INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2022, VOL. 71, NO. 2, PP. 463-475
Manuscript received February 2,2025; revised May, 2025. doi: 10.24425/ijet.2025.153593

Anonymization in web auctions using Zero-Knowledge Proof in Blockchain technology

Marlena Broniszewska, Wiktor B. Daszczuk, and Denny B. Czejdo

***Abstract*—Global digitalization has accelerated, leading to continuous online shopping and services growth. However, the centralized nature of traditional e-commerce platforms raises concerns about data control, privacy, and potential single points of failure. Blockchain technology offers a decentralized alternative that addresses these issues, improving online transactions with enhanced privacy and anonymity for users. This article explores the problem of anonymization in web services by developing an anonymous online auction system using blockchain and zero-knowledge proof (ZKP). We propose a solution that employs ZKP in two stages: first, to verify that a user is eligible to participate in the auction, thereby creating a list of potential bidders; second, to prove that a bid is placed by a user from this list without disclosing their identity. This approach allows participants to engage in auctions anonymously, protecting their interests from competitors and sellers. The system eliminates the need for a trusted third party by leveraging the blockchain immutable ledger for transparency, giving users full control over their data and participation. We present the architecture and implementation details of the system, demonstrating its feasibility and potential to enhance privacy and security in online auctions.**

***Keywords*—blockchain; zero knowledge proof; auction; anonymity; zk-SNARK**

I. Introduction

EVOLUTION of technology [1] and the recent COVID-19 pandemic accelerated world digitalization. We are able to process almost any matter over the Internet. People can purchase or sell items, they can work remotely, and even handle official matters using a computer. Especially, online shopping has gained popularity. Many stores have relocated to the "e-world". Most online shopping platforms are centralized, and run by individual companies like Amazon or eBay. These platforms manage transactions and store all user data. This centralization imposes many restrictions, e.g., single points of control can lead to potential failure. The alternative approach is based on the distribution of data and processes across multiple nodes.

Introduced in 2008, the Bitcoin Whitepaper [2] triggered the fast development of blockchain technologies based on distributed processing. Such technology gives users a tool for storing data without any centralized administrator. The decentralization led to many improvements, including privacy. In e-shopping, privacy is handled through traditional means, such as secure payment gateways and data encryption. The customer identities, purchase histories, and payment information are typically shared with both the seller and the payment processor, making anonymity challenging. In addition, user behavior and preferences can be tracked for personalized marketing. Anonymity is a core feature of blockchain technologies, allowing users to participate in buying/selling transactions, typically through bidding, without revealing their identities or their bidding behavior to other participants or the seller. The use of private and public keys ensures that only authorized users can place bids without exposing sensitive details.

Blockchain technology provides transparency in the transaction process, as anyone can verify transactions without knowing sensitive user information, with a smart contract on a blockchain managing the bids and verification of user participation. This eliminates the need for a trusted third party, as the blockchain serves as an immutable and transparent ledger. This was a breakthrough in creating decentralized, transparent systems where trust is not required between parties. Blockchain-based systems enable users to participate in auctions without worrying their bids were anywise rigged or manipulated by malicious users.

Also, the decentralized nature of the system means that once a smart contract is deployed, no single entity can alter its rules, making it autonomous and resistant to censorship or manipulation. Users have full control over their participation and data in blockchain auctions. Let us contrast it with e-shopping, where users are subject to the policies and terms of the shopping platform. Account suspensions, order cancellations, and policy changes are under the platform control. Users have limited ability to verify how their data is stored and used.

In blockchain applications, the focus is on auctions, where multiple users can place bids on items, and the process is time-constrained. The bidding mechanism adds competitive pricing and can help in flexible variability in the final price paid for a commodity. Most online shopping platforms offer fixed-price sales, although some include auction features such as eBay. However, typically, in online shopping, prices are predetermined by sellers, and users purchase directly without competition from other buyers.

Despite the fact that in online auction buyers cannot examine an item beforehand, e-auctions become a more popular option. They are preferred by many buyers from all around the world because they assure privacy, decentralization, transparency, and security, contrasting significantly with traditional online shopping, which is more centralized, fixed in

This work was not supported by any organization.

First and Second Authors are with Warsaw University of Technology, Department of Electronics and Information Technology, Warsaw, Poland (e-mail: wiktor.daszczuk@pw.edu.pl; m.broniszewska@gmail.com).

Third Author is with Fayetteville State University, Department of Mathematics and Computer Science, Fayetteville, NC 28301, USA (email: bczejdo@uncfsu.edu).

pricing, and relies on standard security practices that do not offer the same level of anonymity or transparency.

A significant improvement in online privacy was the concept of zero-knowledge proof introduced in (ZKP [3]). This cryptographic method for creating digital proofs of knowledge is a method for one party to prove the other party the possession of specific information. This protocol already has a few different implementations and has become widely used to provide individuals with privacy and information security. Using zero-knowledge proof in blockchain-based systems can help to provide anonymity to users. Taking into consideration the above, the problem of anonymization in web services by the example of anonymous online auctions will be explored by the authors. The authors will try to explore whether it is possible to create a system that allows a party to attend an auction anonymously for other competitors and sellers using the latest technologies, such as zero-knowledge proof and blockchain technology. As a result of such research, a system for anonymous auctions was developed and will be assessed in this article.

The essence of our solution is that ZKP is used twice:

- first, proves that a user is eligible to participate in the auction (and maybe they will); it is used to prepare a list of potential bidders,
- second, proves that a bid is placed by a user from the list.

The literature overview, which covers the present situation and prospects for auction anonymization, is addressed in Section II. A short description of blockchain technology is presented in Section III. Section IV shows the Zero-Knowledge proof principle. A variety of blockchain auctions is shown in Section V. The architecture of the proposed solution is covered in Section VI. Section VII presents the implementation details. The article is concluded in Section VIII.

II. Related Work

Blockchain technology is a decentralized, transparent and secure distributed network of nodes that stores transaction data. Anyone can read from a blockchain, since the network is fully transparent. Any user of blockchain can access the entire transactions history, including all the data stored there. This attribute of blockchain technology makes the network secure since anyone can verify the correctness of data stored in the network. Blockchain technology can give users an elusive sense of privacy and anonymity, since users do not have to share personal information to use the network. Blockchain operations are transparent: if someone knows another user account address, they can overlook the transaction history of such an account. They can see the balance of the account, every funds transfer, and plenty of other information. This makes users of a blockchain network pseudo-anonymous.

Zero-knowledge proof can provide such privacy for blockchains. A good example of the usage of the zero-knowledge protocol in blockchain applications is Zcash [4]. Zcash is a blockchain with ZEC cryptocurrency. Zcash uses the zk-SNARK protocol to ensure the confidentiality of transactions. They introduced two types of addresses in their blockchain:

- **Transparent address** (public address) – regular blockchain EOA address, allows a user to send only public transactions.
- **Shielded address** (private address) – address that allows to hide funds that are being received or sent; it allows users to send public and private transactions.

Each transaction is sent from one Zcash address to another Zcash address. By combining different types of Zcash addresses, four transaction types in Zcash blockchain were introduced [4]:

- **Public** – transaction sent from a public address to a public address. Such a transaction is similar to an Ethereum transaction.
- **Shielding** – transaction sent from a public address to a private address. The address of the sender is public, but the address of the receiver remains private.
- **Deshielding** – transaction sent from a private address to a public address. The address of the sender remains shielded, and the address of the receiver is publicly available.
- **Private** – transaction sent from a private address to a private address. Both addresses remain shielded.

This approach gives users more privacy and flexibility, since they can choose whether transactions they send should stay public or become confidential.

Another instance of using zero-knowledge protocol in the blockchain is Tornado Cash [5]. This is an example of when a blockchain application was not used in a legitimate way. Tornado Cash no longer operates, since it became a money laundering platform and was banned by the US government. Tornado Cash was an application that could run on EVM (Ethereum Virtual Machine) compatible blockchains. Tornado Cash was an open-source, decentralized cryptocurrency fumbler, sometimes called by the community a mixer. The main use case of this platform was to ensure assets confidentiality. Users could deposit different amounts of cryptocurrency and then withdraw it using different Ethereum address. A user who wanted to withdraw their funds, had to provide cryptographic proof of a previous deposit. This application used a zero-knowledge proof (precisely zk-SNARK) to ensure that the sender and recipient of the funds could not be linked.

Zero-knowledge protocol is increasingly used in blockchain implementation itself. An instance of such implementation is the Layer 2 Ethereum scaling solution - Polygon Hermez [6]. This solution uses zk-rollups (zero-knowledge rollups) and is an off-chain solution. Zk-rollups execute transactions off-chain (outside of a blockchain) to reduce the amount of data that must be stored on a blockchain. Then, transactions are rolled up into batches. Next, these transaction chunks are recurrently committed to an on-chain contract (contract deployed to Ethereum). To post data to the Ethereum smart contract, the zk-rollups node must calculate the cryptographic proof of the validity of transactions included in the batch (the proof that transactions are not fraudulent). The proof is verified by another Ethereum smart contract (the verifier). This proof is sufficient to include zk-rollups batch of transaction on the base chain (in this case, Ethereum blockchain). To summarize, the zk-rollups solution takes multiple transactions from the base blockchain, combines them into a single transaction, and then

posts a validity proof to an Ethereum smart contract. The proof is verified by the verifier smart contract to ensure transactions legitimacy.

Applying zero-knowledge protocol to blockchain-based solutions can provide more privacy for users. More and more blockchain projects are starting to utilize ZKP in their solutions. As described in this section, examples are only a patch of blockchain projects using zero-knowledge protocol in their projects [6] [7] [8].

The popularity of selling, buying and bidding over the Internet has escalated over the past years. Authors of [9] show that electronic auctions are one of the most effective forms of e-commerce. They recite three factors of the rapid development of such a market. The first one, which is the most important, states that online auctions consume less effort for the buyers and also the sellers to execute a trade.

However, online auctions also come with security issues compared to traditional auctions. Paper [10] inspects various security and privacy matters in online auctions, such as personal data disclosure or auction fraud. In centralized auction systems, bidders and sellers must trust third-party intermediaries in many matters. The most critical factors are data transparency, integrity, and traceability.

Cryptography is a widely used approach for addressing the problem of the central auctioneer. In [11], the authors proposed a system replacing one central auctioneer with several. In this solution, a security risk exists if auctioneers ally together. The authors of [12] use discreet cryptographic proofs to create a fair e-auction with minimum trust required. However, their solution relies on time source for synchronization, which is a potential security risk as well.

Several researches are focusing on the privacy of online auctions using blockchain technology. In [13], the authors propose a blockchain-based system for transparent auctions. This approach eliminates any third party and ensures process legitimacy. For that matter, the authors of [14] use the Ethereum blockchain as a replacement for any third-party auctioneer. They present a security model using existing cryptographic primitives such as designated verifier ring signatures to preserve the confidentiality and anonymity of the bidders. In [15], we can read that another advantage of using blockchain as third-party eliminator is a significant reduction of transaction costs. The authors of [16] designed a system for online auctions on blockchain. Their solution requires a set of trusted authorities. As in [11], potential security risks can be observed if authorities collude altogether. Resembling security issues can be found in [17]. The authors proposed a blockchain-based system for e-auctions using smart contracts and third-party operators. The proposed protocol does not provide a way for a single user to inspect the correctness of auction execution. Another solution for blockchain-based auctions is in [18]. The authors proposed a protocol for anonymous auction based on Consortium blockchain time-release encryption. To authorize legitimate bidders, they embraced a strict digital certificate-based identity mechanism. The solution still relies on third-party auctioneers.

To offer a fully anonymous system, the authors of [19] proposed a protocol for anonymous payments using blockchain technology and zero-knowledge proof encryption. Their system allows them to anonymously exchange payments without any third-parties. To prevent the ownership of attributes disclosure and identity unlinkability, the authors of [20] proposed a system based on blockchain smart contracts and a zero-knowledge proof algorithm. They implemented a system that allows selectively revealing ownership of attributes to service providers to protect personal privacy.

In paper [21] writers propose blockchain-based smart contract as replacement for the central auctioneer. To support the public verifiability and to avoid the secret communication channel, the authors use anonymous veto networks and zero-knowledge proof. They guarantee a privacy of the bid value, unless a bidder exposes their bids voluntarily. However, it is still possible to discover whether someone placed a bid.

Another blockchain-based auction was proposed by the authors of [22]. They use the Ethereum blockchain and zero-knowledge proof for the secrecy and validity of the transaction. Nevertheless, their solution still depends on third-party auctioneers. The authors of [23] proposed an e-auction based on the Ethereum blockchain, Pedersen commitment algorithm, and zero-knowledge proof encryption. However, the main attribute of their system is to hide the bid price. As in [21], other participants can discover whether someone placed a bid. Another solution for blockchain-based online auctions is proposed in [24]. The authors introduced a blockchain-based sealed-bid auction scheme with time-lock encryption. The main point is to ensure that the bidder data will be available only after a specific time elapses. They utilize a time-released blockchain to provide a mechanism for punishing dishonest users and purge the demand for smart contract validation by any third party.

Authors of [25] presented a traceable data exchange using zero-knowledge and non-fungible tokens. According to the authors, their schema ensures data privacy and traceability while still providing fairness during data exchanges. The schema is able to track and record on a blockchain all data transformations, provides zero-knowledge proofs of transformations and data correctness, and warrants data privacy in public storage platforms.

Authors of [26] elaborate blockchain-based mixers that use zero-knowledge proof technology. Such mixers should guarantee privacy and anonymity for depositors and withdrawers. They propose five heuristics that can increase the probability of linking the depositor and the correct withdrawer by 50%. The authors also identify more than 100 attacks abusing zero-knowledge proof-based mixers as the initial and destination funds source.

Online privacy and anonymity give a user control over personal data and identity. A growing number of projects in the blockchain industry include zero-knowledge proof in their systems. An example of such an approach is [4]. They use zero-knowledge proof to ensure blockchain transactions confidentiality. Another example can be an already nonexistent mixer [5]. They used zero-knowledge proof to hide any funds origin. There is also a Polygon Hermez [8] solution that uses zero-knowledge protocol to reduce the amount of data stored on a blockchain.

III. BLOCKCHAIN

Digital Ledger Technology has been gaining popularity since 2008, when an anonymous developer under the pseudonym Satoshi Nakamoto published a whitepaper

demonstrating a model for the Bitcoin blockchain [2]. Blockchain technology is a peer-to-peer decentralized network that is shared around nodes. Blockchain is a digital ledger that stores data. Each node stores a copy of such data. Blockchain is a form of a digital database that collects information in blocks. It differs from a typical database since it does not store data in tables but in chunks (blocks). It is nearly impossible to change, modify or remove data once stored in a blockchain network. This indicates that data stored in such a network is permanently logged. Blockchain technology has three major key features that must be satisfied:

- immutability,
- decentralization,
- consensus.

Immutability - as mentioned before, the transaction, once logged, cannot be changed, deleted, or manipulated. Suppose someone wants to revert changes made by a mistaken transaction, another. In that case, a new transaction must be issued to revert the mistake, e.g., if Alice, by mistake, sends funds to Bob by issuing a transaction, to reclaim the funds, Bob must send the funds back to Alice by issuing another transaction.

Decentralization - there is no one centralized entity that manages the blockchain network. The blockchain network is shared among nodes that create such a network. Anyone can run their own node and actively participate in creating the blockchain network. This approach allows to reduce the need for trust among users.

Consensus - a set of rules must be followed when a node wants to add another block to the blockchain. This set of rules is called a consensus mechanism. The consensus mechanism helps to maintain the blockchain security and is used to validate the authenticity of transactions. Multiple consensus mechanisms are used in blockchains, such as Proof of Work (PoW) or Proof of Stake (PoS).

Numerous blockchain networks operate globally, including the original Bitcoin blockchain, the private IBM blockchain, and the well-established Ethereum blockchain. This article focuses on the Ethereum blockchain, so any subsequent mentions of a blockchain network will refer to Ethereum. Notably, Ethereum transitioned from PoW to PoS consensus mechanism in September 2022.

A. Ethereum Accounts

There are two types of accounts in the Ethereum blockchain:

- Externally Owned Account (EOA)
- Smart Contract Account (SC)

Both accounts are utilized in this article; hence both are closely described in this subsection.

An **Externally Owned Account** is an account that can hold, receive and send funds (e.g., Ether coin or other tokens) and is controlled by anyone with a public and private key pair associated with this account. Each EOA account has its own unique 42-character hexadecimal address, which is derived from the last 20 bytes of the public key [27]. This type of account can only store balance information and does not have other storage for any other data.

A **Smart Contract Account** (or just smart contract - SC) is another type of Ethereum blockchain account. SC does not have a public and private key pair, but it has its own unique 42-character hexadecimal address and balance. The main difference is that this type of account also has a piece of code. SC Account is a smart contract deployed to a network and controlled by its code. Besides balance information, the SC Account can store any other data since it has storage associated with it. A SC Account cannot perform any actions on its own. Account must first receive a transaction from any other EOA or SC Account. Such transactions can trigger code execution, which can perform various actions such as token transfer or other smart contract function call. A smart contract can call other smart contract function, but the call must be within a transaction initiated by EOA. To create such an account, a Smart Contract needs to be deployed to the Ethereum network.

B. Transactions

A transaction is one of the basic concepts in the blockchain network. It is a message containing data. The message must be cryptographically signed to ensure its validity. The Ethereum blockchain uses the ECDSA (Elliptic Curve Digital Signature Algorithm) algorithm to prove the ownership of a private key and to ensure the transaction is not fraudulent. Each transaction must be signed before sending, using the private key. This process creates a signature of the message. The created signature is then validated to ascertain its validity. A transaction is the only instrument to change the state of the blockchain network. Transactions have an upfront specified structure. Each transaction must contain, among others:

- Recipient - the recipient address,
- Value - the amount of Ether coin sent to the recipient,
- Data - data sent to a Smart Contract,
- ECDSA digital signature - signature created using a private key associated with the account that is sending a transaction.

If someone wants to initiate a transaction, they are required to cover the cost of the computational resources (gas) needed for a node to process it. On the Ethereum network, this is done using Ether, which is a cryptocurrency specifically used for transaction fees, such as deploying smart contracts. Different blockchains may have their own cryptocurrencies for this purpose. For example, on the Polygon blockchain, MATIC is the cryptocurrency used to pay for transaction fees.

There are two types of transactions:

- Smart contract creation (deployment) transaction,
- Message call transaction:
  - Ether transfer transaction from one account to another,
  - Smart Contract interaction transaction.

The Ethereum network is a public system, meaning anyone can both read from and write data to the blockchain. This makes all transactions on the Ethereum blockchain publicly accessible for anyone to view. To submit a transaction to the Ethereum blockchain, a fee must be paid. On this network, the payment is made using the cryptocurrency Ether (ETH).

C. Wallet

A Wallet is software or hardware that manages public and private keys for EOA. The wallet can be compared to a banking application - it allows users to send and receive cryptocurrencies, send other transactions and check account balance. Wallets allow users to interact with their EOAs. A software wallet is an application, often called by the community a hot wallet. Hot wallet has an Internet connection, and there are three types of hot wallets: desktop, mobile, and online. Hardware wallets are usually physical devices such as USB sticks. Such wallets are frequently called cold wallets. Cold wallets do not have an Internet connection and are considered safer and more secure than hot wallets.

D. Blocks

As mentioned before, a blockchain network stores data in blocks. Each block is built in a certain way: it contains a header and a set of transactions. Blockchain network can store various types of data, but the most common type is transaction. To put is simply, each block is built of a block header and a batch of transactions. Transactions are grouped in chunks to sustain a synced state of the network and maintain an agreed transaction history in each node. This approach enables the network to commit up to hundreds of transactions in each block. To ensure persistent order of blocks, each block contains a reference to a previous block in the form of a cryptographic hash of the previous block.

Each node in the network has its own copy of the blockchain. Nodes get updated every time new blocks are added to the chain. Blocks are created and added to the chain by miners in the mining process. Miners validate and add transactions to the block and solve hard puzzle to add a block to the chain and receive reward and fees paid by other users that send transactions. Then, the miner that solved the puzzle first broadcasts the new block to other nodes. Other nodes validate the block, and if it is valid, they update their node with the new block. This mechanism was used in the Ethereum blockchain till September 2022. Ethereum blockchain was using Proof of Work (PoW) as its consensus mechanism. Currently the Proof of Stake (PoS) consensus mechanism is used.

E. Proof of Stake

In Proof of Work, miners have to solve hard, cryptographic puzzle to add a block to a chain. The one that solves it first captures the ability to add a new block that other miners will validate next. It requires a lot of energy to solve such a puzzle, hence adding fraudulent block yields with rejecting that block by other users and not receiving reward and fees. This mechanism ensures the honesty of miners. One of the main reasons the Ethereum blockchain switched to the PoS consensus mechanism is energy consumption. Ethereum PoW consumed about 78 TWh/yr [27]. For comparison, in 2020, Finland consumed about 82 TWh [28]. Now, Ethereum PoS consumes about 0.0026 TWh/yr [27]. Besides energy consumption, the main difference between PoW and PoS is that PoS validators/miners explicitly stake their ETH coins as collateral. In case when a validator/miner does not behave honestly, their staked ETH coins can be destroyed as a penalty. There is also no need to solve any puzzle in PoS.

F. Smart Contracts

A smart contract is a piece of code that runs on a blockchain. It can be a set of instructions (functions) that can be invoked by sending a transaction from an Ethereum account. Every smart contract also has a state (data stored on the blockchain). Once deployed, a smart contract cannot be deleted or modified, and the state of a smart contract can only be changed via transaction. Users or other smart contracts can interact with smart contracts by sending a transaction. Such interactions are irreversible, so writing and verifying a smart contract carefully is essential since mistakes made in code cannot be reverted. Ethereum blockchain uses an Ethereum Virtual Machine (EVM) runtime environment for executing smart contract code and updating the state of the Ethereum network after each block is added to the chain. There are multiple smart contracts programming languages for blockchain, such as based on existing programming languages C++, Python, and JavaScript - Solidity. A simple example of a Solidity smart contract is shown in Fig. 1. This Hello smart contract stores a string message value and exposes a function for changing this value. To place a smart contract on blockchain, it needs to be deployed via transaction. To change the state of a smart contract, another transaction has to be sent to this smart contract with proper input data.

```
1.  pragma solidity ^0.8.7;
2.  contract Hello
3.  {
4.    string public message;
5.    constructor
6.      (string memory _initialMessage )
7.    {
8.      message = _initialMessage;
9.    }
10.   function setMessage
11.     (string memory _message ) public
12.   {
13.     message = _message;
14.   }
15. }
```

Fig. 1. Example smart contract

G. Layer 2

There is one main blockchain in the Ethereum network. This blockchain is widely called Layer 1 (L1). Before The Merge (switching from PoW to PoS), The Ethereum blockchain was able to process only about 30 transactions per second, and sometimes it took 10 minutes to process a single transaction. Such a slow network leads to making each transaction expensive. For comparison, Visa can process about 24,000 transactions per second [29]. The network was not fast enough for the second-largest blockchain network used worldwide. After switching to PoS in September 2022, Ethereum is now able to process about 100,000 per second. To reduce fees paid for executing a transaction and to increase the throughput of the network, scaling solutions were introduced. Those scaling solutions are generally called Layer 2 (L2). There are multiple L2s built on top of the Ethereum blockchain. Each Layer 2 is a separate blockchain and it extends the Ethereum base L1 blockchain as well as inherits its security. Scaling solutions utilize various mechanisms to speed up the network and make

it cheaper to use. Those mechanisms are not closely described in this article. The most commonly used L2s that run on top of Ethereum are Polygon, Arbitrum, and Optimism. Polygon L2 blockchain is used in this article since it is the largest scaling solution for the Ethereum blockchain. Polygon can improve transaction speed and lower transaction costs compared to the Ethereum blockchain.

IV. ZERO-KNOWLEDGE PROOF

A zero-knowledge proof (ZKP), sometimes called a zero-knowledge protocol, is used in cryptography as a method for one party to prove to the other party the validity of a statement. Two parties take place in such verification; a prover and a verifier. The prover can prove to the verifier that it owns a specific knowledge without disclosing the knowledge itself. In 1985, Goldwasser and Micali first described the core concept of a zero-knowledge proof [3]. An example use case for zero-knowledge protocol: Alice wants to buy a new house but first, she needs to apply for a mortgage load to a Bank. The Bank must know Alice income. Using ZKP, Alice can prove to the Bank that her income is in the required by Bank range without revealing the exact income value.

A. Interactive zero-knowledge proof

An example of proving possession of a specific piece of information without disclosing it can be a cave with a single entry and two pathways, path A and path B. The pathways are connected with a locked door with a cipher. There are two people, Alice (the verifier) and Bob (the prover). Bob wants to prove to Alice that he knows the door access code. To do that, he walks inside the cave and takes one of the two paths. Bob chooses a path randomly, and Alice does not know which one he takes. Next, Alice asks Bob to walk out of the cave using one of the two paths chosen by her. The only way for Bob to do that is to know the door access code. If Bob originally took path A, and Alice asks him to take path B back, he needs to know the code to the door to complete the task. To increase the probability of the fact that Bob has the knowledge of the access code, the experiment has to be repeated multiple times. The more attempts of the experiment will take place, the higher the probability that Bob does not guess the path that Alice chose. This is an example of an interactive zero-knowledge proof. Such verification requires multiple interactions between the prover and the verifier to minimize the possibility of guessing the poof by the prover.

The following criteria need to be fulfilled in a zero-knowledge protocol:

- **Completeness** — the protocol always returns true if the input is valid.
- **Soundness** — it is nearly impossible to trick the zero-knowledge protocol to return true if the input is invalid.
- **Zero-knowledge** — the protocol reveals nothing more than the statement (the proof).

B. Non-interactive zero-knowledge proof

The verifier can only discover the validity or falsity of the statement. The previous example shows how zero-knowledge protocol can be used to prove possession of specific knowledge without revealing the information itself. However, such an algorithm requires continuous interactions between the prover and the verifier. Sometimes such continuous interactions are impossible to maintain or highly cumbersome. Non-interactive zero-knowledge proof is the answer to purge the necessity of repeating the verification process multiple times. Bob can prove to Alice that he knows the solution of a Sudoku puzzle without showing the solution itself [30]. In order to do that, he places 3 cards with a proper number face down on every grid. Bob groups the cards into 9 rows, 9 columns, and 9 squares. That gives 27 groups of 9 cards. Using Sudoku puzzle principles, Alice can verify if, in every group, numbers from 1 to 9 occur only once. In such an approach, multiple interactions between Bob and Alice are no longer required. A described variant of the zero-knowledge protocol gives more scalability since it does not require constant interactions between the parties. A simple example of how zero-knowledge proof works can be Alice proving to Bob, that she knows the value of x, such that $f(x) : g^x = h \pmod p$, where g and h are publicly known values and p is a large prime number. It is simple to know x and compute $f(x)$, but it is difficult to do it conversely. Alice can prove to Bob that she knows x without revealing anything about the x. She can do it by choosing a random value r and sending $g^r \pmod p$ and $h \times g^{-r} \pmod p$ values to Bob. Then, Bob can verify that $g^r \times (h \times g^{-r}) \pmod p = g^x \times g^{-r} = h \pmod p$, without knowing the value of x.

There are two types of zero-knowledge proofs (ZKP): interactive and non-interactive. Since blockchain requires proofs to be generated through smart contract processes, non-interactive proofs are used. Zk-SNARK stands for "zero-knowledge succinct non-interactive argument of knowledge," and it is a ZKP-based protocol that is gaining popularity in blockchain applications. Compared to standard zero-knowledge protocols, the Zk-SNARK protocol offers additional features [31]:

- **Succinct** — The proof must be neat enough to be verified in milliseconds.
- **Non-interactive** — A single message from the prover to the verifier is required to execute the proof transcript.
- **Argument of knowledge** — It is nearly impossible to create a valid proof without access to the witness (the secret information), and the constructed proof fulfills the *soundness* requirement.

There are three main functions in the zk-SNARK protocol to conduct the proof computation and verification: G, P, V (Fig. 2).

The key generation function G (Fig. 2a) takes two inputs: a randomly generated secret parameter λ and a publicly defined program C. The output of function G consists of two publicly available keys: the proving key (pk) and the verification key (vk). These keys only need to be generated once for a specific program C and are not connected to externally owned account (EOA) keys.

The prover function P (Fig. 2b) uses the proving key (pk) generated by the G function, along with a public input x and a private witness w (which represents the secret information). The x parameter is a SHA256 hash of the witness w, meaning x does not need to be kept private because the original secret (w) cannot be reverse-engineered from x. The output of the P

function is a proof showing that the prover knows the secret information (witness) and that the witness is valid.

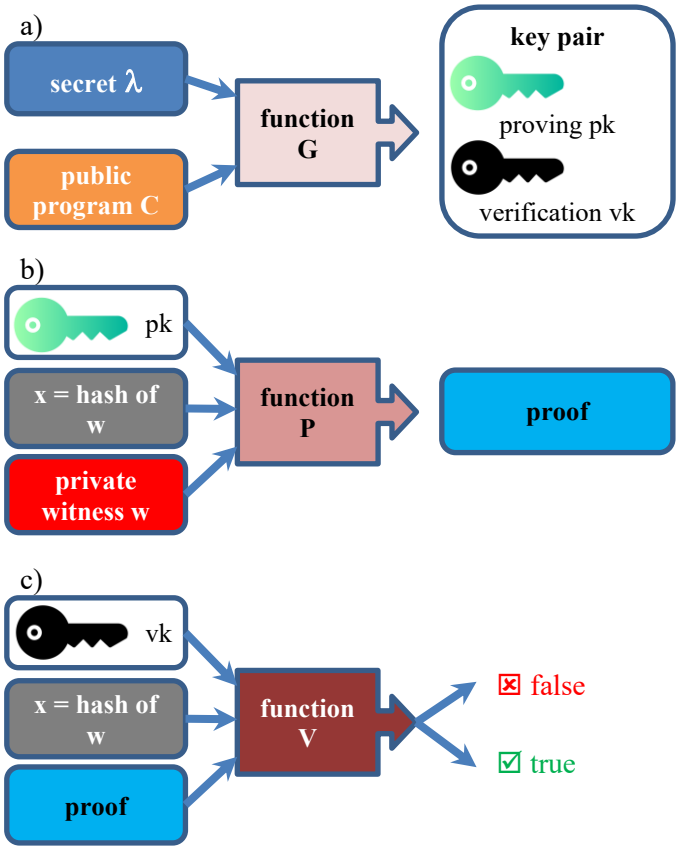

Fig.2. The ZKP functions: a) keys generation G, b) prover P, c) verifier V

The verifier function V (Fig. 2c) uses the verification key (vk) generated by the G function, the proof provided by the prover (P), and the public input *x*. The *x* value is the same as the one used when generating the proof in the P function. The V function checks the validity of the proof: if the proof is valid, it returns *true*; otherwise it returns *false*.

A vulnerability in this algorithm lies in the generation of public parameters, specifically the proving and verification keys, also referred to as the Common Reference String (CRS) [31]. The security of the zk-SNARK protocol depends on the integrity of this process. If someone has access to the secret λ parameter, they are able to create a fake proof that would look valid without knowing the secret *w* parameter. Fake proof can evaluate true for the verification function. It means that anyone who has access to the λ parameter is able to generate fake proofs. To increase the security of this operation, multi-party computation of public parameters was introduced. It requires multiple parties to participate in a trusted setup ceremony [32]. However this approach still requires honesty from each party. To remove the necessity of trust to other parties in the public parameters generation process zk-STARK (Zero-Knowledge Scalable Transparent Arguments of Knowledge) protocol was introduced. It is similar to zk-SNARK protocol with a few differences. The zk-STARK protocol is not closely elaborated since it is not used in this article.

V. Blockchain Auctions

Auction is a time-constrained selling/buying commodities process by placing bids. Sellers sell and buyers compete with each other to purchase goods. Such process can be executed in a traditional way - on-site or in modern way - online. There are multiple auction types. The most commonly used in real world are:

- **English auction** – the sellers starts the auction by announcing the initial price. The buyers compete with each other by placing their bids. Every next bid must be higher than the previous one. The highest bid is a winner.
- **Dutch auction** – an initial price is set to extremely high level. The price lowers in time until at least one bidder accepts the price. That bidder wins the auction and can purchase the commodity.
- **Blind auction** (sealed bid auction) – the bidders submit their secret bids to the seller sight unseen (not knowing other participants bids). The highest bid wins and the offered price must be paid for the commodity.
- **Vickrey auction** (sealed bid auction) – similar to the blind auction. The bidders also submit their secret bids to the seller sight unseen. The highest bid wins, but the price paid by the winner is the second highest bid.

Auctions can also be processed using blockchain technology and smart contracts. The seller creates a smart contract with the auction item information, minimum item price, and duration of the auction. Bidders can place their bids by sending a transaction to the auction smart contract. To ensure that the winning bidder will pay the price, a bidding transaction could also include a suitable amount of cryptocurrency that will be locked in the smart contract. After the auction ends, all lost bidders reclaim their locked funds.

The most popular blockchain auctions involve NFTs (non-fungible tokens) [33]. NFT is a unique, cryptographic, digital asset on a blockchain. NFT token cannot be replicated or divided into smaller pieces (there is only one, particular NFT token). These features of NFT tokens allow for representing real-world assets, like a piece of art or even a deed of ownership. NFT tokens are often associated with digital art. NFT auction operates very similarly to traditional auctions. An NFT auction smart contract takes possession of such NFT token (SC becomes its owner). Buyers make their bids by sending transactions with a specific amount of cryptocurrency to the auction smart contract. A smart contract can lock the funds of each buyer. After the auction ends, the smart contract makes the winner of the auction the new owner of the NFT token, sends the seller a proper amount of cryptocurrency, and gives back the remaining funds to other buyers. This approach covers every party involved in the process. The winning buyer gets the item, and the seller gets the charge.

VI. Solution and Architecture of Zero-Knowledge Proof Based Blockchain Auctions

A. Solution

Currently existing blockchain auctions give users as much privacy and security as blockchain technology itself. As explained in previous sections, anyone can read from a

blockchain and access the entire transaction history. However, there may be situations when a party would like to remain anonymous during auction. At times, it may be essential for a user to keep their interest in a commodity hidden from both other buyers and the seller. In this article a system that allows users to make bids without exposing their interest to other auction bidders and the seller is proposed. The participants privacy is maintained with the use of zero-knowledge proof. As a specific community, the bidders lodge their public keys to the auction administrator. Using those public keys, the auction system is created. To enter the auction, the user must create a ZK proof of private key ownership, associated with the previously lodged public key. The ZK proof must show that the user is the owner of the private key associated with the previously lodged public key and the public key is on the books of public keys eligible to participate in the auction. The proof reveals nothing about any public or private key. Such proof and bid must be sent to the Auction smart contract via transaction. The Auction smart contract verifies the correctness of the received proof. If the proof is valid, it means that the bidder is authorized to participate in the auction. To accept the transaction, the bid must be higher than the currently highest offer. ZKP is used in an action two times:

- first, ZKP proves that a user is eligible to participate in the auction; the auction administrator uses ZKPs prepare a list of potential bidders;
- second ZKP proves that a bid is placed by a user from the previously prepared list.

B. Architecture

A core component of the system is a smart contract that operates on a blockchain. The seller sets up the auction smart contract, including details about the commodity and a list of users who are permitted to take part in the auction. There is a registration process to attend an auction. Each user who wants to participate in the auction is required to send their public key to the seller in order to be counted on the list of public keys eligible to partake in the auction. During an auction, bidders can make bids by sending a blockchain transaction to this smart contract. To participate in the auction, users must send evidence that they are allowed to attend this auction. It could be a name and surname or some kind of identification number. This approach makes it quite effortless to identify the user identity. To hide the participant identity zero-knowledge proof was implemented in the solution. The contract stores a list of public keys associated with private keys that each bidder holds (each bidder holds their own keys) rather than easy-to-identify participant information like name or identification number. The bidder must provide cryptographic proof that they possess the private key linked to the public key listed in the smart contract. The zero-knowledge proof simply confirms whether the user is eligible to participate, without revealing any additional information.

Bidders can mask their interest in attending the auction by using a zero-knowledge protocol in blockchain auctions. Neither sellers nor any other participant will be able to tell if a user placed a bid or not. In this approach, each user could place their bid only once; otherwise, their cryptographic proof of private key ownership could be reused by any other person to change the right bid. To avoid such fraud, once a bid is made, another bid by the same user can be made using the same Ethereum EOA. The proposed application is decentralized (dApp) since it runs on a blockchain network and does not rely on a single entity (e.g., a single server). One of the most significant advantages of dApps is that once a decentralized application is launched, it runs autonomously, and basically nobody can control it. The system consists of two main elements: a blockchain smart contract and a web application that users interact with.

C. DApp architecture

Web application performs two main functions: calculates proof of private key ownership and sends such proof as a transaction to the auction smart contract using EOA. A user activity diagram is pictured in Fig. 3. To send a transaction to the auction smart contract; the user must have an Ethereum account (EOA) with the proper amount of cryptocurrency to pay for the transaction execution and conditionally the proper amount of cryptocurrency to prepay for the bid the user makes. There could be two possibilities in terms of paying the price for the commodity. The first option is to prepay the amount of cryptocurrency each time a user makes a bid and reimburse the user in case of not winning the auction. This requires multiple additional transactions to execute to return the cryptocurrency to the proper users. The second option is not to charge any amount of cryptocurrency during the auction and charge only the winner after the auction ends.

This approach does not secure the seller to receive the price for the commodity and does not require multiple charging and return transactions. However, it allows uses to participate in many auctions with EOA account balances lower than their total bids value. The EOA accounts were described in section III.A. To use the account in the application, user must connect a wallet. After connecting a wallet, the application is able to send a transaction to the smart contract on behalf of a user. To avoid misusing users wallets after connecting it to a dApp, each transaction must be authorized by the user. Authorization of transactions prevents malicious transactions from being performed by decentralized applications. When a wallet is successfully connected to the dApp, a user can insert a private key that is associated with a public key from the smart contract list of eligible public keys. The next step is to insert the amount a user is able to pay for the commodity. After these two values are inserted, the application has all the required data to send a transaction to the smart contract. The dApp sends two transactions to the Auction smart contract. The first transaction sends the proof hash calculated using the SHA256 algorithm. Smart contract stores the proof hash and the sender EOA address to make sure no other user will use the proof. The second transaction sends the actual proof and the bid. After the transactions are sent, the dApp must wait for the SC response. Suppose the proof and the amount were correct (the bid amount must be higher than the currently highest bid). In that case, the bidder will become a temporary winner, until another bidder makes a correct bid or the auction time elapse.

D.Smart contract architecture

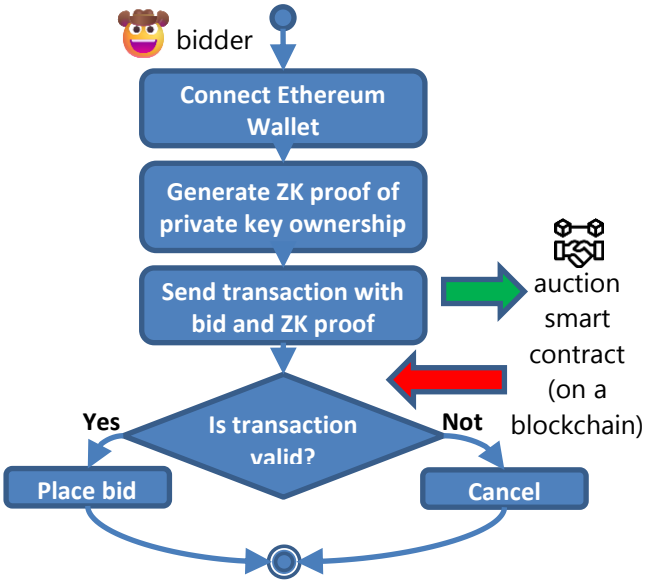

Fig.3. The dApp architecture

Smart contract performs two main functions: validates received from web application proof and records a currently winning bidder and their bid. To participate in the auction, a user must send a transaction with the proof SHA256 hash. A user has to send such hash only once for the smart contract to save the hash and the sender EOA address. When the hash and the address are stored in the smart contract, only the original sender will be able to use this proof to place a bid. The specific proof hash can only be saved once. If a user sends the proof hash to the contract via a particular EOA, it cannot be modified. After the user sends a transaction with proof and a bid, the smart contract starts its action. The auction smart contract first validates the proof. If the proof is valid, it means that a user is allowed to participate in this auction, and the smart contract goes to the second step. Otherwise, the smart contract stops its execution and sends information about this error to the application. Next, the smart contract validates if the bid is high enough (checks if the bid is higher than the currently highest bid). If the bid is correct, the smart contract saves the proof and the bid, and sets the winner to the sender of the transaction.

It is highly important to understand the difference between public and private Ethereum account keys and public and private keys to create proof to attend an auction. Public and private Ethereum keys are required to interact with a smart contract and to send a transaction. These keys are not related to keys for calculating the proof. The Auction smart contract activity is pictured in Fig. 4.

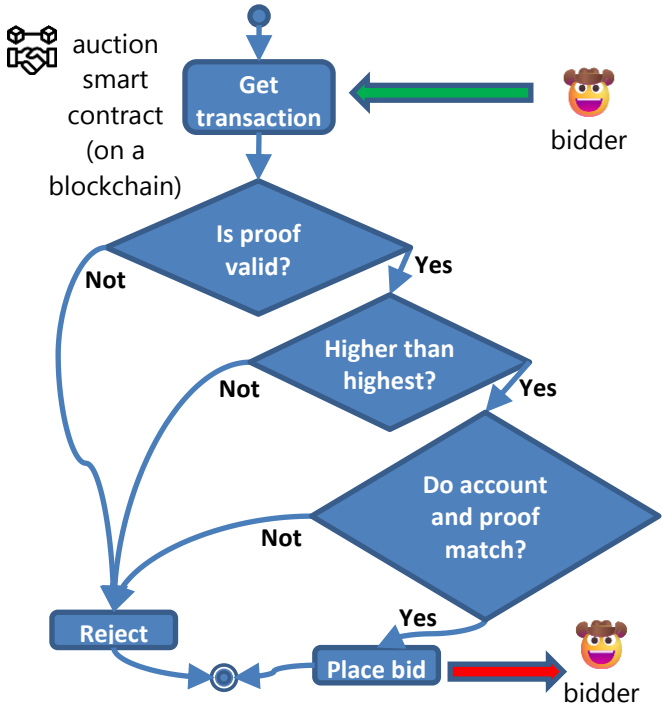

Fig.4. The smart contract architecture

E. Auction process architecture

The whole process starts with a user decision, that they start to participate in an auction. The user, via the dApp, connects their wallet. When it is done, the user can enter their public and private keys and a bid. After those two steps, the dApp is ready to send a transaction to the Auction smart contract. The smart contract verifies the transaction in terms of a proof and a bid. As long as everything goes correctly, the bid is saved in the smart contract. In case any of the steps fails, the bid will not be accepted and not be saved in the Auction smart contract.

VII. Implementation of Zero-Knowledge Proof Based Blockchain Auction

A. ZKP circuit implementation

The auction system was implemented using several tools: ZoKrates was used to create zero-knowledge proof procedures and generate Solidity smart contracts, React.js was employed for developing the web user interface, and Ethers.js was utilized to connect to and interact with the Polygon network and other Ethereum-based blockchains.

To prove ownership of a specific private key, zero-knowledge proof is used. A zk-SNARK circuit was developed using the ZoKrates language. The main function takes 3 arguments as input: a public list of public keys within the ownership will be verified, and two private values: a private key associated with a public key. Private arguments cannot be revealed during any step while using the circuit. The first step of this algorithm is ownership verification. In this step, the algorithm checks if the given public key matches the given private key. This step allows the algorithm to verify, if a user is the owner of the key they are trying to use. The second step is to check if the given public key is in the given list of public keys. The circuit returns true only if the given private and

public keys match and if the given public key in is the list. Otherwise, the circuit returns false.

B. Smart contract implementation

The ZoKrates tool enables a developer to generate a smart contract written in Solidity language from the ZoKrates circuit. Such a smart contract is used as a Verifier (Section IV.B) smart contract. This contract is a parent of the operative Auction smart contract. The Auction smart contract is the main contract that dApp interacts with. This contract has only one public function that can be called from outside of the contract. This is a makeBid() function. This function takes basically 2 arguments: the proof of the private key ownership and the user bid. The first step of the makeBid() function is to verify the provided proof. In this process, the verifyTx() function from the Verifier contract is utilized. If the proof is valid, the verifyTx() function returns true, and other steps of the main function are performed. Otherwise, the verifyTx() returns false, and the transaction is reverted with the "*User not allowed to participate in this auction*" message. After the first validation passes, the makeBid() function checks if the bid is higher than the currently highest bid. If the bid is high enough, the other steps of the main function are performed. Otherwise, the transaction is reverted with the "*Bid is not high enough*" message. The last verification step checks if the given proof has been used before. This check is performed because, as described in Section VI.D, the given proof can only be employed by one user (one EOA). If the correct EOA is used with the given proof, the other steps of the main function are performed. Otherwise, the transaction is reverted with the "*User not allowed to use this proof*" message. The last step of the makeBid() function is to save the winning bid and the winning EOA address.

C. DApp implementation

The developed Auction smart contract, described in Section VII.B, is basically a working auction tool. A user can send a transaction with proof and a bid via, for example, Polygonscan block explorer. To make the system user-friendly and easy to use, a web application has been introduced.

To use the web application, a user must first connect a wallet. In an as-is implementation, the dApp accepts only Metamask wallets. A user can accomplish it by clicking the *Connect wallet* button. This will trigger a Metamask wallet connecting mechanism. After the wallet is connected successfully, the user can view the auction object name and the currently highest bid. To place a bid, the user must enter their public key that is in the smart contract list, the private key associated with this public key, and a bid value. To send a transaction to the smart contract, the user must click the *Send* button (Fig. 5). To make sure that the user provided correct data, a confirmation is required to send the transaction - the user must click the *Confirm* button (that time it replaces the *Send* button). This action starts sending the transaction. The application uses previously connected EOA via Metamask wallet to send the transaction. Before the transaction is sent to the smart contract, the user must confirm it in the wallet - in this case, in the Metamask wallet. When the user confirms the transaction, it is sent to the Auction smart contract. If the transaction is accepted by the Auction smart contract, new value is displayed in the *HIGHEST BID* field, otherwise the user is informed about transaction being reverted.

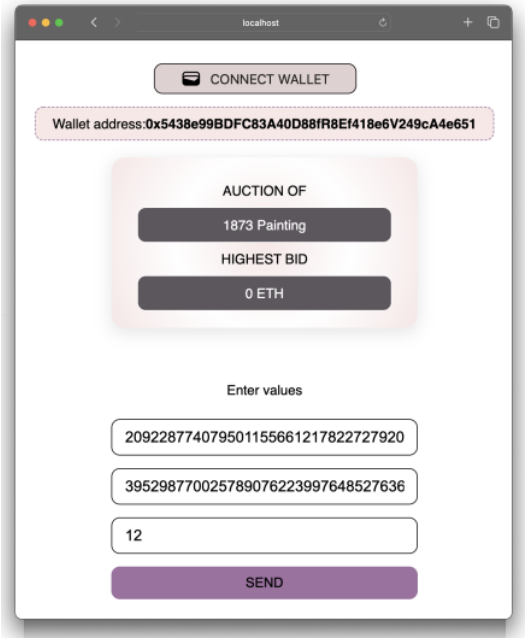

Fig.5. The auction web application

D. Tests

One of the most critical issues in smart contract development is contract security. It is crucial to create and test a contract code carefully, since once the contract is deployed to a blockchain, it can never be modified or removed. This makes it very important to make sure the code is well-tested, because it helps to identify issues and defects that can be fixed before the system is deployed to the blockchain. There are multiple already-defined security vulnerabilities. One of the known vulnerabilities is a front-running attack [34]. This attack involves dishonest miners who add new blocks to the blockchain. Such miners have access to the transaction that a user sends.

A user submits a transaction along with proof, but a dishonest miner could generate their own transaction using the same proof and submit it before the legitimate transaction is added to a new block. In this scenario, an unauthorized user could steal some other person proof and use it to place a malicious bid. To avoid this issue, the user must first send a transaction containing the proof hash, and only after that can they submit transactions with the actual proof and their bid. The auction smart contract keeps track of proof hashes and sender addresses to determine who is authorized to use which proof.

If a dishonest miner steals the proof hash and sends a transaction to the Auction smart contract with this hash, they will not be able to participate in the Auction either way, since the fraud is not able to generate the actual proof from its hash. Another well-known vulnerability is the re-entrance attack [34]. This attack does not concern the Auction smart contract since the contract does not invoke any external contract functions and does not contain any loops. The next vulnerability is function default visibility [34]. By default, functions in solidity are public, which means that anyone can execute them. To restrict some function executions, the function can be marked as e.q. private. There are two public

functions in the Auction smart contract. One of them is for saving the proof hash and the other is for placing a bid. Anyone should be able to invoke those functions. There is also one private function, which only the Auction smart contract can call. This function invokes a function to verify the proof. This function could also be public since it does not change any state of the contract. However, the user does not need to use it to participate in an auction..

1) Functional tests

To make sure every part of the smart contract works as expected, a set of functional tests were introduced. To write and execute tests, described in this section, Chai and Mocha frameworks were used. To enable testing Solidity smart contract, the Ethereum Waffle library was used. To write each functional test, the arrange-act-assert pattern [35] is utilized. This pattern helps to structure a test code so each step of a test is clearly separated and easy to read. The Make bid test in the Arrange section deploys the Auction smart contract to a local blockchain network that is emulated only for the duration of a test. It also creates a sha256 hash of the example zk-proof of private key ownership. In the Act section, a transaction is sent to the Auction smart contract with the proof hash. The last section, Assert, sends the transaction to the Auction smart contract with the actual proof and the bid. The expect() function checks if the transaction emits the BidMade event. This event is emitted only if the transaction is correct and the bid is saved in the contract.

In another functional test two users try to use the same proof. As in the previous test, the Arrange section prepares the environment. In the Act section, two transactions are sent. The first transaction saves the proof hash in the contract, and the second transaction makes bid. In the Assert section, one transaction is sent but using a different EOA address. This was done by connecting the "other" user as the transaction sender via auction.connect(other). The expect() function verifies if this transaction is reverted with a specific message; in this case, "*User not allowed to use this proof*".

In the functional test set, there are also tests that verify other, the most common user scenarios, such as sending invalid proof or trying to make an incorrect bid (not high enough).

2) End-to-end tests

To verify that the dApp cooperates with the blockchain contract as expected, the end-to-end (E2E) tests are introduced. E2E testing is a software testing technique that verifies an application workflow from the beginning to the end. "Happy path" is an example scenario of the E2E test. The proof and the bid are correct and the currently highest bid has been updated in the Auction smart contract. The user can see their bid in the dApp.

Another scenario used in the E2E test checks if the transaction will be reverted when a user tries to use proof already used by another user. The revert message is "*User not allowed to use this proof*". This verifies that malicious users cannot use proof that has already been used.

3) Financial analysis

In Ethereum blockchain, transaction cost is determined in gas. The amount of gas used to execute a transaction is the amount of computational effort needed to execute a specific transaction. It essentially means that the more smart contract code a node has to run to execute a transaction, the more gas will be used. There are two factors that impacts the transaction total cost: the amount of gas used and the current unit of gas cost. Executing two exactly duplicate transactions used the same amount of gas (computational effort to execute a transaction remains the same), but the gas cost may differ, since it is continuously changing. It is important to point out that in the time of testing the MATIC price was around 1.1065 USD and the base gas price is around 15 Gwei (1 Gwei is 0.000000001 MATIC). The total cost of deploying the Auction smart contract to the Polygon blockchain at the moment of writing this article looks as follows: gas used 2 537 390 units, the gas price 15 Gwei. The total cost is 0.00380608503806085 MATIC, which equals to 0.0042 USD. The total cost of sending a transaction to save the proof hash to the Auction smart contract looks as follows: gas used 46 117 units, the gas price 16 Gwei. The total cost is 0.000069175500737872 MATIC, which equals to 0.000077 USD. The total cost of sending a transaction to make a bid to the Auction smart contract looks as follows: gas used 317 747 units, the gas price 16 Gwei. The total cost is 0.000476620505083952 MATIC, which equals to 0.000529 USD. For comparison, if the Auction smart contract was deployed to the Ethereum blockchain, the total cost of would look as follows: gas used 2 537 390 units, the gas price 23 Gwei. The total cost is 0,05860023797619088 ETH, which equals to 93.01 USD. The total cost of sending a transaction to save the proof hash to the Auction smart would cost about 1.69 USD. Table I summarizes comparison between total transactions cost on Polygon blockchain and Ethereum blockchain. Executing transactions on Polygon blockchain as layer 2 for Ethereum blockchain is much cheaper than on Ethereum. The reason that Polygon blockchain has lower gas price is that it can process up to 65 000 transactions per second, whereas Ethereum can handle about 15.

TABLE I
COMPARISON BETWEEN TRANSACTION COSTS ON POLYGON AND ETHEREUM

service	deploy the auction [USD]	save a proof hash [USD]	make a bid [USD]
Polygon	0.0042	0.000077	0.000529
Ethereum	93.01	1.69	11.64

VIII. SUMMARY

A. Conclusion

In this article, we describe a decentralized system for anonymous blockchain auctions. The system comprises two main components: the Auction smart contract and a decentralized web application (dApp). The Auction smart contract maintains a list of public keys eligible to participate and includes a mechanism for zero-knowledge proof validation. The dApp enables users to create a ZKP of private key ownership and submit a transaction to the Auction smart contract with the proof and their bid.

This design allows users to participate in auctions without revealing their identity or whether they have joined the auction, thus maintaining privacy and keeping their interest in the commodity hidden from other buyers and even the seller. Users must create a ZKP to prove their authorization to

participate, but this proof discloses nothing about their identity.

The decentralized web application is crafted to make the ZKP calculation process effortless and user-friendly, requiring minimal effort to create the proof and send a transaction to the blockchain. Blockchain technology secures bids and prevents data manipulation; each transaction is recorded and can be verified by anyone, allowing users to trace their own transactions.

One of the challenges addressed was preventing malicious transactions from being sent to the Auction smart contract. Since anyone can read from the blockchain, it was crucial to prevent unauthorized users from reusing someone else's proof to place a fraudulent bid. This was achieved by sending a SHA256 hash of the proof to the contract before submitting the actual transaction with the proof and bid.

Additionally, the article presents a financial analysis of the system. Service charges are a crucial factor influencing a user's decision to use the system. The total cost from the user's perspective was compared between the Polygon and Ethereum blockchains. The analysis demonstrates that using the system on the Polygon blockchain is more cost-effective than on the Ethereum blockchain.

B. Future work

The proposed auction system is fully functional and could be deployed to a live blockchain network such as the Polygon Mainnet. However, there is room for improvement. The current system uses zk-SNARK as a zero-knowledge protocol. The zk-SNARK approach requires a trusted setup interaction between the prover and the verifier to generate a private λ parameter. This is a potential weak point of the protocol because anyone knowing this parameter is able to generate fake proofs that appear valid to the verifier. Utilizing zk-STARK, as mentioned in Section IV, could eliminate this vulnerability. Currently, there are no official zk-STARK tools that support smart contract generation, nor are there tools (to the authors knowledge) supporting zk-STARK in JavaScript or similar libraries.

Since existing blockchains that support smart contracts currently provide relatively low transaction fees, such auction systems can be strong competitors to current centralized auction providers like OneBid2. In centralized auction systems, the cost of placing a bid can be reduced to zero. This is impossible for a blockchain-based system, as a transaction sender must pay a transaction fee every time they send a transaction. However, as described in Section VII.D.3, utilizing the auction system on the Polygon blockchain is highly cost-effective. Nevertheless, transaction fees are constantly changing, and if they dramatically rise, the system will require adjustment. To ensure the system remains profitable, different types of fees could be implemented. For instance, an entrance fee could be applied to any user who places their first bid in the auction. Additionally, a service fee could be charged to the seller, which might be a percentage of the final sale price or a fixed amount. Zero-knowledge protocols are constantly evolving, much like blockchain technology. Our auction system is designed to be flexible and can be updated or adjusted as new features are introduced by the underlying technologies.

ACKNOWLEDGEMENTS

This article is a significantly extended version of the paper presented on Depcos-Relcomex 2024 conference [36].

REFERENCES

[1] A. McCain, “How fast is technology advancing? [2023]: Growing, evolving, and accelerating at exponential rates.” [Online]. Available: https://www.zippia.com/advice/how-fast-is-technology-advancing/.

[2] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System.” p. 9, 2008. [Online]. Available: https://bitcoinwhitepaper.co/

[3] S. Goldwasser, S. Micali, and C. Rackoff, “The knowledge complexity of interactive proof-systems,” in *17th annual ACM symposium on Theory of computing - STOC ’85, Dec 1985*, 1985, pp. 291–304. https://doi.org/\10.1145/22145.22178

[4] “Zcash.” [Online]. Available: https://z.cash/learn/.

[5] “Tornado Cash.” [Online]. Available: https://en.wikipedia.org/wiki/Tornado_Cash.

[6] “Polygon Solutions.” [Online]. Available: https://pexx.com/chaindebrief/lesson/polygon-solutions/.

[7] “ZKsync - The Elastic Chain.” [Online]. Available: https://zksync.io/.

[8] “Polygon Announces The World’s First Zero-Knowledge (ZK) Scaling Solution Fully Compatible with Ethereum,” 2022. [Online]. Available: https://polygon.technology/blog/polygon-announces-the-worlds-first-zero-knowledge-zk-scaling-solution-fully-compatible-with-ethereum.

[9] P. Bajari and A. Hortaçsu, “Economic Insights from Internet Auctions,” *J. Econ. Lit.*, vol. 42, no. 2, pp. 457–486, May 2004. https://doi.org/10.3386/w10076

[10] J. Trevathan, “Privacy and security in online auctions,” James Cook University, 2007. [Online]. Available: https://researchonline.jcu.edu.au/1788/2/02whole.pdf

[11] M. K. Franklin and M. K. Reiter, “The design and implementation of a secure auction service,” *IEEE Trans. Softw. Eng.*, vol. 22, no. 5, pp. 302–312, May 1996. https://doi.org/10.1109/32.502223

[12] J. A. Montenegro, M. J. Fischer, J. Lopez, and R. Peralta, “Secure sealed-bid online auctions using discreet cryptographic proofs,” *Math. Comput. Model.*, vol. 57, no. 11–12, pp. 2583–2595, Jun. 2013. https://doi.org/10.1016/j.mcm.2011.07.027

[13] I. A. Omar, H. R. Hasan, R. Jayaraman, K. Salah, and M. Omar, “Implementing decentralized auctions using blockchain smart contracts,” *Technol. Forecast. Soc. Change*, vol. 168, p. 120786, Jul. 2021. https://doi.org/10.1016/j.techfore.2021.120786

[14] G. Sharma, D. Verstraeten, V. Saraswat, J.-M. Dricot, and O. Markowitch, “Anonymous Sealed-Bid Auction on Ethereum,” *Electronics*, vol. 10, no. 19, p. 2340, Sep. 2021. https://doi.org/10.3390/electronics10192340

[15] W. Nowiński and M. Kozma, “How Can Blockchain Technology Disrupt the Existing Business Models?,” *Entrep. Bus. Econ. Rev.*, vol. 5, no. 3, pp. 173–188, 2017. https://doi.org/10.15678/EBER.2017.050309

[16] A. Sonnino, M. Król, A. G. Tasiopoulos, and I. Psaras, “AStERISK: Auction-based Shared Economy ResolutIon Markets for Blockchain Platforms,” in *2019 Workshop on Decentralized IoT Systems and Security*, 2019. https://doi.org/10.14722/diss.2019.230001

[17] T. Constantinides and J. Cartlidge, “Block Auction: A General Blockchain Protocol for Privacy-Preserving and Verifiable Periodic Double Auctions,” in *2021 IEEE International Conference on Blockchain Melbourne, Australia, 06-08 Dec 2021*, 2021, pp. 513–520. https://doi.org/10.1109/Blockchain53845.2021.00078

[18] J. Xiong and Q. Wang, “Anonymous Auction Protocol Based on Time-Released Encryption Atop Consortium BlockChain,” *Int. J. Adv. Inf. Technol.*, vol. 09, no. 01, pp. 01–16, Feb. 2019. https://doi.org/10.5121/ijait.2019.9101

[19] E. Boo, J. Kim, and J. Ko, “LiteZKP: Lightening Zero-Knowledge Proof-Based Blockchains for IoT and Edge Platforms,” *IEEE Syst. J.*, vol. 16, no. 1, pp. 112–123, Mar. 2022. https://doi.org/10.1109/JSYST.2020.3048363

[20] X. Yang and W. Li, “A zero-knowledge-proof-based digital identity management scheme in blockchain,” *Comput. Secur.*, vol. 99, p. 102050, Dec. 2020. https://doi.org/10.1016/j.cose.2020.102050

[21] B. Chen, X. Li, T. Xiang, and P. Wang, “SBRAC: Blockchain-based sealed-bid auction with bidding price privacy and public verifiability,” *J. Inf. Secur. Appl.*, vol. 65, p. 103082, Mar. 2022. https://doi.org/10.1016 /

j.jisa.2021.103082

[22] H. S. Galal and A. M. Youssef, "Verifiable Sealed-Bid Auction on the Ethereum Blockchain," in *2nd Workshop on Trusted Smart Contracts in Association with Financial Cryptography and Data Security, Nieuwpoort, Curaçao, 2 March 2018*, 2019, pp. 265–278. https://doi.org/10.1007/978-3-662-58820-8_18

[23] H. Li and W. Xue, "A Blockchain-Based Sealed-Bid e-Auction Scheme with Smart Contract and Zero-Knowledge Proof," *Secur. Commun. Networks*, vol. 2021, pp. 1–10, May 2021. https://doi.org/10.1155/2021/5523394

[24] M. Zhang, M. Yang, and G. Shen, "SSBAS-FA: A secure sealed-bid e-auction scheme with fair arbitration based on time-released blockchain," *J. Syst. Archit.*, vol. 129, p. 102619, Aug. 2022. https://doi.org/\https://doi.org/10.1016/j.sysarc.2022.102619

[25] R. Song, S. Gao, Y. Song, and B. Xiao, : "A Traceable and Privacy-Preserving Data Exchange Scheme based on Non-Fungible Token and Zero-Knowledge," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), Bologna, Italy,10-13 July 2022*, 2022, pp. 224–234. https://doi.org/10.1109/ICDCS54860.2022.00030

[26] Z. Wang *et al.*, "On How Zero-Knowledge Proof Blockchain Mixers Improve, and Worsen User Privacy," in *ACM Web Conference 2023, Austin, TX, 30 April – 4 May 2023*, 2023, pp. 2022–2032. https://doi.org/10.1145/3543507.3583217

[27] "What is an Ethereum Address?" [Online]. Available: https://info.etherscan.com/what-is-an-ethereum-address/.

[28] "Energy consumption in Finland." [Online]. Available: https://www.worlddata.info/europe/finland/energy-consumption.php.

[29] "A Deep Dive Into Blockchain Scalability," 2020. [Online]. Available: https://crypto.com/university/blockchain-scalability.

[30] B. Senabathi, "Zero-knowledge Proofs, the future of privacy friendly digital products," 2022. [Online]. Available: https://uxplanet.org/zkproof-4509f8c83959.

[31] "What are zero-knowledge proofs?," 2024. [Online]. Available: https://ethereum.org/en/zero-knowledge-proofs/.

[32] A. M. Matlala, "Setup Ceremonies," 2021. [Online]. Available: https://zkproof.org/2021/06/30/setup-ceremonies/amp/.

[33] "Non-fungible tokens (NFT)." [Online]. Available: https://ethereum.org/en/nft/

[34] A. Molina, "Solidity smart contracts common vulnerabilities," 2022. [Online]. Available: https://medium.com/coinmonks/smart-contracts-common-vulnerabilities-solidity-e64c5506b7f4.

[35] A. M. Matlala, "Arrange-Act-Assert: a Pattern for Writing Good Tests," 2020. [Online]. Available: https://automationpanda.com/2020/07/07/arrange-act-assert-a-pattern-for-writing-good-tests/

[36] M. Broniszewska, W. B. Daszczuk, and D. B. Czejdo, "Anonymization of Bids in Blockchain Auctions Using Zero-Knowledge Proof," in *19th International Conference on Dependability of Computer Systems DepCoS-RELCOMEX, Brunów, Poland, 1-5 July 2024*, 2024, pp. 19–28. https://doi.org/10.1007/978-3-031-61857-4_2