POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

ZK-KYC-DSIG: An eIDAS2 Compliant Privacy Preserving Identity Verification Framework via Zero Knowledge Proof and Digital Signature

Laurea Magistrale in Computer Science Engineering - Ingegneria Informatica

Author: Matteo Savino

Advisor: Prof. Francesco Bruschi

Co-advisor: Marco Esposito

Academic year: 2023-2024

1. Introduction

In the last decades, digital services have expanded significantly in both the public and private sectors. Service providers must be certain who they are dealing with and/or be able to confirm a user's claims. For this reason, the concept of digital identity, the relationship between an individual and the set of data related to its online presence, has become essential to online interactions. Identity proofing typically involve sharing personal data with institutions and organizations. This environment requires people to disclose an ever increasing amount of personal information, creating privacy concerns. Furthermore, the growing trend of data breaches in an era of personal data commoditization significantly contributes to the surge in identity theft fraud [1]. The concept of Self-Sovereign Identity (SSI) gained popularity in response to the aforementioned circumstances. It revolves around a user-centric identity model (shown in figure 1).

Regulators have acknowledged the need to actively address digital identity challenges. The EU eIDAS2 Regulation established the legal and technical foundations needed for secure and straightforward cross border digital transactions. Despite being technologically neutral, it incorporates new features such as Selective Disclosure and it enables emerging technologies such as Digital Ledger Technologies (DLTs) and Zero Knowledge Proof (ZKP) in the regulatory framework. By doing so, eIDAS 2 has paved the way for innovative approaches that align with SSI principles.

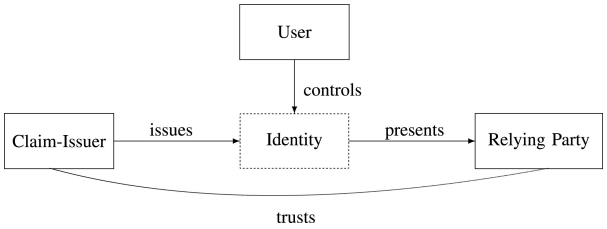

Figure 1: Self-Sovereign Identity actors.

This research presents the ZK-KYC-DSIG framework, a privacy preserving identity proofing system which aims to align with the eIDAS2 regulation and SSI principles. It leverages the usage of eIDAS2 compliant digital signature, EVM compatible blockchain and ZK-SNARKs ZKP. The solution allows users to prove their

identity without disclosing sensitive information. Additionally, a Legal Authority is able to uncover the identity of the prover if mandated for legal purposes, making the approach suitable for KYC scenarios. A thorough evaluation of experimental results, conducted on a working prototype, is provided together with an in depth analysis of opportunities, limitations and directions to overcome these limitations. The study demonstrates feasibility of the approach and aims to become the foundations for a standard in the digital identity field.

2. Background and Method

2.1. PKI and CMS

The primary goal of certificates and Public Key Infrastructure (PKI) is to bind an identifier to a public key. Once, this identifier is confirmed with a trusted authority like Certificate Authorities (CAs) the identity is verified. In addition, PKI also enable the concept of digital signature and encryption, enforced by the double key system. This widespread identity system is already contemplated by EU regulations. On top of that, the PKI infrastructure offers a property which is required by the SSI. A relying party can easily verify a subscriber identity through its certificate without needing to communicate with the third-party CA ensuring decentralization and autonomy, thanks to the certificate chain and the properties of digital signatures. For these reasons, the PKI can be exploited for the ZK-KYC-DSIG framework. Digital signature also exhibit another key property which is the non repudiation property. It ensures that the signature is uniquely and undeniably tied to both the signer private key and the content of the document.
Cryptographic Message Syntax (CMS) establishes a syntax for digitally signing, authenticating, digesting, or encrypting arbitrary messages [2]. It builds upon the PKCS#7 version 1.5 standard and supports PKI adopting X.509 certificates, which are the ones used on the web. Several extensions of the CMS exist, among them the CAdES-BES format is the most widely adopted and satisfies EU legal requirements. Data is structured according to a standard notation called ASN.1, which serves as a framework for defining data types. It supports common data types such as integers, strings, sets and sequences as well as more complex data types like OIDs which are sequences of integers that act as universally unique identifiers. Various encoding rules exist but the ones most commonly used in X.509 certificates are undoubtedly BER and DER. DER, by far the more prevalent, consists of straightforward binary values. An example is provided in figure 2.

```
EncapsulatedContentInfo ::= SEQUENCE {
  eContentType ContentType,
  eContent [0] EXPLICIT OCTET STRING OPTIONAL }

SignerInfo ::= SEQUENCE {
  version CMSVersion,
  sid SignerIdentifier,
  digestAlgorithm DigestAlgorithmIdentifier,
  signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature SignatureValue,
  unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }
```

Figure 2: CMS data structure.

2.2. eIDAS2 regulation

The eIDAS regulations [3] recognizes the importance of privacy and decentralization in digital identity. Furthermore, it also provides legal recognition to technologies like electronic ledgers (e.g. DLTs and blockchains), which support the aforementioned decentralization as well as ZKPs, which enforce the cited privacy. The document also introduces the European Digital Identity Wallets (EUDIW) and the concept of Electronic Attestation of Attributes (EAA) which, in practice, are credentials issued by recognized authorities. Additionally, selective disclosure of these attributes is introduced to minimize the sharing of personal information as much as possible. These novel introductions are a big step toward the recognition of the SSI model. The system has been designed with EU regulatory compliance and SSI principles in mind and systematically verified at each development stage.

2.3. DLTs

In short, a blockchain is a set of sequentially ordered blocks containing transactional records which form a distributed ledgers. Key feature of blockchains are: Transparency, Autonomy, Immutability, Irreversibility and Auditability. These properties are particular useful for an identity verification framework. In 2015, Ethereum, a blockchain platform supporting

smart contracts and decentralized applications, has been publicly launched. The Ethereum blockchain implements a Turing complete virtual machine, known as the Ethereum Virtual Machine (EVM), enabling developers to write sophisticated smart contracts using the Solidity programming language. Ethereum is by far the the most commonly adopted blockchain technology. Consequently, since ZK-KYC-DSIG could contribute to future digital identity standards, support of technologies is a key parameter. Therefore Ethereum has been selected.

2.4. ZKP and ZK-SNARKS

ZKPs are a cryptographic primitive that allow one party, the prover $\mathcal{P}$, to convince another party, the verifier $\mathcal{V}$, that a given statement is true without disclosing any information beyond the validity of the statement itself [4]. Among the existing ZKP flavors, the best suited approach for on-chain verification are ZK-SNARKs due to their reduced proof size and fast verification time. In order to decide which ZK-SNARKs framework to adopt, the key parameters identified were: effective adoption, structured documentation, ease of adoption and possible proof generation in different environments. These considerations outweighed the pursuit of absolute performance, leading to favor a mature, popular approach over a purely optimal one. Namely, the last cited parameter revealed to be critical. Ultimately, the final verdict was in favor of the Circom + SnarkJS approach.

3. State of the art

Existing digital identity solutions that adopt DLTs and ZKPs usually commit deeply on SSI principles limiting their interoperability with established legal frameworks and existing infrastructures consequently hindering their widespread adoption. Conversely, other approaches who seek legal compliance end up relying on third parties losing adherence to the SSI foundation (e.g. delegating proof generation to third party services).

4. System Design and Implementation

4.1. Software product regime

In [5] the author assesses ZKP from a legal standpoint and concludes that the Software as a Product regimen aligns better with existing regulations: "in a pure product scenario of ZKP generation and validation, all aspects, including liability, cybersecurity, software quality, privacy, and market surveillance, are effectively addressed". The identified regime remains completely in line with the SSI principles. Thus, a mobile application offers the most effective solution for deploying ZK-KYC-DSIG. This reinforces the choice of the Circom + SnarkJS framework and also helps to decide which proof system to adopt since SnarkJs with Groth16 retains complete compatibility with Rapidsnark that not only reduce proving times but also comes with native support for proof generation in mobile app environment. Moreover, the Typescript language has been chosen due to its compatibility with SnarkJS, the numerous libraries to interact with PKI and blockchain and, last but not least, its great popularity.

4.2. Framework Objective and Workflow

ZK-KYC-DSIG aims to verify a user identity by proving with ZKP the ownership by the user of both a certificate issued by a trusted CA and a taxID. When issuing a certificate, CAs include within it the taxID of the applicant. Therefore, a signed document, due to the non repudiation property of digital signature, immediately confirm the signer identity. The generated proof can be used to authorize a specific blockchain address to interact with specific smart contract functions. Furthermore, the system also seeks to prove that the user taxID has been encrypted with a Legal Authority public key allowing this entity to retain access to personal information for legal purposes (e.g. KYC).
For clarity, the framework expected workflow is demonstrated through an example. Bob wants to access a decentralized financial service requiring KYC verification. If Bob does not already have a certificate, he must request one from a trusted CA. Then, he needs to initiate the proving process providing the taxID to the mobile application which returns to him a plain text file containing the ciphertext obtained with the Legal Authority public key. Afterwards, Bob digi-

tally signs the file and reload it inside the application which performs the pre-processing phase. The output of this phase is sent to the digital circuit which computes the proof. Once the proof is ready, Bob can share it with the smart contract of the decentralized application for verification. Upon success, Bob blockchain address becomes verified and it is allowed to operate inside the smart contract.

4.3. Inputs

The system needs three main inputs (illustrated in figure 3). In the prototype, they are all provided by the user.

- **Digitally Signed File**: A file digitally signed by the user in the CAdES-BES format. It must be provided directly by the user and the content must consist of encrypted personal data combined with a random salt. The ciphertext must be generated with the Legal Authority public key and encoded in Base64 format.
- **Certificate of the CA**: A classic X.509 public key certificate of the CA that issued and signed the certificate of the user (the one used for the signature). It can be recovered from the device trust store.
- **Certificate of the Legal Authority**: A certificate containing the Legal Authority public key. The user encrypts the structure containing his personal data with this key. It can be included directly in the mobile application or in the device trust store.

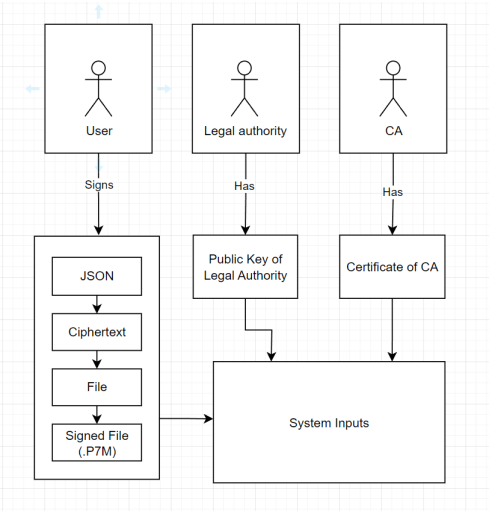

Figure 3: ZK-KYC-DSIG inputs

The framework requires the user to deliver a digitally signed plain text file. Non plain formats such as PDF incorporate complex structures and may apply compression algorithms (e.g. deflate) to the content. At present, no available Circom gadget handles such formats due to the significant computational overhead it would incur. Therefore, files that are not in plain text format would render the content incompatible with a circuit.

4.4. Architecture

The system architecture is composed of three primary components.

- Pre-Processing Layer
- Proof Generation Circuit
- Proof Verification Contract

The mobile application, not implemented in the current prototype, would provide a user interface and interconnect these three elements.

4.4.1 Pre-Processing Layer

The TypeScript Pre-Processing Layer processes raw system inputs into optimized data for the Proof Generation Circuit component. It performs all operations not strictly required by the proof, thereby reducing the proof generation computations. The output JSON structure serves as the input for the ZKP proof. Critical data is extracted by parsing the CAdES-BES signature, the Legal Authority certificate and CA certificate which are encoded in ASN.1 BER and DER formats. Furthermore, the Circom language accepts only integers and arrays of integers as input signals. Hence, the required format conversions are also performed.

4.4.2 Proof Generation Circuit

The ZKP Circom circuit invokes several templates for different operations and employs gadgets from popular and audited libraries. It receives pre-processed data from the Pre-Process layer to offload as much as possible the computational complexity of the ZKP process.

4.4.3 Proof Generation Steps

The ZK-KYC-DSIG ZKP circuit proves user ownership of a specific taxID while establishing a trust link between the relying party and a trusted CA via the certificate chain. (The prototype is currently limited to a certificate chain

with a single root CA.) To generate a valid proof, the circuit enforces the following steps.

- Verify the signature of the digitally signed file (i.e. padded hash of signed attributes matches the digital signature verified with the user public key).
- Validate the user certificate signature (i.e. padded hash of the "To Be Signed" (TBS) data matches the certificate signature verified with the CA public key).
- Confirm the message digest (i.e. hash of the file content equals the message digest).
- Check the encrypted content (i.e. the content of the file matches a data structure containing the taxID encrypted with the Legal Authority public key).
- Match the taxID in the encrypted content with the taxID contained in the certificate TBS.

4.4.4 Proof Verification Contract

The Solidity verifier smart contract has been auto generated by the SnarkJS library tool and subsequently deployed on an EVM compatible blockchain test environment. It confirms proof correctness, registering successful identity verification for the transaction sender address. Additionally, a simple test scenario, ZKIdentityVault has been developed where users deposit funds in a vault and are able to withdraw them after successful identity verification. Security in ZKIdentityVault has been ensured through the OpenZeppelin library. Consequently, compatibility with financial operations has been demonstrated as well.

4.5. Ciphertext storage

ZK-KYC-DSIG aims to be suitable for KYC operations as well. For this reason, the Legal Authority needs to have access to the cipher text of users personal data. The prototype has been tested in two versions:

- **Large Input** version: the public signals contain the cipher text of personal data. Despite the encryption, GDPR and related data protection regulations may still conflict with this approach due to immutability of blockchain and the "Right to be Forgotten".
- **Small Input** version: the public signals do not contain personal data but an ad hoc solutions must be devised to make the Legal Authority retain access to identification data.

The legal authority is entrusted with the critical responsibility of safeguarding a vast amount of sensitive personal data. To mitigate the risk of key compromise or unauthorized access, a multi-signature system should be adopted.

5. Results and Evaluation

The experiments were conducted on two different hardware setups, an older one *Intel i7-4790K CPU and 16GB of DDR3 RAM* and a newer one *Intel i9-13900K CPU and 64GB of DDR5 RAM*. The older system should be closer to a modern mobile device.

5.1. Proof generation time

Proof generation times are shown in figure 4. The experiments include independent runs using both real input data and synthetically crafted data. Overall, the results are in line with the proposed goals and the designed tests.

Proof Generation	Older Machine (s)	Newer Machine (s)
Real Data 0	78.107	39.564
Real Data 1	124.057	45.173
Real Data 2	109.893	41.044
Crafted Data 1	102.806	40.369
Crafted Data 2	115.517	40.690

Figure 4: Proof generation time.

5.2. On chain verification time

The two versions of the framework, Small Input and Large Input, have been tested for on chain verification costs. The test have been conducted on the REMIX IDE platform. The Large Input version has shown a transaction cost of 2,814,320 gas while the Small Input one only of 477,236 gas. Further tests in the Hardhat environment produced comparable results. Therefore, the Small Input version is significantly cheaper than the alternative. Gas fees costs are deeply influenced by the size of the public input and output data. Although deploying on Ethereum mainnet would result in prohibitively high costs, the use of Layer 2 EVM compatible chains significantly reduces gas fees. Figure 5 contains the gas fees equivalent conversion in euros as of February 20, year 2025.

EVM Blockchain	Cost (Large Input) (EUR)	Cost (Small Input) (EUR)
Ethereum	14.6278	2.4815
Optimism	0.7314	0.1241
Avalanche	0.0659	0.0112
Polygon	0.0211	0.0036

Figure 5: Gas fees costs.

6. Future development

Beyond the development efforts needed to transition from prototype to a fully deployable solution, a few key strategic advancements are proposed to tackle pivotal challenges.

6.1. Post Quantum Computers security

The underlying cryptographic assumptions of traditional Groth16 SNARKs are not post quantum secure. The potential advent of large scale quantum computers may pose another security vulnerability for systems adopting this ZKP technology.

6.2. Performance and Scalability

Gas fees are closely related to the size of the transaction input as showed in section 5. Hence, it would be possible to further reduce gas fees by reducing the dimension of the public inputs and outputs of the circuit. A straightforward method, would be passing the hash of both the Legal Authority public key and the CA public key. Unfortunately, this process could add computational complexity to the prover circuit. Furthermore, although proof generation times meet the original objectives and compare favorably with other approaches and although the adoption of Rapidsnark should significantly increase performance, the absolute duration remains non negligible already. A first improvement lies in following usability principles, reducing user perceived time. According to usability engineering, long operations should ideally complete within ten seconds and users should receive timely feedback of what is being executed. If an operation drags further than that, the option to cancel it should be provided. Though, thanks to the mobile environment, a more practical approach could be to allow proof generation to continue in background. After that a real performance improvement should be targeted. The proposed circuit may be scrutinized for additional refinements. However, since it was developed with performance as a design objective, substantial gains in this area are likely to be limited. Thus, enhancing the efficiency of the proof generation process itself, especially for cryptographic operations, represents the most viable options.

7. Conclusions

This work contributes to the evolving field of digital identity management, presenting the ZK-KYC-DSIG framework which has been analyzed from multiple perspectives. The proposed prototype, presented in two versions, successfully generates a proof of an individual identity through a digitally signed file without disclosing any information other than the proof itself. The proof generation times are reasonable. A verifier can confirm the proof on-chain with satisfactory gas fee costs, especially in one of the proposed versions. Regarding regulations, the project aligns with eIDAS 2 by design, demonstrating that SSI principles and regulatory compliance can coexist. Moreover, it has been confirmed that the optimal legal framework for the product would be the Software as a Product model, which is better suited for systems where third-party involvement must be avoided. Despite some non critical limitations, the framework demonstrated great potential and it could serve as a solid foundation for a future standard implementation.

References

[1] Uri Rivner. Identity crisis, detecting account opening fraud in the age of identity commoditisation. *Henry Stewart Publications*, 1(4):316–325, 7 2018.

[2] Russ Housley. Cryptographic Message Syntax (CMS). RFC 5652, September 2009.

[3] The European Parliament and the Council of the European Union. Regulation (eu) 2024/1183 of the european parliament and of the council of 11 april 2024 amending regulation (eu) no 910/2014 as regards establishing the european digital identity framework, 2024.

[4] Nojan Sheybani, Anees Ahmed, Michel Kinsy, and Farinaz Koushanfar. Zero-knowledge proof frameworks: A survey, 2025.

[5] Raül Ramos Fernández. Evaluation of trust service and software product regimes for zero-knowledge proof development under eidas 2.0. *Computer Law & Security Review*, 53:105968, 2024.