zkTax: A pragmatic way to support zero-knowledge tax disclosures

Anonymous for review

ABSTRACT

Tax returns contain key financial information of interest to third parties: public officials are asked to share financial data for transparency, companies seek to assess the financial status of business partners, and individuals need to prove their income to landlords or to receive benefits. Tax returns also contain sensitive data such that sharing them in their entirety undermines privacy. We introduce a zero-knowledge tax disclosure system (zkTax) that allows individuals and organizations to make provable claims about select information in their tax returns without revealing additional information, which can be independently verified by third parties. The system consists of three distinct services that can be distributed: a tax authority provides tax documents signed with a public key; a Redact & Prove Service enables users to produce a redacted version of the tax documents with a zero-knowledge proof attesting the provenance of the redacted data; a Verify Service enables anyone to verify the proof. We implement a prototype with a user interface, compatible with U.S. tax forms, and demonstrate how this design could be implemented with minimal changes to existing tax infrastructure. Our system is designed to be extensible to other contexts and jurisdictions. This work provides a practical example of how distributed tools leveraging cryptography can enhance existing government or financial infrastructures, providing immediate transparency alongside privacy without system overhauls.

CCS CONCEPTS

• **Applied computing → Information integration and interoperability**; **Electronic data interchange**; • **Security and privacy → Cryptography**; • **Human-centered computing** → *Open source software*.

KEYWORDS

private data sharing, zero-knowledge, financial disclosure, decentralization, cryptography, prototype

1 INTRODUCTION

Tax documents aggregate a wealth of information about an individual's finances, ranging from income sources to charitable contributions. Upon submitting these documents to government agencies, individuals declare the information is true and accurate to the best of their knowledge. Similarly, organizations – ranging from nonprofits to corporations – conduct financial audits and report their financial data through tax filings.

The information contained in tax returns, while primarily intended for tax-filing purposes, is often useful across a broad range of contexts. As a key motivating example, public officials can release tax details in the pursuit of increased accountability and transparency. Further, data derived from tax records can be instrumental in assessing eligibility for benefits, such as welfare benefits for low-income households [25, 31]. Additionally, in legal proceedings such as divorce or child support negotiations, financial attestations are often substantiated through the use of this information [25, 31].

Yet, can these requests for financial records be satisfied in a verifiable way without exposing other private data present in tax records? Trustworthy mechanisms to verify the accuracy and authenticity of financial attestations are required.

We present a system to address this problem by leveraging zero-knowledge and public key cryptography. Zero-knowledge (ZK) in this context refers to the ability to computationally prove a statement is true, without exposing the inputs to the statement. The system allows individuals to produce a redacted version of their tax documents such that the information can be verified by any third party. System users may either send the redacted documents to inquirers or publish the redacted versions for anyone else to publicly verify. Simple extensions of this system allow users to prove more complex statements about their tax returns. This allows organizations and individuals to prove their financial information to third parties without revealing sensitive data.

1.1 Motivating Example

Although we present multiple use cases for our system in section 5, we start by highlighting a use case that will follow throughout demonstrating the system.

This system is particularly motivated by the ability to leverage cryptographic tools to improve transparency and accountability in governance systems, of which the common practice of public officials releasing redacted versions of their income tax returns is central. For example, in the US, presidents have historically released their tax returns in a tradition of transparency. Proponents of measures to codify this act of transparency argue that public disclosure of tax returns could expose conflicts of interest, reveal annual tax liability and tax rates, and enable the public to observe whether the President or candidates have engaged in tax evasion, or pursued tax avoidance [6]. Yet, there have been recent cases where public officials have refused to release their taxes, despite demands from

their own constituents and the general public [10]. This work is motivated by the idea that if a zero-knowledge tax disclosure mechanism were available, this refusal to release requested information may not be necessary or defensible. In addition, the information disclosed could be better trusted through verification.

Consider a scenario where citizens inquire about just one field from a public official's tax forms, such as the total amount of tax paid. The official provides the requested amount, but refuses to release their tax returns due to privacy concerns. The citizens may then question whether the disclosed amount is truthful. The ZK tax disclosure system presented here offers a solution, allowing the official to provide verifiable proof of the specified value from their tax form without exposing sensitive information. This balance of privacy and transparency can potentially enhance public trust in governance systems through the application of Privacy by Design principles.

2 BACKGROUND AND RELATED WORK

The system proposed here builds on both redactable signature schemes and recent advances in zero-knowledge proof tools. Here we elaborate on the foundational material of this work and contrast it to alternative systems that have been proposed.

Zero-knowledge proofs are a way to prove the validity of a statement without revealing any additional information beyond the truth of the statement itself [13]. Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge (zk-SNARKs) [14] have become a flexible ecosystem of proof generation software, with general-purpose high-level languages like circom [4], which compiles down to arithmetic equations called 'circuits'. These circuits can then be used by powerful proving systems such as groth16 [15] and halo2 [8] to produce small digitally verifiable proofs.

This work draws on these zk-SNARK advancements and their application to the financial system. These applications have focused on the blockchain space, such as their use in achieving anonymous transactions [5, 28]. ZK proofs have also been used to achieve anonymous e-taxing protocols (using blockchains) [26], provable audits (zkLedger [24]), and proof of reserves [9]. Other related work has also used blockchains to add privacy-preserving tools to tax systems, even without zero-knowledge proofs [22]

The works above assume a closed financial system where transactions occur on a blockchain. This is not today's reality. Unlike these prior works, our system does not assume a blockchain and can be implemented to work alongside and improve current tax and financial infrastructures.

ZK tools such as this work can play a powerful role in creating new user centric privacy pattern selections [1]. Indeed, since the introduction of GDPR there has been an increase in Privacy by Design research from a regulation-centric perspective [11], but user-centric solutions still need research. While important work such as decentralized access control in blockchain-based systems [20] has already been done in this space, the work we present here provides a Privacy by Design approach where less technical users can be in control of their privacy tradeoffs while interacting with systems designed for the general public.

Other cryptographic approaches to providing verifiable redacted data have been taken, such as with content extractable signatures (CES) [33] and redactable signature schemes (RSS) [32]. In terms of our use case, we might consider a tax form as a set of messages, where each form field and its associated value is a message that can separately have an attached signature. A verifiable redaction scheme may then publish a subset of these signed fields. Works studying CES and RSS improve efficiency over providing a signature for each such message. Yet these approaches suffer from drawbacks where either the size of the signature or key must scale with the size of the original message (e.g. original tax document). While this is not the case for our solution with zk-SNARKs, we note that the size of the circuit in our solution does grow with the allowed original tax document(s) size. However, unlike with CES and RSS, this burden does not then fall on the services that handle signing and is instead distributed. Furthermore, unlike CES and RSS, our flexible approach with zk-SNARKs can be easily extended beyond simple redaction to prove relations between form field values while avoiding publishing the values themselves.

A more similar approach to this work is the use of zk-SNARKs to redact images by Ko et al [23]. They used a combination of zk-SNARKs to produce a "public" and "private" version of the image, and utilized the additive properties of a Pederson commitment scheme so that a verifier could verify that commitments on these two versions added up to a signed commitment from the original version. Given the complexity of their approach, signature verification was handled outside of a zk-SNARK, unlike our implementation. This process was designed specifically for images as represented by pixels, while our simpler approach can leverage the structured content of a tax form.

Our system assumes the existence of a digital service that uses public key cryptography to sign tax documents. We note this assumption is not far-fetched given the following work. In 2005 Mexico took on an ambitious public key infrastructure project which allows citizens to sign tax declarations [30], and Estonia's modern government infrastructure allows for a variety of digital signing including with taxes [12]. In general, the global push towards Open Banking [29] beckons a possible future where many financial providers will allow users to access permissions and signed attestations about financial products and balances.

3 ZKTAX SYSTEM

3.1 System goals

To enable and enhance usability the system should achieve the following.

- *Interoperability with existing tax services*: Building on top of existing taxation infrastructure minimizes the burden for tax administrators while facilitating adoption.
- *Non-interactive from the perspective of the tax service*: Tax service providers (e.g. government agencies) should sign tax data once to enable users to generate verifiable attestations from this data; minimizing costs for service providers and mitigating likely bottlenecks.
- *Public verifiability*: Third parties should be able to independently verify the redacted tax information. This should be user-friendly for end-users without a background in cryptography.
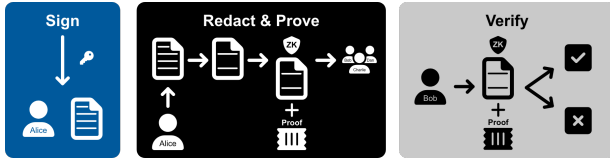
Figure 1: Diagram of the three services that compose the ZK tax disclosure system. (Left) An individual, Alice, retrieves tax data from a Trusted Tax Service (TTS), which signs the tax data with a signature that can be verified with a public key. (Middle) Alice brings the signed tax data to the Redact & Prove service. The service allows her to select how tax data should be redacted and then generates a redacted version and proof that the original data was signed by the TTS. Alice can make the proof and redacted data public for anyone to then verify by using the Verify Service.

- *Flexibility and extensibility in terms of what can be proved*: The system should be adaptable to accommodate different user desires for combinations of attestations and future extensions.
- *Modularity*: A modular design simplifies the system's maintenance, upgrades, and potential integration with other systems.
- *Privacy by design for the taxpayer*: The system should not reveal unauthorized private data to any third party except the trusted tax service and protect against re-identification attacks.
- *Unforgeability*: Individuals should be unable to fraudulently claim that redacted tax information was sourced from real tax information handled by a tax service.

3.2 Architecture

To achieve the above goals, we propose a system consisting of three distinct services. A high-level view of this architecture is illustrated in Figure 1. Figure 2 provides further detail of the system components and how data flows between them. This section describes the proposed architecture at a high level, where the specifics of implementation can vary. Section 4 describes our prototype implementation in further detail.

Table 1 summarizes the distinct participant roles that correspond to the 3 services.

In short, zk-SNARK circuit used to generate and verify proofs is publicly shared among the Redact & Prove and Verify Services. A taxpayer receives their signed and machine-readable tax data from the Trusted Tax Service (TTS), and then uses the Redact & Prove Service to create a redacted version of their tax data along with a proof of its validity. This redacted data and proof can then be verified by any third-party using the Verify Service.

3.2.1 Trusted Tax Service (TTS). Our system assumes that there is a trusted authority – which we refer to as the Trusted Tax Service (TTS) – to which individuals submit their taxes. Such an entity might be the government or a trusted tax preparer that submits taxes to the government on an individual's behalf. Possible examples in the American tax system include the IRS, TurboTax, and

H&R Block. Our proposal assumes these services have the ability to transform the tax data they process into a machine-readable format (e.g. JSON), have a known public key, and can return the machine-readable version of the accepted tax data, along with a signature and their public key, so that other entities can verify the tax data was approved by them. We note the TTS assumed in this work is not currently implemented, yet is reasonable in scope. The IRS published a multiyear modernization plan in 2019 that described a taxpayer payment Application Programming Interface (API) to support WebApps [19]. Intuit currently provides an API [18] for third-party developers to build apps using QuickBooks data in order to publish apps to the QuickBooks app store.

While ZK proofs may not be in the scope of software that these trusted entities will provide in the immediate future, our proposed system allows others to use the signed data returned from the TTS to build ZK proof systems that enable the privacy-preserving disclosures described. Although this approach does rely on existing entities returning signed tax data, we believe this kind of technical progress to be more possible in the short run than alternative solutions which have suggested complete replacement of existing tax services (such as with a distributed ledger [27]).

In our proposed system, we assume the TTS provides a taxpayer's complete and unredacted tax information as JSON (x). The TTS uses the hash-then-sign paradigm by using a hash function (H) to produce a hash of the user's complete tax information $H(x) = y$ and then signs y using the TTS secret key (sk) as $S = sign(sk, y)$. The TTS returns x and S to the user. Note this data is private to the taxpayer, as it still contains the unredacted tax information, x. H and the TTS public key, pk, are publicly known.

3.2.2 Redact & Prove Service. A taxpayer takes the tax data (x) and corresponding signature (S) output from the TTS to a Redact & Prove Service as inputs. This service may be run by any entity, but does require a certain level of trust because the input to the front end of this service is private tax data (x). This service can and should work by operating locally within the user's machine, to avoid sending private data to a server. This can be done by compiling all assets (including files to make proofs from the zk-SNARK circuit) beforehand and serving these files to the user's browser.

The Redact & Prove Service operates in 2 steps. An interface allows the user to insert their tax data from the TTS and select where redactions should be made, producing an intermediate output: redact data. The proof generation step then takes 2 inputs: The output (x, S) from the TTS and the redact data. These inputs are provided to a zk-SNARK circuit that does the following. The circuit first verifies that the input tax data, x, matches the signature, S, by checking the hashed tax data against the TTS signature and public key: $verify(H(x), pk, S)$. If this verification fails, an error is returned to indicate that the tax information x and signature S did not match. Otherwise, if the tax data x are verified as valid, the circuit redacts all data indicated by the redact data input from x to produce x'. The circuit output is both the redacted version of the data, x', and a proof, $\pi_{x'}$. The Redact & Prove Service returns $\pi_{x'}$, x', S, and pk so that others can check the validity of x' (shown as the Reject/Accept output in Figure 2). The original unredacted tax data, x, is the witness for the proof.

	System participants		Services	
	Name	Description	Name	Description
1	Trusted tax authority	Entity that already has access to taxpayers' tax data. E.g. in the U.S. this may be the IRS, or a company that processes taxes and sends them to the IRS on behalf of taxpayers (e.g. TurboTax).	Trusted Tax Service (TTS)	Returns tax data to individuals, where the data is signed and can be verified using a public key published by the TTS.
2	Individual taxpayer	Individual who retrieves their signed tax data from the TTS and uses the Redact & Prove Service to redact fields from the data and produce a ZK proof about the redacted data. The full tax data is private to them. They share the redacted data.	Redact & Prove Service	Shares a zk-SNARK circuit with the Verify Service. Provides an interface for taxpayers to privately select fields to redact from their tax data and produces a proof of the redacted data by using the zk-SNARK.
3	General public or auditor	Any entity who wishes to verify a claim about the redacted tax data. They take the shared redacted tax data and proof to a Verify service.	Verify Service	Shares a zk-SNARK circuit with the Redact & Prove Service. Provides interface for anyone to take shared redacted tax data and corresponding proof, output by the Redact & Prove Service, to verify.

Table 1: Overview of the system participants and their corresponding services in the proposed tax verification system.

3.2.3 Verification service. The verification service allows for anyone to verify that the redacted tax data, x', did indeed originate from some unredacted tax data, x, that was signed by the TTS with with public key pk. It does so by compiling the public circuit and verifying the compiled circuit, proof data (π'_x), and redacted data (x') are compatible. Since the circuit code includes verifying the signature with pk on the full tax data, x, this also means that the redacted tax data must match what the TTS returned.

4 PROOF OF CONCEPT IMPLEMENTATION WITH U.S. TAX FORMS

We developed a proof of concept implementation which includes a public web app[1], a set of reference circuits, and tax form processing code. All code for the prototype is open source[2].

The web app is implemented using React. We implement the ZK proof and verification by leveraging the Circom software ecosystem [4] to write and compile R1CS circuits. We use the snarkjs javascript library [17] to handle the trusted setup, to produce the circuit artifacts used for proof and verification, as well as for implementing proof and verification within the user's browser.

4.1 Example Use Case

To demonstrate how this system can be used, we return to the motivating use case where a public official wishes to make a provable statement about specific information from their tax returns, without sharing other information. In this example we use real data from a public official's released tax forms[3]. We assume the official wishes to redact all information except "Taxable income" (field 15 on Form 1040) and their name. The resulting redacted form is shown in the Appendix Figure 6.

4.2 Data representation

Tax forms are often PDFs. We convert PDFs to and from a flat JSON representation, where form fields map to values, with metadata for

form name and year assumed to be added by the TTS. The JSONs are converted to ASCII strings as input for the zk-SNARK circuit.

```
json: {
// all data from the tax form
. . .
"fname": "name",
"lname": "name",
"SSN": "000000000",
. . .
"f_1": "393,229", // represents field 1
"f_2a": "2,208", // represents field 2a
. . .
// additional metadata
"year": "2020",
"form": "1040"
}
```

In our interface, these machine data representations are converted into human-readable JSONs and filled PDFs.

Our circuit uses an array of bits to represent the redaction data produced by the taxpayer using the Redact & Prove Service, where each element in the array indicates whether the corresponding element in the stringified JSON ASCII array should be redacted or not.

4.3 zk-SNARK circuit

The zk-SNARK is implemented as an R1CS circuit with the following simple logic: the JSON input is hashed and the circuit utilizes the TTS public key (referred to as "pubkey") to assert that the input signature components correspond with the hashed JSON. If the signature verification is unsuccessful, a proof is not generated. Given a valid signature, the circuit yields an output representing the redacted JSON. This output aligns with the input JSON except where indicated by the redaction map bit array, which points to the corresponding elements set to whitespace.

We use EdDSA/MiMC-7 for computing hashes and signatures [3]. The MiMC hash function is chosen for its efficiency within zk-SNARK circuits, which is due to its low multiplicative complexity [2]. We note consistent hash and sign functions must be used within the circuit and TTS.

---

[1]https://18.27.79.206/ (converted to IP address for blind submission, certificate error may occur)
[2]http://REDACTED
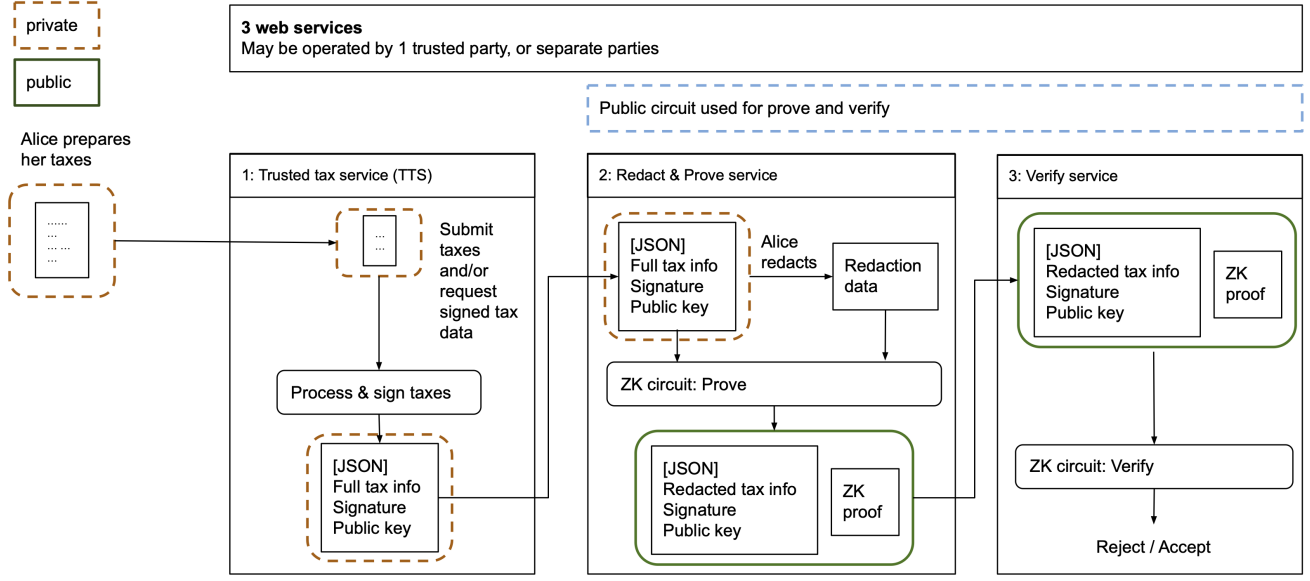[3]Original form available at https://REDACTED

Figure 2: ZK Tax system overview with data flows.

Groth16 is used as the proving system [15] requiring a circuit trusted setup which is handled by using values computed from the Perpetual Powers of Tau ceremony [21].

Once run, the Redact & Prove setup produces a computational `proof.json` and a `signals.json`, the latter of which contains the redacted JSON string as ASCII and the signer public key.

4.3.1 Circuit size complexity. The circuit is parameterized by the maximum JSON string input size such that the size of the circuit grows with the size of the input. For larger circuit sizes, the compilation needs larger Powers of Tau, and produces larger verification files. Our prototype uses a maximum JSON size of 1500. Our open source code provides files for other input sizes.

4.4 Prototype web app

Our initial implementation is based on a US income tax form: the 2020 IRS Form 1040 provided [4]. The web interface for each service is shown in the appendix. Our implementation includes a proof-of-concept TTS to which a user may upload a completed Form 1040 which is parsed into JSON, stringified, and hashed using MiMC before an EdDSA signature is generated from `circomlibjs`[5] which is returned, along with other necessary outputs, for the Redact & Prove Service.

The redact & prove service interface generates the redaction bit array based on the user's selection of fields and the TTS output. Note that this service is privately used by the taxpayer, given the input information contains all of their private tax data. The zk-SNARK circuit files, which are used for redaction, are precompiled using snarkjs and served to the frontend. snarkjs is also used for the groth16 proof computation. All operations are done within the

client browser. The user can download these proof artifacts and share them as they contain no private data.

Similarly, the prototype Verify Service precompiles a verification key from the circuit via snarkjs using groth16 that is served to the service's frontend. Users can provide the `proof.json` and `signals.json` output from the proof. The verification key is then used to verify that the signals and proof are compatible, thereby accepting or otherwise rejecting the proof.

5 DISCUSSION

5.1 Extensible circuit logics

This prototype focuses on using zero-knowledge proofs to prove that redacted tax data originated from official signed tax forms. While this functionality has numerous applications, extensions to this prototype, such as range-proofs and various logical assertions, could also be encoded into R1CS circuits. Zero Knowledge Range Proofs (ZKRP), such as proposed by Boudot [7], could allow a user to assert that a value, such as total taxes paid, falls within a given range. Meanwhile, circuits can be designed to support a wide array of logical assertions, including but not limited to, statements like 'the value of field A is greater than that of field B', or 'the charitable donations made exceed a certain percentage of the income earned'. Services 2 and 3 have the capacity to incorporate ZK circuits that are tailored for each of these specific scenarios. In-line with our system goals, our implementation of a redaction service and circuit is aimed at generalizability and does not require specific fields to be hard-coded. The resulting flexibility enhances our ability to adapt to different use-cases without necessitating substantial alterations to the underlying circuitry.

---

[4]https://www.irs.gov/pub/irs-prior/f1040--2020.pdf
[5]https://github.com/iden3/circomlib/blob/master/circuits/eddsa.circom

5.2 Example Use Cases

Extending beyond the general prototype example and its application to a public official's disclosure, we highlight two additional contexts where zkTax allows individuals or organizations to prove financial information from their tax returns to third parties without sharing sensitive data.

5.2.1 Efficient and verifiable individual tax data attestations. Individuals may need to prove their income or other financial information to potential landlords, employers, and lenders. To this end, they can rely on tax authorities (e.g. IRS) as a TTS and third-party providers to redact and prove their tax return information. The individual can freely share authenticated financial data and associated proofs with any requester (e.g. landlord or bank), who can confirm the authenticity using any third-party verification service.

In this way, zkTax approaches provide a trustworthy and privacy preserving way to share financial data, and when combined with range proofs, obviating the need for exact numbers to be shared directly, instead proving that a given financial figure is above or below a specific threshold.

In addition to individuals and corporations attesting their financial information, this system could streamline the process of eligibility checking for government benefits. Governmental systems often separate data to maintain privacy, meaning that a welfare or benefits agency may not have access to original tax records. For welfare recipients who file tax returns, the proposed tool could make it easier to prove eligibility for benefits.

Not only does this approach minimize the amount of private information that is needlessly transmitted, it also reduces the cost of verification for the recipient, and eliminates the risk of human errors during verification.

5.2.2 Verifying business counterparty risk. Business transactions often involve counterparty risk —- the risk that the other party in a deal will be unable to fulfill its contractual obligations. Assessing counterparty risk generally involves reviewing financial data, but this may expose proprietary business information to competitors.

A business can rely on their tax auditors (e.g., EY or KPMG) to serve as its TTS. When they enter into a new transaction, the counterparty can ask the business to share certain financial items from its tax return. In many cases, it will be sufficient for the business to provide range proofs or to prove relationships between different financial items (e.g. liquidity ratio or accounts payable days). Using the zkTax approach, a business can assemble the requested information and associated proofs to put its transaction partner at ease while minimizing unnecessary disclosures that could impact its competitiveness.

5.3 Implementation requires minimal system overhaul and can be distributed

Our implementation is a single web app to provide all three services, Only (1) the trusted tax service (TTS) that accepts and signs the taxes needs to be operated by a specific trusted tax authority (that is already entrusted with sensitive tax information). The (2) Redact & Prove and (3) Verify services can be operated by one or several other entities. This flexibility is a primary advantage of the modular system architecture we propose. We believe it is a reasonable expectation for a trusted tax service, such as the IRS, to return signed JSON representations of individuals' accepted tax data to those individuals. Yet whether or not they also choose to provide zero-knowledge proof and verify services, such as those described in this work, does not matter for the system's feasibility. Any other entity can then implement services (2) and (3).

5.4 Limitations

A limitation of the proposed system is its reliance on a Trusted Tax Service (TTS) to sign tax documents. Many tax authorities currently do not have mechanisms for digitally signing individual tax documents in a manner compatible with our system. While the architecture is designed to minimally impact existing tax infrastructure, this requirement may pose an adoption hurdle. It is conceivable that interim solutions could be deployed, such as using email signatures from tax authorities [16] as a form of less-secure but more readily available authentication.

Second, this work does not incorporate a user study involving real tax data to understand individual expectations and demands for privacy. Such a study would be essential for capturing the nuances of user interaction with the system and evaluating its practical usability. Moreover, the system's acceptance also hinges on the willingness of third-party requesters to rely on zero-knowledge proofs for data verification. This aspect remains an unexplored variable that could significantly influence the system's possibility of adoption.

6 CONCLUSION

We present a system that leverages zero-knowledge proofs and public key cryptography to address the challenge of verifying tax information without exposing sensitive data. Our system comprises three distinct services: a Trusted Tax Service (TTS), a Redact & Prove Service, and a Verification Service. Together, these services enable taxpayers to make provable claims about their tax data that can then be verified by third parties. The architecture requires minimal changes to existing tax systems, offers flexibility in its deployment, and can be extended to support a variety of claims about financial data.

We demonstrate the feasibility of our system through a publicly available proof-of-concept implementation. This work contributes to financial privacy and data sovereignty for individuals, enhancing political transparency and accountability through public officials' financial disclosures, and allows businesses to assess counterparty risk without requiring proprietary or sensitive information.

AVAILABILITY

The prototype with a public web interface is available at https://18.27.79.206/[6] All code for the prototype web app, zk-SNARK circuits, and tax form processing code, is available at http://REDACTED.

---

[6]Converted to IP address for blind submission; certificate error may occur.

REFERENCES

[1] Ala'a Al-Momani, Kim Wuyts, Laurens Sion, Frank Kargl, Wouter Joosen, Benjamin Erb, and Christoph Bösch. 2021. Land of the lost: privacy patterns' forgotten properties: enhancing selection-support for privacy patterns. *Proceedings of the 36th Annual ACM Symposium on Applied Computing* (2021).

[2] Martin Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. 2016. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I.* Springer, 191–219.

[3] Jordi Baylina and Marta Bellés. [n. d.]. EdDSA For Baby Jubjub Elliptic Curve with MiMC-7 Hash.

[4] Marta Bellés-Muñoz, Miguel Isabel, José Luis Muñoz-Tapia, Albert Rubio, and Jordi Baylina. 2022. Circom: A Circuit Description Language for Building Zero-knowledge Applications. *IEEE Transactions on Dependable and Secure Computing* (2022).

[5] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. *2014 IEEE Symposium on Security and Privacy* (2014), 459–474.

[6] Joshua D. Blank. 2021. Presidential Tax Transparency. https://yalelawandpolicy.org/presidential-tax-transparency. Accessed: 5-16-2023.

[7] Fabrice Boudot. 2000. Efficient Proofs that a Committed Number Lies in an Interval. In *International Conference on the Theory and Application of Cryptographic Techniques.*

[8] Sean Bowe, Jack Grigg, and Daira Hopwood. 2019. Halo: Recursive Proof Composition without a Trusted Setup. *IACR Cryptol. ePrint Arch.* 2019 (2019), 1021.

[9] Gaby G. Dagher, Benedikt Bünz, Joseph Bonneau, Jeremy Clark, and Dan Boneh. 2015. Provisions: Privacy-preserving Proofs of Solvency for Bitcoin Exchanges. *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (2015).

[10] Julie Davis. 2017. Trump Won't Release His Tax Returns, a Top Aide Says. https://www.nytimes.com/2017/01/22/us/politics/donald-trump-tax-returns.html. Accessed: 5-16-2023.

[11] Shirlei Aparecida de Chaves and Fabiane Barreto Vavassori Benitti. 2023. Privacy by Design in Software Engineering: An update of a Systematic Mapping Study. *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing* (2023).

[12] e-Estonia. [n. d.]. e-Tax. https://e-estonia.com/solutions/ease_of_doing_business/e-tax/. Accessed: 3-16-2023.

[13] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1985. The knowledge complexity of interactive proof-systems. In *Symposium on the Theory of Computing.*

[14] Jens Groth. 2010. Short Pairing-Based Non-interactive Zero-Knowledge Arguments. In *International Conference on the Theory and Application of Cryptology and Information Security.*

[15] Jens Groth. 2016. On the Size of Pairing-Based Non-interactive Arguments. *IACR Cryptol. ePrint Arch.* 2016 (2016), 260.

[16] Aayush Gupta. 2022. *ZK Email.* https://blog.aayushg.com/posts/zkemail

[17] iden3. 2023. zkSNARK implementation in JavaScript & WASM. https://github.com/iden3/snarkjs. Accessed: 3-16-2023.

[18] Intuit. [n. d.]. Intuit Developer API Reference. https://developer.intuit.com/app/developer/qbo/docs/api/accounting/all-entities/taxservice. Accessed: 3-16-2023.

[19] IRS. 2019. IRS Integrated Modernization Business Plan. https://www.irs.gov/pub/irs-pdf/p5336.pdf. Accessed: 3-16-2023.

[20] Kristof Jannes, Vincent Reniers, Wouter Lenaerts, Bert Lagaisse, and Wouter Joosen. 2023. DEDACS: Decentralized and dynamic access control for smart contracts in a policy-based manner. *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing* (2023).

[21] Koh Wei Jie. 2019. Announcing the Perpetual Powers of Tau Ceremony to benefit all zk-SNARK projects. https://medium.com/coinmonks/announcing-the-perpetual-powers-of-tau-ceremony-to-benefit-all-zk-snark-projects-c3da86af8377. Accessed: 5-16-2023.

[22] Espen Hammer Kjellstadli, Mariusz Nowostawski, Abylay Satybaldy, and Nader Aeinehchi. 2019. Privacy-preserving tax-case processing. In *2019 17th International Conference on Privacy, Security and Trust (PST).* 1–10. https://doi.org/10.1109/PST47121.2019.8949072

[23] Hankyung Ko, Ingeun Lee, Seunghwa Lee, Jihye Kim, and Hyunok Oh. 2021. Efficient Verifiable Image Redacting based on zk-SNARKs. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security.* 213–226.

[24] Neha Narula, Willy Vasquez, and Madars Virza. 2018. zkLedger: Privacy-Preserving Auditing for Distributed Ledgers. In *IACR Cryptology ePrint Archive.*

[25] Austin Nichols and Jesse Rothstein. 2015. The earned income tax credit. In *Economics of Means-Tested Transfer Programs in the United States, Volume 1.* University of Chicago Press, 137–218.

[26] Hui-Chong Niu, Ting Li, and Xiugang Gong. 2022. A blockchain-based certifiable anonymous E-taxing protocol. *PLoS ONE* 17 (2022).

[27] Hui-Chong Niu, Ting Li, and Xiugang Gong. 2022. A blockchain-based certifiable anonymous E-taxing protocol. *PLoS ONE* 17 (2022).

[28] Alexey Pertsev, Roman Semenov, and Roman Storm. [n. d.]. Tornado Cash Privacy Solution - Version 1.4. Accessed: 3-16-2023.

[29] Anshu Premchand and Anurag Choudhry. 2018. Open Banking & APIs for Transformation in Banking. *2018 International Conference on Communication, Computing and Internet of Things (IC3IoT)* (2018), 25–29.

[30] Luis Rivera-Zamarripa, Lil M. Rodriguez, Miguel Angel Léon Chávez, Nareli Cruz Cortés, and Francisco Rodríguez-Henríquez. 2019. Security Analysis of the Mexican Fiscal Digital Certificate System. *Computación y Sistemas* 23 (2019), 477–490.

[31] Dorothy Rosenbaum. 2013. The relationship between SNAP and work among low-income households. *Center on Budget and Policy Priorities* (2013).

[32] Olivier Sanders. 2020. Efficient redactable signature and application to anonymous credentials. In *Public-Key Cryptography–PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part II.* Springer, 628–656.

[33] Ron Steinfeld, Laurence Bull, and Yuliang Zheng. 2002. Content extraction signatures. In *Information Security and Cryptology—ICISC 2001: 4th International Conference Seoul, Korea, December 6–7, 2001 Proceedings 4.* Springer, 285–304.
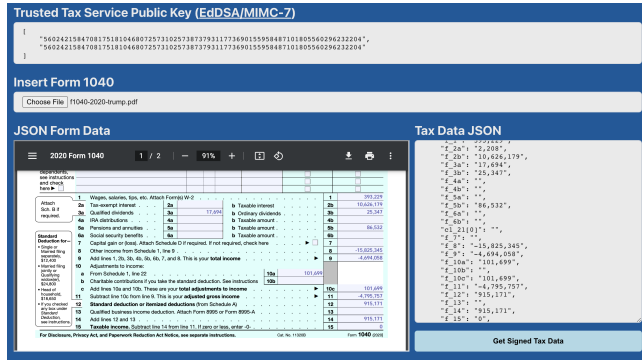
A PROTOYPE INTERFACES

Figure 3: Prototype mock TTS interface. Users can insert Form 1040 and the service returns the JSON representation of the data signed with the TTS key.
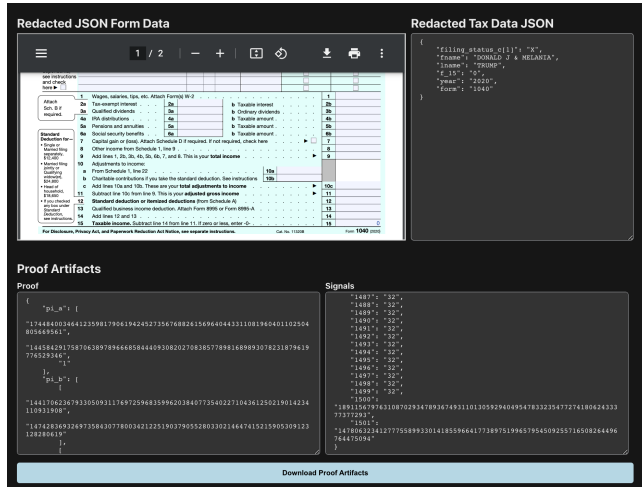

Figure 4: Prototype Redact & Prove Service interface showing fields relevant to the example use case provided.
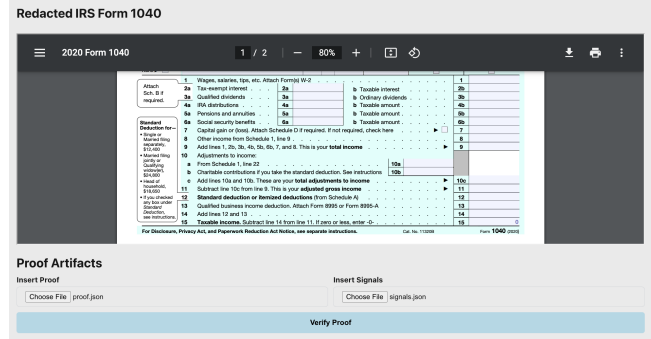

Figure 5: Prototype Verify Service interface showing fields relevant to the example use case provided.

Figure 6: Redacted version of U.S. income tax return Form 1040 for Donald Trump (2020 version), based on publicly released tax returns. All fields are "redacted" (set to blank whitespace) except the requested field 15 and his name (in order to verify his identity for the public release of the redacted version). Note that the form has more pages but the fields relevant for the example are on the first page, so only the first page is shown.