IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 July 2022, accepted 8 August 2022, date of publication 18 August 2022, date of current version 9 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3200051

SURVEY

Blockchain Scaling Using Rollups: A Comprehensive Survey

LOUIS TREMBLAY THIBAULT[1], TOM SARRY[1], AND ABDELHAKIM SENHAJI HAFID[2], (Member, IEEE)
[1]Department of Computer Science and Operational Research, University of Montreal, Montreal, QC H3T 1J4, Canada
[2]Montreal Blockchain Laboratory, Department of Computer Science and Operational Research, University of Montreal, Montreal, QC H3T 1J4, Canada
Corresponding author: Louis Tremblay Thibault (louis.tremblay.thibault@umontreal.ca)

ABSTRACT Blockchain systems have seen much growth in recent years due to the immense potential attributed to the technology behind these systems. However, this popularity has outlined a critical scalability issue that most blockchain systems are now confronted with. With their increasing popularity comes an increasing amount of load on the system. Several scaling solutions that modify either the functioning of the underlying protocol or that build on top of them have already been proposed; however, each of these solutions comes with their advantages and disadvantages. This paper aims to survey the current state-of-the-art of rollups as a scaling solution. We discuss the mode of operation of the different types of rollups, outline state-of-the-art implementations of each type together with their features and limitations. We also conduct a performance analysis comparing these implementations. Finally, we outline avenues for future research around rollups as a scaling solution.

INDEX TERMS Blockchain, scalability, rollups, second layer solutions, survey.

I. INTRODUCTION

Blockchain technology has recently gained more and more attraction with its use as a decentralized public ledger [1]. Popular asset exchange systems like Bitcoin use blockchain technology as a way to secure a transaction history between agents by rendering it decentralized and immutable in time. More and more blockchain systems (public as well as private) [2] are emerging in several fields [3], [4]. Even though more and more users are joining these networks, current public systems are still struggling to achieve performance levels that would enable mass adoption [5]. Performance in terms of transactions processed per second (TPS) is the usual metric used to compare different blockchain systems. By design, popular systems are providing low numbers to their users, e.g. up to 7 TPS for the Bitcoin network [6] and around 15 for Ethereum [7], [8]. More recent blockchain systems (e.g., Solana [9], Avalanche [10]) have been able to achieve higher throughput by proposing novel layer one protocols [11]. One of the main reasons behind these low performances is that using proof of work, as a consensus mechanism, implies that each network node processes each transaction. Having several thousands [12] of networks nodes coordinate on the order in which transactions are processed is a difficult task; this reduces the network capacity, along with other limiting factors like having a maximum block size [13].

Scaling blockchain systems has been an open problem since the emergence of these systems, as capacity limitations were bound to happen by design. Recently, coming up with scaling solutions has been considered with utmost urgency; indeed, these capacity limitations have become one of the biggest challenges to overcome in order to reach mass adoption of blockchain and cryptocurrency systems. Several scaling solutions have been proposed by industry and academia members [14]. These solutions can be classified in two categories: first layer and second layer scaling solutions. Layer one solutions provide performance enhancement by changing parameters of the blockchain system (e.g. block size and block time), changing the consensus mechanism, sharding the network into a set of subnetworks or a combination of these. Second layer solutions are solutions that build "on top" of the first layer; they operate outside of the main blockchain but still interact with it (usually in order to ensure security and data availability to guarantee the soundness of the solution).

The associate editor coordinating the review of this manuscript and approving it for publication was Baoping Cai.

This includes state channels, sidechains, Plasma, rollups and validiums.

This paper aims to regroup available information on blockchain scaling solutions and classify them based on their architecture, operation mechanisms and performance. We focus on giving a thorough explanation of rollup technology and explain the ideas behind different types of rollups. Furthermore, we discuss the operation and performances of several rollup implementations and give a detailed comparison and analysis of these implementations. Lastly, we give insight on future research work that could be conducted to advance the state of rollup technology.

We introduce, to the best of our knowledge, the first paper to regroup and explain current state-of-the-art rollup technology. Our paper also provides an in-depth explanation and comparison of different rollup implementations. Thus, our work bridges the current gap in the literature concerning in-depth explanation, comparison and analysis of rollups as a scaling solution.

II. SCALABILITY

There exist several strategies to increase TPS to compete with classic payment systems; however, these strategies often offer a trade-off between the three major properties of blockchain systems, which are decentralization, scalability and security (see Figure 1).

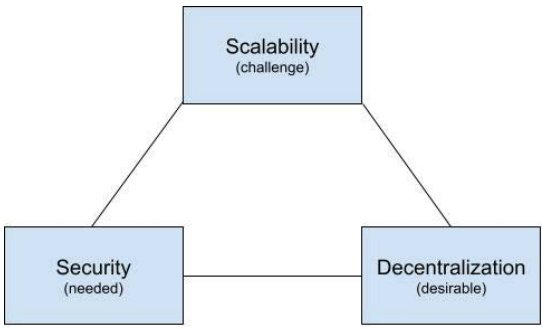

FIGURE 1. The blockchain trilemma.

Indeed, it would be possible to achieve much higher throughput by cutting back on decentralization (e.g., few nodes are involved in the consensus) or security (e.g., vulnerable to attacks) of blockchain systems. It is much more challenging to design a blockchain system that offers scalability (i.e. increased TPS) while maintaining a high level of decentralization and security. In this section, we describe briefly different first and second layer scaling solutions.

A. FIRST LAYER SOLUTIONS

First layer scaling solutions are a modification of the mechanism by which a blockchain system's network handles the distributed blockchain. Several different ideas are currently being studied and implemented. As first layer solutions are out of the scope of this paper, we choose to only present the two most popular, sharding and Proof-of-Stake. Other papers offer an in-depth survey of first layer scaling solutions [6], [14].

1) SHARDING

Sharding is a well-studied concept that was first introduced in database systems in order to increase performance and uptime. The idea behind sharding in a blockchain system is to divide the set of network nodes into subsets called shards (or committees) [15]. In this system, each shard executes a subset of the network transactions. This alleviates the load on network nodes; they are only required to execute and achieve consensus on transactions processed by their committee. Without sharding, network nodes have to execute and achieve consensus on every transaction processed by the network [16]. This system offers parallelism on transaction execution as a processor with several cores would offer parallelism on instructions.

However, sharding comes with its own security risks; malicious actors have the possibility of synchronizing an attack on a specific committee. These actors could successfully perform a 1% attack [17] on a single shard and control transaction execution in this subnetwork. There exist ways to counter such attacks, e.g. assigning nodes to a random shard every certain amount of time. This shuffle of nodes over shards makes it hard for an attacker to gather all its malicious nodes in a single shard and perform the attack. Therefore, the shard takeover attack is practically impossible to execute [18], [19].

Sharded networks often require having a decentralized synchronizing system (e.g., the beacon chain on Ethereum) that manages shards. This implies shuffling nodes every epoch to ensure security and managing cross-shard transactions. Relying on such a system creates a certain overhead in the network; shuffled nodes have to synchronize with their newly assigned shard every shuffling cycle [20], [21]. This overhead is required for a sharded system to behave correctly which enables an increase in transaction throughput.

2) PROOF-OF-STAKE

Proof-of-Stake (PoS) is a consensus mechanism where block creators are selected based on the amount of tokens (or stake) they hold in the system. Whereas Proof-of-Work (PoW) is a consensus mechanism where block creators are selected based on the amount of hash power they possess. [22]. This has several advantages; in particular, it makes blockchain systems much more energy efficient [23] and frees block creators (and therefore the whole blockchain system) from energy costs. Block creators being bounded by energy costs in a system based on an adjusted difficulty Proof-of-Work consensus mechanism can lead to unexpected behavior. For example, if the price of electricity is too high, miners can reduce their mining effort to decrease the block difficulty. The game theoretic strategies at play can lead to inconsistency in block time in a Proof-of-Work network [24]. Proof-of-Stake systems avoid these problems by dissociating block creation from electrical power.

PoS can suffer from attacks in which block creators work on several chain forks at the same time. These forks can be

incoherent; for example, two conflicting transactions (e.g., same UTXOs or same nonce) from the same sender may be included in different forks. It is feasible for block creators to work on several different forks since working in a fork does not require intensive computing power. This means that these block creators can reap benefits from creating blocks while hindering the system from achieving consensus. This is commonly referred to the ''nothing at stake'' attack. There are some proposals to counter this type of attack. For example, it is possible to slash the stake of a block creator that has approved several conflicting blocks [25]. It is worth noting that the Ethereum system intends to switch to a PoS scheme in the future.

B. SECOND LAYER SOLUTIONS

Second layer scaling solutions build on top of the first layer chain. They only interact with the layer one chain in order to publish some data that needs to be secured and made always available. In other words, they rely on the consensus mechanism of the layer one network for security. Second layer solutions do not require changes to the first layer mechanism, i.e. they are an add-on. These solutions can be classified into four categories; state channels (e.g., Lightning Network [26]), sidechains (e.g., Liquid Network [27]), Plasma [28], rollups, and validiums (e.g., zkPorter [29]).

1) STATE CHANNELS

The performance problems regarding blockchain systems are in part due to the fact that every change in the state of a blockchain system needs to be agreed upon by all network nodes. State channels reduce load on the network by bypassing this mechanism. First of all, users need to open a channel by transacting with the blockchain. While the channel is open, the users involved in the channel creation can transact freely off-chain. Once the users are ready to settle, they transact once more with the blockchain to close the channel. Figure 2 shows how actors interact in a state channel system.

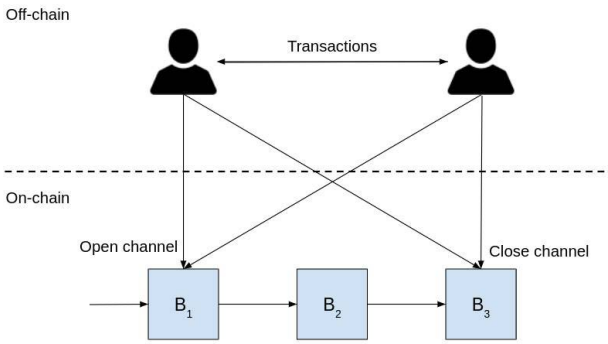

FIGURE 2. Interactions between actors in state channels.

State channels ensure that an arbitrary number of state transitions can happen between users off-chain while only having two transactions finalized on-chain [30]. In particular, payment channels are channels that restrict state transitions to payment transactions, i.e. that only allow changes of the state of an account balance (or UTXOs associated to an address). Providing a secondary network for payments between two actors allows for very small fees and delays before payment finalization. The small fees come from the fact that not every transaction needs to go through layer one. Overall, the user experience when using this solution is more similar to classic payment systems, i.e. there is no need to wait long delays before being certain that the transaction is secured by the network [26].

In state channels, users must be online at all times to be able to distribute transactions through the network and signal misbehavior from other actors (e.g., an actor unilaterally attempts to close a channel). Therefore, there is an assumption that all actors always *live* on the network.

2) SIDECHAINS

Sidechains (e.g., Polygon PoS) are secondary chains that operate in parallel to a parent chain with their own consensus mechanism; they are connected to the parent chain by a two-way bridge. The fact that sidechains are employing a different consensus mechanism (e.g., Proof-of-Stake) means that there is leeway for innovation and performance. This also means that security is not guaranteed by the layer one protocol [31].

Sidechains are pegged to a main blockchain system. Tokens from the main blockchain network can be transferred to the sidechain via a two-way peg. In fact, users can deposit (lock) their tokens by sending a transaction to a smart contract on the main blockchain. They are then converted to tokens that live on the sidechain. Users can then transact on the sidechain with these tokens, and transfer them back to main chain tokens if they want to.

This interoperability is a key functionality of sidechains. This way, a sidechain could be used to execute recurrent transactions (e.g. minting tokens) that do not necessarily need to go through the layer one consensus mechanism. Then, the previously minted tokens can be transferred back to the main chain.

It is worth noting that sidechains do not rely on their main chain for security or consensus; they only rely on the main chain for asset transfer. Therefore, no data critical to the well-functioning of the sidechain is published on the main chain.

3) PLASMA

A plasma system [28] is a separate blockchain that is pegged to a main blockchain. Plasma systems run independently from the main chain (i.e., as sidechains do) but publish data to the main chain if it is needed. The published data usually consists of data that needs to be secured by the layer one network or that needs to always be available (e.g., the plasma block headers). The original idea behind Plasma chains comes from the fact that layer one blockchains would ideally not store smart contract code for sophisticated systems. This would bloat the layer one chain and make it harder to operate. Thus, it is better practice to create another chain (Plasma) and have the sophisticated software live on this network.

Plasma chains operate in an optimistic manner where transactions are supposed to be correct until an agent, called a watcher, observes a fraudulent transaction and calls for a dispute resolution. In this case, the network achieves consensus on the faulty actor (the transaction publisher or the watcher) and penalizes them. This optimistic mechanism, while being very efficient most of the time, comes with an important drawback: there needs to be a period of time during which watchers must be able to call for a dispute resolution. This necessary period increases the time needed for a transaction to achieve finality.

Plasma works with an optimistic system; it is assumed that at least one watcher is active at every moment to ensure transactions published to the first layer are not fraudulent. As with state channels, there is an assumption about the liveness of certain actors; at least one watcher needs to be live for the system to be secure.

4) ROLLUPS

Rollups involve a secondary blockchain network that is pegged to a main blockchain system. In a rollup system, transaction execution is moved to layer two, but all the data from these transactions executed on layer two is published on the main chain. This makes it so all of the transactions that have been processed by the layer two blockchain are available on the layer one blockchain. This data availability provides the possibility to re-execute all the transactions of the system, if need be. Different types of rollups are discussed in-depth in Section III.

Optimistic rollups require watchers to function properly. Therefore, there is a liveness assumption, i.e. it is assumed that there is at least one active watcher at every moment.

5) VALIDIUMS

Validiums are novel systems similar to rollups, but differ in their data availability mechanism. In rollups, the transaction data as well as a the hash of the state root is published to the base layer (on-chain) for security. In contrast, validiums store this data off-chain; the availability of such data can be regulated by a data availability committee, including trusted actors like centralized oracles, for instance.

By avoiding the need to go through layer one, validiums offer fees that are not tied to the main chain. They also provide a much bigger throughput. However, validiums do not rely on the security of layer one. If the trusted data availability committee becomes dishonest, it can choose to not provide data when a network node requires it. Since this committee is the only actor that saves the transaction data, a corrupt committee could lead to redacted transactions.

Since a few important actors in the rollups ecosystem also develop validiums, it will be interesting to present a brief overview of these solutions and compare performances.

Table 1 compares different second layer scaling solutions in terms of data availability, liveness assumption (i.e., whether or not users need to be active to ensure security) and withdrawal time (i.e., time before users can retrieve the funds they locked in layer one to use the solution).

III. ROLLUPS

A. OVERVIEW

Blockchain systems are still far from offering a quality of service comparable to what centralized systems currently offer in terms of transactions per second [6], [8]. Popular systems are held back from offering efficient and accessible services mainly because of resource limitations. Indeed, each network node participating in the Proof-of-Work consensus mechanism needs to validate every transaction. This takes large amounts of computing power, storage capacity, and time. Rollups are a way of reducing the resources and time required to validate transactions by reducing the amount of data each node needs to process. They provide this enhancement by using a second layer network composed of actors who process transactions outside of the main chain. Then, the transaction data is *rolled up* in batches which are published to the layer one blockchain. Figure 3 shows how layer one and two interact in a rollup context.

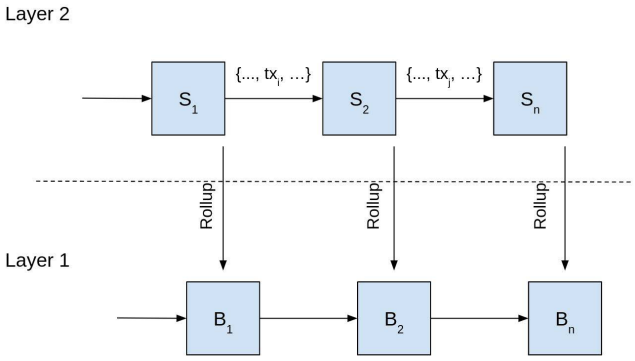

FIGURE 3. Interactions between layer one and two in rollups.

Rollup systems are implemented through layer one smart contracts. Different actors in rollups systems (e.g., aggregators, sequencers, verifiers) interact with this smart contract. Aggregators can publish the transaction data and other information (e.g., state root) through this contract. Verifiers also use it to dispute transactions when need be. Figure 4 shows the relations between the layer one and two users, the aggregators and the layer one smart contracts in rollup systems.

Users engage with rollups by transacting with the rollup smart contract. Such a contract lives on the first layer; this means that layer one fees need to be paid in order to start using the layer two system. Users first need to deposit funds via this contract. This deposit gives the user an amount of second layer tokens of proportional value. With these tokens, users can transact on the second layer system. They are then less bounded by the first layer fees. To transact on the layer two system, users send their transactions to aggregators either directly or through a layer one smart contract (the latter not shown in Figure 4). Aggregators then select a set

TABLE 1. Comparison between second layer scaling solutions.

Layer 2 solution	Data availability	Liveness required	Withdrawal time
State channels	Layer 1	✓	Instant
Sidechains	Layer 2	✗	Instant
Plasma	Delegated	✓	One week
Rollups	Layer 1	(✓, ✗)[1]	Few minutes to one week
Validiums	Delegated	✗	Few minutes

[1] Required for optimistic rollups only.

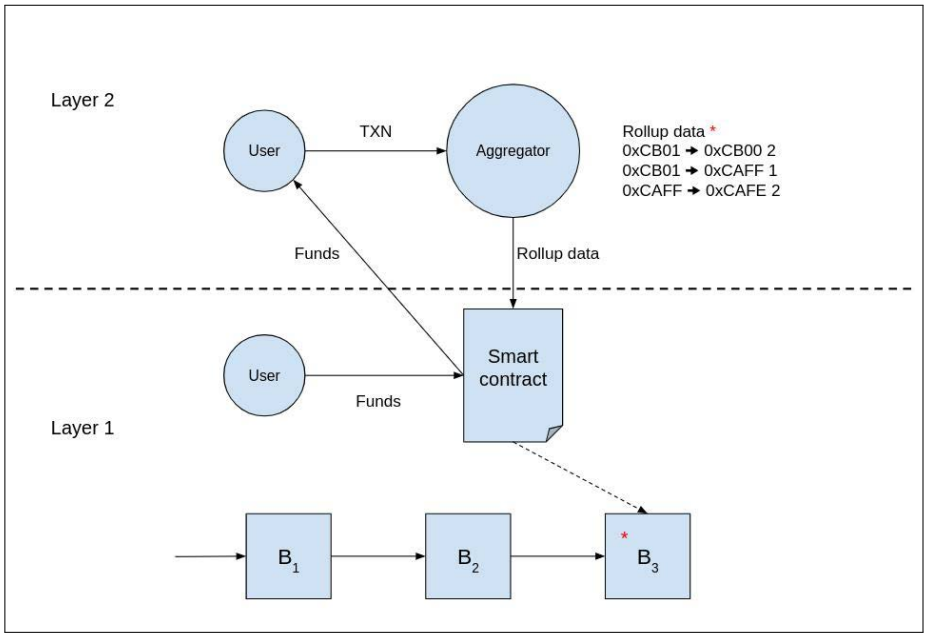

FIGURE 4. Smart contract interactions in rollups. TXN represents a transaction sent by the user to the aggregator. Rollup data represent the second layer transactions that were computed by the aggregator and that are being published to layer one.

of transactions they received, execute them, and publish the compressed transaction data (also called rollup data) on the layer one chain. The rollup data then becomes public and immutable. In order to reclaim their layer one tokens, users have to transact with the smart contract and pay layer one fees once more.

B. DEFINITIONS

There exist two main types of rollups which are currently being discussed and implemented: optimistic rollups and zero knowledge (ZK) rollups.

1) OPTIMISTIC ROLLUPS

The name *optimistic* comes from the way these rollups function. In this type of rollup, a layer two actor, which we will call aggregator, executes transactions off-chain and publishes only transaction data and the new state root on-chain. No proof of validity is included; layer one and two nodes optimistically suppose that the computation that was executed by this actor is valid. Nodes on the main network do not need to process every transaction every time a new rollup (or batch of transactions) is posted; this reduces the load on layer one nodes.

While detailed steps vary depending on the specific implementation, optimistic rollups systems usually follow the same high-level structure.

There are at least two second layer actors involved in optimistic rollups: *aggregators* (also called operators) and *verifiers* (also called watchers). First, transactions are received by an aggregator either directly from a layer two network broadcast or by an inter-chain smart contract. An inter-chain smart contract is a contract that allows communication between the layer one and layer two chains. In this case, an inter-chain contract can be used to receive transactions of layer one and forward them to an aggregator (see Figure 4). The aggregator then selects an arbitrary number of transactions, processes them, and publishes the compressed transaction data and state root on the layer one chain. Verifiers continuously monitor data published by the aggregator. If a verifier disagrees with the published data, it initiates a dispute phase. This happens when the verifier disagrees that the published transactions, once executed, lead to the published state root. We will discuss implementation specific dispute resolution protocols in section IV; they are generally concluded by a generated fault proof that determines the inconsistency.

Since the dispute resolution process is generally resource intensive, attempts at fraud and disputes are disincentivized

with bonds. Aggregators stake a bond which will be slashed if a fault proof shows that they were dishonest. Verifiers stake a bond that will be slashed if a dispute phase they triggered shows that the aggregator was honest. The finality of a transaction is guaranteed after a certain amount of time has been spent without the system entering the dispute resolution process.

Verifiers are incentivized to initiate dispute phases when they find an inconsistency in transaction batches. When a resulting fault proof shows that an aggregator was being dishonest, the verifier is rewarded with half of the bond of the aggregator. The other half of the bond is burned to avoid malicious behavior from aggregators.

Indeed, an aggregator could voluntarily submit fraudulent transaction batches. A verifier monitoring the system could detect a discrepancy and submit a transaction to initiate the dispute phase. Then, an aggregator could also submit (from a different address) a transaction with higher fees to initiate the dispute phase. Assuming this transaction is executed before the first one, the malicious actor would claim its own bond. Such a strategy is countered by burning half the bond of the fraudulent aggregator.

Optimistic rollups reduce processing power needed to run a first layer node by extracting transaction execution outside the layer one chain and onto the layer two chain. First layer nodes only need to process the transactions in the case of a dispute, which rarely happens. Dispute resolution protocols vary in different implementations and are discussed in section IV-B. Once the dispute resolution and bonds are slashed, the second layer transaction execution is resumed.

Intuitively, this process is secure with only one active verifier. Nodes running verifier software need to have a great amount of resources available in order to verify that aggregators are staying honest. The performance advantage brought by optimistic rollups comes from the fact that every layer one node does not need to act as a verifier; this task is delegated to a smaller number of computationally powerful layer two nodes.

2) ZK ROLLUPS

Zero knowledge (ZK) rollups also achieve performance improvement by moving transaction execution from layer one to layer two. However, instead of employing a *innocent-until-proven-guilty* scheme, aggregators in ZK rollups submit a proof that the state root published with the transaction data is correct, i.e. the transaction execution computation is exact. Such a proof is called a *validity proof* and is crafted using cryptographic tools such as Zero-Knowledge Succinct Non-interactive Arguments of Knowledge (ZK-SNARKs); zero-knowledge means that the input data needed for the computation does not have to be shared with the verifier for it to be convincing; succinct means that the size of the proof and the time needed to verify it grow slower than the time and size of the computation itself; non-interactive means that no back and forth interactions have to take place between the prover and the verifier. Indeed, a prover is able to convince a verifier with a single message that a valid computation (i.e. executed with coherent inputs and outputs) has been made. Furthermore, this is achieved without the verifier having to make all of the computations themselves. The fact that these proofs are non-interactive is a key element of their usefulness in blockchain technology and peer-to-peer networks. Blockchain systems are far too large to require back and forth communication between several actors; ZK-SNARKs have the advantage of convincing peers that a computation is exact with only one message.

However, ZK-SNARKs require a trusted setup for the initialization of a proof system, which implies trusted actors. Blockchain systems have been designed to function under a majority of honest participants without having to trust any single participant to be honest. Requiring a trusted setup means that in order to use ZK-SNARKs, one actor needs to allow some level of trust to another actor. This can be done by way of an in-person ceremony. Once the system is setup, an arbitrarily large amount of proofs can be generated and verified. In contrast, ZK-STARKs (Zero-Knowledge Scalable Transparent Arguments of Knowledge) require no trusted setup, yet produce larger proofs and require more time for generation and verification (see Table 2).

Zero knowledge proofs are not a new concept; however, they have become increasingly interesting in the past few years as their potential in several fields including blockchain technology has been highlighted. With this increase in interest has come significant work on ZK-SNARK and ZK-STARK technology, making ZK rollups a viable solution for blockchain layer two scaling [32], [34].

While implementation-specific mechanisms will be discussed in section IV, let us present the general idea behind ZK rollups. Similarly to optimistic rollups, ZK rollups work with second layer nodes (i.e., aggregators) that aggregate transactions received on the layer two network. Upon reception of the transactions, the aggregator executes them. Then, it publishes to the main chain the new state root, the compressed data of the transactions, and a proof of validity. This proof of validity ensures the computation made to execute the transactions was correct (i.e., coherent inputs and outputs). This publication of data is done through function calls to a layer one smart contract. Such a contract verifies that the ZK proof is correct. If it is, then everything is recorded on the blockchain. If it is not, the transaction batch is rejected. In contrast to optimistic rollups, ZK rollups do not require second layer verifiers and there is no dispute resolution. This means transactions achieve finality rapidly; there is no extended period of time where verifiers can trigger a dispute phase.

An in-depth explanation of how ZK-SNARKs work is out of the scope of this paper but let us overview the idea behind this kind of proof. Such a proof involves two actors: the prover, who generates the proof, and the verifier, who verifies that the proof is correct. In this scheme, the prover is trying to convince the verifier that a computation was executed correctly.

TABLE 2. Complexity and requirements of ZK technologies.

	ZK-SNARK	ZK-STARK
Prover algorithm complexity	$\mathcal{O}(n \cdot \log(n))$	$\mathcal{O}(n \cdot (\log(n))^c)$
Verifier algorith complexity	$\mathcal{O}(1)$	$\mathcal{O}((\log(n))^c)$
Proof size	$\mathcal{O}(1)$	$\mathcal{O}((\log(n))^c)$
Requires trusted setup	Yes	No
Cryptographic requirements	Discrete logarithm hardness and bilinear mappings	Cryptographic hash function

With c a constant.
Sources: [32], [33]. Note that ZK-SNORKs have the same properties as ZK-SNARKs but only require one universal trusted setup instead of requiring one trusted setup per application.

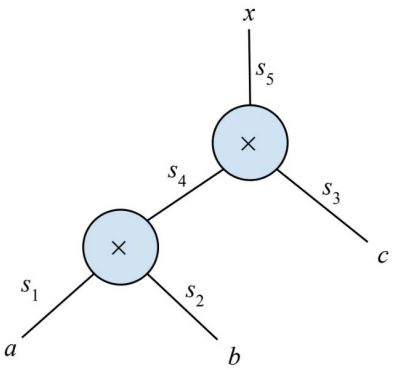

FIGURE 5. An example arithmetic circuit.

First, to craft the proof, the set of basic computation instructions that makes up the program that is to be verified is agreed upon by both parties (the prover and the verifier). These instructions, or code, are transformed to their atomic components into an arithmetic circuit. For example, the computation $(a \times b) \times c = x$ would be transformed in the following circuit, composed of $n = 5$ wires and $m = 2$ gates (see Figure 5).

In this example, an argument is made about the knowledge of a solution $s = (s_1, s_2, s_3, s_4, s_5)$ such that $(s_1 \times s_2) \times s_3 = s_5$.

After this transformation, for each gate, vectors of constants a_i, b_i and c_i are found such that

$$(s \cdot a_i) \times (s \cdot b_i) = (s \cdot c_i), \forall i \in \{1, \ldots, m\} \tag{1}$$

These are the constraints of our system. Once we have these sets of vectors, we can derive 3 sets of polynomials (one per set of constraints) such that for all $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$,

$$p_j(i) = a_i[j] \tag{2}$$
$$q_j(i) = b_i[j] \tag{3}$$
$$r_j(i) = c_i[j] \tag{4}$$

These are constructed using some method of polynomial interpolation, e.g. Lagrange interpolation. Once we have these polynomials, we compute $h(x)t(x)$ defined by

$$(\sum_{j=1}^{n} s_j p_j(x)) \cdot (\sum_{j=1}^{n} s_j q_j(x)) - (\sum_{j=1}^{n} s_j r_j(x)) \tag{5}$$

where $t(x) = (x-1)(x-2)\ldots(x-m)$.

Note that by construction, the polynomial computed in (5) will have roots at the target points $\{1, \ldots, m\}$ when a valid solution s is used in the calculations.

What we have described here is called a quadratic arithmetic program (QAP) [35]. We have reduced our "computation" to a polynomial on which we can do some checks (e.g. divide (5) by $t(x)$ and verify that the remainder is 0) that ensure a coherent solution s is known by a prover without necessitating disclosure of the solution. We now understand the idea with which a prover can transform a computation into a mathematical object that guarantees the computation was executed correctly. Using this simple construction, the zero-knowledge and succinct properties of ZK-SNARKs are not attained. Intricate details of the different approaches used to achieve these properties in zero knowledge proofs are out of the scope of this paper. Broadly, zero-knowledge is achieved using strong homomorphic encoding over finite fields before communication to the verifier. Succinctness is achieved through performing checks on random samples of the polynomial $h(x)t(x)$ [34], [36].

Once these steps are done, every node on the network can verify (in a computationally efficient manner) that the computation result published by the aggregator (i.e., the prover) is coherent with some input.

ZK-STARK (Zero-Knowledge Scalable Transparent Arguments of Knowledge) technology is a newer concept that provides similar functionalities as ZK-SNARKs, but without the need for an initially trusted setup between the two parties. While the construction of such proofs varies greatly from ZK-SNARKs, the basic idea of reducing a computation to a polynomial remains. ZK-STARKs do not rely on elliptic curve cryptography to achieve zero-knowledge as ZK-SNARKs do; hash functions are used instead. Finally, the proof size is generally several orders of magnitude bigger (see Table 2).

We can summarize that the role of rollups is to provide relief on the layer one network by executing transactions off-chain and ensuring data availability by publishing transaction data on-chain. Rollups either work optimistically with watchers or using zero-knowledge proofs.

3) DATA COMPRESSION

Data compression is critical in the context of rollups to reduce the amount of data, published in the layer one blockchain. This helps making transactions on layer two chains cheaper in

TABLE 3. Transaction data size (in bytes).

Field	Ethereum transaction	Rollup
Nonce	1 — 32	0
Gas Price	1 — 32	0.5
Gas Limit	1 — 32	0.5
Recipient	21	4
Amount	1 — 32	3

terms of processing fees [37]. Data compression is achieved by only publishing to the main blockchain the transaction information needed to ensure a deterministic re-execution of transactions. Table 3 shows the data size of certain fields, of an Ethereum transaction, and their sizes in a compressed format [38], [39].

The nonce can be omitted since this field is simply incremented by a value of 1 at each transaction from a specific sender. Once the transaction has been processed by the layer two network, there is no need to store the nonce on the layer one chain as it can be retrieved by replaying all prior transactions. Gas price and limit can be reduced to 1 byte by letting users choose from a set of values (e.g. successive powers of two) instead of letting them choose and exact price and amount. The recipient address can be reduced to an index in the address tree which only takes 4 bytes. Finally, the amount of the transaction can be specified in scientific notation with only a few significant figures.

IV. IMPLEMENTATIONS

A. HISTORY

In this section, we provide historical dates for the main events of rollups research & development in order to give more context about its progress.

The first and most important observation that can be made is how fast scaling solutions were proposed in the case of Arbitrum (see Table 4). In fact, it only took one year after V. Buterin's Ethereum paper [8] for a team of Princeton students to propose the first iteration of their platform [41], that would overcome Ethereum's limitations (mainly scalability by offering a larger throughput, and pseudo-privacy concerning smart contract execution for example).

Arbitrum and Optimism are some of the most important actors in the optimistic rollups environment currently. Even though Optimism was the first team to implement them in 2019, both have been made available at the same time around late 2021, either only for developers or fully open on mainnet, respectively. Furthermore, both solutions are pegged to the Ethereum network in order to benefit from its battle-tested security and decentralization. Finally, we choose to present these two implementations because of their different approaches taken, especially in dispute resolution.

As for ZK rollups, while many solutions have appeared in the last few years, we present four main actors: StarkEx and StarkNet by Starkware, zkSync by Matter Labs, Polygon Hermez by Hermez DAO, and Loopring by the Loopring Foundation. These solutions primarily differ in the type of cryptographic proofs used in their protocols: (a) Loopring uses ZK-SNARKs [40]. These proofs require an application-specific trusted setup; (b) zkSync and Polygon Hermez use Universal ZK-SNARKs, also called ZK-SNORKs. These proofs require a universal trusted setup and was developed by Aztec [44]; and (c) Starkware uses ZK-STARKs [32].

B. OPTIMISTIC ROLLUPS

1) OPTIMISM

Optimism is the first team to ever implement Optimistic Rollups in 2019 with an **Ethereum Virtual Machine (EVM) compatible** environment - the Optimism Virtual Machine (OVM). This compatibility meant additional work for every application to be used on Optimism. In fact, even for a simple DeFi smart contract application like *Uniswap*, custom code was required.

However, the Optimism team did not think this paradigm was suited for the exponentially growing Ethereum ecosystem. They then decided to transition to make the OVM an **EVM equivalent** environment to benefit from all the mature infrastructure already available for Ethereum. Such a change would imply that this second layer environment complies with Ethereum's Yellow Paper [38], making every application and more broadly software, natively work on Optimism. In order to reach this goal, the Optimism team decided to keep a minimal codebase, and built their client using *Geth* - the official Go implementation of the EVM.[1] Hence, the whole protocol has been build with simplicity as a core value.

Users interact with a *sequencer* (i.e., an aggregator) on layer two; it collects the transactions, executes them, and creates a rollup to send the data to layer one. If a verifier detects a fraudulent rollup, it submits a fault proof for a single transaction in the batch. This is done by interacting with the rollup smart contract. This challenge is public and second layer nodes provide the appropriate data for the smart contract on layer one to verify the batch of transactions. This smart contract automatically finds the fraudulent transaction by verifying state transitions one by one for every transaction in the batch. If the fraudulent transaction exists, it will be invalidated along with all the transactions following it. The sequencer will also be penalized. However, if the challenge fails, the verifier will get penalized.

This dispute resolution mechanism is simple and allows for a smaller codebase but requires layer one to compute a whole transaction. This could range from a simple token transfer to a complex smart contract execution, hence placing the burden on the Ethereum chain on every disagreement.

However, a lot more progress is expected for Optimism, and especially on the sequencer node. In fact, the only one to date is operated by Optimism itself, which is a major threat for censorship resistance.[2] In order to decentralize this role, the

[1] https://github.com/ethereum/go-ethereum

[2] The sequencer technically cannot censor any user because the protocol allows them to interact directly with the smart contract on layer one, bypassing the Sequencer - but forcing the users to pay higher gas fees and waiting for at least 24 hours.

TABLE 4. Timeline for most important rollups technologies.

2013	Ethereum white paper [8] & SNARK whitepaper [40]
2015	Arbitrum academic start [41] & Ethereum project launch
2017	Loopring project start [42]
2018	STARK whitepaper [32], Arbitrum white paper [43] & StarkWare start
2019	SNORK whitepaper [44], Optimism & zkSync start, StarkDex launch
2020	Loopring Exchange & zkSync project launch (v1)
2021	Arbitrum One developer, Optimism & StarkNet launch
	...

Optimism team proposed to implement a **Miner Extractable Value (MEV) Auction**. The basic idea behind a MEV auction is based on the separation of transaction inclusion (i.e., which transactions will be executed) and ordering (i.e., in what order they will be executed).

In a MEV Auction, block proposers propose a set of transactions to be executed within a period of N blocks. They do this via a smart contract which can be deployed on layer one or two. Sequencers then bid via this contract for the right to choose the ordering of the proposed set of transactions.

The MEV problem is a well known problem in blockchain systems where block producers extract value by manipulating the transactions to be included in a block. These transactions can be delayed, ignored or reordered by the block producer in order to maximize profit [45]. The MEV Auction proposed by Optimism solves the MEV problem by separating transaction ordering and processing.

In order to test the capability of rollups, the Optimism team organized a contest with Synthetix, letting users trade on a decentralized exchange (DEX), the Synthetix Exchange. While Optimism demonstrated record performance during the demo,[3] it is expected, under normal circumstances, to reach 2 000 TPS[4] with fees ten times lower than Ethereum [46], for a finality time of under one second.

2) ARBITRUM

Arbitrum is a second layer network based on optimistic rollups being developed by Offchain Labs since 2018. Unlike Optimism, this solution aims at minimizing the computation on the Ethereum network during dispute resolutions at the expense of simplicity. In fact, when a verifier node (also called validator in Arbitrum) submits a fault proof for a rollup, it will enter a dispute period alongside the aggregator using a K-way Dissection protocol. We present here a simplified Bisection version of the algorithm, as initially stated by the Princeton students.

Arbitrum Challenge Period setup:

- Assume B is a large amount of Ether.
- The game consists of two players, an *aggregator* Alice and a *validator* Bob.
- Assume Alice submits a rollup containing a number of transactions representing N instructions to L1 alongside a bond B.
- Bob disagrees and submits a fault proof for it, also staking a bond B. They then enter the Challenge period.

Algorithm 1 Bisection Algorithm Pseudo-Code

Require: $len(range) \geq 1$
Ensure: The single conflicting instruction
1: $range \leftarrow [0, N]$
2: **while** $len(range) \neq 1$ **do**
3: Alice bisects the range into r_1 and r_2
4: Bob has to select r_s from r_1 and r_2 and claim it contains a fraudulent instruction
5: $range \leftarrow r_s$
6: **end while**
7: $ins \leftarrow$ single instruction obtained
8: **return** *ins*

Arbitrum's Challenge Period is composed of two phases: a **dissection** (finding the conflicting instruction), and the **one-step proof** (proving to layer one that the instruction is valid, or failing at doing so). After entering the Challenge Period, Alice and Bob will play an iterative, multi-round game to find the contentious instruction (see Algorithm 1). After obtaining a single instruction *ins*, Alice will have to send a one-step proof for it to layer one.

The internal design of the **Arbitrum Virtual Machine** (AVM) was designed to minimize the data required to disclose when the aggregator has to submit a one-step proof. Indeed, the AVM uses a approach similar to lazy lists to obfuscate the rest of the data. Such a data structure can be recursively defined as follows:

```
List -> Data:: List
| NULL
#:: being the append operator
```

When having to provide a one step proof, the aggregator can provide the first value alongside its hash as well as the hash of the rest of the list. Since the state root hash of the previous transaction was built with the root hash of this list, it is easy to verify that this data structure is the same as the one at the end of the previous transaction. After receiving values and hashes for the data and instruction of the contested

[3]Fees were on average 143 times cheaper, and the transaction finality was 0.3 seconds.

[4]https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/optimistic_rollups/

operation, the smart contract on layer one can easily execute it and determine its validity.

If it is valid, Alice will receive half of Bob's bond; otherwise, she will forfeit half of her bond to Bob. From this algorithm, multiple properties emerge:

- Correctness: If Alice provided a valid rollup, she will always be able to win the game. If it is incorrect however, Bob will always be able to win.
- Efficiency: No matter the size of a rollup, the dispute resolution will always yield a unique instruction to resolve the conflict in logarithmic time. ($\log_2(N)$ for the simplified version, even less for the K-way Dissection[5]).
- Pseudo-Privacy: In the process, Alice will only have to show the state of the machine on a single instruction to prove that she did not violate the state transition function. Therefore, achieving pseudo-privacy by hiding most of the inner working of the AVM for earlier and upcoming instructions.

On the technical level, even though Arbitrum announces 4 500 expected TPS[6] in their documentation and a fee reduction of about ten times compared to layer one, the network is currently throttled. Thus, it cannot reach these performances, but will gradually run to its full potential. The team also released **Anytrust** in early 2022, which are sidechains working alongside the Arbitrum's optimistic rollup. Anytrust chains are expected to offer 40 000 TPS and fees a thousand times cheaper than on layer one.

Finally, like Optimism, Offchain Labs currently operates the only *Sequencer* on Arbitrum; however, the team plans to implement a **Decentralized Fair Sequencing** algorithm (also called Decentralized Fair Ordering algorithm). This approach is based on a committee of servers to collectively decide on the ordering of transactions. Even though this approach is meant to decentralize the role of the *Sequencer*, it is important to note that a super-majority of non-malicious servers is needed in this scenario to ensure a fair ordering.

3) COMPARISON

We present an example of a fraudulent rollup and the dispute resolution mechanism for Optimism and Arbitrum. Figure 6 shows the initial layer two state tree associated with the ERC20 token smart contract. The leaves of the tree represent the account addresses and balances on layer two. The root of this state tree and the list of transactions have been published on layer one by the aggregator. This is the last valid state tree; some earlier batch of transactions published in an honest rollup lead to this state tree.

Let us consider the case where the aggregator has published a rollup containing the three transactions shown in Figure 7 and the root of the resulting state tree. This is published by sending a layer one transaction to the rollup smart contract. Note that the last transaction of the rollup is fraudulent; the amount sent is 2 but the aggregator increased the balance of the recipient by 990 instead of 2.

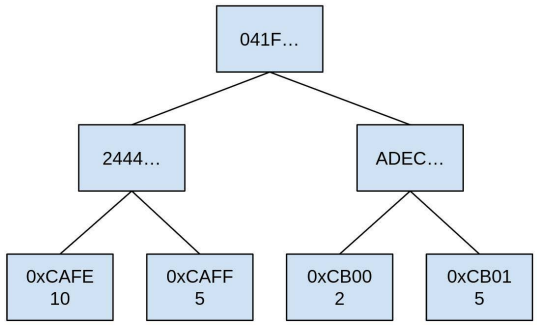

FIGURE 6. Initial state tree. The leaves of the tree represent the address of the account (e.g., `0xCAFE`) and their balance (e.g., 10).

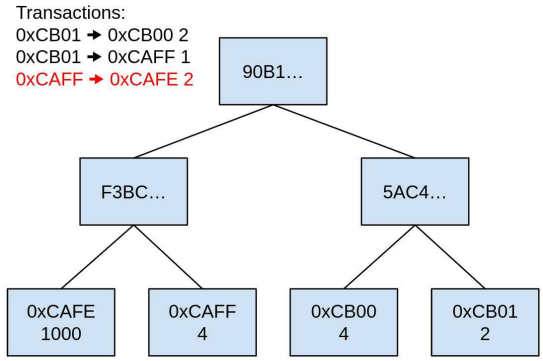

FIGURE 7. Fraudulent state tree.

When this rollup is published, a verifier will replay the published transactions on its local state and compare the resulting state roots. A discrepancy will be found and the verifier will trigger a fault proof. This is where the behaviors of Optimism and Arbitrum fork.

In Optimism, the verifier triggers the fault proof by targeting one invalid transaction from the rollup batch. It is worth noting that in Optimism, intermediate state roots (state roots obtained after the execution of a single transaction on layer two) are published by aggregators and stored on layer two. A verifier can identify which transaction is fraudulent by replaying them one by one and comparing the intermediate state roots. In our example, there are three intermediate state roots: S_1, S_2 and S_3. S_1 is obtained after executing the first transaction, and so on. The verifier triggers the fault proof for the third transaction (i.e., for S_3). It is worth noting that in S_2, the balances for accounts `0xCAFE` and `0xCAFF` are `10` and `6` respectively; this is stored on layer two and associated to the layer two ERC20 smart contract.

A fault proof is played out as follows:

1) The verifier deploys on layer one the ERC20 token smart contract that was on layer two. This is done so layer one can simulate the disputed transaction.

2) Then, the verifier asks the rollup smart contract to verify that this freshly deployed smart contract has the same bytecode as the one on layer two. At this stage, the rollup smart contract has ensured that the layer one and two ERC20

[5]The general K-way Dissection converging after $\log_K(N)$.
[6]https://developer.offchainlabs.com/docs/rollup_basics#throughput

TABLE 5. Performance comparison between optimistic rollups implementations.

Implementation	TPS	Fee reduction	Fault proof execution	General computation	EVM compatibility
Optimism	2 000	10x	Layer 1[1]	✓	compatible
Arbitrum	4 500	10x	Mostly Layer 2[2]	✓	compatible

[1] Transactions provided one by one on the first layer until the fraudulent transaction is identified.
[2] Second layer mechanism to find the fraudulent transaction; only this transaction is executed on layer one.

contracts have the same bytecode. The layer one duplicate smart contract has no data from layer two yet.

3) The verifier then provides to the rollup smart contract the balances in the last undisputed state (e.g., S_2) of the sender and receiver of the fraudulent transaction (i.e., `(0xCAFE, 10), (0xCAFF, 6)`). This is the minimum information needed for layer one to simulate the disputed transaction.

4) The verifier then asks the rollup smart contract to simulate the disputed transaction (i.e., `0xCAFF` → `0xCAFE 2`) to the ERC20 duplicate contract on layer one. This simulation is played out on layer one; no fraud can happen here. During the simulation, the two account balances will change; the balances for accounts `0xCAFE` and `0xCAFF` will then be `12` and `4` respectively.

5) The verifier then provides to the rollup smart contract a Merkle proof for every changed account balance (see Figure 8). This is done to prove that the changes in balance computed by the verifier and the smart contract are the same.

In the example, the Merkle proof consists of sending the value of the node that contains the hash of the two right leaves (i.e., `"5AC4..."`). Using this Merkle proof, the rollup smart contract can compute the correct state root S_3' and provide it to layer two. Any transaction coming after the proven fraudulent transaction is reverted. After this whole process, layer two execution continues normally.

In contrast, Arbitrum does not work with the state tree of the smart contracts involved in the dispute. Instead, Arbitrum works with the state root of the Arbitrum Virtual Machine (AVM). This enables a high level of granularity on what has to be proven and therefore small computation time (e.g., $\mathcal{O}(1)$ for Arbitrum instead of $\mathcal{O}(\log(n))$[7]) for Optimism. In Arbitrum, fault proofs are played out as follows:

1) Algorithm 1 is used on the layer two network to identify the very instruction that is fraudulent. This is more granular that in Optimism; a transaction requires several instructions to be executed. This is played out between the aggregator that published the rollup and the verifier that challenged the rollup. In our example, the fraudulent assertion made by the aggregator would be the following: starting with some AVM state of root `"ABCD..."` (see Figure 9), the execution of instruction

```
ADD $0xCAFE 2
```

will lead to AVM state root `"ABCE..."`. Note that in the instruction, `0xCAFE` represents the register where the balance of account `0xCAFE` is stored.

[7]Where n is the size of the data associated with the smart contracts involved in the disputed transaction.

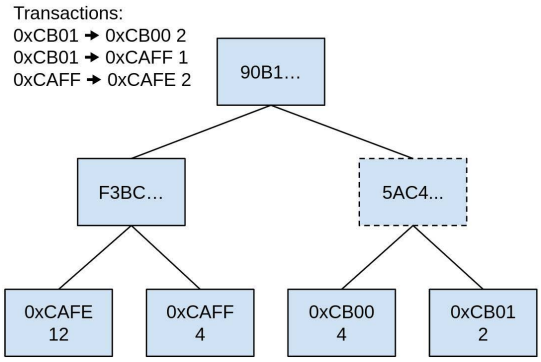

FIGURE 8. Merkle proof provided by the verifier to the rollup smart contract. The rollup smart contract will already have the involved account balances; the verifier only needs to provide one tree node to be able to compute the tree root.

2) In order to prove that the assertion is false, the verifier will provide to the layer one rollup smart contract the necessary information for the execution of the instruction. The AVM is a stack-based virtual machine; therefore, only the data at the top of the stack and the hash of the rest of the stack needs to be sent to layer one for the proof to be complete.

Explicitely, the verifier will send the `ADD` opcode, the hash of the rest of the instruction stack, the value of `$0xCAFE`, the immediate value 2, and the hash of the rest of the data stack (see Figure 9).

3) With this information, the rollup smart contract will execute the instruction by emulating the AVM. The resulting AVM state root will be computed by the rollup smart contract using the freshly computed values and the provided hashes (see Figure 10).

4) The post-execution AVM state root is computed by the rollup smart contract. The new state root (i.e., `"ABCF..."`) is different than the one asserted by the aggregator (i.e., `"ABCE..."`); therefore, the aggregator was fraudulent.

In both Optimism and Arbitrum, the bond the aggregator had previously staked is slashed; half is burned and half is claimed by the verifier that triggered the dispute.

Finally, Table 5 shows a comparison between Optimism and Arbitrum in terms of TPS, fee reduction, fault proof execution, general computability, and EVM compatibility.

C. ZK ROLLUPS

1) StarkDex/StarkNet

As decentralized exchange (DEX) transactions represent a significant portion of all Ethereum traffic,[8] Starkware

[8]At the day of writing, about $3B was transacted on DEXs out of the $18B total daily volume.

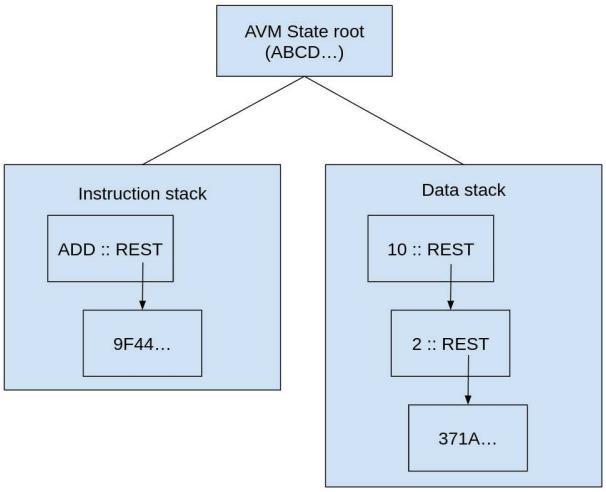

FIGURE 9. Pre-execution state of the emulated AVM on the layer one smart contract. All of the information has been provided by the verifier.

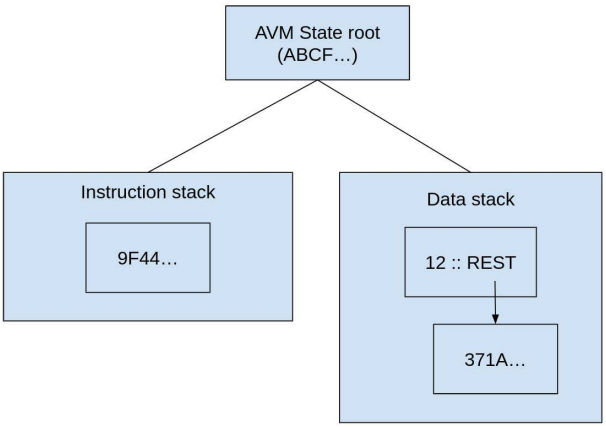

FIGURE 10. Post-execution state of the emulated AVM on the layer one smart contract.

decided to first scale this kind of application using ZK rollups and STARK validity proofs. They therefore launched **StarkDex** in 2019: a DEX leveraging the security and speed of layer two ZK rollups. This would be achieved by creating both software and hardware for fast and efficient STARK proof creation and verification.

With time, StarkDex became **StarkEx**, a proprietary scalability engine used to build DEXs. This software allows developers to choose the data availability mechanism for their application, by either using ZK rollups (on-chain) or Validiums (off-chain). Some of these customers include: dYdX,[9] Sorare,[10] Immutable X[11] or even DeversiFi.[12]

While an increased efficiency and throughput for an application is always desirable, this does not resolve Ethereum's important network congestion. The next step was therefore to offer a platform to scale general purpose computation such as smart contract execution - **StarkNet**. This new layer two solution is EVM compatible, meaning that it is not trivial to have equivalent applications on this network. Users will have to write their smart contracts in a new programming language developed by Starkware: Cairo. Even though this requirement may add friction during the transition of developers, Starkware also made available **Warp**,[13] a Solidity to Cairo transpiler.

StarkEx already offers cutting-edge performance. Platforms like DiversiFy allows traffic reaching 9 000 TPS for trades and going up to 18 000 TPS for self-custodial token transfers when data is stored off-chain for example [47]. Starkware also participated in Reddit Scaling Bake-Off in 2020 to present their technology stack [48]. This initiative launched by Reddit and the Ethereum Foundation was created to see if the Ethereum Network was ready to support big scale applications like Reddit (reporting 430 million monthly users). Starkware submited a proposal based on their ZK rollup technology and reached 3 000 TPS with only 315 gas units per transaction (i.e., $0.005 for Ether price $238.36), a record in the Rollups space. Finally with StarkNet, expected throughput and benchmarks were not disclosed at the time of writing. However, the platform is expected, in the long run, to offer fees that are cheaper by a factor bigger than a hundred compared to layer one.

Starkware innovations are far from being over as the team is currently working on **Volition**. This is a novel project combining the security of ZK rollups with the speed of Validiums to offer a flexible service. Users will choose the data availability scheme they want to use for each of their transactions: either secured on layer one (on-chain) or stored off-chain and regulated through a data availability Committee. This would allow users to have a fine grained control over their transactions, as they will be able to use this service based on their risk tolerance and desired throughput.

2) zkSync

Matter Labs wants to offer layer two scaling to general purpose computation through their product, **zkSync**. This protocol is open source and allows users to build their applications on top of it to benefit from the low transaction cost and speed of ZK rollups. Even though this project is also based on ZK rollups, it uses ZK-SNORKs for their validity proofs, requiring a trusted setup: initial parameters to generate the ZK-SNORKs have been determined during the Multi-party Computation Ceremony.[14] Like StarkNet, zkSync is EVM compatible and uses Zinc, a new programming language that is not Turing-complete, for the development of Smart Contracts.

Regarding performance, Matter Labs claimed a theoretical throughput of 2 000 TPS[15] with the previous block gas limit of 12.5M.[16] This means fee reduction ranging from 20 times for ETH transfers up to 100 times for ERC20 tokens.

[9]https://dydx.exchange/
[10]https://sorare.com/
[11]https://www.immutable.com/
[12]https://deversifi.com/
[13]https://github.com/NethermindEth/warp
[14]https://ignition.aztecprotocol.com/
[15]https://docs.zksync.io/userdocs/tech.html#maximum-throughput
[16]The maximum limit at the time of writing is 30M.

Similar to their counterparts, Matter Labs also want to bring exponential scaling to layer two with the help of Validiums and their expected new product, **zkPorter** would be one. The platform is expected to have possibly more than 20 000 TPS with constant fees of about 1 or 2 cents [29]. Users will have to choose either ZK rollups or zkPorter for their second layer accounts. The team did announce effortless interoperability while transacting between two accounts, regardless of their type.

3) POLYGON HERMEZ

The sidechain Polygon is actively collaborating with various layer two proposals in order to have a full suite of decentralized Ethereum scaling solutions. Many of them use an underlying Rollup technology: Polygon Hermez, Polygon Nightfall, Polygon Miden and Polygon Zero are all ZK rollups. Polygon Avail is an optimistic rollup. We present here the only technology currently live on Ethereum mainnet, Polygon Hermez - developed by the Hermez DAO. Polygon Hermez uses ZK-SNARKs for their validity proofs.

Polygon Hermez is only a payment system; it does not support general computation. However, this system is interesting because it offers a way to support the Ethereum community when electing their aggregator (also called coordinator), through a **Proof of Donation**. In this scheme, every node of the network can enter a public auction, using the Hermez utility token HEZ, to become the node that creates and submits the rollups for a given 10 minutes time slot. This auction is described by the following rules: (a) the minimum bid is set to 10 HEZ;[17] and (b) Each bid should be at least 10% greater than the previous one.

Upon begin elected, 30% of the bid will be burned, another 40% will get transferred to the Ethereum Foundation to fund community projects and finally the remaining will be allocated for network incentives, like staking for example. While coordinators loose the whole bid, they will receive transaction fees for each transfer they process in a rollup. Based on Hermez DAO's point of view, this mitigates the risk of censorship since coordinators will be incentivised to include all transactions to earn back the money lost in the bid.

Regarding performance, Polygon Hermez claims a throughput of 2 000 TPS, with a 90% fee reduction compared to layer one.[18] In fact, at the time of writing sending ETH costs only $0.25 on Hermez, against layer one averaging under $10.[19]

4) LOOPRING

Loopring is the first ZK rollup layer two protocol implemented on Ethereum, relying on ZK-SNARKs for its validity proofs. It is an open source protocol [42] and is available for anyone to build payment applications and non-custodial DEXs on Ethereum.

The Loopring Foundation also developed the first publicly available decentralized exchange platform (DEX) using ZK rollups - **Loopring Exchange**. This platform differs from current DEXs; it uses automated market makers like *Uniswap* to match buyers and sellers and implements orderbooks like most centralized exchanges (CEX).

The Loopring protocol uses ring miners and liquidity providers to fill orders. Both these actors get rewarded for filling orders either in Loopring utility token (LRC) or by getting a margin on the filled order. The LRC token is also used by users who want to use Loopring to create a DEX: such users have to deposit between 250 000 and a million LRC, depending of the usage.

Similarly to the other discussed solutions, Loopring reports an expected 2 025 TPS with transaction fees ranging from 450 to 800 gas units[20] (i.e., between $0.05 and $0.1 for Ether price $2,600). For comparison, the average gas used for a first layer DEX is about 150 000.[21]

Table 6 shows a comparison between the different ZK rollup implementations in terms of validity proof type, TPS, fee reduction, general computation ability and EVM compatibility.

D. ADOPTION

Adoption of rollup technology is still low; users have locked in about 2.7% of all locked funds Ethereum.[22] Optimistic rollups have had an undeniable head start as the technology is orders of magnitude less complex thank ZK rollups and general computation is easier to attain. Indeed, about $\frac{3}{4}$ of all money locked on layer two comes from optimistic rollups. However, quoting Vitalik Buterin [39], it is believed that ZK rollups will gain more in popularity in the long term compared to optimistic rollups thanks to their privacy advantages. However, there is no EVM equivalent ZK rollup currently available to the public, making it had for developers to build application on ZK rollups.

There are several possible causes for the low adoption of rollups; 1) users are still unfamiliar with this technology and their usability is not developed enough. For example, merchants and charities generally do not accept cryptocurrency payments via rollups [39]; 2) rollups are not yet decentralized [49] In fact, many of the protocols display it as their core value yet are the only one operating a sequencer in their platforms. To our knowledge, Polygon Hermez was the only available protocol working in a fully decentralized manner at the time of writing; 3) optimistic rollups have long withdrawal periods. Although applications like fast exits via liquidity pools already exist, the two most important optimistic rollups do not offer a fast and easy way to withdraw funds.

[17] ≈ 50 USD at the time of writing.

[18] https://polygon.technology/solutions/polygon-hermez/

[19] https://www.l2fees.info

[20] https://loopring.org/#/protocol

[21] https://crypto.com/defi/dashboard/gas-fees

[22] ≈ $134.64B total Ethereum market capitalization and ≈ $4.64B locked in rollups as of June 2022.

TABLE 6. Performance comparison between ZK rollups implementations.

Implementation	Validity proof type	TPS	Fee reduction	General computation	EVM compatibility
StarkEx	STARK	3 000	200x[1]	✗	○
StarkNet		?	100x / 200x	✓	◐
zkSync	SNORK	2 000	100x	✓	◐
Polygon Hermez		2 000	10x	(✗, ✓)[2]	(○, ◐)[2]
Loopring	SNARK	2 025	150x	✗	○

[1] Performance observed. [2] Not supported at the moment but present in the roadmap for Polygon Hermez v2.
● EVM equivalent ◐ EVM compatible ○ Not EVM equivalent nor EVM compatible.

V. ANALYSIS

In this section, we discuss in more detail the performance of some of the current rollup implementations. Table 7 compares both the theoretical throughput and the expected fee reduction of each project, as well as the validity proof type, if applicable. Finally, we differentiate projects capable of running an arbitrary application using smart contracts and their compatibility with the EVM.

Both Arbitrum and Optimism offer similar fees that are about ten times cheaper than layer one for any application. However, Arbitrum puts forward a greater throughput; more than twice than the throughput of Optimism. More generally, Arbitrum offers a more important throughput than any other general purpose rollup platform. Furthermore, both projects are currently being throttled so we are still far from observing the theoretical performances. This is magnified by low rates of adoption; neither Arbitrum nor Optimism reaches more than one TPS on average on a given day.[23] At last, both projects suffer from a long withdraw period of 7 days. This is required for validators to provide a challenge in case of a fraudulent rollup. In comparison, it only takes a few minutes for a ZK rollup to validate a withdrawal and give access to the funds. However, the Boba Network provides fast exits from the second layer with a liquidity pool system.

This opens a new market for entities with liquidity to fund these fast-exit liquidity pools. Users who are willing to exit the layer two system without the 7-day dispute period can swap their assets via such a pool in exchange for a fee. The entities funding these pools are guaranteed to receive the funds within 7 days if they play the role of a rollup verifier. Indeed, when they verify the batch including the withdrawal transaction, they are certain that they will receive their funds within one week.

For ZK rollups, EVM equivalence or even EVM compatibility is a harder property to obtain. This leads to more layer two solutions being application-specific: DEXs for StarkEx and Loopring as well as payment systems for Polygon Hermez in particular.

Furthermore, the reported expected fees given by the different rollups vary greatly, from a factor of 10 for Polygon Hermez to reportedly up to 200 for StarkNet. Even if it is too early to tell how the fees will evolve with adoption, it is still important to mention that more costly payments will have to be made on all platforms to deposit and withdraw funds. Although the fees for transactions differ between ZK Rollups implementations, every implementation reports a similar theoretical throughput of around 2 000 to 3 000 TPS. Some solutions also offer an off-chain data storage and allow for even greater throughput, ranging from 16 000 to 20 000 TPS, at the cost of security. As validiums are not rollups, their performance numbers were not included in Table 7

We can estimate the usage with the Total Value Locked (TVL), which is currently $6.3B, for each of these layer two solutions. Arbitrum is undoubtedly the most important rollup with more than half of all TVL.[24] Although Optimism represents a smaller share (only 10%), Metis Andromeda and Boba Network, being forks of this protocol, collectively account for another 7%. Projects built using StarkEx also account for an important part of the TVL, with dYdX aggregating a little less than 16% by itself. Loopring represents 5% of the TVL; most of it comes from the staking of LRC tokens to operate a DEX. Finally, zkSync registers less than 2%, Polygon Hermez 0.01% and StarkNet has yet to have any TVL as it does not have a bridge on mainnet at the moment.

Table 2 outlines the differences in complexity of different zero-knowledge and their cryptographic assumptions. While ZK-SNARKs offer relatively smaller proof sizes and computation time for both provers and verifiers, they come with the burden of requiring a trusted setup before being able to perform proofs for a specific application [34]. Therefore, every time a new application (i.e. smart contract) is deployed on the blockchain, trusted parties are required to communicate honestly and agree on some key bits of information that will be used to craft the ZK-SNARK at an ulterior moment.

Acquiring a trusted setup in trustless environments like blockchain systems can be an arduous task; ZK-SNORKs and ZK-STARKs can help with this challenge. ZK-SNORKs can relieve users of this burden by reusing and modifying one initial trusted setup for all further smart contracts deployed [44]. ZK-STARKs no not require a trusted setup at all.

As for the cryptographic requirements, ZK-SNARKs and SNORKs rely on the hardness of the elliptical curve discrete logarithm problem (ECDLP) and on elliptic curve bilinear mappings (or pairings). Note that this means that

[23] https://pro.nansen.ai/multichain/arbitrum

[24] 56% at the time of writing, see https://l2beat.com/.

TABLE 7. Performance comparison between rollups implementations.

Rollup type	Implementation	Validity proof type	TPS	Fee reduction	General computation	Compatibility
Optimistic	Optimism	N/A	2 000	10x	✓	●
	Arbitrum		4 500	10x	✓	●
ZK	StarkEx	STARK	3 000	200x[1]	✗	○
	StarkNet		?	100x / 200x	✓	◐
	zkSync	SNORK	2 000	100x	✓	◐
	Polygon Hermez		2 000	10x	(✗, ✓)[2]	(○, ◐)[2]
	Loopring	SNARK	2 025	150x	✗	○

[1] Performance observed. [2] Not supported at the moment but present in the roadmap for Polygon Hermez v2.
● EVM equivalent ◐ EVM compatible ○ Not EVM equivalent nor EVM compatible.

ZK-SNARKs and SNORKs are vulnerable to attacks from malicious actors having access to a decently sized quantum computer [50]. On the other hand, ZK-STARKs only need a collision-resistant cryptographic hashing function (e.g. SHA-256) in order to work properly.

VI. FUTURE WORK

In this section, we highlight any future work that could improve the state of rollup technology.

There are three key factors that need to be more thoroughly studied in order to improve the state of rollup technology and eventually achieve mass-adoption. First of all, decentralizing rollups will allow for users to use the technology in a trustless and secure setting. Secondly, studying alternatives to the fixed withdrawal period in optimistic rollups (e.g., fast-exit liquidity pools) will likely improve their usability. Lastly, with more and more rollups emerging, inter-rollup operability is at stake. EVM equivalent rollups are not directly compatible with EVM compatible rollups. Therefore, transferring funds from one rollup to another might not be easy for users. Some protocols are already offering such services, such as Connext[25] or Hop Exchange,[26] but further work needs to be done to unify rollups.

VII. CONCLUSION

This paper surveys all first and second layer blockchain scaling solutions. We give a brief explanation of every scaling solution and we give an in-depth explanation of rollups and their mode of operation. We also present current implementations or different rollup types. Furthermore, we offer a comparative analysis of these implementations together with their respective features and drawbacks. Finally, we offer some insight on the possible areas of research and development regarding rollups.

REFERENCES

[1] H. Treiblmaier and T. Clohessy, *Blockchain and Distributed Ledger Technology Use Cases*. Springer, 2020.

[25]https://www.connext.network/
[26]https://hop.exchange/

[2] B. K. Mohanta, S. S. Panda, and D. Jena, "An overview of smart contract and use cases in blockchain technology," in *Proc. 9th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2018, pp. 1–4.
[3] P. Zhang, D. C. Schmidt, J. White, and G. Lenz, "Blockchain technology use cases in healthcare," *Adv. Comput.*, vol. 111, pp. 1–41, Jan. 2018.
[4] D. Bumblauskas, A. Mann, B. Dugan, and J. Rittmer, "A blockchain use case in food distribution: Do you know where your food has been?" *Int. J. Inf. Manage.*, vol. 52, Jun. 2020, Art. no. 102008.
[5] P. McCorry, M. Möser, S. F. Shahandasti, and F. Hao, "Towards bitcoin payment networks," in *Proc. Australas. Conf. Inf. Secur. Privacy*. Springer, 2016, pp. 57–76.
[6] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2016, pp. 106–125.
[7] L. M. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *Proc. 41st Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2018, pp. 1545–1550.
[8] V. Buterin. (2014). *Ethereum: A Next Generation Smart Contract & Decentralized Application Platform*. [Online]. Available: https://www.ethereum.org/whitepaper/
[9] A. Yakovenko, "Solana: A new architecture for a high performance blockchain V0. 8.13," White Paper, 2018.
[10] K. Sekniqi, D. Laine, S. Buttolph, and E. G. Sirer. (2020). *Avalanche Platform*. [Online]. Available: https://www.avalabs.org/whitepapers
[11] T. Rocket. *Snowflake to avalanche: A Novel Metastable Consensus Protocol Family for Cryptocurrencies*. Accessed: Apr. 12-2018.
[12] J. A. D. Donet, C. Pérez-Sola, and J. Herrera-Joancomartí, "The bitcoin P2P network," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2014, pp. 87–102.
[13] S. Nakamoto. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://www.bitcoin.org/bitcoin.pdf
[14] A. Hafid, A. S. Hafid, and M. Samih, "Scaling blockchains: A comprehensive survey," *IEEE Access*, vol. 8, pp. 125244–125262, 2020.
[15] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proc. Int. Conf. Manage. Data*, Jun. 2019, pp. 123–140.
[16] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *Proc. 16th USENIX Symp. Networked Syst. Design Implement. (NSDI)*, 2019, pp. 95–112.
[17] A. Hafid, A. S. Hafid, and M. Samih, "A tractable probabilistic approach to analyze sybil attacks in sharding-based blockchain protocols," *IEEE Trans. Emerg. Topics Comput.*, early access, Jun. 7, 2022, doi: 10.1109/TETC.2022.3179638.
[18] M. Bez, G. Fornari, and T. Vardanega, "The scalability challenge of ethereum: An initial quantitative analysis," in *Proc. IEEE Int. Conf. Service-Oriented Syst. Eng. (SOSE)*, Apr. 2019, pp. 167–176.
[19] R. Han, J. Yu, H. Lin, S. Chen, and P. Esteves-Veríssimo, "On the security and performance of blockchain sharding," *Cryptol. ePrint Arch.*, 2021.
[20] R. Han, J. Yu, and R. Zhang, "Analysing and improving shard allocation protocols for sharded blockchains," *Cryptol. ePrint Arch.*, 2020.
[21] M. Hunseler and K. Lemke-Rust, "Simulating an ethereum 2.0 beacon chain network," in *Proc. 8th Int. Conf. Softw. Defined Syst. (SDS)*, Dec. 2021, pp. 1–8.

[22] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," vol. 19, no. 1, pp. 1–6, Aug. 2012.
[23] R. Zhang and W. K. V. Chan, "Evaluation of energy consumption in block-chains with proof of work and proof of stake," *J. Phys., Conf.*, vol. 1584, no. 1, 2020, Art. no. 012023.
[24] A. Fiat, A. Karlin, E. Koutsoupias, and C. Papadimitriou, "Energy equilibria in proof-of-work mining," in *Proc. ACM Conf. Econ. Comput.*, Jun. 2019, pp. 489–502.
[25] W. Li, S. Andreina, J.-M. Bohli, and G. Karame, "Securing proof-of-stake blockchain protocols," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2017, pp. 297–315.
[26] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," Tech. Rep., 2016.
[27] J. Nick, A. Poelstra, and G. Sanders, "Liquid: A bitcoin sidechain," Liquid White Paper, 2020. [Online]. Available: https://blockstream.com/assets/downloads/pdf/liquid-whitepaper.pdf
[28] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," White Paper, 2017, pp. 1–47.
[29] MatterLabs. (2021). *Zkporter: A Breakthrough in L2 Scaling*. [Online]. Available: https://blog.matter-labs.io/zkporter-a-breakthrough-inl2-scaling-ed5e48842fbf
[30] L. D. Negka and G. P. Spathoulas, "Blockchain state channels: A state of the art," *IEEE Access*, vol. 9, pp. 160277–160298, 2021.
[31] A. Singh, K. Click, R. M. Parizi, Q. Zhang, A. Dehghantanha, and K.-K.-R. Choo, "Sidechain technologies in blockchain networks: An examination and state-of-the-art review," *J. Netw. Comput. Appl.*, vol. 149, Jan. 2020, Art. no. 102471.
[32] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev. (2018). *Scalable, Transparent, and Post-Quantum Secure Computational Integrity*. [Online]. Available: https://eprint.iacr.org/2018/046.pdf
[33] J. Groth, "On the size of pairing-based non-interactive arguments," in *Proc. Annu. Int. Conf. Theory Appl. Cryptol. Technol.* Springer, 2016, pp. 305–326.
[34] T. Chen, H. Lu, T. Kunpittaya, and A. Luo, "A review of Zk-SNARKs," 2022, *arXiv:2202.06877*.
[35] G. Danezis, C. Fournet, M. Kohlweiss, and B. Parno, "Pinocchio coin: Building zerocoin from a succinct pairing-based proof system," in *Proc. 1st ACM Workshop Lang. Support Privacy-Enhancing Technol.*, 2013, pp. 27–30.
[36] M. Petkus, "Why and how Zk-SNARK works," 2019, *arXiv:1906.07221*.
[37] C. Sguanci, R. Spatafora, and A. Mario Vergani, "Layer 2 blockchain scaling: A survey," 2021, *arXiv:2107.10881*.
[38] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf
[39] V. Buterin. (2021). *An Incomplete Guide to Rollups*. [Online]. Available: https://vitalik.ca/general/2021/01/05/rollup.html
[40] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. (2013). *Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture*. [Online]. Available: https://eprint.iacr.org/2013/879.pdf
[41] C. Guo, F. Jiang, A. Kattis, L. Mayer, H. Qian, and Y. Sagiv. (2015). *Arbitrum: Blockchain-Based Arbitration*. [Online]. Available: https://www.youtube.com/watch?v=BpzrLOk4Zy0
[42] D. Wang, J. Zhou, A. Wang, and M. Finestone. (2018). *Loopring: A Decentralized Token Exchange Protocol*. [Online]. Available: https://loopring.org/resources/en_whitepaper.pdf
[43] H. Kalodner, S. Goldfeder, X. Chen, S. M. Weinberg, and E. W. Felten, "Arbitrum: Scalable, private smart contracts," in *Proc. 27th USENIX Secur. Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, 2018, pp. 1353–1370. [Online]. Available: https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-kalodner.pdf
[44] A. Gabizon, Z. J. Williamson, and O. Ciobotaru. (2021). *Plonk: Permutations Over Lagrange-Bases for Oecumenical Noninteractive Arguments of Knowledge*. [Online]. Available: https://eprint.iacr.org/2019/953.pdf
[45] P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, "Flash boys 2.0: Frontrunning, transaction reordering, and consensus instability in decentralized exchanges," 2019, *arXiv:1904.05234*.
[46] Optimism. (2022). *Transaction Fees*. [Online]. Available: https://help.optimism.io/hc/en-us/articles/4411895794715-Transaction-fees
[47] StarkWare. (2020). *Stark over MainNet*. [Online]. Available: https://medium.com/starkware/starks-over-mainnet-b83e63db04c0
[48] (2020). *The Great Reddit Bake-Off*. [Online]. Available: https://www.reddit.com/r/ethereum/comments/hbjx25/the_great_reddit_scaling_bakeoff/
[49] B. Jiang. (2022). *Huobi Research Rollup Report*. [Online]. Available: https://blog.huobi.com/wp-content/uploads/Huobi-Research-Rollup-Report-March-2022.pdf
[50] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," 2003, *arXiv:0301141*.

LOUIS TREMBLAY THIBAULT received the B.Sc. degree in computer science from the University of Montreal, in 2022, where he is currently pursuing the master's degree under the supervision of Abdelhakim Senhaji Hafid. His research interests include blockchain technology, applied cryptography, and formal verification.

TOM SARRY received the B.Sc. degree in computer science from McGill University, in 2022. He is currently pursuing the M.Sc. degree in computer science with the University of Montreal. His main research interests include blockchain systems and usability of second layer scaling solutions.

ABDELHAKIM SENHAJI HAFID (Member, IEEE) was an Assistant Professor with Western University (WU), Canada, the Research Director of the Advance Communication Engineering Center (venture established by WU, Bell Canada, and Bay Networks), Canada, a Researcher with CRIM, Canada, a Visiting Scientist with GMD-Fokus, Germany, and a Visiting Professor with the University of Evry, France. He is currently a Full Professor with the University of Montreal. He is also the Founding Director of the Network Research Laboratory and the Montreal Blockchain Laboratory, and a Research Fellow with CIRRELT, Montreal, QC, Canada. Prior to joining the University of Montreal, he spent several years, as a Senior Research Scientist at Bell Communications Research (Bellcore), Piscataway, NJ, USA, working in the context of major research projects on the management of next generation networks. He has extensive academic and industrial research experience in the area of the management and design of next generation networks. His current research interests include the IoT, fog/edge computing, blockchain, and intelligent transport systems.