IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 27, 2020, accepted December 13, 2020, date of publication December 21, 2020,
date of current version December 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3046025

Non-Interactive Zero-Knowledge for Blockchain: A Survey

JUHA PARTALA[1], TRI HONG NGUYEN[2], AND SUSANNA PIRTTIKANGAS[2], (Member, IEEE)
[1]Center for Machine Vision and Signal Analysis, University of Oulu, 90570 Oulu, Finland
[2]Center for Ubiquitous Computing, University of Oulu, 90570 Oulu, Finland

Corresponding author: Juha Partala (juha.partala@oulu.fi)

This work was supported in part by a strategic research project called "Personalization, Privacy and Quality Control for MaaS with Blockchain - TrustedMaaS" through the focus institute Infotech Oulu, Faculty of Information Technology and Electrical Engineering (ITEE), University of Oulu, Finland, and in part by the Academy of Finland 6Genesis Flagship under Grant 318927.

ABSTRACT We survey the state-of-the-art non-interactive zero-knowledge argument schemes and their applications in confidential transactions and private smart contracts on blockchain. The main goal of this paper is to serve as a reference for blockchain application developers in finding the most suitable scheme for a particular use case. We give an overview and compare the state-of-the-art protocols for confidential transactions and private smart contracts regarding the protection of the transaction graph and amounts, data and functionality. However, our main focus is on state-of-the-art zero-knowledge argument schemes. We briefly describe their backgrounds, proof lengths and computational complexities and discuss their cryptographic security models. Our focus is on seminal, otherwise notable and, especially, implemented methods that are ready to be applied in practice. We also survey the existing implementations of transforming computations into circuit representations required by those methods. We note that the existing schemes have different strengths and drawbacks regarding usability, setup, proof length and proving and verification costs.

INDEX TERMS Distributed computing, peer-to-peer systems, cyber trust, cyber security, cryptography, privacy.

I. INTRODUCTION

We are increasingly dependent on services provided by smart devices and communication networks. Massive collection of data has enabled the development of new consumer devices such as smart home appliances, as well as the improvement of the efficiency of industrial applications. However, massive data collection has also lead to security and privacy issues. For example, a centralized Internet-of-Things (IoT) architecture involves a single point of trust which is a major target for malicious entities. Decentralization can provide a solution to this problem. In particular, blockchain has found use in the creation of trustless, decentralized environments for managing the integrity of data for applications such as IoT [1]–[3].

Virtually any functionality can be implemented on a blockchain network using smart contracts that enable devices to automatically verify and execute transactions between entities [4]. Blockchain and smart contracts enable decentralized applications (DApps) [5] to be run on a peer-to-peer network. These applications are controlled and run by the network as a whole enabling the users to retain the ownership and control of their data. Progress in such a direction is important. For example, the General Data Protection Regulation (GDPR) requires better protection on sensitive data and grants European Union citizens more control over it.

Blockchain itself does not ensure that the user data stays private. Even though designed to hide user identities using pseudonyms, it is a misconception that a blockchain is anonymous. There are several challenges regarding the security and privacy of blockchain. For example, transaction privacy cannot be guaranteed due to the visibility of the sender and receiver addresses and transaction amounts [6], [7]. In addition, pseudonyms can be linked to individuals based on their connection patterns [8].

There are privacy-protection techniques for blockchain such as mixing services, blind and ring signatures and homomorphic commitments. However, *zero-knowledge argument schemes* are going to be one of the main building blocks of privacy-preserving blockchain applications. They can be used

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

to assure the validity of a statement or the knowledge of secret values without disclosing any information apart from the fact that the prover knows those things. Such functionality enables us to guarantee the validity of the data on the blockchain without disclosing sensitive information. It can also implement *verifiable computing* which enables computation offloading to untrusted entities for edge computing. Through verifiable computing a client can verify the validity of the computation from a proof of its correctness.

In order to implement privacy-preserving applications for blockchain, developers need to be aware of the existing privacy protection and zero-knowledge schemes, their properties and suitability for different use cases. In this paper, we have conducted a survey of zero-knowledge based privacy-protection methods targeted for blockchain applications and an extensive survey on the state-of-the-art zero-knowledge argument schemes. However, there is a vast literature on the development of such protocols and it would be intractable to include all of that work in the limited space. We have concentrated on peer-reviewed schemes that either 1) are seminal or otherwise notable, 2) are recent and amongst the most optimized variants of their class or 3) have a practical implementation that blockchain application developers could immediately apply in practice. We classify these schemes into five main classes based on their properties. Regarding practical implementations, there are also general purpose libraries that have implemented multiple approaches to zero-knowledge and we devote a separate section to those. In order to achieve the required functionality, all of the schemes require computations to be transformed into a circuit representation. Therefore, we also discuss existing implementations for circuit generation.

The paper is organized as follows. Section II gives an overview of blockchain technology, its use cases and privacy issues. In Section III, we describe the concept of a zero-knowledge proof and the requirements for their practical applicability. Section IV presents a survey of zero-knowledge based privacy-protection schemes for blockchain applications. Section V is devoted to circuit generation tools that are required for the applications of zero-knowledge arguments. In Sections VI to X, we discuss the five classes of zero-knowledge argument schemes. In Section XI, general purpose libraries are presented. Section XII is devoted to the discussion of the theoretical and practical performance of these schemes. Finally, Section XIII provides the conclusion.

A. RELATED WORK

We concentrate on zero-knowledge argument schemes for blockchain applications. To the best of our knowledge, there does not exist a recent survey on the topic. There exists a large body of work concentrating on cryptocurrencies and on the blockchain itself [9]–[12], as well as on the security threats and challenges related to blockchain technology [13], [14]. Privacy aspects have been considered in [15]–[17] and privacy-preservation in public blockchains in [18]. Range proofs based on zero-knowledge have been surveyed in [19].

Surveys concentrating on threats and challenges to the security of blockchain-based IoT can be found in [20]–[22]. Contrary to our survey, the performance of practical implementations of zero-knowledge argument schemes has not been considered in these surveys. However, for general information on the threats and challenges regarding security and privacy, we refer to the extensive work in the aforementioned surveys.

II. BLOCKCHAIN AND PRIVACY

Blockchain can be considered as a database which is distributed and decentralized over a peer-to-peer network. It utilizes cryptographic algorithms to verify and record data into a series of blocks which is distributed to the peers. These blocks are linked into an immutable chain and new verified blocks are connected to previous ones using cryptographic hashing. As a result, the blockchain is tamper-resistant; it is computationally infeasible to alter data that has already been published in the chain. In order to form a unique blockchain, the network peers need to be able to verify the validity of the data for the next block and to achieve a consensus on it. The central novelty of blockchain is the elimination of intermediaries or trusted parties in this process.

Combining confidentiality with public verifiability is not easy. Privacy issues related to blockchain technology can be found, for example, in the surveys of Khalilov and Levi [17], Feng *et al.* [23] and Herskind *et al.* [24]. All nodes need to be able to verify the validity of the data. This leads to issues such as de-anonymization by linking identities to blockchain addresses or vice versa. In addition, the contents of the blockchain can be analyzed to determine the real world identities corresponding to each bit of data. The behavioral aspects of the submitted data can be also analyzed. For example, the time of the day of the submission may betray information about the geo-location of the blockchain address controller. Other attributes can also be monitored to create a profile of the individual. Regarding smart contracts, fields in the contract can be analyzed to determine the identities of the contractual parties. Even if certain fields of the contract are declared private, the transactions related to the contract may reveal information about the contract.

The only way to prevent these kinds of attacks is to make the stored data confidential. However, if data is encrypted it is impossible to publicly verify its validity using conventional methods. Fortunately, zero-knowledge schemes provide such functionality.

III. ZERO-KNOWLEDGE PROOFS

Non-interactive zero-knowledge proofs are the main building blocks for privacy on the blockchain. They enable us to demonstrate the validity of a statement without leaking any other information. A zero-knowledge protocol is a versatile building block for many privacy-oriented applications. They have been used to realize, among others, digital signature schemes, electronic voting, verifiable computing and user identification protocols.

Goldwasser, Micali and Rackoff considered the question of proving a proposition in zero-knowledge in their seminal work in [25]. Independently, a similar question was posed by Brassard, Chaum and Crépeau [26]. In general, there are two participants in a zero-knowledge proof system. The *prover*, in possession of a possible witness to the validity of a proposition, wants to convince a *verifier*. The following three properties need to be satisfied:

1) **Completeness.** Knowing a witness to the validity of a statement, the prover is able to convince the verifier.
2) **Soundness.** A malicious prover is not able to convince the verifier in the case that the statement is false.
3) **Zero-knowledge.** The verifier learns nothing except that the statement is true.

In applications, we often require that the prover can efficiently extract a witness for the statement when the verifier is convinced. Such a strengthening of the soundness condition is called a **proof of knowledge**.

Under the assumption that there is a secure encryption scheme, all problems in the class of decision problems solvable in non-deterministic polynomial time, NP, have interactive zero-knowledge proofs [27]. Depending on the requirements, the three properties can be satisfied either perfectly, probabilistically or using computational restrictions. For example, in zero-knowledge with *computational soundness*, the soundness is only satisfied against polynomial time adversaries. Zero-knowledge schemes with computational soundness are called *argument systems* in order to distinguish them from zero-knowledge proofs that always have perfect soundness. In argument systems the prover is restricted to polynomial-time computation which is a reasonable assumption in practice. Analogous conditions can be defined for computational zero-knowledge.

A. NON-INTERACTIVE ZERO-KNOWLEDGE ARGUMENTS

Zero-knowledge proofs (in the traditional sense) are interactive, online protocols. However, for blockchain applications we need offline functionality such as non-interactive verification. Only the class of decision problems solvable in bounded-error probabilistic polynomial time, BPP, has non-interactive zero-knowledge proofs in the standard model without any setup [28], [29] meaning that additional assumptions or functionality are needed.

There are multiple widely used methods for transforming interactive zero-knowledge proofs into non-interactive ones. They make different assumptions regarding the setup of the system or the underlying cryptographic model. The following approaches have been adopted by the methods surveyed in this paper:

1) **The common reference string (CRS) model.** Blum, Feldman and Micali gave the first non-interactive zero-knowledge (NIZK) proof for any problem in NP based on the participants sharing a string [30]. The CRS needs trusted setup and all participants need to have access to the same string. Any scheme following this model can be secure only if the CRS was generated correctly and securely. For a large number of participants the CRS generation can be a complex and time consuming process. Therefore, CRS-based schemes are hard to setup. However, they are typically also efficient to operate and the proofs are small.
2) **The Fiat-Shamir heuristic** [31] is a method of creating a digital signature scheme from an interactive proof of knowledge protocol that uses public randomness. The Fiat-Shamir transform replaces interaction and (a part of) the randomness by an application of a public cryptographic hash function, whose outputs, in some sense, can be interpreted as the CRS. The security is preserved in the *random oracle model* (ROM) [32]: we assume that the output of the hash function is uniformly random and independent for different inputs. The random oracle model is significantly stronger than the standard model and not all security properties of zero-knowledge are preserved [33]. However, no complex setup is needed.
3) **Unruh transformation** [34] is an alternative technique to the Fiat-Shamir heuristic. In general, the Fiat-Shamir method is not secure against quantum computing and yields insecure schemes for some protocols [35]. The Unruh transformation gives a provably secure NIZK proof for any interactive one against quantum adversaries in the ROM. Similar to Fiat-Shamir, no setup is needed.
4) **Kalai *et al.*** [36] suggested an argument system for any decision problem using private information retrieval. The method applies a model of multi-prover interactive proof systems (MIP). A MIP is converted into an argument system based on the methods of Aiello *et al.* [37] and the construction works in the standard model without the random oracle assumption. This approach is used in some of the proofs-for-muggles based zero-knowledge arguments (see Section VII).

In general, there are two main classes of zero-knowledge arguments:

1) Σ-protocols are interactive proof-of-knowledge protocols that can be transformed into non-interactive versions using the Fiat-Shamir paradigm. Methods based on Σ-protocols are efficient at proving algebraic statements such as those related to the discrete logarithm problem. For non-algebraic statements and, in particular, for general computations Σ-protocols tend to be inefficient.
2) Zero-knowledge succinct non-interactive arguments of knowledge (SNARKs) [38] follow the common reference string model. They typically produce very short proofs compared to Σ-protocols, but the CRS needs to be generated in advance. Some schemes require trusted setup or the application of costly methods such as secure multiparty computation for the CRS [39].

In addition, in general the CRS cannot be reused if the computation changes. However, there are emerging models well suited for blockchain applications that address these issues by maintaining a CRS that can be updated [40]. If only public randomness can be used, the scheme is called *transparent*.

B. REPRESENTING STATEMENTS AND COMPUTATIONS

If not proving algebraic statements, state-of-the-art zero-knowledge argument systems require the computation to be presented as a circuit. Depending on the method, the circuit needs to be either Boolean or arithmetic. In order to apply an existing zero-knowledge argument scheme, a developer thus needs to transform the computation into a circuit. Fortunately, there are tools that can be applied to make the conversion. We have collected existing circuit generation methods into Section V.

C. OUR CLASSIFICATION

We attempt to classify state-of-the-art zero-knowledge arguments and their implementations into groups with similar design choices and properties. A non-overlapping classification is difficult, because suggestions typically borrow techniques from each other and combine techniques from Σ-protocols and SNARKs. Here, we attempt a classification that is based on both the design approach and the typical differences in the operating characteristics of the schemes, such as prover and verifier complexities, proof lengths and best-case applicability.

In our classification, there are five main classes of constructions: 1) zero-knowledge based on interactive oracle proofs (IOP) and probabilistically checkable proofs (PCP), 2) schemes based on the so called "proofs-for-muggles" approach, 3) schemes based on linear PCPs and the discrete logarithm problem, 4) zero-knowledge based on quadratic arithmetic programs (QAP) and 5) schemes based on secure multiparty computation (MPC). We present these approaches briefly together with existing schemes following them. A sixth class could be added to the list: an approach based on non-interactive computing protocols using fully homomorphic encryption [41]. However, these methods are still too expensive in terms of computation to be considered practically feasible.

There is a vast literature on the theory of zero-knowledge argument systems and their applications and it would be impractical to include all of that work into this survey. Therefore, we have concentrated on methods that are seminal or otherwise influential, methods that are recent and, especially, those that have an implementation. Further references can be found from the cited literature.

The next section presents a survey of zero-knowledge based privacy-protection methods for blockchain applications. Zero-knowledge argument schemes have been surveyed in the subsequent sections. Finally, there are general purpose libraries that gather several implementations into a single package and provide tools for circuit generation. These methods have been collected into Section X.

IV. ZERO-KNOWLEDGE IN BLOCKCHAIN APPLICATIONS

Zero-knowledge arguments are used as a building block in many privacy solutions for blockchain-based applications. In this section, we have surveyed such solutions. Due to the history of blockchain technology, many applications concentrate on financial blockchains and transactions. However, there are also schemes that target a wider application scenario using smart contracts.

Smart contract based solutions provide flexible mechanisms without relying on trusted parties by transferring the required trust into the trust of the cryptocurrency. They enable fair exchange of goods, services and data using, for example, zero-knowledge contingent service payments (ZKCSP) [42]. In ZKCP, users can lock funds in the blockchain based on a puzzle and to release those funds once a solution is provided. Anyone demonstrating a solution, often in zero-knowledge, can claim the funds. Zero-knowledge arguments are also used to prove the solvency of cryptocurrency exchanges [43] and in blockchain applications involving Internet-of-Things (IoT) devices [44] and electronic voting [45].

Transaction data needs to be publicly verifiable for the consensus algorithm to work. Therefore, public verifiability needs to be maintained while sensitive data is protected. In the following, we have surveyed the methods for preserving the privacy of blockchain transactions and smart contracts. Zero-knowledge proofs and arguments often provide the key functionality of these methods.

A. PRIVACY OF TRANSACTIONS AND ASSETS

For commercial reasons, it is imperative to hide both your transaction amounts, as well as the parties you are doing business with. Privacy-preserving transaction schemes attempt to provide such functionality while maintaining the public verifiability of transaction validity. In the literature, *confidentiality* refers to the protection of account or wallet balances and transaction amounts. *Anonymity* refers to the protection of the transaction graph; the obfuscation of the sender and receiver addresses to make it impossible to trace who transferred assets to whom.

In Bitcoin, coins are stored in *unspent transaction outputs* (UTXO). Whenever a valid transaction is made, the unspent coins are forwarded into an UTXO. These unspent transaction outputs can be used as inputs for future transactions. Several cryptocurrencies follow the UTXO model and there are privacy-preservation schemes that apply only to this model. Ethereum, on the other hand, follows a different model, where the balance of a user is stored globally in an account. Cryptocurrencies based on the account model require their own privacy-preservation schemes.

Preliminary attempts to provide transaction privacy were limited due to their designed compatibility with the Bitcoin network. Such services (so called *mixes, laundries* or *tumblers*), such as *CoinJoin*, attempt to scramble transactions by

TABLE 1. Privacy-protection schemes for transactions, the corresponding transaction model (unspent transaction outputs (UTXO) or account) and blockchain type, regulatory compliance, protection of sender and receiver addresses and transaction amounts.

Scheme	Model	Blockchain	Regulation	Sender privacy	Receiver privacy	Amount secrecy
CoinJoin (and other tumblers)	UTXO	public	No	Partial	Partial	No
Confidential Transactions	UTXO	public	No	No	No	Yes
ValueShuffle	UTXO	public	No	Partial	Partial	Yes
CryptoNote	UTXO	public	No	Partial	Partial	Partial
Monero (orig.)	UTXO	public	No	Partial	Partial	Yes
Monero 2020	UTXO	public	No	Yes	Yes	Yes
Mimblewimble	UTXO	public	No	Partial	Partial	Yes
Zerocoin	UTXO	public	No	Yes	No	No
Zerocash	UTXO	public	No	Yes	Yes	Yes
Policy-enforced Zerocash	UTXO	public	Yes	Yes	Yes	Yes
Quisquis	UTXO	public	No	Partial	Partial	Yes
DCAP	UTXO	public	Yes	Yes	Yes	Yes
Androulaki et al.	UTXO	permissioned	Yes	Yes	Yes	Yes
BlockMaze	account	public	No	Yes	Yes	Yes
AttriChain	account	permissioned	Yes	Yes	Yes	Yes
Solidus	account	public	Yes	Yes	Yes	Yes
PGC	account	public	Yes	No	No	Yes
zkLedger	account	permissioned	Yes	Yes	Yes	Yes

bundling those of multiple entities together into an *anonymity set*. The aim is to make it too hard for anybody to trace transactions back to a specific individual. There exists other transaction obfuscation protocols and services inspired by CoinJoin such as CoinShuffle and TumbleBit [46]. However, the size of the anonymity set is often limited and it has been observed that these methods are not effective in practice [47]. See, for example, [23] for a summary of existing mixing services.

Transaction confidentiality can be achieved using commitment and secret sharing schemes [48] together with zero-knowledge arguments. The zero-knowledge argument is typically used to show that the sum of the inputs in a transaction is greater than that of the outputs and that the output amounts are positive. It is also infeasible to use exact values without leaking information. Therefore, ranges of values are used instead resulting in so called *range proofs*. One of the major limiting factors regarding the application of zero-knowledge is the actual size of such proofs. Since these proofs need to be included into the blockchain, their size should be compact.

The protection of anonymity and the confidentiality of transactions has attracted a lot of interest from the scientific community. In the following, we have presented the seminal, the most notable, as well as the most recent schemes for the protection of transactions. Most of these schemes work for public blockchains, enabling any party to join, submit transactions and become a validator. However, some schemes work only for permissioned blockchains, where participation is restricted. A comparison has been collected into Table 1.

1) **Confidential Transactions** [49], [50] is designed to hide transaction amounts. Sender and receiver addresses are not protected. The construction applies additively homomorphic Pedersen commitments [48] and zero-knowledge range proofs to enable public verification of the protected transactions. The protocol is compatible with transaction obfuscation schemes and can be used in conjunction with them.
2) **ValueShuffle** [51] is based on the CoinJoin concept and the CoinShuffle proposal that removes CoinJoin's need for a trusted third party. In addition, the Confidential Transactions protocols is applied to hide transaction amounts. Due to the application of anonymity sets, the transaction graph is only partially protected.
3) **CryptoNote** is a protocol for the implementation of untraceable and unlinkable cryptocurrencies using anonymity sets. It applies ring signatures, originally suggested by Rivest *et al.* [52] to hide the identities of the payer and the recipient, as well as the amount. However, the transaction amount is only partially hidden and attacks have been identified against CryptoNote-style blockchains [53].
4) **Monero** is a cryptocurrency originally based on CryptoNote and its strengthening with the Ring Confidential Transactions (RingCT) protocol [54] that combines ring signatures with the Confidential Transactions protocol. Due to attacks on the linkability of the transactions [55], Monero has since moved to considerably stronger full zero-knowledge arguments using the Bulletproofs scheme [56] (see Section VIII).
5) **Mimblewimble** [57], [58] is another combination of transaction obfuscation CoinJoin together with Confidential Transactions. Non-interactive version of CoinJoin due to Saxena *et al.* [59] is applied to obfuscate sender and receiver addresses, while Confidential Transactions hides the transaction amounts. Transactions can be also aggregated to improve scalability and the need to download old transactions when new ones need to be verified is eliminated. The protocol has been proved secure against coin theft and inflation [60].
6) **Zerocoin** [61] is a cryptographic extension to Bitcoin to improve the privacy of transactions. Its goal is to prevent the tracking of individuals by third parties by hiding the sender addresses. However, destinations and amounts remain visible. The protocol applies zero-knowledge arguments based on Schnorr

signatures [62]. Non-interactivity is achieved using the Fiat-Shamir heuristic (see Section III-A). Due to the heavy application of the discrete logarithm problem, the proofs are big and the verification is slow.

7) **Zerocash** [63] is a cryptocurrency protocol implementing fully anonymous transactions based on SNARKs (see Section III-A). They are applied to implement a decentralized anonymous payment (DAP) scheme, where users can place coins into a shielded pool. Compared to Zerocoin, the proof length and verification time are significantly reduced, but a common reference string has to be generated beforehand. Zerocash hides the transaction graph and the amounts. Basically, any zero-knowledge argument scheme can be used as a building block for Zerocash. The original suggestion applies the Pinocchio scheme [64] (see Section IX). The Zerocash protocol is used in the Zcash cryptocurrency.
8) **Policy-enforced Zerocash** attempts to address the issue of cryptocurrency regulation [65]. One of the biggest problems related to decentralized currencies is their use for illegal activities such as extortion and money laundering. Policy-enforced Zerocash is a decentralized anonymous payment system that is capable of enforcing compliance to regulatory policies. The Zerocash protocol is augmented to have such properties using simulation-extractable SNARKs.
9) **Quisquis** [66] improves on Monero using updatable public keys. In Monero and Zerocash the size of the UTXO set increases due to transaction address obfuscation. Quisquis attempts to remove this drawback by enabling users to create anonymity sets by themselves for private transactions. Privacy is ensured by updating the public keys and zero-knowledge arguments are used to show that the keys have been updated and coins have not been stolen. However, anonymity sets do not provide perfect anonymity.
10) **Decentralized Conditional Anonymous Payments (DCAP)** [67] is another protocol for regulated cryptocurrencies in the UTXO model. The authors apply a zero-knowledge signature of knowledge scheme to provide conditional anonymity that is compatible with regulation requirements. However, trusted nodes are required and the authors suggest a permissioned blockchain to be used.
11) **Androulaki *et al.*** [68] propose a privacy-preserving scheme for transactions on permissioned blockchains following the UTXO model. Using zero-knowledge arguments based on bilinear pairings (see Section VIII), the scheme supports fine-grained auditing and regulation. Public blockchains are not supported. Performance measurements based on a prototype implementation using Hyperledger Fabric can be found in [68].
12) **BlockMaze** [69] offers private transactions for account-based blockchains such as Ethereum. SNARKs are used to hide transaction amounts and sender and recipient addresses. The protocol is proven secure under a formal security model and implemented using the general purpose library libSNARK (see Section XI).
13) **AttriChain** [70] offers traceable user anonymity and private transactions on a permissioned blockchain. Due to traceability, the protocol supports regulation and policy enforcement. Threshold cryptography protects users from malicious validators and enables accountability against those regarded by the network as malicious. Zero-knowledge arguments are used to provide privacy against other users. Public blockchains are not supported.
14) **Solidus** [71] is a protocol for confidential transactions on public blockchains and follows the account model. Contrary to contemporary cryptocurrencies, the user accounts are maintained by banks that mediate the transactions hiding amounts and parties. However, each transaction is publicly verifiable on the blockchain using zero-knowledge arguments. Due to banks, regulation is possible, but auditing requires to open the transactions and cannot be done publicly.
15) **PGC** [72] is a system for confidential transactions on an account-based blockchain. The scheme trades anonymity for the support for privacy-preserving audits and regulation. The transaction graph is not hidden and a party can request users to prove their compliance with a set of policies. An integrated signature and encryption scheme is used to provide confidentiality and a non-interactive zero-knowledge argument enables users to prove compliance.
16) **zkLedger** [73] is a distributed ledger system designed for banks for transaction privacy and public verification with auditing. The transaction amount is protected and the transaction graph with the sender and receiver addresses is hidden. Auditing can be performed publicly. Non-interactive zero-knowledge arguments based on Σ-protocols (see Section III-A) are applied together with cryptographic commitments. The size of a transaction grows linearly in the number of participants meaning that the protocol does not scale well. It is thus targeted for ledgers with only a moderate number of users.

B. PRIVATE SMART CONTRACTS

Smart contracts attempt to provide flexibility by enabling self-enforcement and automatically resolving transactions. In general, any computation can be included into a smart contract and they are considered as one of the most potential future applications of blockchain. The Ethereum platform offers a Turing-complete scripting language and thus enables the implementation of virtually any functional-

TABLE 2. Privacy-protection schemes for smart contracts, the transaction model and blockchain type, protection of the sender and receiver addresses, protection of input and output data, obfuscation of the computation and the supported computational functionality.

Scheme	Model	Blockchain	Sender privacy	Receiver privacy	Data privacy	Function privacy	Functionality
Hawk	account	public	Yes	Yes	Yes	No	Full
zkay	account	public	No	No	Yes	No	Full
Zether	account	public	Partial	Partial	Yes	No	Full
Zexe	UTXO	public	Yes	Yes	Yes	Yes	Partial
DSC	account	public	No	No	Yes	No	Full
PPChain	account	permissioned	Partial*	Partial*	Partial*	Partial*	Full

* protected against other users, but not from validating nodes

ity. Zero-knowledge schemes can be applied to protect sensitive information and the computations stored inside a smart contract.

While there are suggestion for the protection of smart contract privacy that are not based on zero-knowledge such as Ekiden [74], which applies a trusted execution environment, the majority of schemes use zero-knowledge arguments. In the following, we have briefly described the state-of-the-art smart contract protection schemes. A comparison can be found in Table 2.

1) **Hawk** [7] implements transaction privacy in a programmable decentralized blockchain such as Ethereum. The protocol hides the transaction amounts between the pseudonyms and enables the binding of those transactions using programmable logic for offline computation. Confidentiality is not provided against parties participating in the contract and the computed function is not hidden. However, the protocol provides security against cheating parties that deviate from the protocol or abort it. A compiler can be used to transform the private portion of a contract into a circuit representation. SNARKs are applied to enforce the correctness of the contract execution, which is handled by a managing third party who can see the user's inputs and needs to be trusted. However, the manager cannot affect the execution of the protocol even by colluding with some of the parties.
2) **zkay** [75] is a language for writing smart contracts with encrypted data. Zero-knowledge arguments are used to prove the correctess of the data and its usage. However, user anonymity is not provided. The language has been implemented for Ethereum and the proving functionality is provided by the ZoKrates library [76] (See Section V-A).
3) **Zether** [77] is a mechanism for making private transactions on smart contract based platforms. It is itself implemented as a smart contract that can keep account balances hidden and to deposit, transfer and withdraw funds. However, the sender and receiver addresses are only partially protected by an anonymity set. A custom zero-knowledge scheme called Σ-Bullets is used to incorporate advantages of both Σ-protocols and SNARKs. The Bulletproofs scheme (see Section IX) is used as the SNARK. Interoperability is provided for other smart contracts enabling arbitrary distributed applications to be privacy-preserving.
4) **Zexe** [78] is a system that enables users to generate publicly verifiable transactions that prove the correctness of an offline computation using SNARKs. This means that, contrary to other schemes, functions can be computed privately. Stateful computations are not supported meaning that only partial smart contract functionality is provided. However, the security model is stronger than that of other schemes: function privacy is also provided.
5) **DSC** [79] is a framework for protecting the balance and transaction amounts for smart contracts on account model blockchains. Homomorphic encryption and zero-knowledge arguments are applied to protect transactions with a mechanism for programmability. The sender and receiver are not protected. However, contrary to Hawk, no trusted parties are required.
6) **PPChain** [80] is a privacy-preserving permissioned blockchain architecture based on Ethereum and the account model. Regulation and smart contracts are supported. Group signatures and broadcast encryption are used to protect data from other users. However, validating and recording nodes see all data and privacy-protection is therefore only partial. In addition, a trusted manager in charge of tracing malicious behavior is needed. Smart contract operations executed through transactions are protected.

V. BOOLEAN AND ARITHMETIC CIRCUIT GENERATION

Zero-knowledge argument schemes require computations to be presented as a circuit. The efficiency of the schemes depends mainly on the size of the circuit C measured in the number of gates denoted by $|C|$. Therefore, circuit generation is one of the most important factors affecting practical performance. The conversion of the computation can be done with manual circuit construction tools or automatically using compilers. Manual conversion tends to produce more optimized circuits, while automatic conversion is more convenient for the developer. Manual conversion tools are typically required for performance critical application.

There are a number of both high-level language compilers and low-level circuit composition tools. We have discussed the most notable of these in the following subsections.

A. HIGH-LEVEL COMPILERS

High level compilers provide developers an easy method of transforming a computation into a circuit. These compilers accept code written in a high-level language. Therefore, both new and existing algorithms can be easily converted. However, some restrictions may have been placed on the structure of the code in order to produce a circuit of adequate size.

1) **Pinocchio** [64], [81] is a complete suite for producing non-interactive zero-knowledge arguments. It contains a high-level compiler to transform C code into a Boolean or arithmetic circuit representation. The resulting circuit is then converted by the compiler into a quadratic arithmetic program (QAP) (see Section IX) for the zero-knowledge argument protocol execution.
2) **TinyRAM** is a random-access machine designed to produce concise assembly code from programs written in high-level languages [82]. The assembly code is converted into an arithmetic circuit.
3) **Buffet/Pequin** is a system for the generation of circuits with easy programmability and data-dependent flows [83]. It is a part of the Pepper/Pequin project implementation (see Section XI). The system attempts to generate efficient circuits from a program written in a subset of the C language.
4) **Geppetto** [84] is a compiler that transforms LLVM code into a QAP. It has a library for low-level control of the circuits, as well as high-level C libraries for the optimization of several programming patterns such as loops.
5) **xJsnark** framework [85] is a compiler for automatic transformation of programs into optimized arithmetic circuits. A developer can write a program in a Java-like language. Experimental evaluation of the performance can be found in [85]. Code is available in Github.[1]
6) **ZoKrates** [76] is a toolbox for zero-knowledge off-chain computations on the Ethereum blockchain. Code written in a custom high-level imperative language is compiled into a Rank-1 Constraint System (R1CS) used by some schemes. A proof-of-concept implementation is available.[2]
7) **Isekai** is a tool for the automatic generation of Boolean and arithmetic circuits from C/C++ code. It can produce circuits for most of the contemporary zero-knowledge argument schemes. The framework is available in Github.[3]
8) **genSTARK** is a library for the generation of zero-knowledge proofs using the STARK scheme (see Section VI) based on JavaScript. Functionality can be implemented in a custom language called AirScript which can be compiled into a constraint problem especially intended for the STARK scheme. The library is available in Github.[4]

[1]https://github.com/akosba/xjsnark
[2]https://github.com/Zokrates/ZoKrates
[3] https://github.com/sikoba/isekai
[4]https://github.com/GuildOfWeavers/genSTARK

B. LOW-LEVEL TOOLS

Low-level circuit construction tools are often needed in scenarios where the performance of the zero-knowledge argument scheme is critical. Compared to the high-level compilers, these tools require more effort and skill from the developer, but produce circuits that are typically much smaller.

1) **libSNARK** is a full C++ library for SNARKs. It implements both circuit construction tools and general purpose proof systems. The circuit building modules, called Gadget Libraries, can be applied to construct a R1CS or systems of polynomial equations for zero-knowledge protocols. The zero-knowledge functionality of libSNARK is described in Section XI. The library is available in Github.[5]
2) **jsnark** is a Java-based library for constructing circuits. Gadgets can be combined to express the functionality of the computation. Circuits produced by the Pinocchio compiler can be also integrated. Code is available in Github.[6]
3) **Bellman** is a Rust-based library for the formulation of constraint systems. Currently, it is under refactorization into a generic proving library.
4) **snarky** is an OCaml front-end for the implementation of circuits for SNARKs based on R1CS. The tool is available in Github.[7]
5) **Circom** is a language for the generation of arithmetic circuits. A javascript-like language is used to express the functionality of the circuit. The tool is available in Github.[8]
6) **gnark** is an open source library for writing circuits for zero-knowledge argument protocols. The library is developed in the Go language. Currently, only Groth16 (see Section VIII) protocol is supported. The source code is available in Github.[9]

VI. ZERO-KNOWLEDGE BASED ON PROBABILISTICALLY CHECKABLE PROOFS

Kilian [86] constructed the first interactive zero-knowledge argument scheme for NP that achieves polylogarithmic communication. The scheme is built using collision-resistant hash functions, interactive proof systems (IP) [25], [87] and probabilistically checkable proofs (PCP) [88]–[91]. The prover and the verifier (considered as randomized algorithms) interact in multiple rounds of communication during which the verifier tests the knowledge of the prover regarding a statement. Usually, only one sided error is considered: a prover is always able to argue for a true statement, but the verifier may accept a false statement with low probability. Micali transformed the scheme into a one-message non-interactive scheme using the Fiat-Shamir heuristic [92] (see Section III-A). Subsequent

[5]https://github.com/scipr-lab/libsnark
[6]https://github.com/akosba/jsnark
[7]https://github.com/o1-labs/snarky
[8]https://github.com/iden3/circom
[9]https://github.com/consensys/gnark

works have applied Kilian's machinery in the construction of schemes using extractable collision-resistant hash functions [38], [93].

The following implementations can be considered to follow this approach.

1) **Scalable Computational Integrity (SCI)** [94] is an interactive computational integrity protocol based on PCPs and follows the original suggestion of Kilian [86] and methods of Ben-Sasson *et al.* [95], [96]. The protocol provides publicly verifiable proofs without trusted setup. However, complete zero-knowlegde is not provided. The time and memory complexities of the prover are quasilinear in the program execution length. The proofs are succinct, but slow to verify. SCI has been implemented in practice and measurements on its performance can be found in [94].
2) **STARK** is an improvement on SCI [97]. The scheme is based on interactive oracle proofs (IOP) with public randomness, but can be made non-interactive with the Fiat-Shamir paradigm in the random oracle model. The construction is post-quantum secure. However, the security depends on a non-standard cryptographic assumption regarding Reed-Solomon codes. Asymptotically, the proof length is $O(\log^2 |C|)$ and verification runs in $O(|C|)$ time. Proving is more costly: $O(|C| \log^2 |C|)$. STARK has an implementation in C++[10] and an evaluation of its performance can be found in [97]. There are also open-source implementations called Hodor[11] and OpenZKP[12] both written in Rust, as well as the JavaScript library genSTARK.
3) **Aurora** is a succinct non-interactive argument (SNARG) based on STARK [98]. It is an argument system specially designed for rank-1 constraint system (R1CS). Non-interactivity is based on the Fiat-Shamir construction. The design does not use asymmetric cryptography and can thus be considered post-quantum secure. The proof length is $O(\log^2 |C|)$, can be verified in $O(|C|)$ and proving runs in $O(|C| \log |C|)$. The Aurora protocol has been implemented in **libiop** available in Github.[13] AuroraLight [99] applies the methodology of Aurora to improve Sonic (see Section VIII). However, AuroraLight has not yet gone through peer-review.
4) **Fractal** [100] is a pre-processing SNARK using R1CS and based on IOP. Proofs can be recursively composed and the setup uses only public randomness. The method can be considered post-quantum secure in the random oracle model. The proof length is $O(\log^2 |C|)$. Proving takes $O(|C| \log |C|)$ and verification $O(\log |C|)$. An implementation is included in **libiop**.

[10]https://github.com/elibensasson/libSTARK
[11]https://github.com/matter-labs/hodor
[12]https://github.com/0xProject/OpenZKP
[13]https://github.com/scipr-lab/libiop

VII. ZERO-KNOWLEDGE USING PROOFS FOR MUGGLES

Contrary to the setting of computational assumptions considered by Kilian [86], Goldwasser *et al.* [101] addressed the problem of constructing interactive proofs for polynomial time provers in the original interactive proof model. They presented a novel approach called "proofs-for-muggles" that works for a large class of problems. These proofs can be made non-interactive using the transformation of Kalai *et al.* [36] (see Section III-A). For details, see for example [101].

1) **CMT**. The first practical implementation of the "proofs-for-muggles" approach is due to Cormode *et al.* [102] who applied the machinery developed by Goldwasser *et al.* [101]. The practical performance of CMT can be found in [102].
2) **Hyrax** [103] is a zero-knowledge argument system without trusted setup based on interactive proofs and cryptographic commitment schemes [104], [105]. Non-interactivity is provided using the Fiat-Shamir heuristic. The construction applies the discrete logarithm problem and is not post-quantum secure. Hyrax has been implemented based on the compiler of Giraffe[14] [106] and the full source code is freely available.[15]
3) **Libra** [107] is zero-knowledge proof system based on the "proofs-for-muggles" approach that achieves optimal prover complexity of $O(|C|)$. The proof length and verification complexity are both $O(d \log |C|)$, where d is the depth of the circuit. A trusted setup is needed and its complexity depends on the input to the circuit. The scheme applies bilinear pairing and the knowledge-of-exponent assumption and is not post-quantum secure. Non-interactivity can be implemented based on the Fiat-Shamir method.
4) **Spartan** [108] applies polynomial commitments to achieve improvement in verification complexity. Computations are presented as R1CS and zero-knowledge is based on existing compilers such as Libra [107]. However, no trusted setup is needed. Non-interactivity is implemented using the Fiat-Shamir transform. The prover complexity is $O(|C| \log |C|)$ while both the proof length and the verifier complexity are $O(\log^2 |C|)$.

VIII. SCHEMES BASED ON LINEAR PCPs AND THE DISCRETE LOGARITHM PROBLEM

Ishai *et al.* [109] suggested the possibility of applying additively homomorphic public-key cryptography to reduce the communication complexity of interactive linear PCPs. Recent results for the complexity of fully linear PCPs can be found in [110]. Groth *et al.* suggested the first NIZK schemes based on the discrete logarithm problem [111]–[113] and bilinear pairings that achieves perfect completeness, computational soundness and perfect zero-knowledge [114]. Similarly to

[14]https://github.com/pepper-project/giraffe
[15]Hyrax reference implementation: https://github.com/hyraxZK

the PCP technique, the general construction at first expresses the statement as an algebraic constraint satisfaction problem. However, cryptographic commitments similar to the Pedersen commitment scheme [48] are applied to achieve sub-linear proof length and non-interactivity without the Fiat-Shamir heuristic. However, a large CRS is required and the protocol depends on a strong cryptographic assumption known as the ''knowledge-of-exponent''.

With a sufficiently long CRS, a constant proof length can be achieved [114] ultimately consisting of only three underlying group elements [115], [116]. Proving and verification are costly due to exponentiations. For additional application scenarios, a stronger security model called simulation-extractability has been suggested [117].

The following schemes are based on linear PCPs and/or the discrete logarithm problem. None of these are post-quantum secure.

1) **Groth's Linear SNARK**. The seminal work of Groth [113] achieves communication complexity that is proportional to the square root of the size of the circuit.
2) **BCCGP**. Bootle *et al.* [118] implemented the first non-interactive zero-knowledge proof protocol based on the discrete logarithm problem and using the techniques of Groth [113], [119].
3) **Bulletproofs** [56] is based on the techniques of BCCGP [118]. The protocol is designed to provide communication-efficient proofs especially for confidential transactions through range proofs which can be efficiently aggregated. However, any NP-problem is supported and the protocol does not require trusted setup. The protocol can be extended to achieve post-quantum security and non-interactivity is provided through the Fiat-Shamir heuristic. The length of the proof is logarithmic in the number of multiplication gates of the verification circuit. The prover and verifier complexities are linear in the size of the witness w. Bulletproofs has a full implementation using arithmetic circuits.
4) **Groth16** applies elliptic curve pairings and the knowledge-of-exponent assumption [114]. The method achieves a constant proof length [116], but requires pre-processing for the CRS with length quadratic in $|C|$. The verifier complexity is linear, while the prover complexity is quadratic in $|C|$. The resulting scheme satisfies perfect completeness and zero-knowledge with computational soundness.
5) **Sonic** [120] is a zero-knowledge scheme based on the updatable CRS model of Groth [117] using a polynomial commitment scheme, pairings and arithmetic circuits. Trusted setup for the CRS is needed and can be implemented, for example, with secure multiparty computation. However, once the CRS has been generated, any circuit of a given size is supported. The prover complexity is $O(|C| \log |C|)$, while verification is $O(N)$ in the instance input length N. There is an open source implementation in Rust.[16] PLONK reduces the proof length of Sonic and improves the efficiency of the prover [121]. However, it has not yet gone through peer-review.
6) **Marlin** [122] is another method based on Sonic. The CRS is updatable and universal. Verification is improved using special encoding for the statement. Non-interactivity is achieved with the Fiat-Shamir paradihm. The algorithm has been implemented in Rust and the source code is available.[17]
7) **Supersonic** [123] applies a novel polynomial commitment scheme to turn Sonic into a zero-knowledge scheme that does not require trusted setup. The proof length and the verification complexity are $O(\log |C|)$. The prover complexity is $O(|C| \log^2 |C|)$. The construction is not post-quantum secure.

IX. ZERO-KNOWLEDGE BASED ON QUADRATIC ARITHMETIC PROGRAMS

Gennaro *et al.* [124] formulated classes of algebraic satisfaction problems called quadratic span programs (QSP) for proving boolean circuit satisfiability and QAP for proving arithmetic circuit satisfiability with more efficient and simpler proof checking. QAP-based constructions are able to achieve constant length proofs and the verification is fast: linear in the length N of the input. A CRS is needed. In addition, the length of the CRS grows in the number of gates. The prover complexity is typically linear in $|C|$.

Existing suggestions related to this approach include the following ones.

1) **GGPR**. Gennaro *et al.* pioneered the quadratic program based approach in [124]. It was the first construction to compactly encode computations as QSP or QAP.
2) **Pinocchio** is a complete suite for zero-knowledge. Its zero-knowledge arguments are based on the GGPR using QAP and designed for verifiable computing [64], [81]. Computations can be verified publicly; an untrusted computer can generate proofs that anybody can check for the correctness of computation. The setup and proof generation are $O(|C|)$. Verification is linear time in the lengths of the inputs and outputs and the proof length is constant. A compiler is provided (See Section V) and can be used to transform a C language program into a QAP.

X. ZERO-KNOWLEDGE FROM SECURE MULTIPARTY COMPUTATION

Zero-knowledge protocols are a special case of secure multiparty computation (MPC) with two participants, where Alice and Bob hold secret values a and b, respectively, and want to compute the output $c = f(a, b)$ of a function f while keeping their inputs private. The problem of secure two party computation was originally suggested by Yao [125] who

[16]https://github.com/zknuckles/sonic
[17]https://github.com/scipr-lab/marlin

showed how to solve it using the so called ''garbled circuits'' approach, which has been applied in many zero-knowledge argument schemes.

Ishai *et al.* [126] pioneered the IKOS approach (initials of the authors) of constructing zero-knowledge arguments using secure MPC together with PCPs and cryptographic commitment. The prover simulates secure MPC ''in its head'' for the verification function between n parties. The verifier gets to choose a subset of these verifiers to check the consistency of their state. Another approach applying the MPC is due to Jawurek *et al.* [127] who implemented zero-knowledeg by transforming statements into garbled circuits.

Proofs generated following the MPC-in-the-head paradigm are large and verification is costly. However, the approach is Boolean circuit friendly and the prover complexity is relatively low. In addition, the approach provides post-quantum security. Compared to the MPC-in-the-head, the Jawurek *et al.* approach has more efficient proving, but the proofs are larger and non-interactivity is hard to achieve.

The following schemes apply multiparty computation.

1) **ZKGC** [127] is an interactive zero-knowledge argument protocol based on Yao's garbled circuits. The construction applies secure MPC. Oblivious transfer with commitment properties is used by the prover to commit to the circuit output. The scheme has a proof-of-concept implementation that has been evaluated in [127].
2) **ZKBoo** [128] is based on the ''MPC-in-the-head'' approach and works for both Boolean or arithmetic circuits. No pre-processing is required and non-interactivity is realized using the Fiat-Shamir method. Proving and verifying of statements is fast. However, the proof length is $O(|C|)$ and thus large. The protocol has a a proof-of-concept implementation.
3) **ZKB++** [129] is an optimized variant of ZKBoo [128]. The proof length is reduced to half compared to ZKBoo. ZKB++ has been evaluated for both the Fiat-Shamir heuristic and a modified version of the Unruh transformation (see Section III-A), which has larger proofs. The protocol has a practical implementation and its performance evaluation can be found in [129].
4) **Ligero** [130] is an interactive zero-knowledge protocol for any NP-problem without trusted setup. Non-interactivity can be realized with the Fiat-Shamir method. The protocol is based on symmetric cryptography, MPC [131] and PCPs following the work of Ishai *et al.* [126]. Public-key cryptography is not used and the scheme can be considered post-quantum secure. The asymptotic proof length is proportional to the square-root of the verification circuit size. The prover and verifier complexities are $O(|C| \log |C|)$. If uniformity assumptions can be made regarding the circuit, the verifier complexity can be made linear. There is a C++ implementation and its practical performance can be found in [130]. Ligero is also included in **libiop** library (see Section XI).

XI. GENERAL PURPOSE IMPLEMENTATIONS FOR ZERO-KNOWLEDGE

General purpose implementations provide several approaches to zero-knowledge arguments and combine those with circuit generation tools. The goal is to provide a library of schemes such that one of them can be picked, potentially with automatic optimization, depending on the use case. The following libraries have implemented several zero-knowledge argument schemes following the approaches mentioned above.

1) **libSNARK** is a C++ library that implements several general purpose proof systems. Currently, seven constructions are supported: 1) An extension to the zero-knowledge argument of Ben-Sasson *et al.* [132] which follows the same approach as Pinocchio [64], [81] using R1CS, 2) SNARKs for problems expressed as arithmetic circuits, 3) SNARKs for problems expressible as unitary-square constraint systems [133], 4) SNARKs for Boolean circuits for the language of two-input boolean circuit satisfiability, 5) simulation-extractable SNARKs based on the method of Groth and Maller [117], 6) ADSNARK [134] for efficient SNARKs on authenticated data from a trusted source, and 7) method for proof-carrying data (PCD) for recursive composition of SNARKs to extend SNARKs to the setting of distributed networks of verifiers and provers based on the method of Bitansky *et al.* [135].
2) **Pepper/Pequin**. Pequin, previously known as Pepper, is a complete tool chain for verifiable computing and zero-knowledge arguments. LibSNARK is used to implement the SNARK functionality. Giraffe can be used to transfer computations into constraint problems. Source code is freely available.[18]
3) **libiop** is a C++ library for zero-knowledge arguments that apply interactive oracle proofs. It includes implementations for Ligero, Aurora and Fractal and provides a tool chain for the transformation of probabilistic interactive proofs into transparent and quantum-secure zero-knowledge arguments. Computations are expressed as R1CS. The library is available on Github.[19]

XII. DISCUSSION ON THEORETICAL AND PRACTICAL PERFORMANCE

Typically, the performance of algorithms is measured by their asymptotic complexities. The asymptotic complexity gives a comparative measure for the efficiency of algorithms when the input is large. However, the asymptotic notation hides the constant terms which determine the performance for small inputs. For existing applications of zero-knowledge schemes,

[18]https://github.com/pepper-project/pequin
[19]https://github.com/scipr-lab/libiop

TABLE 3. The reported asymptotics of the surveyed schemes including pre-processing for CRS generation, proof length and prover and verifier complexities. In the table, C is the computation expressed as a circuit, $|C|$ is the number of gates in C, G is the width of C, d is the depth of C, x is the input, w is the witness, $s(|x|)$ is the amount of memory taken by the computation, $\mathbb{F}$ is the underlying finite field, N is the length of the inputs and outputs of the computation, M is the number of multiplication (AND) gates in the circuit.

Scheme	Preprocessing	Communication / Proof Length	Prover	Verifier	Practical Evaluation
PCP					
SCI	No	$O(\text{polylog }\lvert C\rvert)$	$O(\lvert C\rvert \cdot \text{polylog }\lvert C\rvert)$	$O(\text{polylog }\lvert C\rvert)$	[94]
STARK	No	$O(\log^2 \lvert C\rvert)$	$O(\lvert C\rvert \cdot \log^2 \lvert C\rvert)$	$O(\lvert C\rvert)$	[97], [103]
Aurora	No	$O(\log^2 \lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\lvert C\rvert)$	[98]
Fractal	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(N + \log \lvert C\rvert)$	N/A
Muggles					
CMT	$O(\text{poly}(\lvert C\rvert))$	$O(d \cdot \log \lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\lvert x\rvert \cdot s(\lvert x\rvert) \cdot \log \lvert x\rvert + d \cdot \text{polylog }\lvert C\rvert)$	[102]
Hyrax $(i \geq 2)$	No	$O(d \cdot \log G + \lvert w\rvert^{1/i})$	$O(\lvert C\rvert)$	$O(\lvert x\rvert + d \log G + \lvert w\rvert^{(i-1)/i})$	[103]
Libra	$O(\lvert x\rvert)$	$O(d \log \lvert C\rvert)$	$O(\lvert C\rvert)$	$O(d \log \lvert C\rvert)$	[107]
Spartan	$O(\lvert C\rvert)$ *	$O(\log^2 \lvert C\rvert)$	$O(\lvert C\rvert \log \lvert C\rvert)$	$O(\log^2 \lvert C\rvert)$	[108]
DLP					
Groth's Linear	$O(1)$	$O(\sqrt{\lvert M\rvert})$	$O\left(\frac{M}{\log M}\right)$	$O\left(\frac{\sqrt{M}}{\log M}\right)$	N/A
BCCGP	$O(1)$	$O(\log(M))$	$O(M)$	$O(M)$	[118], [103]
Bulletproofs	$O(1)$	$O(\log(M))$	$O(M)$	$O(M)$	[56], [103]
Groth16	$O(\lvert C\rvert^2)$	$O(1)$	$O(\lvert C\rvert^2)$	$O(\lvert C\rvert)$	N/A
Sonic	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(1)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(N + \log \lvert C\rvert)$	[120]
Marlin	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(N + \log \lvert C\rvert)$	[122]
Supersonic	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\log \lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\log \lvert C\rvert)$	N/A
QAP					
GGPR	$O(\lvert C\rvert)$	$O(1)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(N)$	N/A
Pinocchio	$O(\lvert C\rvert)$	$O(1)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(N)$	[81], [118], [128]
MPC					
ZKGC	No	$O(\lvert w\rvert + \lvert C\rvert)$	$O(\lvert w\rvert + \lvert C\rvert)$	$O(\lvert w\rvert + M)$	[127], [128]
ZKBoo	No	$O(\lvert C\rvert)$	$O(\lvert C\rvert)$	$O(M)$	[128]
ZKB++	No	$O(\lvert C\rvert)$	$O(\lvert C\rvert)$	$O(M)$	[129], [130], [103]
Ligero	No	$O(\sqrt{\lvert C\rvert})$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	$O(\lvert C\rvert \cdot \log \lvert C\rvert)$	[130], [103]

* one time public computation by the verifier, not repeated for different inputs

the circuits are relatively small and both the asymptotic complexities and the constant terms are important.

The asymptotic performance of the surveyed methods has been collected into Table 3. These asymptotics are based on the reported values in the cited publications. For some schemes the complexities were not available and those have been evaluated by the authors of this survey and should be interpreted as approximations. For ZKGC [127], the asymptotic complexities have been evaluated for the reported proof-of-concept implementation which is not optimal. We have also provided references to the practical evaluations of the methods whenever such evaluations have been available. Typically, these evaluations are based on a simple computation such as the computation of a cryptographic hash function. It should be noted that the performance is largely affected by the size of the generated circuit which depends on the circuit generation tool. However, the circuit is not fixed across all existing comparisons in the literature and cannot be fixed for schemes following a different approach (for example Boolean vs. arithmetic vs. R1CS). A comprehensive evaluation with several optimized circuits for different types of computations using a fixed computational platform would be useful for comparison. However, such an evaluation does not seem to exist and is out of the scope of this survey.

There are other properties in addition to the computational complexity that are relevant for the practical applicability of the scheme. For example, some methods require trusted setup or its implementation using secure multiparty computation or another complex pre-processing phase, while others do not. The type and size of the CRS and the required randomness also vary and the cryptographic security model is not the same for all methods. These performance affecting properties have been collected into Table 4 together with the post-quantum security status and the dependence on the random oracle model.

The selection of the most efficient scheme for a specific task requires skill and knowledge from the developer. Currently there is not a single zero-knowledge argument scheme that outperforms others in every situation. At least the following aspects affect the performance: 1) the nature of the computation or statement, 2) the type of the computation (is it sequential in nature or highly parallel), 3) the input size of the computation, 4) the optimization aspects of the scenario, that is, if proving and verification complexities or proof length should be optimized and 4) the level of security required for the application. These issues need to be evaluated based on the use case.

Regarding the nature of the computation, Σ-protocols are sufficient and more efficient than SNARKs for algebraic statements such as those involving the demonstration of the knowledge of a discrete logarithm a given a public key g^a. If the computation involves the computation of block ciphers and/or hash functions then SNARKs outperform Σ-protocols [136]. The type of the computation also has a profound effect. For example, Hyrax has been optimized for parallel circuits while Ligero works better for iterated computations [97]. For practical cases, QAP tends to outperform QSP [124]. The size of the computation determines whether the asymptotics or the constant terms are relevant regarding performance. For current zero-knowledge argument schemes, practical computations can be performed only for relatively small circuits. In fact, many optimizations suggested in the literature are based on lowering the constant terms of the existing suggestions. For example, ZKBoo and ZKB++ have exactly the same asymptotic complexities, but ZKB++ has a proof length that is only half of that of ZKBoo.

For many blockchain applications, the proof length may be the deciding factor meaning that methods with constant length proofs, such as Groth16, may be optimal. However, those schemes incur a significant pre-processing penalty and have a big CRS that needs to be managed. For a developer, a collection of schemes may prove to be optimal. Automatic tools that choose the best option depending on the computation will help the developer to choose an optimal approach.

TABLE 4. The type of randomness and/or the length and type of the common reference string (CRS), post-quantum security (PQ) and the dependence on the random oracle model (ROM) for the surveyed zero-knowledge argument schemes.

Scheme	Randomness / CRS	PQ	ROM
PCP			
SCI	transparent	Yes	Yes
STARK	transparent	Yes	Yes
Aurora	transparent	Yes	Yes
Fractal	$O(1)$, transparent, universal	Yes	Yes
Muggles			
CMT	transparent	Yes	No
Hyrax	$O(\sqrt{\lvert w \rvert})$, transparent	No	Yes
Libra	trusted, trapdoor, universal	No	Yes
Spartan	transparent, universal	Yes*	Yes
DLP			
Groth's Linear	transparent	No	Yes
BCCGP	transparent	No	Yes
Bulletproofs	$O(\lvert C \rvert)$, transparent	No	Yes
Groth16	$O(\lvert C \rvert^2)$, trapdoor	No	No
Sonic	$O(\lvert C \rvert)$, trusted, trapd., univ.	No	No
Marlin	$O(\lvert C \rvert)$, trapdoor, universal	No	Yes
Supersonic	$O(1)$, transparent, universal	No	Yes
QAP			
GGPR	$O(\lvert C \rvert)$, trapdoor	No	No
Pinocchio	$O(\lvert C \rvert)$, trapdoor	No	No
MPC			
ZKGC	non-transparent	No	Yes
ZKBoo	transparent	Yes	Yes
ZKB++	transparent	Yes	Yes
Ligero	transparent	Yes	Yes

* depends on the variant, see [108]

XIII. CONCLUSION

We survey the state-of-the-art zero-knowledge argument schemes suitable for blockchain and verifiable computing and their applications in anonymous and confidential transactions and private smart contracts. We explain the concept of a zero-knowledge proof and its non-interactive variants. In order to prove in zero-knowledge, the developer needs to transform the corresponding computation into a circuit representation. Therefore, we also survey the existing circuit generation tools and provide references to their implementations.

We give an overview of the protocols applying zero-knowledge for confidential transactions and private smart contracts on blockchain. We summarize their differences regarding the blockchain model and type, regulation capability and the protection of the transaction graph and amounts. For private smart contracts, we consider the privacy of data, function privacy and the supported functionality.

We conduct an extensive survey of state-of-the-art zero-knowledge argument schemes. We list the properties, asymptotic computational complexities and proof lengths of the zero-knowledge argument schemes that are peer-reviewed, seminal or otherwise notable and/or have implementations. We also provide references to their practical evaluations and comparisons. Finally, we conclude that the schemes and approaches have their own advantages and disadvantages and that skill and knowledge is needed from the developer to choose the correct one for a particular use case.

DECLARATION OF INTERESTS

Declarations of interest: none

REFERENCES

[1] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.
[2] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *Proc. 19th Int. Conf. Adv. Commun. Technol. (ICACT)*, 2017, pp. 464–467.
[3] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Gener. Comput. Syst.*, vol. 82, pp. 395–411, May 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X17315765
[4] N. Szabo, "Formalizing and securing relationships on public networks," *1st Monday*, vol. 2, no. 9, Sep. 1997. [Online]. Available: https://journals.uic.edu/ojs/index.php/fm/article/view/548
[5] W. Cai, Z. Wang, J. Ernst, Z. Hong, C. Feng, and V. Leung, "Decentralized applications: The blockchain-empowered software system," *IEEE Access*, vol. 6, pp. 53019–53033, 2018.
[6] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A fistful of bitcoins: Characterizing payments among men with no names," in *Proc. Conf. Internet Meas. Conf.*, New York, NY, USA, 2013, pp. 127–140.
[7] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 839–858.
[8] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in bitcoin P2P network," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2014, pp. 15–29.
[9] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research perspectives and challenges for bitcoin and cryptocurrencies," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 104–121.
[10] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2084–2123, 3rd Quart., 2016.
[11] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where is current research on blockchain technology?—A systematic review," *PLoS ONE*, vol. 11, no. 10, Oct. 2016, Art. no. e0163477.
[12] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.
[13] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Gener. Comput. Syst.*, vol. 107, pp. 841–853, Jun. 2020.
[14] A. Lazarenko and S. Avdoshin, "Financial risks of the blockchain industry: A survey of cyberattacks," *Adv. Intell. Syst. Comput.*, vol. 881, pp. 368–384, Nov. 2019.
[15] M. Conti, E. Sandeep Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3416–3452, 4th Quart., 2018.
[16] D. Genkin, D. Papadopoulos, and C. Papamanthou, "Privacy in decentralized cryptocurrencies," *Commun. ACM*, vol. 61, no. 6, pp. 78–88, May 2018, doi: 10.1145/3132696.
[17] M. Can Kus Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2543–2585, 3rd Quart., 2018.
[18] L. Peng, W. Feng, Z. Yan, Y. Li, X. Zhou, and S. Shimizu, "Privacy preservation in permissionless blockchain: A survey," *Digit. Commun. Netw.*, Dec. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352864819303827
[19] E. Morais, T. Koens, C. van Wijk, and A. Koren, "A survey on zero knowledge range proofs and applications," *Social Netw. Appl. Sci.*, vol. 1, p. 946, Aug. 2019.
[20] A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, "Blockchain and IoT integration: A systematic survey," *Sensors*, vol. 18, no. 8, p. 2575, 2018. [Online]. Available: http://www.mdpi.com/1424-8220/18/8/2575
[21] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with IoT. Challenges and opportunities," *Future Gener. Comput. Syst.*, vol. 88, pp. 173–190, Nov. 2018.
[22] E. Jesus, V. Chicarino, C. De Albuquerque, and A. Rocha, "A survey of how to use blockchain to secure Internet of Things and the stalker attack," *Secur. Commun. Netw.*, vol. 2018, Apr. 2018, Art. no. 9675050.
[23] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," *J. Netw. Comput. Appl.*, vol. 126, pp. 45–58, Jan. 2019.
[24] L. Herskind, P. Katsikouli, and N. Dragoni, "Privacy and cryptocurrencies—A systematic literature review," *IEEE Access*, vol. 8, pp. 54044–54059, 2020.
[25] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM J. Comput.*, vol. 18, no. 1, pp. 186–208, Feb. 1989.
[26] G. Brassard, D. Chaum, and C. Crépeau, "Minimum disclosure proofs of knowledge," *J. Comput. Syst. Sci.*, vol. 37, no. 2, pp. 156–189, 1988. [Online]. Available: http://www.sciencedirect.com/science/article/pisi/0022000088900050
[27] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems," *J. ACM*, vol. 38, no. 3, pp. 690–728, Jul. 1991.
[28] O. Goldreich and Y. Oren, "Definitions and properties of zero-knowledge proof systems," *J. Cryptol.*, vol. 7, no. 1, pp. 1–32, Dec. 1994.
[29] O. Goldreich and H. Krawczyk, "On the composition of zero-knowledge proof systems," *SIAM J. Comput.*, vol. 25, no. 1, pp. 169–192, 1996.
[30] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications," in *Proc. 20th Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 1988, pp. 103–112.
[31] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology*, A. M. Odlyzko, Ed. Berlin, Germany: Springer, 1987, pp. 186–194.
[32] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in *Advances in Cryptology*, U. Maurer, Ed. Berlin, Germany: Springer, 1996, pp. 387–398.
[33] R. Pass, "On deniability in the common reference string and random oracle model," in *Advances in Cryptology*, D. Boneh, Ed. Berlin, Germany: Springer, 2003, pp. 316–337.
[34] D. Unruh, "Non-interactive zero-knowledge proofs in the quantum random oracle model," in *Advances in Cryptology*, E. Oswald and M. Fischlin, Eds. Berlin, Germany: Springer, 2015, pp. 755–784.
[35] A. Ambainis, A. Rosmanis, and D. Unruh, "Quantum attacks on classical proof systems: The hardness of quantum rewinding," in *Proc. IEEE 55th Annu. Symp. Found. Comput. Sci.*, Oct. 2014, pp. 474–483.
[36] Y. T. Kalai, R. Raz, and R. D. Rothblum, "How to delegate computations: The power of no-signaling proofs," in *Proc. Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2014, pp. 485–494.

[37] W. Aiello, S. Bhatt, R. Ostrovsky, and S. R. Rajagopalan, "Fast verification of any remote procedure call: Short witness-indistinguishable one-round proofs for np," in *Automata, Languages and Programming*, U. Montanari, J. D. P. Rolim, and E. Welzl, Eds. Berlin, Germany: Springer, 2000, pp. 463–474.

[38] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer, "From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again," in *Proc. 3rd Innov. Theor. Comput. Sci. Conf.*, New York, NY, USA, 2012, pp. 326–349.

[39] E. Ben-Sasson, A. Chiesa, M. Green, E. Tromer, and M. Virza, "Secure sampling of public parameters for succinct zero knowledge proofs," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 287–304.

[40] J. Groth, M. Kohlweiss, M. Maller, S. Meiklejohn, and I. Miers, "Updatable and universal common reference strings with applications to zk-SNARKs," in *Advances in Cryptology*, H. Shacham and A. Boldyreva, Eds. Cham, Switzerland: Springer, 2018, pp. 698–728.

[41] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Advances in Cryptology*, T. Rabin, Ed. Berlin, Germany: Springer, 2010, pp. 465–482.

[42] M. Campanelli, R. Gennaro, S. Goldfeder, and L. Nizzardo, "Zero-knowledge contingent payments revisited: Attacks and payments for services," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2017, pp. 229–243.

[43] G. G. Dagher, B. Bánz, J. Bonneau, J. Clark, and D. Boneh, "Provisions: Privacy-preserving proofs of solvency for bitcoin exchanges," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2015, pp. 720–731.

[44] T. Hardjono and N. Smith, "Cloud-based commissioning of constrained devices using permissioned blockchains," in *Proc. 2nd ACM Int. Workshop IoT Privacy, Trust, Secur.*, New York, NY, USA, 2016, pp. 29–36.

[45] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *Financial Cryptography Data Security*, A. Kiayias, Ed. Cham, Switzerland: Springer, 2017, pp. 357–375.

[46] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, and S. Goldberg. (2016). *Tumblebit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub*. [Online]. Available: https://eprint.iacr.org/2016/575

[47] M. Möser and R. Böhme, "The price of anonymity: Empirical evidence from a market for bitcoin anonymization," *J. Cybersecur.*, vol. 3, no. 2, pp. 127–135, Jun. 2017.

[48] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology*, J. Feigenbaum, Ed. Berlin, Germany: Springer, 1992, pp. 129–140.

[49] G. Maxwell. (2015). *Confidential Transactions*. Accessed: Mar. 27, 2019. [Online]. Available: https://people.xiph.org/~greg/confidential_values.txt

[50] A. Poelstra, A. Back, M. Friedenbach, G. Maxwell, and P. Wuille, "Confidential assets," in *Proc. Financial Cryptography Bitcoin Workshop*, 2017, pp. 1–5.

[51] T. Ruffing and P. Moreno-Sanchez, "ValueShuffle: Mixing confidential transactions for comprehensive transaction privacy in bitcoin," in *Financial Cryptography Data Security*, M. Brenner, K. Rohloff, J. Bonneau, A. Miller, P. Y. Ryan, V. Teague, A. Bracciali, M. Sala, F. Pintore, and M. Jakobsson, Eds. Cham, Switzerland: Springer, 2017, pp. 133–154.

[52] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Advances in Cryptology*, C. Boyd, Ed. Berlin, Germany: Springer, 2001, pp. 552–565.

[53] Z. Yu, M. H. Au, J. Yu, R. Yang, Q. Xu, and W. F. Lau, "New empirical traceability analysis of CryptoNote-style blockchains," in *Financial Cryptography Data Security*, I. Goldberg and T. Moore, Eds. Cham, Switzerland: Springer, 2019, pp. 133–149.

[54] S. Noether, A. Mackenzie, and T. M. Research Lab, "Ring confidential transactions," *Ledger*, vol. 1, pp. 1–18, Dec. 2016.

[55] A. Kumar, C. Fischer, S. Tople, and P. Saxena, "A traceability analysis of monero's blockchain," in *Computer Security*, S. N. Foley, D. Gollmann, and E. Snekkenes, Eds. Cham, Switzerland: Springer, 2017, pp. 153–173.

[56] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," in *Proc. IEEE Symp. Secur. Privacy (SP)*, Dec. 2018, pp. 319–338.

[57] T. E. Jedusor. (2016). *Mimblewimble*. Accessed: Dec. 18, 2020. [Online]. Available: https://docs.beam.mw/Mimblewimble.pdf

[58] A. Poelstra. (2016). *Mimblewimble*. Accessed: Mar. 27, 2019. [Online]. Available: http://mimblewimble.cash/20161006-WhitePaperUpdate-e9f45ec.pdf

[59] A. Saxena, J. Misra, and A. Dhar, "Increasing anonymity in bitcoin," in *Financial Cryptography Data Security*, R. Böhme, M. Brenner, T. Moore, and M. Smith, Eds. Berlin, Germany: Springer, 2014, pp. 122–139.

[60] G. Fuchsbauer, M. Orrù, and Y. Seurin, "Aggregate cash systems: A cryptographic investigation of mimblewimble," in *Advances in Cryptology*, Y. Ishai and V. Rijmen, Eds. Cham, Switzerland: Springer, 2019, pp. 657–689.

[61] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed E-Cash from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 397–411.

[62] C. P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, 1991.

[63] E. Ben Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, May 2014, pp. 459–474.

[64] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 238–252.

[65] C. Garman, M. Green, and I. Miers, "Accountable privacy for decentralized anonymous payments," in *Financial Cryptography Data Security*, J. Grossklags and B. Preneel, Eds. Berlin, Germany: Springer, 2017, pp. 81–98.

[66] P. Fauzi, S. Meiklejohn, R. Mercer, and C. Orlandi, "Quisquis: A new design for anonymous cryptocurrencies," in *Advances in Cryptology*, S. D. Galbraith and S. Moriai, Eds. Cham, Switzerland: Springer, 2019, pp. 649–678.

[67] C. Lin, D. He, X. Huang, M. Khurram Khan, and K.-K. Raymond Choo, "DCAP: A secure and efficient decentralized conditional anonymous payment system based on blockchain," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2440–2452, 2020.

[68] E. Androulaki, J. Camenisch, A. D. Caro, M. Dubovitskaya, K. Elkhiyaoui, and B. Tackmann, "Privacy-preserving auditable token payments in a permissioned blockchain system," in *Proc. 2nd ACM Conf. Adv. Financial Technol.* New York, NY, USA: Association for Computing Machinery, 2020, pp. 255–267, doi: 10.1145/3419614.3423259.

[69] Z. Guan, Z. Wan, Y. Yang, Y. Zhou, and B. Huang, "Blockmaze: An efficient privacy-preserving account-model blockchain based on zk-SNARKs," *IEEE Trans. Dependable Secure Comput.*, early access, Sep. 18, 2020, doi: 10.1109/TDSC.2020.3025129.

[70] W. Shao, C. Jia, Y. Xu, K. Qiu, Y. Gao, and Y. He, "Attrichain: Decentralized traceable anonymous identities in privacy-preserving permissioned blockchain," *Comput. Secur.*, vol. 99, Dec. 2020, Art. no. 102069. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404820303424

[71] E. Cecchetti, F. Zhang, Y. Ji, A. Kosba, A. Juels, and E. Shi, "Solidus: Confidential distributed ledger transactions via pvorm," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 701–717, doi: 10.1145/3133956.3134010.

[72] Y. Chen, X. Ma, C. Tang, and M. H. Au, "PGC: Decentralized confidential payment system with auditability," in *Computer Security*, L. Chen, N. Li, K. Liang, and S. Schneider, Eds. Cham, Switzerland: Springer, 2020, pp. 591–610.

[73] N. Narula, W. Vasquez, and M. Virza, "Zkledger: Privacy-preserving auditing for distributed ledgers," in *Proc. 15th USENIX Symp. Netw. Syst. Des. Implement. (NSDI)*, S. Banerjee and S. Seshan, Eds. Renton, WA, USA: USENIX Association, Apr. 2018, pp. 65–80. [Online]. Available: https://www.usenix.org/conference/nsdi18/presentation/narula

[74] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. Johnson, A. Juels, A. Miller, and D. Song, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *Proc. IEEE Eur. Symp. Secur. Privacy*, Jun. 2019, pp. 185–200.

[75] S. Steffen, B. Bichsel, M. Gersbach, N. Melchior, P. Tsankov, and M. Vechev, "ZKAY: Specifying and enforcing data privacy in smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2019, pp. 1759–1776, doi: 10.1145/3319535.3363222.

[76] J. Eberhardt and S. Tai, "ZoKrates - scalable privacy-preserving off-chain computations," in *Proc. IEEE Int. Conf. Internet Things*, Dec. 2018, pp. 1084–1091.

[77] B. Bünz, S. Agrawal, M. Zamani, and D. Boneh, "Zether: Towards privacy in a smart contract world," in *Financial Cryptography Data Security*, J. Bonneau and N. Heninger, Eds. Cham, Switzerland: Springer, 2020, pp. 423–443.

[78] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, and H. Wu, "ZEXE: Enabling decentralized private computation," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 947–964.

[79] S. Ma, Y. Deng, D. He, J. Zhang, and X. Xie, "An efficient NIZK scheme for privacy-preserving transactions over account-model blockchain," *IEEE Trans. Dependable Secure Comput.*, early access, Jan. 24, 2020, doi: 10.1109/TDSC.2020.2969418.

[80] C. Lin, D. He, X. Huang, X. Xie, and K.-K.-R. Choo, "PPChain: A privacy-preserving permissioned blockchain architecture for cryptocurrency and other regulated applications," *IEEE Syst. J.*, early access, Sep. 17, 2020, doi: 10.1109/JSYST.2020.3019923.

[81] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," *Commun. ACM*, vol. 59, no. 2, pp. 103–112, 2016.

[82] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "SNARKs for C: Verifying program executions succinctly and in zero knowledge," in *Advances in Cryptology*, R. Canetti and J. A. Garay, Eds. Berlin, Germany: Springer, 2013, pp. 90–108.

[83] R. S. Wahby, S. Setty, Z. Ren, A. J. Blumberg, and M. Walfish, "Efficient RAM and control flow in verifiable outsourced computation," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15.

[84] C. Costello, C. Fournet, J. Howell, M. Kohlweiss, B. Kreuter, M. Naehrig, B. Parno, and S. Zahur, "Geppetto: Versatile verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 253–270.

[85] A. Kosba, C. Papamanthou, and E. Shi, "XJsnark: A framework for efficient verifiable computation," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 543–560.

[86] J. Kilian, "A note on efficient zero-knowledge proofs and arguments (extended abstract)," in *Proc. 24th Annu. ACM Symp. Theory Comput.*, 1992, pp. 723–732.

[87] L. Babai and S. Moran, "Arthur-merlin games: A randomized proof system, and a hierarchy of complexity classes," *J. Comput. Syst. Sci.*, vol. 36, no. 2, pp. 254–276, Apr. 1988.

[88] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, "Proof verification and the hardness of approximation problems," *J. ACM*, vol. 45, no. 3, pp. 501–555, May 1998.

[89] S. Arora and S. Safra, "Probabilistic checking of proofs: A new characterization of NP," *J. ACM*, vol. 45, no. 1, pp. 70–122, Jan. 1998.

[90] L. Babai, L. Fortnow, L. A. Levin, and M. Szegedy, "Checking computations in polylogarithmic time," in *Proc. 23rd Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 1991, pp. 21–32.

[91] L. Babai, L. Fortnow, and C. Lund, "Non-deterministic exponential time has two-prover interactive protocols," *Comput. Complex.*, vol. 1, no. 1, pp. 3–40, Mar. 1991.

[92] S. Micali, "Computationally sound proofs," *SIAM J. Comput.*, vol. 30, no. 4, pp. 1253–1298, 2000.

[93] I. Damgård, S. Faust, and C. Hazay, "Secure two-party computation with low communication," in *Theory Cryptography*, R. Cramer, Ed. Berlin, Germany: Springer, 2012, pp. 54–74.

[94] E. Ben-Sasson, I. Bentov, A. Chiesa, A. Gabizon, D. Genkin, M. Hamilis, E. Pergament, M. Riabzev, M. Silberstein, E. Tromer, and M. Virza, "Computational integrity with a public random string from quasi-linear PCPs," in *Advances in Cryptology*, J.-S. Coron and J. B. Nielsen, Eds. Cham, Switzerland: Springer, 2017, pp. 551–579.

[95] E. Ben-Sasson and M. Sudan, "Short PCPs with polylog Query complexity," *SIAM J. Comput.*, vol. 38, no. 2, pp. 551–607, Jan. 2008.

[96] E. Ben-Sasson, A. Chiesa, D. Genkin, and E. Tromer, "Fast reductions from RAMs to delegatable succinct constraint satisfaction problems: Extended abstract," in *Proc. 4th Conf. Innov. Theor. Comput. Sci.*, New York, NY, USA, 2013, pp. 401–414.

[97] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable zero knowledge with no trusted setup," in *Advances in Cryptology*, A. Boldyreva and D. Micciancio, Eds. Cham, Switzerland: Springer, 2019, pp. 701–732.

[98] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward, "Aurora: Transparent succinct arguments for R1CS," in *Advances in Cryptology*, Y. Ishai and V. Rijmen, Eds. Cham, Switzerland: Springer, 2019, pp. 103–128.

[99] A. Gabizon. (2019). *AuroraLight: Improved Prover Efficiency and SRS Size in a Sonic-Like System*. [Online]. Available: https://eprint.iacr.org/2019/601

[100] A. Chiesa, D. Ojha, and N. Spooner, "Fractal: Post-quantum and transparent recursive proofs from holography," in *Advances in Cryptology*, A. Canteaut and Y. Ishai, Eds. Cham, Switzerland: Springer, 2020, pp. 769–793.

[101] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "Delegating computation: Interactive proofs for muggles," in *Proc. Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2008, pp. 113–122.

[102] G. Cormode, M. Mitzenmacher, and J. Thaler, "Practical verified computation with streaming interactive proofs," in *Proc. 3rd Innov. Theor. Comput. Sci. Conf.*, 2012, pp. 90–112.

[103] R. S. Wahby, I. Tzialla, A. Shelat, J. Thaler, and M. Walfish, "Doubly-efficient zkSNARKs without trusted setup," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 926–943.

[104] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway, "Everything provable is provable in zero-knowledge," in *Advances in Cryptology*, S. Goldwasser, Ed. New York, NY, USA: Springer, 1990, pp. 37–56.

[105] R. Cramer and I. Damgård, "Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free?" in *Advances in Cryptology*, H. Krawczyk, Ed. Berlin, Germany: Springer, 1998, pp. 424–441.

[106] R. S. Wahby, Y. Ji, A. J. Blumberg, A. Shelat, J. Thaler, M. Walfish, and T. Wies, "Full accounting for verifiable outsourcing," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2017, pp. 2071–2086.

[107] T. Xie, J. Zhang, Y. Zhang, C. Papamanthou, and D. Song, "Libra: Succinct zero-knowledge proofs with optimal prover computation," in *Advances in Cryptology*, A. Boldyreva and D. Micciancio, Eds. Cham, Switzerland: Springer, 2019, pp. 733–764.

[108] S. Setty, "Spartan: Efficient and general-purpose zkSNARKs without trusted setup," in *Advances in Cryptology*, D. Micciancio and T. Ristenpart, Eds. Cham, Switzerland: Springer, 2020, pp. 704–737.

[109] Y. Ishai, E. Kushilevitz, and R. Ostrovsky, "Efficient arguments without short PCPs," in *Proc. Annu. IEEE Conf. Comput. Complex. (CCC)*, Jun. 2007, pp. 278–291.

[110] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai, "Zero-knowledge proofs on secret-shared data via fully linear PCPs," in *Advances in Cryptology*, A. Boldyreva and D. Micciancio, Eds. Cham, Switzerland: Springer, 2019, pp. 67–97.

[111] J. Groth, R. Ostrovsky, and A. Sahai, "Non-interactive Zaps and new techniques for NIZK," in *Advances in Cryptology*, C. Dwork, Ed. Berlin, Germany: Springer, 2006, pp. 97–111.

[112] J. Groth and A. Sahai, "Efficient non-interactive proof systems for bilinear groups," in *Advances in Cryptology*, N. Smart, Ed. Berlin, Germany: Springer, 2008, pp. 415–432.

[113] J. Groth, "Linear algebra with sub-linear zero-knowledge arguments," in *Advances in Cryptology*, S. Halevi, Ed. Berlin, Germany: Springer, 2009, pp. 192–208.

[114] J. Groth, "Short pairing-based non-interactive zero-knowledge arguments," in *Advances in Cryptology*, M. Abe, Ed. Berlin, Germany: Springer, 2010, pp. 321–340.

[115] H. Lipmaa, "Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments," in *Theory Cryptography*, R. Cramer, Ed. Berlin, Germany: Springer, 2012, pp. 169–189.

[116] J. Groth, "On the size of pairing-based non-interactive arguments," in *Advances in Cryptology*, M. Fischlin and J.-S. Coron, Eds. Berlin, Germany: Springer, 2016, pp. 305–326.

[117] J. Groth and M. Maller, "Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs," in *Advances in Cryptology*, J. Katz and H. Shacham, Eds. Cham, Switzerland: Springer, 2017, pp. 581–612.

[118] J. Bootle, A. Cerulli, P. Chaidos, J. Groth, and C. Petit, "Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting," in *Advances in Cryptology*, M. Fischlin and J.-S. Coron, Eds. Berlin, Germany: Springer, 2016, pp. 327–357.

[119] J. Groth, "Efficient zero-knowledge arguments from two-tiered homomorphic commitments," in *Advances in Cryptology*, D. H. Lee and X. Wang, Eds. Berlin, Germany: Springer, 2011, pp. 431–448.

[120] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2019, pp. 2111–2128, doi: 10.1145/3319535.3339817.

[121] A. Gabizon, Z. J. Williamson, and O. Ciobotaru. (2019). *PLONK: Permutations Over Lagrange-Bases for Oecumenical Noninteractive Arguments of Knowledge*. [Online]. Available: https://eprint.iacr.org/2019/953

[122] A. Chiesa, Y. Hu, M. Maller, P. Mishra, N. Vesely, and N. Ward, "Marlin: Preprocessing zkSNARKs with universal and updatable SRS," in *Advances in Cryptology*, A. Canteaut and Y. Ishai, Eds. Cham, Switzerland: Springer, 2020, pp. 738–768.

[123] B. Bünz, B. Fisch, and A. Szepieniec, "Transparent SNARKs from DARK compilers," in *Advances in Cryptology*. Berlin, Germany: Springer, 2020.

[124] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic span programs and succinct NIZKs without PCPs," in *Advances in Cryptology*, T. Johansson and P. Q. Nguyen, Eds. Berlin, Germany: Springer, 2013, pp. 626–645.

[125] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 1982, pp. 160–164.

[126] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Zero-knowledge from secure multiparty computation," in *Proc. 29th Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2007, pp. 21–30.

[127] M. Jawurek, F. Kerschbaum, and C. Orlandi, "Zero-knowledge using garbled circuits: How to prove non-algebraic statements efficiently," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2013, pp. 955–966.

[128] I. Giacomelli, J. Madsen, and C. Orlandi, "ZKBoo: Faster zero-knowledge for Boolean circuits," in *Proc. 25th Secur. Symp.*, Austin, TX, USA, 2016, pp. 1069–1083.

[129] M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, and G. Zaverucha, "Post-quantum zero-knowledge and signatures from symmetric-key primitives," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2017, pp. 1825–1842.

[130] S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam, "Ligero: Lightweight sublinear arguments without a trusted setup," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2017, pp. 2087–2104.

[131] I. Damgård and Y. Ishai, "Scalable secure multiparty computation," in *Advances in Cryptology*, C. Dwork, Ed. Berlin, Germany: Springer, 2006, pp. 501–520.

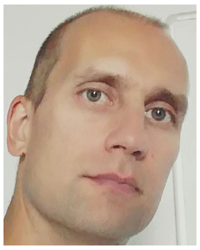
[132] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von Neumann architecture," in *Proc. 23rd Secur. Symp.*, San Diego, CA, USA, 2014, pp. 781–796.

[133] G. Danezis, C. Fournet, J. Groth, and M. Kohlweiss, "Square span programs with applications to succinct NIZK arguments," in *Advances in Cryptology*, P. Sarkar and T. Iwata, Eds. Berlin, Germany: Springer, 2014, pp. 532–550.

[134] M. Backes, M. Barbosa, D. Fiore, and R. M. Reischuk, "ADSNARK: Nearly practical and privacy-preserving proofs on authenticated data," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 271–286.

[135] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer, "Recursive composition and bootstrapping for SNARKS and proof-carrying data," in *Proc. 45th Annu. ACM Symp. Symp. theory Comput.*, New York, NY, USA, 2013, pp. 111–120.

[136] S. Agrawal, C. Ganesh, and P. Mohassel, "Non-interactive zero-knowledge proofs for composite statements," in *Advances in Cryptology*, H. Shacham and A. Boldyreva, Eds. Cham, Switzerland: Springer, 2018, pp. 643–673.



JUHA PARTALA was born in Oulu, Finland, in 1980. He received the M.Sc. and D.Sc. (Tech.) degrees in computer science from the University of Oulu, Oulu, in 2005 and 2015, respectively.

Since 2015, he has been a Postdoctoral Researcher with the Center for Machine Vision and Signal Analysis, Faculty of Information Technology and Electrical Engineering, University of Oulu. He has published in a wide range of topics related to information security, such as theoretical and applied cryptography, steganography, security of biomedical systems, and blockchain. He serves as a reviewer in top journals in information security. His research interests include cryptography, theory of computation, algebra, and security and privacy of biomedical systems.

TRI HONG NGUYEN was born in Ho Chi Minh City, Vietnam, in 1993. He received the B.Sc. degree in computer science from the University of Information Technology–Vietnam National University, Vietnam, in 2015, and the M.Sc. degree in computer science from the University of Pisa, Italy, in 2018. He is currently pursuing the Ph.D. degree with the Center for Ubiquitous Computing, University of Oulu.

His research interests include distributed systems, blockchain technology, and information security.

SUSANNA PIRTTIKANGAS (Member, IEEE) was born in Kemi, Finland, in 1973. She received the M.Sc. degree in theoretical mathematics and the D.Sc. (Tech.) degree in embedded systems from the University of Oulu, in 1998 and 2004, respectively.

She is currently working as a Research Director with the Center for Ubiquitous Computing, University of Oulu. She made her postdoctoral visits to Waseda University, Japan, from 2005 to 2006, Tokyo Denki University, Japan, in 2008, and Tsinghua University, China, in 2011. Her research team Interactive Edge develops adaptive, reliable, and trusted edge computing. She is an Active Member of the international research community as a Workshop and Conference Organizer, as well as serving as a reviewer and a PC member for top journals and conferences in her field. She also works as a freelance lead AI Scientist in a Finnish Company Silo.AI.