Syscoin 4.0: A Peer-to-Peer Electronic Cash System Built For Business Applications

JAGDEEP SIDHU, MSC[1], and IAN C. MOORE, PHD[2]
[1]Syscoin Core Developer, Blockchain Foundry Inc.(e-mail: jsidhu@blockchainfoundry.co)
[2](e-mail: ic3moore@gmail.com)

ABSTRACT Syscoin 4.0 introduces a novel implementation of a decentralized identityz masternodes providing bonded validators for a PoW/PoS hybrid consensus model (DAG) / zero knowledge proofs and instant pseudo-interactive zero-confirmation cryptocurrency transactions with double-spend protection.

INDEX TERMS Network EVM, DAG, Decentralized Identity, Zero Knowledge Proofs

I. INTRODUCTION

Syscoin 4.0 builds upon Syscoin 3.0 with additional implementation of an Ethereum Bridge, Offers/Escrow, Lightening Networks, and Decentralized Identity. As previously featured, anything pertaining to the market place (eg, digital sales, auctions, marketplace modification, etc.) has been deprecated. The full release will included the Syscoin Network-Enhanced Virtual Machine (NEVM) which will utilize the Ethereum Virtual Machine (EVM) together with a Zero Knowledge Proof (ZKP) system to build scalable applications and the introduction of a decentralized cost model around Ethereum Gas fees (see Table 1).

High gas fees and low transaction throughput are some of the key issues that are hindering ERC-20 projects from running at scale. However, Ethereum 2.0 is currently under development to address these issues, where its final release is anticipated to be completed several years from now [3]. Instead of building a competing smart contract system, we have decided to work with the EVM protocol, and to provide a faster cost effective alternative over Ethereum 2.0. The roadmap for Syscoin 4.0 proposes a four-layer tech stack to address this using: (a) Syscoin as the base layer to provide security; (b) EVM for programmability; (c) ZKP for scalability; and (d) applications to provide decentralized usability for the next phase of the Internet's evolution, known as Web 3.0.

In Section II we address the protocol attributes, which include Z-DAG, order-of-events, point-of-sale applications, assets, non-fungible tokens, lightning networks, offers/escrow, and masternodes. In Section III we lay out the roadmap for Syscoin 4.0, which includes scalability, design proposal, evidence, and applications. In Section IV we outline the protocol specifications, followed by the conclusion in Section V.

TABLE 1: Syscoin 4.0 Upgrades and Modifications

Syscoin Feature	Update (from 3.0)
Nakamoto ZDAG (ie, Z-Dag)	no change
Assets	modified
Offers and decentralized marketplace	
Digital Sales	deprecated
Auctions	deprecated
Reselling w/ whitelists	modified
Feedback and rating	deprecated
Multiple payment options	deprecated
Shipping notification system	deprecated
Marketplace moderation	deprecated
Data Anchoring	deprecated
Instant Encrypted Messages	re-purposed
Blockchain pruning	deprecated
Certificates (see Identity)	modified
Decentralized Identity	new feature
Certificates	modified
Lightning networks	new feature
Offers / Escrow	new feature
Roadmap to Web 3.0	
Network EVM (NEVM)	proposed
Zero Knowledge Proofs	proposed
NEVM Applications	proposed

depreciated → removed from 4.0; modified → repurposed from 3.0; new feature → upgrade from 3.0; no update → carry-over from 3.0

II. PROTOCOL CHARACTERISTICS

A. Z-DAG

Zero-Confirmation Directed Acyclic Graph (Z-DAG) is an instant settlement protocol functioning across all Syscoin services. Syscoin services consist of Alias Identities, Certificates, Escrow, Offers and Assets. Each service is controlled via an Alias, in which ownership is proven through a private key that matches each unique address. Z-DAG organizes transactions based on dependencies to build the state in a deterministic fashion. This helps protect against double spends where an asset is transferred falsely by creating multiple transactions through multiple nodes within a short time.

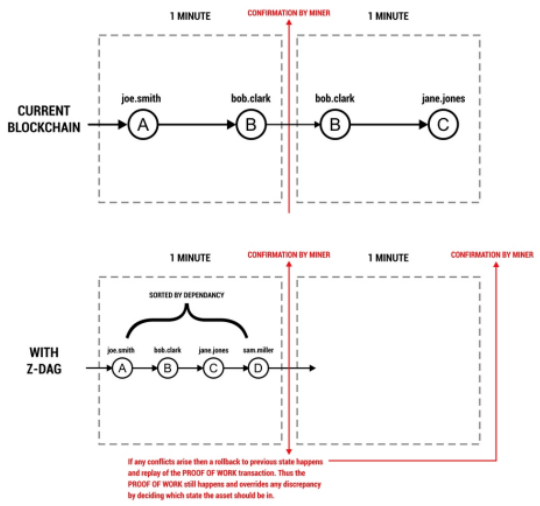

FIGURE 1: Current Blockchain design vs Z-DAG

Figure 1 shows the difference between a regular blockchain and Syscoin's Z-DAG implementation. Syscoin now has two consensus layers. The first layer consists of a Z-DAG graph of transactions are represented in the mempool without a block, providing settlement in real-time. The secondary layer provides confirmation and conflict resolution preventing double-spend events through existing Proof-of-Work (PoW) consensus.

Verifying client nodes creates a graph of transactions by looking at the sender/receiver list of asset transfers. A circuit detection algorithm is applied (read below Hawick cycle detection) and used to remove cycles from within the graph to create a DAG (Directed Acyclic Graph). Once the DAG is created, it is topologically sorted with an asset transfer consensus code processed in sequence, to form a deterministic state among the entire network in consensus.

Z-DAG can be applied to state changes (similar to UTXO updates to Bitcoin). If any discrepancy occurs, PoW will override and replay the correct order of events according to the miner; this is ensured by saving the previous state on every block and rolling back to the previous state prior to every block.

A User Interface (UI) layer will notify the user of conflicts in real-time, making the transaction occur between 3-5 seconds (ie, the amount of time needed to ensure that network takes to notice that 2 double spend transactions are conflicting with each other). This is possible through Syscoin's Masternode layer. Every Masternode is connected to 25 or more peers providing high-throughput relay across the network, averaging one or more network hops to transmit from the sender to the receiver nodes.

The partition tolerance of the protocol is the PoW block timing, which is set to 60 seconds on Syscoin 3. For every block, a new DAG is constructed from the transactions within that block. Unlike other implementations that have no PoW to fall back on, there is no case where the DAG tends toward an incorrect state over time. To accurately detect double-spends, other implementations such as Phantom or Spectre [4] must use an algorithm to replace the longest chain rule to derive the order of events and attacker sequences in a graph. These implementations end up in a more complex game theoretical situation that has not been proven mathematically to be accurate in all cases. Syscoin relies on the thoroughly tested Nakamoto consensus model to arrive at a consensus over time rather than simply relying on a DAG.

B. ORDER-OF-EVENTS PRESERVATION AND CONFLICT RESOLUTION

Mauris ipsum. Nulla metus metus, ullamcorper vel, tincidunt sed, euismod in, nibh. Quisque volutpat condimentum velit. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nam nec ante. Sed lacinia, urna non tincidunt mattis, tortor neque adipiscing diam, a cursus ipsum ante quis turpis. Nulla facilisi. Ut fringilla. Suspendisse potenti. Nunc feugiat mi a tellus consequat imperdiet. Vestibulum sapien. Proin quam. Etiam ultrices. Suspendisse in justo eu magna luctus suscipit.

C. CAP THEOREM

The CAP theorem [6] states that it is impossible for a distributed data store to simultaneously provide more than two out of the following three guarantees: consistency, availability or partition tolerance. Bitcoin attempts to provide a guarantee that transactions are settled, but is not able to do so according to theory. Z-DAG partially trades consistency for availability. Since it is unlikely that a double-spend or reorganization will change the state of someone's balance, we do not wait for settlement finality by waiting for blocks to confirm. This drastically increases the usability in point-of-sale applications.

The allowance of instant settlements and increased availability requires paying additional attention to users simply trying to double-spend or send transactions too quickly, which may cause the miner view to change from the general network view. The DAG will order the dependency graph of transactions and process in sequence, allowing for better availability when it comes to instant state changes. The same CAP constraints as PoW will provide a more responsive money transfer mechanism that may be used as a transaction processor instead of simply a settlement layer.

D. POINT-OF-SALE APPLICATIONS

The combination of using assets with Z-DAG will allow for point-of-sale applications, providing service in real-time exchange for cryptocurrency asset tokens.

E. ASSETS

Sed lectus. Integer euismod lacus luctus magna. Quisque cursus, metus vitae pharetra auctor, sem massa mattis sem, at interdum magna augue eget diam. Vestibulum ante ipsum

primis in faucibus orci luctus et ultrices posuere cubilia Curae; Morbi lacinia molestie dui. Praesent blandit dolor. Sed non quam. In vel mi sit amet augue congue elementum. Morbi in ipsum sit amet pede facilisis laoreet. Donec lacus nunc, viverra nec, blandit vel, egestas et, augue. Vestibulum tincidunt malesuada tellus. Ut ultrices ultrices enim. Curabitur sit amet mauris.

F. NON-FUNGIBLE TOKENS

Non-Fungible Tokens (NFTs) are unique Syscoin-based digital assets that can not be replaced with something else. Some example uses are: (a) gaming; (b) digital art; (c) physical assets; and (d) digital collectibles. Some advantages of using Syscoin over other platforms include scalability, divisibility, efficiency, and notary capabilities.

By utilizing the NEVM, Syscoin will offer NFTs running off the application layer of its tech-stack which will offer a competitive advantage over Ethereum in terms of scalability. Another interesting feature Syscoin will be offering is divisibility. A good example of this is land ownership where fractions of such assets can be apportioned between multiple parties. Finally, developers will be able to employ notary capabilities by applying custom rule sets to transactions involving NFTs.

G. DECENTRALIZED IDENTITY

Sed lectus. Integer euismod lacus luctus magna. Quisque cursus, metus vitae pharetra auctor, sem massa mattis sem, at interdum magna augue eget diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Morbi lacinia molestie dui. Praesent blandit dolor. Sed non quam. In vel mi sit amet augue congue elementum. Morbi in ipsum sit amet pede facilisis laoreet. Donec lacus nunc, viverra nec, blandit vel, egestas et, augue. Vestibulum tincidunt malesuada tellus. Ut ultrices ultrices enim. Curabitur sit amet mauris.

1) Selfish mining

Integer lacinia sollicitudin massa. Cras metus. Sed aliquet risus a tortor. Integer id quam. Morbi mi. Quisque nisl felis, venenatis tristique, dignissim in, ultrices sit amet, augue. Proin sodales libero eget ante.

2) Chain locks

With a subset of nodes offering sybil resistance through the requirement of bonding 100,000 SYS to become active, plus the upcoming deterministic masternode feature in Syscoin 4.2. We have enabled Chain Locks which attempts to solve a long-standing security problem in Bitcoin [9]. Dashcore was the first project to implement this idea [20] which the industry has since widely accepted as a viable solution [16]. Our implementation is an optimized version of this, in that we do not implement Instant Send or Private Send transactions and thus Syscoin's Chain Lock implementation is much simpler. Because of merged-mining functionality with Bitcoin, we believe our chain coupled with Chain Locks becomes more secure via solving Bitcoin's most vulnerable attack vector (ie, selfish mining).

These Chain Locks are made part of Long-Living Quorums (LLMQ) which leverage aggregatable Boneh–Lynn–Shacham (BLS) signatures that have the property of being able to combine multiple signers in a Distributed Key Generation (DKG) event to sign on decisions. In this setup, a signature can be signed on a group of parties under threshold constraints without any one of those parties holding the private key associated with that signature. In our case, the signed messages would be a ChainLock Signature (CLSIG) which represent claims on what block hashes represent on the canonical chain [20]. This model suggests a very efficient threshold signature design was needed to arrive to consensus across the Masternode layer to decide on chain tips and lock chains, hence preventing selfish mining attacks. See [17] to understand the qualities of BLS signatures in the context of multi-sig use cases.

Ethereum 2.0 caters to the use of BLS signatures through adding precompile opcodes in the Ethereum Virtual Machine (EVM) for the BLS12-381 curve [18] which Syscoin has adopted. This curve was first introduced in 2017 by Bowe [19] to the ZCash protocol. Masternodes on Syscoin have adopted this curve and have a BLS key that is associated with each validator. See [20] for performance comparison to ECDSA (Secp256k1) which discusses its usefulness in contrast to what Bitcoin and Syscoin natively use for signature verification.

3) Reselling and Whitelists

Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. Mauris ipsum.

H. LIGHTNING NETWORKS

Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. Mauris ipsum. Nulla metus metus, ullamcorper vel, tincidunt sed, euismod in, nibh. Quisque volutpat condimentum velit. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nam nec ante. Sed lacinia, urna non tincidunt mattis, tortor neque adipiscing diam, a cursus ipsum ante quis turpis. Nulla facilisi. Ut fringilla. Suspendisse potenti. Nunc feugiat mi a tellus consequat imperdiet.

I. OFFERS / ESCROW

Vestibulum sapien. Proin quam. Etiam ultrices. Suspendisse in justo eu magna luctus suscipit. Sed lectus. Integer euismod lacus luctus magna. Quisque cursus, metus vitae pharetra auctor, sem massa mattis sem, at interdum magna augue eget diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Morbi lacinia molestie dui.

Praesent blandit dolor. Sed non quam. In vel mi sit amet augue congue elementum. Morbi in ipsum sit amet pede facilisis laoreet. Donec lacus nunc, viverra nec, blandit vel, egestas et, augue. Vestibulum tincidunt malesuada tellus.

J. MASTERNODES

With 2400+ masternodes running fullnodes, Z-DAG becomes more dependable, as does the propagation of blocks and potential forks. Masternodes are bonded through a lossless strategy of putting 100,000 Syscoin in an output and running full nodes in exchange for block rewards. A seniority model incentivizes masternodes to share long-term growth by paying more for the longer period of service. Half of the transaction fees are also shared between the PoW miners and masternodes to ensure long term alignment once subsidy becomes negligible. Coins are not locked at any point, and there is no slashing condition if masternodes decide to move their coins, the rewards to those masternodes simply stop. Sharing Bitcoin's compact block design, it consumes very little bandwidth to propagate blocks assuming the memory pool of all these nodes is roughly synchronized [14]. The traffic on the network primarily consists of propagating the missing transactions to validate these blocks. Having a baseline for a large number of full-nodes that are paid to be running allows us to create a very secure environment for users. It proposes higher costs to would-be attackers who either have to attempt a 51% attack of Syscoin (ie, effectively trying to attack the Bitcoin network), or try to game the mesh network by propagating bad information which is made more difficult by incentivized full-nodes. The health of a decentralized network consists of the following; (a) the mining component or consensus to produce blocks, and (b) the network topology to disseminate information in a timely manner in conditions where adversaries might be lurking. Other attacks related to race conditions in networking or consensus code are mostly negligible as Syscoin follows a rigorous and thorough continuous development process. This includes deterministic builds, Fuzz tests, ASAN/MSAN/TSAN, functional/unit tests, multiple clients and adequate code coverage. Syscoin and Bitcoin protocol code bases are merged daily such that the build/signing/test processes are all identical, allowing us to leverage the massive developer base of Bitcoin. Code quality is reflective of taking worst case situations into account. The most critical engineers and IT specialists need confidence that value is secure should they decide to move their business to that infrastructure. It's true that there are numerous new ideas, new consensus protocols and mechanisms for achieving synchronization among users in a system through light/full node implementations. However, in our experience in the blockchain industry over the last eight years, we understand that it takes years, sometimes generations to bring those functionalities to production level quality useful for commercial applications.

III. ROADMAP TO WEB 3.0

Bitcoin was the first to offer a practical outcome to the General's Dilemma using Crypto Economic rationale and incentives. Ethereum was the first to abstract the concept of turing completeness within similar frameworks assumed by Bitcoin. What Syscoin presents is a combination of both Bitcoin and Ethereum with intuitions built on top to achieve a more efficient financial computing platform which leverages coordination to achieve consensus using Crypto Economic rationale and incentives. We propose a four-layer tech stack using Syscoin as the base (host) layer, which provides an efficient (ie, low gas cost per transaction) platform. Some of the main advantages include building scalable decentralized applications, the introduction of a decentralized cost model around Ethereum Gas fees. This new model proposes stateless parallelized execution and verification models while taking advantage of the security offered by the Bitcoin protocol. We may also refer to this as Web 3.0.

A. SCALABILITY AND SECURITY

Scalability in blockchain environments is typically measured by total Transactions per Second (TPS). This suggests full trustlessness, decentralization and liveness properties as evidenced by something like Bitcoin. If trade-offs are made to achieve higher scale, it means another property is affected. A full node is one that creates blocks and/or fully validates every block of transactions. For the purpose of this discussion, we will refrain on expounding on designs where light-clients are used to give semblance of higher throughput, etc. However, if two nodes are running the same hardware and doing the same work, the one that provides more TPS performance than the other is considered more scalable. This is not to be confused with throughput which is the measure of output that can be increased by simply adding more hardware resources. Hence, more throughput does not mean more scalable. Some blockchains require the producers of blocks to run on higher specifications, offering higher throughput but not necessarily more scale. However, there are projects which employ parallel processing to try to achieve higher scale whilst also enforcing more capable hardware to provide a more efficient overall system [33]. As a logical experiment, the throughput of a system divided by the scalability of the system is what we define as efficiency. In the following sections, we will outline our proposal for improved efficiency.

1) Efficiency

The holy grail of blockchain design resides in the ability to have a ledger that can claim to be sublinear while retaining consistency, fault tolerance, and full availability (ie, CAP Theorem). This concedes there are roughly constant costs for an arbitrary amount of computation performed and being secured by that ledger. This has always been thought of as impossible, unless acceptable trade-offs appear in application designs if they are easy to understand and work around. Most experts make the assumption that an O(1) ledger is simply

impossible and thus design blockchains and force applications to work in particular ways as a result. We will remove such assumptions and let business processes dictate how they work by giving the ability to achieve O(logk n) for some constant k (ie, polylogarithmic) efficiency with trade-offs. A polylogarithmic design would give the ability for almost infinite scaling over time for all intents and purposes. The only bottlenecks proposed are fast information that can be propagated across the network. This in turn would improve over time as telecom infrastructure naturally evolves and increases in both capability and affordability.

Put in context, even Lightning Networks for transactional counts qualifies as a form of sublinear scaling on a transactional basis but not per user, as users must enter the main chain first before entering a payment channel. It requires the state of the blockchain to include users joining the system. This state (ie, UTXO balances) is the primary factor of efficiency degradation in Bitcoin. Users need to first start on the main chain and then move into the payment channel system to receive money, meaning that scale is at best O(N) where N is the number of users. There are some solutions to this problem of state storage on Bitcoin by reducing it via an alternative accumulator strategy to the cost of increased bandwidth [25]. This approach would make the chain stateless, however validation costs would remain linear to the number of transactions being done. When combined with payment channels, only costs to get in/out are factored into the validation. This offers an interesting design for payments themselves while providing for on-chain availability. We consider this as a satisfactory path for futuristic scalable payments. Hence, it is not possible to employ that strategy with general computations. With this design, we are still left with the issue on how to do general computations at higher efficiency.

What we present is the ability to have a polylogarithmic chain at the cost of availability for both payments and general computations where business processes dictate availability policies, and users fully understand these limitations when using such systems. Users will ensure availability for themselves and others at their discretion. This will be expanded upon in the following sections.

2) State Liveness and State Safety

While many compelling arguments can be made migrating to a state-less design [26], it is not possible to achieve sublinear efficiency without sacrificing some other desired component as outlined above. To achieve polylogarithmic efficiency, it is necessary to have a mix of state and stateless nodes working together in harmony on a shared ledger [26]. This should be accomplished so that business processes can dictate direction, and users can choose to pay a little more for security either by using a stateful (yet very scalable ledgering mechanism) or by paying to ensure their own data availability amortized over the life of that user on such systems. Presenting the ability for users to make these choices allows us to separate the consensus of such systems and reduce overall complexity. However, in whatever solution we adopt, we need to ensure that the final implementation allows for both the liveness and safety of that state, which are defined as follows:

- **State Liveness**: Transferring coins in a timely manner
- **State Safety**: Private custody

It is important to adhere to these concepts; if one cannot move one's coins, then it is as useful as if one burned them. Hence, if we had third party custody in place, this would give rise to custodial solutions. The loss of decentralized and trustless aspects of the solution is not desired.

The options as described would allow users to decide their state liveliness at their own discretion, while state safety is a required constraint throughout any system design we provide. The doorway to possibilities of sublinear design is opened by giving users the ability to decide.

3) Avoiding Re-execution of Transactions

In order to scale arbitrarily, independent of the number of transactions - a desired property of increasing throughput - one requires a mechanism to avoid re-executing transactions [27]. Ideally, it would be able to batch these transactions together for a two-fold scaling proposition. There are a few mechanisms in literature that attempted to solve re-execution: (a) TrueBit; (b) Plasma; and (c) Arbitrum avoided re-execution. Unfortunately, they require challenge response systems to ensure security, which leads to intricate attack vectors of unbounded risk/reward scenarios.

Multi-Party Computation (MPC) is a mechanism to have parties act under a threshold to decide on actions such as computational integrity of a smart contract. MPC is used in Syscoin for BLS threshold signatures for Chain Locks and Proof-of-Service in quorums of validators deterministically chosen using Fiat-Shamir heuristics on recent block hashes. The problem with this approach is that validators may become corrupt, hence they need to be wrapped in a consensus system along with DKG and random deterministic selection. This was an interesting topic of discovery for the Syscoin team in the preliminary stages as a way to potentially scale smart contract execution. It was ultimately discarded due to the incentive for risk/reward scenarios to favour attacks as the value of the transactions increases.

Hardware enclaves (eg, Intel SGX through remote attestation) were also of particular interest to the Syscoin team as a way to offload execution and avoid re-execution costs. However, there are a myriad of attack vectors and censorship concerns on the Intel platform. We should also note that the Antarctica model was interesting but required a firmware update from Intel to support such a feature which raises concerns over censorship long term [28].

The theme amongst all of these approaches is that although re-execution is avoided, the communication complexity is largely still linear with the number of transactions on the main chain. The security and trust models are also different from that of the layer 1 assumptions which was not desired.

Lacking solvent solutions to avoid re-execution and enable sublinear overall complexity, we were led - in the development of Syscoin 4.0 - to build a trust-minimized two-way bridge between Syscoin and the Ethereum mainchain, offloading the concerns around smart contracts to Ethereum. With the advent of such promising technology as ZKP and the optimizations happening around them [18, 19], we have re-considered the possibilities and believe this will have an integral role in the development of Web 3.0. This realization led us to re-test our assumptions and options related to our desired design.

ZKP allows for the desired superlinear scaling trait we had been looking to achieve, but also offer other benefits; namely privacy is very easy to introduce and will not add detectable costs and complexities to verification on the mainchain. With users controlling their own data, the mainchain and systems may be designed such that only balance adjustments are recorded, not transaction sets (we will explain the case with full data availability below). In this scenario, there is no advantage for a miner to gain when colluding with users that launch attacks on systems such as Decentralize Finance (DeFi), pools and provenance of transactions. The flexibility has to be present for application developers that need experiences consistent with those we have today with Bitcoin/Syscoin/Ethereum. This would enable the privacy use-cases without requiring extra work, knowledge or costs.

4) Validity Proof Systems Overtop Proof-of-Work Systems

Prior to the use of Proof Systems, the only option for "Validity Proofs" in a permissionless system involved naive replay, and as such greatly limited scalability. Essentially, this is still practised in Layer-1 blockchain (L1) solutions, with the known penalty to scalability. Proof Systems offer a very appealing trait known as succinctness: in order to validate a state transition, one needs to only verify a proof, and this is done at a cost that is effectively independent of the size of the state transition (ie, polylogarithmic in the size of the state transition).

For maximal financial security, the amount of value being stored should depend on the amount of security provided on the settlement side of the ledger. PoW offers the highest amount of security guarantees (see Appendix A). Our next generation financial systems begin with optimal ledgering security and add proof systems on top for scaling. Block times are not as important in a world where most users and activity are on Layer-2 blockchain (L2) validity proof based systems. This liberates engineers who are focused on scalability to better define blocks, safe block times in addition to maximal amount of data bandwidth that can be safely propagated in a time sensitive manner across full nodes in the network. With Syscoin, there are incentivized full nodes (ie, deterministic masternodes), so again we can maximize the bandwidth of ledgering capabilities while retaining Bitcoin PoW security through merged-mining.

5) Quantum Resistance

Hashing with the SHA256 algorithm is regarded to be quantum safe because it requires Grover's algorithm to crack in the post-quantum world, and at best a quantum computer will offer only 50% reduction in time to break [32]. On the other hand, where Shor's algorithm applies, any pair based cryptographic system will be broken in hours.

For L2, we propose to implement ZKP in the SDK Layer (see Fig 2); namely Non-Interactive Zero Knowledge Proofs (NIZKP). Popular implementations of NIZKP include Zero-Knowledge Succinct Non-interactive ARgument of Knowledge (zk-SNARKS) and Zero-Knowledge Scalable Transparent ARguments of Knowledge (zk-STARKS) [33]. There are some zk-STARK/zk-SNARK friendly cipher's employed in zkRollup designs such as MiMC and Pederson hashes for which we lack certainty on classical security, yet are hopeful and would offer quantum resistance within ZKPs.

It is essential to acknowledge that Bitcoin was developed with change addresses in mind exposing the hash of a public key requires a quantum computer to use Grover's Algorithm in order to attempt stealing that Bitcoin. Each time a Bitcoin Unspent Transaction Output (UTXO) is spent, the public key is exposed and a new change address - which does not expose the public key - is used as change.

With this in mind, any scalable L2 solution should be quantum resistant because otherwise we undermine Bitcoin design as the gold standard of security.

B. DESIGN PROPOSAL FOR WEB 3.0

The following describes the 4-layers (see Fig 2) of Syscoin's proposed tech stack for Web 3.0:

1) **Host Layer:** Bitcoin's design is the gold standard for security and decentralization. PoW and Nakamoto Consensus settlement security are widely regarded by academics as the most hardened solution for ledgering value [12]. This could possibly change, however it's also arguable that the intricate design encompassing Game Theory, Economics, risk reward ratios for attack, and the minimal amounts of compromising attack vectors is likely not to change for the foreseeable future. UTXO's (and payments with them) are more efficient than account-based or EVM-based. That said, Bitcoin itself suffers from not being expressive enough to build abstraction for general computation.
2) **Operating System Layer** EVM/eWASM is the gold standard for general computation because of its wide adoption in the community. Anyone building smart contracts are likely using this model, or will continue to use it as the standard for autonomous general computation with consensus.
3) **SDK Layer** Zero-knowledge proofs are the gold standard for generalized computation scaling for blockchain applications. They enable one-time execution via a prover and enable aggregate proof checking instead of re-execution of complex transactions. zk-STARKs or zk-SNARKs using collision resistant hash functions that

work with only weak cryptographic assumptions and therefore are quantum safe. At the moment generalized smart contracts are not there yet but we are quickly approaching the day (eg, Cairo, Zinc) when there will be abstractions made to have most Solidity code trans-compile into a native zero-knowledge aware compiler similar to how .NET runtime and C# allows an abstraction on top of C/C++ as an interpretive layer on top

4) **Application Layer** Verticals or applications applying the above SDK to define business goals.

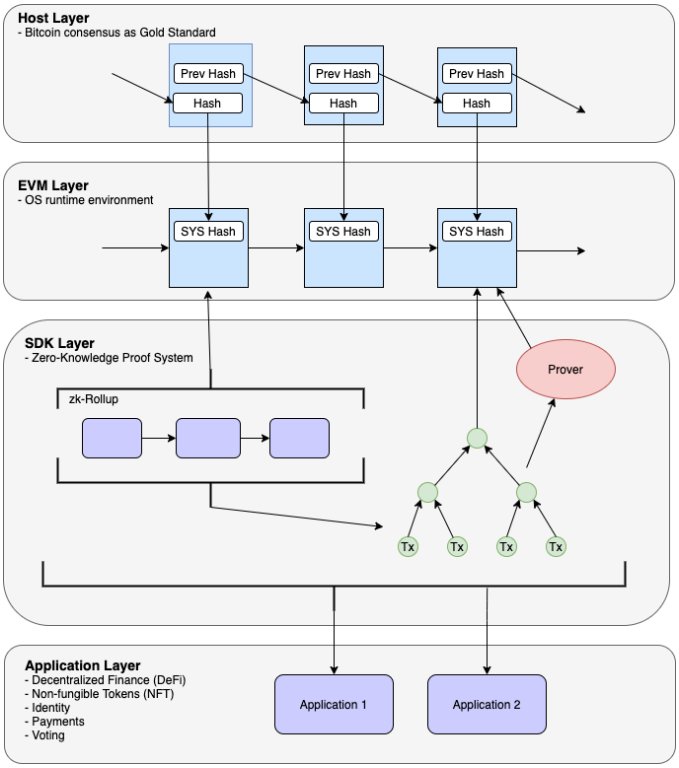

FIGURE 2: Proposed 4-layer tech stack for Syscoin

Surprisingly, these ideals represent a design that is not shared with any other project in the industry, including Bitcoin or Ethereum. We recommend that these ideals, fashioned together in a singular protocol, could possibly present a grand vision for a "World Computer" blockchain infrastructure.

Syscoin has already implemented Geth + Syscoin nodes in one application instance already (ie, release 4.2), we foresee that there will not be any challenges associated with building a consensus basis working together to form a dual chain secured by Syscoin's PoW.

Figure 5 describes a system where nodes are running two sets of software processes, the Syscoin chain protocol and an EVM/eWASM chain protocol which are kept in sync through putting the EVM tip hash into the Syscoin block. Both have their own individual mempools and effectively the Ethereum contracts, tools and processes can directly integrate as is into the EVM chain as it stands. Note that the two chains are processes running on the same computer together. Thus a SYS NODE and EVM NODE would be operating together on one machine instance (ie, Masternode) with ability to communicate with each other directly through Interprocess Communication (IPC). The intersection between the two processes happens in three points:

1) Miner of the EVM chain collects the latest block hash and places it into the Syscoin block.
2) When validating Syscoin blocks, nodes confirm the validity of the EVM tip by consulting the EVM chain software locally
3) Fees for the EVM chain are to be paid in SYS. We need an asset representing SYS on the EVM chain, which will be SYSX. We will enable this through a similar working concept that we've already established (SysEthereum Bridge). We may also enable pre-compiles on the EVM chain side to extract Syscoin block hashes and merkle roots to confirm validity of SYS to SYSX burn transactions.

As seen in Figure 6, work done on BTC is reused to create SYS blocks through the merged-mining specification. Concurrently, the miner will execute smart contracts in the memory pool of the node running the EVM chain. Once a chain hash has been established post-execution, it will be put into a Syscoin block and published to the network. Upon receiving these blocks, every node would verify that the EVM chain, which can be locally executed (ie, similar to the miner), matches the state described by the Syscoin block. Technically, one would want to ensure both the latest and previous EVM block hashes inside of their respective Syscoin blocks are valid. The block $->$ evmblock $==$ evmblock && block $->$ prev $==$ evmblock $->$ prev is all that is needed to link the chains together with work done by Bitcoin which is propagated to Syscoin through AUXPOW and can serve as a secure ledgering mechanism for the EVM chain.

Since (a) we may use eWASM; (b) there are paid full nodes running on the network; and (c) the mining costs are shared with Bitcoin miners, we should be able to safely increase the amount of bandwidth available in the EVM chain while remaining secure from large uncle orphan rates. There has been much discussion as to what the safe block size should be on Ethereum. Gas limits are increasing as optimizations are made on the Ethereum network. However, since this network would be ledgered by the Syscoin chain through PoW, there shouldbe no concern for uncle orphaning of blocks since the blocks must adhere to the policy set inside of the Syscoin block. We should therefore be able to increase bandwidth significantly and parameterize for a system that will scale globally yet still be centered around L2 rollup designs. A very important distinction to note is that the design of Ethereum 2.0 centers around a Beacon chain and sharding served by a Casper consensus algorithm [34]. The needs of the algorithm require a set of finality guarantees necessitating a move towards Proof-of-Stake (PoS). This has large security

implications for which we may not have formal analysis for a long time, however we do know it comes with big risk [35]. We offer similar levels of scalability on a network while retaining Nakamoto Consensus security. The simpler design which has been market tested and academically verified to work would result in a more efficient system as a whole with less unknown and undocumented attack vectors. Hence, we need only to consider researching the optimal parameterization of the gas limit taking into account an L2 centric system; but also a safe number of users we expect to be able to serve before fee market mechanisms begin to regulate the barrier of entry for these users. This proposed system should be scalable enough to serve the needs of global generalized computation while sticking to the core fundamentals set forth in the design explained above. Our upcoming whitepaper will have more analysis on these numbers but we will include some theoretical scaling metrics at the end of this article.

1) Related Works

The following organizations offer various open source third party L2 scaling solutions:

- Starkware
- ZEXE(Aleo)
- Matter labs
- Hermez
- Connext

Starkware is built using a general purpose language (Cairo) with Solidity (EVM) in mind, as is Matter labs with the (Zinc) language. Hermez developed custom circuits tailor-suited to fast transactions and Decentralized Exchange (DEX) like capability. These will be able to directly integrate into Syscoin without modification. As such, the optimizations and improvements they make should directly be portable to Syscoin, hence becoming partners to our ecosystem.

Aleo uses Zero knowledge EXEcution (Zexe) for zk-SNARK proof creation through circuits created from R1CS constraints. The interesting thing about Aleo is that there is a ledger itself that is purpose-built to only verify these Zexe proofs for privacy preserving transactability. The consensus is PoW, while the proof system involves optimizing over the ability to calculate the verifications of these proofs efficiently. The more efficient these miners become at verifying these proofs, the faster they are able to mine and thus the system provides sybil resistance through providing resources to verify Zexe proofs as a service in exchange for block creation. However, these proof creations can be done in parallel based on the business logic for the systems the developers need to create. There is no direct need for on-chain custom verification as these can be done in an EVM contract, similar to what Cairo Generic Proving Service (GPS) verifier and Zinc Verification do. The goal of Aleo is to incentivize miners to create specialized hardware to more efficiently mine blocks with verification proofs. However, provers can also do this as we have seen with Matter Labs' recent release of FPGA to do more efficient zkSNARK proofs [37]. It is a desirable property to use PoW to achieve "world-view" consensus in Aleo; however they focus on private transactions. They are typically not batched and employ a recursive outer proof to guarantee execution of an inner proof where the outer proof is sent to the blockchain to be verified. This proof is a limited 2-step recursion, consequently batching of arbitrary amounts of transactions is not supported. As a result, the cost of proof verification is relatively constant with a trade-off of limiting the recursion depth. Aleo is not meant to be a scalable aggregator of transactions, but mainly oriented towards privacy in their zk-SNARK constructions using Zexe.

2) Functional Overview

For scalable simple payments, one can leverage our Syscoin Platform Token (SPT) asset infrastructure and payment channels to transact at scale. Unique characteristics of SPTs include a generalized 8 byte field for the asset ID which is split between the upper and lower 4 bytes; the upper 4 are issued and definable (ie, NFT use cases) and lower 4 are deterministic. This enables the ability to have a generalized asset model supporting both Non-fungible Tokens (NFT) and Fungible Tokens (FT) without much extra cost at the consensus layers. 1 extra byte is used for all tokens at best case and 5 extra bytes are used for NFT at worst case. See [38] for more information on Syscoin's NFTs. This model promotes multiple assets to be used as input and consequently as outputs, suggesting that atomic swaps between different assets are possible within 1 transaction. This has some desirable implications when using payment channels for use cases such as paying in one currency when merchants receive another atomically. A multi-asset payment channel is a component that is desired so users are not constrained to single tokens within a network. Composability of assets as well as composability across systems (such as users from one L2 to another) is a core fundamental to UX and convenience that needs to be built into our next generation blockchain components that we believe will enable mass adoption. The Connext box shows how potentially you can move from one L2 on one network to another as described in [39]. This would promote seamless cross-chain L2 communication without the high gas fees. Since these L2's are operating under an EVM/eWASM model, there are many ways to enable this cross-communication.

An EVM layer will support general smart contracts compatible with existing Ethereum infrastructure and L2 rollups will enable massive scale. The different types of zkRollups will allow businesses and rollup providers to offer ability for custom fee markets (ie, pay for fees in tokens other than base layer token SYS). In addition, it will remove costs and thus improve scale of systems by offering custom data availability consensus modules. This design here shares similarities to the zkPorter design where a smart contract would sign off on data availability checks that would get put into the ZKP as part of the validity of a zkBlock which goes on chain.

The overall idea of the zkPorter design is that the zkRollup

system would be called a "shard", and each shard would have a type either operating in "zkRollup" mode or operating in "normal" mode.

This concludes that shards can define different consensus modules for data availability (censorship resistance mechanisms) via separating concerns around ledgering the world-view of the state (ie, ZKP that is put on L1) and the data that represents the state. Doing so would allow shards to increase scale, offload costs of data availability to consensus participants.

A few note-worthy examples of consensus for data availability are:

1) Non-committee, non fraud proof based consensus for data availability checks. No $2/3$ online assumption; see ethresear.ch post [40].
2) Sublinear block validation of ZKP system. Use something like Lazy Ledger as a data availability proof engine and majority consensus; see ethresear.ch post [41].
3) Use a combination of above, as well as masternode quorum signatures for any of the available quorums to sign a message committing to data availability checks as well as data validity. Using masternodes can provide a deterministic set of nodes to validate decisions as a service. The data can be stored elsewhere accessible to the quorums as they reach consensus that it is indeed valid and available.

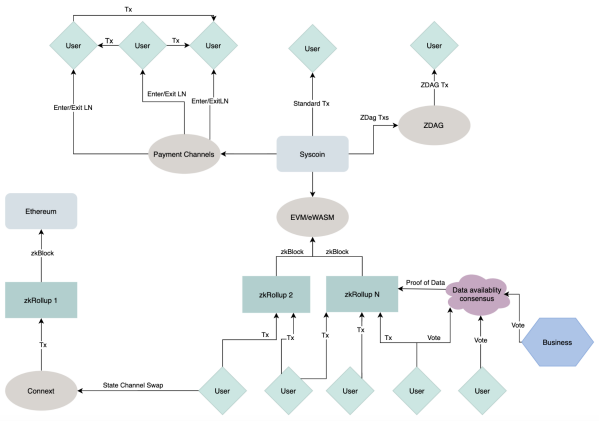

FIGURE 3: Network EVM: High level description

3) Optimistic vs ZkRollup

ZKP are excellent for complex calculations above and beyond simple balance transfers. For payments, we feel UTXO payment channels combined with something like Z-DAG is an optimal solution. However, we are left with rollup solutions for generalized computation involving more complex calculations requiring consensus.

The solution we adopt has to be secured by L1 consensus that is considered decentralized and secure, which we achieved via merged-mining with Bitcoin.

There are two types of rollup solutions today: (a) Optimistic roll ups (OR); and (b) zkRollups; which offer trade-offs.

Consensus about which chain or network you are working on is a difficult problem that is solved for us by Nakamoto consensus. We build on that secure longest chain rule (supplemented by Chain Locks to prevent selfish mining) to give us the world-view of the rollup states. The executions themselves can be done once by a market of provers, never to be re-executed, only verified, suggesting it becomes an almost constant cost on an arbitrarily large number of executions batched together. With OR you have the same world-view but the world-view is editable without verifying executions. The role of determining the validity of that world-view is delegated to someone watching who provides guarantees through crypto-economics. Zero-knowledge proofs remove crypto-economics on execution guarantees and replace them with cryptography.

See [36] to see contrasting benefits between fraud proofs (optimistic) vs validity proofs (zk). Key takeaways from this article are as follows:

- Eliminate a nasty tail risk: theft of funds from OR via intricate yet viable attack vectors;
- Reduce withdrawal times from 1–2 weeks to a few minutes;
- Enable fast tx confirmations and exits in practically unlimited volumes;
- Introduce privacy by default.

An area that has been overlooked is interoperability. A generalized form of cross-chain bridging can be seen in Chain A locking tokens based on a preimage commitment by Chain B to create a zero-knowledge proof, followed by verification of that proof as the basis for manifesting equivalence on Chain B. Any blockchain with the functionality to verify these proofs could participate in the ecosystem.

Our vision here is described using a hzkRollup centric world-view, yet it can be replaced with other technologies should they be able to serve the same purpose. As an infrastructure we are not enforcing one or the other as developers can build on what they feel best suits their needs. We believe we are close to achieving this, and that the technology is nearing the point of being ready for the vision set forth in this article.

C. EVIDENCE

Since payment channels work with UTXO's and also benefit from on-chain scaling via Z-DAG, 16MB blocks (with segwit weight) assumed, we will see somewhere around 8MB-12MB effectively per minute (per block). We foresee that is sufficient to serve Seven Billion people who may enter and exit the payment channel networks once a year (ie, 2 transactions on chain per person per year) for a total of 14 Billion transactions. Let's conservatively assume 8MB blocks and 300 bytes per transaction. Once on a payment channel, the number of transactions is not limited to on-chain

bandwidth, but to network related latencies and bandwidth costs. Therefore, we will conclude that our payment scalability will be able to serve billions of people doing 2 on-chain transactions per year which is arguably realistic based on the way we envision payments to unfold; whether that is an L2 or payment channel network that will hold users to pay through instant transaction mechanisms. On-chain, we have some metrics on Z-DAG throughput [1]; in those cases someone needs to transact for point-of-sale using the Syscoin chain. The solution for payments ends up looking like a hybrid mechanism of on-chain (ie, Z-DAG) and off-chain (ie, payment channel) style payments.

Complex transactions such as smart contracts using zkRollups require a small amount of time to verify each proof. In this case, we assume that we will host data off-chain while using an off-chain consensus mechanism to ensure data availability for censorship resistance; so the only thing that goes on the chain are validity proofs. We will assume that we will assign 16MB blocks for the EVM chain per minute.

A proof size will be about 300kB for about 300k transactions batched together which will take about 60-80ms to verify and roughly 5 to 10 minutes to create such proofs. These are the Reddit bake-off estimates using zk-STARKs, which present quantum resistance and no trusted setup. After speaking with Eli Ben-Sasson, we were made aware that proving and verifications metrics are already developed compared to what is currently presented by Starkware [43]. Hence, zk-SNARKs offer even smaller proofs and verification times at the expense of trusted setups and stronger cryptography assumptions (not post-quantum safe). We foresee that these numbers will improve over time as the cryptography improves, but current estimates suggest a rough theoretical capacity of around 1 Million TPS.

1) Gas Costs and Block Sizing

Starkware was able to process 300k transactions over 8 blocks with a total cost of 94.5M gas; final throughput was 3000 TPS (see Reddit bake-off estimates). For the following calculations, let's assume one batch-run to be 300k transactions. Ethereum can process 200kB of data per minute, with a cost limit of 50M gas per minute. Therefore, considering the Starkware benchmark test, and assuming a block interval of 13 seconds, we would achieve 3000 TPS (ie, 300 k transactions per batch-run / (8 blocks per batch-run * 13 seconds per block))

It is estimated that Syscoin will be able to process 16MB of data per minute on the EVM layer (ie, SYSX in Fig 3), which is 80x gain over Ethereum; thus a cost limit of 4B gas (ie, 80*50M) per minute. Therefore, if the Starkware benchmark test was run on Syscoin, it is estimated that Syscoin could run the equivalent of 42 batch-runs per minute (ie, 4B gas per minute / 94.5 M gas per batch-run). That would result in an equivalent of 210 k TPS (ie, 42 batch-runs per minute * 300 k transactions per batch-run / 60 seconds per minute).

If we were to consider using Validum on the Syscoin EVM layer, we estimate that we could achieve 800 batch-runs per minute (ie, 4B gas per minute / 5 M gas per batch-run). That would equate to an equivalent of 4M TPS (ie, 800 batch-runs per minute * 300 k transactions per batch-run / 60 seconds per minute).

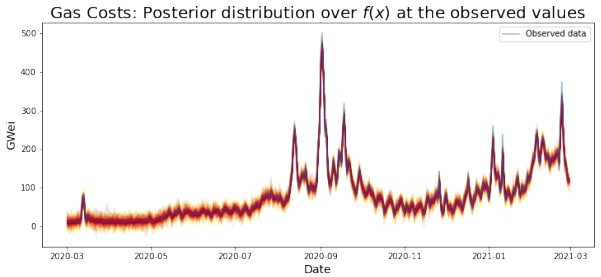

FIGURE 4: Ethereum gas gosts in GWei from Mar 2020 to Mar 2021. The region about the observed gas costs represents the posterior distribution of $f(x*)$ given observations, y, for a set of dates, $x*$; see Appendix G for details

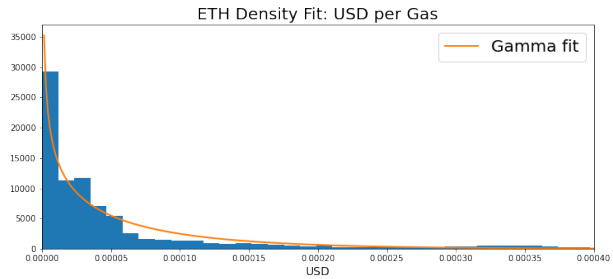

FIGURE 5: Gamma fit of Ethereum gas costs in USD using generated samples from Figure 4 and corresponding ETHUSD prices

The results from these aforementioned estimates have been tabulated in Table 2; more specifically, median estimates (along with its 5th and 95th percentiles) for the costs in USD terms for both ETH and SYSX. The estimates for Ethereum were determined using historical gas costs from March 2020 to March 2021 as shown in Figure 4, and its distribution fit in Figure 5.

With regards to the SYSX cost estimates, simulated gas costs were generated using Ethereum's proposed EIP-1559 (Figure 9) and an alternative pricing mechanism that Syscoin is currently exploring (Figure 7). This alternative approach is called the BLock Occupancy Cost (BLOC) function (15), and is driven by the occupancy of the previously generated block; see Figure 6 for simulations Appendix D for details.

Because of the higher throughput capabilities of baseline EVM, we may look to dynamically adjust costs of the gas limits [42] to thwart DOS attacks.

So far we looked at EIP-1559 and the BLOC model which both dynamically adjusted gas costs. A third option we are

Chain	Gas Limit	Block Time	Mode	Cost 300K Tx	Amortized Cost per Tx	Total TPS	USD / 300K Tx (Mar 20 to Mar 21) median	lwr 5%	upr 95 %
ETH	12.5M	13 sec	L1	6.3B gas	21,000 gas	45	159,328.24	10,669.40	1,914,394.79
			L2 zk-Rollup	94.47M gas	315 gas	3,000*	2,389.16	159.99	28,706.81
			L2 Validium	5M gas	17 gas	56,000**	126.45	8.47	1,519.36
SYSX	4B	60 sec	L1	6.3B gas	21,000 gas	3,100*	9.31453	1.76308	37.01621
			L2 zk-Rollup	94.47M gas	315 gas	210,000**	0.13967	0.02644	0.55507
			L2 Validium	5M gas	17 gas	4,000,000	0.00739	0.00140	0.02938

TABLE 2: Comparison of Gas and USD costs between ETH and SYSX. The confidence intervals for USD estimates were determined using historical gas costs for ETH (see Figure 4), and simulated gas costs for SYSX (see Figure 7)

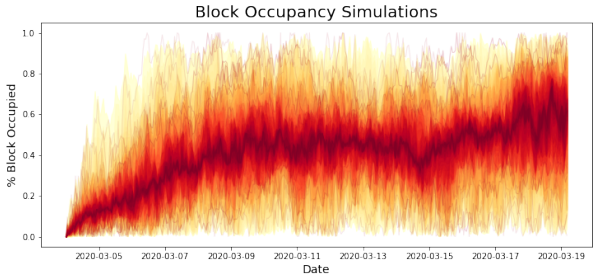

FIGURE 6: Simulated block occupancies generated using methodology outlined in Appendix E; the y axis represents percentage of block occupied at given point in time. As can be seen by the gradual upward trend, this model was selected to simulate gradual growth adoption over time.

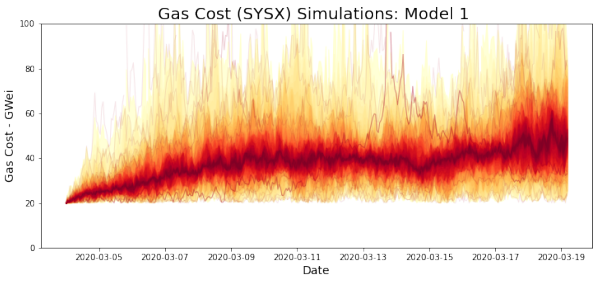

FIGURE 7: Gas cost simulations of SYSX using variant 1 of BLOC function; see Appendix E

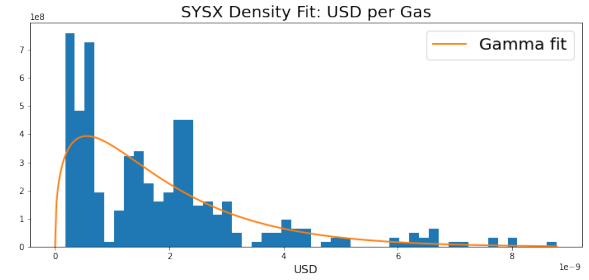

FIGURE 8: Gamma fit of SYSX gas cost simulations in USD using generated samples from Figure 7 and SYSUSD prices from Mar 2020 to Mar 2021. These estimates are hypothetical, and are be subject to market price of SYS. Note that if significant adoption occurs due to future demand for cost effective alternatives, we foresee future prices being slightly higher.

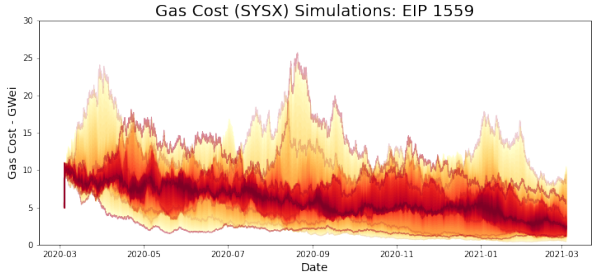

FIGURE 9: Gas cost simulations of SYSX using EIP 1559; see Appendix G. We can visual see the non-stationary decline in price over time. To our knowledge, treating this non-stationary behaviour requires better understanding of the mathematical assumptions, which have not yet been analytically researched.

looking at is maintain the auctioning with a dynamic block-size system. To facilitate this we developed a methodology driven by block failure frequency, see Appendix F for the details. Refer to Figure 10 for components of this methodology and Figure 11 for simulations .

2) Discussion

The aforementioned calculations demonstrate the full State Safety of the main chain secured by Bitcoin, and no asynchronous network assumptions which make theoretical calculations impractical in many other claims of blockchain throughput due to execution model bottlenecks. These results were extrapolated based on real results with constant overhead added that becomes negligible with optimizations. It is imperative to note that transactions in this strategy are not re-executable; there is little to no complexity in this model other than verifying succinct proofs. The proof creation strategy is parallelized organically using this model. The verifications on the main chain can also be parallelized as they are executed on separate shards or rollup networks. Dual parallel execution and verification gives exponentially more scalability than other architectures. Additionally, privacy can be built into these models at minimal to no extra cost, depending on the business model. Lastly, we suggest these are sustainable throughput calculations and not burst capacity numbers which would be much higher (albeit with

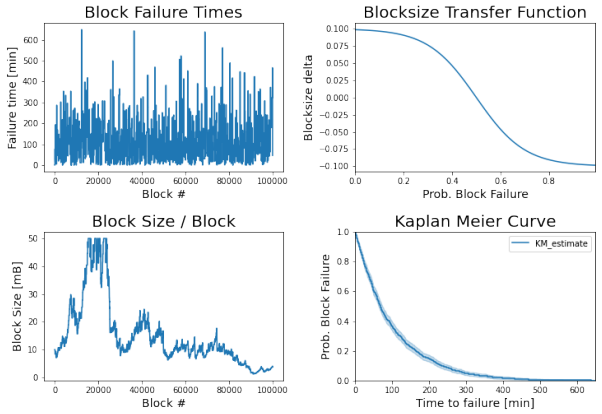

FIGURE 10: Block Resize on Failure Model (top left) block failure times; (bottom left) block sizings; (top right) block size transfer function; and (bottom right) Kaplain Meier curve; for this simulation we assumed a failure rate of 1%; once a failure event occurs we estimate the probability of block failure which is fed to the transfer function to determine the block size adjustment

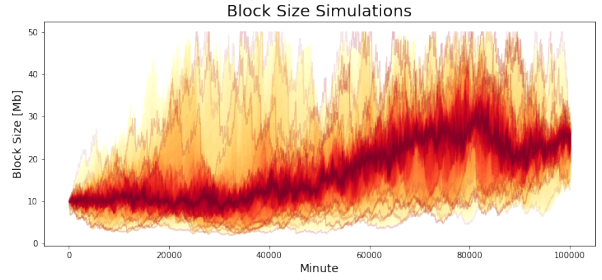

FIGURE 11: Block size simulations using the Block Resize on Failure Model as detailed in Figure 10 and Appendix F; the assumed a failure rate was 2%

a marginally higher fee based on fee markets). For example, Ethereum is operating at 15 TPS but there are around 150k transactions pending, and the average cost is about 200 GWei currently. The fee rate is based on the calculation that it takes around 10000 seconds to clear, assuming this many transactions, no new transactions, and there is demand to settle earlier. Extrapolating on 4M TPS the ratio would become 40B transactions pending with 4M TPS to achieve the same fee rate on Ethereum today assuming the memory pool is big enough on nodes to support that many pending transactions. Since masternodes on Syscoin are paid to provide uptime, we can expect network bandwidth to scale up naturally to support higher throughput as demand for transaction settlement increases. Today, the ability to transact at a much higher rate using the same hardware provides the ability for a greater scale than the state-of-the-art in blockchain design without the added desired caveat of avoiding asynchronous network assumptions. We believe this proposed design will become the new state-of-the-art blockchain, which is made viable due to its security, flexibility and parallelizable computational capacity.

In regards to uncle rates with higher block sizes, keep in mind we make uncle rates and re-organizations in general negligible through the use of the PoW chain mining Syscoin along with Chain Locks. We provide intuition that block sizes can be increased substantially without affecting network health. Furthermore, gas limits can be adjusted by miners up to 0.1% from the previous block and so a natural equilibrium can form where even if more than 4B gas is required it can be established based on demand and how well the network behaves with such increases. There is a lot to unpack with such statements and so we will cover this in a separate technical post as it is out-of-scope for this discussion.

3) Decentralized Cost Model

Decentralized cost models lead to efficiency gains in economies of scale. We set forth a more efficient design pertaining to user intent. This design uses the UTXO model to reflect simple state transitions and a ZKP system for complex computations leading to state transitions. This leads to ideal scalability for a system by allowing people to actively make their trade-off within the same ecosystem, driven by the same miners securing that ecosystem backed by Bitcoin itself.

Furthermore, a decentralized cost model contributes to scalability in that ZKP gates can generalize complex computation better than fee-market resources like gas or the CPU/memory markets of EOS, etc. This leads to more deterministic and efficient consumption of resources maximizing efficiency in calculations. It also provides opportunity for those to scale up or down based on economic incentives without creating monopolistic opportunities unlike ASIC mining. In other words, the cost is dictated by what the market can offer, via the cost of compute power (as dictated by Moore's law) instead of constrained costs of doing business on the blockchain itself. This model could let the computing market dictate the price for gas instead of being managed by miners of the blockchain. The miners would essentially only dictate the costs of the verification of these proofs when they enter the chain rather than the executions themselves.

We can begin to see computational optimization through hardware happening with ZKP and with a decentralized cost model. It will be much simpler to understand costs of running prover services as well as know how the costs scale based on the number of users and parameters of systems that businesses may employ. All things considered, it will be more efficient to make accurate decisions on data availability policies and the consensus systems needed to keep the system censorship resistant and secure.

Rollups will be friends, that is, users of one rollup system doing X TPS and users of another doing Y TPS, with the same trust model, will in effect get us to global rates of X*Y (where X is TPS of the sidechains/rollups and Y is

the number of sidechains and rollups that exist). X is fairly static in that the execution models of rollups do not change drastically (and if they do, the majority of those rollup or sidechain designs end up switching to the most efficient design for execution over time).

D. APPLICATIONS

Commercial enterprises may start to create proprietary prover technologies where costs will be lower than market in an attempt to create an advantage for user adoption. This design is made possible since the code for the prover is not required for the verifier to ensure that executions are correct. The proof is succinct whether or not the code to make the proof is available.

While the barrier to entry is low in this industry, we've seen the open source model and its communities optimize hardware and software and undergo academic peer review using strategies that outpace private funded corporations. That is plausible to play out over the long term. However, an organic market will likely form on its own, forging its own path leading to mass adoption through capitalist forces. The point here is that the privately funded vs open source nature of proving services does not change the mechanism of secure and scalable executions of calculations that are eventually rooted to decentralized and open ledgers secured by Bitcoin.

The utmost interesting propositions are the verticals that become possible by allowing infrastructure that is parameterized to scale into those economies where they are needed most, and where trust, security and auditability of value are concerns. Smart cities, IoT, AI and Digital sovereignty are large markets that intersect with blockchain as a security blanket. Although ZKP are tremendously useful on their own, applying them to consensus systems for smart contract executions drive them to another level due to the autonomous nature of "code-is-law" and provable deterministic state of logic. We believe a large part of the future economy will depend on many of the ideas presented here. Blockchain Foundry is working with commercial and enterprise adopters of blockchain technology. Our direct interaction with clients combined with our many collective years of experience in this field are reflected in this design.

IV. SPECIFICATIONS

General specifications for Syscoin 4.0:

- **Max Coins:** 888 Million
- **Deflation:** 5 percent per year until Max Coins
- **Consensus:** PoW/PoS Hybrid. PoW is SHA256 Merge-mined with Bitcoin
- **Block time:** 60 seconds
- **Rewards:** 38.5 Syscoin deflated 5 percent per year of which 10 percent is allocated to governance proposals (3.85 Syscoin). 75 percent of the result goes to masternode and 25 percent to miner.
- **Difficulty algorithm:** Dark Gravity Wave
- **Masternode collateral requirement:** 100,000 Syscoin
- **Masternode seniority**: 3 percent every 4 months until 27 percent over 3 years
- **Governance proposals payout schedule**: every month
- **Governance funding per round**: 168,630 Syscoin per month

V. SUMMARY

We have presented what is currently under way for Syscoin 4.0 which will come in the form of a four-layer tech stack to work with the Ethereum ecosystem via our NEVM. With the ever-increasing cost of gas prices, we present the data showing a 1000X+ improvement over Ethereum 2.0 in efficiency in terms of low gas cost per transaction. Syscoin will utilize the best features of the top two cryptos, namely Bitcoin and Ethereum. Hence, Syscoin will provide the security offered by Bitcoin while maintaining the programmability of Ethereum. Scalable applications will be mounted on this system via ZKP which will introduce our proposed decentralized cost model on Ethereum gas fees.

.

APPENDIX A SECURITY: POW VS POS

Proof of Stake (PoS) systems have security threats that are not inherent in Proof of Work (PoW) systems as outlined in Table 3, of which Bitcoin employs. Hence, PoS is less secure than PoW. In terms of security, we call the Bitcoin PoW protocol the gold standard, which is the same security mechanisim that Syscoin uses.

Attack type	Vulnerability		
	PoW	PoS	Delegated PoS
Short range attack	-	+	-
Long range attack	-	+	+
Coin age accumulation attack	-	maybe	-
Precomputing attack	-	+	-
Denial of service	+	+	+
Sybil attack	+	+	+
Selfish mining	maybe	-	-

TABLE 3: Comparing PoW to PoS consensus mechanism; plus sign (+) indicates vulnerability; see [12]. As shown, PoW system has the least number of vulnerabilities

APPENDIX B ZDAG FUNDAMENTALS

Zero-Confirmation Directed Acyclic Graph (Z-DAG) is an instant settlement protocol that solves the 'fast-transaction' problem. It functions by using a Directed Acyclic Graph (DAG) where validating nodes verify the sequential ordering of transactions that are received in their memory pools. More specifically, Syscoin approach is centred around the verification of signatures and relaying of transactions. It does this by prioritizing the relay of transactions over signature verification and recipient receives funds faster; see Figure 12. See Appendix A in [2] for the verification process.

A. TRANSMISSION PATH

The expected number of common peers between k Masternodes is:

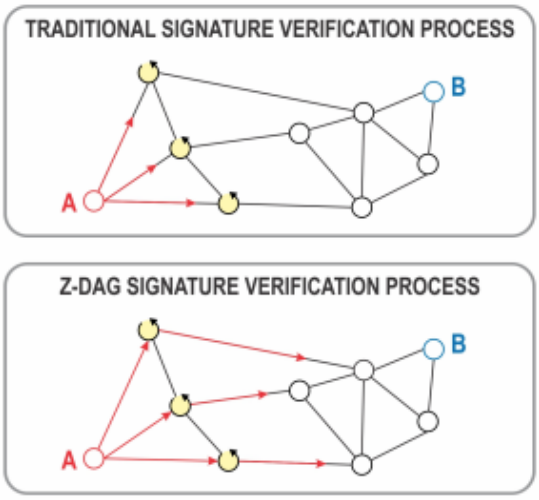

FIGURE 12: Traditional blockchain networks require each node to first check the signatures of incoming transactions before relaying them; this blocking technique bottlenecks broadcasting speed. The Z-DAG process as displayed above immediately relays transactions before checking signatures, resulting in significantly faster movement across the network

$$E\{cp_k\} = pk - m\left(1 - \left(1 - \frac{p}{m}\right)^k\right), \tag{1}$$

where p is the number of common peers, and X_i is 1 when the i^{th} masternode is a common peer and 0 otherwise. The probability of having n common peers is given by:

$$P(|cp| = n) = \frac{\binom{M}{c}\binom{M-c}{p-c}\binom{M-p}{p-c}}{\binom{M}{p}} \tag{2}$$

B. TRANSMISSION DELAY

Using a dataset (containing 2086 samples) of average transmission times of ten transmissions between two hosts, it was modelled using the Gamma distribution as shown in Figure 13. Using this model, the probability of transmission arriving in a given time can be estimated at various time points.

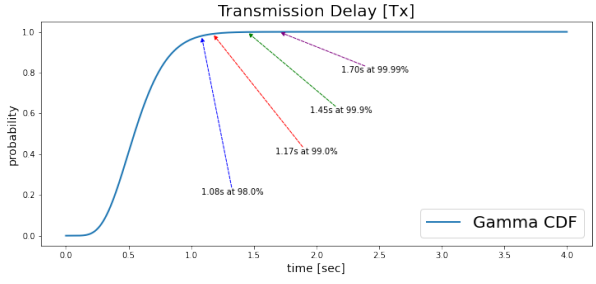

FIGURE 13: Probability of Tx arrival after 5 hops modelled via $X \sim Gamma(7.358, 0.0777)$

APPENDIX C ZERO KNOWLEDGE PROOFS FUNDAMENTALS

The motivation of a Zero Knowledge Proof (ZKP) is to allow for one party (ie, prover) to convince another party (ie, verifier) of facts without revealing information (ie, zero knowledge). An example of a ZKP would be: allowing a subscriber (ie, prover) to gain access to an online service (ie, verifier) without revealing any personal data, other than the fact than that party is a paid subscriber. A ZKP must satisfy three conditions:

1) **Completeness:** If statement is true, verifier will be convinced by prover
2) **Soundness:** If statement is false, a cheating prover cannot (except with some small probability) convince verifier it is true
3) **Zero-knowledge:** If statement is true, verifier knows nothing else

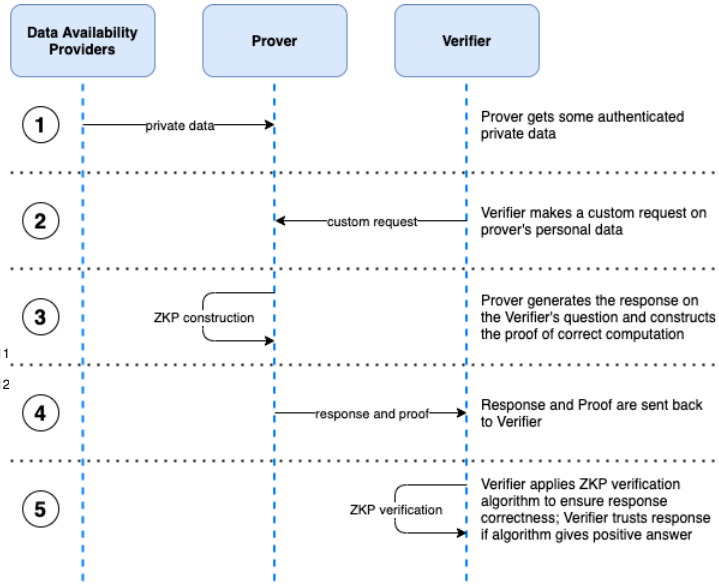

FIGURE 14: Data exchange in a Zero-knowledge proof system

There are two types of ZKP:

1) **Interactive ZKP:** Comprised of multiple challenge / response messages (commitment, challenge and response); requires a stable communication channel
2) **Non-interactive ZKP:** Only needs single message and is more efficient; can be run offline

zk-SNARK and zk-STARKS are types of Non-interactive ZKP , where no interaction is required between prover and verifier.

APPENDIX D BLOCK OCCUPANCY COST (BLOC) FUNCTION

We considered two approaches to model the gas costs, $y_{c,k}$ at the k^{th} block, using model 1:

$$\mathbf{y}_{c,k} = \frac{1}{\gamma}\log\left(\frac{1 + \mathbf{x}_{o,k-1}}{1 - \mathbf{x}_{o,k-1}}\right) + \mathbf{y}_{c,0}; \tag{3}$$

and model 2:

$$\mathbf{y}_{c,k} = \left| \frac{1}{\gamma(\mathbf{x}_{o,k-1} - 1)} \right| + y_{c,0} - \frac{1}{\gamma}, \tag{4}$$

where $x_{o,k} \in [0,1] \forall k$ is the block occupancy, $y_{c,0}$ is the base cost, and γ is an adjustable scaling term. We call y_c the BLock Occupancy Cost (BLOC) function; see Figure 15.

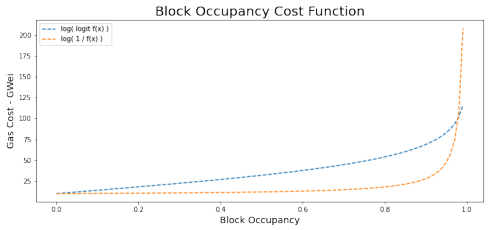

FIGURE 15: BLock Occupancy Cost (BLOC) functions; model 1 and 2

Other variants of $\mathbf{y}_c$ can be utilized highlighting various characteristics. For instance, Model 1 has the characteristic of being less volatile, but with a higher expected base cost than Model 2.

APPENDIX E BLOCK OCCUPANCY AND GAS SIMULATIONS

To simulate arbitrary block occupancies, we use the Beta distribution:

$$\mathbf{x}_{o,t} \sim Beta(\alpha_t, \beta_t) \tag{5}$$

where t is time, and $\beta_t = 1 - \alpha_t$

To exhibit slow upward growth over time for α_t, we used a MinMax scaled ARIMA(0,1,1) model, otherwise known as a simple exponential smoothing model with growth, which is of the form:

$$\hat{\alpha'}_t = \mu + \alpha'_{t-1} - \theta_1 \epsilon_t, \tag{6}$$

and then applied MinMax scaling to ensure $\alpha_t \in [0,1]$:

$$\hat{\alpha_t} = \frac{\alpha'_t - min(\alpha'_t)}{max(\alpha'_t) - min(\alpha'_t)}. \tag{7}$$

Block Occupancy simulations, $\mathbf{x}_{o,t}$, were fed into the BLOC function, to achieve a set of gas simulations, $\mathbf{y}_{c,t}$, at each instance of time t.

APPENDIX F BLOCK RESIZE ON FAILURE MODEL

This Block Resize on Failure model dynamically adjusts the blocksize based on time to last block failure. If the time interval between failures is short then the probability of failure is high and the size for the next block gets decreased. However, when the time interval between failures is long then the probability of failure is low and the size for the next block size gets increased. This methodology is updated upon each failure, and consists of three main steps which include:

- Generate a Kaplan Meier estimator based on last N failures
- Using time interval between last failure get estimated probability of failure using Kaplan Meier estimator
- Feed the estimated probability of failure into a block size transfer function to get the block size adjustment for the next block

The goal of the Kaplan Meier estimator is to estimate the survival function, S, defined as:

$$S(t) = Pr(\tau > t), \tag{8}$$

where t is time, and $\tau > 0$ be a random variable. The survivor function for the Kaplan Meier estimator function is given by:

$$\hat{S}(t) = \prod_{i;t_i<t} 1 - \frac{d_i}{n_i} \tag{9}$$

where t_i is a time when at least one black failure happened, d_i the number of failures, and n_i the blocks that have not failed.

The block size transfer function constructed from the following logit function:

$$y_{tr} = -2 * A * \frac{e^{k(x-x_0)}}{1 + e^{k(x-x_0)}} + A \tag{10}$$

where preconfigured parameters A, k determine how aggressive the resizing is upon each block failure. These parameters are typically determined aprior based on simulations run on the test net.

APPENDIX G EIP-1559

EIP-1559 is a new proposed pricing mechanism for the Ethereum protocol that includes a stationary network fee during states of no congestion which dynamically adjusts during states of congestion. The original gas fee model uses a auction system where miners choose transactions with the highest bids. This has led to several inefficiencies such as instability of blockchains with no reward, needless delays, fee overpayment, and mismatch fee levels between volatility ad social cost.

The basic premise of the new proposal begins with setting the base fee which increases when the network capacity exceeds the target per-block gas usage, and decreases when the capacity is below the target. The calculation of the updated basefee, b_{k+1} at the $k + 1^{th}$ block, is as follows:

$$b_{k+1} = b_k f_k, \tag{11}$$

$$f_k = 1 + \frac{\delta_k}{c} \tag{12}$$

where $\delta_k = usage_k - target\ gas\ fee$, and $c = (target) * (basefee\ max\ change)$.

Ideally, we would like to see b_k behave as a stationary process for states of congestion and non-congestion. A stationary process is when b_k and b_{k+N} have the same probability distributions for all N. Hence, this research problem can be broken into these two main categories. We foresee the latter problem being significantly more difficult to mathematically

model than the former, as there are many non-stationary edge-cases that can arise from various attacks which have not yet been defined. For the sake of brevity, we assume δ_k to be a stationary Gaussian process in this discussion.

It is quite possible that this problem can be addressed using a state space model or or some other kind of probabilistic graphical model. Approaching the problem this way could be an effective way of optimally ensuring (under predefined assumptions) that the statistics of δ_k stay relatively stable over k, and a good way to mathematically define various attack vectors.

APPENDIX H GAUSSIAN PROCESSES

Assume observed data, y, are the sum of a Gaussian Process (GP) and Gaussian noise, given by:

$$\mathbf{y} = f(x) + \epsilon \tag{13}$$

$$\epsilon \sim N(0, \Sigma) \tag{14}$$

The unknown function, $f(x)$, is modelled as

$$f(x) \sim GP(\mu(x), k(x, x')), \tag{15}$$

where $\mu(x)$ and $k(x, x')$ are the mean and covariance function respectively; the covariance function is often called the kernel function, and can take on several forms based on the structure of the input data, x. In this work, we use the Matern 5/2 kernel, which is of the form:

$$k(x, x') = \left(1 + \frac{\sqrt{g(x, x')}}{\ell} + \frac{g(x, x')}{3\ell^2}\right) \exp\left[\frac{g(x, x')}{\ell}\right] \tag{16}$$

where

$$g(x, x') = 5(x - x')^2. \tag{17}$$

For this work, we looked at observations, $\mathbf{y}$, probabilistically, which can be determined via:

$$p(y|x) = \int p(y|f, x)p(f, |x)df; \tag{18}$$

this is otherwise known as the marginal likelihood; from a Bayesian context, this may be referred to as the evidence, which is the normalizing part of Bayes rule. The marginal likelihood can be utilized to generate conditional distribution for Gaussians to get the posterior distribution of $f(x*)$ given $\mathbf{y}$. To do this we used Markov Chain Monte Carlo (MCMC) sampling using the PyMC3 python package; for more detail along with coding examples, see [44].

REFERENCES

[1] J. Sidhu, *Syscoin 3.0: A Peer-to-Peer Electronic Cash System Built For Business Applications*, Blockchain Foundry Inc, Feb. 2018. Accessed on: Dec 2020. [Online]. Available: https://syscoin.org/$syscoin3_whitepaper.pdf$

[2] J. Sidhu, E, Scott, and A. Gabriel, *Z-DAG: An interactive DAG protocol for real-time crypto payments with Nakamoto consensus security parameters*, Blockchain Foundry Inc, Feb. 2018. Accessed on: Dec 2020. [Online]. Available: https://syscoin.org/$syscoin3_whitepaper.pdf$

[3] V. Buterin, *Blockchain Resource Pricing*, Apr 2019. Accessed on: May 2021. [Online]. Available: https://github.com/ethereum/research/blob/master/papers/pricing/ethpricing.pdf

[4] Y. Sompolinsky, and A. Zohar, *Secure High-rate Transaction Processing in Bitcoin*, Proc. 19th Int. Conf. Financial Cryptogr, Data Secur. (FC'20), Jan 2015, pp. 507-527

[5] G. Birmpas, E. Koutsoupias, P. Lazos, F.J. Marmolejo-Cossío, *Fairness and Efficiency in DAG-Based Cryptocurrencies*, Proc. 24th Int. Conf. Financial Cryptogr, Data Secur. (FC'15), Jan 2015, pp. 507-527

[6] Eric Brewer, *CAP twelve years later: How the 'rules' have changed*, Computer, Volume 45, Issue 2 (2012), pg. 23–29

[7] N. Word, *UTXO vs Global State*, Nov. 2019. Accessed on: Jan 2021. [Online]. Available: https://word.site/2019/11/20/utxo-vs-global-state/

[8] P. Sztorc *Nothing is Cheaper than Proof of Work*, Aug. 2015. Accessed on: Jan 2021. [Online]. Available: https://www.truthcoin.info/blog/pow-cheapest/

[9] I. Eyal, and E.G. Sirer, *Majority is not Enough: Bitcoin Mining is Vulnerable*, Communications of the ACM., vol. 61, no. 7, Jun. 2018.

[10] B. Cao, Z. Zhang, D. Feng, S. Zhang, L. Zhang, M. Peng, and Y Li, *Performance analysis and comparison of PoW, PoS and DAG based blockchains*, Digital Communications and Networks., vol. 6, no. 4, Nov. 2020, pp 480-485

[11] V. Buterin, *Using polynomial commitments to replace state roots*, Ethereum Research, Mar. 2020. Accessed on: Jan 2021. [Online]. Available: https://ethresear.ch/t/using-polynomial-commitments-to-replace-state-roots/7095

[12] G., *Proof of Stake Versus Proof of Work. Technical Report*, BitFury Group, 2015. Accessed on: Jan 2021. [Online]. Available: http://bitfury.com/content/5-white-papers-research/pos-vs-pow-1.0.2.pdf

[13] E. Duffield, and D. Diaz, *Dash: A Payments-Focused Cryptocurrency*, Dash, Aug 2018. Accessed on: Jan 2021. [Online]. Available: https://github.com/dashpay/dash/wiki/Whitepaper

[14] *Bitcoin Core FAQ, Compact Blocks FAQ*, Accessed on: Feb 2021. [Online]. Available: https://bitcoincore.org/en/2016/06/07/compact-blocks-faq/

[15]] A. Block, *Mitigating 51% attacks with LLMQ-based ChainLocks*. Accessed on: Feb 2021. [Online], Nov 2018. Available: https://blog.dash.org/mitigating-51-attacks-with-llmq-based-chainlocks-7266aa648ec9

[16] J. Valenzuela, Andreas Antonopoulos, *Calls Dash Chain Locks "a Smart Way of" Preventing 51% Attacks*. Aug 22, 2019. Accessed on: Feb 2021. [Online]. Available: https://dashnews.org/andreas-antonopoulos-calls-dash-chainlocks-a-smart-way-of-preventing-51-attacks/

[17] D. Boneh, M. Drijvers, and G. Neven, *BLS Multi-Signatures With Public-Key Aggregation*, Mar 2018. Accessed on: Feb 2021. [Online]. Available: https://crypto.stanford.edu/ dabo/pubs/papers/BLSmultisig.html

[18] J. Drake. *Pragmatic signature aggregation with BLS*, May 2018. Accessed on: Feb 2021. [Online]. Available: https://ethresear.ch/t/pragmatic-signature-aggregation-with-bls/2105

[19] S. Bowe, *BLS12-381: New zk-SNARK Elliptic Curve Construction*, Mar 2017. Accessed on: Feb 2021. [Online]. Available: https://electriccoin.co/blog/new-snark-curve/

[20] A. Block, *BLS: Is it really that slow?*, Jul 2018. Accessed on: Feb 2021. [Online]. Available: https://blog.dash.org/bls-is-it-really-that-slow-4ca8c1fcd38e

[21] S. de la Rouvier. *Interplanetary Linked Computing: Separating Merkle Computing from Blockchain Computational Courts*, Jan 2017. Accessed on: Feb 2021. [Online]. Available: https://media.consensys.net/interplanetary-linked-computing-separating-merkle-computing-from-blockchain-computational-courts-1ade201ecf8a

[22] Anonymous Kid, Why the fuck did Satoshi implement the 1 MB block-size limit? [Online forum comment], Jan 2018, Accessed on: Feb 2021. [Online]. Available: https://bitcointalk.org/index.php?topic=2786690.0

[23] *Zero-Knowledge Proofs What are they, how do they work, and are they fast yet?* Accessed on: Feb 2021. [Online]. Available: https://zkp.science/

[24] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, *Scalable, transparent, and post-quantum secure computational integrity*, IACR Cryptol, 2018, pp 46

[25] Dryja, T, Utreexo: A dynamic hash-based accumulator optimized for the bitcoin UTXO set, IACR Cryptol. ePrint Arch., 2019, p. 611.

[26] G.I. Hotchkiss, *The 1.x Files: The State of Stateless Ethereum*, Dec 2019. Accessed on: Feb 2021. [Online]. Available: https://blog.ethereum.org/2019/12/30/eth1x-files-state-of-stateless-ethereum/

[27] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, H. Wu: *Zexe: Enabling decentralized private computation*. Cryptology ePrint Archive, Report 2018/962 (2018). Accessed on: Feb 2021. [Online]. Available: https://par.nsf.gov/servlets/purl/10175111

[28] A. Nilsson, P.N. Bideh, J. Brorsson, *A survey of published attacks on Intel SGX*. 2020, arXiv:2006.13598

[29] C. Nelson, *Zero-Knowledge Proofs: Privacy-Preserving Digital Identity*, Oct 2018. Feb 2021. Accessed on: [Online]. Available: https://www.slideshare.net/SSIMeetup/zeroknowledge-proofs-privacypreserving-digital-identity-with-clare-nelson

[30] D. Boneh, *Discrete Log based Zero-Knowledge Proofs*, Apr 2019, Accessed on: Feb 2021 [Online]. Available: https://www.youtube.com/watch?v=wB3DlND7KEw

[31] *Quantum Computing's Implications for Cryptography (Chapter 4)*, National Academies of Sciences, Engineering, and Medicine: Quantum Computing: Progress and Prospects. The National Academies Press, Washington, DC, 2018.

[32] S. Naihin, *Goodbye Bitcoin… Hello Quantum*, Apr 2019, Accessed on: Feb 2021 [Online]. https://www.linkedin.com/pulse/goodbye-bitcoin-helloquantum-silen-naihin

[33] L.T. do Nascimento, S. Kumari, and V. Ganesan, *Zero Knowledge Proofs Applied to Auctions*, May 2019, Accessed on: Feb 2021 [Online]. Available: https://courses.csail.mit.edu/6.857/2019/project/18-doNascimento-Kumari-Ganesan.pdf

[34] V. Buterin and V. Griffith, Casper the Friendly Finality Gadget. CoRR, Vol. abs/1710.09437, 2017. arxiv: 1710.09437, http://arxiv.org/abs/1710.09437

[35] M. Neuder, D.J. Moroz, R. Rao, and D.C. Parkes, *Low-cost attacks on Ethereum 2.0 by sub-1/3 stakeholders*, 2021. arXiv:2102.02247, https://arxiv.org/abs/2102.02247

[36] Starkware, *Validity Proofs vs. Fraud Proofs*, Jan 2019, Accessed on: Feb 2021, [Online]. Available: https://medium.com/starkware/validity-proofs-vs-fraud-proofs-4ef8b4d3d87a

[37] A. Gluchowski, *World's first practical hardware for zero-knowledge proofs acceleration*, Jul 2020, Accessed on: Feb 2021 [Online]. Available: https://medium.com/matter-labs/worlds-first-practical-hardware-for-zero-knowledge-proofs-acceleration-72bf974f8d6e

[38] *Introducing an NFT Platform Like No Other*, Accessed on: Feb 2021. [Online]. Available: https://syscoin.org/news/introducing-an-nft-platform-like-no-other

[39] A. Bhuptani, *Vector 0.1.0 Mainnet Release, The beginning of a multi-chain Ethereum ecosystem*, Jan 2021, Accessed on: Feb 2021. [Online]. Available: https://medium.com/connext/vector-0-1-0-mainnet-release-9496ae52c422

[40] V. Buterin, *With fraud-proof-free data availability proofs, we can have scalable data chains without committees*, Jan 2020, Accessed on: Feb 2021. [Online]. Available: https://ethresear.ch/t/with-fraud-proof-free-data-availability-proofs-we-can-have-scalable-data-chains-without-committees/6725

[41] M. Al-Bassam, *A data availability blockchain with sub-linear full block validation*, Jan 2020, Accessed on: Feb 2021. [Online]. Available: https://ethresear.ch/t/with-fraud-proof-free-data-availability-proofs-we-can-have-scalable-data-chains-without-committees/6725

[42] T. Chen, X. Li, Y. Wang, J. Chen, Z Li, X. Luo, M. H. Au, and X. Zhang. *An adaptive gas cost mechanism for Ethereum to defend against under-priced DoS attacks*. Proceedings of Information Security Practice and Experience - 13th International Conference ISPEC, 2017

[43] Starkware Team, Rescue STARK Documentation – Version 1.0, Jul 2020

[44] C. Fonnesbeck, *Advanced Statistical Computing, Bios 8366*, Vanderbilt University's Department of Biostatistics, Nashville, TN, May 2020, , Accessed on: Mar 2021. [Online]. Available: https://github.com/fonnesbeck/Bios8366/blob/master/notebooks/Section5_1-Gaussian-Processes.ipynb

[45] I. Moore and J. Sidhu. *Stochastic Properties of EIP 1559 Basefees*, 2021. arXiv:2105.03521, https://arxiv.org/abs/2105.03521