

Sui-Related Research Papers

This document contains a list of research papers that are relevant to Sui and that one or more Sui team members have co-authored. Some of the ideas of these papers are currently being integrated into Sui, others are in our roadmap, and others are not in our roadmap but could be integrated in the future. Start with the [Sui Smart Contract Platform](#) white paper, which contains our latest design inspired by previous works below.

Latest

Mysticeti: Reaching the Limits of Latency with Uncertified DAGs

- **Link:** <https://arxiv.org/pdf/2310.14821>
- **Publication:** Network and Distributed System Security Symposium (NDSS), accepted for 2025
- **Relevance:** We introduce Mysticeti-C, the first DAG-based Byzantine consensus protocol to achieve the lower bounds of latency of 3 message rounds. Since Mysticeti-C is built over DAGs it also achieves high resource efficiency and censorship resistance. Mysticeti-C achieves this latency improvement by avoiding explicit certification of the DAG blocks and by proposing a novel commit rule such that every block can be committed without delays, resulting in optimal latency in the steady state and under crash failures. We further extend Mysticeti-C to Mysticeti-FPC, which incorporates a fast commit path that achieves even lower latency for transferring assets. Unlike prior fast commit path protocols, Mysticeti-FPC minimizes the number of signatures and messages by weaving the fast path transactions into the DAG. This frees up resources, which subsequently result in better performance. We prove the safety and liveness in a Byzantine context. We evaluate both Mysticeti protocols and compare them with state-of-the-art consensus and fast path protocols to demonstrate their low latency and resource efficiency, as well as their more graceful degradation under crash failures. Mysticeti-C is the first Byzantine consensus protocol to achieve WAN latency of 0.5s for consensus commit while simultaneously maintaining state-of-the-art throughput of over 200k TPS. Finally, we report

on integrating Mysticeti-C as the consensus protocol into the Sui blockchain, resulting in over 4x latency reduction.

Sui Lutris: A Blockchain Combining Broadcast and Consensus

- **Link:** <https://arxiv.org/pdf/2310.18042>
- **Publication:** Conference on Computer and Communications Security (CCS), 2024
- **Relevance:** Sui Lutris is the first smart-contract platform to sustainably achieve sub-second finality. It achieves this significant decrease by employing consensusless agreement not only for simple payments but for a large variety of transactions. Unlike prior work, Sui Lutris neither compromises expressiveness nor throughput and can run perpetually without restarts. Sui Lutris achieves this by safely integrating consensusless agreement with a high-throughput consensus protocol that is invoked out of the critical finality path but ensures that when a transaction is at risk of inconsistent concurrent accesses, its settlement is delayed until the total ordering is resolved. Building such a hybrid architecture is especially delicate during reconfiguration events, where the system needs to preserve the safety of the consensusless path without compromising the long-term liveness of potentially misconfigured clients. We thus develop a novel reconfiguration protocol, the first to provably show the safe and efficient reconfiguration of a consensusless blockchain. Sui Lutris is currently running in production and underpins the Sui smart-contract platform. Combined with the use of Objects instead of accounts it enables the safe execution of smart contracts that expose objects as a first-class resource. In our experiments Sui Lutris achieves latency lower than 0.5 seconds for throughput up to 5,000 certificates per second (150k ops/s with transaction blocks), compared to the state-of-the-art real-world consensus latencies of 3 seconds. Furthermore, it gracefully handles validators crash-recovery and does not suffer visible performance degradation during reconfiguration.

zkLogin: Privacy-Preserving Blockchain Authentication with Existing Credentials

- **Link:** <https://arxiv.org/pdf/2401.11735>

- **Publication:** Not published
- **Relevance:** For many users, a private key based wallet serves as the primary entry point to blockchains. Commonly recommended wallet authentication methods, such as mnemonics or hardware wallets, can be cumbersome. This difficulty in user onboarding has significantly hindered the adoption of blockchain-based applications. We develop zkLogin, a novel technique that leverages identity tokens issued by popular platforms (any OpenID Connect enabled platform such as Google, Facebook, and other platforms) to authenticate transactions. At the heart of zkLogin lies a signature scheme allowing the signer to sign using their existing OpenID accounts and nothing else. This improves the user experience significantly as users do not need to remember a new secret and can reuse their existing accounts. zkLogin provides strong security and privacy guarantees. By design, zkLogin builds on top of the underlying platform's authentication mechanisms, and derives its security from there. Unlike prior related works however, zkLogin avoids the use of additional trusted parties (such as trusted hardware or oracles) for its security guarantees. zkLogin leverages zero-knowledge proofs (ZKP) to ensure that the link between a user's offchain and onchain identities is hidden, even from the platform itself. zkLogin is implemented and deployed on the Sui blockchain as an alternative to traditional digital signature-based addresses. Due to the ease of Web3 on-boarding just with social login, without requiring mnemonics, many hundreds of thousands zkLogin accounts have already been generated in various industries such as gaming, DeFi, direct payments, NFT collections, ride sharing, sports racing and many more.

Be Aware of Your Leaders

- **Link:** <https://arxiv.org/abs/2110.00960>
- **Publication:** Financial Cryptography and Data Security (FC), 2022
- **Relevance:** Provides a performant leader election algorithm for partially-synchronous consensus protocols such as Bullshark. This work informed earlier designs for shared [object](#) support in Sui before the adoption of Mysticeti.
- **Summary:** Advances in blockchains have influenced the State-Machine-Replication (SMR) world and many state-of-the-art blockchain-SMR solutions are based on two pillars: Chaining and Leader-rotation. A predetermined round-robin mechanism used for Leader-rotation, however, has an undesirable behavior: crashed parties become designated leaders infinitely often, slowing down overall system performance. In this paper, we provide a new Leader-Aware SMR framework that, among other desirable properties,

formalizes a Leader-utilization requirement that bounds the number of rounds whose leaders are faulty in crash-only executions. We introduce Carousel, a novel, reputation-based Leader-rotation solution to achieve Leader-Aware SMR. The challenge in adaptive Leader-rotation is that it cannot rely on consensus to determine a leader, since consensus itself needs a leader. Carousel uses the available onchain information to determine a leader locally and achieves Liveness despite this difficulty. A HotStuff implementation fitted with Carousel demonstrates drastic performance improvements: it increases throughput over 2x in faultless settings and provides a 20x throughput increase and 5x latency reduction in the presence of faults.

Bullshark: DAG BFT Protocols Made Practical

- **Link:** <https://arxiv.org/abs/2201.05677>
- **Publication:** Not published yet (under submission)
- **Relevance:** Provides a partially-synchronous consensus protocol running over Narwhal. Bullshark was a predecessor to Mysticeti and informed the consensus design of Sui before Mysticeti was adopted.
- **Summary:** We present Bullshark, the first directed acyclic graph (DAG) based Byzantine Fault Tolerant (BFT) protocol that is optimized for partial synchrony. Bullshark inherits all the desired properties of its predecessor (DAG-Rider) such as optimal amortized complexity, asynchronous liveness, zero-overhead, and post-quantum safety; but at the same time Bullshark provides a practical low-latency fast-path that exploits synchronous periods. In addition, we introduce a standalone partially synchronous version of Bullshark and evaluate it against the state of the art. The resulting protocol is embarrassingly simple 20 LOC on top of a DAG-based mempool implementation) and highly efficient, achieving for example, 125k transactions per second and 2 seconds latency with 50 nodes.

FastPay: High-Performance Byzantine Fault Tolerant Settlement

- **Link:** <https://arxiv.org/abs/2003.11506>

- **Publication:** ACM Conference on Advances in Financial Technologies (AFT), 2020
- **Relevance:** FastPay describes the core protocol at the heart of Sui.
- **Summary:** FastPay allows a set of distributed validators, some of which are Byzantine, to maintain a high-integrity and availability settlement system for pre-funded payments. It can be used to settle payments in a native unit of value (crypto-currency), or as a financial side-infrastructure to support retail payments in fiat currencies. This is not the protocol Sui uses, yet it proposes the basic safety mechanism that Sui extends. FastPay is based on Byzantine Consistent Broadcast as its core primitive, foregoing the expenses of full atomic commit channels (consensus). The resulting system has low-latency for both confirmation and payment finality. Remarkably, each validator can be sharded across many machines to allow unbounded horizontal scalability.

HammerHead: Score-based Dynamic Leader Selection

- **Link:** <https://arxiv.org/pdf/2309.12713>
- **Publication:** IEEE International Conference on Distributed Computing Systems (ICDCS), 2024
- **Relevance:** The need for high throughput and censorship resistance in blockchain technology has led to research on DAG-based consensus. This paper explores ideas pioneered by Carousel on providing Leader-Utilization and presents HammerHead as an improvement over earlier leader-based DAG consensus protocols such as Bullshark. Unlike Carousel, which is built with a chained and pipelined consensus protocol in mind, HammerHead does not need to worry about chain quality as it is directly provided by the DAG, but needs to make sure that even though validators might commit blocks in different views the safety and liveness is preserved. Our implementation of HammerHead shows a slight performance increase in a faultless setting, and a drastic 2x latency reduction and up to 40% throughput increase when suffering faults (100 validators, 33 faults).

Narwhal and Tusk: A DAG-based Mempool and Efficient BFT Consensus

- **Link:** <https://arxiv.org/abs/2105.11827>
- **Publication:** EuroSys, 2022
- **Relevance:** Narwhal and Tusk provided the foundational DAG-based mempool and consensus design that informed earlier versions of Sui's shared-object support. Sui now uses Mysticeti as its consensus protocol.
- **Summary:** We propose separating the task of reliable transaction dissemination from transaction ordering to enable high-performance Byzantine fault-tolerant quorum-based consensus. We design and evaluate a mempool protocol, Narwhal, specializing in high-throughput reliable dissemination and storage of causal histories of transactions. Narwhal tolerates an asynchronous network and maintains high performance despite failures. Narwhal is designed to easily scale-out using multiple workers at each validator, and we demonstrate that there is no foreseeable limit to the throughput we can achieve. Composing Narwhal with a partially synchronous consensus protocol (Narwhal-HotStuff) yields significantly better throughput even in the presence of faults or intermittent loss of liveness due to asynchrony. However, loss of liveness can result in higher latency. To achieve overall good performance when faults occur we design Tusk, a zero-message overhead asynchronous consensus protocol, to work with Narwhal. We demonstrate its high performance under a variety of configurations and faults. As a summary of results, on a WAN, Narwhal-Hotstuff achieves more than 130,000 tx/sec at less than 2-sec latency compared with 1,800 tx/sec at 1-sec latency for Hotstuff. Additional workers increase throughput linearly to 600,000 tx/sec without any latency increase. Tusk achieves 160,000 tx/sec with about 3 seconds latency. Under faults, both protocols maintain high throughput, but Narwhal-HotStuff suffers from increased latency.

SybilQuorum: Open Distributed Ledgers Through Trust Networks

- **Link:** <https://arxiv.org/abs/1906.12237>
- **Publication:** Not published
- **Relevance:** Less related to Sui than the other papers, and the paper is in its early stages. It presents an algorithm to strengthen proof-of-Stake systems (like Sui). The paper is, however, theoretical and not on our roadmap.
- **Summary:** The Sybil attack plagues all peer-to-peer systems, and modern open distributed ledgers employ a number of tactics to prevent it from proof of work, or other resources such as space, stake or memory, to traditional

admission control in permissioned settings. With SybilQuorum we propose an alternative approach to securing an open distributed ledger against Sybil attacks, and ensuring consensus amongst honest participants, leveraging social network based Sybil defenses. We show how nodes expressing their trust relationships through the ledger can bootstrap and operate a value system, and general transaction system, and how Sybil attacks are thwarted. We empirically evaluate our system as a secure Federated Byzantine Agreement System, and extend the theory of those systems to do so.

Twins: BFT Systems Made Robust

- **Link:** <https://arxiv.org/abs/2004.10617>
- **Publication:** International Conference on Principles of Distributed Systems (OPODIS), 2021
- **Relevance:** Less related to Sui than the other papers, this provides a way to test implementations of consensus systems, such as Tusk and Bullshark. The paper is, however, theoretical and not on our roadmap.
- **Summary:** This paper presents Twins, an automated unit test generator of Byzantine attacks. Twins implements three types of Byzantine behaviors: (i) leader equivocation, (ii) double voting, and (iii) losing internal state such as forgetting 'locks' guarding voted values. To emulate interesting attacks by a Byzantine node, it instantiates twin copies of the node instead of one, giving both twins the same identities and network credentials. To the rest of the system, the twins appear indistinguishable from a single node behaving in a 'questionable' manner. Twins can systematically generate Byzantine attack scenarios at scale, execute them in a controlled manner, and examine their behavior. Twins scenarios iterate over protocol rounds and vary the communication patterns among nodes. Twins runs in a production setting within DiemBFT where it can execute 44M Twins-generated scenarios daily. Whereas the system at hand did not manifest errors, subtle safety bugs that were deliberately injected for the purpose of validating the implementation of Twins itself were exposed within minutes. Twins can prevent developers from regressing correctness when updating the codebase, introducing new features, or performing routine maintenance tasks. Twins requires only a thin wrapper over DiemBFT; we thus envision other systems using it. Building on this idea, one new attack and several known attacks against other BFT protocols were materialized as Twins scenarios. In all cases, the target protocols break within fewer than a dozen protocol rounds. Hence it is realistic for the Twins approach to expose the problems.

Zef: Low-latency, Scalable, Private Payments

- **Link:** <https://arxiv.org/abs/2201.05671>
- **Publication:** Not published yet (under submission)
- **Relevance:** Extends the FastPay design to support objects (rather than accounts), what Sui actually uses. An additional contribution of this paper is to add strong privacy to FastPay transactions (but Sui does not plan to do this).
- **Summary:** We introduce Zef, the first Byzantine-Fault Tolerant (BFT) protocol to support payments in anonymous digital coins at arbitrary scale. Zef follows the communication and security model of FastPay: both protocols are asynchronous, low-latency, linearly-scalable, and powered by partially-trusted sharded validators. Zef further introduces opaque coins represented as offchain certificates that are bound to user accounts. In order to hide the face values of coins when a payment operation consumes or creates them, Zef uses random commitments and NIZK proofs. Created coins are made unlinkable using the blind and randomizable threshold anonymous credentials of [Coconut](#). To control storage costs associated with coin replay prevention, Zef accounts are designed so that data can be safely removed once an account is deactivated. Besides the specifications and a detailed analysis of the protocol, we are making available an open source implementation of Zef in Rust. Our extensive benchmarks on AWS confirm textbook linear scalability and demonstrate a confirmation time under one second at nominal capacity. Compared to existing anonymous payment systems based on a blockchain, this represents a latency speedup of three orders of magnitude, with no theoretical limit on throughput.



© 2026 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER [CC BY 4.0](#)