REPORT DRAFT FOR MASTER'S THESIS

Implementation and Evaluation of Programmable Money Using Verifiable Credentials and Zero-Knowledge Proofs

Yat Wai Wong

Supervisor:

Prof. Wolfgang Prinz, PhD
Prof. Stefan Decker, PhD

Date

RWTH Aachen University

Contents

1 Things to be included in the report

-Third party library cannot be controlled, e.g.,
sometimes metamask not working
sometimes privado not working
-Using Privado's web wallet to generate proof: no way to log proof generation time, e.g. no api for that, cannot change their code.
-proof generation seems slow. To track proof generation time on website, use selenium. May also track proof generation time on JS-SDK.
-using metamask vs transaction done on backend: users trust metamask vs trust our backend

Future work:
-find zkSNARK, anonymous credentials, with faster proof generation time
-post-quantum secure implementation with blockchain and benchmarking, e.g. lattice snark with public verifiable, lattice snark with designated verifier, lattice-based anonymous credentials

2 Zero-Knowledge Proofs

A Zero-Knowledge Proof (ZKP) [GMR85] is a protocol that allows one party (the prover P) to prove to another party (the verifier V) that a statement is true, without revealing any information beyond the fact that the statement is indeed true. In other words, the prover can convince the verifier that they possess certain knowledge or have performed a specific computation correctly, without disclosing the actual details of that knowledge or computation. Schnorr's protocol is one of the examples, and protocols having the structure of Schnorr's protocol (the '3 moves' structure) are called Σ protocols. The main properties of ZKP are completeness, soundness, zero-knowledge, and knowledge soundness. In some applications using non-interactive ZKP, a relaxed zero-knowledge (*special honest-verifier zero-knowledge*) suffices, and the non-interactive version of ZKP is achieved by Fiat-Shamir transformation [FS87] in random oracle model [BR93].

Using Σ protocol as an example, ZKP is defined as follows:

Definition 2.1 (NP Relation)**.** Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be an NP relation (it can

be evaluated in polynomial time), $x \in \{0,1\}^*$ be the statement, and $w \in \{0,1\}^*$ be the witness for the statement x. We define the language L_R for the relation as:

$$L_R := \{ x : \exists\, w,\ (x,w) \in R \}.$$

In other words, $x \in L_R$ iff there exists a witness for x. This can also be denoted as $(x,w) \in R$ or $R(x,w) = 1$.

Definition 2.2 (Proof System)**.** A proof system for R is an interactive protocol between a prover P and a verifier V for the relation R. The interaction between P (with x, w as its input) and V (with x as its input) is denoted as $\langle P(x,w), V(x) \rangle$. After interaction, V outputs a bit b to decide whether to accept or reject, denoted as:

$$\text{accept or reject} \leftarrow \langle P(x,w), V(x) \rangle$$

.

Definition 2.3 (3-move structure)**.** Σ-protocol follows a specific 3-move structure:

Commitment: The prover computes an initial message, the commitment t, and sends it to the verifier.

Challenge: The verifier generates a random challenge *chal* from a challenge space ($chal \xleftarrow{\$} C$) and sends it to the prover.

Response: The prover computes a response *resp* based on their witness w, the commitment t, and the challenge *chal*, and sends it to the verifier. The verifier then checks if *resp* is valid with respect to t and *chal*.

Definition 2.4 (Completeness)**.** This property ensures that an honest prover with a valid witness can always convince an honest verifier. For every valid statement-witness pair $(x,w) \in R$, the probability that the verifier accepts the proof should be overwhelming:

$$\Pr\big[\text{accept} \leftarrow \langle P(x,w), V(x) \rangle\big] \ \geq\ 1 - \mathsf{negl}_{\mathsf{completeness}}(\lambda).$$

Here, $\mathsf{negl}(\lambda)$ denotes a negligible function for the completeness property, and $\lambda \in \mathbb{N}$ is the security parameter of the protocol. In the perfect completeness case, this probability equals 1, meaning the honest verifier always accepts a valid proof from an honest prover.

Definition 2.5 (Soundness)**.** This property ensures that no malicious prover can convince an honest verifier to accept a false statement. For every statement $x \notin L_R$ (i.e., statements for which no valid witness exists) and every computationally bounded cheating prover P^*, the probability that the verifier accepts should be negligible:

$$\Pr\big[\text{accept} \leftarrow \langle P^*(x), V(x) \rangle\big] \ \leq\ \mathsf{negl}_{\mathsf{soundness}}(\lambda).$$

In the perfect soundness case, this probability is exactly 0.

Definition 2.6 (Zero-Knowledge)**.** This property ensures that all polynomial-time verifiers V^* (possibly malicious, e.g. pick challenge adaptively) learns nothing about w and beyond the validity of the statement x. More precisely, for all V^* there exists a PPT simulator S that produces a transcript without accessing w, that is statistically indistinguishable from a accepting transcript from the interaction between P and V:

$$SD(\langle P(x,w), V^*(x)\rangle, S(x)) \leq \mathsf{negl}_{\mathsf{zk}}(\lambda).$$

$SD(\langle P(x,w), V^*(x)\rangle, S(x))$ is the statistical distance between the distribution of the real interaction view (which includes all messages exchanged) and the trascript from the simulator.

Definition 2.7 (Special Honest-Verifier Zero-Knowledge (SHVZK))**.** A relaxed version of zero-knowledge. For all honest V, which follows the protocol faithfully, e.g. pick the challenge uniformly at random, there exists a PPT simulator S which, given x, can produce a simulated transcript that is indistinguishable from the real one.

SHVZK is sufficient in some cases, e.g. the ZKP protocol is transformed into non-interactive zero-knowledge (NIZK) proofs in the random oracle model [BR93] using the Fiat-Shamir heuristic [FS87].

Definition 2.8 (Knowledge Soundness)**.** If a prover P^* can convince V to accept a statement $x \in L_R$ with non-negligible probability, then P^* must 'know' the corresponding witness w. Formally, for any such successful prover P^*, there exists a PPT 'knowledge extractor' algorithm E that takes accepting transcripts $\langle P^*(x, w_i), V(x)\rangle$ as input, and output the witness w' such that $(x, w') \in R'$. The extractor is given oracle access to the transcripts and can 'rewind' the interaction between P^* and V to the point before V generates and sends the random challenge, to get the fresh randomness from V:

$$\Pr\big[(x,w') \in R' | w' \leftarrow E^{O}(x)\big] \;\geq\; \Pr\big[\text{accept} \leftarrow \langle P^*(x,w), V(x)\rangle\big] - \kappa.$$

Here, $\kappa \in negl(\lambda)$ is the soundness error, and $O := \langle P^*(x, w_i), V(x)\rangle$. This ensures that a valid proof can only be generated by a party who genuinely possesses the secret information.

Definition 2.9 (Non-interactivity)**.** Fiat-Shamir heuristic [FS87] is a method to transform an interactive proof system into a non-interactive one in the random oracle model [BR93], i.e P can generate a proof without any interaction with V, and V can check the proof independently. For example, in Schnorr's protocol, the prover can compute the challenge *chal* using a hash function (modeled as a random oracle) on the commitment t, the generator g, and the public information g^x, and sends the response *resp* directly to V without waiting for a challenge from V.

3 zkSNARKs

While the previous section laid out the properties of ZKP, some practical applications, including ZKP on blockchain, require an efficient ZKP system with small (or succinct) proof sizes. This leads us to the concept of zkSNARG proposed by Kilian [Kil92], which stands for *zero-knowledge Succinct Non-interactive ARGument*.

In addition to the definitions in section 2, A zkSNARG is a special type of argument system that has the following properties:

Definition 3.1 (Argument System)**.** An argument system $\Pi = (\mathsf{Setup}, P, V)$ consists of a PPT setup algorithm Setup, an interactive PPT prover P, and an interactive PPT verifier V. Setup generates a common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$. The interaction between P (with crs, x, w as its input) and V (with crs, x as its input) is denoted as $\langle P(\mathsf{crs}, x, w), V(\mathsf{crs}, x)\rangle$. After interaction, V outputs a bit b to decide whether to accept or reject, denoted as:

$$\text{accept or reject} \leftarrow \langle P(\mathsf{crs}, x, w), V(\mathsf{crs}, x)\rangle$$

.

Definition 3.2 (Succinctness)**.** The proof size is short and verification time is fast (w.r.t. to λ, x, w). More formally:

- The communication size is polylogarithmic in the size of w, i.e. $O(\text{poly}(\lambda, \log |w|))$.
- The verification time is polylogarithmic in the size of x, w, i.e. $O(\text{poly}(\lambda, |x|, \log |w|))$.

Definition 3.3 (Non-interactivity)**.** Typically, an argument system can be transformed into a non-interactive version using Fiat-Shamir heuristic [FS87] in the random oracle model.

Definition 3.4 (zkSNARK)**.** If a zkSNARG is also knowledge sound, it is called a zkSNARK (*zero-knowledge Succinct Non-Interactive ARgument of Knowledge*), which in this context is often referred to as an *Argument of Knowledge*.

These properties make zk-SNARKs highly suitable for applications like blockchain scaling solutions, private transactions, and verifiable computation, where efficiency and scalability are critical. One of the most well-known and widely deployed zk-SNARK constructions is Groth16 [Gro16], where the proof is composed of three group elements/points on elliptic curves, and the verification time is fast, which is a single equation computing three pairings.

References

[BR93] Mihir Bellare and Phillip Rogaway. "Random oracles are practical: a paradigm for designing efficient protocols". In: *Proceedings of the 1st ACM Conference on Computer and Communications Security*. CCS '93. Fairfax, Virginia, USA: Association for Computing Machinery, 1993, pp. 62–73. ISBN: 0897916298. DOI: 10.1145/168588.168596. URL: https://doi.org/10.1145/168588.168596.

[FS87] Amos Fiat and Adi Shamir. "How to prove yourself: practical solutions to identification and signature problems". In: *Proceedings on Advances in Cryptology—CRYPTO '86*. Santa Barbara, California, USA: Springer-Verlag, 1987, pp. 186–194. ISBN: 0387180478.

[GMR85] S Goldwasser, S Micali, and C Rackoff. "The knowledge complexity of interactive proof-systems". In: *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*. STOC '85. Providence, Rhode Island, USA: Association for Computing Machinery, 1985, pp. 291–304. ISBN: 0897911512. DOI: 10.1145/22145.22178. URL: https://doi.org/10.1145/22145.22178.

[Gro16] Jens Groth. "On the Size of Pairing-Based Non-interactive Arguments". In: *Advances in Cryptology – EUROCRYPT 2016*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 305–326. ISBN: 978-3-662-49896-5.

[Kil92] Joe Kilian. "A note on efficient zero-knowledge proofs and arguments (extended abstract)". In: *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*. STOC '92. Victoria, British Columbia, Canada: Association for Computing Machinery, 1992, pp. 723–732. ISBN: 0897915119. DOI: 10.1145/129712.129782. URL: https://doi.org/10.1145/129712.129782.