

User privacy in DAOs

Zero-Knowledge General Research



Ignas Apšega

3620557

Version Control

Version	Date of change	Change description
0.1	24-10-2022	Initial draft
0.2	25-10-2022	Document format, branding
0.3	29-10-2022	Introduction, context
0.4	17-11-2022	Zero-Knowledge Proctols, implementations and libraries
0.5	21-11-2022	Conclusion, advise
0.6	15-12-2022	Added: 1. Research question 2. Advise using zkSNARKs 3. More context on “Zero-Knowledge Proof implementations” 4. A more clear description of ZKP libraries' decision matrix. 5. More clear Results and Advise

Glossary

Term	Explanation
1. Decentralized Autonomous Organisation (DAO)	DAO, or Decentralised Autonomous Organisation, is an organization without any hierarchy among its members. A DAO also has a set of “rules” of how to manage its treasury (DAO’s money) and how to handle votings, which are essential for making decisions, and a DAO usually has its own Token (Cryptocurrency, in this case). All these aspects are described inside a DAO smart contract.
2. Zero-Knowledge Proofs (ZKP)	Zero-Knowledge Proofs (ZKPs) — a method for one party to cryptographically prove to another that they possess knowledge about a piece of information without revealing the actual underlying information. In the context of blockchain networks, the only information revealed on-chain by a ZKP is that some piece of hidden information is valid and known by the prover with a high degree of certainty.
3. API	API is the acronym for <i>Application Programming Interface</i> , which is a software intermediary that allows two applications to talk to each other.
4. Domain-specific language (DSL)	It is a computer language specialized to a particular application domain . (“Domain-specific language”)
5. Arithmetic circuit	An arithmetic circuit is a set of gates with a separate set of inputs for each number

	that has to be processed. The gates are connected so as to carry out an arithmetic action, and the outputs of the gate circuit are the digits of the result (addition, subtraction, multiplication, or division). Arithmetic circuits are used to define problem in arithmetic way(Ouaddah)
6. zk-SNARK	The zk-STARK is a type of highly secure cryptographic testing that uses Zero-Knowledge Testing (ZKP) principles to create encrypted data that can be easily verified without revealing sensitive information about such data and, best of all, with resistance to quantum computing guaranteeing its security in the not so distant future. (“zk-STARKs vs zk-SNARKs: Differences in zero-knowledge technologies”)
7. Bulletproofs	They are short, non-interactive zero-knowledge proofs that can convince a verifier that an encrypted value lies within a stated range without disclosing any information about the number. (“Bulletproofs Stanford Applied Crypto Group”)
8. zk-STARK	As another type of non-interactive zero-knowledge proof, zk-STARKs are less popular than their zkSNARK counterparts, mostly because they are a newer model. The term stands for zero-knowledge scalable transparent arguments of knowledge, and they are faithful to the fundamentals of zero-knowledge proofs, allowing users to share validated data or perform computations with third parties without disclosing any actual information to the third party (“What are zk-STARKs?”)
9. Trusted setup	Trusted setup — a trusted setup is a party (or parties) that generates random secrets and then is supposed to delete them. It is called “trusted” because users blindly trust this party to handle secret deletion; otherwise, it may cause severe

	vulnerabilities.
10. Proof-size	Size of a proof(encrypted data) in Bytes (B), Kilobytes (KB)
10. (Secure) Multiparty Computation (MPC / SMPC)	(Secure) Multiparty Computation is a cryptographic protocol that distributes computation across multiple parties where no individual party can see the other parties' data.” (source , also check this source for a better high-level understanding of the MPC concept).
11. Post-quantum	Post-quantum cryptography (sometimes referred to as quantum-proof, quantum-safe, or quantum-resistant) refers to cryptographic algorithms (usually public-key algorithms) that are thought to be secure against a cryptanalytic attack by a quantum computer . (“Post-quantum cryptography”)
12. zk-Rollup	A ZK-rollup is a Layer-2 blockchain protocol that processes transactions, performs computations, and stores data off-chain while holding assets in an on-chain smart contract. (“ZK-rollup projects: Inner workings, importance & analysis”)
13. Layer2	Layer 2 refers to a secondary framework or protocol that is built on top of an existing blockchain system. (Paszke)

Table of Contents

User privacy in DAOs	0
Zero-Knowledge General Research	0
Version Control	1
Glossary	2
Table of Contents	5
1. Introduction	6
1.1 Purpose of document	6
1.2 Context	6
2.1 Research method	6
2.2 Research strategy	7
2.3 Research methods	7
3. Research Results	8
3.1 Zero-Knowledge Protocols	8
Interactive Proofs	8
Advantages	9
Disadvantages	9
Non-interactive Proofs	9
Advantages	9
Disadvantages	9
3.2 Zero-Knowledge Proof implementations	12
3.3 Zero-Knowledge Proof libraries	12
Explanation	12
Coloring	13
Results	15
Conclusion	15
Advice	16

1. Introduction

1.1 Purpose of document

This document is part of the User Privacy in DAOs[1] research assignment. This document aims to take an in-depth look into protocols, implementations, and libraries of Zero-Knowledge Proofs[2]: zk-Snarks, Bulletproofs, and zk-Starks.

1.2 Context

The purpose of this research is to discover which protocol, implementation, and libraries of Zero-Knowledge are the best options to be used within User Privacy in the DAOs context.

2. Research

2.1 Research Questions

This document was made to help understand what is Zero-Knowledge Proofs their libraries and implementations. This research is part of the following sub-questions, which were also researched in the DAO Analysis document:

- **What tools are there for improving privacy for DAOs?**

2.3 Research strategy

For this research, mainly the strategy **Library** and **Workshop** were chosen.

2.4 Research methods



Library is done to explore what is already done and what guidelines and theories exist that could help







Workshop It is done to explore opportunities. Helps to gain insights into what is possible and how things could work.



Available product analysis

Available product analysis was used to answer most of the research questions. This research method helps to find what has already been done. This means already existing DAOs were researched, including how they operate.



			
	X	✓	✓
	✓	X	✓
	X	✓	X

Multi-criteria decision making

Multi-criteria decision making was used to research all the Zero-Knowledge Proofs libraries and tooling. Several criteria were chosen to narrow down and pick the right tools for in-depth testing and research.



Literature study

Literature study was used to research online literature to understand what DAO is, how it operates, and if it has any privacy problems.

3. Research Results

After finding that there are no specific tooling or solution to improve the privacy of the DAOs - more on this in the document - ***”DAO analysis and advice”***. Thus, possible solutions on the blockchains were looked at where Zero-Knowledge Proofs technology was found. In addition, this technology was suggested by Byont to research if it could help

To begin with the results, an overview of Zero-Knowledge Proofs and their types, protocols, and implementations is needed.

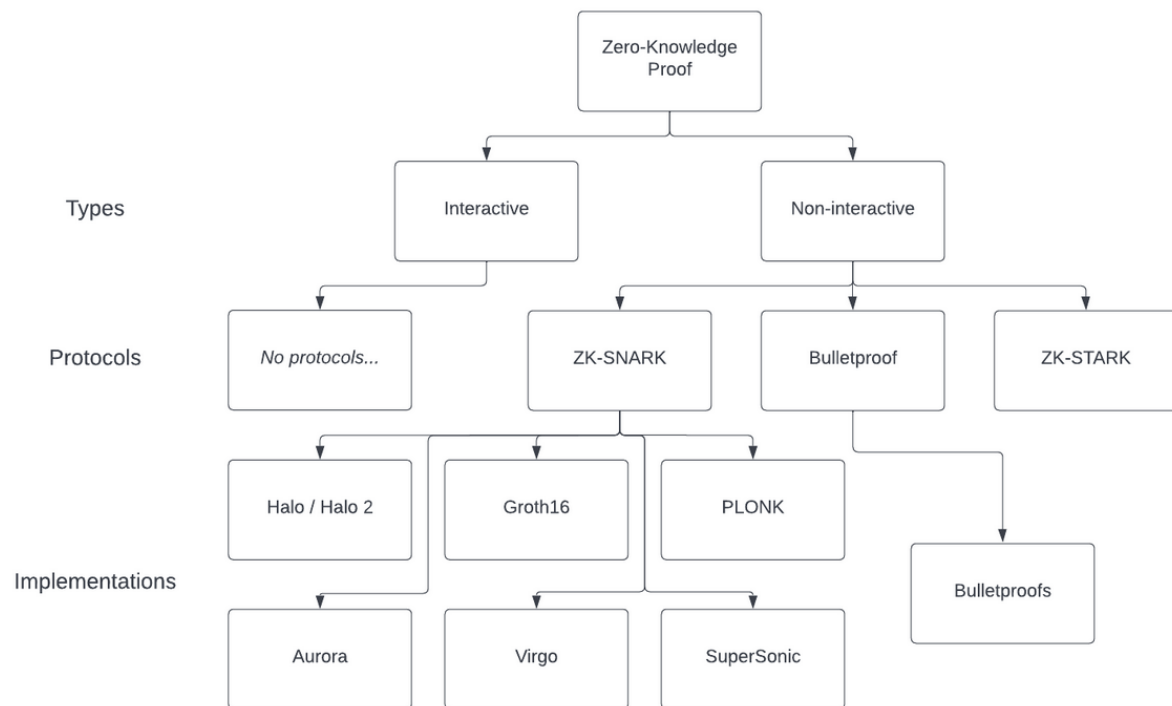


Figure 1. An overview of ZKP: types, protocols, implementations

3.1 Zero-Knowledge Protocols

Not all Zero-Knowledge Proofs are the same. There are two main types of Zero-Knowledge Proof, and they both have pros and cons depending on the use case they apply.

Interactive Proofs

This type of ZK-Proof involves a “**prover**” and a “**verifier**” continuously interacting with each other until the **verifier** finishes verifying the data (sometimes this data is also called “**witness**”).

Advantages

- Easy to execute A **verifier** simply interact with a **prover** until the first one is convinced that the data is correct.

Disadvantages

- **Limited transferability:** To prove the same information again to another **verifier**, the entire process must be repeated.
- **Not scalable:** Interactive ZKPs require both the **verifier** and **prover** to be online at the same time, which makes the entire process non-scalable.

Non-interactive Proofs

In this ZKP type, the **verifier** doesn’t need to interact with the **prover** all the time until the first is convinced. In this ZKP type, the data/**witness** can be verified publicly without a prover being online.

Advantages

- **Scalable:** It does not require a **prover** or **verifier** to be online all the time.
- **Transferable:** If the **prover verifies** the proof of the **witness** once, it can be made public, and the same process is not to be repeated for the different verifier.

Disadvantages

- **Requires a lot of computation power:** To verify the data non-interactively, a machine must solve mathematical equations many times, which may take up to several minutes to verify the **witness** (an annoying amount of time for web and desktop applications).

Interactive ZKPs can be used in real-life examples, as it is easier to keep both the **prover** and **verifier** interacting. For example, it is used for [Nuclear disarmament](#).

But **non-interactive ZKPs** are used more frequently nowadays, as they are more suitable

for usage on the internet and in applications. Moreover, **non-interactive Zero-Knowledge Proof** is used within a blockchain. Thus further focus will shift only to them.

To continue, several decision matrices were made to narrow down the research towards choosing the right **Zero-Knowledge Proofs** implementation. The most widely used ones are zk-SNARK[6], Bulletproof[7], zk-STARK[8]:

	SNARKs	STARKs	Bulletproofs
Algorithmic complexity: prover	$O(N * \log(N))$	$O(N * \text{poly-log}(N))$	$O(N * \log(N))$
Algorithmic complexity: verifier	$\sim O(1)$	$O(\text{poly-log}(N))$	$O(N)$
Communication complexity (proof size)	$\sim O(1)$	$O(\text{poly-log}(N))$	$O(\log(N))$
- size estimate for 1 TX	Tx: 200 bytes, Key: 50 MB	45 kB	1.5 kb
- size estimate for 10.000 TX	Tx: 200 bytes, Key: 500 GB	135 kb	2.5 kb
Ethereum/EVM verification gas cost	$\sim 600k$ (Groth16)	$\sim 2.5M$ (estimate, no impl.)	N/A
Trusted setup required?	YES 😞	NO 😊	NO 😊
Post-quantum secure	NO 😞	YES 😊	NO 😞
Crypto assumptions	DLP + secure bilinear pairing 😞	Collision resistant hashes 😊	Discrete log 😞

Figure 2. Comparison of Zero-Knowledge Protocols

In summary:

- SNARKs: **fast, cheap**, usually rely on a trusted setup[9] - a party (or parties) that generates random secrets and then is supposed to delete them.
- Bulletproofs: have **moderate proof size** [10] and verification time and are believed not to rely on a trusted setup but use Multiparty Computation(MPC)[11].
- STARKs: relatively fast, **post-quantum**[11] secure, too big proof-size for blockchain.

It is worth mentioning that zk-SNARKs need an initial trusted setup phase to generate the randomness required to generate Zero-Knowledge Proofs. Which can bring a lack of transparency and security.

However, besides technical research of these protocols, research on their usability was carried out also:

	zk-SNARKs	zk-STARKs	Bulletproofs
Popularity	+	+-	-
Documentation	+	+-	-
Production-ready	+	+-	+-
Amount of libraries	+	-	-
Adoption	+	+-	+-

Table 1. Decision Matrix of Zero-Knowledge Protocols

Advise

In conclusion, even though zk-STARKs and Bulletproofs are the newer solutions of Zero-Knowledge, zk-SNARKs wins in our case. Currently, Bulletproofs only offer privacy to public blockchains like Bitcoin, which can resolve the widespread privacy concerns associated with traditional blockchain networks.

However, zkSNARKs and zkSTARKs, are able to surpass Bulletproofs in terms of features. These other types are less resource-intensive and are cost-friendly. Because of this, recent developments of [privacy-preserving protocols](#) and [scaling solutions](#) have skewed towards zkSNARKs and zkSTARKs instead of bulletproofs. Because zk-SNARKs is the oldest solution, it is battle tested and pretty mature to have a good amount of sources to study and implement it. Moreover, zk-SNARKs are already being used in privacy protocols of the blockchain because it is cheap and fast compared to zk-STARKs which are used for blockchain scalability.

All in all, based on the above points. zkSNARKs protocol looks like it is the most suitable implementation of Zero-Knowledge technology.

3.2 Zero-Knowledge Proof implementations

After researching Zero-Knowledge Protocol to build on (zk-Snarks). It was found that there are several implementations of protocols. These implementations differ mainly in

Proof size - how big the proving data can be

Trustliness - if it needs a trusted setup[9].

Scalability - how easy is it to scale the applications on this implementation

Universality - is one that does not require a separate trusted setup for each circuit[5].

Verification time - how long does it take to verify the proof.

An overview and decision matrix was made to review the implementations, and these 4 came as the best implementations.

	Proof size	Trustless	Scalable	Universal	Verification time	Languages for implementation
Groth16	0.2 kB	No	No	No	10 <u>ms</u>	Rust
PLONK	0.5 - 1 kB	No	Yes	Yes	~10 <u>ms</u>	Rust
Halo 2	3.5 kB	Yes	Yes	Yes		Rust
<u>SuperSonic</u>	10 kB	Yes	Yes	Yes	100 <u>ms</u>	

Figure 3. Zero-Knowledge Proof implementations decision matrix.

In summary, Groth16, PLONK, and Halo2 look like the best options because they have smaller proof sizes and fast verification time.

3.3 Zero-Knowledge Proof libraries

To continue the research, a multi-criteria decision matrix was made. It was chosen to filter out the unnecessary libraries of Zero-Knowledge Proofs. The requirements for the right libraries are:

Explanation

1. **ZKP Implementation:** shows on which implementation of Zero-Knowledge Proof the tooling is built. The preference is either **Halo 2**, **SuperSonic**, **Groth16**, **Marlin**, or **PLONK** implementations.
2. **Language:** shows the language for which the tooling is built. The preference is either **Rust**, **JavaScript**, or **TypeScript**. Other widely used languages or **Domain Specific Languages** (DSLs) designed specifically for writing Zero-Knowledge solutions can be considered.
3. **Version:** shows the latest version of the tool. The preference is the Versions \geq 1.0.0, which are suitable for use in production, seen as a perfect fit, and marked with green background. Versions not suitable for production can be considered but are not preferred.
4. **Use-cases:** shows for which purposes the tool is used. Preference is for the tools which help with DAOs' privacy, anonymity, and voting issues.

Coloring

- **Green background** — Perfectly suits needs/intentions.
- **Orange background** — Doesn't meet requirements but can be considered.
- **Purple background** — Is not going to be considered. Works as a knockout criteria

Ideally, at least 2 green columns would be worth researching in the library. In addition, one of the main factors to look into is the version and production-ready usability together with the library's documentation - meaning how beginner-friendly it is.

	ZKP implementation	Language	Version	Use-Cases	Sources
halo2	Halo 2	Rust	0.1.0-beta.2	General-use ZKP implementation without a trusted setup	GitHub , crates.io , docs
libsnark	Groth16	C++	Suitable for production	1. General-purpose proof systems; 2. Gadget libraries for constructing R1CS instances; 3. Examples of applications	GitHub , tutorial
bellman	Groth16	Rust	0.13.1	A tool for building zk-SNARK circuits	GitHub , crates.io , docs
gnark	Groth16 or PLONK	GO + API	0.7.1	Fast zk-SNARK library that offers a high-level API to design circuits	GitHub , docs
snarky	Groth16	OCaml	Should not be used in production	OCaml DSL for verifiable computation	GitHub , docs
ZoKrates	ZoKrates	ZoKrates	Not tested for production	A toolbox for zkSNARKs on Ethereum. Generates proofs that can be verified in Solidity	GitHub , docs
zokrates.js	ZoKrates	JavaScript	Not tested for production	JavaScript bindings for ZoKrates	GitHub , docs
Circom	Groth16	Circom	2.1.0	Efficient circuit framework for programmable zero-knowledge	GitHub , YouTube , docs , website
circomlib	Groth16	JavaScript	2.0.5	Library of basic circuits for circom	GitHub , docs
SnarkJS	Groth16 and PLONK	JavaScript	0.5.0	General-use ZKP implementation with a trusted setup. Uses Powers of Tau ceremony	GitHub , simple project
snarkyjs	Probably Groth16	TypeScript,	0.6.1	Makes it easy to write and	GitHub ,

		JavaScript		test zkApps with the help of zkApp CLI	docs
Noir	PLONK, Groth16, Marlin	Noir	0.2.0	Domain Specific Language for SNARK proving systems	GitHub , docs
Leo	Not clear	Leo	1.5.3, alpha	A Programming Language for formally verified, Zero-Knowledge applications	GitHub , docs , live playground
Aztec SDK	PLONK	Typescript	2.1.45	Provides a simple API for creating accounts, depositing and withdrawing tokens, and interacting with Ethereum smart contracts anonymously	GitHub , docs
Light Protocol SDK	Groth16	Rust	Some of their SDKs are public some private	Verifies zkSNARK proofs to enable anonymous transactions on Solana.	GitHub , docs , available SDKs
Aleo SDK	Not clear	Rust	0.2.0	An SDK for Zero-Knowledge Transactions	GitHub , docs , awesome Aleo
arkworks ecosystem	Groth16, Marlin	Rust	Depends on repository	Libraries in the ecosystem provide efficient implementations for building zkApps: from generic finite fields to R1CS constraints for common functionalities.	GitHub , crates.io
espresso ecosystem: jellyfish, cape and other tools	PLONK	Rust	Use at your own risk	1. Private payments 2. Configurable private assets 3. Privacy-preserving asset management	GitHub , website
plonky2	PLONK & FRI no trusted setup	Rust	Most likely can be used for production	Plonky2 is built for speed, and features a highly efficient recursive circuit. Used for Ethereum	GitHub , YouTube

				scalability in Polygon Zero	
zkay	Default Groth16, but you can import other proving-schemes	zKay	V0.3.0 Not ready for production	A Language for Private Smart Contracts on Ethereum	GitHub, docs

Table 2. Decision Matrix of Zero-Knowledge Proofs libraries.

Results

After looking deeper into these libraries, it appears that the most suitable options for us are **bellman**, **Circom**, and **arkworks** ecosystem. However, **Circom** and its ecosystem look the most promising because it was already used in production-ready projects and has some documentation that looks the most beginner-friendly compared to other libraries and solutions.

Conclusion

In summary, Zero-Knowledge Proofs are one of the hottest concepts in the blockchain space right now. Its oldest non-interactive protocol, zk-SNARKs, was introduced in 2012, with its Privacy-protecting digital currency [Zcash](#) implemented in 2016. The concept of proving data without revealing it brings a lot of use cases in the blockchain and DAO context. Which could:

- **Hiding the payment sums and recipients.**
- **Hiding the reward sums and recipients.**
- **Hiding the people who vote and make proposals.**

Advise

Even though there are multiple protocols of **Zero-knowledge**. I advise using the one that is more mature and has production-ready tools and real-life projects, and by this definition, **zk-SNARKs** wins over other protocols. Moreover, according to [this article](#) -

Because **zk-SNARKs** can quickly generate and verify zero-knowledge proofs:

1. **Proven tech** - More privacy-enhancing protocols have been used them.
2. Privacy-enabled blockchains which hide transactions from the public eye were made on zk-SNARKs: [Zcash](#), [Mina](#)
3. It can be adapted for identity verification, allowing users to pass [KYC and AML](#) requirements without putting their sensitive information at risk. Using zk-SNARKs, these protocols can allow users to submit zero-knowledge proof of their identity instead of documents, confirming that they have the required data to be verified without revealing any extra information.

On the other hand, we have zk-STARKs. They are known to scale better than zk-SNARKs:

1. Adopted by blockchain scalability solutions like ZK-rollups[12] and Layer-2[13] blockchain solutions.
2. These protocols compute transactions and store data off-chain, then submit zero-knowledge proofs on-chain to update the network's stat.

After comparing both, it leads to the conclusion that the former **zk-SNARKs** are more adopted in the privacy field than the other protocols.

To continue, I choose **Circom** and **SnarkJS** (they are from the same Circom/iden3 ecosystem) libraries because they implement **zk-SNARKs** protocol and can implement **Groth16** or **PLONK** implementations. Moreover, these libraries are already adopted in real-life production, such as

1. [Tornado Cash](#) - crypto mixer. A is a service that mixes potentially identifiable cryptocurrency funds with others to obscure the trail back to the fund's original source
2. [Polygon Hermez](#) - scalability solution for Ethereum. Uses zk-SNARKs.
3. [Dark Forest](#) - Zero-Knowledge proofs game on the blockchain.

This brings us to the conclusion that I advise using the **zk-SNARKs** protocol of **Zero-Knowledge Proofs** together with **Circom** and **SnarkJS** libraries.

Citation and Reference

“Bulletproofs | Stanford Applied Crypto Group.” Applied Cryptography Group,
<https://crypto.stanford.edu/bulletproofs/>. Accessed 25 October 2022.

“Bulletproofs | Stanford Applied Crypto Group.” Applied Cryptography Group,
<https://crypto.stanford.edu/bulletproofs/>. Accessed 17 November 2022.

“Circom 2 Documentation.” Circom 2 Documentation, <https://docs.circom.io/>. Accessed 25
October 2022.

“dalek-cryptography/bulletproofs: A pure-Rust implementation of Bulletproofs using
Ristretto.” GitHub, <https://github.com/dalek-cryptography/bulletproofs>. Accessed
25 October 2022.

“Domain-specific language.” Wikipedia,
https://en.wikipedia.org/wiki/Domain-specific_language. Accessed 25 October
2022.

“iden3/circom: zkSnark circuit compiler.” GitHub, <https://github.com/iden3/circom>.
Accessed 25 October 2022.

Ouaddah, Aafaf. “Arithmetic Circuit.” ScienceDirect, 2019,
<https://www.sciencedirect.com/topics/engineering/arithmetic-circuit>. Accessed 25
10 2022.

Paszke, Antoni. “Layer 2.” Binance Academy,
<https://academy.binance.com/en/glossary/layer-2>. Accessed 21 November 2022.

“Post-quantum cryptography.” Wikipedia,
https://en.wikipedia.org/wiki/Post-quantum_cryptography. Accessed 17 November

2022.

“ZK-rollup projects: Inner workings, importance & analysis.” Panther Protocol Blog, 26

August 2022,

<https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/>. Accessed 21 November 2022.

“zk-STARKs vs zk-SNARKs: Differences in zero-knowledge technologies.” Panther Protocol

Blog, 2 September 2022,

<https://blog.pantherprotocol.io/zk-snarks-vs-zk-starks-differences-in-zero-knowledge-technologies/#what-are-zk-starks>. Accessed 17 November 2022.