Zero Knowledge Set Membership

Eduardo Morais
Cees van Wijk
Tommy Koens

October 15, 2018

Abstract

Since 2014 ING has been working with Distributed Ledger Technology (DLT), including blockchain. One of the main topics of interest, given its importance, is the research and development of privacy mechanisms, as for example is the case of Zero Knowledge Proofs (ZKP). ZKP is a cryptographic technique that can be used to hide information that is put into the ledger, while still allowing to perform validation of this data. In 2017 [26] ING released an open source implementation [21] of a specific type of ZKP, called Zero Knowledge Range Proofs (ZKRP). Furthermore, a proof of concept was implemented in Ethereum. In this work we describe the Zero Knowledge Set Membership (ZKSM) scheme proposed by Camenisch et al [6] and discuss possible use cases. It is important to remark that ZKSM allows to construct more applications when compared to ZKRP, thus it offers more functionality and flexibility. Additionally, we present an open source implementation of this scheme which we implemented in a Go-Ethereum library.

1 Introduction

DLT and blockchain have been subject to intense research in last years, because it allows to construct consensus among parties that do not fully trust each other, without the necessity of a trusted third party. However, in public and permissionless ledgers, transactions can be viewed by everyone in the network. This fact is a hindrance that we must overcome if those transactions contain privacy-sensitive information.

In order to protect private information, a possible alternative is to use a Trusted Execution Environment (TEE), like Intel SGX [20] technology. The idea is that any private data must appear in the blockchain in encrypted form. Only the owners of the subjacent cryptographic keys will be able to decrypt it. Validation of this information must be done in the TEE system, where the cryptographic keys can be embedded. Therefore, private data will only be visible after decryption, which occurs inside a controlled environment. Putting differently, a TEE offers protection against information leakage by restricting

manipulation of private data to a region of memory that can not be accessed by other processes in the same machine, or even by its administrator. Nevertheless, attacks [13, 25] to SGX where proposed in literature, showing that this technology is vulnerable to *branch prediction* and *side-channel* attacks, respectively.

A different approach to secure private data is ZKP, which is a cryptographic technique that have been used to provide *privacy by design* in the context of DLT and blockchain. Shortly, ZKP allows an entity called *prover* to argue to another party, called *verifier*, that a determined statement is true without revealing more information than strictly necessary to convince her.

In a previous paper [26] ING described how to implement a ZKRP protocol. In summary, ZKRP allows to prove that a secret integer belongs to a certain interval. For example, if we define this interval to be all numbers between 18 and 200, a person can use the ZKRP scheme to prove that she is over 18. This gives her permission, according to some regulation, to consume a determined service, but without revealing her specific age. In the context of payment systems, if party A wants to transfer money to party B, then it is possible to utilize ZKRP to prove that the amount of money in the transaction is positive, otherwise, if the amount is negative, such transaction would in fact transfer money in the opposite direction, i.e. from B to A. A limitation of ZKRP is, however, that it can be used for numeric intervals only, and it is not possible to use a generic set.

With ZKSM we can define generic sets and still maintain privacy requirements. In this document we describe the ZKSM scheme presented by Camenisch et al [6]. ZKSM is very similar to ZKRP, the difference is that instead of the numeric interval used in ZKRP, we have a *generic set* in ZKSM. In other words, imagine that the set is formed by all countries in the European Union. Hence, if the private information is given by a country name, for instance the country of residence of a particular user, then she can use ZKSM to generate a zero knowledge proof that the private data is indeed an element from this set, therefore proving that she lives in the EU. This kind of cryptographic building block is interesting for any situation that includes sets and includes a strong privacy component. More concretely, next we describe possible use cases for ZKSM:

- **Over 18.** ZKRP is a special case of ZKSM, due to the fact that any numeric interval is also a set. Therefore if the ZKSM is more efficient than ZKRP, what turns out to be true in certain scenarios, then ZKSM can replace ZKRP to improve performance.
- **KYC.** As explained above, ZKSM allows to validate that a determined piece of private information belongs to a set of valid values. This property may be used to ensure compliance, while preserving a client's privacy. For example, an interesting use case is the so-called *anonymous credentials*, where a trusted party can attest that a user credential contains attributes whose values are correct, namely the country of residence of a person being validated by government, allowing the user to later prove that she lives in a country that belongs to the European Union, without revealing which country.

– **Board membership.** ZKSM can be used to construct *ring signatures*, which allows someone to digitally sign a message in behalf of a group of users. Afterwards, anyone can verify the signature indeed was generated by a member of the group. This is interesting for example to allow a member of a directing board to anonymously sign a contract.

– **Anti-Money Laundering (AML).** If we define the ZKSM set to be a list of entities that can consume a determined service, then we can construct a *whitelist* and an anonymous entity can prove that it is whitelisted and thus has permission to use that service. Similarly, it is possible to construct a *blacklist* formed by criminals, or by countries that are considered to be non-cooperative against money laundering, as is the case of the Financial Action Task Force [14] (FATF) blacklist. Hence an anonymous entity can prove that it does not belong to the blacklist, ensuring AML compliance.

– **Reputation validation.** Consider a set formed by companies that have good reputation, either because they are compliant to some regulation or due to the fact that they are good payers, or, in general, because they respect certain conditions. Then it is possible to use ZKSM to produce a *proof of reputation*. This use case is a little bit different compared to the previous ones, since in many practical scenarios we can not make public the set of companies that have reputation or not. In other words, this set itself is private. In this case, we must have a solution that is a little bit different from the construction presented here in the paper. Actually, there is line of research devoted to this topic, which is called *cryptographic accumulators*. Although accumulators can not be directly constructed based on ideas presented here in this document, there is indeed a close relation between ZKSM and accumulators. In fact, one of the authors of the ZKSM paper [6] described here, namely Camenisch, has many papers [8, 7, 1] in this area.

– **Common Reporting Standard (CRS).** In 2014 forty-seven countries agreed on the CRS proposal [22], whose main goal is to provide *transparency* in a global level regarding financial information, in particular to avoid tax fraud and tax evasion. The CRS allows automatic exchange of information, based on XML schemas that are responsible to dynamically describe the data format and validation patterns. Using ZKSM it is possible to carry on some of those possible validations, such as enumerations and integer ranges. Hence we have that private-sensitive data can be validated even if it is sent in its encrypted form. Therefore ZKSM may be considered an important tool that can be *reused* to provide *privacy on demand*.

In the following sections we describe in detail the algorithms necessary to implement ZKSM and instantiate the underlying parameters in order to optimize the protocol for the ZKRP use case. Interestingly, ZKSM not only offers more functionality than ZKRP, but also allows more efficient ZKRPs to be

constructed for medium-size intervals, as shown in [12]. We also provide some numbers regarding our implementation, showing that it indeed has good performance.

1.1 Organization

In Section 2 we give fundamental results that are important to understand the rest of the document. In Section 3 we describe in detail how to implement ZKSM and ZKRP. In Section 4 we give some final remarks.

2 Fundamentals

In this section we define commitment schemes and zero knowledge proofs.

Notation. By $x \leftarrow v$ we denote the operation of attributing value v to variable x. Notation $x \in_R S$ is used when variable x is set to a random element of set S. We are going to use Camenisch and Stadler [11] notation for proofs of knowledge:

$$\mathrm{PK}\{(\delta, \gamma) : y = g^{\delta} h^{\gamma} \wedge (u \leq \delta \leq v)\},$$

which denotes a proof of knowledge of integers δ and γ such that $y = g^{\delta} h^{\gamma}$ and $u \leq \delta \leq v$. In other words, this notation means that y is the commitment to the secret value δ, which is contained in the interval $[u, v)$. Greek letters are used to denote values that must be known only to the prover. For instance, we have that δ is her private data, while γ is a random value that is used to hide δ.

2.1 Zero Knowledge

Zero Knowledge Proofs (ZKP) were proposed in 1989 by Goldwasser, Micali and Rackoff [17]. Using this kind of cryptographic primitive it is possible to show that some statement is true about a secret data, without revealing any other information about the secret beyond this statement. Since then, ZKP became an important field of research, because it provides a new characterization of the complexity class NP, using the so-called *interactive programs*, and also because it is very useful to construct many cryptographic primitives. Given an element x of a language $\mathcal{L} \in NP$, an entity called *prover* is able to convince a verifier that x indeed belongs to $\mathcal{L}$, i.e. there exists a witness w for x. In particular we are interested in *proof of knowledge* (PoK), where the prover not only convinces about the existence of some witness, but also shows that the prover in fact knows a specific witness w. A desirable characteristic of such proof systems is *succinctness*, informally meaning that the proof size is small and thus can be verified efficiently. Such constructions are called zk-SNARKs [18]. However, although asymptotically good, zk-SNARKs still have some limitations and for some specific problems it turns out that different approaches achieve better performance, as we will show in this document.

Nowadays ZKP is being used to provide privacy to DLT and blockchain. For instance, it allows to design private payment systems. In summary, we would like to permit parties to transfer digital money, while hiding not only their identities but also the amount being transferred, known as *denomination*. ZKP can be used to hide this information, but still permitting validation of transactions. An important validation is showing that the denomination is positive, otherwise some payer would be able to receive money by using negative amounts. In this context we have that zk-SNARKs don't provide good performance when compared to protocols designed specifically for this purpose. The focus of this document is the description of a particular construction, by Camenisch, Chaabouni and shelat [6], which allows to build Zero Knowledge Range Proofs (ZKRP) and Zero Knowledge Set Membership (ZKSM) schemes. The former allows some party Alice, known as the *prover*, and who possesses a secret δ, to prove to another party Bob, known as the *verifier*, that δ belongs to the interval $[u, v)$, for arbitrary integers u and v. The later allows Alice to prove that her secret δ belongs to an arbitrary set S. Hence, as a first observation, we have that ZKRP is a special case of ZKSM, where we have that $S = [u, v)$.

Here we describe the concepts necessary to understand the ZKRP and ZKSM schemes proposed by Camenisch et al [6].

2.1.1 Commitment

Shortly, a cryptographic commitment scheme allows someone to compute a value that hides some message without ambiguity, in the sense that no one later will be able to argue that this value corresponds to a different message. In other words, given the impossibility to change the hidden message, we say that the user committed to that message. The purpose of using a commitment scheme is to allow a prover to compute zero knowledge proofs where the hidden message is the underlying witness w.

Definition 1. *A* ***commitment scheme*** *is defined by algorithms* Commit *and* Open *as follows:*

- $c = \text{Commit}(m, r)$. *Given a message m and randomness r, compute as output a value c that, informally, hides message m and such that it is hard to compute message m' and randomness r' that satisfies* $\text{Commit}(m', r') = \text{Commit}(m, r)$. *In particular, it is hard to invert function* Commit *to find m or r.*
- $b = \text{Open}(c, m, r)$. *Given a commitment c, a message m and randomness r, the algorithm returns true if and only if* $c = \text{Commit}(m, r)$.

A commitment scheme has 2 properties:

- **Binding.** Given a commitment c, it is hard to compute a different pair of message and randomness whose commitment is c. This property guarantees that there is no ambiguity in the commitment scheme, and thus after c is published it is hard to open it to a different value.

- **Hiding.** It is hard to compute any information about m given c.

A very well known commitment scheme is called *Pedersen commitment* [23]. Given group $\mathbb{Z}_p$, of prime order p, where the discrete logarithm problem is hard, the commitment is computed as follows:

$$c = \text{Commit}(m, r) = g^m h^r.$$

In order to open this commitment, given message m and randomness r, we simply recompute it and compare with c. An interesting property is that Pedersen commitment is *homomorphic*. Namely, we have that for arbitrary messages m_1 and m_2 and randomness r_1 and r_2, such that $c_1 = \text{Commit}(m_1, r_1)$ and $c_2 = \text{Commit}(m_2, r_2)$, respectively, then

$$c_1.c_2 = \text{Commit}(m_1 + m_2, r_1 + r_2).$$

Pedersen commitment is commonly implemented using groups over elliptic curves instead $\mathbb{Z}_p$. Also, it is important to remark that if the discrete logarithm of h with respect to g is known, then it is actually easy to generate m' and r' such that $\text{Commit}(m', r') = \text{Commit}(m, r)$, breaking the binding property. Thus in order to generate h securely, we must use a hash function that maps binary public strings to elliptic curve points [4].

Definition 2. *A **Non-Interactive Zero Knowledge (NIZK) proof** scheme is defined by algorithms* Setup*,* Prove *and* Verify *as follows:*

- Setup *algorithm is responsible for the generation of parameters. Concretely, we have that* $\text{params} = \text{Setup}(\lambda)$*, where the input is the security parameter* λ *and the output is the parameters of the ZKP system of algorithms.*
- Prove *syntax is given by* $\text{proof} = \text{Prove}(x, w)$*. The algorithm receives as input an instance* x *of some NP-language* $\mathcal{L}$*, and the witness* w*, and outputs the zero knowledge proof.*
- Verify *algorithm receives the proof as input and output a bit* b*, which is equal to* 1 *if the verifier accepts the proof.*

It is important to remark that not all ZKP schemes are non-interactive. On contrary, most ZKP protocols described in the literature are in fact interactive. In general, the prover must answer *challenge messages* sent by the verifier in order to convince him that the proof is valid, what requires multiple rounds of communication. In the context of DLT and blockchain applications, we would like to avoid this communication, because either (i) validating nodes can not properly agree on how to choose those challenges, since in many constructions we have to choose them randomly, while the verification algorithm must be deterministic in order to reach consensus; or (ii) because it would make the communication complexity of the system very poor. Nevertheless, the Fiat-Shamir heuristic [15] is a generic technique that allows to convert interactive

ZKP schemes into non-interactive protocols. The drawback of this heuristic is that it makes the cryptosystem secure under the *random oracle model* [3] (ROM). In particular, it is straightforward to make the ZKSM scheme described in this document non-interactive using the Fiat-Shamir heuristic.

A zero knowledge proof scheme has the following properties:

- **Completeness.** Given a witness w that satisfies instance x, we have that $\text{Verify}(\text{Prove}(x, w)) = 1$.
- **Soundness.** If the witness w does not satisfy x, then the probability $\text{Prob}[\text{Verify}(\text{Prove}(x, w)) = 1]$ is sufficiently low.
- **Zero Knowledge.** Given the interaction between prover and verifier, we call this interaction a *view*. In order to capture the zero knowledge property we use a polynomial-time *simulator*, which has access to the same input given to the verifier (including its randomness), but no access to the input of the prover, to generate a *simulated view*. We say that the the ZKP scheme has *perfect zero knowledge* if the simulated view, under the assumption that $x \in \mathcal{L}$, has the same distribution as the original view. We say that the ZKP scheme has *statistical zero knowledge* if those distributions are *statistically close*. We say that the ZKP scheme has *computational zero knowledge* if there is no polynomial-time distinguisher for those distributions. Intuitively, the existence of such a simulator means that whatever the verifier can compute from the interaction with the prover, it was already possible to compute before such interaction, hence the verifier learned nothing from it. Also, we say that it is a *proof of knowledge* if we can find an *extractor*, who has *rewindable* black-box access to the prover, that can compute the witness w with non-negligible probability.

2.2 Bilinear Pairings

The construction of ZKRP and ZKSM will be based on the existence of a secure bilinear map $\mathbf{bp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e, g_1, g_2)$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_t$ are groups of sufficiently large prime order, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively and e is an appropriate choice of bilinear map, satisfying the usual requirements: (i) non-degeneracy; (ii) efficiently computable and (iii) bilinearity. This cryptographic primitive is key to the constructions we will present in the next sections and it is important to remark that care must be taken when instantiating such primitive [16, 19]. Barreto-Naehrig [2] elliptic curves permit to implement bilinear maps efficiently.

3 Zero Knowledge Set Membership

In this section we describe in detail the algorithms for ZKRP and ZKSM.

3.1 ZKSM Construction

The idea of the protocol is that the verifier initially computes digital signatures for each element in the target set S. The prover then blinds this digital signature by raising it to a randomly chosen exponent $v \in \mathbb{Z}_p$, such that it is computationally infeasible to determine which element was signed. The prover uses the pairing to compute the proof, and the bilinearity of the pairing allows the verifier to check that indeed one of the elements from S were initially chosen. Algorithm 1 shows the details of the this protocol.

Algorithm 1 Set Membership

INPUT g, h, a commitment C, and a set S.
PROVER INPUT δ, γ such that $C = g^{\delta}h^{\gamma}$ and $\delta \in S$.
Verifier picks $x \in_R \mathbb{Z}_p$ and sends $y \leftarrow g^x$ and $A_i \leftarrow g^{\frac{1}{x+i}}$ for every $i \in S$.
Prover picks $\tau \in_R \mathbb{Z}_p$ and sends $V \leftarrow A_{\delta}^{\tau}$.
Prover and Verifier run $\text{PK}\{(\delta, \gamma, \tau) : C = g^{\delta}h^{\gamma} \wedge V = g^{\frac{\tau}{x+\delta}}\}$.
Prover picks $s, t, m \in_R \mathbb{Z}_p$ and sends $a \leftarrow e(V, g)^{-s}.e(g, g)^t$ and $D \leftarrow g^s h^m$.
Verifier sends a random challenge $c \in_R \mathbb{Z}_p$.
Prover sends $z_{\delta} = s - \delta c$, $z_{\tau} \leftarrow t - \tau c$, $z_{\gamma} \leftarrow m - \gamma c$.
Verifier checks that $D = C^c h^{z_{\gamma}} g^{z_{\delta}}$ and that $a = e(V, y)^c.e(V, g)^{-z_{\delta}}.e(g, g)^{z_{\tau}}$.

3.2 Range Proof

In order to obtain ZKRP, we can decompose the secret δ into base u, as follows:

$$\delta = \sum_{0 \leq j \leq \ell} \delta_j u^j.$$

Therefore, if each δ_j belongs to the interval $[0, u)$, then we have that $\delta \in [0, u^{\ell})$. Algorithm 1 can be easily adapted to carry out this computation, as shown in Algorithm 2.

In order to obtain Zero Knowledge Range Proofs for arbitrary ranges $[a, b)$ we show that $\delta \in [a, a + u^{\ell})$ and $\delta \in [b - u^{\ell}, b)$, using 2 times the ZKRP scheme described in Algorithm 2. Namely, we have to prove that $\delta - b + u^{\ell} \in [0, u^{\ell})$ and $\delta - a \in [0, u^{\ell})$.

3.3 Implementation

We implemented Algorithms 1 and 2 in Golang and based on libsecp256k1 library, available in Go-Ethereum. We used BN128 pairing-friendly elliptic curves, thus accomplishing 128 bits of security. The performance is summarized in Table 1, and the measurement was carried out in a computer with a 64-bit Intel i5-6300U 2.40GHz CPU, 16 GB of RAM and Ubuntu 18.04. The implementation is available on Github [21] and is a proof of concept, thus it should not be used in production without first spending the effort to review it where necessary.

Algorithm 2 Range Proof for interval $[0, u^\ell)$

INPUT g, h, u, ℓ and a commitment C.
PROVER INPUT δ, γ such that $C = g^\delta h^\gamma$ and $\delta \in [0, u^\ell)$.

Verifier picks $x \in_R \mathbb{Z}_p$ and sends $y \leftarrow g^x$ and $A_i \leftarrow g^{\frac{1}{x+i}}$ for every $i \in \mathbb{Z}_u$.
Prover picks $\tau_j \in_R \mathbb{Z}_p$ and sends $V_j \leftarrow A_{\delta_j}^{\tau_j}$ for every $j \in \mathbb{Z}_\ell$, such that $\delta = \sum_j \delta_j u^j$.

Prover and Verifier run $\text{PK}\{(\delta_j, \gamma, \tau_j) : C = h^\gamma \prod_j (g^{u^j})^{\delta_j} \wedge V_j = g^{\frac{\tau_j}{x+\delta_j}}\}$.
Prover picks $s_j, t_j, m_j \in_r \mathbb{Z}_p$ for every $j \in \mathbb{Z}_\ell$ and sends $a_j \leftarrow e(V_j, g)^{-s_j}.e(g, g)^{t_j}$ and $D \leftarrow \prod_j (g^{u^j s_j} h^{m_j})$.
Verifier sends a random challenge $c \in_R \mathbb{Z}_p$.
Prover sends $z_{\delta_j} \leftarrow s_j - \delta_j c$, $z_{\tau_j} \leftarrow t_j - \tau_j c$ for every $j \in \mathbb{Z}_\ell$, and $z_\gamma = m - \gamma c$.
Verifier checks that $D = C^c h^{z_\gamma} \prod_j (u^j z_{\delta_j})$ and that $a_j = e(V_j, y)^c.e(V_j, g)^{-z_{\delta_j}}.e(g, g)^{z_{\tau_j}}$ for every $j \in \mathbb{Z}_\ell$.

Optimal values for u and ℓ can be calculated as described in the original paper [6]. We used $u = 57$ and $\ell = 5$ for the interval $[347184000, 599644800)$, obtaining communication complexity equal to 30976 bits, while the previous work, based on Boudot's proposal [5], has 48946 bits.

	Setup (ms)	Prove (ms)	Verify (ms)
ZKSM	31.78	70.18	98.95
ZKRP	331.41	579.32	851.89

Table 1: Time complexity

Scheme	Communication
This work	30976 bits
Previous work [26]	48946 bits

Table 2: Communication complexity

Compared to other proposals in the literature, we found that for very big intervals, Boudot's [5] scheme is better, since verification doesn't depend on the size of the secret. However, the complexity of Prove algorithm in Boudot's proposal is $\mathcal{O}(n^4)$, and if the set S is fixed *a priori*, such that digital signatures don't need to be recomputed for each execution of the protocol, then the construction described here is almost 10 times faster. On the other hand, for small secrets, Schoenmakers's strategy [24] is the most efficient scheme. A more detailed comparison can be found on the paper [12] by Canard, Coisel, Jambert and Traor.

4 Related work and final remarks

In this document we described in detail the construction of ZKRP and ZKSM protocols, which where implemented over Go-Ethereum library. Another way to obtain Zero Knowledge Set Membership protocols is by using *cryptographic accumulators* [9]. Also, the underlying digital signature scheme used, namely Boneh-Boyen signatures, can be replaced and the construction presented here can be adapted to use the digital signature proposed by Camenisch and Lysyanskaya [10]. Nevertheless, both modifications would make it necessary to assume hardness of the strong RSA assumption.

In the context of DLT applications, it is possible to use Zero Knowledge Set Membership to validate user information without revealing it. A possible scenario is to perform KYC operations. For example, it would be possible to validate that the country of residence of a user is one belonging to the European Union, without revealing which country. In the case of Zero Knowledge Range Proofs, a commonly mentioned application is validating that someone is over 18 and thus is allowed to use a certain service, without revealing the age. In Section 1 we discussed several other applications, like reputation systems and AML or CRS compliance.

Recent breakthroughs in cryptography permit us to construct new protocols and achieve *privacy on demand*. These new cryptographic algorithms can be ultimately considered as tools that can be reused in different problems. Therefore ING is following the steps to build the knowledge that is necessary in order to construct a *toolbox* to deal with the above-mentioned complex problems.

As a future work, we will integrate this implementation to Ethereum, such that ZKSM can be used in a smart contract. In order to do that, it would be interesting to rewrite the Verify algorithm in Solidity, avoiding the necessity of using our modified Go-Ethereum client. Also, we will research other ZKP protocols that may be used to enhance privacy on DLT and blockchain.

References

[1] Foteini Baldimtsi, Jan Camenisch, Maria Dubovitskaya, Anna Lysyanskaya, Leonid Reyzin, Kai Samelin, and Sophia Yakoubov. Accumulators with applications to anonymity-preserving revocation. In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*, pages 301–315, 2017.

[2] Paulo Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography*, pages 319–331, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[3] 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, pages 62–73, New York, NY, USA, 1993. ACM.

[4] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, ASIACRYPT '01, pages 514–532, Berlin, Heidelberg, 2001. Springer-Verlag.

[5] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 431–444, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[6] Jan Camenisch, Rafik Chaabouni, and abhi shelat. Efficient protocols for set membership and range proofs. In Josef Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008*, pages 234–252, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[7] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Public Key Cryptography - PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings*, pages 481–500, 2009.

[8] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, pages 61–76, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[9] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, pages 61–76, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[10] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Giuseppe Persiano, and Clemente Galdi, editors, *Security in Communication Networks*, pages 268–289, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[11] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In Burton S. Kaliski, editor, *Advances in Cryptology — CRYPTO '97*, pages 410–424, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[12] Sébastien Canard, Iwen Coisel, Amandine Jambert, and Jacques Traoré. New results for the practical use of range proofs. In Sokratis Katsikas and Isaac Agudo, editors, *Public Key Infrastructures, Services and Applications*, pages 47–64, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[13] Guoxing Chen, Sanchuan Chen, Yuan Xiao, Yinqian Zhang, Zhiqiang Lin, and Ten H. Lai. Sgxpectre attacks: Stealing intel secrets from sgx enclaves via speculative execution. `https://arxiv.org/abs/1802.09085` (visited on 19/09/2018).

[14] FAFT. Financial action task force - countries. `http://www.fatf-gafi.org/countries/`.

[15] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

[16] Steven Galbraith, Kenny Paterson, and Nigel Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113 – 3121, 2008. Applications of Algebra to Cryptography.

[17] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM.

[18] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, pages 321–340, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[19] Mehmet Sabr Kiraz and Osmanbey Uzunkol. Still wrong use of pairings in cryptography. Cryptology ePrint Archive, Report 2016/223, 2016. `http://eprint.iacr.org/`.

[20] Tommy Koens. Consensus by trusted hardware, 2018. `https://www.linkedin.com/pulse/consensus-trusted-hardware-tommy-koens`.

[21] Eduardo Morais, Peter Rudgers, Cees van Wijk, Tommy Koens, and Coen Ramaekers. Zero knowledge range proof implementation. Github, 2018. `https://github.com/ing-bank/zkrangeproof`.

[22] OECD. Declaration on automatic exchange of information in tax matters. `http://www.oecd.org/mcm/MCM-2014-Declaration-Tax.pdf`.

[23] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 129–140, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[24] Berry Schoenmakers. Interval proofs revisited. Slides presented at the International Workshop on Frontiers in Electronic Elections, 2005.

[25] Michael Schwarz, Samuel Weiser, Daniel Gruss, Clémentine Maurice, and Stefan Mangard. Malware guard extension: Using sgx to conceal cache attacks. In Michalis Polychronakis and Michael Meier, editors, *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 3–24, Cham, 2017. Springer International Publishing.

[26] Coen Ramaekers Tommy Koens and Cees van Wijk. Efficient zero-knowledge range proofs in ethereum. ING media. `https://www.ingwb.com/media/2122048/zero-knowledge-range-proof-whitepaper.pdf`.