean, its members, managers, employees representatives or affiliates, and no liability is accepted by such persons for the accuracy or completeness of any such information.

Projected figures set forth in this presentation are hypothetical in nature and are shown for illustrative, informational purposes only. This material is not intended to forecast or predict future events, but rather to demonstrate the anticipated business activities of Promethean.
This presentation contains certain "forward-looking statements," which may be identified by the use of such words as "believe," "expect," "anticipate," "should," "planned," "estimated," "potential," "outlook," "forecast," "plan" and other similar terms. Examples of forward-looking statements include, without limitation, estimates with respect to financial condition, results of operations, and success or lack of success of Promethean projects. All are subject to various factors, including, without limitation, general and local economic conditions, changing levels of competition within certain industries and markets, changes in legislation or regulation, and other economic, competitive, governmental, regulatory and technological factors, any or all of which could cause actual results to differ materially from projected results.

This presentation does not constitute an offer to sell or the solicitation of an offer to purchase any securities of Promethean or any of its affiliates. Any such offer or solicitation may be made only by means of the delivery of a confidential private placement memorandum and other definitive legal documentation, which will contain material information not included herein, and which will supersede, amend and supplement this presentation in its entirety.

Direct and indirect investments in distributed ledger technologies, cryptocurrencies and other digital assets involve substantial risks due in part to the highly speculative nature of such investments, risks relating to the regulatory regimes governing such technologies and other assets, and uncertainty relating to technology. Promethean cannot anticipate every possible current or future regulation or technological development that may affect Promethean's businesses and operations. Future developments may have a significant impact on Promethean's businesses and operations causing it to lose some or all of its working capital. The information set forth in this presentation does not constitute legal, tax, investment or other advice, or a recommendation to purchase or sell any particular security.