Exploring Privacy in Blockchain through ZoKrates: Fundamentals, Applications and Future Directions

GOSHGAR C. ISMAYILOV, Researcher, Turkey

Zero-knowledge proof is a special cryptographic technique that allows a prover to convince a verifier about the correctness of a claim without explicitly disclosing the claim itself. With the advancements of blockchain technologies, zero-knowledge proof has been successfully integrated into many decentralized applications over the years. ZoKrates, with its ease-of-use and direct integration to blockchain platforms, has emerged as a leading framework for developing, generating and verifying zero-knowledge proofs. This survey compiles a corpus of 347 documents that cite the original research work of ZoKrates by considering the period ranging from 2018 to 2025. Out of this corpus, this survey systematically selects and analyzes a total of 87 different documents including only peer-reviewed publications and excluding the gray literature. To the best of our knowledge, this is the first survey in the literature to follow a systematic approach to analyze the privacy-preserving applications in blockchain from the perspective of ZoKrates. This survey presents three different classifications over the documents with respect to (i) the applications they develop, (ii) the challenges they frequently encounter and (iii) the metrics they often use to measure performance of their techniques. Based on the challenges identified, this survey finally discusses numerous future research directions to promote potential advancements in the field and attract the attention of scientific and industrial communities. Feedback from readers regarding any inaccuracies or misinformation in this survey is welcome.

CCS Concepts: • **Security and privacy** → **Privacy-preserving protocols**; • **Computer systems organization** → *Peer-to-peer architectures*; • **Computing methodologies** → Distributed algorithms.

Additional Key Words and Phrases: Blockchain, Zero-Knowledge Proof, ZoKrates, Privacy, Security

1 Introduction

Blockchain is *lenticular*—with interpretations changing across different angles. As a data structure, it is a transparent, immutable and secure linked list of separate blocks. As a technology, it is a sophisticated stack of consensus models, cryptographic protocols and contract executions. As an infrastructure, it is a database platform to share the exact copy of global ledger across multiple physical nodes over a P2P (Peer-to-Peer) network. As a socio-economic model, it is a revolutionary paradigm to shift economy and governance from centralization to decentralization. It was first proposed in the seminal work of *Bitcoin: A Peer-to-Peer Electronic Cash System* [92] in 2009. Later, many challenging problems have benefited from its merits, (e.g., internet-of-things [1], healthcare [6], artificial intelligence [18], voting [57]). At the same time, blockchain has raised its own set of problems over this period (e.g., interoperability [9], security [13], scalability [63], privacy [102]).

The literature addresses privacy in blockchain via various techniques (e.g., trusted execution environment [41], zero-knowledge proof [112], secure multi-party computation [132]). Recent advancements in the blockchain ecosystem have particularly contributed to the theoretical foundation of modern zero-knowledge proof protocols (e.g., **zkSNARKs** [12]) and their practical applications (e.g., payments [49], authentication [77] and cross-chain communication [128]). Without loss of generality, modern proof protocols are often mathematically-complex and computationally-heavy. To mitigate this issue, several high-level development frameworks (e.g., Circom [10], ZoKrates [26]) have been proposed. There also exist no-code platforms leveraging LLMs (Large-Language Models) [51] for an automatic and faster development without prior domain and programming expertise.

Especially, **ZoKrates** has attracted significant attention in recent years and has been adopted in a diverse range of applications (e.g., digital identity [2], federated learning [27], voting [31] and energy trading [59]). The number of publications adopting it has been in a steady increase, (see

Author's Contact Information: Goshgar C. Ismayilov, Researcher, Istanbul, Turkey, goshgar.ismayilov@boun.edu.tr.

Fig. 4a), clearly justifying that attention. *However, studies about ZoKrates are highly-fragmented and lack coherent classifications over their contributions.* In addition, most of these studies tailor narratives only to their own theoretical and practical contributions. The lack of clear guidance in the field unfortunately compels novice researchers and practitioners to expend substantial time and effort while reinventing the wheel. These observations lead to the motivations of this survey.

Indeed, there exist comprehensive and well-summarized works in the literature to provide surveys about zero-knowledge proof in blockchain [99, 112]. But, they approach the domain from a broader perspective and overlook the critical aspects of ZoKrates, (e.g., its specific properties, practical challenges, development constraints, domain-specific language limitations). *Currently, there is no work that systematically reviews privacy and privacy-preserving applications in blockchain, from the perspective of ZoKrates.* Therefore, this article focuses on carrying out a systematic analysis to fill this literature gap and discusses a wide range of potential future research directions. The overall focus of this article is organized around the four different **RQs (Research Questions)**:

- **RQ1**: What are the emerging privacy-preserving applications of ZoKrates in blockchain?
- **RQ2**: What are the popular performance measures for ZoKrates in different applications?
- **RQ3**: What critical challenges and open issues are directed to ZoKrates?
- **RQ4**: What are the promising research directions of ZoKrates in the future?

1.1 Contributions

The main contributions of this article are summarized in the following way:

(1) *Novelty and Core Contribution.* To the best of our knowledge, this is the first article in the literature to perform a systematic analysis of privacy in the scope of blockchain through the lens of the ZoKrates framework.
(2) *Literature Coverage.* This article retrieves a corpus of 347 works from the literature by covering the period from 2018 to 2025 and out of this corpus, it selects 87 relevant works to answer the research questions identified.
(3) *Applications, Challenges and Future Directions.* This article systematically (i) analyzes the emerging privacy-preserving applications of ZoKrates from the literature, (ii) identifies the popular metrics to measure the performance of ZoKrates, (iii) draws attention to the current challenges and open issues and (iv) discusses the future research directions.
(4) *Classification and Taxonomy.* This article delineates three in-depth hierarchical classifications for the applications, performance measures and open challenges. In addition, it provides a high-level decision diagram to employ ZoKrates in the upcoming applications.

1.2 Organization

This article is organized into the following sections. Section 2 presents the background for zero-knowledge proof and reviews the related works from the literature. Section 3 discusses the systematic methodology to perform the survey by identifying data sources, search strategies and eligibility criteria. Section 4 reviews the works with respect to their applications. Section 5 discusses the popular performance measures in these works. Section 6 addresses the critical challenges and presents potential future works based on these challenges. Section 7 provides a high-level decision diagram to identify the need to ZoKrates. Finally, Section 8 concludes the article. Refer to Table 1 for the list of abbreviations used throughout this survey.

Table 1. List of abbreviations used in the article

Abbreviation	Description	Abbreviation	Description
API	Application Programming Interface	CLI	Command-Line Interface
CNN	Convolutional Neural Network	CUDA	Compute Unified Device Architecture
DApp	Decentralized Application	ETH	Ethereum Native Currency
EVM	Ethereum Virtual Machine	FedAvg	Federated Averaging
HTTP	HyperText Transfer Protocol	IoT	Internet-of-Things
IPFS	InterPlanetary File System	LLM	Large-Language Model
P2P	Peer-to-Peer	PSO	Particle Swarm Optimization
PlonK	Permutations over Lagrange-bases for Oecumenical Noninteractive Arguments of Knowledge	PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
R1CS	Rank-1 Constraint System	ReLU	Rectified Linear Unit
RQ	Research Questions	SHA	Secure Hash Algorithm
SVM	Solana Virtual Machine	TEE	Trusted Execution Environment
TSP	Traveling Salesman Problem	UAV	Unmanned Aerial Vehicle
USD	United States Dollar	VANET	Vehicular Ad Hoc Network
zkEC	Zero-Knowledge Evolutionary Computing	zkML	Zero-Knowledge Machine Learning
ZKP	Zero-Knowledge Proof	zkVML	Zero-Knowledge Verifiable Machine Learning
zkSNARKs	Zero-Knowledge Succinct Non-Interactive Argument of Knowledge	zkSTARKs	Zero-Knowledge Scalable Transparent Argument of Knowledge

2 Background and Related Work

This section provides background on the fundamentals of zero-knowledge proof and the ZoKrates framework. It also reviews the relevant surveys from the literature to underline how the contributions of this article differ from the prior surveys.

2.1 On Zero-Knowledge Proof

Loosely speaking, computer science relies on a set of complexities to assess algorithmic performance. Time and memory complexities describe how runtime and space of an algorithm grow as a function of input size. In this regard, **knowledge complexity** refers to the amount of knowledge revealed while proving the validity of a statement. Zero-knowledge proof is a special cryptographic technique that allows a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ about the correctness of a statement by disclosing no meaningful information (i.e., with the knowledge complexity of zero) about that statement except the validity of the statement itself. This counter-intuitive notion was first proposed in the seminal paper of *The Knowledge Complexity of Interactive Proof-Systems* [36] in 1980s.

A proof protocol for a language L between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ is said to be a zero-knowledge proof if and only if it satisfies the following three cryptographic guarantees:

- *Completeness*: If the statement x is true (i.e., $x \in L$), $\mathcal{P}$ can convince $\mathcal{V}$ about the correctness of that statement with a very high probability of $1 - \epsilon$.

$$x \in L \rightarrow \mathbb{P}[\langle \mathcal{P}, \mathcal{V} \rangle[x] = 1] > 1 - \epsilon \tag{1}$$

- *Soundness*: If the statement is false (i.e., $x \notin L$), $\mathcal{P}$ can convince $\mathcal{V}$ with only a very negligible probability of ϵ.

$$x \notin L \rightarrow \mathbb{P}[\langle \mathcal{P}, \mathcal{V} \rangle[x] = 1] < \epsilon \tag{2}$$

- *Zero-Knowledge*: Interaction between a prover and a verifier must not yield any meaningful information except the correctness of that statement itself. In other words, there must exist an efficient simulator $\mathcal{SIM}$ to generate an interaction that is indistinguishable from the

actual interaction for every verifier $\mathcal{V}^*$.

$$\langle \mathcal{P}, \mathcal{V}^* \rangle[x] \approx \langle \mathcal{SIM}, \mathcal{V}^* \rangle[x] \tag{3}$$

Zero-knowledge proof can be classified with respect to (i) the definitions of zero-knowledge and soundness properties (e.g., perfect, statistical, computational), (ii) the interaction model (e.g., interactive, non-interactive) and (iii) the problem constructions (e.g., number-theoretic, graph-theoretic). From the historical perspective, there has been a notable shift from interactive to non-interactive proof protocols to improve communication efficiency and succinctness. The introduction of blockchain has also accelerated the development of modern proof systems. Some of such popular systems include zkSNARKs [12], zkSTARKs [11] and Bulletproofs [15].

Zero-knowledge proof protocols can be expressed with their high-level functionalities (e.g., key generation, proof generation, proof verification). To start, an arithmetic circuit refers to a collection of arithmetic constraints to represent computation through some basic mathematical gates (e.g., addition, multiplication). Let C be an arithmetic circuit with an input $x \in \mathbb{F}$ and a witness $w \in \mathbb{F}$ to result in an output $C(x, w) \in \mathbb{F}$ over a finite field $\mathbb{F}$. A zero-knowledge proof has the following (but not limited to) functions:

- *Key Generation.* $(pk, vk) \leftarrow KEYGEN(1^\lambda, C)$: It generates a single pair of public proving key pk and public verification key vk over a security parameter λ and an arithmetic circuit C. The proving and verification keys are used during proof generation and verification, respectively.
- *Proof Generation.* $\pi \leftarrow \mathcal{P}.PROVE(x, w, pk)$: It allows the prover $\mathcal{P}$ to feed the input x, witness w and proving key pk to compute a proof π without disclosing the witness. This proof is later submitted to the verifier $\mathcal{V}$ for verification.
- *Proof Verification.* $0/1 \leftarrow \mathcal{V}.VERIFY(\pi, x, vk)$: It allows the verifier $\mathcal{V}$ to accept or reject the proof π with public inputs x and verification key vk. Note that $\mathcal{V}$ does not use the witness w during verification.

2.2 On Privacy-Preserving Off-Chaining Framework: ZoKrates

ZoKrates [26] is a privacy-preserving framework for off-chain proof generation and on-chain proof verification, based on zkSNARKs [12]. The main contribution of ZoKrates to the literature is to outsource complex computations from blockchain to the external (i.e., delegated) nodes while still guaranteeing their correctness through public verifiability of proofs generated off-chain. This drastically improves cost-efficiency (e.g., gas consumption) and practicality of privacy-preserving applications in blockchain. In addition, it defines a high-level domain-specific language so that developers can easily represent their arbitrary computations.

ZoKrates relies on six modules (i.e., parser, flattener, witness generator, R1CS converter, libsnark library [73] and contract generator) where their relations are shown in Fig. 1. ZoKrates executes these modules to proceed with the following workflow: (i) code compilation, (ii) trusted setup, (iii) witness generation, (iv) proof generation, (v) contract generation and (vi) proof verification, (see Fig. 2 for the complete workflow). This article describes this workflow in the following way:

- *Code Compilation*: It parses and flattens initial program code to compile it to a verifiable computation scheme. This compilation expresses the high-level instructions as a set of mathematical constraints. It raises any syntactic and semantic error (e.g., array out-of-bounds) in case of unsuccessful compilation as well. The resulting verifiable scheme will be subsequently used for key and proof generations in the next modules.
- *Trusted Setup*: ZoKrates performs an initial trusted setup ceremony to generate proving and verification keys used for proof generation and verification, respectively. However, this setup

is trusted where it requires the secret data of that ceremony (i.e., toxic waste) to be forgotten afterwards for protocol soundness (see Section 2.1). If an adversarial party compromises it, that party might generate fake but still valid proofs. In addition, the trusted setup of ZoKrates is not universal, requiring a new ceremony every time there is a modification in the program.

- *Witness Generation*: It executes the result of compilation with the user-provided public and private inputs to compute all necessary variable assignments (i.e., witness). This phase requires the correct set of inputs for a successful witness generation. The witness includes sensitive data and hence, should not be disclosed.
- *Proof Generation*: ZoKrates relies on the external *libsnark* library [73] for proof generation. This is the most challenging and compute-intensive phase where its complexity varies regarding the complexity of the initial program (i.e., the number of constraints in the circuit). This phase requires a corresponding proving key to be available to generate a succinct (i.e., short) proof. This succinctness is especially critical for network and storage efficiency.
- *Contract Generation*: It automatically exports self-reliant smart contracts with a proper verification scheme (e.g., Groth16 [38]). Contracts are embedded with verification keys to be able to verify proofs on-chain. Their instructions are written in Solidity and can be executed only on **EVM-compatible** blockchains (e.g. Ethereum, Avalanche). Other contracts can transfer proofs to these contracts via external calls to the public *verifyTx* function. This function accepts only proof and public inputs to return a boolean variable for proof validation. The caveat is that contracts need to be re-generated in case the initial programs change.
- *Proof Verification*: There exist two different approaches for proof verification: (i) *off-chain* to locally verify using CLI and (ii) *on-chain* to globally verify in a decentralized network. This phase is computationally lightweight and predictable since proof size and public inputs are constant in zkSNARKs. This phase also does not require any secret inputs since their on-chain submission would naturally compromise privacy.

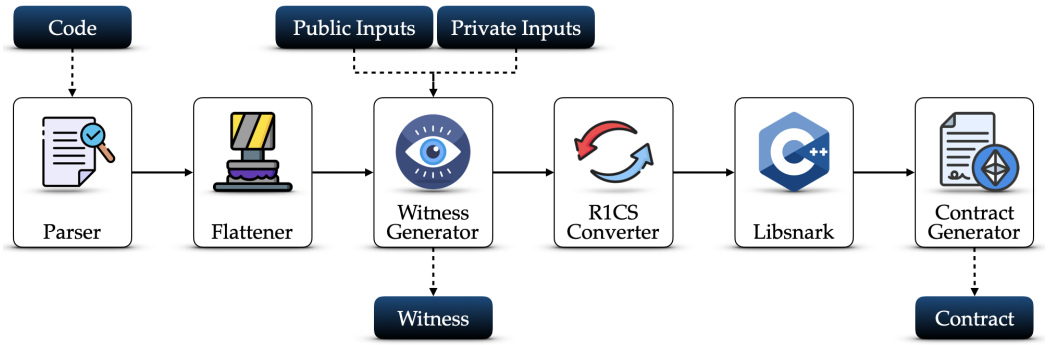


Fig. 1. Architecture of ZoKrates

2.3 Related Work

In the literature, there exist several survey papers that focus on zero-knowledge proofs and their applications in blockchain [89, 99, 112, 121]. Sun et al. [112] provide an overview of the existing proof protocols (e.g., zkSNARKs [12], Bulletproofs [15]) and evaluate their current states in blockchain

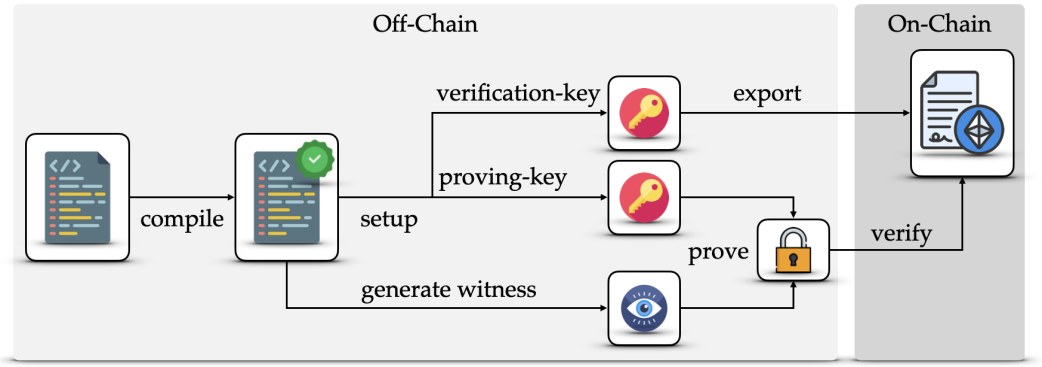


Fig. 2. Workflow of ZoKrates

from the perspective of some popular implementations (e.g., Hawk [70], Zerocoin [87]). However, it is not sufficiently comprehensive to cover the majority of privacy-preserving applications in the literature. Partala et al. [99] also focus on a specific set of practical applications for privacy-preserving transactions, assets and smart contracts. As stated in [99] itself, the main focus is rather on the zero-knowledge argument schemes including their high-level compilers (e.g., ZoKrates [26]) and low-level tools (e.g., libsnark [73]). Furthermore, the domain has progressed significantly since their publications of [99] in 2020 and [112] in 2021 (see Fig. 4a).

Morais et al. [89] focus on zero-knowledge range proofs and their efficient construction. It presents various construction strategies for this purpose (e.g., square decomposition, signature-based and Bulletproofs construction). Its empirical study reveals that Bulletproofs is the most efficient approach to implement range proofs in general. It reviews some practical applications as well such as mortgage risk assessment, investment grading, electronic voting and auctions. As one of the latest surveys, Xing et al. [121] approach zero-knowledge proofs on communication networks from the viewpoint of verifiable machine learning. Overall, there is currently no survey in the literature that performs a comprehensive analysis of privacy-preserving applications in blockchain from the viewpoint of ZoKrates.

Wust et al. [119] present a decision diagram to identify which blockchain solution is more beneficial by considering certain problem-specific requirements. The solutions considered are permissionless, public permissioned, private blockchains or no blockchain at all. Ernstberger et al. [32] provide another decision diagram to answer which proof system is useful to outsource a computation. However, applicability of this diagram directly to ZoKrates is limited due to the framework-specific requirements and considerations (e.g., trusted setup of ZoKrates). Therefore, inspired from these two works, this survey provides a similar high-level but more ZoKrates-tailored diagram by considering its cryptographic, language, environmental and operational requirements.

3 Systematic Survey Methodology

This survey follows a systematic methodology (i.e., *Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA)* [88]) (i) to identify the data sources and search strategies, (ii) to define the eligibility criteria with exclusion decisions and (iii) to minimize any authorial bias as well. Refer to Fig. 3 for the complete *PRISMA* flowchart.

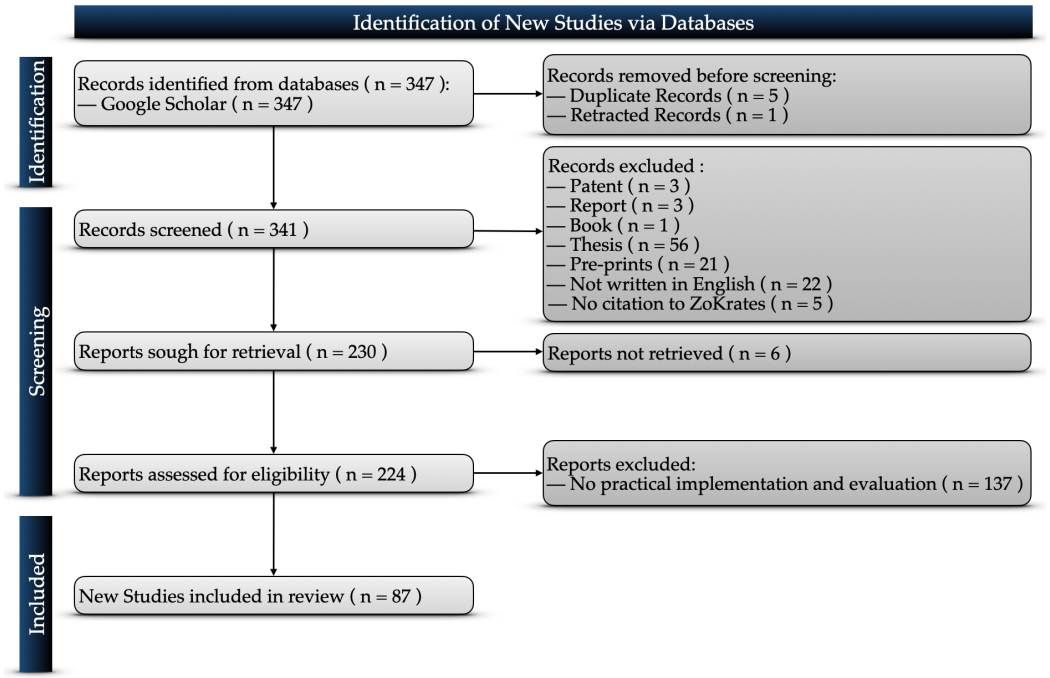


Fig. 3. PRISMA Flowchart

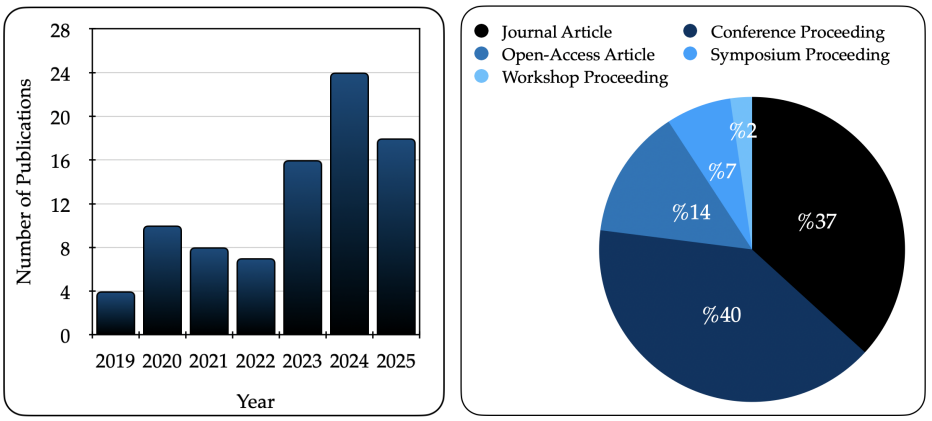
3.1 Data Source and Search Strategy

This article considers *Google Scholar* as the primary data source since it comprehensively indexes the scholarly articles and provides easy access to the citations of publications. Using the *cited-by* feature of Google Scholar, the article retrieves the initial corpus of records and later reinforces this corpus with the records from the other data sources (e.g., IEEE Xplore, ScienceDirect). This search strategy results in a total corpus of 347 records. Regardless of data sources, this article follows the assumption that any publication employing ZoKrates must cite its original work, namely [26].

3.2 Eligibility Criteria

The resulting corpus of the documents between the years of 2018 and 2025 passes through a manual screening with respect to exclusion criteria of this article. In the first stage, duplicated (5 records) and retracted (1 record) documents are eliminated based on their titles and contents. In the second stage, the documents that are not written in English (22 records) or do not cite the original work of ZoKrates in their references (5 records) are eliminated. In the next stage, the documents from the gray literature (i.e., not undergoing a scientific peer-review process) are excluded such as patents (3 records), technical reports (3 records), white-papers (0 records), books (1 records), university thesis (56 records) and pre-prints (21 records) where a total of 230 documents (including journal, conference, symposium, workshop and poster papers) remain left. Unfortunately, 6 documents cannot be retrieved from any scientific databases on the internet. At the last stage, this article excludes the documents that do not present any empirical analysis and evaluation of ZoKrates in blockchain (137 records). This results in a total of 87 different documents to be thoroughly reviewed to answer the research questions identified and enrich the overall discussions. Refer to Fig. 4 for the statistical distribution of these publications. The list of **exclusion criteria** is given as follows:

- Duplicated and retracted records are excluded.
- Records not citing the original ZoKrates work are excluded.
- Records written in languages other than English are excluded.
- Gray literature (e.g., patents, reports, thesis, white-papers, books, pre-prints) is excluded.
- Records with no empirical analysis and evaluation of ZoKrates in blockchain are excluded.



(a) Distribution over Years (b) Distribution over Research Types

Fig. 4. Distributions of Resulting 87 Publications over Years and Research Types

4 Emerging Privacy-Preserving Applications of ZoKrates in Blockchain

This section identifies the privacy-preserving applications of ZoKrates in blockchain to answer the first research question (**RQ1**). It delineates a taxonomy by asking the following two critical questions: (i) **task** identification: why ZoKrates is used (e.g., access control, integrity) and (ii) **application** identification: what ZoKrates is used for (e.g., federated learning, healthcare). Refer to Table 2 for a summary in terms of publication year, task, application, blockchain, security analysis, open-sourceness and key limitation. Security analysis classifies a work with respect to formal analysis (●), informal analysis (◐) and no analysis (○). Open-sourceness assesses whether a work includes a direct link to a publicly accessible repository (e.g., GitHub) in its body.

Before moving forward, it is crucial to provide the descriptions of the tasks used in task identification to prevent potential ambiguities in their meanings:

- **Access Control** refers to the mechanisms of authentication and authorization where authentication checks the identity of entities (i.e., who they really are) while authorization checks their permissions and access rights (i.e., what they are allowed to do).
- **Integrity** refers to the authenticity and immutability of data (or state) without disclosing data itself (e.g., root hash in Merkle tree, patient medical records in healthcare).
- **Compliance** refers to the adherence of data (or action) to certain regulatory constraints and domain-specific rules (e.g., verifying temperature with range proof).
- **Verifiable Computation** refers to the correctness of a complex arbitrary computation without disclosing the input data of computation (e.g., training model in federated learning).

When the work falls within the scope of multiple tasks and applications, this article intentionally diversifies its classification in order to address a wider audience.

4.1 Finance

ZoKrates has been integrated into finance to provide compliance of some business actions to the existing regulations and agreements. For instance, Kalra et al. [60] focus on the asymmetry between transparency and privacy in which too much transparency might reveal investment strategies of fund managers while non-verifiable privacy might reduce accountability to the individual investors. Therefore, the work requires fund managers to convince investors about the range of risks taken with respect to the minimum and maximum risk thresholds. Indeed, **threshold-based approach** is not only effective to build range proofs, but also relatively straightforward to implement in ZoKrates. Here, adopting a more intricate risk calculation (e.g., Monte Carlo simulation) for real-life scenarios is likely to be non-trivial.

Similarly, Peter et al. [103] propose an auditing mechanism over privacy-preserving transactions against anti-money laundering and financing terrorism. It enables an auditee to verify the following five compliance items to an auditor: (i) sender must have sufficient balance to transfer assets, (ii) the assets must be deducted from sender, (iii) the assets must be given to recipient, (iv) recipient must be in whitelist and (v) Merkle root hashes must be correct. For performance concerns, the work addresses these items under three separate proofs as balances proof for the initial three items, whitelist proof for the fourth item and Merkle proof for the fifth item. This approach is similar to **vertical batching** proposed in the work [39], which splits a complex program into several sub-programs for efficiency. The drawback of the work [103] is that it does not clearly state how an auditee can access the secret transaction details of others (e.g., transaction amounts). This might raise some trust issues against the auditee (e.g., single point of failure, collusion).

Not only compliance but also identity authentication is addressed in finance. For a credit investigation system, the work [126] appoints a trustworthy agency to own, control and provide credit reports to the consumers once they pass identity authentication. The work requires the transmission of proving key so that the consumers can generate their identity proofs, which leads to additional network and storage overhead. Both works [103, 126] suffer from the centralization around trusted entities (i.e., auditee in [103], agency in [126]).

4.2 Federated Learning

Federated learning refers to distributed machine learning techniques to train a global model with the data of local nodes without disclosing the data itself. Heiss et al. [44] address globally-verifiable off-chain training of a custom neural network over private datasets. The workflow requires each trainer to retrieve the model parameters (e.g., weights, biases) from blockchain at first, executes a publicly-verifiable training over private dataset and returns the resulting model with a corresponding proof. This work is noteworthy for implementing the feed-forwarding and backpropagation algorithms by using the limited expressivity of ZoKrates. However, the work still suffers from two issues: (i) the simplicity of the model tested with only one hidden layer and (ii) the storage of model parameters directly in contract. For the second issue, integration of a decentralized storage layer might help. For instance, Ebrahimi et al. [27] prefer storing model parameters on IPFS.

While the work [44] focuses on neural networks, Gu et al. [40] follow a more comprehensive approach by comparing three different models (including logistic regression, k-nearest neighbor and neural network) with five different training strategies. These strategies include (i) enclave-based (refer to the work [17] for TEE in healthcare), (ii) verifiable computation, (iii) voting-based, (iv) incentive-based and (v) distributed byzantine. Similarly, Aziz et al. [7] address multiple models including k-nearest neighbor, decision tree and CNN for a general-purpose zkVML (Zero-Knowledge Verifiable Machine Learning). This work is noteworthy to introduce modular and reusable libraries

to be imported for matrix multiplication, two-dimensional convolution, distance calculation, max-pooling, ReLU and tree traversing. These libraries serve as fundamental blocks for building larger applications, enabling developers to focus on higher-level design concerns without the need to manage low-level details. This work also measures the overhead of zkVML through an ablation study in which the execution times increase from 30 to 207 seconds for CNN with 200 parameters.

Keshavarzkalhori et al. [62] concentrate on the specific problem of **model poisoning** in federated learning. Later, Keshavarzkalhori et al. [61] complements the previous work by addressing the concern of **data poisoning** as well. The work states the following data poisoning attacks: (i) using artificial instead of genuine data, (ii) changing data labels for misclassification and (iii) ignoring valid datasets for poor accuracy. The proposed scheme protects the integrity of genuine data by verifying their hash values in ZoKrates. It applies optimization (e.g., grouping data for batch verification) since hashing every data point excessively would be expensive, increasing the circuit size and proof generation times. The work [74] also addresses model and data poisoning by ensuring end-to-end learning integrity with data source verifiability.

Beyond conventional federated learning pipelines, there exist some unique applications as well. For instance, the following work [29] focuses on verifiable machine un-learning to ensure removing data (e.g., to comply with the **right-to-be-forgotten** of individuals) from training dataset and to re-train the model. The following work [93] incorporates PSO (Particle Swarm Optimization) instead of the traditional FedAvg (Federated Averaging) technique for model aggregation in the problem of breast cancer prognosis and empirically achieves more accurate models. In scope of federated learning, ZoKrates is mostly employed to guarantee the computational verifiability of model training and inference. However, it has persistent limitations across the works in terms of the number of hidden layers [44] and the number of total parameters [7]. Reflection of these limitations is visible in execution times (e.g., 9425 seconds for 3700 parameters [7]) and circuit size.

4.3 Networking

ZoKrates has been extensively integrated into various problems within the domain of computer networks. For instance, Chen et al. [20] propose an authentication scheme for vehicular ad hoc networks (VANETs) to prevent cyber-attacks to resource-constrained and dynamic network environments. The scheme consists of four main phases as (i) initialization to setup public parameters, (ii) registration to register road-side units and vehicles, (iii) authentication to login vehicles anonymously with their identities and (iv) communication to share data from vehicles to road-side units (e.g., location, weather, traffic data). The security analysis considers both formal and informal validations including several guarantees (e.g., unforgeability, anonymity, zero-knowledge, unlinkability) and attacks (e.g., node collusion, information tampering, replay, denial-of-service). The other work of the same authors [130] now focuses on traffic safety data (e.g., road congestion information) that is shared between requesters and providers.

The following joint works [33, 34] provide two different authentication schemes (token-based and commitment-based) for connected electric vehicles. In the first scheme, charging service provider returns a service token after verifying the authentication proof of the vehicle while the second scheme allows the vehicle to commit a certain charging slot by eliminating additional overheads of the first scheme (e.g., scheduling transaction). Hence, the second scheme is empirically evaluated to be more efficient. Xu et al. [122] also focus on the vehicle charging service, now following a different approach in which the proofs of the charging schedule are signed with ring signatures without explicitly disclosing who is responsible for signing in the set of members. For mobile crowd-sensing networks, Wang et al. [114] employ a hybrid blockchain architecture where multiple private cluster chains are hierarchically connected to a single public chain. In this architecture,

each cluster chain is represented through a cluster head node with the highest balance and the identity authentication of such nodes to the public chain is protected with zero-knowledge proof.

For digital identity management of network devices, there exist several works [43, 125] that follow the **issuer-holder-verifier** trust model. Issuer refers to the central entity to issue credentials for holders while holders collect data (e.g., through their sensor measurements) and verifiers verify the correctness of these credentials and later consume data. Heiss et al. [43] particularly address privacy in decentralized physical infrastructure networks to protect device credentials (e.g., their identities through privacy-preserving authentication) and device attributes (e.g., their locations and minimum firmware version compliance). Yu et al. [125] present a system for secure data sharing between holders (i.e., IoT devices) and verifiers (i.e., service providers). Refer to the following works [2, 3] that rely on the same trust model to develop a traceability system for inter-organizational business processes. In addition, the following work [110] manages access controls of IoT devices in blockchain through a token-based mechanism.

As in federated learning, data poisoning is also among the concerns of networking. For instance, the work [74] extends its end-to-end verifiability to the data source in federated learning. Different forms of such corruption might arise in the context of networking as well. For instance, Heiss et al. [42] present the issue of undetected manipulation during preprocessing data on off-chain IoT nodes. Just to ensure end-to-end integrity, it proposes a trustworthy data processing scheme by integrating both zero-knowledge proof and enclave-based approach (i.e., TEE). Refer to the works [17, 40] for other enclave-based applications covered in this article.

4.4 Oracles

Oracle refers to the trusted service to supply external real-world data to blockchain. Gu et al. [39] propose an off-chain computation and storage service to solve two critical concerns: (i) trust to the untrusted nodes and (ii) inefficiency of proof generation. For the first concern, data sources (e.g., IoT sensors) are only responsible for collecting data while they are the off-chain provers to perform computations over raw data by generating proofs of these computations. For the second concern, it introduces two different techniques (i.e., **horizontal and vertical batching**) that result in 550x speed-up with respect to the baseline proof generation, according to the empirical study. The horizontal batching splits the datasets into several sub-datasets while the vertical batching splits the complete program into the several sub-programs. The techniques are evaluated on three tasks as key-value updates, logistic regression training and neural network inference. On the other hand, Park et al. [98] focus more on the authentication of external data to prove that the data owner does not manipulate the data. The distinct difference between these works is that the work [39] assumes that the data sources are honest but the provers are not about their computations while the work [98] specifically prefers to authenticate data sources themselves.

4.5 Cloud Computing

Cloud computing is among the recent paradigms to outsource computations to gain efficiency for several critical dimensions (e.g., cost, time, energy [50]). Dorsala et al. [24] focus on the verifiable cloud computing and the problem of fairness in cloud service payments. For verifiability, the work discusses two approaches: (i) proof-based by providing zero-knowledge proof for computational correctness and (ii) replication-based by collecting and comparing results of multiple resources. For fairness, it requires the service providers to provide only the correct results so that they can receive their service payments in return. However, the work considers a small input size (only 10 items) to test the performance of ZoKrates in three different operations (e.g., searching, sorting and primality). Liu et al. [79] propose a blockchain-based logging system for cloud computing to protect logs against malicious modifications. The system both ensures undeniability during auditing and

preserves user anonymity at the same time. However, the work has no empirical study regarding the performance in blockchain (e.g., on-chain verification gas consumption).

4.6 Metaverse

The metaverse refers to a common virtual environment that users interact and exchange digital assets on a decentralized network. Here, user authentication is especially critical to prevent illegal activities. Hence, Kim et al. [66] address this issue by verifying the identity of adults (e.g., verifying age without disclosing their birth dates). The work discusses the identity management over a special non-transferable and non-tradable non-fungible token (i.e., soulbound tokens), which might serve as a foundation towards the decentralized society. Ethereum already supports such token standards (e.g., ERC-5192, ERC-5727, ERC-6454). However, the work relies on trusted issuers to issue and store credentials across the system, which remains as an obstacle to achieve a fully-decentralized system. In addition, empirical justification of its applicability on real metaverse ecosystems through the growing number of users might be also beneficial.

4.7 Healthcare

Indeed, healthcare data is among the most sensitive personal information and its privacy must be protected against adversarial attacks in digital applications. Hence, Cao et al. [17] develop a privacy-preserving electronic health record authentication and insurance compensation scheme on a hybrid architecture (i.e., public and private blockchains). In the proposed design, an oracle server retrieves patient data from the hospital health record database and computes compensation with a proof. After its correct verification, the company transfers that compensation to the hospital. The oracle server relies on a trusted execution environment (TEE) to perform the computations. However, this design might suffer from the single point of failures of oracle servers. Yang et al. [124] also focus on the authentication of electronic health records based on range proofs of certain health attributes (e.g., age, blood pressure, glucose level). However, its empirical study considers only the age attribute, limiting insight about the actual cost when multiple attributes are involved.

Sharma et al. [108] propose a privacy-preserving nation-wide healthcare framework, based on zero-knowledge proof and proxy re-encryption. Differently from the previous works, this work focuses on identity verification of patients to access to their electronic health records. However, the claim of this work to be nation-wide also needs an empirical justification with the growing number of patients. Luong et al. [80] address identity authentication of IoT devices, collecting health data and sharing with the health service providers. However, this work requires the hash values to be stored inside the arithmetic circuit, which drastically increases the circuit complexity with the growing number of users. Proof generation times in its experiments increase from approximately 2 to 18 seconds for 1,000 and 20,000 users, which supports this observation. This proof design degrades system scalability and efficiency by requiring continuous software updates for every new users. Differently from the previous works, Egala et al. [28] follows a more complete healthcare framework by integrating IoT layer to collect data, edge computing to perform local data computation, fog computing to provide identity validation and cloud computing to perform global computation.

4.8 Insurance

Insurance refers to a formal agreement where insurance companies provide financial protection and compensation for the losses of individual users. Previously, the works [17, 108] address insurance schemes in healthcare. Now, Itanyi et al. [56] focus on a privacy-preserving insurance claim system in blockchain for car incidents (e.g., accidents, theft). The work employs ZoKrates both for insurance authorization and identity authentication at the same time. For insurance authorization, the insurance company generates a proof to securely establish a mapping between the car owner

and the car itself. For identity authentication, the car owner generates a proof to verify identity without disclosing sensitive information (e.g., private key). Unfortunately, the work lacks the formal security and privacy analysis to justify its theoretical foundation.

4.9 Reporting

People might be hesitant to report serious incidents and situations to authorities for fear of revealing their identities. Nyato et al. [95] focus on near-miss (i.e., incidents that might have resulted in injury) reporting in constructions to incentivize proactive actions of workers and to cope with the data manipulation and poor traceability of the traditional systems. The work proposes privacy-preserving identity authentication and role-based (e.g., frontline staff, safety officer, project manager) authorization on Ethereum and Polygon. In the future, it might be beneficial to integrate the ideas from the works [17, 108] to build a more holistic framework in which workers in constructions get compensation from insurance companies for their injuries if their private reports are approved. In addition, Musamih et al. [91] focus on privacy-preserving child labor identification and reporting on blockchain with zero-knowledge proof and machine learning. The learning model uses CNN for labor detection and classification on images. After detection, zero-knowledge proof is used to provide access control to the resulting labor reports. However, the work trains the model with a dataset collected primarily in Japan, which might degrade its generalization for the other regions around the world. Refer to the work [46] for verifiable emission reporting in supply chain.

4.10 Payment

Traditional blockchains are transparent by publicly presenting the transaction attributes (e.g., balances). However, such transparency can conflict with confidentiality requirements of certain real-world applications. Previously, the work [103] provides an auditing scheme over these attributes of privacy-preserving transactions. In that scheme, an external auditee has to access the attributes after the transactions are committed. Now, Ismayilov et al. [49] develop a privacy-preserving payment protocol on Ethereum where the sender generates its own proof to deposit assets to the contract and later the recipient generates another proof to withdraw. These two proofs together ensure the transactional integrity by preventing adversarial manipulation (e.g., creation of fake assets). The work also proposes a novel attack (i.e., balance range disclosure attack) to the privacy-preserving transaction systems in general by using the minimum cost flow networks.

Differently from the work [49], Li et al. [76] introduces the shielded pool to where users can deposit their plain tokens. This work relies on zero-knowledge proof to build a proof-of-ownership based authentication mechanism when users remove their assets from this shielded pool. The work addresses cost efficiency through roll-ups to batch multiple transactions. However, the gas consumption of the withdrawal operation is still very high, reaching approximately 30$ which substantially limits its practical usage. In the extended work [75], authors now provide quite rigorous cryptographic descriptions of the protocol, supporting with extensive formal security analysis (e.g., ledger indistinguishability, transaction non-malleability) and attack scenarios (e.g., double spending, front running, ledger manipulation). The protocol proposed in work [75] also supports privacy-preserving cross-chain payments.

The transfer protocols proposed in the works [48, 49] are similar from the perspective of two proof generations separately for token deposition and later withdrawal. In both works, token deposition proof is more complex because it additionally checks if balance is already sufficient. Later, both works require sharing the knowledge of transaction amounts to the receiver in a confidential channel. Moreover, the work [48] compares the performance of two different proof systems (i.e., zkSNARKs versus Bulletproofs) and prefers Bulletproofs with no trusted setup assumption. However, transfer protocols relying on both proof systems need huge improvements in terms of throughput, when

compared with the traditional VISA transfer system. Lastly, Boo et al. [14] propose a lightweight privacy-preserving payment protocol by considering the constraints of resource-constrained devices (e.g., IoT devices) and empirically show improvements in latency and energy consumption.

4.11 Energy Market

Involving prosumers with no trust to each other, the energy market requires privacy-preserving and verifiable solutions. Eberhardt et al. [25] propose a netting technique for households within a community. The technique requires sensors to commit their measurements to blockchain at first so that a netting server aggregates these measurements to obtain the net production value with a verifiable proof. The *3+1 layer model* in the work [101] peels energy market into four layers as: Layer-0 for physical infrastructure, Layer-1 for trustworthy computations, Layer-2 for applications/services and Layer-3 for market/regulations. It classifies the netting in the work [25] as a post-consumption activity of Layer-1. In addition to netting, the work [101] provides the other use cases of energy trading in blockchain as tokenization, accounting, contracting and compensation.

Jiang et al. [59] focus more on trading of energy from suppliers to buyers in the existence of a dispatching station where suppliers submit verifiable reports about their trading actions. The station verifies both (i) authentication of suppliers and buyers and (ii) data integrity over energy trading to prevent malicious modifications. However, both works [25, 59] suffer from the centralization around trusted entities (i.e., netting server in [25], dispatching station in [59]). In the future, the work [59] might mitigate this issue by replacing that station with a committee of multiple stations. Differently from these works, the work [127] approaches energy markets from the perspective of optimization (i.e., maximization of total revenue) and implements the Simplex algorithm in ZoKrates to solve it.

4.12 Cross-Chain Interoperability

Cross-chain interoperability refers to the ability of different blockchain networks to securely exchange data and digital assets. Westerkamp et al. [117] provide a verifiable chain relay to enable cross-chain state proofs from Bitcoin to Ethereum. First, the relay collects multiple block headers (rather than a single header) from the source chain to construct Merkle proof on ZoKrates. Later, it submits the resulting proof to the target chain for verification. This approach enables constant-size efficiency in proof verification, regardless of the number of headers included. There exists a trade-off between resource consumption and cross-synchronization since generating proofs more frequently to keep chains synchronized requires more computational consumption. But, non-synchronization due to the infrequent generations might cause finality issues (e.g., resulting in lost assets).

Similar to the previous work, Wei et al. [115] address efficient (constant verification gas for arbitrary batch size) cross-chain communication from Bitcoin to Ethereum, but by following a more systematic approach for security considerations (e.g., reusable randomness, difficulty adjustment, compositional security) and attacks (e.g., upfront mining attack). However, the work faces the limitation of off-chain proof generation latency for real-time applications, ranging between 25 and 42 seconds. Refer to the works [75, 84] as well for cross-chain communication in licensing and payment applications, respectively.

4.13 Voting & Auctions

Voting is among indispensable aspects of modern democracy and perfects illustrations of collective human decision-making. Blockchain has been integrated to mitigate the drawbacks of traditional voting. Emami et al. [31] propose a privacy-preserving election framework on Ethereum to protect the privacy of votes. The work considers distinct actors including election authority, ballot boxes, voters and decryptors. Especially, ballot boxes refer to the group of volunteer servers residing

in a public network to collect and tally encrypted votes via verifiable proofs. Although the work naively introduces ballot boxes to improve scalability, these boxes might refuse to submit votes to contracts. As stated in the work, users might choose another ballot box freely to re-submit vote in such a scenario, but it definitely hinders system practicality. Furthermore, this layer of ballot boxes presents additional attack surface.

Emami et al. [30] propose a privacy-preserving second-price sealed-bid auction on Ethereum for data trading. The work introduces a broker to retrieve, decrypt and sort encrypted bids to decide the winner with a verifiable proof. This proof is a justification of the correct computation the broker carries out without disclosing the bids themselves. The work provides a comprehensive analysis on assumption, security (e.g., for bid indistinguishability, collusion resistant, zkSNARKs verifiability) and requirement validation (e.g., privacy, public-verifiability, fairness, integrity, non-repudiation, anti-collusion). As stated in the work itself, the proposed mechanism might be further analyzed with respect to side-channel attacks. Overall, both works [30, 31] rely on central parties in their workflows (i.e., ballot boxes in [31] and broker in [30])

4.14 Data Sharing

Privacy-preserving data sharing refers to the exchange of data from multiple parties so that they can be jointly utilized while preserving their privacy. Ismayilov et al. [55] propose a privacy-preserving data sharing scheme on dual hypercube networks to support certain arithmetic operations (e.g., addition). The protocol requires data owners to securely pair-wise share their data encryptions until all the data are aggregated into their sum. The dual networks specifically improve system scalability with the increasing number of data owners with logarithmic overhead per party, $O(logn)$. However, the overall system overhead grows with $O(nlogn)$, which might degrade throughput for a platform with fixed-size blocks. Later, the work [53] extends this baseline protocol for privacy-preserving prefix sum and shows its applicability on voting with Euler Tour Technique.

Data sharing can also be a part of a larger system. For instance, Sober et al. [109] perform data sharing to achieve a verifiable distributed key generation scheme in blockchain. It follows the phases of (i) share distribution compute encrypted shares with symmetric encryption, (ii) dispute to issue a dispute against an invalid share and (ii) key derivation to compute and prove the shared key. Another work [8] performs data sharing to enable central servers to prove their compliance to certain privacy guarantees during noise generation (e.g., minimum noise threshold). On the contrary, the previous works [53, 55, 109] follow approaches with no central point of trust. For data sharing via IPFS, refer to the work [85] as well.

4.15 Crowdsourcing

Crowdsourcing refers to the practice of collecting data or solutions from a distributed group of individuals. To be more specific, **crowd computing** refers to leveraging a group of distributed resources to carry out a computational task cooperatively. For instance, the following work [72] proposes a general-purpose anonymous verifiable crowdsourcing framework in blockchain to protect privacy of individuals while still qualifying their resulting answers. Using verifiable crowd computing to solve optimization problems in blockchain is also applicable. For instance, Korbel et al. [69] propose a computational task offloading scheme (between service providers and consumers) and show its applicability over the well-known Traveling Salesman Problem (TSP). ZoKrates verifies the solutions to be constructed properly with respect to several constraints (e.g., cities appearing in path once, hash validations of cities) and the objective (e.g., length of total distance traveled).

Ismayilov [52] follows a similar approach to propose a privacy-preserving evolutionary computation in blockchain where multiple service providers search for optimal solutions at separate subspaces. As **zkML (zero-knowledge-based Machine Learning)**, this work introduces the

idea of **zkEC (zero-knowledge-based Evolutionary Computation)** for the first time in the literature. It empirically demonstrates the convergence of zkEC to optimal solutions via two simple benchmark problems where its further justification might be needed in more complex problems. The main difference between these works is that the work [69] focuses on the final solution itself (i.e., path) while the work [52] guarantees that the service providers follow a policy (i.e., evolutionary computation) while constructing their solutions. Finally, the work [54] employs **Bellman-Ford** negative cycle detection algorithm on ZoKrates to find bartering solutions while the work [127] implements Simplex algorithm for optimization of energy markets.

4.16 Licensing

Licensing refers to a formal agreement where a licensee leases certain rights and permissions from a licensor. Maesa et al. [84] address a privacy-preserving intellectual property license agreement on single network and multi-networks (i.e., network of networks) architectures. The work employs ZoKrates to outsource the total royalty fee computation off-chain without disclosing the sensitive data (e.g., sales figures). In multi-network architecture, the representative validators of the networks must cooperate to compute the total royalty fee along with corresponding proof through a secure multi-party computation. However, this cooperation and orchestration in the multi-network architecture increase the complexity for real-world adoptions. In addition, this work requires the licensor and the licensee to cooperate to destroy the toxic waste of ZoKrates initial trusted setup. Raj et al. [107] focus on software license management system in blockchain without disclosing critical software code and several metadata (e.g., validity period, license signature, company name).

4.17 Logistics & Supply Chain

Traceability and accounting (i.e., monitoring, reporting and verification) are among the major concerns of recent supply chain applications. However, revealing private business data for the sake of these concerns might compromise sensitive commercial information, leading to financial loss. Therefore, Heiss et al. [46] focus on verifiable off-chain carbon accounting to protect end-to-end data integrity and the privacy of business emission data (e.g., static allocation factor). The work shows the compliance of that business data to the existing carbon emission regulations. Werner et al. [116] address smart and self-organized logistic boxes for pharmaceutical transportation. The boxes prove that sensor data measured inside (e.g., temperature) is within the compliance range without revealing actual data. However, the work lacks empirical analysis to justify its practical feasibility. For instance, performing proof generation frequently might drain energy of these self-reliant boxes too quickly. Additionally, the scheme alone does not guarantee sensor reliability, (e.g., physical sensor tampering). Overall, both works [46, 116] are sensor-driven to measure various environmental parameters (e.g., carbon, temperature). For the application of relational-model multi-agent systems on food logistics, refer to the work [104] as well.

4.18 Unmanned Aerial Vehicles (UAVs)

UAVs have been used for various mission-critical tasks (e.g., search, rescue, military) and privacy can be among the parameters to determine their success. Koulianos et al. [71] address this concern by protecting their identities and three-dimensional locations. The motivation is to eliminate (i) malicious drone connection to the ground control station and (ii) incorrect data transmission. For drones with limited energy supplies, power consumption for proof generation has critical importance since their flight times should not be reduced to a degree to prevent their actual missions. For real-world scenarios (e.g., challenging weather conditions and terrains), network latency should be also taken into consideration along with the proof generation times. In the future, application of the proposed technique to the multi-drone (i.e., swarm) environment might be promising. Lastly,

some side-channel attacks should be analyzed with respect to UAVs transaction patterns and frequencies since they might unravel some meaningful information about their locations.

In the literature, there also exist other works to develop authentication protocols (e.g., based on digital identity verification, set membership proof) [45, 65, 81, 118, 123, 131] and generic access control policies (e.g., attribute-based access control, consent-based) [47, 82, 83, 90, 106] as well without providing their specific applications. For a detailed security analysis for set membership proof-based authentication, refer to the work [118]. The work [81] also provides analysis for different security guarantees such as unforgeability, anonymity and traceability. The work [97] compares the performance of zkSNARKs and zkSTARKs for generic authentication in blockchain and presents the list of development libraries with relative specifications.

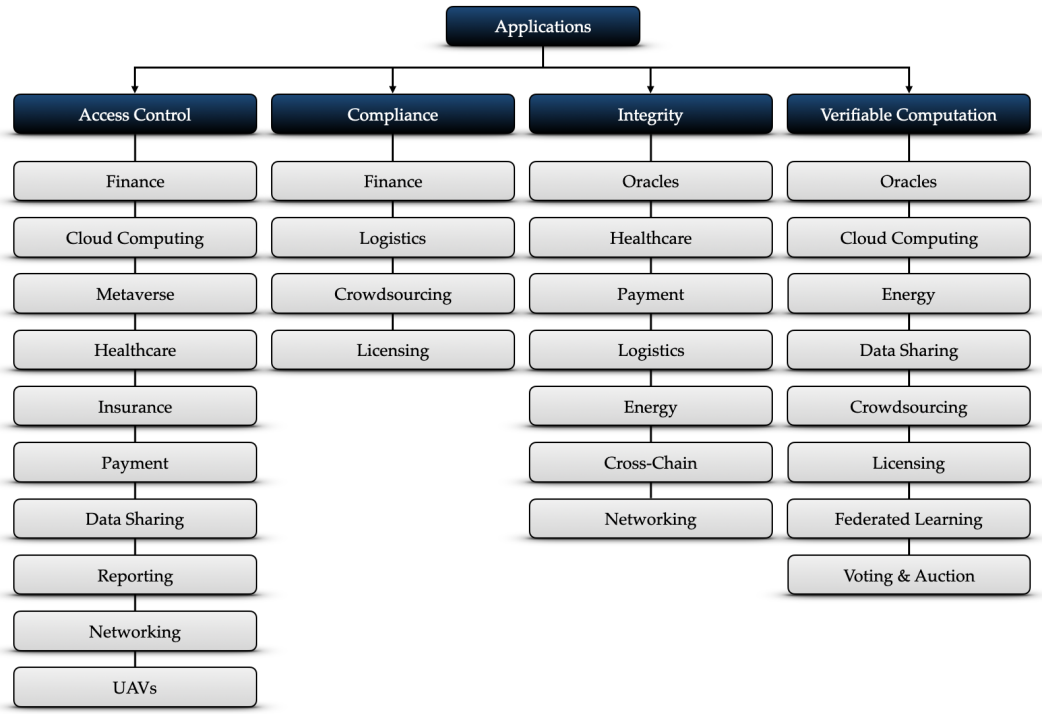


Fig. 5. Classification for Privacy-Preserving Applications of ZoKrates

5 Popular Performance Measures

This section identifies the popular performance measures for ZoKrates in the literature to answer the second research question (**RQ2**). The objectives of this section are to present a standardization and taxonomy over the measures; and to help researchers to select appropriate ones for their own applications. The measures in this survey are classified into five main categories as: (i) time-related, (ii) blockchain-related, (iii) storage-related, (iv) qualitative and (iii) problem-specific metrics. Refer to Fig. 6 for the complete classification.

The **time-related** performance measures are defined in the following way:

- *Code Compilation Time.* It is Parser and Flattener (see Fig. 1) that compiles the initial DSL program into a corresponding verifiable scheme (i.e., the structured list of variable

Table 2. Analysis and Comparison of Publications Using ZoKrates

Paper	Year	Task	Application	Blockchain	Formal	Open	Main Limitation
[60]	2021	Compliance	Finance	Ropsten/IPFS	○	×	Difficulty of implementing complex investment risk calculation
[103]	2024	Compliance	Finance	Consortium	○	✓	Centralization around auditee proving transaction compliance
[126]	2021	Access Control	Finance	Rinkeby	◐	×	Central trustworthy investigation agency for credit reports
[39]	2024	Vrf. Computation	Oracles	Goerli	○	×	No formal security analysis of proposed oracle system
[98]	2020	Integrity	Oracles	Ropsten	○	×	No formal security analysis of proposed oracle system
[24]	2020	Vrf. Computation	Cloud Computing	Ethereum	○	×	High gas consumption for small input size of 10
[79]	2019	Access Control	Cloud Computing	-	○	×	No empirical study targeting performance on blockchain
[66]	2023	Access Control	Metaverse	Ethereum	◐	×	Centralization around entity storing user credentials
[17]	2022	Integrity	Healthcare	Ethereum	●	×	Oracle servers as single point of failure
[124]	2024	Integrity	Healthcare	Ropsten/IPFS	●	×	No formal security and privacy analysis
[108]	2020	Access Control	Healthcare	Ethereum/IPFS	◐	×	Insufficient experiment for national framework
[80]	2022	Access Control	Healthcare	Ropsten	●	×	Inefficiently-designed arithmetic circuit
[28]	2023	Access Control	Healthcare	IPFS	◐	×	Too complex orchestration for real-world applications
[56]	2023	Access Control	Insurance	-	○	×	No formal security and privacy analysis
[49]	2025	Integrity	Payment	Ethereum	●	✓	High gas consumption for on-chain verifications
[76]	2023	Access Control	Payment	Ethereum	○	×	Very high gas cost for asset withdrawal
[75]	2025	Access Control	Payment	Ethereum	●	×	Long proof generation times (up to 30 s)
[14]	2021	Integrity	Payment	Ethereum	◐	×	Anonymity trade-off for efficient shallow Merkle tree
[48]	2020	Integrity	Payment	Ethereum	◐	✓	Lack of nation-wide scalability and batch proof verifications
[46]	2023	Integrity	Logistics	Ethereum	◐	✓	No formal security analysis
[116]	2024	Compliance	Logistics	Consortium	○	×	Open to sensor tampering
[25]	2020	Vrf. Computation	Energy	Ethereum	○	✓	Centralization around netting server aggregating measurements
[59]	2023	Integrity	Energy	Ethereum	●	×	Centralization around energy dispatch station
[127]	2021	Vrf. Computation	Energy	Ethereum	◐	✓	Extendability to more complex optimization algorithms
[53]	2024	Vrf. Computation	Data Sharing	Sepolia	●	✓	Not scalable communicational and computational overheads
[55]	2025	Vrf. Computation	Data Sharing	Ethereum	◐	✓	Orchestration complexity of dual hypercube networks
[85]	2025	Access Control	Data Sharing	Ethereum/IPFS	◐	✓	Generic approach with no domain-specific consideration
[52]	2025	Vrf. Computation	Crowdsourcing	Sepolia	●	✓	Too simple benchmark problems to evaluate performance
[69]	2021	Vrf. Computation	Crowdsourcing	Ethereum	○	✓	No formal security analysis
[54]	2025	Vrf. Computation	Crowdsourcing	Ethereum	○	✓	No consideration of multiple bartering solutions in graph
[72]	2024	Compliance	Crowdsourcing	Rinkeby/Goerli	●	✓	Adoption of more complex worker quality score calculations
[117]	2020	Integrity	Cross-Chain	Ethereum/Bitcoin	○	✓	Trade-off for resource consumption and cross-synchronization
[115]	2025	Integrity	Cross-Chain	Ethereum/Bitcoin	●	×	High off-chain latency for real-time scenarios
[95]	2024	Access Control	Reporting	Ethereum/Polygon	◐	×	Adoption of complex framework in real-world constructions
[91]	2025	Access Control	Reporting	Ethereum	◐	✓	Potential biases of the training dataset limiting generalization
[43]	2024	Access Control	Networking	Ethereum	○	×	No energy consideration for resource-constrained devices
[125]	2025	Access Control	Networking	Sepolia	◐	×	Centralization around issuers to issue the credentials
[20]	2025	Access Control	Networking	Ethereum/IPFS	●	×	Centralization around traffic authority
[130]	2024	Access Control	Networking	IPFS	◐	×	Centralization around traffic authority
[114]	2023	Access Control	Networking	Ropsten	◐	×	Orchestration overhead of multiple hierarchical chains
[122]	2021	Access Control	Networking	-	◐	×	Scalability and orchestration of ring signature for large sets
[42]	2021	Integrity	Networking	Ethereum	◐	×	No formal security and privacy analysis
[33]	2019	Access Control	Networking	Ethereum	◐	×	No empirical study with growing number of electric vehicles
[34]	2020	Access Control	Networking	Ropsten	◐	×	High contract deployment and authentication gas cost
[84]	2025	Vrf. Computation	Licensing	Ethereum	◐	×	Complex orchestration for real-world applications
[107]	2025	Compliance	Licensing	Ethereum/IPFS	◐	×	Centralization around regulatory organization to manage licenses
[61]	2021	Vrf. Computation	Fed. Learning	Ethereum	◐	✓	Lack of optimized hash function (i.e., Poseidon) for efficiency
[93]	2023	Vrf. Computation	Fed. Learning	Ethereum	○	×	Heavy computation of PSO for model aggregation
[44]	2022	Vrf. Computation	Fed. Learning	Ethereum	○	✓	Simple neural network model with single hidden layer
[40]	2023	Vrf. Computation	Fed. Learning	Goerli	◐	×	No formal security analysis of training strategies
[62]	2023	Vrf. Computation	Fed. Learning	Ethereum	◐	✓	Management of large proving keys up to 13 GB
[74]	2024	Vrf. Computation	Fed. Learning	Ethereum	◐	✓	Reliance on a central certificate authority
[27]	2024	Vrf. Computation	Fed. Learning	Ethereum/IPFS	◐	✓	No prevention mechanism against data poisoning
[7]	2024	Vrf. Computation	Fed. Learning	Ethereum	○	×	Execution time overhead of introducing privacy to models
[31]	2023	Vrf. Computation	Voting	Ethereum	●	✓	Security issues of malicious ballot boxes
[30]	2024	Vrf. Computation	Auction	Ethereum	●	✓	Lack of efficient comparison for encrypted bids
[71]	2024	Access Control	UAVs	Sepolia	◐	×	No consideration of multiple (swarm) drones

definitions and assertions). This metric measures machine-dependent time to successfully perform this compilation off-chain. Theoretically, compilation times are expected to increase with the complexity of the program to be compiled. To eliminate non-deterministic factors, multiple measurements are recommended where the exact number of runs might vary with system reliability precision. The following works use this metric in their experimental studies [17, 46, 60, 74, 79, 108, 118, 126, 130]. Some works also measure time to perform key generations during the initial trusted setup [45, 56, 59, 79, 82, 108, 118, 130].

- *Witness Generation Time.* Witness generator in ZoKrates is responsible for generating witness based on the compiled program and user-provided inputs, (see Fig. 1). This metric measures time to successfully generate a single witness. As the code compilation time, the witness generation time also increases regarding the program complexity. The following works are identified to use this metric [14, 17, 43, 45, 46, 56, 60, 69, 74–76, 103, 108, 114, 123, 125, 130].
- *Off-Chain Proof Generation Time. Libsnark* in ZoKrates is responsible for generating proof by using witness and proving key, (see Fig. 1). This metric measures time to successfully perform a single proof generation. As stated in the original ZoKrates work [26], the proof generation is the main bottleneck and is highly-dependent to the implementation of internal *libsnark* C++ library [73]. Majority works in the literature consider this metric in their empirical evaluations [4, 14, 17, 25, 29, 43, 45, 46, 49, 52–56, 59, 60, 68, 69, 72, 74–76, 80, 82, 98, 103, 108, 109, 114, 117, 118, 123, 125, 126, 130]. Note that ZoKrates supports proof generation only off-chain (on-chain computation would be too expensive), reporting on-chain proof generation is not applicable. The following works [60, 108, 130] also measure smart contract extraction time.
- *Off-Chain Proof Verification Time.* Contract Generator in ZoKrates automatically extracts contracts to verify proofs, (see Fig. 1 and Fig. 2). As described in Section 2.2, ZoKrates allows two different proof verifications as off-chain and on-chain. This metric specifically measures machine-dependent time to successfully perform a single off-chain proof verification on CLI. Unlike proof generation, proof verification does not vary with respect to program complexity and takes only negligible times (in the scale of milliseconds on modern computers). The following works are identified to use this metric [29, 43, 54, 59, 72, 75, 76, 80, 82, 114, 117, 118].

The **blockchain-related** performance measures are defined in the following way:

- *Proof-Verifying Contract Deployment Gas Cost.* ZoKrates-generated contracts must be deployed onto blockchain to be used across blockchain. Deployment is often among the most expensive operations since these contracts need to store verification keys and perform heavy mathematical computations to verify proofs. However, it requires only one-time gas cost to be covered to permanently store contract byte-codes on-chain. This metric measures the total amount of gas costs (in terms of gas units) to deploy a single contract. Unlike the time-related measures, the results are deterministic if no other condition is explicitly stated (e.g., a certain architectural property of underlying blockchain infrastructure). However, exact gas consumption might slightly vary (i) across distinct blockchain platforms, (ii) EVM versions and (iii) ZoKrates versions. It is also possible to express the results of this metric with other correlated metrics (e.g., gas costs in ETH or USD). Note that exchange rates between these currencies might fluctuate over time. In addition, gas consumption increases by the increasing number of public inputs because the contracts need to iterate over them. The following works are identified to use this metric [20, 24, 49, 53, 55, 58, 59, 108, 124].
- *On-Chain Proof Verification Gas Cost.* Following successful deployments of the proof-verifying contracts, they can start verifying proofs through public external calls to the

function *verifyTx*. This function itself is defined as *view* (i.e., read-only function without changing chain state) and does not actually incur any cost. However, it is often subjected to external calls from the state-changing functions in the other contracts, results in gas consumption. So, this metric measures gas consumption of such functions to verify a single proof on-chain. Degree of gas consumption changes by complexity of the effects committed to chain state. The following works are identified to use this metric [17, 20, 24, 25, 39, 43, 46, 53, 58, 60, 71, 74, 76, 98, 108, 117]. Practically, it is often recommended to follow the **checks-effects-interactions** pattern when developing secure contract functions (e.g., to prevent re-entrancy attack). Therefore, the proof-involved functions need to verify proofs during the *checks* phase so that they can later make decisions on applying effects (e.g., transfer assets if proof is correct).

The **storage-related** performance measures are defined in the following way:

- *Number of Constraints in Circuit.* Constraints refer to mathematical expressions to check validity of an assignment in an arithmetic circuit (e.g., *R1CS - Rank-1 Constraint System*). A complex DSL program yields a larger circuit with more constraints. Therefore, the number of constraints available in the arithmetic circuit is among key metrics to determine the actual complexity of the program written. This metric measures the number of such constraints, which is deterministic for the same program unless it is modified. ZoKrates reports this value after the compilation phase. The following works use this metric [4, 8, 14, 29–31, 49, 53, 54, 60, 61, 68, 96, 118, 125–127, 131].
- *Proving Key Size.* ZoKrates generates a public proving key during the initial trusted setup and uses it during off-chain proof generation. This metric measures the amount of memory to store a single proving key. Theoretically, the proving key size linearly grows up with the increasing number constraints, (in the scale of MB/GB). Modern computers are often well-equipped to tolerate that level of storage. Nevertheless, management and transmission of proving keys (as in the work [126]) might still require a careful consideration (e.g., slowdown while loading a webpage embedded with proving keys in gigabytes). The result of this metric is deterministic for the same program compiled. The following works use this metric [14, 43, 45, 46, 52–54, 62, 68, 74, 83, 103, 113, 118, 123, 125, 126].
- *Verification Key Size.* During the trusted setup, ZoKrates also generates a public verification key to be used during proof verification. This metric measures the amount of memory to store a single verification key. Length of a verification key is critical to determine the contract deployment gas cost because it is inserted into the proof-verifying contract during automatic extraction. Hence, a larger verification key would yield higher gas consumption. Fortunately, verification key size of ZoKrates is fixed and succinct. The list of works that use this metric are [43, 45, 46, 52–54, 74, 103, 113, 118, 123, 125, 126].
- *Proof Size.* This metric measures the amount of memory to store a proof. As verification key, a proof is processed on-chain after the function call to *verifyTx*. Luckily, proof size remains constant and succinct, regardless of the program complexity. Short proofs are beneficial during fast network transmission and efficient on-chain proof verification. The list of works that use this metric are [46, 52–54, 98, 103, 113, 118, 126]. In the literature, there are also works to measure witness size [14].

The **qualitative** performance measures are defined in the following way:

- *Formal Security and Privacy Analysis.* Ensuring security and privacy under a clearly-defined threat model and assumptions is among the fundamental performance requirements. There exist various formal methods to establish a theoretical basis in the literature. For instance, reduction proofs aim to show that attacking a protocol is at least as difficult as breaking

the underlying hard problem, (e.g., ZoKrates is secure if and only if zkSNARKs are secure). In addition, analysis of zero-knowledge properties (i.e., completeness, soundness and zero-knowledge) might be also beneficial to guarantee privacy. Specifically, the zero-knowledge property is critical to test if a protocol is fully privacy-preserving or it leaks meaningful information about the statement to be proven. Refer to the following works [17, 20, 30, 31, 49, 52, 72, 75, 115] for formal security and privacy analysis.

- *Problem-Specific Requirement Validation.* Other than regular privacy and security guarantees, some applications might impose additional properties to be satisfied. These properties include (but are not limited to) anonymity, scalability, consistency, maintainability, usability, flexibility, interoperability, transparency, auditability, fairness. Providing explicit discussions on the approach of validation (e.g., how to achieve fairness) and the degree of validation (e.g., how anonymous with varying transactions revealed [79]) can help more rigorous comparisons across existing works. This article avoids defining these requirements explicitly since their precise definitions might vary across problems. For such discussions, refer to the following works [20, 30, 55, 59, 67, 72, 78–81, 83, 122, 126].
- *Attack Surface Analysis.* Attack surface describes the set of entry points that can be internally or externally exploited. Internal attacks refer to the adversarial actions system participants follow by leveraging their legitimate privileges. For instance, a subset of nodes might intentionally and collaboratively work to stall the progress of a protocol (e.g. awaiting the proof submissions for a long time) in collusion attacks. Or, a node individually might attempt to submit the same proof multiple times to invoke a specific code block in contract (e.g., to transfer assets to own account) in replay attacks. On the other hand, external attacks refer to the adversarial actions of participants outside the system. For instance, a node might attempt to crack the hash values publicly stored in contract in enumeration attacks. To understand the settings a protocol can operate reliably under well-defined threat models and assumptions, rigorous attack surface analysis is critical. For such analysis, refer to the following works [2, 33, 34, 49, 55, 75, 84, 107, 114, 122, 131].

The **application-specific** measures refer to metrics that do not evaluate performance of ZoKrates directly but are still under its influence. For instance, prediction accuracy is among the key metrics to assess model quality in federated learning. However, limitations ZoKrates imposes (e.g., constraints on the maximum number of model parameters supported) might lead the model to perform poorly. The following works [27, 44, 74, 91] use accuracy while the work [91] use other learning-based metrics as well including precision, recall, F-1 Score, confusion matrix and loss curve. The other example is service latency that refers to the amount of end-to-end delay to respond to a request. ZoKrates might implicitly contribute to service latency with long proof generation times. The following works use service latency [2, 14, 17, 20, 78, 106]. There exist wide range of such application-specific measures as well including packet loss rate [20], energy consumption (e.g., in watts) [14, 71], throughput (e.g., in transactions/operations per second) [14, 17, 28, 48, 78, 84, 114, 125], degree of anonymity [79, 95] and number of compilation failures and line coverage [120]. The explanation of these metrics are not within the scope of this survey since they need to be precisely described within their own settings.

6 From Challenges and Open Issues to Future Research Directions

This section identifies the widely-encountered challenges of ZoKrates and later presents the potential future directions to answer the third and fourth research questions (**RQ3**, **RQ4**). The objectives of this section are (i) to present a taxonomy over challenges after their proper identification and (ii) accelerate developments on certain research directions. The challenges are classified into five

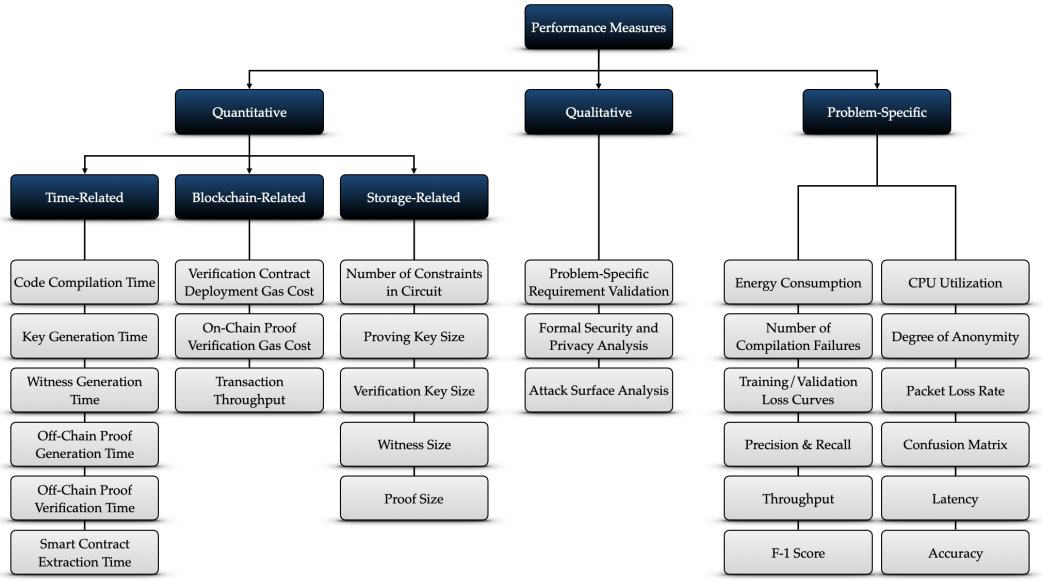


Fig. 6. Classification of Performance Measures

main groups as: (i) computational efficiency, (ii) scalability, (iii) developer & user experience, (iv) cryptographic security and (v) integration & interoperability. Refer to Fig. 7 for the the classification.

The **computational efficiency** challenges are described in the following way:

- *Off-Chain Proof Generation Cost.* Proof generation is the main performance bottleneck in ZoKrates as it involves computation-intensive instructions. The works already suffer from this overhead with varying form (as time, energy and operational) and degree [25, 31, 52, 53, 69, 71, 72, 75, 109, 115, 123]. Time cost refers to the amount of time to complete a single proof generation, which might take up to several minutes. This might hinder application responsiveness, especially when multiple parties need to generate proofs [55]. Energy cost refers to amount of energy to complete a generation. Although this cost is tolerable in modern computers, it still needs attention for IoT devices with limited capabilities. Finally, operational cost refers to upgrading such devices to enable proof generation. There are different solutions (e.g., circuit simplification, parallelism, accelerated hardware) to mitigate these costs [4].
- *On-Chain Proof Verification Cost.* On-chain costs are critical since they require direct payments to deploy contracts at first and verify proofs on these contracts later. Several works already highlight this overhead [24, 27, 49, 76]. Contract deployment happens one-time and authorities often cover the cost. However, participants have to cover the verification cost every time they submit their proofs on-chain. The resulting transactions also need to be validated across the network and appended to the fixed-size blocks, which degrades system throughput and scalability. In the literature, the work [58] presents a list of guidelines (e.g., input initialization, function designs) to reduce the overall gas consumption. The work [113] compares the proof verification costs of ZoKrates and Circom libraries on three different computational problems (e.g., knowledge of Hamiltonian cycle in graph), which reports that Circom is cheaper [10].

- *Circuit and Key Memory Consumption.* As noted earlier, size of arithmetic circuits and proving keys increases with program complexity [127]. For large-scale applications, proving keys can reach up to the several gigabytes [54]. This poses significant challenges for efficient key management including fast network transmission or key synchronization on distributed platform [74]. The issue is even more amplified if several proving keys are available (e.g., for depositing and later withdrawing assets [49]). Large proving keys might also slow down the application (e.g., loading from disk to memory) or drain more energy from resource-constrained devices (e.g., limited battery of IoT devices). Therefore, further optimization (e.g., key size reduction, decentralized storage integration) might be a promising research avenue [4, 74]. Luckily, verification keys in ZoKrates are short and fixed-size.
- *Lack of Proof Parallelism.* Proof parallelism refers to two distinct techniques: (i) generating independent proofs on separate resources and (ii) parallelizing what can be parallelizable in a single proof. For the first technique, orchestration is a critical challenge (e.g., distributing proofs over resources and collecting results). The work [7] suggests to leverage Nvidia CUDA [94] for proof parallelism and acceleration. The work [23] servicifies ZoKrates to process and reply multiple proof requests at the same time. On the other hand, the second technique requires cryptographic expertise to modify internal designs of proof systems. Overall, incorporating parallelism surely benefits to the scalability and adoption of ZoKrates.
- *Lack of Recursive and Batch Proof Verification.* Several works in the literature already suggest more efficient proof verification techniques (e.g., recursive, batch verification) [27, 48, 71, 74]. **Recursive verification** refers to verifying a single high-level proof that verifies multiple low-level proofs in itself [22]. On the other hand, **batch verification** refers to verifying several separate proofs simultaneously through a single verification scheme [16]. Those techniques are beneficial to improve system scalability and cost optimization reducing on-chain verification cost [74]. However, ZoKrates currently does not support these techniques.
- *Inefficient Hash Functions.* Choosing the proper hash functions to be used in zero-knowledge proof is among the decisions to affect computational overheads [81]. ZoKrates natively supports several standardized hash functions including SHA-256, MiMC [5], Pedersen [100] and Poseidon [37]. Not all these hash functions are ZK-friendly and efficiently compatible to zkSNARKs, (e.g., SHA-256 [17, 49, 55]). Here, special attention should be paid to **Poseidon** [61]. According to the empirical study in the work [37], Poseidon-128 needs 7,290 constraints to prove a leaf knowledge in Merkle tree of 2^{30} elements while Pedersen and SHA-256 need 41,400 and 826,020 constraints, respectively. The reduction in the constraint size improves the size of proving key and off-chain proof generation time as well, implying that more complex applications can be now efficiently supported. The work [37] also provides security guarantees of this hash function by analyzing attack vectors including statistical (e.g., linear distinguishing) and algebraic (e.g., interpolation). In addition, the work [4] distills the critical constraints contributing the circuit security and removes the rest, which further improves the performance of Poseidon by %20 in time and memory. Another work [68] has more comprehensive approach by comparing five different hash functions (e.g., SHA256, SHA3, Poseidon, MiMC, Blake2), where Poseidon shows the best performance again.
- *Further Optimization.* Several works present their proof-of-concept implementations and leave further optimization later [60, 103, 126]. Optimization in this setting might applied to various levels as protocol, contract and platform. For protocol-level, the work [4] performs constraint simplification for circuit reduction. This work [39] provides (i) horizontal batching that splits single dataset (with no data-dependency) into several subsets (e.g., federated learning with local datasets) and (ii) vertical batching that splits program into

several sub-programs. The work [103] follows this vertical batching approach to design modular transaction auditing with three independent sub-programs as Merkle proof, balances proof and whitelist proof. This work [129] presents (i) **precomputation** to perform proof generation during resource idle times, (ii) **asynchronous verification** to asynchronously verify proofs without awaiting packet transmission and (iii) **batching** to reuse some of previous computations for batch verification. For the contract-level, the work [58] makes several modifications (e.g., compiler selection, number of functions, function names, variable initialization) over ZoKrates contract for more efficient on-chain performance. For the platform level, the work [7] suggests parallel computing platforms (e.g., CUDA [94]) to be used.

The **scalability** challenges are described in the following way:

- *Growing Number of Users.* Supporting growing number of users is among the key challenges [48, 108]. Especially, decentralized applications differ from traditional centralized applications in achieving scalability. For instance, cloud applications might follow horizontal scaling (e.g., adding more resources to a cluster) or vertical scaling (e.g., upgrading specifications of resources such as memory). On the other hand, DApps with no control on the heterogeneous resources across network, might need more versatile techniques (e.g., contract-level optimization, better task distribution to the existing parties, integration of decentralized storage networks). Quantifying scalability for comparison is also possible in terms of computational, communicational or storage overheads [55].
- *Multi-Party Proof Computation.* Multi-party proof computation refers to distributing responsibilities over multiple parties across a network to achieve a common proof-involving task (e.g., crowdsourcing [52, 55, 69, 72], federated learning [27, 44, 74, 93]). Fairness is among the critical issues in this setting where transaction costs principally should not vary across different parties (e.g., a conditional statement invoking a certain code block for specific cases). This might lead to unfairness and consequently poor incentivization among participants. The second issue is the coordination of these participants with varying computational (while generating proofs) and communicational abilities, (while interacting with chain). The third issue is liveness since some application designs might need all participants to complete a certain phase before moving to the next phase [55]. Adversarial participants intentionally exploit this to slow down or stall the progress of application for certain benefits, (e.g. for more profit during that prolonged period).

The **developer & user experience** challenges are described in the following way:

- *Limited Domain-Specific Language.* Introducing a high-level DSL is among the most critical contributions of ZoKrates. This language is easy to learn and covers both most fundamental and advanced concepts (e.g. conditional statements, for loops, hashing libraries). However, it suffers from the lack of certain instructions (e.g., unbounded loops, floating-point arithmetic and negative numbers). Especially, floating-points are essentials for most of the neural network applications to represent model weights and biases. These limitations might be temporarily overcome through certain tricks, such as expressing negative numbers through offsetting and floating numbers through integers; or packing multiple data elements into a single variable through bit manipulation to minimize variable count. However, their permanent resolution is important to develop more reliable and complex applications (e.g., evolutionary computation [52], federated learning [40, 44]).
- *Lack of Integrated Development Environment.* Privacy-preserving application development on ZoKrates requires a series of complex instructions (e.g., performing trusted setup, compiling program, generating witness, generating proof) to be precisely carried out. This is

especially time-consuming, tedious and error-prone for large-scale applications. However, ZoKrates does not have its own development environment where developers currently rely on CLI to manually execute instructions. In this regard, a ZoKrates-specific end-to-end integrated development environment bringing all essentials tools (e.g., for code editing, compiling, debugging, testing, on-chain deploying, version controlling, logging and monitoring) together is definitely beneficial. Such an environment is promising for error prevention, powerful debugging, comprehensive testing and overall proof management with more well-grounded and secure design decisions. In addition, it might be further integrated into LLMs for automatic code assistance and generation [51]. For inspiration, the work [67] develops profiler and reporter tools to compare different proof systems. The work [120] proposes a testing tool to find compilation failures and logic errors of zero-knowledge compilers using metamorphic relations. These tools might be considered to be the parts of this development environment.

- *Lack of Large-Language Models.* Although the recent high-level languages (e.g., ZoKrates DSL itself) benefits proof development, they still require a certain degree of domain and programming expertise. In the literature, the work [111] defines (i) a new language to directly write privacy-preserving smart contracts and (ii) their automatic transformations of the equivalent Solidity contracts. But, it relies on quite strict language rules. Another work [21] successfully employs LLMs for Solidity code generation. In this respect, integration of LLMs for automatic ZoKrates code generation as well might be a promising solution. This integration might be in different forms with respective strengths and weaknesses: (i) using pre-trained models of the external services (e.g. those accessible from OpenAI APIs) or (ii) fine-tuning a specifically-tailored model to ZoKrates. For instance, the second solution is likely to result in better performance while it requires annotated datasets of diverse ZoKrates programs for model training. In another perspective, this article also anticipates that the current trend of LLMs will shift from public to privacy-preserving paradigms in the future as privacy concerns of individuals and companies continue to grow over their data, (e.g., zkGPT [105]). However, implementing privacy-preserving LLMs is non-trivial for various reasons, (e.g., very large model size with billions of parameters, heavy training computations, long proof generation times with intolerable latencies). Therefore, **ZK-friendly large-language models** might be another prospect to explore in the future without still compromising privacy and security guarantees.
- *Client-Side Proof Generation.* Real-world applications often prefer user interactions to be simple and intuitive. In this respect, expecting end-users to manually perform proof generations sounds not reasonable. Abstracting this complexity through user interface elements and actions is both performed [31, 48, 49, 55, 95] and suggested in the literature [90]. There exist different mitigation options for this issue. The first option is to bundle ZoKrates modules into a web application (i.e., webpacking). But, it still might require further optimization (e.g., lazy-loading large files, caching keys, performing network optimization, showing progress bar). The second option is to offload proof generation to the external APIs (i.e., ZoKrates API [23]). Still, there exist certain caveats: (i) privacy violation if private inputs need to be submitted to external services via public network where on-premise solutions might be followed, (ii) transmission latency of the network and (iii) orchestration and validation of HTTP requests and responses.

The **cryptographic security** challenges are described in the following way:

- *Initial Trusted Setup.* ZoKrates relies on an initial trusted setup ceremony to generate proving and verification keys. Party performing this setup is expected to destroy the secret data (i.e.,

toxic waste) of that ceremony afterwards. In any compromise, an adversarial party might generate fake proofs to effectively break the soundness of proof system. There exist works that complains about this issue as well [30]. ZoKrates already attempts to mitigate the issue by performing the setup with multi-party computation where one honest party involving the ceremony is sufficient to keep the resulting system secure. Unfortunately, this solution is complex and requires strict orchestration among parties. Furthermore, this setup is not universal and requires repetition for every new program, which degrades scalability and usability. Therefore, transitioning to a more universal setup (e.g., PlonK [35]) or no-setup at all (e.g., Bulletproofs [15]) might be taken into consideration.

- *Resistance to Quantum Attacks.* ZoKrates relies on zkSNARKs as the underlying proof protocol which is based on elliptic curve cryptography and discrete logarithm. Specifically, the problem of discrete logarithm is known to be computationally hard to solve for modern computers, but still might be vulnerable to powerful quantum computers [64]. This may pose a long-term security threat to the integrity and security of ZoKrates. In the literature, there exist examples of **quantum-resistant zero-knowledge proof schemes** (e.g., for data authentication with zkSTARKs [19]). Similarly, reinforcing zkSNARKs and ZoKrates against potential quantum attacks is significant in the future (e.g., for quantum-safe confidential transactions [86]).
- *Formal Security Analysis and Attacks.* This article identifies the works that consider formal security and privacy analysis in Table 2. Contrarily, there are also works that omit this analysis [31, 39, 54, 56, 69, 71, 76, 124]. As a cryptography framework, ZoKrates might be subject to various security risks. First, attack vectors might target the theoretical foundation and cryptographic assumptions of underlying zkSNARKs system, (e.g., compromising toxic waste during trusted setup ceremony). Second, ZoKrates might include its own faulty implementations (e.g., incorrect memory management, faulty circuit generation). Third, adversarial parties might use side-channel attacks based on behaviors of ZoKrates under varying scenario (e.g., timing attacks, memory access patterns) [27]. This article recommends formal security analysis to be taken into consideration for the upcoming applications.

The **integration & interoperability** challenges are described in the following way:

- *Lack of Modern Proof Protocols.* As reviewed in Section 2, there exist different modern proof protocols such as zkSNARKs [12], zkSTARKs [11] and Bulletproofs [15] with different relative strengths and weaknesses. ZoKrates was initially and successfully developed for zkSNARKs. However, this design choice limits developers to leverage the benefits of other proof systems (e.g., no trusted setup property of Bulletproofs). Instead, a modular and pluggable integration of multiple protocols to ZoKrates could be a promising future research direction. Such a design would enable developers to select the most appropriate protocol(s) for their own application requirements. For inspiration, the work [96] proposes a shared compiler scheme to design several novel cryptographic compilers. It supports the languages of C, Circom and ZoKrates, by still allowing further extensions possible. Its empirical study shows the performance of the new framework reinforced with some optimization techniques (e.g., common sub-expression elimination) outperforms the original ZoKrates compiler.
- *Lack of Non-EVM Compatible Blockchains.* ZoKrates contracts are designed to be executed on EVM-compatible blockchains (e.g., Ethereum, Avalanche) [66] while no native support for other virtual machines (e.g., SVM - Solana Virtual Machine). This limitation prevents (i) adoption of ZoKrates for privacy-preserving applications targeting other virtual machines and (ii) construction of privacy-preserving cross-chain applications where at least one chain is not EVM-compatible [75, 115, 117]. Therefore, extending ZoKrates in this dimension is a

promising but at the same time challenging research direction. For instance, verifier logic in ZoKrates contracts need to be re-implemented by considering a novel set of libraries and infrastructure-specific issues (e.g., maximum transaction gas limit).

- *Limited Real-World Applications.* Inclusion of diverse and interesting real-world applications is already suggested in the literature [90]. So far, ZoKrates has been applied to numerous real-world problems [44, 46, 52, 54, 66, 69, 71, 91, 93, 116]. This article observes the increasing trend in the number of relevant works (from 4 papers in 2019 to 24 papers in 2024, as seen in Fig. 4a) and anticipates the numbers to surge up in the future as privacy concerns of individuals and companies deepen. Among these applications, federated learning excels itself out, which could be a fruitful prospect to explore further in the future.
- *Cross-Chain Interoperability.* There exist several techniques to introduce cross-chain interoperability including trusted relays, side-chains and hash-time lock contracts [9]. Privacy-preserving version of cross-chain interoperability through zero-knowledge proof is significantly challenging since heterogeneous chains might have (i) different execution models (i.e., virtual machines) for on-chain proof verification, (ii) different consensus models with varying immutability guarantees, (iii) different cryptographic libraries natively supported. Therefore, the number of works specifically addressing ZoKrates for this context is quite limited [115, 117]. Nevertheless, exploring capabilities of ZoKrates in this setting might be still promising [75, 84].
- *Translation of Theory into Practice.* Theoretical studies for zero-knowledge proof often focus on formal guarantees, asymptotic efficiency or provable security under well-defined mathematical ground and assumptions. However, practical studies often need to consider additional and unique real-world challenges (e.g. user experience, technological limitations, compliance and compatibility). Hence, translations of intangible theoretical advancements to practical applications on tangible machines remains as a significant challenge. Additionally, these translations might require comprehensive testing and continuous maintenance afterwards to remain robust against potential attack vectors.
- *Weak Developer Community.* Zero-knowledge proof is a complex cryptographic concept to understand, especially for novice researchers with limited domain and programming expertise. Several approaches might be adopted to mitigate this issue. First and foremost, this article encourages researchers to publicly share their source-codes associated with their publications. The following works are identified to share their source codes [24, 25, 27, 30, 31, 44, 46, 48, 49, 52–55, 61, 62, 69, 72, 74, 85, 91, 103, 127], only 22 out of the total 87 works. In the future, this approach might also help building annotated datasets to train LLMs specifically for ZoKrates. In addition, the number of community-driven events (e.g., workshops, hackathons, developer meetups), online resources (e.g., forums), high-quality tutorials might further facilitate more efficient knowledge sharing among researchers.

7 Do You Need ZoKrates?

This article proposes a new high-level and ZoKrates-tailored decision-making flowchart by asking a set of questions. The flowchart shown in Fig. 8 asks five sets of questions: (i) the objective, (ii) cryptographic, (iii) language, (iv) platform and (v) operational. The **objective** question initially determines whether the use-case genuinely involve a computation to privatize. The **cryptographic** requirements identify if ZoKrates can meet the cryptographic constraints of application in focus, (e.g., tolerating trusted setup). The **language** requirements underline the limitations of ZoKrates DSL. For instance, developing an application requiring negative numbers is not currently feasible in ZoKrates. The **environment** requirements focus on the target blockchain to execute ZoKrates

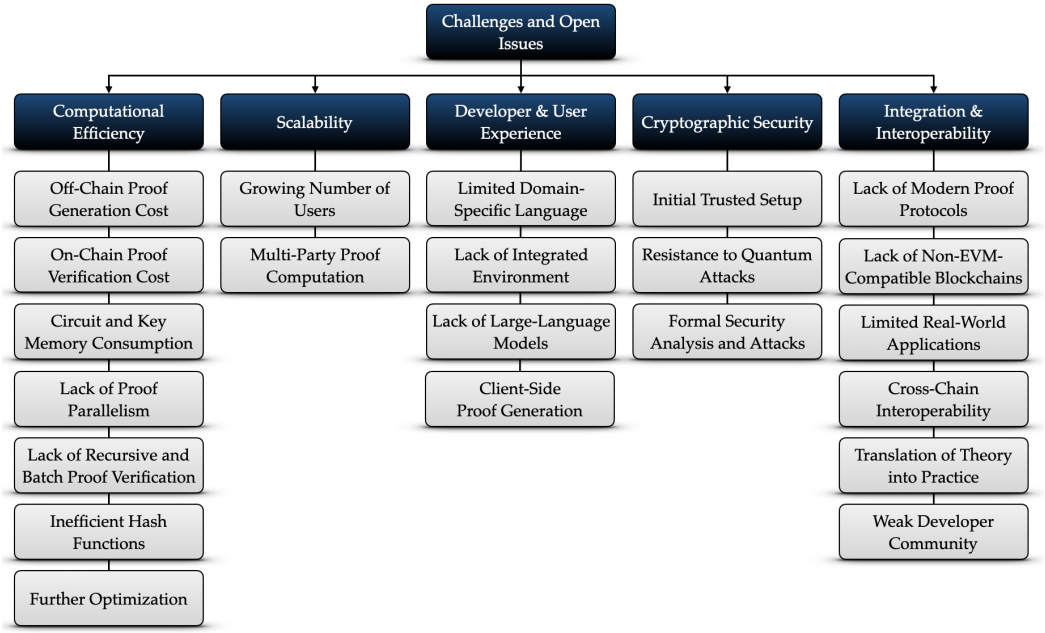


Fig. 7. Classification of Challenges and Open Issues

contracts (e.g., EVM-compatibility). Finally, the **non-functional** requirements evaluate practical needs to meet target performance. This flowchart is beneficial to find if it is worthwhile to invest into ZoKrates before spending substantial time and money.

8 Conclusion

For the first time in the literature, this article conducts a systematic literature review to explore privacy-preserving applications in blockchain from the perspective of ZoKrates. For this review, it first collects a corpus of 347 documents from the literature and carefully selects 87 different ones for further analysis by excluding gray literature. The primary objective of this article is to conduct this analysis over the resulting set of documents with the attempt to answer four critical research questions about (i) the privacy-preserving applications of ZoKrates in blockchain, (ii) the popular performance measures, (iii) the current challenges and open issues and finally (iv) the future research directions. The performance measures are classified into five categories as (i) time-related, (ii) blockchain-related, (i) storage-related, (i) qualitative, (i) problem-specific. Similarly, the challenges are classified into five categories as: (i) computational efficiency, (ii) scalability, (iii) developer & user experience, (iv) cryptographic security and (v) integration & interoperability. From the perspective of this article, ZoKrates will survive as a promising framework to be adopted in the upcoming decentralized privacy-preserving applications if its major open issues are well-addressed. In the future, (i) a comprehensive comparison of ZoKrates with the other proof frameworks to better highlight its relative strengths and weaknesses; or (ii) a more delicate investigation of zero-knowledge proof on a specific field (e.g., federated learning) might be useful as well.

References

[1] Abdelzahir Abdelmaboud, Abdelmuttlib Ibrahim Abdalla Ahmed, Mohammed Abaker, Taiseer Abdalla Elfadil Eisa, Hashim Albasheer, Sara Abdelwahab Ghorashi, and Faten Khalid Karim. 2022. Blockchain for IoT applications:

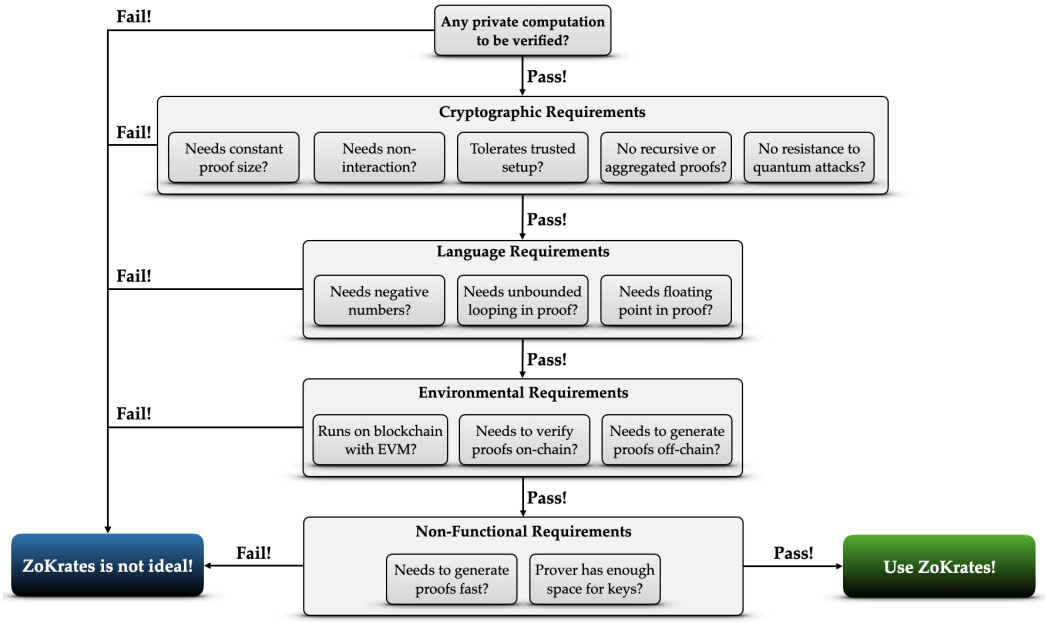


Fig. 8. High-Level Decision-Making Diagram for ZoKrates

taxonomy, platforms, recent advances, challenges and future research directions. *Electronics* 11, 4 (2022), 630.

[2] Amal Abid, Saoussen Cheikhrouhou, Slim Kallel, and Mohamed Jmaiel. 2025. A privacy-preserving traceability system for self-sovereign identity-based inter-organizational business processes. *Computer Standards & Interfaces* 92 (2025), 103930.

[3] Amal Abid, Slim Kallel, and Mohamed Jmaiel. 2024. TraSSI: A Confidential Traceability System for Self-sovereign Identity-Based Processes. In *International Conference on Service-Oriented Computing*. Springer, 241–246.

[4] Elvira Albert, Marta Bellés-Muñoz, Miguel Isabel, Clara Rodríguez-Núñez, and Albert Rubio. 2022. Distilling constraints in zero-knowledge protocols. In *International Conference on Computer Aided Verification*. Springer, 430–443.

[5] Martin Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. 2016. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 191–219.

[6] J Andrew, Deva Priya Isravel, K Martin Sagayam, Bharat Bhushan, Yuichi Sei, and Jennifer Eunice. 2023. Blockchain for healthcare systems: Architecture, security challenges, trends and future directions. *Journal of Network and Computer Applications* 215 (2023), 103633.

[7] Mohammad Bilal Aziz, Ali Shah Naushad, Maryam Siddiqui, and Jawwad Ahmed Shamsi. 2024. ZkVML: Zero-Knowledge Verifiable Machine Learning. In *International Conference on Asia Pacific Advanced Network*. Springer, 220–239.

[8] Rezak Aziz, Youakim Badr, and Samia Bouzefrane. 2025. Enhancing Trust in Central Differential Privacy Using zk-SNARKs and Cryptographic Hashes. In *International Conference on Advanced Information Networking and Applications*. Springer, 163–176.

[9] Rafael Belchior, André Vasconcelos, Sérgio Guerreiro, and Miguel Correia. 2021. A survey on blockchain interoperability: Past, present, and future trends. *Acm Computing Surveys (CSUR)* 54, 8 (2021), 1–41.

[10] Marta Bellés-Muñoz, Miguel Isabel, Jose Luis Muñoz-Tapia, Albert Rubio, and Jordi Baylina. 2022. Circom: A circuit description language for building zero-knowledge applications. *IEEE Transactions on Dependable and Secure Computing* 20, 6 (2022), 4733–4751.

[11] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. 2018. Scalable, transparent, and post-quantum secure computational integrity. *Cryptology ePrint Archive* (2018).

[12] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. 2014. Succinct {Non-Interactive} zero knowledge for a von neumann architecture. In *23rd USENIX Security Symposium (USENIX Security 14)*. 781–796.

[13] Muhammad Nasir Mumtaz Bhutta, Amir A Khwaja, Adnan Nadeem, Hafiz Farooq Ahmad, Muhammad Khurram Khan, Moataz A Hanif, Houbing Song, Majed Alshamari, and Yue Cao. 2021. A survey on blockchain technology: Evolution, architecture and security. *Ieee Access* 9 (2021), 61048–61073.

[14] EunSeong Boo, Joongheon Kim, and JeongGil Ko. 2021. LiteZKP: Lightening zero-knowledge proof-based blockchains for IoT and edge platforms. *IEEE Systems Journal* 16, 1 (2021), 112–123.

[15] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. 2018. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE symposium on security and privacy (SP)*. IEEE, 315–334.

[16] Matteo Campanelli, Dario Fiore, Semin Han, Jihye Kim, Dimitris Kolonelos, and Hyunok Oh. 2022. Succinct zero-knowledge batch proofs for set accumulators. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 455–469.

[17] Sheng Cao, Qian Zhang, Dongdong Wang, Peng Xiangli, and Xiaosong Zhang. 2022. Hybrid smart contracts for privacy-preserving-aware insurance compensation. In *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 1533–1538.

[18] Vincent Charles, Ali Emrouznejad, and Tatiana Gherman. 2023. A critical analysis of the integration of blockchain and artificial intelligence for supply chain. *Annals of Operations Research* 327, 1 (2023), 7–47.

[19] Usama Habib Chaudhry, Razi Arshad, Ayesha Khalid, Indranil Ghosh Ray, and Mehdi Hussain. 2025. zk-DASTARK: A quantum-resistant, data authentication and zero-knowledge proof scheme for protecting data feed to smart contracts. *Computers and Electrical Engineering* 123 (2025), 110089.

[20] Xingxing Chen, Xiaohong Zhang, Shaojiang Zhong, and Shuling Liu. 2025. Anonymous authentication based on blockchain and zero-knowledge proof for vehicular ad hoc networks. *The Journal of Supercomputing* 81, 15 (2025), 1416.

[21] Gabriele De Vito, Damiano D'Amici, Fabiano Izzo, Filomena Ferrucci, and Dario Di Nucci. 2025. LLM-Based Generation of Solidity Smart Contracts from System Requirements in Natural Language: The AstraKode Case. In *2025 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 170–180.

[22] Sai Deng and Bo Du. 2023. zkTree: A Zero-Knowledge Recursion Tree with ZKP Membership Proofs. *Cryptology ePrint Archive* (2023).

[23] Alvaro Alonso Domenech, Jonathan Heiss, and Stefan Tai. 2024. Servicifying zk-SNARKs Execution for Verifiable Off-chain Computations. In *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 670–671.

[24] Mallikarjun Reddy Dorsala, VN Sastry, and Sudhakar Chapram. 2020. Fair payments for verifiable cloud services using smart contracts. *Computers & Security* 90 (2020), 101712.

[25] Jacob Eberhardt, Marco Peise, Dong-Ha Kim, and Stefan Tai. 2020. Privacy-preserving netting in local energy grids. In *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 1–9.

[26] Jacob Eberhardt and Stefan Tai. 2018. Zokrates-scalable privacy-preserving off-chain computations. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 1084–1091.

[27] Elmira Ebrahimi, Michael Sober, Anh-Tu Hoang, Can Umut Ileri, William Sanders, and Stefan Schulte. 2024. Blockchain-based federated learning utilizing zero-knowledge proofs for verifiable training and aggregation. In *2024 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 54–63.

[28] Bhaskara Santhosh Egala, Ashok Kumar Pradhan, and Shubham Gupta. 2023. Block-privacy: Privacy preserving smart healthcare framework: Leveraging blockchain and functional encryption. In *IFIP International Internet of Things Conference*. Springer, 114–132.

[29] Thorsten Eisenhofer, Doreen Riepel, Varun Chandrasekaran, Esha Ghosh, Olga Ohrimenko, and Nicolas Papernot. 2025. Verifiable and provably secure machine unlearning. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 479–496.

[30] Ashkan Emami, Ghazaleh Keshavarz Kalhori, Sheyda Mirzakhani, and Mohammad Ali Akhaee. 2024. A blockchain-based privacy-preserving anti-collusion data auction mechanism with an off-chain approach. *The Journal of Supercomputing* 80, 6 (2024), 7507–7556.

[31] Ashkan Emami, Habib Yajam, Mohammad Ali Akhaee, and Rahim Asghari. 2023. A scalable decentralized privacy-preserving e-voting system based on zero-knowledge off-chain computations. *Journal of Information Security and Applications* 79 (2023), 103645.

[32] Jens Ernstberger, Stefanos Chaliasos, Liyi Zhou, Philipp Jovanovic, and Arthur Gervais. 2024. Do you need a zero knowledge proof? *Cryptology ePrint Archive* (2024).

[33] David Gabay, Kemal Akkaya, and Mumin Cebe. 2019. A privacy framework for charging connected electric vehicles using blockchain and zero knowledge proofs. In *2019 IEEE 44th LCN symposium on emerging topics in networking (LCN Symposium)*. IEEE, 66–73.

[34] David Gabay, Kemal Akkaya, and Mumin Cebe. 2020. Privacy-preserving authentication scheme for connected electric vehicles using blockchain and zero knowledge proofs. *IEEE Transactions on Vehicular Technology* 69, 6 (2020), 5760–5772.

[35] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. 2019. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *Cryptology ePrint Archive* (2019).

[36] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. 1985. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, STOC '85*. 291–304.

[37] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. 2021. Poseidon: A new hash function for {Zero-Knowledge} proof systems. In *30th USENIX Security Symposium (USENIX Security 21)*. 519–535.

[38] Jens Groth. 2016. On the size of pairing-based non-interactive arguments. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 305–326.

[39] Binbin Gu and Faisal Nawab. 2024. zk-oracle: Trusted off-chain compute and storage for decentralized applications. *Distributed and Parallel Databases* 42, 4 (2024), 525–548.

[40] Binbin Gu, Abhishek Singh, Yinan Zhou, Juncheng Fang, and Faisal Nawab. 2023. Ml on chain: the case and taxonomy of machine learning on blockchain. In *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 1–18.

[41] Zhikang Guo, Heng Pan, Ang He, Yueyue Dai, Xiaoyan Huang, Xueming Si, Chau Yuen, and Yan Zhang. 2025. Trusted Execution Environments for Blockchain: Towards Robust, Private, and Scalable Distributed Ledgers. *IEEE Internet of Things Journal* (2025).

[42] Jonathan Heiss, Anselm Busse, and Stefan Tai. 2021. Trustworthy pre-processing of sensor data in data on-chaining workflows for blockchain-based IoT applications. In *International Conference on Service-Oriented Computing*. Springer, 133–149.

[43] Jonathan Heiss, Fernando Castillo, and Xinxin Fan. 2024. Towards credential-based device registration in dapps for depins with zkps. In *2024 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 583–590.

[44] Jonathan Heiss, Elias Grünewald, Stefan Tai, Nikolas Haimerl, and Stefan Schulte. 2022. Advancing blockchain-based federated learning through verifiable off-chain computations. In *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 194–201.

[45] Jonathan Heiss, Robert Muth, Frank Pallas, and Stefan Tai. 2022. Non-disclosing credential on-chaining for blockchain-based decentralized applications. In *International Conference on Service-Oriented Computing*. Springer, 351–368.

[46] Jonathan Heiss, Tahir Oegel, Mehran Shakeri, and Stefan Tai. 2023. Verifiable carbon accounting in supply chains. *IEEE Transactions on Services Computing* 17, 4 (2023), 1861–1874.

[47] Jonathan Heiss, Max-R Ulbricht, and Jacob Eberhardt. 2020. Put your money where your mouth is–towards blockchain-based consent violation detection. In *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 1–9.

[48] Lasse Herskind, Alberto Giaretta, Michele De Donno, and Nicola Dragoni. 2020. BitFlow: Enabling real-time cash-flow evaluations through blockchain. *Concurrency and Computation: Practice and Experience* 32, 12 (2020), e5333.

[49] Goshgar Ismayilov and Can Özturan. 2025. PTTS: Zero-knowledge proof-based private token transfer system on Ethereum blockchain and its network flow based balance range privacy attack analysis. *Journal of Network and Computer Applications* 233 (2025), 104045.

[50] Goshgar Ismayilov and Haluk Rahmi Topcuoglu. 2018. Dynamic multi-objective workflow scheduling for cloud computing based on evolutionary algorithms. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*. IEEE, 103–108.

[51] Goshgar Can Ismayilov. 2025. Towards DevOps of Zero-Knowledge Proofs on Blockchain: LLM-Enhanced Proof Generation and Contract Deployment. In *2025 International Conference on Big Data, Knowledge and Control Systems Engineering (BdKCSE)*. IEEE, 1–8.

[52] Goshgar Can Ismayilov. 2025. zkEC@ 0.0. 1: Evolutionary Computation Meets Programmable Cryptography on Blockchain. In *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing*. 154–156.

[53] Goshgar C Ismayilov and Can Ozturan. 2024. PRFX: A Privacy-Preserving Prefix Summation Protocol on Blockchain with Zero-Knowledge Proof. In *2024 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 362–369.

[54] Goshgar C Ismayilov and Can Ozturan. 2025. Privacy-Preserving Multi-Objective Optimization using Bellman-Ford Algorithm via Zero-Knowledge Proofs. (2025).

[55] Goshgar C Ismayilov and Can Özturan. 2025. Trustless privacy-preserving data aggregation on Ethereum with hypercube network topology. *Computer Communications* 230 (2025), 108009.

[56] Mamudu Francis Itanyi, B Modi, and Mamudu Friday. 2023. A car insurance claim processing prototype using smart contracts and zero-knowledge proofs. *Dutse Journal of Pure and Applied Sciences* 9, 4b (2023), 362–371.

[57] Uzma Jafar, Mohd Juzaiddin Ab Aziz, and Zarina Shukur. 2021. Blockchain for electronic voting system—review and open research challenges. *Sensors* 21, 17 (2021), 5874.

[58] Ya-wen Jeng, Yung-chen Hsieh, and Ja-Ling Wu. 2019. Step-by-step guidelines for making smart contract smarter. In *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 25–32.

[59] Shunrong Jiang, Jinpeng Li, Xiaoyan Zhang, Hao Yue, Haiqin Wu, and Yong Zhou. 2023. Secure and privacy-preserving energy trading with demand response assistance based on blockchain. *IEEE Transactions on Network Science and Engineering* 11, 1 (2023), 1238–1250.

[60] Komal Kalra, Sandeep Kumar Shukla, and Shubham Sahai. 2021. Investment Compliance in Hedge Funds using Zero Knowledge Proofs. *The Journal of The British Blockchain Association* (2021).

[61] Ghazaleh Keshavarzkalhori, Cristina Perez-Solá, Guillermo Navarro-Arribas, and Jordi Herrera-Joancomartí. 2024. Building Resilient AI: A Solution to Data and Model Poisoning Prevention. In *2024 17th International Conference on Security of Information and Networks (SIN)*. IEEE, 1–8.

[62] Ghazaleh Keshavarzkalhori, Cristina Perez-Sola, Guillermo Navarro-Arribas, Jordi Herrera-Joancomarti, and Habib Yajam. 2023. Federify: A verifiable federated learning scheme based on zksnarks and blockchain. *IEEE Access* 12 (2023), 3240–3255.

[63] Dodo Khan, Low Tang Jung, and Manzoor Ahmed Hashmani. 2021. Systematic literature review of challenges in blockchain scalability. *Applied Sciences* 11, 20 (2021), 9372.

[64] Jing Huey Khor, Michail Sidorov, Nathan Tze Min Ho, and Tze Hank Chia. 2022. Public blockchain-based lightweight anonymous authentication platform using zk-snarks for low-power iot devices. In *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 370–375.

[65] Geunyoung Kim, Yunsik Ham, and Jaecheol Ryou. 2024. Privacy-preserving credential smart contracts using Zokrates. *KSII Transactions on Internet & Information Systems* 18, 8 (2024).

[66] Geunyoung Kim and Jaecheol Ryou. 2023. Digital authentication system in avatar using DID and SBT. *Mathematics* 11, 20 (2023), 4387.

[67] Max Kobelt, Michael Sober, and Stefan Schulte. 2023. A benchmark for different implementations of zero-knowledge proof systems. In *2023 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 33–40.

[68] DO Kondyrev. 2024. Comparative Efficiency Analysis of Hashing Algorithms for Use in zk-SNARK Circuits in Distributed Ledgers. *Programming and Computer Software* 50, 4 (2024), 283–291.

[69] Benjamin Körbel, Marten Sigwart, Philip Frauenthaler, Michael Sober, and Stefan Schulte. 2021. Blockchain-based result verification for computation offloading. In *International Conference on Service-Oriented Computing*. Springer, 99–115.

[70] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. 2016. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)*. IEEE, 839–858.

[71] Athanasios Koulianos, Panagiotis Paraskevopoulos, Antonios Litke, and Nikolaos K Papadakis. 2024. Enhancing unmanned aerial vehicle security: A zero-knowledge proof approach with zero-knowledge succinct non-interactive arguments of knowledge for authentication and location proof. *Sensors* 24, 17 (2024), 5838.

[72] Vlasis Koutsos, Sankarshan Damle, Dimitrios Papadopoulos, Sujit Gujar, and Dimitris Chatzopoulos. 2024. AVeCQ: Anonymous Verifiable Crowdsourcing With Worker Qualities. *IEEE Transactions on Dependable and Secure Computing* 22, 1 (2024), 406–423.

[73] SCIPR Lab and contributors. 2012–2017. *libsnark: A C++ Library for zkSNARKs.* https://github.com/scipr-lab/libsnark Accessed: 2025-12-24.

[74] Chaehyeon Lee, Jonathan Heiss, Stefan Tai, and James Won-Ki Hong. 2024. End-to-end verifiable decentralized federated learning. In *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 434–442.

[75] Aoxuan Li, Gabriele D'Angelo, Su-Kit Tang, Frank Fang, and Baron Gong. 2025. An auditable confidentiality protocol for blockchain transactions. *Blockchain: Research and Applications* (2025), 100380.

[76] AoXuan Li, Su-Kit Tang, and Gabriele D'Angelo. 2023. Implementation and preliminary evaluation of an auditable confidentiality mechanism for DeFi. In *2023 IEEE 43rd International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE, 49–54.

[77] Wanxin Li, Collin Meese, Hao Guo, and Mark Nejad. 2023. Aggregated zero-knowledge proof and blockchain-empowered authentication for autonomous truck platooning. *IEEE Transactions on Intelligent Transportation Systems* 24, 9 (2023), 9309–9323.

[78] Qi Lin, Binbin Gu, and Faisal Nawab. 2024. RollStore: hybrid onchain-offchain data indexing for blockchain applications. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[79] Ji-Yao Liu, Yun-Hua He, Chao Wang, Yan Hu, Hong Li, and Li-Min Sun. 2019. An anonymous blockchain-based logging system for cloud computing. In *International Conference on Blockchain and Trustworthy Systems*. Springer, 288–301.

[80] Duc Anh Luong and Jong Hwan Park. 2022. Privacy-preserving blockchain-based healthcare system for IoT devices using zk-SNARK. *IEEE Access* 10 (2022), 55739–55752.

[81] Duc Anh Luong and Jong Hwan Park. 2023. Privacy-preserving identity management system on blockchain using Zk-SNARK. *IEEE Access* 11 (2023), 1840–1853.

[82] Maher Maalla et al. 2024. Enhancing attribute-based access control with Ethereum and ZK-SNARK technologies. *Journal Scientific and Technical Of Information Technologies, Mechanics and Optics* 157, 5 (2024), 797.

[83] Damiano Di Francesco Maesa, Andrea Lisi, Paolo Mori, Laura Ricci, and Gianluca Boschi. 2023. Self sovereign and blockchain based access control: Supporting attributes privacy with zero knowledge. *Journal of Network and Computer Applications* 212 (2023), 103577.

[84] Damiano Di Francesco Maesa, Matteo Loporchio, and Frank Tietze. 2025. Privacy-Preserving and Automated Intellectual Property License Agreements over Heterogeneous Blockchain Networks. *Blockchain: Research and Applications* (2025), 100288.

[85] Godwin Mandinyenya and Vusumuzi Malele. 2025. A Hybrid Framework for Enhancing Privacy in Blockchain-Based Personal Data Sharing using Off-Chain Storage and Zero-Knowledge Proofs. *Journal of Information Systems and Informatics* 7, 2 (2025), 1977–2005.

[86] R Manjula Devi, P Keerthika, P Suresh, R Venkatesan, M Sangeetha, C Sagana, and K Devendran. 2022. Post-Quantum Confidential Transaction Protocols. *Quantum Blockchain: An Emerging Cryptographic Paradigm* (2022), 201–219.

[87] Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. 2013. Zerocoin: Anonymous distributed e-cash from bitcoin. In *2013 IEEE symposium on security and privacy*. IEEE, 397–411.

[88] David Moher, Alessandro Liberati, Jennifer Tetzlaff, Douglas G Altman, Prisma Group, et al. 2010. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *International journal of surgery* 8, 5 (2010), 336–341.

[89] Eduardo Morais, Tommy Koens, Cees Van Wijk, and Aleksei Koren. 2019. A survey on zero knowledge range proofs and applications. *SN Applied Sciences* 1, 8 (2019), 946.

[90] Stefan More, Sebastian Ramacher, Lukas Alber, and Marco Herzl. 2022. Extending Expressive Access Policies with Privacy Features. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 574–581.

[91] Ahmad Musamih, Abduraouf Hassan, Khaled Salah, Raja Jayaraman, Mohammed Omar, and Ibrar Yaqoob. 2025. Leveraging blockchain and machine learning to promote child labor-free sustainable development. *Distributed Ledger Technologies: Research and Practice* 4, 1 (2025), 1–32.

[92] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf. Accessed: 2025-12-24.

[93] Reza Nourmohammadi, Iman Behravan, and Kaiwen Zhang. 2023. Privacy-preserving genomic analysis via PSO-driven federated learning on blockchain. In *2023 3rd Intelligent Cybersecurity Conference (ICSC)*. IEEE, 17–25.

[94] NVIDIA Corporation. 2025. *NVIDIA CUDA Toolkit*. https://developer.nvidia.com/cuda-toolkit Accessed: 2025-12-24.

[95] Eric Joshua Nyato, Emmanuel Kimito, Jaehun Yang, Doyeop Lee, and Dongmin Lee. 2024. Blockchain-integrated zero-knowledge proof system for privacy-preserving near-miss reporting in construction projects. *Automation in Construction* 168 (2024), 105825.

[96] Alex Ozdemir, Fraser Brown, and Riad S Wahby. 2022. CirC: Compiler infrastructure for proof systems, software verification, and more. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2248–2266.

[97] Andreea-Elena Panait and Ruxandra F Olimid. 2020. On using zk-SNARKs and zk-STARKs in blockchain-based identity management. In *International Conference on Information Technology and Communications Security*. Springer, 130–145.

[98] Junhoo Park, Hyekjin Kim, Geunyoung Kim, and Jaecheol Ryou. 2020. Smart contract data feed framework for privacy-preserving oracle system on blockchain. *Computers* 10, 1 (2020), 7.

[99] Juha Partala, Tri Hong Nguyen, and Susanna Pirttikangas. 2020. Non-interactive zero-knowledge for blockchain: A survey. *IEEE Access* 8 (2020), 227945–227961.

[100] Torben Pryds Pedersen. 1991. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual international cryptology conference*. Springer, 129–140.

[101] Marco Peise, Jörn Kuhlenkamp, Anselm Busse, Jacob Eberhardt, Max-R Ulbricht, Stefan Tai, Jörg Baus, Martin Kassebaum, and Thorsten Zörner. 2021. Blockchain-based local energy grids: advanced use cases and architectural considerations. In *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 130–137.

[102] Li Peng, Wei Feng, Zheng Yan, Yafeng Li, Xiaokang Zhou, and Shohei Shimizu. 2021. Privacy preservation in permissionless blockchain: A survey. *Digital Communications and Networks* 7, 3 (2021), 295–307.

[103] Bertalan Zoltán Péter and Imre Kocsis. 2024. Privacy-Preserving Noninteractive Compliance Audits of Blockchain Ledgers with Zero-Knowledge Proofs. *Acta Polytech. Hung* 21 (2024), 7–27.

[104] Massimiliano Pirani, Alessandro Cucchiarelli, Tariq Naeem, and Luca Spalazzi. 2025. Verifiable Actor Model Systems Through Relational-Model Multi-Agent System and Zero-Knowledge Proofs. In *2025 IEEE 8th International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 01–06.

[105] Wenjie Qu, Yijun Sun, Xuanming Liu, Tao Lu, Yanpei Guo, Kai Chen, and Jiaheng Zhang. 2025. zkGPT: An Efficient Non-interactive Zero-knowledge Proof Framework for LLM Inference. In *34st USENIX Security Symposium (USENIX Security 25)*.

[106] Vinh Quach, Ram Dantu, Sirisha Talapuru, Shakila Zaman, and Apurba Pokharel. 2024. ZCube: A Zero-Trust, Zero-Knowledge, and Zero-Memory Platform for Privacy and yet Secured Access. In *2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*. IEEE, 166–175.

[107] Anupam Raj, Ananya Gupta, Aditya Biswakarma, Rohan Tirkey, Junaid Alam, and Soumyadev Maity. 2025. OpenAudit: Enhancing Software Compliance with Blockchain and Zero-Knowledge Proofs. In *International Conference on Network Security and Blockchain Technology*. Springer, 207–223.

[108] Bhavye Sharma, Raju Halder, and Jawar Singh. 2020. Blockchain-based interoperable healthcare using zero-knowledge proofs and proxy re-encryption. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, 1–6.

[109] Michael Sober, Max Kobelt, Giulia Scaffino, Dominik Kaaser, and Stefan Schulte. 2023. Distributed key generation with smart contracts using zk-SNARKs. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*. 231–240.

[110] Lihua Song, Xinran Ju, Zongke Zhu, and Mengchen Li. 2021. An access control model for the Internet of Things based on zero-knowledge token and blockchain. *EURASIP Journal on Wireless Communications and Networking* 2021, 1 (2021), 105.

[111] Samuel Steffen, Benjamin Bichsel, Mario Gersbach, Noa Melchior, Petar Tsankov, and Martin Vechev. 2019. zkay: Specifying and enforcing data privacy in smart contracts. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 1759–1776.

[112] Xiaoqiang Sun, F Richard Yu, Peng Zhang, Zhiwei Sun, Weixin Xie, and Xiang Peng. 2021. A survey on zero-knowledge proof in blockchain. *IEEE network* 35, 4 (2021), 198–205.

[113] Domenico Tortola, Andrea Pelosi, Giuseppe Gabriele Russo, Paolo Mori, Laura Ricci, et al. 2024. zkSNARKs Libraries for Blockchains: a Comparative Study. In *CEUR WORKSHOP PROCEEDINGS*, Vol. 3791. CEUR-WS.

[114] Taochun Wang, Huimin Shen, Jian Chen, Fulong Chen, Qingshan Wu, and Dong Xie. 2023. A hybrid blockchain-based identity authentication scheme for mobile crowd sensing. *Future Generation Computer Systems* 143 (2023), 40–50.

[115] Bohang Wei, Yang Yang, Shihong Xiong, Minghang Li, Qianhong Wu, and Bo Qin. 2025. TrustBlink: A zkSNARK-Powered On-Demand Relay for PoW Cross-Chain Verification With Low Costs. In *International Conference on Information and Communications Security*. Springer, 119–138.

[116] Robert Werner, Dominique Briechle, and Marit Mathiszig. 2024. Hitchhikebox: a decentral, verifiable, and privacy-protecting automated logistic transport concept for pharmaceuticals. In *Proceedings of the 2024 10th International Conference on Computer Technology Applications*. 86–93.

[117] Martin Westerkamp and Jacob Eberhardt. 2020. zkrelay: Facilitating sidechains using zksnark-based chain-relays. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 378–386.

[118] Christopher Wiraatmaja and Shoji Kasahara. 2025. Scalable Anonymous Authentication Scheme Based on Zero-Knowledge Set-Membership Proof. *Distributed Ledger Technologies: Research and Practice* 4, 1 (2025), 1–24.

[119] Karl Wüst and Arthur Gervais. 2018. Do you need a blockchain?. In *2018 crypto valley conference on blockchain technology (CVCBT)*. IEEE, 45–54.

[120] Dongwei Xiao, Zhibo Liu, Yiteng Peng, and Shuai Wang. 2025. Mtzk: Testing and exploring bugs in zero-knowledge (zk) compilers. In *NDSS*.

[121] Zhibo Xing, Zijian Zhang, Ziang Zhang, Zhen Li, Meng Li, Jiamou Liu, Zongyang Zhang, Yi Zhao, Qi Sun, Liehuang Zhu, et al. 2025. Zero-Knowledge Proof-Based Verifiable Decentralized Machine Learning in Communication Network: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials* (2025).

[122] Shiyuan Xu, Xue Chen, and Yunhua He. 2021. EVchain: An anonymous blockchain-based system for charging-connected electric vehicles. *Tsinghua Science and Technology* 26, 6 (2021), 845–856.

[123] Xiaohui Yang and Wenjie Li. 2020. A zero-knowledge-proof-based digital identity management scheme in blockchain. *Computers & Security* 99 (2020), 102050.

[124] Xiaohui Yang and Kun Zhang. 2024. A Fair and Trusted Trading Scheme for Medical Data Based on Smart Contracts. *Computers, Materials & Continua* 78, 2 (2024).

[125] Hao Yu, Guijuan Wang, Anming Dong, Yubing Han, Yawei Wang, and Jiguo Yu. 2025. Blockchain-enabled privacy protection scheme for IoT digital identity management. *High-Confidence Computing* (2025), 100320.

[126] Ke Yuan, Yingjie Yan, Tong Xiao, Wenchao Zhang, Sufang Zhou, and Chunfu Jia. 2021. Privacy-protection scheme of a credit-investigation system based on blockchain. *Entropy* 23, 12 (2021), 1657.

[127] Andreas Zeiselmair, Bernd Steinkopf, Ulrich Gallersdörfer, Alexander Bogensperger, and Florian Matthes. 2021. Analysis and application of verifiable computation techniques in blockchain systems for the energy sector. *Frontiers in Blockchain* 4 (2021), 725322.

[128] Boyang Zhang, Jiping Xu, Xiaoyi Wang, Zhiyao Zhao, Shichao Chen, and Xin Zhang. 2023. Research on the construction of grain food multi-chain blockchain based on zero-knowledge proof. *Foods* 12, 8 (2023), 1600.

[129] Collin Zhang, Zachary DeStefano, Arasu Arun, Joseph Bonneau, Paul Grubbs, and Michael Walfish. 2024. Zombie: Middleboxes that {Don't} snoop. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. 1917–1936.

[130] Xiaohong Zhang, Xingxing Chen, Shuling Liu, and Shaojiang Zhong. 2024. Anonymous authentication and information sharing scheme based on blockchain and zero knowledge proof for vanets. *IEEE Transactions on Vehicular Technology* (2024).

[131] Xinjian Zhao, Fei Xia, Hanning Xia, Yunlong Mao, and Shi Chen. 2024. A zero-knowledge-proof-based anonymous and revocable scheme for cross-domain authentication. *Electronics* 13, 14 (2024), 2730.

[132] Jiapeng Zhou, Yuxiang Feng, Zhenyu Wang, and Danyi Guo. 2021. Using secure multi-party computation to protect privacy on a permissioned blockchain. *Sensors* 21, 4 (2021), 1540.