UZH Blockchain Center

CfC St. MORITZ

CfC St. Moritz Academic Research Track 2024

Do You Need a Zero Knowledge Proof?

Jens Ernstberger*, Stefanos Chaliasos†, Liyi Zhou‡,
Philipp Jovanovic§, Arthur Gervais¶

Abstract. Zero-Knowledge Proofs (ZKPs), a cryptographic tool known for decades, have gained significant attention in recent years due to advancements that have made them practically applicable in real-world scenarios. ZKPs can provide unique attributes, such as succinctness, non-interactivity, and the ability to prove knowledge without revealing the information itself, making them an attractive solution for a range of applications.

This paper aims to critically analyze the applicability of ZKPs in various scenarios. We categorize ZKPs into distinct types: SNARKs (Succinct Non-Interactive Arguments of Knowledge), Commit-then-Prove ZKPs, MPC-in-the-Head, and Sigma Protocols, each offering different trade-offs and benefits. We introduce a flowchart methodology to assist in determining the most suitable ZKP system, given a set of technical application requirements. Next, we conduct an in-depth investigation of three major use cases: Outsourcing Computation, Digital Self-Sovereign Identity, and ZKPs in networking. Additionally, we provide a high-level overview of other applications of ZKPs, exploring their broader implications and opportunities. This paper aims to demystify the decision-making process involved in choosing the right ZKP system, providing clarity on when and how these cryptographic tools can be effectively utilized in various domains — and when they are better to be avoided.

1 Introduction

In recent years, Zero Knowledge Proofs (ZKPs) emerged as a pivotal tool, commanding the attention of both academia and industry. Initially perceived as a theoretical idea,[1] ZKPs have undergone advancements, rendering them feasible in real-world applications.[2] This paper aims to provide a methodology to analyze which applications should, or maybe should not use ZKPs.

The purpose of this paper is twofold: First, it seeks to aid in the critical thinking process surrounding the application of ZKPs, providing insight for both academic research and industry implementation. Second, it endeavors to bridge the gap between theoretical understanding and practical application of ZKPs. By demystifying the nuanced trade-offs and considerations inherent in choosing a ZKP system, this paper provides a structured approach to determine the most suitable ZKP variant for specific technical requirements and application scenarios.

In doing so, we critically analyze various use-cases where ZKPs hold potential. We introduce

* Jens Ernstberger (jens.ernstberger@tum.de)
† Stefanos Chaliasos (s.chaliasos21@imperial.ac.uk)
‡ Liyi Zhou, (liyi.zhou@imperial.ac.uk)
§ Philipp Jovanovic, (p.jovanovic@ucl.ac.uk)
¶ Arthur Gervais, (arthur@gervais.cc)

a structured methodology, serving as a decision-making tool to guide users in selecting the ZKPs — or a recommendation of not using any ZKP solution. Utilizing this methodology, we conduct an in-depth investigation of three major use cases: Outsourcing Computation, Digital Self-Sovereign Identity, and ZKPs in networking. Additionally, we explore other potential applications of ZKPs, highlighting their broader implications and opportunities in different domains. Our approach is grounded in the recognition that while ZKPs offer robust privacy and verifiability, they may not be a necessity in every application scenario. This paper examines situations where less computationally intensive alternatives might suffice, and contexts where the unique attributes of ZKPs are indeed indispensable. Ultimately, this paper aims to provide clarity and direction in the rapidly developing field of ZKPs, enabling informed decisions in the application of ZKPs in both academia and industry settings.

2 On Argument Systems, SNARKs and Zero-Knowledge

Numerous applications highlight ZKPs as a groundbreaking technology. However, we observe that recent applications misuse the term *zero-knowledge*, such that some applications do not demand *zero knowledge* at all, but rather rely on other properties, such as *non-interactivity* and *succinctness* to facilitate the use-case in question. To clarify the predominating misconception, in majority only identifiable by domain experts, we briefly introduce the notion of *proofs* (Section 2.1), outline the properties that are to be achieved by a proof system (Section 2.2) and introduce common constructions of proof and argument systems applied in practice (Section 2.3).

2.1 What is a Proof?

In the realm of computational theory, a "proof" constitutes a foundational mechanism through which one can assert the veracity of a proposition within a mathematical or computational framework. Conceptually, a proof is a sequence of logical deductions derived from axioms and previously established statements, which culminates in the demonstration of a theorem or claim.

Interactive Proofs. Diverging from this classical view, *interactive proofs* introduce a dynamic discourse between a prover and a verifier; an iterative exchange that permits the verifier to pose random queries and the prover to furnish responses.[3,4] Hence, an interactive proof allows for interaction between the prover and the verifier, as well as a negligible, but non-zero, probability that an invalid proof is correctly verified.

Interactive Proofs' notable power largely stems from two key results in complexity theory.

- IP = PSPACE indicates that Interactive Proofs, involving message exchanges between a prover and verifier, match PSPACE's power—the class of problems solvable with polynomial memory.[5] Notably, PSPACE is believed to exceed NP, making IP more potent than static NP proofs (NP equals deterministic IP, i.e., the verifier holds no randomness).
- The Probabilistically Checkable Proof (PCP) theorem states that any problem that can be verified by a traditional proof can also be verified by only looking at a few bits of a specially constructed proof, randomly chosen.[6]

Interactive Oracle Proofs (IOPs) are a generalization of PCPs and IPs, where the verifier has oracle access to the prover's message.[7]

Arguments of Knowledge. Many academic papers refer to *proofs of knowledge*,[8] where the extension *of knowledge* denotes a requirement that the protocol cryptographically ensures that the prover is in actual possession of the (potentially secret) input to the statement that is proven. An *argument of knowledge*, as opposed to a proof of knowledge, relies on assumptions of computational hardness; it relaxes the inflexibility of proofs by stipulating that certain computational tasks are infeasible for the prover, hence ensuring that the verifier cannot be misled by a computationally bounded prover. This nuanced distinction amplifies the practical relevance of arguments of knowledge, particularly in cryptographic applications where verification efficiency and security against adversaries with bounded computational resources are paramount.

All the above-mentioned classes of proof systems (IP, PCP, IOP) can be compiled to a *succinct* and *non-interactive* argument of knowledge, by utilizing polynomial commitments and further applying the Fiat-Shamir heuristic.[9]

2.2 Properties

Formally, the condition of an interactive proof to be resistant to a computationally restricted, malicious prover is denoted as *computational soundness*. Further, if every satisfying and correctly computed proof leads to an accepting verification, the interactive proof satisfies *perfect completeness*. We informally summarize the essential security properties of an interactive proof:
Completeness: An honest prover can always convince the verifier of the truth of a valid statement.
Soundness: A dishonest prover cannot convince the verifier of the validity of a false statement, except with a negligible probability.

In addition, common zero-knowledge proofs also strive for the following attributes that are essential for their effectiveness and deployment in real-world scenarios:
Succinctness: Proofs should be short when compared to the size of the statement to be proven, and efficient to verify.
Non-Interactivity: The proofs are designed to be non-interactive, allowing them to be sent and verified without ongoing communication between the prover and verifier.
Zero Knowledge: The proofs convey no additional information other than the veracity of the assertion, preserving the confidentiality of the underlying data.

2.3 Proof Systems Applied in Practice

Generally, ZKPs can be constructed for any language, with some optimized for specific languages (e.g., algebraic statements) or certain properties. We discuss high-level trade-offs between different constructions and their applicability scenarios. Note that this overview is non-exhaustive, we focus on proof systems that, to the best of our knowledge, are feasible for practical applications.
SNARKs. Recently, Succinct Non-interactive Arguments of Knowledge (SNARKs) gained in popularity due to theoretical and practical breakthroughs in the past decade. Initially derived from linear PCPs,[10–12] SNARKs enjoy widespread practical support through a variety of development frameworks.[13–21] SNARKs from linear PCPs yield the shortest proofs, which renders the protocol of Groth[12] especially interesting for on-chain proof verification due to the shortest overall proof size. More recent SNARKs are based on IOPs, where the general recipe combines an IOP with a polynomial commitment scheme to obtain a SNARK.[22] SNARKs built with this recipe commonly rely on either polynomial commitment schemes based on the hardness of the Discrete

Logarithm (DL) assumption, or the KZG[23] polynomial commitment scheme. The downside of the above protocols is that they necessitate either a trusted, per-circuit setup[12] or a universal setup.[24] SNARKs based on the IOP-based FRI polynomial commitment scheme enjoy the benefit of a *transparent* setup that is untrusted, at the cost of larger non-constant proof size. The primary goal of a SNARK is to obtain proof sizes and verification time that are succinct, i.e., (at most) polylogarithmic in the size of the computation at hand.[25] The bottleneck of SNARKs is the prover algorithm, as it necessitates a number of public key operations that are linear in the size of the computation.[26] Further, SNARKs perform well for statements formulated as circuits. However, algebraic statements incur a significant overhead.[27] SNARKs' cost-efficiency makes them suitable when multiple verifications of proofs are required by one or several parties.

Commit-then-Prove ZKPs. Commit-then-prove ZKPs differ in the conceptual approach taken in the protocol. Assuming that the computation is represented as an arithmetic circuit, the commit-then-prove approach to designing a ZKP lets the prover commit to all input values, as well as all intermediate wires in the circuit, with a hiding and binding cryptographic commitment. Successively, the prover proves to the verifier that the committed values satisfy the relationship described by the arithmetic circuit, consisting of addition and multiplication gates. Recent works instantiate the commitment scheme with information-theoretic MACs (also referred to as VOLE),[28–30] and establish protocols with a computationally efficient prover and low memory overhead. Communication complexity and proof size of a protocol following the commit-then-prove design paradigm is linear in the number of multiplication gates present in the circuit. A common drawback of commit-then-prove protocols is that they introduce an interactive offline phase that can not be made non-interactive as the protocol is a private coin, i.e., the verifier holds randomness that is not disclosed to the prover.[31] Hence, the applicability of commit-then-prove protocols is, thus far, limited with regard to blockchains. An interesting avenue of research is the establishment of threshold designated-verifier ZKPs from commit-then-prove protocols.[29] A threshold set of verifiers can allow, e.g., verification of proofs by validator nodes, and represents a trade-off between public verifiability and prover performance.

MPC-in-the-Head. The idea of the MPC-in-the-Head (MPCitH) paradigm is to leverage Multi Party Computation (MPC) to build efficient ZK proof systems.[32,33] Conceptually, the prover emulates an MPC protocol "in its head", i.e., the prover simulates a set of virtual servers. Then, the prover commits to the views of the individual simulated servers running the emulated MPC protocol, and the verifier challenges the prover to open the commitment to a random subset of the views. The verifier verifies the consistency of individual views of the MPC protocol. The privacy guarantee of the emulated MPC protocol ensures the *zero-knowledge* property, given that the verifier is only allowed to open a small subset of commitments. The cost profile of ZKP protocols designed with the MPC-in-the-head paradigm is similar to the cost profile of commit-then-prove ZKPs.[26] However, the main advantage is that the protocol is entirely *public-coin*, i.e., it can be made non-interactive by applying the Fiat-Shamir transform. Recent work explores the benefit of combining the commit-then-prove approach with MPCitH by proposing VOLE-in-the-Head to establish public verifiability of a commit-then-prove style protocol.[31]

Sigma Protocols. Sigma protocols embody a trio of sequential steps to prove algebraic statements: Commitment, Challenge, and Response. Sigma Protocols are considered a standard ZK technique, and perform especially well when aiming to prove knowledge of an algebraic statement.[34] Common Sigma protocols are known for multiple alebraic statements, such as

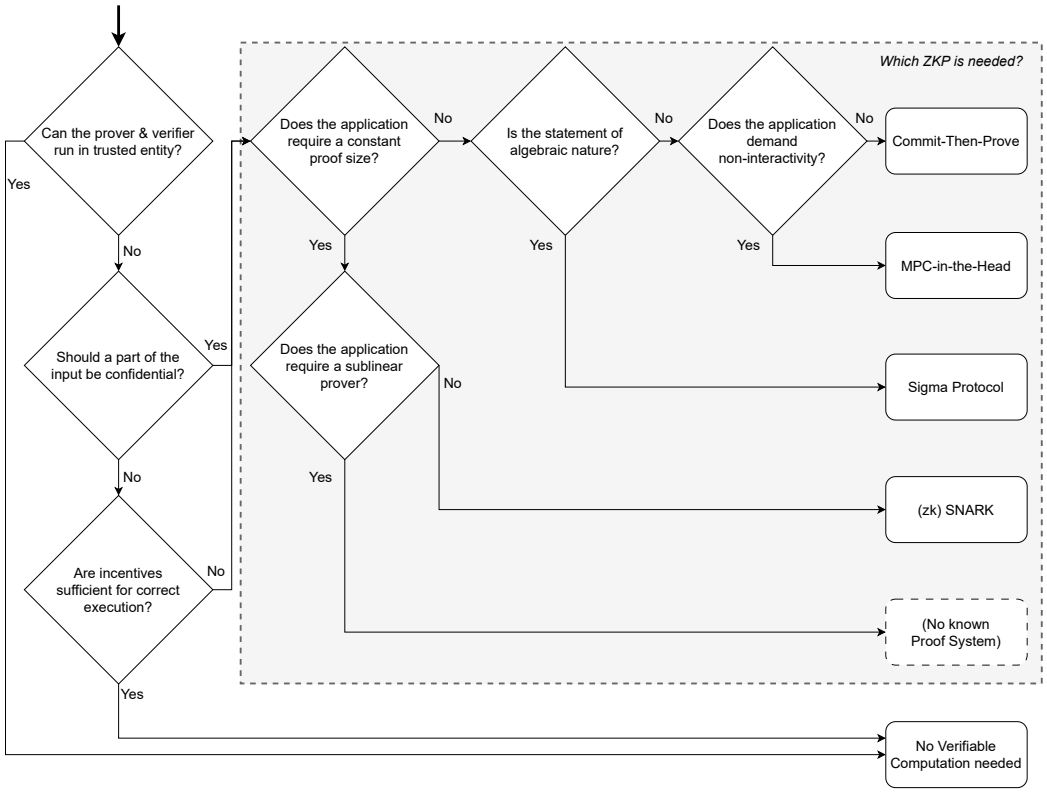

Fig. 1: A flow chart to determine which proof system is the appropriate technical solution to establishing cryptographic verifiability of outsourced computations. We assume that the *computation* is executed between a *prover* and a *verifier*.

proving possession of a discrete logarithm,[35] and are mostly used in applications that require rather simple statements. The downside of Sigma protocols is that they do not perform well when aiming to prove a computation represented as a boolean or arithmetic circuit, as the complexity of the prover and verifier scales linearly with the size of the circuit.[34]

In addition to differing properties concerning the size of the proof, communication as well as prover and verifier runtime, the above protocols differ in the underlying security assumption. We consider detailed considerations with regard to security *out of scope* for this high-level survey.

3 Do you need a Zero Knowledge Proof?

Zero-knowledge is pertinent in applications handling sensitive data. However, many future applications might need just cryptographic verifiability to outsource intensive computations or authenticate non-sensitive data. Different applications may require varied proof systems with specific trade-offs, as outlined in Section 2.3. To ease the high-level decision-making process when choosing a proof system for an application, we provide a decision tree in Figure 1. In the following, we describe proof systems as ZKPs, even when they do not fulfill the ZK property.

The decision tree assumes that a computation is carried out between a prover and a verifier. The tree bifurcates into multiple decision paths based on the requirements of the application at hand, such as the necessity for a constant proof size or non-interactivity. SNARKs are pinpointed as the go-to for applications demanding brief proof sizes. We highlight that there is currently no known Zero Knowledge Proof (ZKP) that provides both constant proof sizes and sublinear proof

computation for generic circuits. A commit-then-prove style protocol is recommended when the statement in question is not algebraic and does not require non-interactivity.

3.1 When are ZKPs futile?

While ZKPs provide strong privacy and verifiability, less expensive solutions can suffice for many applications. In the following, we discuss in which cases ZKPs, or specific instances thereof, are futile or not the adequate solution for a specific problem.

High Trust Environments. In scenarios with high mutual trust among parties, the necessity for a proof of correct computation becomes less critical. Consider the case of a proprietary trading firm where access to sensitive financial algorithms is restricted to trusted developers and analysts. In such a scenario, the emphasis is on protecting intellectual property, rather than safeguarding against internal actors. Lightweight cryptographic mechanisms, such as hash-based verifications or even simple access logs, can provide assurance that the computations are untampered with, without incurring the computational costs associated with ZKPs.

Single Verifier Scenarios. SNARKs have garnered widespread attention, particularly for enhancing blockchain privacy and scalability. However, their original purpose of outsourcing generic computation to untrusted servers is often impractical due to high prover costs and memory requirements (cf. Appendix 1).[36] For example, using SNARKs for Deep Neural Network inference demonstrates significant prover overhead even for simple tasks.[37] However, in most scenarios, succinctness is not strictly demanded. Succinctness is only beneficial if the verifier has to verify many proofs, or when the proof is evaluated by many verifiers. The challenge with SNARKs lies in balancing their overhead with benefits; they are justifiable when the privacy gain or verification by numerous parties offsets the prover's computational load. This is evident in cases like ZCash,[2] where transaction privacy is paramount, or in scaling networks like Ethereum with high transaction demand. However, applications without these needs, like single-client neural network queries, might find alternatives with lower computational demands more suitable.

Applications with Sufficient Alternative Incentives. Certain applications rely on ZKPs to only ensure the correct execution of a computation with strong cryptographic verifiability. An example is the use of a SNARK in a scenario, where a prover computes a succinct proof to only demonstrate execution correctness to a verifier. Alternatively, optimistic protocols rely on economic incentives and a challenge period. In this setup, an operator performs an execution, publishing the resultant state and necessary data for replication.[38–41] Challengers can issue a fraud proof against any detected malicious behavior by the operator, potentially profiting from successful challenges. This approach rely on the assumption that strong economic incentives discourage operator misbehavior. While optimistic protocols may be more cost-effective due to this reliance on incentives, they require a challenge period for security assurance. In contrast, ZKPs, though costlier due to proving expenses, offer quicker resolution times (bounded to proving time) and necessitate less data disclosure. This optimistic mechanism parallels that in credit networks, where misconduct directly impacts reputation. Opting for incentive-aligned protocols over ZKPs can be advantageous in contexts where security-performance trade-offs are acceptable, but they also necessitate more rigorous game-theoretic analysis.

When is the Sole Usage of ZKPs insufficient? A critical, yet often-ignored aspect across applications is the underlying trust assumptions. The intricacies of enabling operation within a

Table 1: Overview of Selected Zero-Knowledge Proof Applications.

Application Group	On-Chain Verification	Properties			Proof System	Reference
		Succinctness	Non-Interactivity	Zero-Knowledge		
Blockchain Applications						
zk-Rollups (zkEVMs)	✔	✔	✔	✗	SNARK	[46,51–53]
Privacy-Oriented Blockchains	✔	✔	✔	✔	SNARK	[2,54–56]
ZK Coprocessors	✔	✔	✔	✗	SNARK	[57]
ZK Bridges	✔	✔	✔	✗	SNARK	[45]
On-Chain ZK Dapps	✔	✔	✔	✔	SNARK	[58,59]
Proof-of-Storage	✔	✔	✔	✗	SNARK	[60]
Other Applications						
ZK Virtual Machines and Compilers	✗	✔	✔	✗	SNARK	[61,62]
TLS Oracles	✗	✔	✔	✔	SNARK	[63–65]
	✗	✗	✗	✔	CTP	[66,67]
Networking Applications	✗	✔	✔	✔	SNARK	[68]
Proving Software Exploits	✔	✔	✔	✔	SNARK	[69]
	✗	✗	✗	✔	CTP	[70]
	✗	✗	✔	✔	MPCitH	[71]
zkML	✗	✔	✔	✔	SNARK	[37]
	✗	✗	✗	✔	CTP	[72]
zkIdentity	✔	✔	✔	✔	SNARK	[73,74]
	✗	✗	✔	✔	Sigma	[34,75]
	✔	✗	✔	✔	Sigma	[76]
Document Editing for Fighting Disinformation and Privacy	✗	✔	✔	✔	SNARK	[77–79]

trustless framework necessitate a distributed trust model, which standalone ZKPs are incapable of establishing. ZKPs can either ensure input data confidentiality and cryptographic verifiability of computations, or solely the latter. Where privacy of input data is paramount over public cryptographic verifiability of correct computation execution, alternative secure computing techniques like MPC,[42] Fully Homomorphic Encryption (FHE),[43] or Trusted Execution Environment (TEE)[44] may be more suitable. ZKPs can further augment these methods, for instance, by assuring correct execution of MPC computations.[42] When it comes to minimizing trust in computing and verifying the proof, solutions include threshold proof generation,[45–48] and threshold proof verification.[49,50] Yet, some applications even offer services, where the untrusted prover is hosted by the same company developing the implementation of smart contract based verifiers. In the future, we expect that practical implementations of ZKP enabled applications will put additional consideration in separation of trust domains, especially in combination with additional techniques for secure computing, like TEE, MPC and FHE.

4 ZKP Use Cases

This section explores various ZKP use cases, focusing on three key applications: scaling legacy blockchains, enhancing digital self-sovereign identity, and improving common networking protocols. An extended review of ZKP applications is available in the Appendix. Our analysis spans both academic and industrial perspectives, assessing whether a ZKP is necessary, exploring available alternatives, and determining the most suitable ZKP based on system constraints.

Outsourcing Computation. One of the primary uses of ZKPs is the outsourcing of computation from a constrained environment to a computationally powerful, untrusted server. For example, a blockchain is a primary example of a slow, global computing environment that is replicated on many machines. Rollups have emerged as the most prominent solution to addressing the limited transaction throughput in a blockchain, boasting over $12.5 billion in Total Value Locked (TVL).

They *execute* and *batch* transactions on a Layer 2 (L2) chain and successively submit them to the main Layer 1 (L1) chain. A zk-rollup processes transactions off-chain and subsequently submits a validity proof and the resultant state change to the L1, substantially reducing costs. Rollups relying on ZKPs employ specialized circuits to generate proofs for EVM-based transactions.[51–53] Similarly, generic ZK virtual machines (zkVMs) allow provable execution of generic programs based on modeling a differing instruction set, such as RISC-V.[62] Generally, systems for outsourcing computation utilize SNARKs due to their non-interactive and succinctness (i.e., for verification on Ethereum). The constrained device outsources the task of executing the computation, whereas the constrained device simply obtains the result and succinctly verifies the correct execution. Recent efforts mainly rely on recursive aggregation of proofs to emulate the step-wise computation of a virtual machine and further minimize the prover overhead.[80]

Digital Self-Sovereign Identity. The proposal for verifiable credentials,[75] which forms a cornerstone on upcoming solutions for decentralized identity, relies on signatures of knowledge in their credential specification to facilitate selective disclosure of attributes with zero-knowledge. Practical anonymous credential systems are generally built around sigma-protocol ZKPs. Commonly, the credential is a signature on the holders' personal attributes. The attributes themselves are committed, and the holder of the credential proves with a Sigma protocol knowledge of a signature on the committed attributes. The committed attributes are commonly algebraic relations, and hence Sigma protocols perform exceptionally well in this case. Constructions for this purpose demand application-specific properties. For example, they additionally necessitate *unlinkability* of the proofs created.[81] Verification of this type of verifiable credential in an on-chain environment is expensive, and hence alternative constructions based on SNARKs with succinct proof and verification serve as an alternative for on-chain KYC.[73,74] Zero-knowledge is ultimately necessary, as personal data is considered sensitive and is only disclosed at the discretion of the credential holder. For this use case, alternatives to ZKPs are sparse, as the privacy requirement is strong and not trivially replaceable. Given our decision tree, decentralized identity with on-chain verification would necessitate zk-SNARKs, whereas proving personal attributes with ZKPs yields Sigma protocols due to expressing them as algebraic relations.

ZK for Networking. A very nascent application of ZKPs is their use to enhance and augment networking protocols. For example, recent work utilizes SNARKs for proving data provenance,[64] whereas an orthogonal avenue of research utilizes SNARKs to confidentially filter network traffic.[68] Whereas industrial efforts for proving data provenance are already implemented in practice, both with SNARKs[63] and commit-then-prove style protocols,[66] filtering network traffic with ZKPs is yet to be deployed. Interestingly, although the system design is conceptually similar, commit-then-prove style protocols are yet to be applied for filtering network traffic — even though in both cases, a single verifier can evaluate the proof's correctness, such that our methodology suggests a designated verifier protocol. In both cases, maintaining confidentiality introduces a significant overhead, as encryption needs to be executed in-circuit in order to establish ciphertext integrity. When plaintexts are disclosed, however, encryption can be performed out of the circuit. This highlights a strict trade-off between maintaining the zero-knowledge property or giving it up depending on the application requirements at hand.

5 Conclusion

Deciding whether a ZKP enhances the utility of an application is a non-trivial endeavor. To the best of our knowledge, this work provides the first methodology to decide whether and which type of ZKP is appropriate in different scenarios. Our methodology includes trust assumptions, confidentiality measures, and performance traits pertinent to both blockchain and non-blockchain applications. We elaborate our methodology at the example of three popular use cases in differing states of practical development. We conclude that there is a valid class of applications that do not demand for ZKPs, and the choice of ZKP should be tailored to the application context.

Acknowledgements

We thank Deevashwer Rathee and the independent reviewers for their valuable comments. This work is partially supported by Decentralized Intelligence AG (D23E). The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the programme of "Souverän. Digital. Vernetzt.". Joint project 6G-life, project identification number: 16KISK002.

Notes and References

[1] Goldwasser, S., Micali, S., Rackoff, C. "The Knowledge Complexity of Interactive Proof-Systems." In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing* Association for Computing Machinery 291–304 (1985) .

[2] Sasson, E. B., *et al.* "Zerocash: Decentralized anonymous payments from bitcoin." In *2014 IEEE symposium on security and privacy* IEEE 459–474 (2014) .

[3] Goldwasser, S., Micali, S., Rackoff, C. "The Knowledge Complexity of Interactive Proof Systems." *SIAM Journal on Computing* **18.1** 186–208 (1989).

[4] Babai, L. "Trading group theory for randomness." In *Proceedings of the seventeenth annual ACM symposium on Theory of computing* 421–429 (1985) .

[5] Shamir, A. "Ip= pspace." *Journal of the ACM (JACM)* **39.4** 869–877 (1992).

[6] Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M. "Proof verification and the hardness of approximation problems." *Journal of the ACM (JACM)* **45.3** 501–555 (1998).

[7] Ben-Sasson, E., Chiesa, A., Spooner, N. "Interactive oracle proofs." In *Theory of Cryptography: 14th International Conference, TCC 2016-B, Beijing, China, October 31-November 3, 2016, Proceedings, Part II 14* Springer 31–60 (2016) .

[8] Rackoff, C., Simon, D. R. "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack." In *Annual international cryptology conference* Springer 433–444 (1991) .

[9] Fiat, A., Shamir, A. "How to prove yourself: Practical solutions to identification and signature problems." In *Conference on the theory and application of cryptographic techniques* Springer 186–194 (1986) .

[10] Gennaro, R., Gentry, C., Parno, B., Raykova, M. "Quadratic span programs and succinct NIZKs without PCPs." In *Advances in Cryptology–EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings 32* Springer 626–645 (2013) .

[11] Parno, B., Howell, J., Gentry, C., Raykova, M. "Pinocchio: Nearly practical verifiable computation." *Communications of the ACM* **59.2** 103–112 (2016).

[12] Groth, J. "On the size of pairing-based non-interactive arguments." In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35* Springer 305–326 (2016) .

[13] ZCash "halo2." (2023) URL `https://github.com/zcash/halo2`.

[14] zkcrypto "bellman: zk-SNARK library." (2023) URL `https://github.com/zkcrypto/bellman`.

[15] scipt lab "libsnark." (2020) URL `https://github.com/scipr-lab/libsnark`.

[16] labs, O. "kimchi." (2023) URL `https://github.com/o1-labs/proof-systems`.

[17] labs, O. "snarkyjs." (2023) URL `https://github.com/o1-labs/snarkyjs`.

[18] Facebook "winterfell." (2023) URL `https://github.com/facebook/winterfell`.

[19] Kosba, A. "jsnark." (2022) URL `https://github.com/akosba/jsnark`.

[20] arkworks contributors "`arkworks` zkSNARK ecosystem." (2022) URL `https://arkworks.rs`.

[21] Botrel, G., Piellard, T., Housni, Y. E., Kubjas, I., Tabaie, A. "ConsenSys/gnark: v0.8.0." (2023) doi: 10.5281/zenodo.5819104 URL `https://doi.org/10.5281/zenodo.5819104`.

[22] Bünz, B., Fisch, B., Szepieniec, A. "Transparent SNARKs from DARK compilers." In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39* Springer 677–706 (2020) .

[23] Kate, A., Zaverucha, G. M., Goldberg, I. "Constant-size commitments to polynomials and their applications." In *Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16* Springer 177–194 (2010) .

[24] Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N. "Marlin: Preprocessing zkSNARKs with universal and updatable SRS." In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39* Springer 738–768 (2020) .

[25] Zhang, J., Xie, T., Zhang, Y., Song, D. "Transparent polynomial delegation and its applications to zero knowledge proof." In *2020 IEEE Symposium on Security and Privacy (SP)* IEEE 859–876 (2020) .

[26] Thaler, J., *et al.* "Proofs, arguments, and zero-knowledge." *Foundations and Trends® in Privacy and Security* **4.2–4** 117–660 (2022).

[27] Baldimtsi, F., Chatzigiannis, P., Gordon, S. D., Le, P. H., McVicker, D. "gotzilla: Efficient disjunctive zero-knowledge proofs from mpc in the head, with application to proofs of assets in cryptocurrencies." *Cryptology ePrint Archive* .

[28] Weng, C., Yang, K., Katz, J., Wang, X. "Wolverine: fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits." In *2021 IEEE Symposium on Security and Privacy (SP)* IEEE 1074–1091 (2021) .

[29] Baum, C., Jadoul, R., Orsini, E., Scholl, P., Smart, N. P. "Feta: efficient threshold designated-verifier zero-knowledge proofs." In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* 293–306 (2022) .

[30] Baum, C., Dittmer, S., Scholl, P., Wang, X. "SoK: Vector OLE-Based Zero-Knowledge Protocols." *Cryptology ePrint Archive* .

[31] Baum, C., *et al.* "Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head." In *Annual International Cryptology Conference* Springer 581–615 (2023) .

[32] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A. "Zero-knowledge from secure multiparty computation." In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing* 21–30 (2007) .

[33] Giacomelli, I., Madsen, J., Orlandi, C. "{ZKBoo}: Faster {Zero-Knowledge} for Boolean Circuits." In *25th USENIX Security Symposium (USENIX Security 16)* 1069–1083 (2016) .

[34] Chase, M., Ganesh, C., Mohassel, P. "Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials." In *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III 36* Springer 499–530 (2016) .

[35] Schnorr, C.-P. "Efficient signature generation by smart cards." *Journal of cryptology* **4** 161–174 (1991).

[36] Ernstberger, J., *et al.* "zk-Bench: A Toolset for Comparative Evaluation and Performance Benchmarking of SNARKs." *Cryptology ePrint Archive* .

[37] Kang, D., Hashimoto, T., Stoica, I., Sun, Y. "Scaling up Trustless DNN Inference with Zero-Knowledge Proofs." *arXiv preprint arXiv:2210.08674* .

[38] Khalil, R., Zamyatin, A., Felley, G., Moreno-Sanchez, P., Gervais, A. "Commit-chains: Secure, scalable off-chain payments." *Cryptology ePrint Archive* .

[39] Seres, I. A., Glaeser, N., Bonneau, J. "Naysayer proofs." *Cryptology ePrint Archive* .

[40] Kalodner, H., Goldfeder, S., Chen, X., Weinberg, S. M., Felten, E. W. "Arbitrum: Scalable, private smart contracts." In *27th USENIX Security Symposium (USENIX Security 18)* 1353–1370 (2018) .

[41] Ye, Z., Misra, U., Song, D. "Specular: Towards Trust-minimized Blockchain Execution Scalability with EVM-native Fraud Proofs." *arXiv preprint arXiv:2212.05219* .

[42] Kanjalkar, S., Zhang, Y., Gandlur, S., Miller, A. "Publicly Auditable MPC-as-a-Service with succinct verification and universal setup." In *2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* IEEE 386–411 (2021) .

[43] Bünz, B., Agrawal, S., Zamani, M., Boneh, D. "Zether: Towards privacy in a smart contract world." In *International Conference on Financial Cryptography and Data Security* Springer 423–443 (2020) .

[44] Cheng, R., *et al.* "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts." In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)* IEEE 185–200 (2019) .

[45] Xie, T., *et al.* "zkbridge: Trustless cross-chain bridges made practical." In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* 3003–3017 (2022) .

[46] Liu, T., Xie, T., Zhang, J., Song, D., Zhang, Y. "Pianist: Scalable zkRollups via Fully Distributed Zero-Knowledge Proofs." *Cryptology ePrint Archive* .

[47] Garg, S., Goel, A., Jain, A., Policharla, G.-V., Sekar, S. "zkSaaS: Zero-Knowledge SNARKs as a Service." *Cryptology ePrint Archive* .

[48] Ozdemir, A., Boneh, D. "Experimenting with Collaborative {zk-SNARKs}:{Zero-Knowledge} Proofs for Distributed Secrets." In *31st USENIX Security Symposium (USENIX Security 22)* 4291–4308 (2022) .

[49] Boneh, D., Boyle, E., Corrigan-Gibbs, H., Gilboa, N., Ishai, Y. "Zero-knowledge proofs on secret-shared data via fully linear PCPs." In *Annual International Cryptology Conference* Springer 67–97 (2019) .

[50] Boneh, D., Boyle, E., Corrigan-Gibbs, H., Gilboa, N., Ishai, Y. "Lightweight techniques for private heavy hitters." In *2021 IEEE Symposium on Security and Privacy (SP)* IEEE 762–776 (2021) .

[51] Polygon `https://polygon.technology/polygon-zkevm` "Polygon zkEVM." (2023) URL `https://polygon.technology/polygon-zkevm`.

[52] Labs, M. `https://era.zksync.io/` "zkSync Era." (2023) URL `https://era.zksync.io/`.

[53] Scroll `https://scroll.io/` "Scroll zkEVM." (2023) URL `https://scroll.io/`.

[54] Bonneau, J., Meckler, I., Rao, V., Shapiro, E. "Mina: Decentralized cryptocurrency at scale." *New York Univ. O (1) Labs, New York, NY, USA, Whitepaper* 1–47.

[55] Chin, C., Wu, H., Chu, R., Coglio, A., McCarthy, E., Smith, E. "Leo: A programming language for formally verified, zero-knowledge applications." *Cryptology ePrint Archive* .

[56] Aztec https://docs.aztec.network/ "Aztec Network." (2023) URL https://docs.aztec.network/.

[57] "Axiom." (2023) https://docs.axiom.xyz/zero-knowledge-proofs/introduction-to-zk.

[58] Pertsev, A., Semenov, R., Storm, R. "Tornado Cash Privacy Solution Version 1.4." *Tornado cash privacy solution version* **1**.

[59] Bender, C., Kraut, J. https://renegade.fi/whitepaper.pdf "Renegade Whitepaper." (2023) Renegade Whitepaper URL https://renegade.fi/whitepaper.pdf.

[60] Protocol Labs "Filecoin: A Decentralized Storage Network." (2023) https://filecoin.io/filecoin.pdf.

[61] Polygon "Miden VM." (2023) https://polygon.technology/polygon-miden.

[62] Bruestle, J., Gafni, P., the RISC Zero Team "RISC Zero zkVM: Scalable, Transparent Arguments of RISC-V Integrity." (2023) https://dev.risczero.com/proof-system-in-detail.pdf.

[63] Reclaim "Reclaim Protocol." (2023) https://www.reclaimprotocol.org/.

[64] Zhang, F., Maram, D., Malvai, H., Goldfeder, S., Juels, A. "Deco: Liberating web data using decentralized oracles for tls." In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* 1919–1938 (2020) .

[65] Lauinger, J., Ernstberger, J., Finkenzeller, A., Steinhorst, S. "Janus: Fast Privacy-Preserving Data Provenance For TLS 1.3." *Cryptology ePrint Archive* .

[66] PADO "PADO Labs." (2023) https://padolabs.org/.

[67] Xie, X., Yang, K., Wang, X., Yu, Y. "Lightweight Authentication of Web Data via Garble-Then-Prove." *Cryptology ePrint Archive* .

[68] Grubbs, P., Arun, A., Zhang, Y., Bonneau, J., Walfish, M. "{Zero-Knowledge} Middleboxes." In *31st USENIX Security Symposium (USENIX Security 22)* 4255–4272 (2022) .

[69] "ZK Proof of Exploit." (2023) https://github.com/zkoranges/zkPoEX.

[70] Cuéllar, S., Harris, B., Parker, J., Pernsteiner, S., Tromer, E. "Cheesecloth: Zero-Knowledge Proofs of Real-World Vulnerabilities." *arXiv preprint arXiv:2301.01321* .

[71] Green, M., Hall-Andersen, M., Hennenfent, E., Kaptchuk, G., Perez, B., Laer, G. V. https://eprint.iacr.org/2022/1223 "Efficient Proofs of Software Exploitability for Real-world Processors." (2022) Cryptology ePrint Archive, Paper 2022/1223 URL https://eprint.iacr.org/2022/1223.

[72] Weng, C., Yang, K., Xie, X., Katz, J., Wang, X. "Mystique: Efficient conversions for {Zero-Knowledge} proofs with applications to machine learning." In *30th USENIX Security Symposium (USENIX Security 21)* 501–518 (2021) .

[73] Rathee, D., Policharla, G. V., Xie, T., Cottone, R., Song, D. "Zebra: Anonymous credentials with practical on-chain verification and applications to kyc in defi." *Cryptology ePrint Archive* .

[74] Rosenberg, M., White, J., Garman, C., Miers, I. "zk-creds: Flexible anonymous credentials from zksnarks and existing identity infrastructure." In *2023 IEEE Symposium on Security and Privacy (SP)* IEEE 790–808 (2023) .

[75] Consortium, W. W. W., *et al.* "Verifiable credentials data model 1.0: expressing verifiable information on the web." *https://www. w3. org/TR/vc-data-model/?# core-data-model* .

[76] Sonnino, A., Al-Bassam, M., Bano, S., Meiklejohn, S., Danezis, G. "Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers." *arXiv preprint arXiv:1802.07344* .

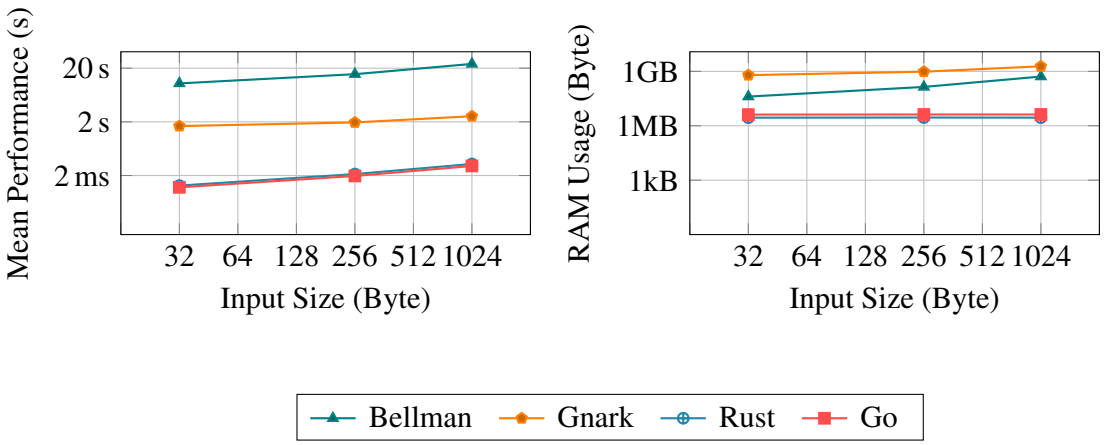

Fig. 2: Comparison between proving SHA-256 in a SNARK and execution of SHA-256 on a CPU.

[77] Kang, D., Hashimoto, T., Stoica, I., Sun, Y. "ZK-IMG: Attested Images via Zero-Knowledge Proofs to Fight Disinformation." (2022) 2211.04775.

[78] Datta, T., Boneh, D. accessed: 2023-11-10 "Using ZK Proofs to Fight Disinformation." (2022) https://medium.com/@boneh/using-zk-proofs-to-fight-disinformation-17e7d57fe52f.

[79] Ko, H., Lee, I., Lee, S., Kim, J., Oh, H. "Efficient Verifiable Image Redacting Based on Zk-SNARKs." In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security* New York, NY, USA: Association for Computing Machinery 213–226 (2021) doi:10.1145/3433210.3453110 URL https://doi.org/10.1145/3433210.3453110.

[80] Arun, A., Setty, S., Thaler, J. "Jolt: Snarks for virtual machines via lookups." *Cryptology ePrint Archive* .

[81] Camenisch, J., Lysyanskaya, A. "Signature schemes and anonymous credentials from bilinear maps." In *Annual international cryptology conference* Springer 56–72 (2004) .

[82] Miers, I., Garman, C., Green, M., Rubin, A. D. "Zerocoin: Anonymous distributed e-cash from bitcoin." In *2013 IEEE Symposium on Security and Privacy* IEEE 397–411 (2013) .

[83] "Bonsai." (2023) https://dev.risczero.com/api/bonsai/.

[84] Vesely, P., *et al.* https://eprint.iacr.org/2021/1361 "Plumo: An Ultralight Blockchain Client." (2021) Cryptology ePrint Archive, Paper 2021/1361 URL https://eprint.iacr.org/2021/1361.

[85] "Succinct Labs Telepathy." (2023) https://blog.succinct.xyz/introducing-telepathy/.

[86] "Dark Forest." (2023) https://zkga.me/.

[87] Fang, Z., Darais, D., Near, J. P., Zhang, Y. "Zero Knowledge Static Program Analysis." In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* New York, NY, USA: Association for Computing Machinery 2951–2967 (2021) doi:10.1145/3460120.3484795 URL https://doi.org/10.1145/3460120.3484795.

[88] Xing, Z., *et al.* "Zero-knowledge Proof Meets Machine Learning in Verifiability: A Survey." *arXiv preprint arXiv:2310.14848* .

Appendix A: Comparison — CPU Execution vs SNARK Proving

Setup. The experiments were conducted on a system equipped with an M1 max chip and 64 GB of RAM. For SNARK proving, we utilized the GNARK and Bellman libraries using groth16. These were compared against standard CPU execution using Go (with crypto/sha256) and Rust (sha2) to gauge the efficiency and resource usage. As inputs we provide preimages of sizes 32, 256, and 1024 bytes.

Table 2: Multiplicative Overhead of SHA-256 Execution in SNARK vs. CPU Environment

Input Size (Byte)	Performance Ratio (Bellman/Rust)	RAM Ratio (Bellman/Rust)	Performance Ratio (Gnark/Go)	RAM Ratio (Gnark/Go)
32	514523.9	14.7	2649.6	152.5
256	382408.1	49.4	979.9	233.7
1024	389473.7	187.9	606.4	462.0

Goal. The primary objective of these experiments was to evaluate the computational overhead of using SNARKs for proving, as compared to normal CPU execution. This involves assessing not just the execution time but also the RAM requirements.

Results. As shown in Figure 2, we observe significant disparities in both performance and RAM usage. Specifically, when using Bellman and Rust, SNARK proving requires 3,445,283,941 nanoseconds, compared to just 8,846 nanoseconds for plain CPU execution, making the former approximately 389,473 times slower. For the same input, GNARK, a more optimized library, exhibits a slowdown of 606 times (4,132,991 nanoseconds vs 6,816 nanoseconds for Go). In terms of memory consumption, Bellman performs better in comparison to GNARK but still demonstrates significantly higher resource usage. For example, plain Rust execution requires only 3 MB of memory, whereas Bellman consumes about 553 MB. Similarly, Go requires 4 MB, in stark contrast to GNARK, which consumes around 2 GB.

Concluding Remarks. These preliminary findings, indicate that while SNARK proving is powerful, it comes with considerable resource demands. Further experiments are required to obtain a more comprehensive understanding. Initial results, however, clearly demonstrate the substantial cost associated with SNARK proving, both in terms of wall time and RAM requirements. Therefore, it is essential to thoroughly assess and consider more efficient alternatives, whenever possible, before opting to to use ZKPs.

Appendix B: ZKP Blockchain-related Applications

The blockchain industry led the way in harnessing the power of Zero-Knowledge Proofs (ZKPs), with Zcash[2] being the first to implement and utilize ZKPs on a real-world application, setting a precedent in the field. This initial use case primarily addressed on-chain privacy. Over recent years, as ZKPs have become more practical and efficient, their application has spurred extensive research and development in the blockchain domain. These applications span from scalability solutions, like zkEVMs, to advanced privacy solutions (e.g., private smart contracts), and interoperability solutions.

Privacy-Oriented Blockchains. In the realm of privacy-focused blockchain technologies, Zcash[2] stands as a significant pioneer. As the first blockchain platform deployed with Zero-Knowledge Proofs (ZKPs) for enabling private transactions, Zcash's development was influenced by the foundational ideas in ZeroCoin.[82] While Zcash made significant strides in private transactions, it does not support programmable privacy. However, there are newer solutions that support private smart contracts as a first-class citizen, including Mina,[54] Aleo,[55] and Aztec.[56]

ZK Coprocessors. ZK coprocessors facilitate the transfer of complex on-chain computations off-chain, utilizing Zero-Knowledge proofs to ensure the validity of these computations. This

approach significantly enhances the efficiency of smart contracts by offloading extensive tasks and enabling the processing of historical on-chain data. Unlike zkRollups, which are integrated at the chain protocol layer, ZK coprocessors are primarily used by smart contracts for accessing and analyzing past on-chain states. Axiom[57] and Bonsai[83] are two notable examples. Axiom specializes in processing historical on-chain data off-chain and creating zero-knowledge proofs for its validation. Bonsai, on the other hand, extends the utility of ZK coprocessors by facilitating the creation of proofs for various off-chain computations, which can then be verified on-chain.

ZK Bridges. ZK Bridges are a solution to the scalability and security challenges faced by traditional cross-chain bridges, which suffer from the need for extensive computing power and storage to track multiple chain states and are often vulnerable to attacks due the trust assumptions they require. Leveraging Zero-Knowledge proofs, particularly SNARKs, ZK Bridges enable the verification of a source blockchain's state on a target blockchain efficiently and securely.[84] They use on-chain light clients to confirm source chain consensus within the target chain's environment, integrating this information into the target chain's consensus without additional trust assumptions. This approach, akin to running a full node, provides robust verification of both state transitions and consensus. Among the leading projects in this space are Telepathy,[85] which uses zkSNARKs for secure, trustless Ethereum interoperability without relying on less secure multisigs or centralized entities, and zkBridge,[45] a versatile framework that employs zk-SNARKs for efficient communication and state transition verification between chains.

On-Chain ZK Dapps. A significant and growing category in the blockchain space is the use of Zero-Knowledge Proofs (ZKPs) in decentralized applications (dApps), commonly referred to as zkApps. These applications leverage the zero-knowledge property of ZKPs to implement various functionalities, primarily focusing on enhancing privacy. Notable examples include privacy-centric dApps such as smart contract mixers like TornadoCash,[58] which facilitate anonymous transactions. In the realm of on-chain gaming, Dark Forest[86] stands out as an innovative use of ZKPs, offering unique gameplay mechanics. In the decentralized finance (DeFi) sector, a notable example is Renegade Finance,[59] a novel type of decentralized exchange known as an on-chain dark pool.

Proof-of-Storage. Proof-of-Storage schemes serve as a mechanism for clients to verify that a storage provider is faithfully storing the outsourced data. This concept is prominently utilized in Filecoin,[60] a peer-to-peer network designed for file storage with integrated economic incentives and cryptographic measures to ensure reliable long-term storage. In Filecoin's ecosystem, users pay storage providers to store their files. These providers are tasked with not only storing the files but also consistently proving their correct storage over time. A critical feature of Filecoin is its reliance on the efficiency of zk-SNARKs, which enables the platform to demonstrate the validity of storage using significantly less data than traditional methods. This efficiency is vital in enhancing the scalability and feasibility of the Proof-of-Storage system, making it a practical and secure solution in the decentralized storage market.

Appendix C: ZKP Other Applications

While blockchain technology has been a significant driving force in the evolution of Zero-Knowledge Proofs (ZKPs), their potential extends far beyond this domain. ZKPs are proving their versatility in a range of non-blockchain applications. This includes their integration into

diverse fields like disinformation countermeasures and network security, offering novel solutions to contemporary digital challenges. Moreover, ZKPs are finding synergy in hybrid applications that intersect with blockchain technology, such as zkVMs (Zero-Knowledge Virtual Machines) and zkML (Zero-Knowledge Machine Learning), showcasing their broad applicability and transformative potential across various technological industries.

ZK Virtual Machines and Compilers. ZK Virtual Machines (zkVMs) are an innovative concept where a virtual machine (VM) is implemented as a circuit within a ZKP system. This approach shifts the focus from proving the execution of a specific program to proving the execution of a VM itself, with the VM running as a zk circuit. Among the prominent zkVM projects is the Miden VM,[61] written in Rust and capable of automatically generating STARK-based proofs of execution for any program it runs. These proofs enable verification of correct program execution without needing to re-execute or know the program's details. Another significant project is RISC Zero,[62] which features a zkVM functioning like an embedded RISC-V microprocessor, thereby simplifying the development of powerful and verifiable zk applications. These zkVMs represent a leap in computational verification, offering robust, efficient, and transparent ways to verify complex computations. Adding to this innovation, zkLLVM presents another approach. It is an LLVM-based circuit compiler capable of proving computations in languages supported by LLVM, such as C++, Rust, JavaScript/TypeScript, among others. Unlike a traditional VM, zkLLVM functions as a circuit compiler. It takes high-level code and transforms it into a circuit, which is then utilized to prove computations.

Proving Software Exploits. The application of ZKPs in software security marks a significant advancement, enabling security reasearchers to prove the existence of vulnerabilities without revealing sensitive details. Key initiatives in this domain include CHEESECLOTH,[70] a proof-statement compiler that preprocesses programs and selectively reveals control segment information while maintaining confidentiality. Further contributions come from Green et al.,[71] who demonstrate the practicality of using ZK proofs for real-world exploits in processor architectures without needing source code. Complementing these efforts, Fang et al.[87] introduce zero-knowledge static analysis, a method that allows for the verification of program properties via zero-knowledge proofs, avoiding the disclosure of the program's source code.

Zero-Knowledge Machine Learning. The integration of zero-knowledge proofs with machine learning is advancing rapidly, offering innovative solutions in various domains. Kang et al.[37] present a method for verifying ML model execution on ImageNet-scale using ZK-SNARKs, addressing challenges in ML-as-a-Service (MLaaS) by ensuring correct model predictions and model accuracy in the face of unreliable service providers. Additionally, Xing et al.[88] provide a comprehensive survey on zero-knowledge proof-based verifiable machine learning (ZKP-VML), highlighting its potential in addressing trustworthiness issues in outsourced learning and federated learning scenarios. Further, Weng et al.'s development of the Mystique[72]system demonstrates efficient ZK conversions in large-scale neural network inference, illustrating the significant progress in interactive ZK proofs for complex computations like matrix multiplication, with minimal accuracy loss in private inference.

Document Editing for Fighting Disinformation and Privacy. The innovative application of Zero-Knowledge (ZK) proofs in verifying transformations of digital images marks a crucial advancement in combating disinformation. This method facilitates the authentication of edited

images' authenticity and provenance, typically involving scaling down for serving images through a website. Datta and Boneh[78] have focused on validating the authenticity and integrity of these image transformations. Their approach ensures that only permissible edits are applied to images originally authenticated by signature-producing cameras, which digitally sign each photo along with its metadata, such as location and timestamp. This process is vital for confirming that images have not been misleadingly manipulated and that their origin and timestamp are verifiable. Complementing this, the work of Daniel Kang et al. with ZK-IMG[77] and that of Hankyung Ko et al.[79] on zk-SNARKs-based image redacting further enhances these capabilities. These projects are pivotal in attesting to image transformations and authenticating redacted images, thereby preserving the fidelity of the original images.