Zokrates - Scalable Privacy Preserving Off-Chain Computation

Tejas Crs2222 , Pushpendra Crs2209

December 22, 2023

Blockchain Overview

Blockchain Overview

Blockchain is a decentralized and distributed digital ledger technology that securely records and verifies transactions across a network of computers. It consists of a chain of blocks, each containing a list of transactions.

Some key properties of blockchain include:

1: Decentralization

2: Transparency

3: Immutability

4: Security

5: Consensus

6: Smart Contracts

Blockchain Overview

Let us focus on the main point of this discussion:

So in blockchain, every node needs to validate and process the transaction, In addition, all data should be present for validation and processing at each node, which leads to the problem of scalability

Not only scalability the problem of privacy also arises for public blockchains like Bitcoin, Ethereum, etc

Let's formally define the challenge of **scalability** and **privacy** in the world of Blockchain

Scalability Challenge:

Scalability is the ability of a blockchain to efficiently handle a growing number of transactions and users.
The current practice of processing transactions at every node in the blockchain network can limit the system's ability to scale.

Privacy Challenge

Transactions and data are visible to all participants on the network, essentially making information public.

> **Scalability** and **Privacy** are the main concerns in the era of blockchains.
> To address the above issues, the idea of **off-chain computation** comes into play.

Off-Chain Computation

Off-Chain Computation

Instead of processing everything on the main blockchain, we suggest moving certain computations off-chain. This off-chain processing can significantly reduce the load on the blockchain network.

Taking computation and data off the blockchain can have significant benefits for performance, cost, and privacy, Moving computations from the blockchain to the external nodes without compromising its properties is a more challenging problem.

Off-chain Computation

Delegating computation to arbitrary nodes that publish the results of the blockchain can have two main benefits compared to on-chain execution:

1: A delegate node can use private information during the execution of a computation without publicly revealing it. This is not possible for on-chain computations.

2: Ideally, a delegate would only write the **result of a computation** to the blockchain, which would highly increase efficiency as the only operation performed redundantly, each node would be storing the result. This would **largely increase** the transaction throughput.

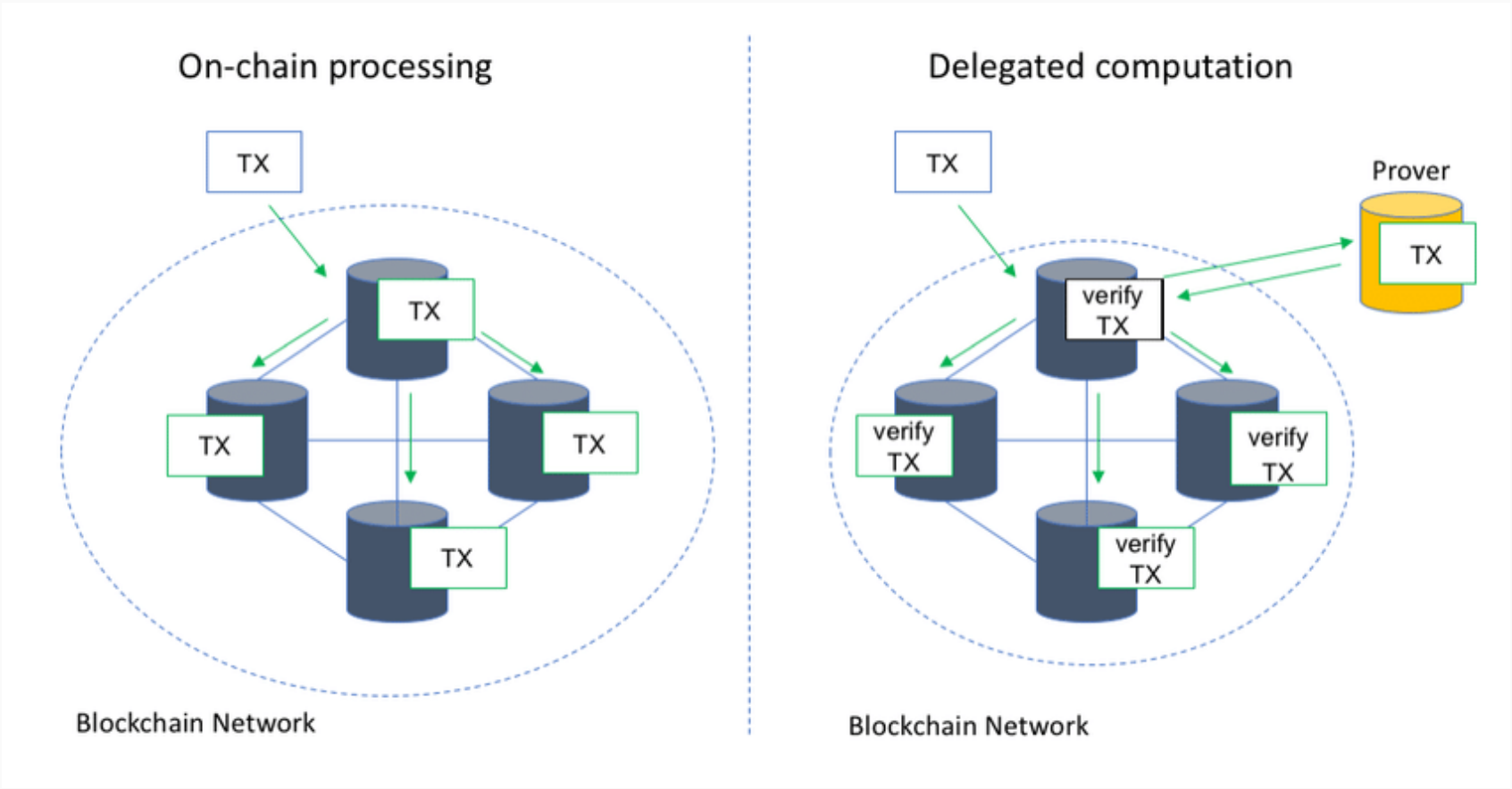

Figure 1: On-chain vs delegated computation

Scalability achieved

So from above, it is clear that **scalability** is achieved. **What about privacy**?

Privacy achieved

Privacy is also achieved.

In traditional on-chain execution, computations are redundantly performed on all nodes in the network. This redundancy requires that **all nodes possess all the data involved in the computation**, which may include private or sensitive information. This lack of privacy is a challenge for applications that involve confidential data.

Privacy cont:

By delegating the computation to specific nodes off-chain, these delegate nodes can use **private information** during the execution **without publicly revealing** it to the entire network. This **selective sharing** of data allows for the execution of computations involving sensitive information without **compromising privacy**.

Crucial Fact

In this way, we resolve the scalability and privacy challenges of the blockchain.

As we are delegating the computation the fundamental thing we are assuming till now is that the **delegated node is the trusted node**, as it is using private data and providing the results of the computations.

which violates the blockchain's key property of trustlessness

Verifiable Computation

To deal with this situation,

we propose to employ a **verifiable computation scheme**, so that the delegate node acts as a prover, providing both the **cryptographic proof** and the **computation result**.

The proof is a cryptographic guarantee that the computation was carried out **correctly**, and the result is the **actual output** of the computation.

Together, they enable network participants to **verify the correctness of the off-chain computation** without having to **trust** the delegate node, aligning with the trustless nature of blockchain technology.

Key properties of VC

As we are employing a verifiable computation scheme, that scheme needs to satisfy the following key properties:

1) **Short proofs and cheap Verification**
2) **Zero Knowledge**

Note: The private data handled by the delegated node is **encrypted**.

Verfiable Computation scheme: ZkSNARK

Verifiable Computation Scheme (ZkSNARK)

Verifiable computation schemes enable **computationally weak verifiers** to outsource computations to **untrusted provers**, who then perform the computation and return the result, including proof that the result is correct.

ZkSNARKs (Zero-Knowledge Succinct Non-interactive Arguments of Knowledge) are a set of verifiable computation schemes with the following key properties:

Properties

1: **Proofs are short and non-interactive**, i.e., a prover can convince a verifier with one message.

2: A prover can use private information during proof generation, and the verifier learns nothing about that information.(**Zero Knowledge**)

3: The verification cost is independent of the computational complexity of the input.

ZkSNARK

For a result to be **accepted in a blockchain network**:

1: The accompanying proof has to be verified first.

2: The cost of this verification replaces the cost of on-chain execution of the computation in the first place.

3: On-chain verification needs to be cheaper than the on-chain execution of the computation in the first place (**to realize scalability**)

Zokrates

Implementation

ZoKrates is designed to support the whole process from specifying a provable program to generating proof and verifying its correctness on a blockchain.

Process Steps:

Specification of Computation: The program is compiled into flattened code, an abstraction easily convertible into R1CS (Rank-1 Constraint System) compatible with zkSNARK proof systems.

Setup Phase: A setup phase is performed, resulting in two public keys: a long proving key and a short verification key.

Implementation Process

Smart Contract Generation: Based on the verification key, a verification smart contract is generated. This contract is capable of verifying proofs on the blockchain.

Proof Generation: A prover executes the flattened program, finding a solution to the program via the "compute-witness" step. Using the proving key and the solution, the prover creates a proof.

Verification on Blockchain: The prover sends the proof to the verification contract on the blockchain to verify its correctness.

Zokrates Process

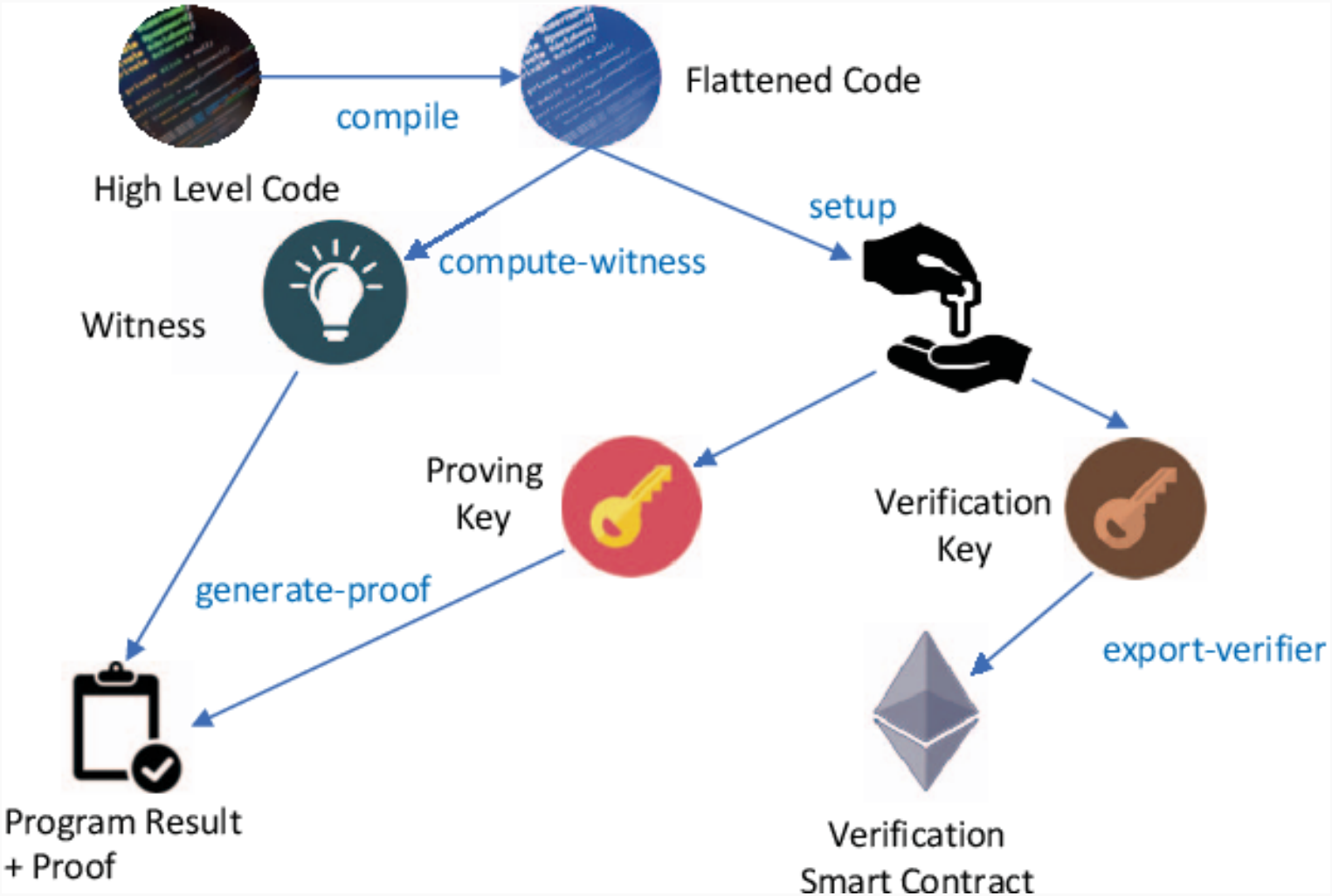

Figure 2: Full Zokrates Process

Zokrate Toolbox

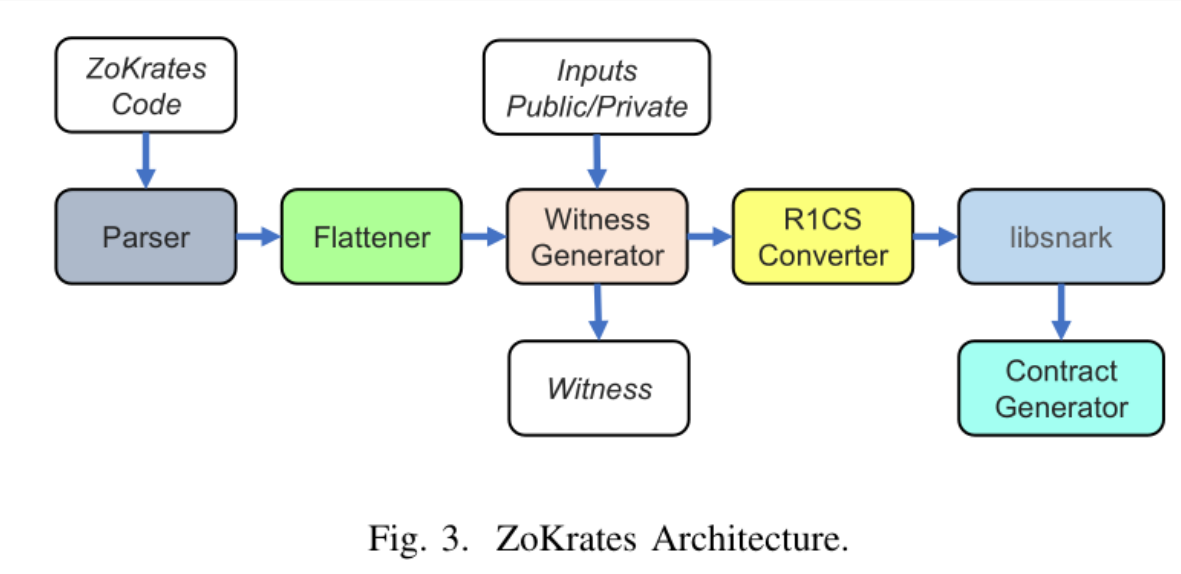

Fig. 3. ZoKrates Architecture.

Figure 3: Zokrates architecture

Zokrate Toolbox - Compiler

The compiler, consisting of the parser and the flattener in Fig. 3, translates DSL code into so-called flattened code.

Zokrate Toolbox - Witness Generator

Before a proof attesting correct execution can be generated, th**e program first has to be executed**. Means, finding a satisfying variable assignment for a flattened program. A witness for a given program can be found using the witness generator which acts as an interpreter for flattened code. It **takes a program's private inputs as parameters, executes the flattened code and stores all variable values**, thus finding a witness in the process.

Zokrate Toolbox - Setup Proof Generation

zkSNARK verifiable computation scheme **re- quires a one-time trusted setup to be performed**. It is trusted in the sense that who executes it needs to forget private information used in the process. If that information is not forgotten, the owner could potentially create fake proofs. However, the zero-knowledge property of other proofs would not be impaired. To address this issue, **we can use MPC with dishonest majority.** which distributes the setup phase over n nodes and is secure as long as there is at least one honest participant.

Implementation

```
# compile
zokrates compile -i root.zok
# perform the setup phase
zokrates setup
# execute the program
zokrates compute-witness -a 337 113569
# generate a proof of computation
zokrates generate-proof
# export a solidity verifier
zokrates export-verifier
# or verify natively
zokrates verify
```

Figure 4: Commands

Conclusion

In conclusion, Zokrates emerges as a pivotal solution in addressing the scalability challenges of blockchains through the innovative integration of off-chain computation and verifiable computation. By offloading certain computations away from the main blockchain and leveraging zero-knowledge proofs for verification, Zokrates not only enhances efficiency but also ensures the privacy and integrity of off-chain processes.

References

(1) ZoKrates. (n.d.). Getting Started. Retrieved December 19, 2023.
https://zokrates.github.io/gettingstarted.html

(2) Eberhardt, J., & Tai, S. (n.d.). ZoKrates - Scalable Privacy-Preserving Off-Chain Computations. Information Systems Engineering (ISE), TU Berlin, Berlin, Germany.
{je,st}@ise.tu-berlin.de

THANK YOU.