ETHWORKS

Ethworks Reports

Zero-Knowledge Blockchain Scalability

SUMMER '20

Table of Contents

01

Introduction

Oh, Boy... Scaling Again...

You probably know the story all too well. When **Bitcoin sneaked into the mainstream media** back in 2017, and more people than ever became interested in the blockchain, the world saw the major challenge connected with the technology. As the platform became popular, and loads of transactions were carried out on it, the network congestion reached a new, previously unseen level. This led to skyrocketing transaction fees, effectively making bitcoin transfers impractical. The problem has resurfaced recently when the **gas prices on Ethereum have increased** 30-50 times compared to the “boring times”.

All this demonstrates what the community had known before it happened: that the original design of the blockchain makes it unsuitable for mass use. Its maximum throughput of just a couple of transactions per second positions it nowhere close to mainstream networks such as Visa or PayPal. This limitation is often referred to as **the blockchain scalability problem.**

Ethereum was designed to confirm transactions faster than Bitcoin, but it’s still far from solving the problem. Its transaction throughput—higher than the one of Bitcoin, but still of no more than several dozen TPS—is not enough to make the network suitable for a mass audience. It tends to be congested, and even simple smart contract operations can cost a couple of dollars.

The Real Blockchain Throughput

Due to the design of the **Bitcoin** platform, **its theoretical throughput reaches only 7 transactions per second**. In practice, because of issues such as empty blocks, complex input and output transactions, etc., the average number used to be just 3. After the SegWit protocol update in 2017 that required a soft fork, it grew to 4.6, which is still fewer than the already-not-very-impressive 7.

Similarly, **the original transaction throughput of Ethereum was 15 TPS**. However, this limit is dynamic (determined by the gas limit set by miners). With the current network configuration of 12 mln block gas limit, one could argue that the real throughput reaches over 36 TPS. Nonetheless, this statement is justified only in reference to ether transactions as token transfers tend to be 2.5 times more expensive, not to mention more complex interactions with smart contracts—they're typically even more costly.

The blockchain space has seen **many scalability projects failing**. Or succeeding only on (white)paper. Or bending the definition of the blockchain itself. It would be justified to acknowledge that there is no ultimate solution to the problem, and that new solutions will always add extra complexity and roughness.

Even after accepting this reality, we saw relatively little progress in the scalability field in the past years. However, finally, it seems there's light at the end of the tunnel, and **things are about to change.**

We saw a revolution just around the corner earlier this year when one of our customers asked us to carry out a research on the scalability market. More recently, the teams leading several promising projects have announced their milestones, and the buzz in the Ethereum community started to rise.

We know that the excess of information may be misleading, making it hard to understand what the whole concept is about, so **we've decided to give you a hand**. With this report, we'd like to share the findings of our scalability research with blockchain developers and a broader blockchain community. We believe that it'll help you understand the potential of zero knowledge and L2 scaling solutions and give you a dive deep into the technology behind them.

Report Contents

The scalability market has already grown and got highly fragmented with **countless companies working on such solutions** right now. It's easy to get lost in the sea of these technologies, and identifying the most promising ones may be really tough.

That's why we've decided to take a closer look at them and **compare the ones that we find most promising** to help entrepreneurs and developers alike make the right choices. We're focusing on **L2 zero-knowledge-based solutions** as we find them to have the biggest long-term potential for their high security and relatively short exit times.

There are at least two other promising categories we're not describing here: optimistic rollups and state channels. They're not based on zero knowledge and, thus, are beyond the scope of this report.

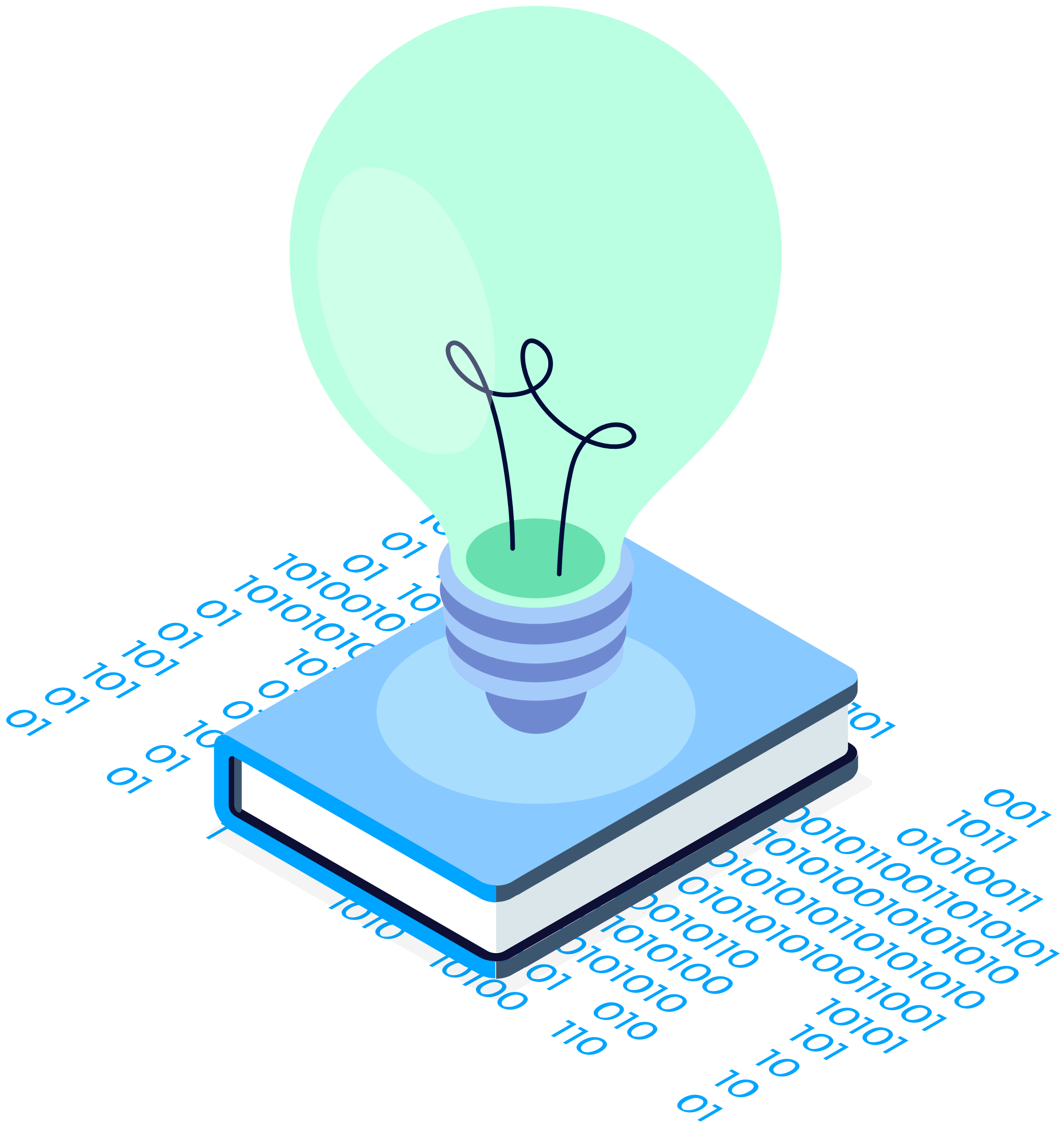
What Are L2 Solutions?

Layer 2 scalability solutions aim to increase Ethereum's maximum throughput (the speed of processing transactions) and reduce the transaction fees paid by end users by **adding additional protocols** to the existing blockchain. Unlike L1 solutions (such as Ethereum 2.0), they don't try to change the Ethereum consensus algorithm or any other core concepts—they use the base protocol as a decentralized security layer on top of which another one is built.

L2 solutions **offload Ethereum** by moving computations outside of it while using the blockchain as a source of security. Thanks to performing these operations off-chain the amount of data stored on the root chain gets significantly reduced, which results in cheaper and potentially faster transactions.

02

Zero Knowledge



About Zero Knowledge

The solutions described in detail in this report utilize zero-knowledge cryptography to ensure transaction security and facilitate off-chain computations. Therefore, before diving into the scalability solutions themselves, one needs to have a basic understanding of the zero knowledge technology.

Zero knowledge (abbreviated to ZK) is a division of cryptography that has been an object of great interest in the blockchain community for several years now. Zero-knowledge proofs are used to prove to one party (the verifier) that another party (the prover) possesses some knowledge but without revealing the knowledge itself or any other information that might be used to reconstruct it. The only information conveyed and proven to the verifier is that the prover does possess this knowledge.

Confusing? Let us give you an example of a simple non-mathematical zero-knowledge proof. →

Real-Life Example

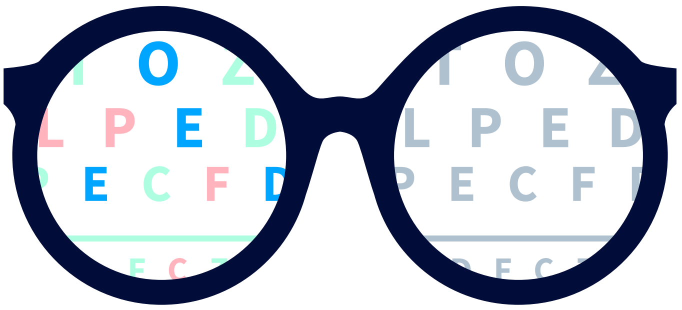
Konstantinos Chalkias came up with an excellent **explanation of how zero-knowledge proofs work**. Imagine Victor and Peggy: a couple of friends that enjoy juggling balls in a park at the weekends. Victor is color-blind which means that he doesn't distinguish red and green colors. For him, they're identical.

Our friends decided to go to Central Park to do their standard juggling routine. Peggy took her own balls, one of them green and the other red. Due to his color-blindness, **Victor cannot see any difference** between them.

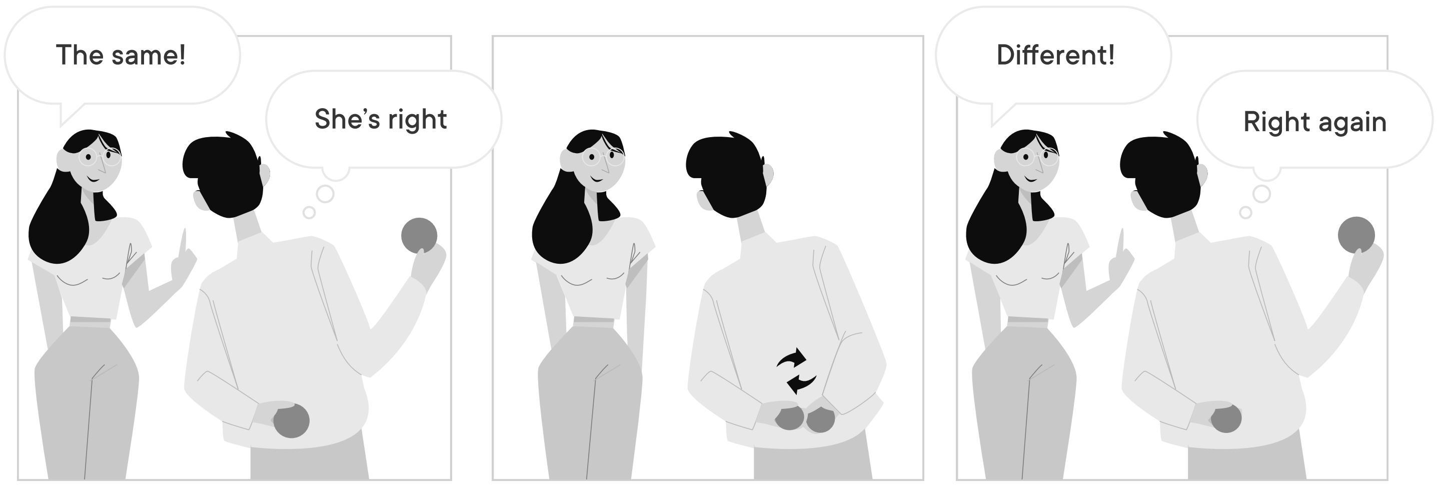
To prove the difference to Victor, Peggy asks him to grab one ball in each hand and put them behind his back. Then, he repeatedly switches them in his hands and displays one of the two to Peggy, asking if it is different from the previous one.

Every time Victor switches the balls, Peggy is able to tell if the one he's displaying is different because she distinguishes red and green colours. Thus, **Victor can be pretty sure that Peggy knows the difference** between the balls although, for him, these balls are identical. The color of each ball is never revealed to him, which forms a zero-knowledge proof.

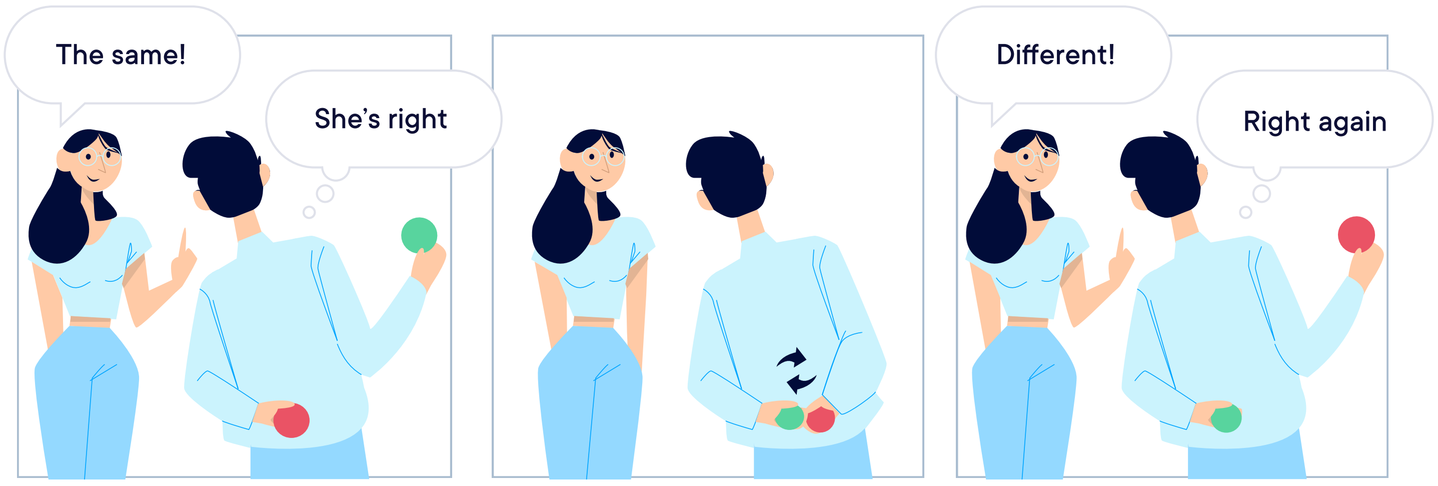
Although she can guess it once or twice with a fairly good probability of 50%, the more they repeat the process, the closer the probability is to impossibility. Therefore, Victor can be pretty sure that the balls are of two different colors.



What Victor (verifier) sees



What Peggy (prover) sees



Zero Knowledge and Blockchain

Zero-knowledge encryption can be used to generate **cryptographic proofs that some computation has been performed** in accordance with predefined rules. These proofs are generated programmatically and verified automatically.

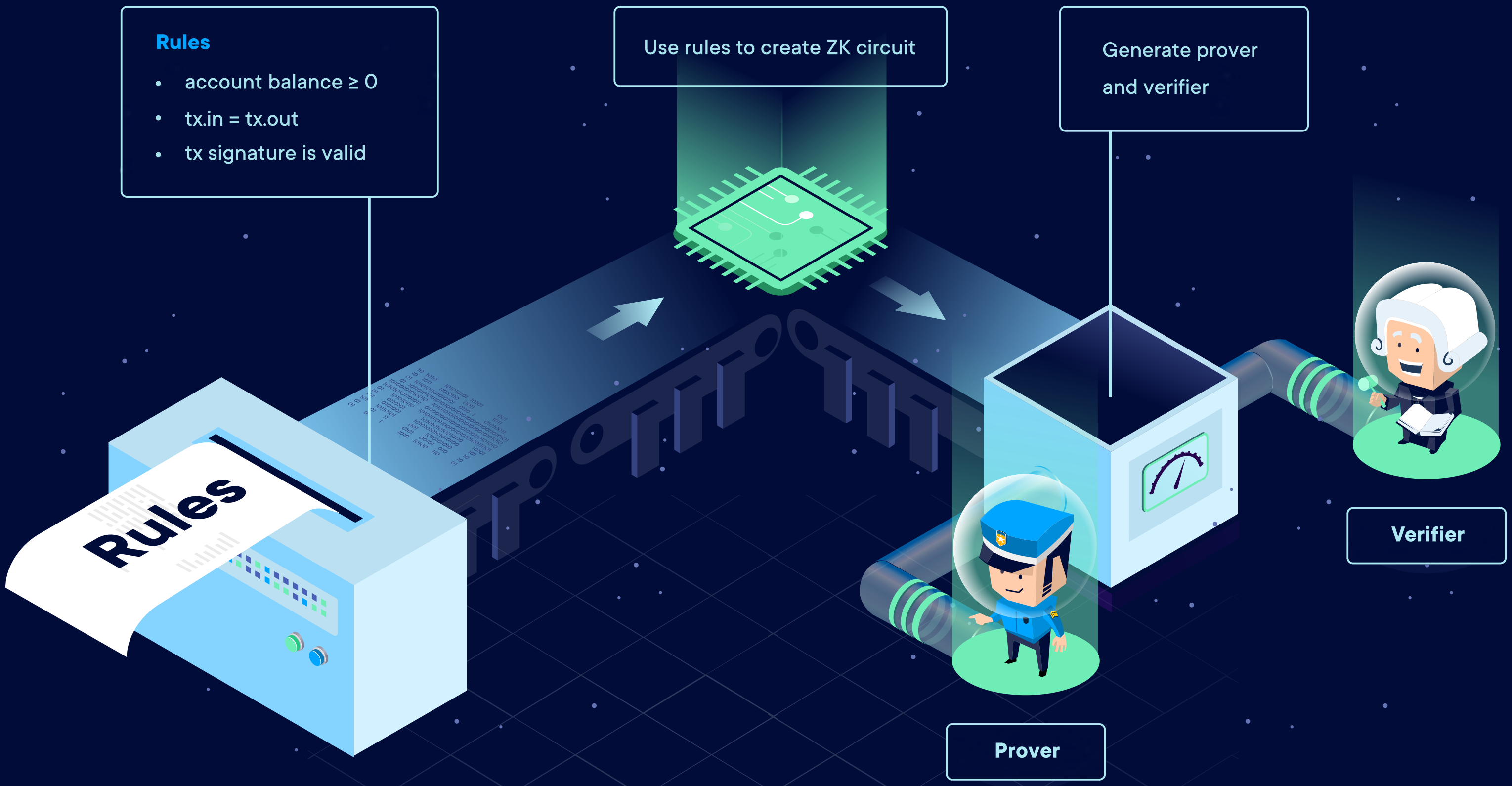
The scaling potential of the zero-knowledge encryption lies in the proofs themselves: **they're significantly smaller** than the data they represent, and verifying them is relatively cheap. Additionally, thanks to their properties, they can be used for anonymizing transactions.

ZK-based scalability solutions don't really modify the fundamental scale of blockchains. They rather **change the purpose behind them**: instead of computing small payloads on the main chain, they verify exponentially larger payloads computed outside of it.

See the example →

Creating a zero-knowledge proof within an L2 scaling solution starts with **a set of predefined rules**. In the case of a transaction system, they could be similar to blockchain consensus rules, e.g. that each transaction needs to have a correct signature or that users cannot spend more than they possess.

The system transforms the rules into a series of intermediate mathematical representations (circuits and then polynomials) that are later used to create two computer programs that are necessary in the zero-knowledge process: **the prover and the verifier.**

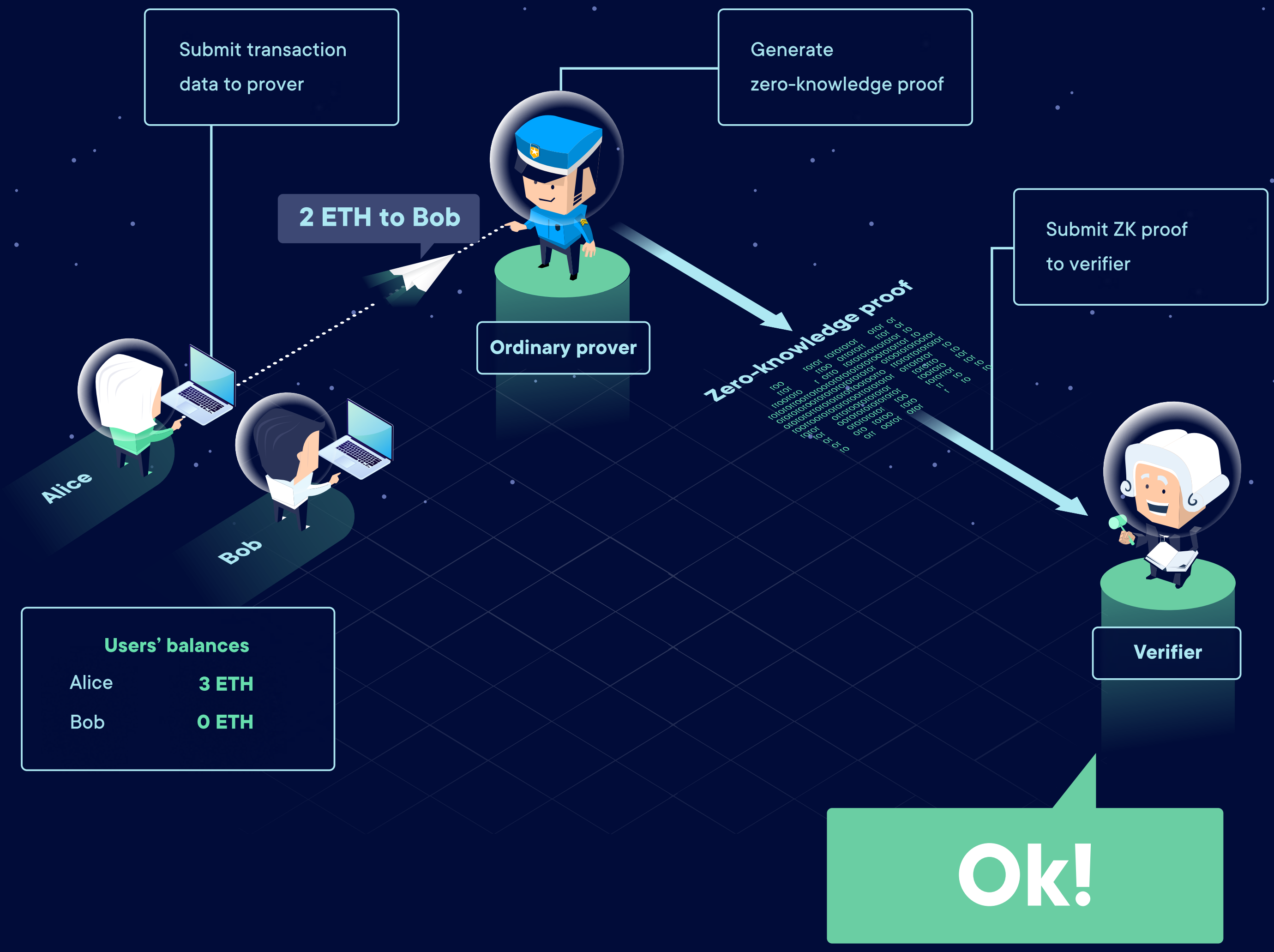


Once we have the prover and the verifier, we can use the system to generate and verify transactions. Imagine Alice who has 3 ETH in her wallet and wants to send some funds to Bob. She signs her transaction, and the transaction data is submitted to the prover. The program uses it to generate a zero-knowledge proof which is later sent to the verifier. Now, let's consider two scenarios.

→

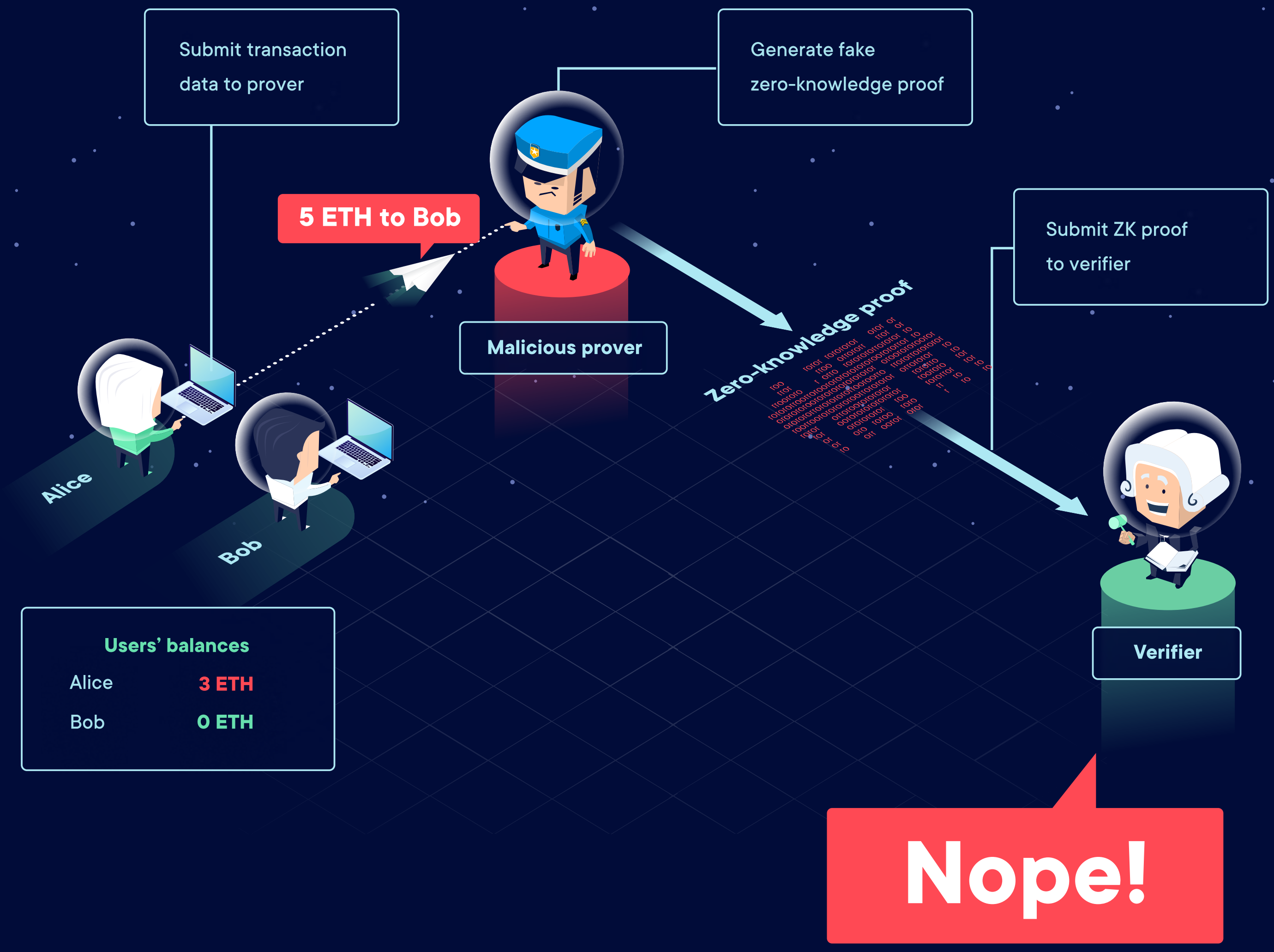
Generating a VALID Zero-Knowledge Proof

Without knowing anything about the transaction itself, the verifier checks whether Alice carried it out according to the predefined rules. One of them says that she can't spend more funds than she has, so if she wants to send **2 ETH**, the verifier will accept the operation.



Generating a FAKE Zero-Knowledge Proof

Now, imagine that Alice wants to send **5 ETH**. The transaction should be rejected at the prover stage, but even if the program turns out to be malicious, at the end of the day, the transfer won't get accepted because the verifier won't verify it. That's because it wasn't carried out according to the rules.



SNARKs vs. STARKs

In relationship to zero knowlage, several cryptographic proofs have emerged so far, with SNARKs and STARKs being the most popular names mentioned. The relationship between them all is non-trivial.

SNARK stands for:

- **s**uccinct: the proof is significantly smaller than the data it represents and can be verified quickly,
- **n**on-interactive: only one set of information is sent by the prover to the verifier, thus there's no back-and-forth interaction between them,
- **ar**gument of **k**nowledge: the proof is considered computationally sound—a malicious prover isn't likely to cheat the system without possessing the knowledge to support its statement.

SNARKs used in scalablility solutions require a **trusted setup** between the prover and the verifier. It's a set of initial public parameters that resemble the rules of a game. They're generated during a so-called trusted setup ceremony. It's a joint computation performed in an arranged time by a group of voluntary participants. As long as at least one of them behaves honestly, the parameters are generated securely. Thus, the more parties participate, the more "trusted" the ceremony is.

STARKs (e.g. FRI-AIR STARK developed by StarkWare) **don't require trusted setup** — thus the "T" that stands for "transparent". This eliminates the potential single point of failure for the whole system. Even though the proof size is larger for STARKs, the amortized computation cost is lower for big transaction batches. Therefore, they allow to achive higher scalablity.

Solutions based on the early SNARK technology (i.e. Groth16) require conducting the ceremony for every new version of the product. That's why Loopring described in our report later on needed to conduct one before launching the latest version of their protocol last year.

Another variant called Universal SNARKs or SNORKs (e.g. PLONK and SONIC), leverages **universal trusted setup**. For example, zkSync creators didn't have to conduct their own ceremony while launching the product: they re-used the ignition multi-party computing performed last year with approx. 200 reputable figures such as Vitalik Buterin. Universal trusted setup also allows them to extend and upgrade the zero-knowledge part of the protocol without conducting another ceremony.

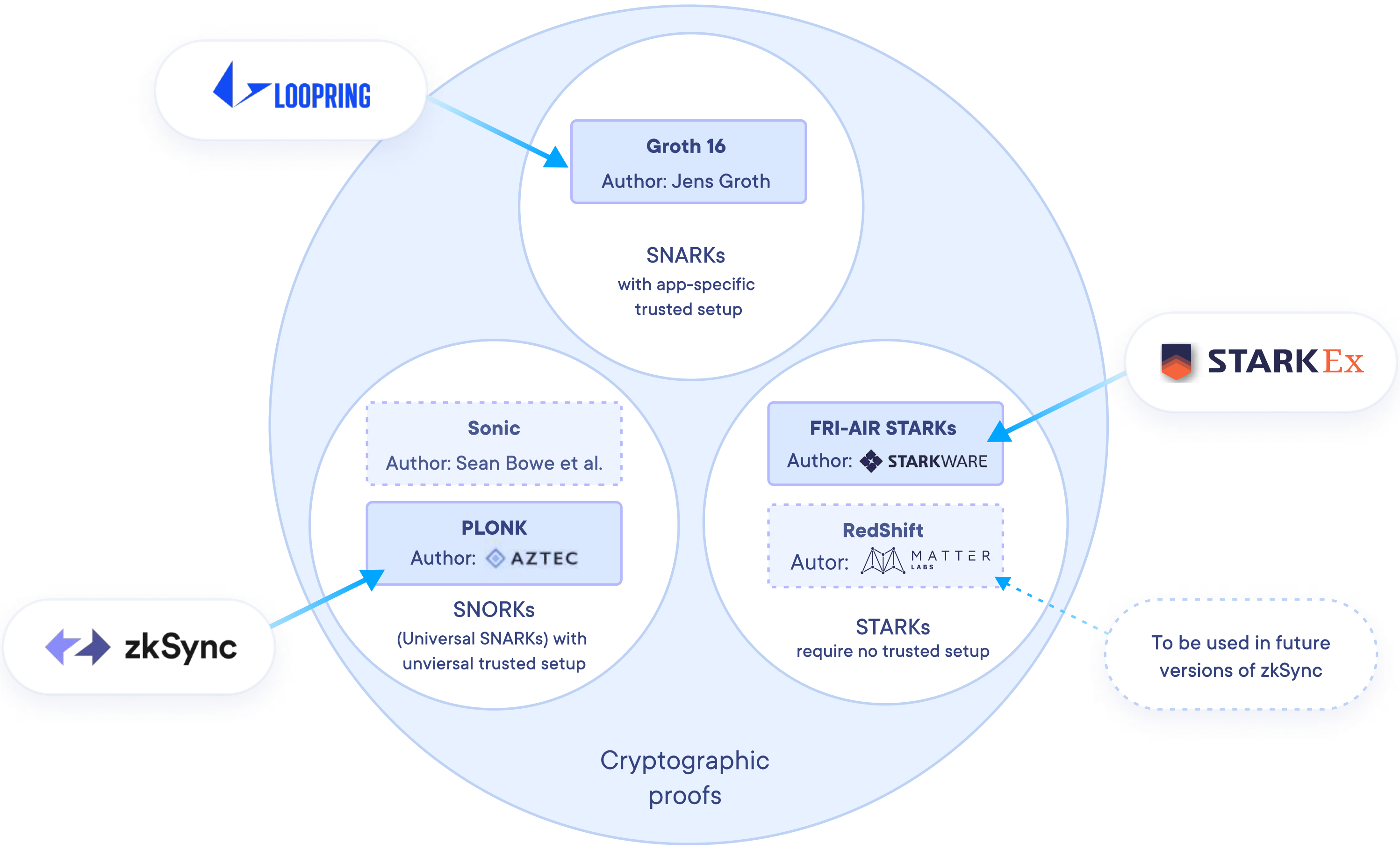


Diagram presenting the relationship between various cryptographic proofs

Image courtesy of Alex Gluchowski

03

Architectures

The common denominator of the solutions described in our report is the use of zero-knowledge cryptography. What makes them diverse, by contrast, is the data availability issue.

Data Availability Problem

Transaction data and information about users' balances may be held on the blockchain or outside of it, which results in the **fundamental trade-off between scalability and security.**

Storing data on-chain has similar security guarantees as having assets directly on Ethereum without additional actions taken on the user side. It makes the data available for them any time, which gains importance when a scalability solution provider's server ceases to exist or turns malicious. On-chain data availability enables users to construct a proof that they hold a certain amount of tokens and withdraw them directly from the smart contract without interacting with the system. Zero-knowledge-based solutions keeping data on-chain are referred to as **zkRollups.**

The scalability solutions **storing data off-chain** weaken Ethereum's security guarantees by introducing the data availability problem. When a scalability solution provider stops collaborating, regular users cannot withdraw their funds unless they have access to the data representing their balances. Such solutions are called **validiums**. To mitigate the data availability problem, they may introduce **multi-party committees** responsible for storing the copies of the data and sharing them with users in case of malicious or uncooperative behaviour.

There’s, however, a significant advantage of storing data off-chain: **higher scalability.** Solutions taking advantage of such storage aren’t subject to their blockchain’s limitations. Thanks to that, the potential for increasing transaction throughput is higher than in the case of on-chain storage.

Recently, StarkWare proposed a **hybrid solution** that could allow users to pick whether their data will be stored on-chain or off-chain. They can decide on that for every single transaction, which makes the choice dynamic. Such scalability solutions are referred to as **volitions.**

	Naming	
On-chain data	Volition	zkRollup
Off-chain data		Validium

Source: *Volition and the Emerging Data Availability spectrum,* StarkWare,

zkRollup

zkRollups use **operators**: servers that process users' transactions. They act as provers, collecting transaction input data and transforming them into lighter zero-knowledge proofs. Computations happen off-chain, so transactions are not processed by a smart contract. It only verifies the light cryptographic proof, and that's what reduces transaction fees.

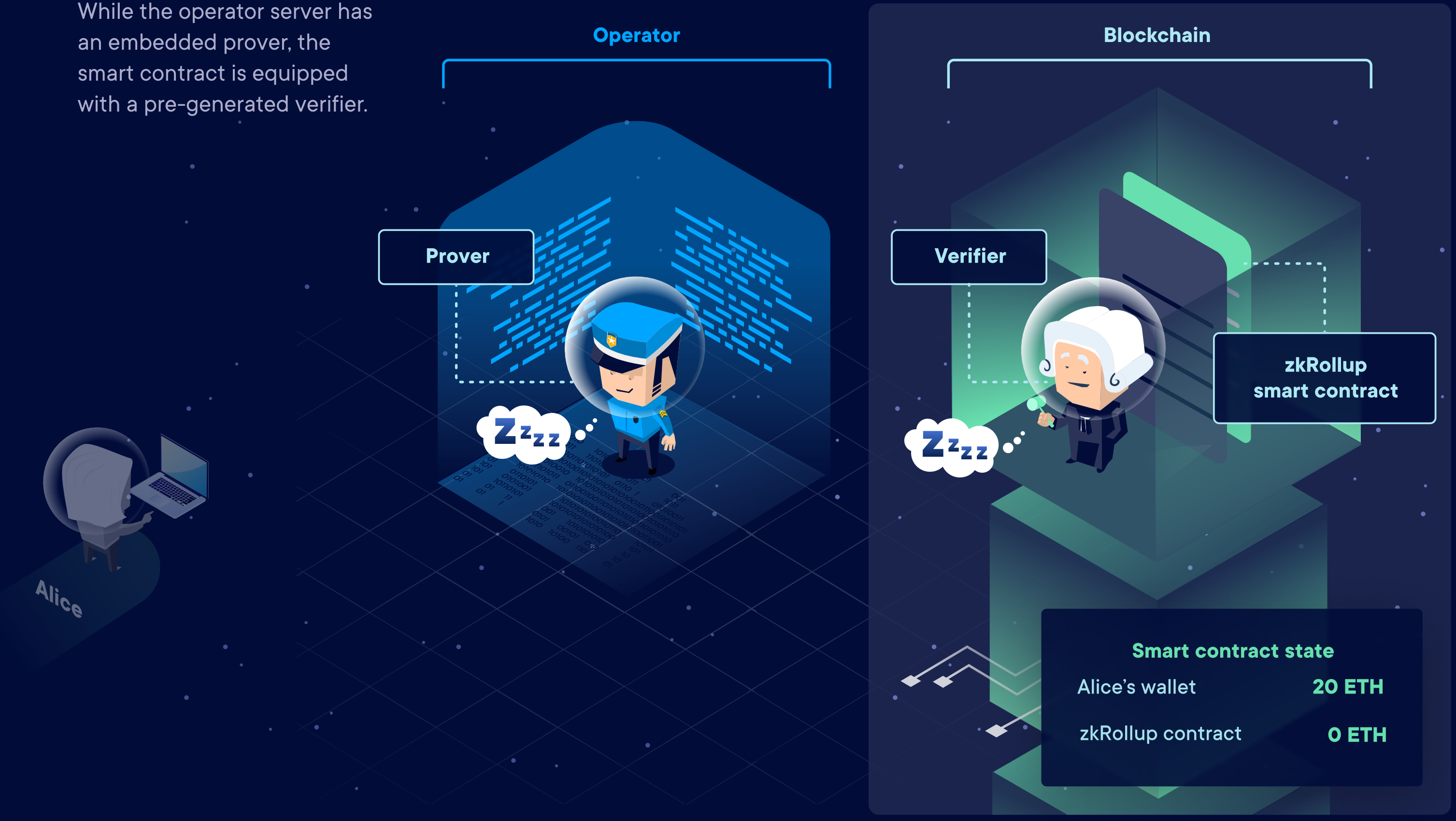
Several important features make up for the high security of zkRollups:

- **operators cannot steal users' funds or corrupt the system in any way:** thanks to zero-knowledge validity proofs, the network can only function in a valid state (to change it, another validity proof would have to be constructed),
- **there's no need for the users to monitor the network:** unlike in Plasma or optimistic rollups that require the users to detect any signs of misbehaviour, zkRollups store all data on-chain and always require validity proofs; thanks to that, it is impossible for an operator to cheat,
- **users can always withdraw their funds onto the mainnet** without any cooperation with operators: data availability enables zkRollups users to construct a proof that they hold a certain amount of tokens and withdraw them directly from the smart contract.

Let's take a look at the zkRollup flow →

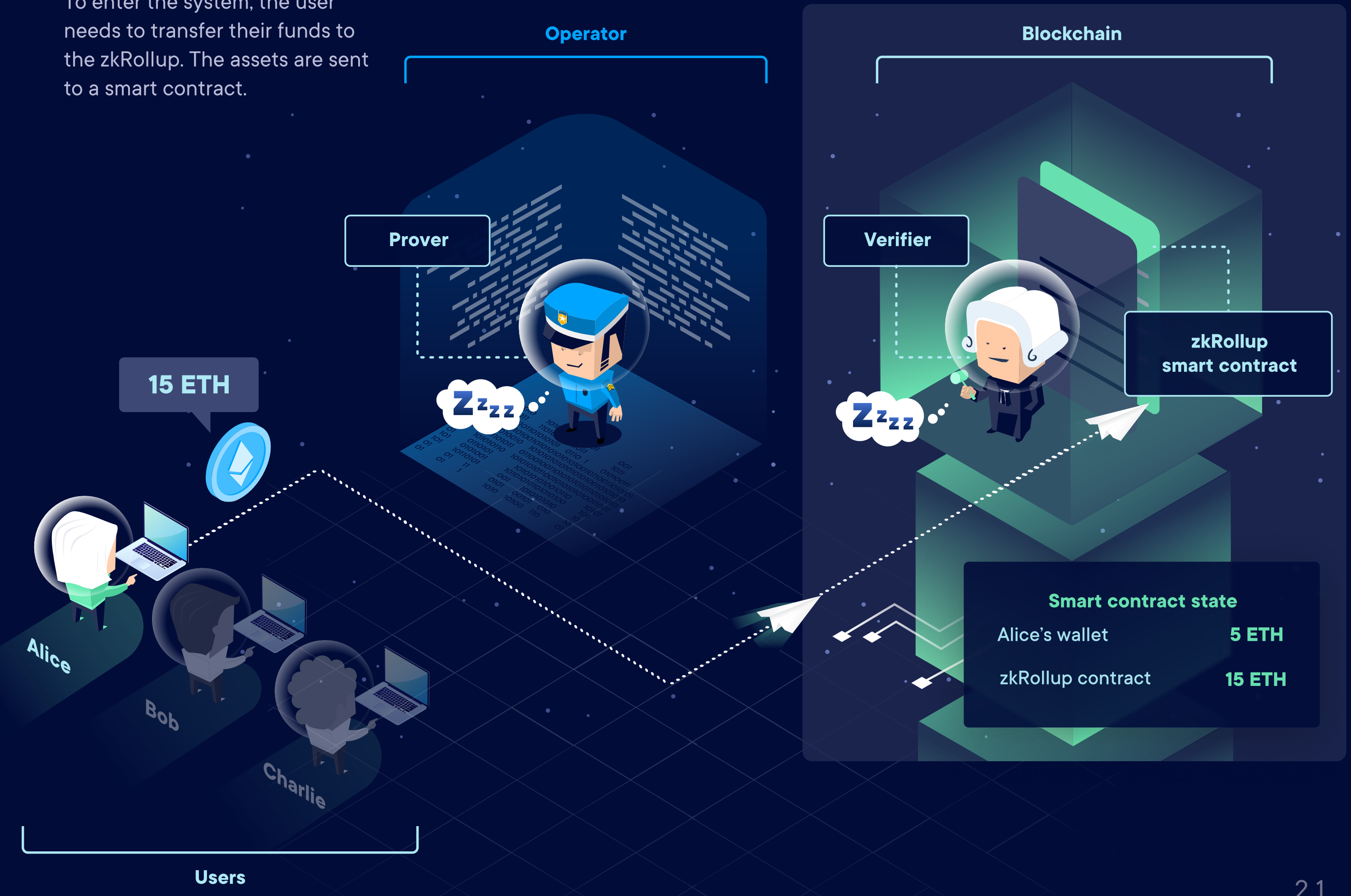
01. Start

While the operator server has an embedded prover, the smart contract is equipped with a pre-generated verifier.



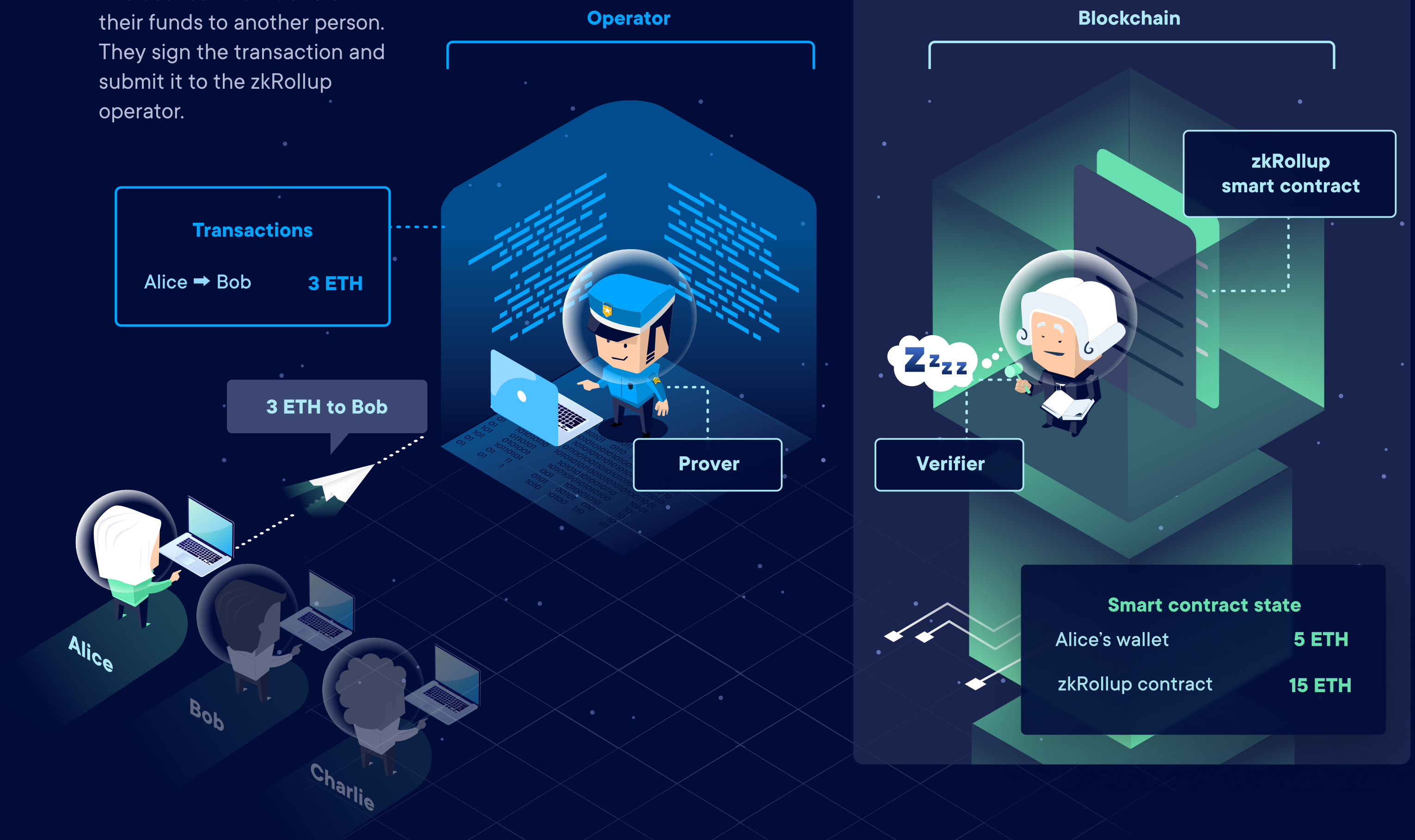
02. Alice's Enter

To enter the system, the user needs to transfer their funds to the zkRollup. The assets are sent to a smart contract.

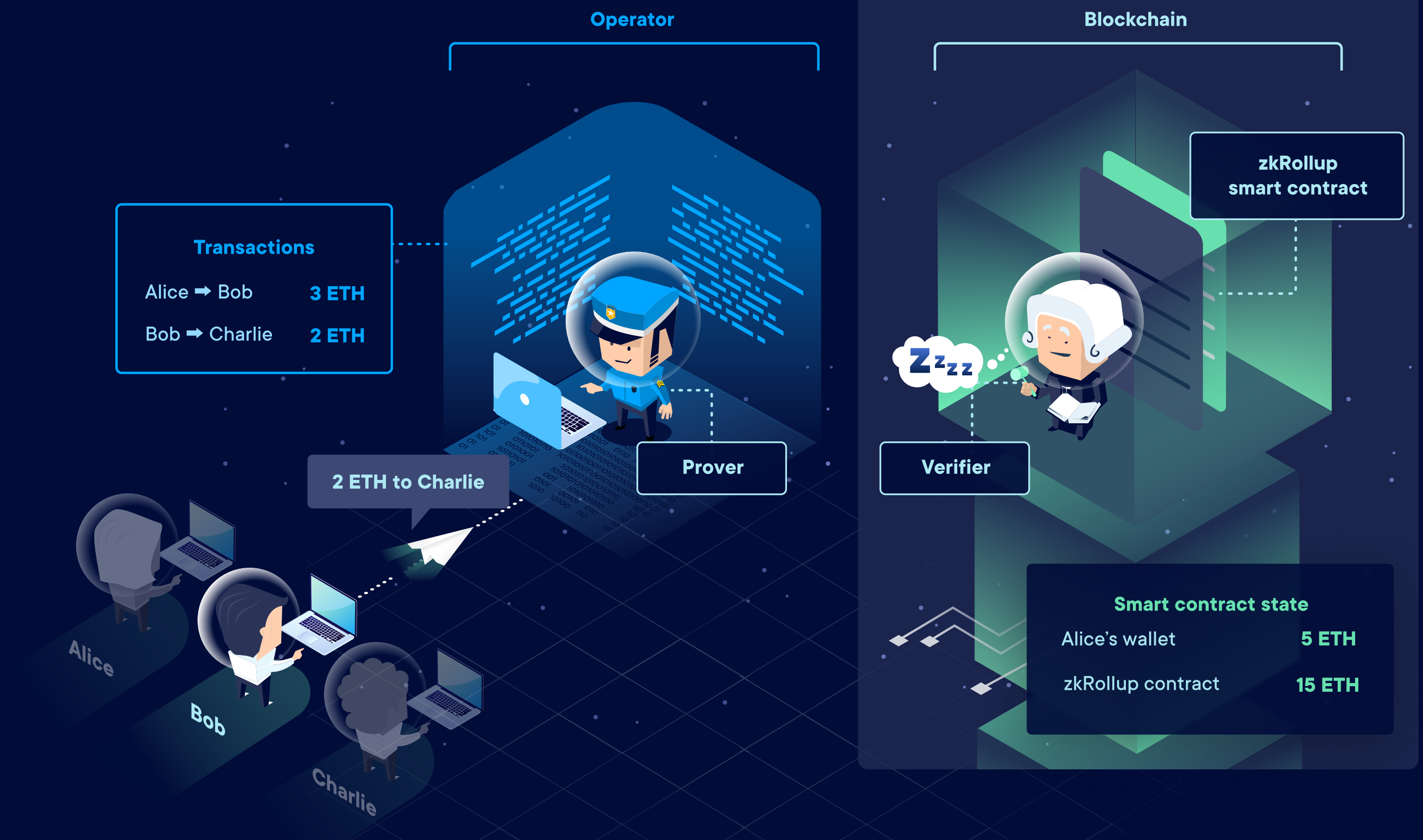


03. Alice's Transfer

The user can now transfer their funds to another person. They sign the transaction and submit it to the zkRollup operator.

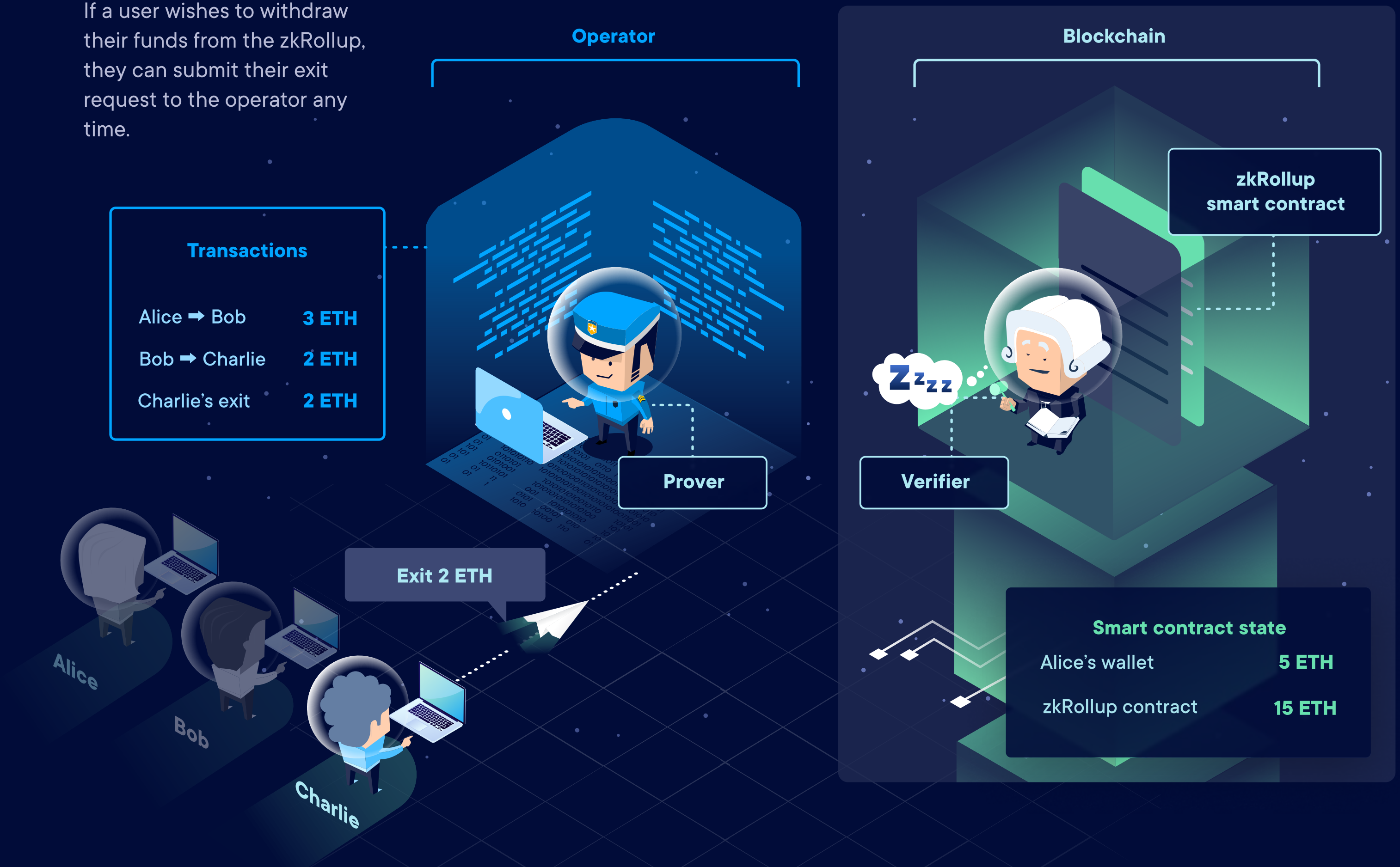


04. Bob's Transfer



05. Charlie's Exit

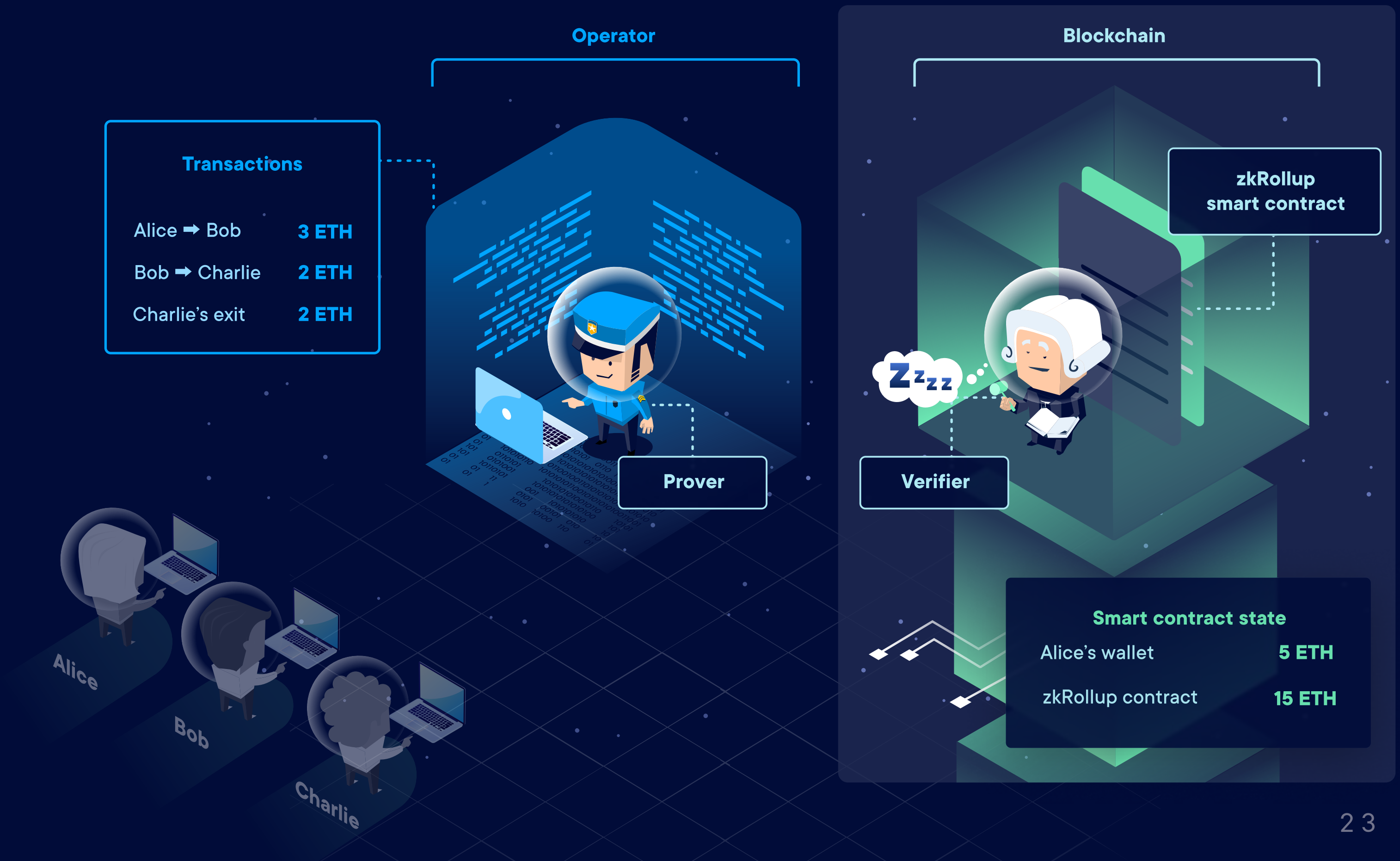
If a user wishes to withdraw their funds from the zkRollup, they can submit their exit request to the operator any time.



06. Collecting Transactions

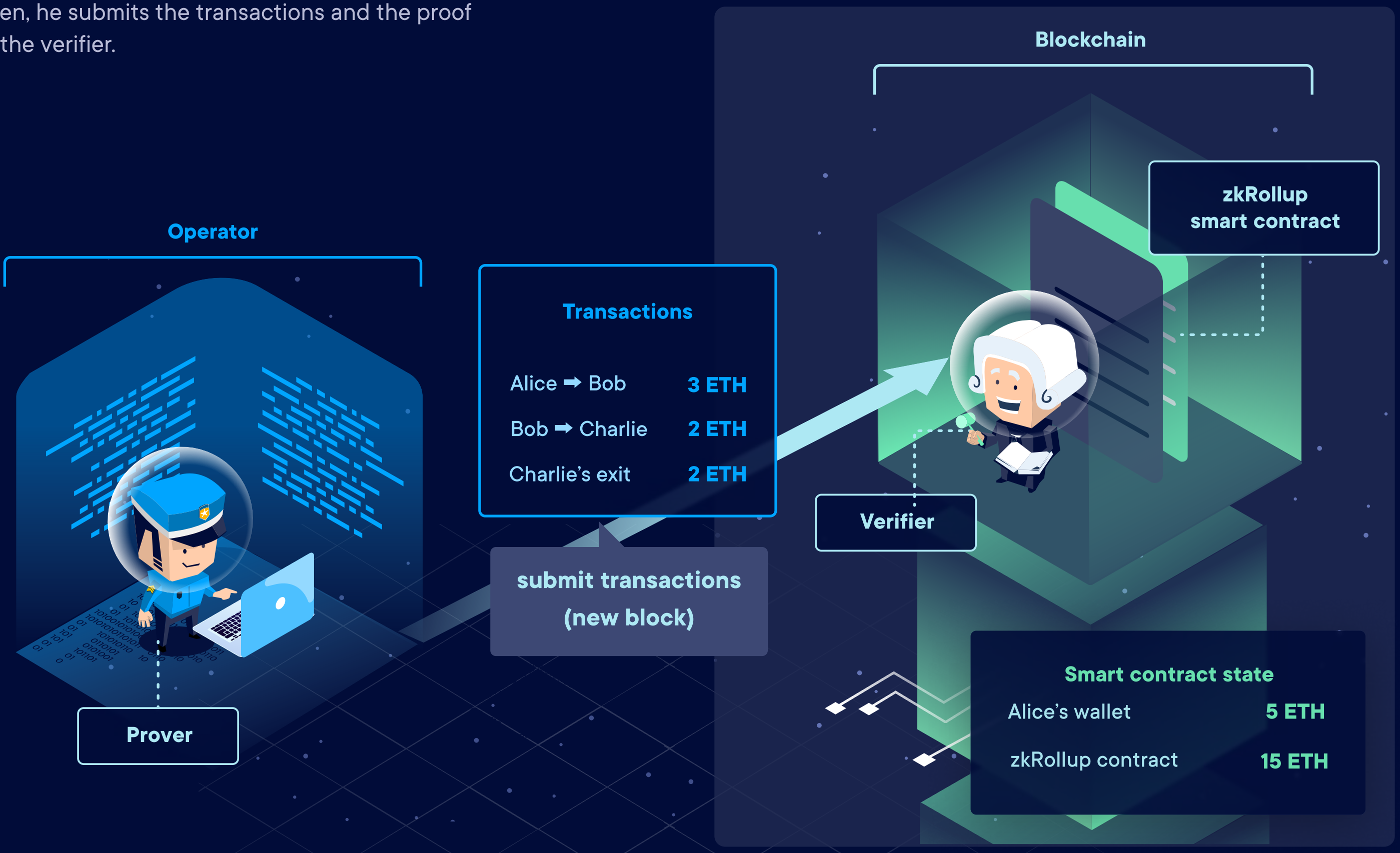
In the meantime, the operator collects transactions and exit requests from many users.

* Note that even if Bob and Charlie didn't have any funds on the zkRollup, they could still receive transfers from other users.



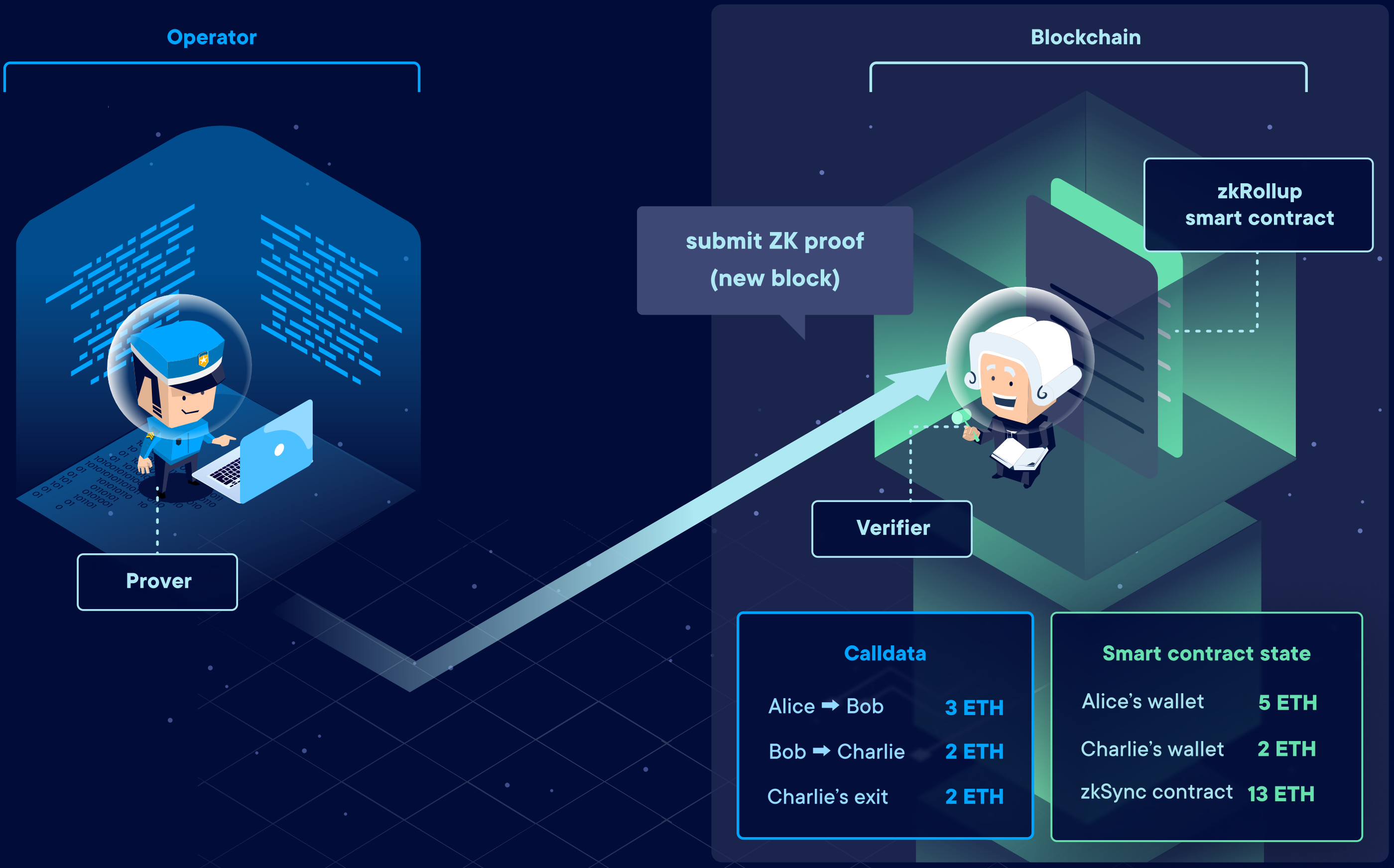
07. Submitting Transactions

Once in a while, the operator bundles the collected transactions together and generates a ZK proof. Then, he submits the transactions and the proof to the verifier.



08. Submitting ZK Proof

The smart contract verifies the transactions and the proof. Once it's done, the transactions are finalized.



To fully comprehend zkRollups' scaling potential, one needs to understand two mechanisms first.

- The operator pays for **storing the proof** in the blockchain state and verifying it, which is much cheaper than storing and processing the transactions themselves.
- The transaction data is still **stored on the blockchain**, yet not in the blockchain state but in transaction data (**calldata**), which is much cheaper, too.

What Makes Calldata Cheaper

Calldata is not stored as a part of the blockchain state. It means that you can run a secure node (a full or fully validated one) without storing the history.

This technological nuance is what makes zkRollups efficient: **there's no need for full nodes to store this memory,** although a fully validated node needs to fast forward the whole history before it can be considered secure.

The blockchain state needs to be searchable. It is stored in a data structure called Merkle tree which requires fast access. Therefore, it's usually **stored on fast SDD drives**. Transaction and history data can be **stored on cheaper and slower HDD drives** or even in a cheap cloud.

This difference is so significant that it's not the network latency or the cost of disk space in general that is considered **the biggest scalability bottleneck of Ethereum**. It's the SSD drives speed and the access to the Merkle-tree-based database.

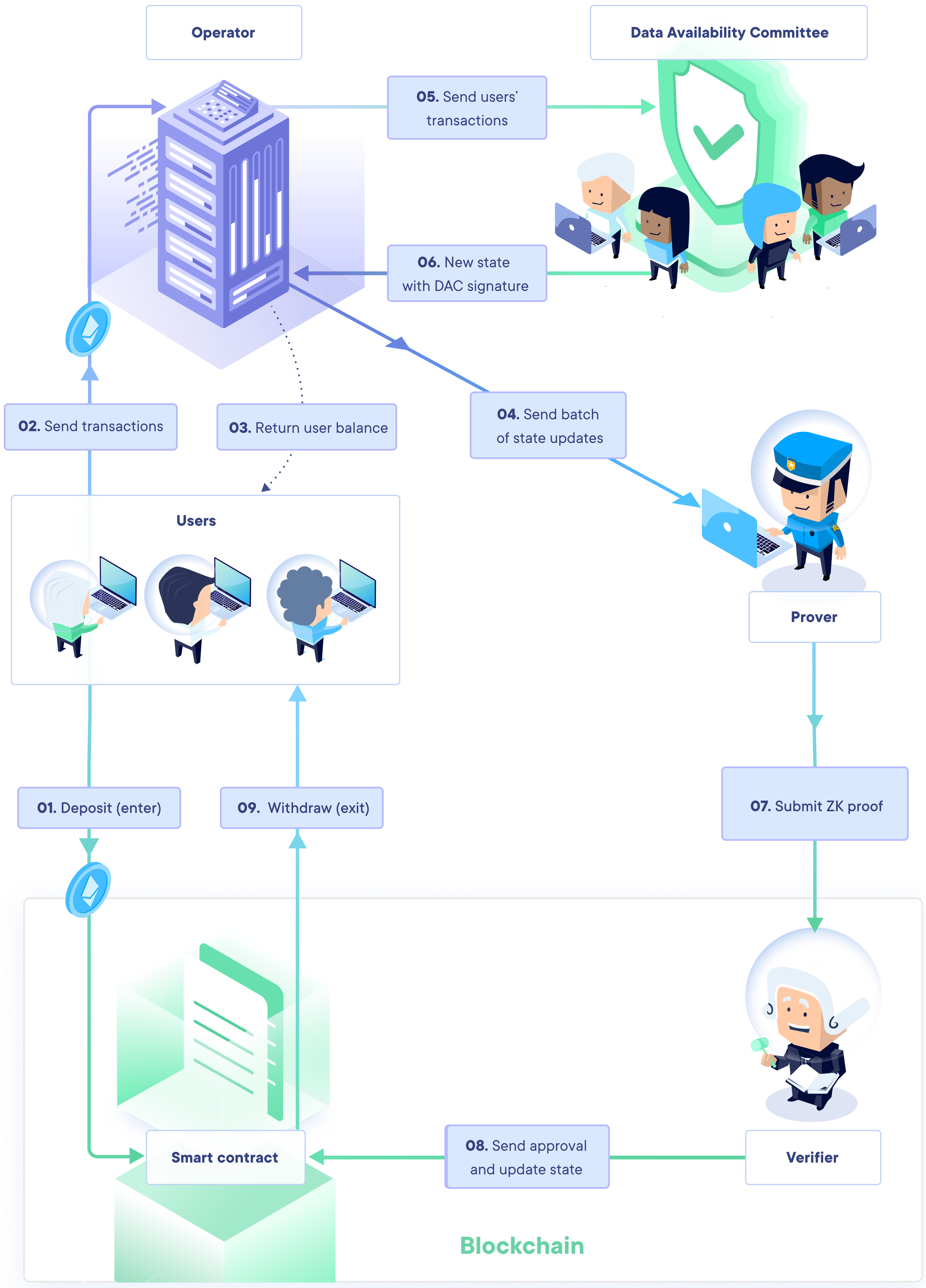
Validium

We'll explain how validiums work on the example of StarkEx as integrated in DeversiFi (a decentralized exchange running on the mainnet). According to our best knowledge, it's the only product that supports the validium operation mode for now.

Validiums leverage **operators** just like zkRollups. These generate light zero-knowledge proofs that are transferred to the validium smart contract.

As validiums hold the information about users' balances off-chain, they provoke **the data availability problem**. To mitigate it, StarkEx introduced another actor participating in operations: the Data Availability Committee (DAC) responsible for holding the copies of the data. It alleviates the emergency exit concern by sharing the copies with users' when the operator turns malicious.

Let's take a look at the validium operation flow. →


Operator
Data Availability Committee
05. Send users' transactions
06. New state with DAC signature
02. Send transactions
03. Return user balance
04. Send batch of state updates
Users
Prover
01. Deposit (enter)
09. Withdraw (exit)
07. Submit ZK proof
Smart contract
08. Send approval and update state
Verifier
Blockchain

How it works?

1. To deposit their funds, users send them to the validum smart contract. The operator watches for smart contract changes and updates the off-chain state.

2. To perform off-chain operations, users send their requests to the operator.

3. The operator updates the off-chain state and shows appropriate balances to the users.

4. The operator batches state transitions up and sends them to the prover.

5. The operator verifies the requests, computes the new state, and sends them to the Data Availability Committee (DAC).

6. The DAC signs the commitment to it. If the operator ever stops working, users will be able to access their funds with the help of DAC.

7. The prover generates a ZK proof of the state transitions and sends it to the verifier. The DAC signatures are transfered to a smart contract.

8. The smart contracts verify the proof and the DAC signatures and approve the state transition. Verifying the signatures is critical as it ensures that users will be able to use the DAC to exit. The validium smart contract updates the stored state.

9. The funds of the users that have requested an exit are released from the validum smart contract.

Volition

To turn the tables on the fundamental scaling trade-off, StarkWare (the company behind StarkEx) has proposed a **hybrid solution** that would allow users to **choose between security and scalability**: volition. It would enable choosing between storing data on-chain and off-chain for every single transaction to benefit from both zkRollup and validium advantages.

A great use case for leveraging volitions could be cryptocurrency exchanges. Users actively trading on an exchange could store data off-chain (à la validium). On the other hand, when holding their funds, they could switch to the off-chain mode (à la zkRollup).

We haven't seen any working deployments of the volition design yet, so it's too early to judge it or dive deep.

What About Transaction Throughput?

Our report doesn't mention figures such as transaction throughputs or speed limits on purpose. We believe that **it's too early** to talk about them as the described technologies are still under development, and the final numbers might be far from initial theoretical estimations.

04

Technologies

To demonstrate how the approaches described earlier are implemented in practice, we're going to present an overview of several prominent ZK-based projects.

We've chosen **zkSync, StarkEx, and Loopring** to describe in detail.

zkSync

STARKWARE

LOOPRING

zkSync

📢 General Information

zkSync is an open-source scalability solution falling into the category of zkRollups that uses zero-knowledge SNARK proofs. Matter Labs first introduced the vision behind it in December 2019, and its first working version was launched on the mainnet in June this year. Right now, it can be used for ETH and ERC20 transfers, but the company's also planning to introduce an exchange functionality.

In addition to zkSync, Matter Labs is developing **Zinc:** a generalized programming language that'll allow zkSync users to build smart contracts using zero-knowledge validity proofs.

💻 Usability

We've tested zkSync's usability based on its **beta application** that requires connecting an Ethereum wallet (currently MetaMask or any WalletConnect-compatible one). Once it's done, the user can transfer their funds (ETH or ERC20) to zkSync. Then, these funds can be sent to any Ethereum wallet as the receiver doesn't have to use zkSync (it's an important feature as it enables migrating any payment system from Ethereum to zkSync).

The application is **user-friendly**: easy to navigate and fairly intuitive. The interface looks nice and facilitates using the product. Both entering funds and making transfers to another user are simple. The transaction finality time is approx. 10 minutes, which is the time necessary to generate a SNARK zero-knowledge proof, but the system already provides instant transaction confirmations displayed in the recipient's UI and API (in the future, zkSync team wants to make generating proofs faster). Based on how it looks right now, we expect the future product's usability to be quite high.

zkSync

💰 Business Model

zkSync is an open-source protocol, so everyone can build on top of it. Matter Labs monetizes the product by introducing transfer fees that origin in two sources:

1. off-chain part: the cost of storing the state on the blockchain and generating a zero-knowledge proof,
2. on-chain part: the cost of verifying the zero-knowledge proof and publishing the transaction data.

The fees are charged from the ether and tokens transferred by the user.

🔒 Security

zkSync inherits all the **security advantages of zkRollups** described in the Architectures section: thanks to the on-chain data storage, it doesn't cope with the data availability problem. However, the **trusted setup requirement** of the SNARK technology is its weak link. If all the parties participating in the trusted setup ceremony collude, it's possible for them to take over users' funds. Matter Labs mitigates this risk by leveraging the universal trusted setup described earlier. In the future, the company plans to switch to the STARK technology.

zkSync

🦉 Support

Currently, the product can only be used for **ETH and ERC20 transfers**, and new tokens need to be manually whitelisted by the zkSync operator. However, the company's going to make adding new tokens permissionless soon.

zkSync product roadmap contains **smart contract** support. Users will soon be able to write them with the use of **Zinc**—a programming language that's currently being developed by Matter Labs.

Zinc

Writing code in **Zinc** doesn't require advanced knowledge of the underlying cryptography. Zinc is heavily **inspired by Rust** when it comes to syntax and a general feel. However, the language is **not Turing-complete**. With unbounded looping and recursion prohibited in Zinc, R1CS circuit construction is efficient, formal verification is easy, and gas computation issues are largely eliminated.

Currently, Zinc compiles to an intermediate representation (R1CS circuits), with compilation to EVM on the way.

The following page presents a simple example of a function written in Zinc. It takes a pair of hashes as input and calculates a hash of their concatenation. This operation is used heavily in the context of blockchain for basic operations on Merkle Patricia tries.

zkSync

```
fn merkle_node_hash(left: Sha256Digest, right: Sha256Digest) -> Sha256Digest {
    let mut data = [false; 496];
    let left_bits = std::convert::to_bits(left);
    let right_bits = std::convert::to_bits(right);
    for i in (0 as u16)..(248 as u16) {
        data[i] = left_bits[i];
        data[(248 as u16) + i] = right_bits[i];
    }
    let digest_bits = truncate(sha256(data), 248);
    std::convert::from_bits_unsigned(digest_bits)
}
```

Summary

Company: Matter Labs

Stage: mainnet (beta application)

Technology: zkRollup

WEBSITE

BETA

BLOCK EXPLORER

GITHUB

STARKWARE

📢 General Information

StarkEx (previously StarkDEX) is a scalability engine that can operate as either zkRollup or validium designed to boost the throughputs of non-custodial exchanges. As opposed to zkSync, it uses STARKs. It has been developed by StarkWare since 2018, and the first platform to fully integrate it was DeversiFi (a decentralized exchange) that released its version 2.0 using StarkEx in the validium mode in June this year.

Apart from StarkEx, StarkWare has also developed a complementary subset for making transfers: **StarkPay.** Built upon the same technology, it allows sending ETH and ERC20 tokens to other users.

💻 Usability

For the sake of testing StarkEx usability, we've played around with the features of DeversiFi: a non-custodial exchange that runs on the engine. After establishing the connection with the wallet (currently, it's integrated with WalletConnect and Ledger), the user can enter their funds (ETH or ERC20).

The platform is **easy to use** and fairly intuitive. The entering and exiting procedures are simple just like exchanging one's tokens. The exchange operations themself are fast, but withdrawals don't happen immediately (however, it doesn't result from StarEx design but rather from DeversiFi's implementation). Nonetheless, it's still significantly faster than with optimistic rollups or state channels, and it's actually a similar time that is needed to withdraw funds from a custodial exchange. Due to that, we find DeversiFi usability high.

STARKWARE

💰 Business Model

StarkWare builds custom solutions for their partners. As StarkEx is propietary software, it’s necessary to establish such a partnership to use it to scale one’s product.

🔒 Security

As stated above, StarkEx can function as both validium and zkRollup. DeversiFi has chosen the first architecture for the sake of its users’ **transaction data privacy**. In result, the network’s latency is higher than in the case of opting for the zkRollup mode, but this design provokes the data availability problem.

StarkWare has introduced several features to alleviate this concern, the most important one being the **Data Availability Committee (DAC)** figure. It consists of 5 reputable organizations: Infura, ConsenSys, Nethermind, iqlusion, and Cephalopod. As they keep the copies of the data, users’ are able to access it in case of StarkEx operators refusing to process exit requests. According to StarkWare, this emergency scenario together with other security features eliminates the need to trust the operators completely.

Ultimately, StarkWare wants to mitigate the data availability problem by allowing users to pick between on-chain and off-chain storage on the transaction level, making its product a volition.

STARKWARE

🦉 Support

StarkWare products support **ETH and ERC20 token** transfers and exchanges. Similarly to zkSync, the company plans introducing general computation in the future.

Summary

Company: StarkWare
Customers: DeversiFi, Gods Unchained
Stage: mainnet
Technology: validium, zkRollup, volition (soon)

WEBSITE

DOCUMENTATION

DEVERSIFI

LOOPRING

📢 General Information

Loopring is a zkRollup scalability protocol developed since 2017. Just like zkSync, it stores users' data on-chain and leverages the SNARK technology. Initially, it was designed to address the needs of decentralized exchanges, but Loopring Project launched a new application meant for transfers (Loopring Pay) in June this year.

The latest version of the protocol, Loopring 3.1, powers **Loopring Exchange:** the company's own DEX available in a beta version on the mainnet. Apart from trading, it allows making ETH and ERC20 token transfers. Although the protocol is open-sourced and Loopring Project is open for partnerships with those who want to build on top of it, the company focuses on developing its own products available at www.loopring.io. They're also working on a mobile version of their application.

In Loopring, data is stored on-chain but the company claims that it enables its partners to **switch off the on-chain data availability.** After doing so, data is stored off-chain, which makes the product function as a validium. Nonetheless, the default settings have the data availability option on and this is how Loopring Exchange operates right now.

💻 Usability

We've assessed the usability of Loopring based on the Loopring Exchange transfers feature. To send their funds to another person, the sender needs to connect their Ethereum wallet (currently MetaMask or any WalletConnect-compatible) and create a Loopring Exchange account. As the registration is on-chain, the time needed to do that depends on gas price and the speed of confirming Ethereum blocks.

Entering funds and transferring them is **easy and intuitive.** The interface looks nice and every option is easily reachable. Depositing is quick and although it takes more time to withdraw funds, the usability of the product is quite high.

💰 Business Model

Loopring Project is open for collaborating with companies that want to build on top of its open-source protocol. In such cases, the customised solution comes with **transaction fees** established on the individual customer level.

However, Loopring Project focuses on **developing its own products** (currently, a DEX featuring a payment application) and making profit from their transaction fees. Trading at Loopring Exchange is free for the maker but the taker is charged between 0.06% and 0.1% of the value of the order. Transfers, by turn, are totally free at the moment.

🦉 Support

Loopring products support **ETH and ERC20 tokens transfers and trading,** but new tokens need to be registered within the system first. Loopring Project doesn't plan to introduce smart contract support to its products or other form of general computation support.

Summary

Company: Loopring Project
Products: loopring.io
Stage: mainnet (v 3.1)
Technology: zkRollup

WEBSITE

DOCUMENTATION

LOOPRING EXCHANGE

05

Summary

The described layer 2 solutions constitute a promise for the Ethereum community. A promise of secure scaling that will help Ethereum become a true alternative to the traditional financial system.

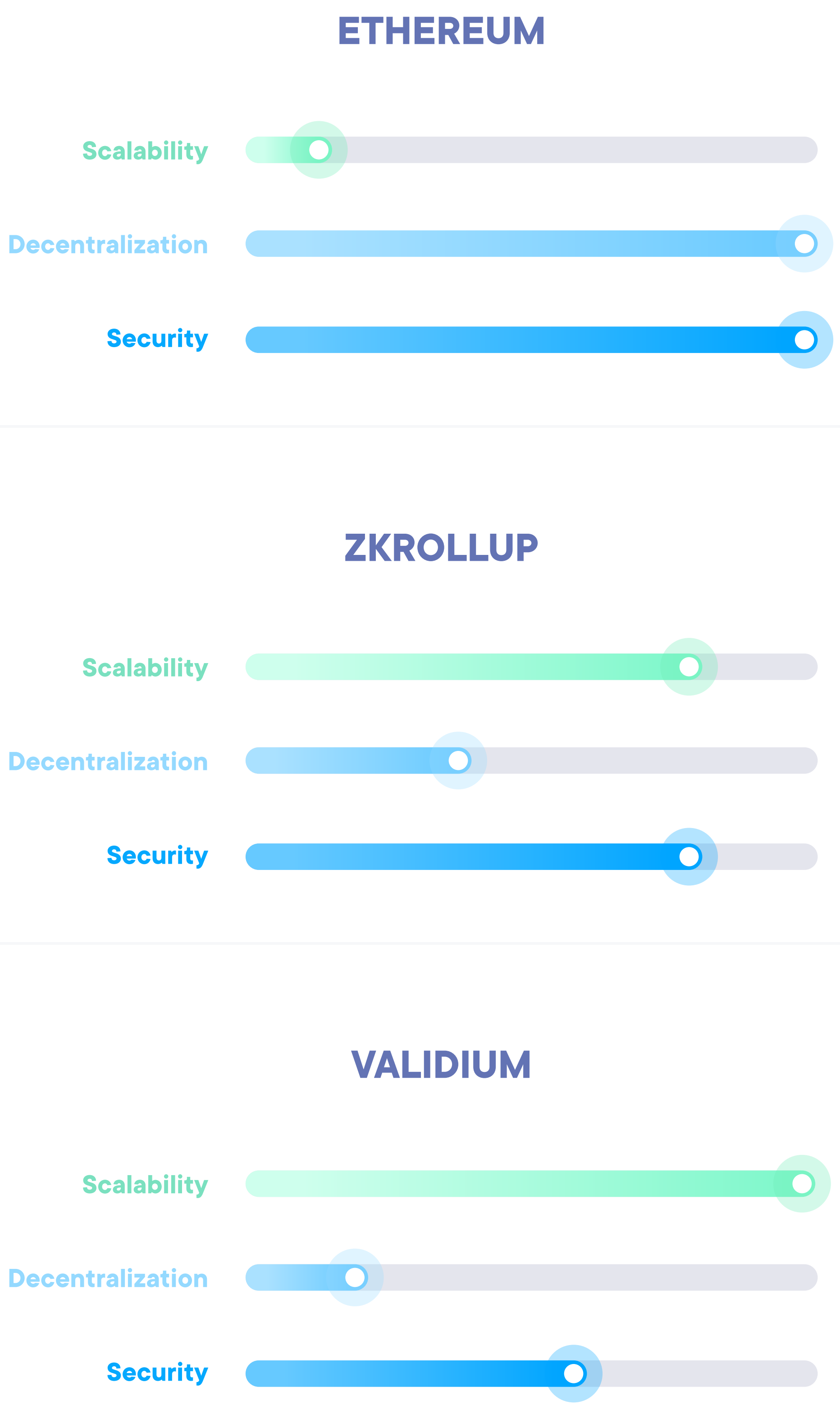
We can see the potential of Ethereum's role shifting from a simple distributed ledger that records transactions to a decentralized central bank. The new role of the blockchain would be to do settlement for L2 technologies and secure their users' business.

All the three described solutions push the limit of Ethereum scalability forward but can't fully escape the blockchain trilemma which represents the **fundamental trade-off between scalability, security, and decentralization**. It's impossible to provide all of them to the maximum extent, and one needs to make sacrifices to accommodate for this.

Blockchains like Bitcoin or Ethereum sacrifice scalability to achieve radically high security and decentralization. L2 solutions improve on scalability, but have to compromise decentralization without compromising security significantly.

While zkRollups provide higher security, it is the validium approach that scales the blockchain more efficiently. The future volition design might push the boundaries even further.

The Blockchain Trilemma



However, keep in mind that all of them are still in what we could call an experimental phase. None of them is a real game-changer yet, but we believe **the revolution might be just around the corner**.

As much as we would love to see rock-solid and battle-tested L2 scaling solutions, nothing like this exists on the market yet. We're very excited to observe the development of the solutions based on zero-knowledge cryptography with huge potential to move things forward, and we have high hopes for their future.

The window for building prototypes has just opened.
The time to gain the first-mover advantage is now.

Do you agree with our **opinions**?
Or do you find them **controversial**?
Did we miss anything **important**?

Let us know on Twitter:

@ethworks

Authors

We are Ethworks.

A truly remarkable team for your blockchain project.
To always stay up to date with what's going on
in the blockchain, follow us on Twitter.

Idea and supervision:
Marek Kirejczyk

Principal technical input:
Piotr Szlachciak

Additional technical input:
Krzysztof Jelski
Dmytro Maretskyi

Editor:
Arleta Więch

Design:
Joanna Charczuk

Additional design input:
Natalia Kirejczyk

Acknowledgements

We'd like to especially thank the **Golem team**.
The initial research we conducted for them inspired us to create this report.

We'd also like to thank **Alex Gluchowski** from zkSync,
Brecht Devos and **Matthew Finestone** from Loopring,
and **Tom Brand**, **Eli Ben-Sasson** and **Uri Kolodny** from StarkWare
for providing their in-depth technical input and feedback.

ETHWORKS

@ethworks

ethworks.io