Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 28 (2025), No. 3, pp. 849–867
DOI : 10.47974/JDMSC-2136

A stateless blockchain-based public key infrastructure

Amalan Joseph Antony *
Department of Computer Science and Engineering
Indian Institute of Information Technology, Design and Manufacturing
Kancheepuram
Chennai 600127
Tamil Nadu
India

Kunwar Singh †
Department of Computer Science and Engineering
National Institute of Technology
Tiruchirapalli 620015
Tamil Nadu
India

Abstract

Various designs exist for a Public Key Infrastructure (PKI), each with its own inherent security and optimization issues. Certain properties of the Bitcoin blockchain and its derivatives - mainly the safeguards against the manipulation of data once it is stored - solved many of the PKI issues, and various designs for a blockchain-based PKI were proposed. Such blockchain-based PKIs required enormous amounts of storage space as the number of blocks and the number of users increased and thus became less efficient in performing operations such as inclusion, key retrieval, key verification, etc. In this article, we propose a new approach to the blockchain-based PKI, which while being as tamper-proof as the current blockchain-based PKIs, is very efficient in its space requirements and the time taken for key verification.

Subject Classification: *11T71, 14G50, 94A60.*

Keywords: *Cryptography, PKI, Blockchain, Accumulator, Stateless.*

* *ORCID-ID:* **0000-0002-8570-9822**
† *ORCID-ID:* **0000-0001-5889-0964**

* *E-mail:* **amalan@iiitdm.ac.in (Corresponding Author)**
† *E-mail:* **kunwar@nitt.edu**

© Taru Publications

1. Introduction

A Public Key Infrastructure (PKI) is a collection of various hardware and software entitites that together form a system for the secure and reliable distribution of public keys. Originally, PKIs were modeled as Certificate Authorities (CAs) [1]. The CA is a trusted third party that issues a certificate to an identity asserting that this identity owns a particular key, and this certificate is recognised and accepted by every identity that trusts the CA. Such CAs are single points of failure, since corrupting a CA causes the entire network to fail. Further, the process of certificate issuance is not entirely transparent. Multiple and independent CAs undermine identity retention, thereby facilitating an adversary to impersonate a legitimate and registered identity [2]. Revoking an issued certificate through Certificate Revocation Lists (CRLs) consumes significant time. The breach of the CA DigiNotar, and the ensuing fake certificates for Google, among the other noteworthy high-profile domains, has cast doubts on the reliability of CAs [3].

A decentralised trust was proposed in Phil Zimmerman's PKI modelled as a Web of Trust (WoT) [4]. An Identity in the network would trust another identity if it is trusted by another trusted identity. Even in such an approach, a compromised identity can corrupt all the chains of trust that it is a part of. Further, the entry barrier to join the network is very high. And, as in the case of CAs, identity retention is not ensured. In fact, identities can be aliased within the same network, and still be trusted by different trusted members of the same network.

1.1 *Related work*

With the advent of the Bitcoin [5], the possibility of building an alternative to PKI based on the blockchain technology was proposed. A neat PKI model that ensures identity retention was proposed by Fromknecht et al. [2], using dictionary data structures for key lookup. Eventually, many proposals were made, improving on the original design, to gain on one aspect or the other.

Yakubov et al. explored the possibility of using a blockchain-based PKI for issuing, validating and revoking X.509 certificates [3].

Axon et al. considered the suitability of a blockchain-based PKI for contexts in which the linking of an identity with its public key is undesirable [4].

Amalan et al. gave a simplified model of a blockchain-based PKI [6], and went further in enhancing the privacy of the identities and their key

updates, by proposing a model that does not require the public linking of an identity to its key, or that of the current key to the previous key [7].

[8] proposed extending the X509 certificateâ€™s structure, and using Bloom filters to keep track of revoked certificates, but did not achieve any optimization in the space and time requirements for registering and updating keys.

[9] proposed BPKI - a secure and scalable blockchain-based PKI solution that is secure against a domain name preemption attack, but failed to automate the interaction processes among the users.

[10] designed a lighter smart contract using a threshold value, but still came short of achieving a significant reduction in the storage space as the number of users increase.

In all the blockchain-based PKI models that have been proposed so far [3, 4, 6-10], there has been no substantial improvement in reducing the storage space required for the blockchain, which grows ever increasingly with the passage of time and when more users are added to the network. Similarly, verifying whether a given key belongs to a given identity involved a linear search of the identities, taking $O(n)$ time, where n is the number of users. Thus, the main advantage provided by a blockchain-based PKI - the computational infeasibility of manipulating a key once posted in the blockchain - came at a great cost.

Our contribution:

In this paper, we propose a new approach to the blockchain-based PKI. Whereas conventional blockchain-based PKIs store each identity-key pair as a transaction and thus require enormous amounts of storage, our method requires the storage of only one transaction for a larger number of identities. Our approach, while being as tamper-proof as the current blockchain-based PKIs, is an order of magnitude superior in both the space required for storage and the time required for key verification, in as much as both are drastically reduced from $O(n)$ to $O(1)$. We draw our inspiration from the work of Boneh et al. [11] which designed RSA-based accumulators for a stateless UTXO transaction validation. But while Boneh et al. [11] retain the blockchain as such and propose an optimization mechanism for some of the miner nodes, we go further and propose a novel method to optimise the blockchain itself. In order to achieve this, we do not perform all the operations on the blockchain itself - some are done offline.

Paper outline:

The rest of this paper is organised in this manner: in Section 2 we give some relevant background information on the prevailing structure of a blockchain-based PKI following [2], accumulators, RSA groups and smart contracts. In Section 3, we present our stateless blockchain-based PKI with optimized space requirements and verification operations. We then analyse our proposal in Section 4.

2. Preliminaries

2.1 *Blockchain*

In 2009, Satoshi Nakamoto introduced the concept of a blockchain as a distributed ledger which grows in size in discrete steps called *blocks*, which are sequentially linked to each other through hashes satisfying certain properties. As this did away with the *third party* required in conventional digitial currency transactions, currency systems built on blockchains became viable alternatives to digitial currencies.

For a more detailed introduction to blockchain and cryptocurrencies, the reader is referred to [12] and [13].

2.2 *Blockchain-based PKI*

A blockchain-based PKI makes use of three key algorithms:

- The $keygen(1^k)$ algorithm that generates the secret and public key pair (sk, pk).
- The $sig(sk, \mu)$ algorithm that generates the digital signature σ on the message μ with the secret key sk.
- The $ver(pk, \sigma, \mu)$ algorithm which verifies if σ is the genuine signature on the message μ by the secret key sk corresponding to the public key pk.

The above algorithms are instrumental to the functionalities of the blockchain-based PKI: registering a new identity with its public key, updating the public key of a registered identity, retrieving the public key of an identity, verifying an identity's public key, and revoking an identity's public key.

An identity registers itself by posting $(id, register, online, (pk_n, \sigma_n))$ and $(id, register, offline, (pk_f, \sigma_f))$ to the blockchain, where id is the identity, pk_n and pk_f are the online and offline public keys, and σ_n and

σ_f are cryptographically secure EUF-CMA signatures ([14, 15]) of the associated id string produced with the secret keys sk_n and sk_f corresponding to pk_n and pk_f respectively.

If, for each registration request, both $ver(pk_n, \sigma_n, id)$ and $ver(pk_f, \sigma_f, id)$ evaluate to true, then the request qualifies to be included in the blockchain.

To update a key, the identity posts $(id, update, type, (pk_{old}, pk_{new}, \sigma_1, \sigma_2))$, where *type* could be *online* or *offline* depending on which key is to be updated, pk_{old} is the old public key that is to be replaced, pk_{new} is the new public key, $\sigma_1 = sig(sk_{old}, (id, pk_{new}))$ and $\sigma_2 = sig(sk_{new}, id)$ are signatures which prove the knowledge of the old secret key sk_{old} corresponding to the old public key pk_{old} and therefore the ownership thereof, that pk_{new} is to be the new public key, and the knowledge of the new secret key sk_{new} corresponding to the new public key pk_{new}.

Thus, for each new identity to be included in the blockchain, a dedicated space is required to store the identity and key details, making the space required by each miner for the storage of the blockchain $O(n)$. Also, verification of an identity's key by look up involves a linear traversal through all the identities currently in the blockchain, making the time complexity for verification $O(n)$.

For a complete description of the blockchain-based PKI, the reader is referred to [2].

2.3 *Accumulator*

An accumulator is a primitive that serves as a compact representation of a set of elements, which serves as a binding commitment to the set. When a new element is added to the set, a witness is generated which testifies the inclusion of the element. The accumulator can be queried to determine whether an element is included in the set or not. For a detailed explanation on the working of an accumulator, the reader is referred to [16].

The Merkle tree is the simplest accumulator, which grows logarithmically in the number of elements committed. Dynamic accumulators, according to [17] are those in which the commitments can be updated efficiently at unit cost, as elements are added or removed from the set, independent of the number of the elements in the set.

A dynamic accumulator uses the algorithms *initialize* to generate a group $\mathbb{G}$ of unknown order and to initialize the generator g of that group which will be used for further operations, *add* that takes A_t, the

accumulator at stage t and computes A_{t+1}, the accumulator at stage $t+1$ that includes new element(s), *delete* that removes the element from the current state of the accumulator, *mem_wit_create* and *non_mem_wit_create* that create witnesses for membership and non-membership respectively, and *ver_mem* and *ver_non_mem* that verify membership and non-membership respectively.

2.4 *RSA hidden group*

The generator g for the Accumulator described above is selected from an RSA hidden group [18], i.e., an RSA group of unknown order which is defined as follows:

Choose two prime numbers p and q such that $p = 2p' - 1$ and $q = 2q' - 1$, where p' and q' are also prime, and compute $n = pq$. Now define the group $\mathbb{G} := \{x \mid x \equiv y^2 (\mathrm{mod}\, n)\,,\ y \in \mathbb{Z}_n^*\}$. Thus, $\mathbb{G}$ is the set of all quadratic residues modulo n, and is a cyclic group with generator g [18]. The order of this group $O(\mathbb{G}) = \frac{(p-1)(q-1)}{4}$ is unknown, since the factors p and q of n are unknown.

2.5 *Security model*

A generic definition of the security of an accumulator, and some of the properties required of an accumulator are given by [19] and [20]. In our work, the security model is adapted from [11], which captures the idea that an adversary should not be able to construct a membership witness for a non-member element, nor a non-membership witness for a member element. This is called the *undeniability property* of an accumulator.

An accumulator A is secure if an adversary cannot construct the tuple (x, w, w') where x is a valid element, w is a witness for membership of x (i.e., is proof that x in included in the accumulator), and w' is a witness for non-membership of x (i.e., is proof that x in not included in the accumulator), or alternately, the probability that an adversary is able to construct witness w and w' for both the membership and the non-membership of an element x is negligbile, i.e.,

$$Pr(ver\_mem(A, x, w) \wedge (ver\_non\_mem(A, x, w')) = negl \tag{1}$$

2.6 *Smart contracts*

Smart contracts are programs on a blockchain that automatically execute certain actions when certain predetermined conditions are met

and verified. Once the actions are completed, a transaction is produced which is added to the blockchain, rendering the transaction unchangeable. Since smart contracts do away with third parties and intermediaries, time delays are minimized, and the transactions are efficient and transparent. In our work, smart contracts will be used to generate commitments by calculating the accumulator values.

2.7 *UTXO*

A UTXO - unspent transaction output - is a feature common to most blockchain ledgers. We consider only those blockchains where the unspent transactions form a set of UTXOs, and where a miner is the entity who produces a new block to be appended to the ledger. Each block consumes a certain number of unspent transaction and spends the corresponding transactions by invalidating them. However, our PKI modelled on a blockchain does not require these UTXOs to have monetary value. In bitcoin, the public keys are associated with monetary attributions such as 1 BTC belongs to public key of Alice. In this paper we concern ourselves with the process of updating the public keys associated to identities which are more explicitly published inside the blockchain and can be arbitrary strings (names or identifiers). The older associations entries are invalidated as and when they are replaced by new ones which are properly authorised by the legitimate owner of the previous entry (using the owner's private key and digital signature).

A stateless blockchain according to [21] enables the constituent nodes to take part in validating transactions without having to store the entire UTXO set. An accumulator commitment to all the UTXOs in the set suffices, and does away with need to store voluminous data which do not serve any other purpose.

3. Proposal: A space optimized stateless blockchain-based PKI

3.1 *Problems with the existing solutions*

While a blockchain offers an excellent safeguard against data manipulation, there are two significant problems with a blockchain-based PKI as the number of the users increase. As each new identity's registration adds a new entry (with such details as the id, the key, signatures to prove authenticity, etc.) to the block, the size of a block grows linearly with the number of the identities. Also, since the verification of an identity's key involves a linear traversal through the contents of a block to locate the

identity and its key, the time taken for key-verification too grows linearly with the number of the identities.

This makes devices with comparitively lesser storage capacity or computational power incapable of the required operations.

3.2 *A shift in the paradigm*

The main reason for choosing a blockchain as the basis for a PKI is the protection offered by the blockchain against manipulation of the data stored in it. To ensure this integrity without, however, having to store the data in the blockchain itself, we propose exchanging some data offline, and storing just a commitment to the data in the blockchain. Thus, our new PKI system is interleaved with both online and offline transactions:

3.2.1 Key generation, offline

An identity id_i first generates its key pair (pk_i, sk_i). A method of generating secure key pairs using the Elliptic Curve Cryptographic (ECC) principles is presented in [7].

3.2.2 Storing a binding commitment, online

When the identity id_i wishes to share its public key pk_i to another identity id_j, its ownership of the public key pk_i is first verified through its signature, and then a commitment is generated to bind pk_i to id_i as follows:

The commitment scheme

At the onset, we use a dynamic accumulator to generate a group $\mathbb{G}$ of unknown order and initialize the generator g of that group. Two state variables - P_t (initialized to 1) to store the product of the existing odd primes at time t, and A_t (initialized to g) to store the accumulator value at time t - are included in the smart contract. When an identity id_i wishes to register itself on the blockchain, it computes the signature σ_i on $[id_i \| pk_i]$ where $\|$ denotes a concatenation operator, using the identity's secret key sk_i corresponding to its public key pk_i. The identity then sends $\theta_i = (id_i, pk_i, \sigma_i)$ to a smart contract which pools all the incoming tuples in an array $tuple\_array$, and once the block time is over, verifies every signature σ_i using the corresponding public key pk_i and checking if it evaluates to $[id_i \| pk_i]$. If the verification test passes, then the add function maps $[id_i \| pk_i]$ to an odd prime op_i, and calculates the new product of all

the odd primes P_{t+1}, and the next state of the accumulator A_{t+1} as presented in Algorithm 1.

Algorithm 1 *Next_state*(*op*)

1. $len \leftarrow length(op)$
2. $new\_primes \leftarrow 1$
3. for $i \leftarrow 1$ to len do
4. $\quad new\_primes \leftarrow new\_primes * op[i]$
5. end for
6. $product\_of\_primes \leftarrow product\_of\_primes * new\_primes$
7. $accumulator \leftarrow accumulator^{new\_primes}$

At the end of each block time, the new values of the product of the odd primes and the accumulator is sent to the mining network to be included in the blockchain. The overall scheme for including new members is presented in Algorithm .

Algorithm 2 $add(A_t, \theta_1, \theta_2, ..., \theta_n)$

1. $op_i \leftarrow odd\_prime([id_i \parallel pk_i])$
2. $P_{t+1} \leftarrow P_t * \prod_1^n op_i$
3. $A_{t+1} \leftarrow A_t^{\prod_1^n op_i}$

Thus, at any time t, the accumulator commitment value for n identities is $A_t = g^{op_1 . op_2 ... op_n}$ where $op_i = odd\_prime([id_i \parallel pk_i])$.

It may be noted here that in the conventional blockchain-based PKI, each identity-key pair needs to be saved as a separate transaction, and hence several transactions needed to be added to the blockchain. But in the above approach, only one transaction - the output of the smart contract at the end of each block time - needs to be added to the blockchain, which leads to an important reduction in storage space.

3.2.3 Key sharing, offline

Thus, an identity id_i wishing to share its key to a communicating identity, say id_j, at time t, generates the witness $w_i = g^{P_t / op_i}$ and sends the tuple $\phi_i = (id_i, pk_i, \sigma_i, w_i)$ offline to id_j.

3.2.4 Witness verification, online

On receiving the tuple $\phi_i = (id_i, pk_i, \sigma_i, w_i)$, the communicating identity id_j verifies the witness against the accumulator commitment A_t in the blockchain for the inclusion of id_i using the *ver_mem* function.

Algorithm 3 $ver\_mem(A_t, w_i, op_i)$

1. if $w_i^{op_i} = A_t$ then
2. return 1
3. else
4. return 0
5. end if

If the verification passes, it proves to the communicating identity id_j that the identity-key pair had been committed in the accumulator as such.

To provide for the removal of an identity-key pair, for instance, because of a revocation following a key compromise, another accumulator *D* is maintained, containing the commitments of the deleted identity-key pairs. An identity that is to be removed is simply added to accumulator *D*. During the course of public key validation, the accumulator *D* is first checked to see if the identity in question is in the list of deleted identities. If not, then the accumulator *A* is checked to see if the identity is included there.

3.2.5 Signature verification, offline

Finally, id_j ascertains the genuineness of the key by verifying the signature σ_i with the public key pk_i.

3.3 *Proof of correctness*

Let $S = \{id_1, id_2, ...id_n\}$ be the set of identities registered on the blockchain, and let $\{op_1, op_2, ...op_n\}$ be their corresponding odd primes. Then, from Algorithm 2, we have:

$$P_t = \prod_1^n op_i \tag{2}$$

$$A_t = g^{op_1 . op_2 ... op_n} \tag{3}$$

An identity id_i computes its witness of inclusion w_i as:

$$w_i = g^{P_i/op_i} \tag{4}$$

$$= g^{\frac{op_1.op_2...op_n}{op_i}} \tag{5}$$

$$= g^{\prod_{j \neq i} op_j} \tag{6}$$

For verifying this witness, the receiving identity computes:

$$w_i^{op_i} = (g^{\prod_{j \neq i} op_j})^{op_i} \tag{7}$$

$$= (g^{\frac{op_1.op_2...op_n}{op_i}})^{op_i} \tag{8}$$

$$= g^{op_1.op_2...op_n} \tag{9}$$

$$\therefore w_i^{op_i} = A_t \tag{10}$$

3.4 *Security proof*

The security of our proposal is based on the strong RSA assumption, that it is hard to compute an arbitrary root of a random group element in the generic group model[11, 22, 23].

Theorem 1: *Let the chosen group* $\mathbb{G}$ *be a group in which the strong RSA assumption holds. Then our accumulator satisfies the undeniability property, and is therefore secure.*

Proof: We shall prove this by the principle of contradiction - we shall show that, if there exists an adversary who can break the undeniability property of an accumulator, then this adversary can be used to construct another adversary who can break the strong RSA assumption.

Let $\mathcal{A}_{Acc}$ be an adversary who can construct the witness w that the identity *id* is included in the set S (say) of members corresponding to the accumulator A, and also the witness w' that the identity *id* is not included in the set S corresponding the accumulator A.

Since w is a witness of inclusion of *id*, we have $w^{op} = A$, where *op* is the odd prime corresponding to the identity *id*.

Since w' is a witness that *id* is not included in S, we have $gcd(s^*, op) = 1$ where $s^* = \prod_{s \in S} s$. Now, by extended Euclidean algorithm, $\mathcal{A}_{Acc}$ can find integers a and b such that $a(s^*) + b(op) = gcd(s^*, op) = 1$, and compute $g^b = B$, say. Note that:

$$(Bw^a)^{op} = B^{op}(w^a)^{op} \tag{11}$$
$$= (g^b)^{op}(w^{op})^a \tag{12}$$
$$= (g^b)^{op}((g^{s^*/op})^{op})^a \tag{13}$$
$$= (g^b)^{op}(g^{s^*})^a \tag{14}$$
$$= g^{b(op)+a(s^*)} \tag{15}$$
$$= g^{a(s^*)+b(op)} \tag{16}$$
$$= g^1 \tag{17}$$
$$\therefore \quad (Bw^a)^{op} = g \tag{18}$$
$$\Rightarrow \quad Bw^a = g^{\frac{1}{op}} \tag{19}$$

Thus, from the values of w, a and B, it is possible to construct an adversary $\mathcal{A}_{RSA}$ who can compute the op-th root of g as Bw^a, which contradicts the strong RSA assumption. Hence, if the strong RSA assumption holds in the chosen group $\mathbb{G}$, our accumulator is secure.

4. Analysis of The Proposal

4.1 *Space and time optimizations*

The space optimization is evaluated as follows:

Let n be the number of new identities to be added to the blockchain, and let the maximum permitted size of a block in the blockchain be B, and let the size of a transaction be s. This would allow $\frac{B}{s}$ transactions to be stored in a block. Thus, in the conventional approach to a blockchain-based PKI, n identity-keypairs would be represented in n transactions occupying a space $O(ns)$, requiring a total of $\lceil \frac{ns}{B} \rceil$ blocks.

In our approach, let the maximum number of requests that the smart contract can handle in a block time be S. Each block time would produce a new and updated accumulator state which needs to be included in the blockchain as a transaction. Thus, n identity-keypairs would require $\lceil \frac{n}{S} \rceil$ executions of the smartcontracts, producing $\lceil \frac{n}{S} \rceil$ transactions occupying a space $O(\lceil \frac{n}{S} \rceil s)$. Since $S >> n$, the space required is practically a constant equal to the size of a transaction s.

Saving just the current state of the accumulator in the blockchain drastically reduces the required storage space from $O(n)$ to nearly a constant, in terms of the size of the accumulator. Such a design of the commitment scheme saves a lot of space on the blockchain.

Futher, unlike the linear search that would precede retrieving an identity and its key in the conventional stateful blockchain-based PKI, the only operation performed here is a verification of whether the odd prime corresponding to the signature is included in the accumulator. Thus, the time required for verification of an identity-key pair in the blockchain is reduced to a constant.

Manipulation of the accumulator states that are stored in the blockchain is as computationally infeasible as manipulating the blockchain itself. Thus, the integrity assurance of storing just the accumulator states in the blockchain is equivalent to the integrity assured by storing the identity-key pairs of the entire set of identities in the blockchain. Therefore, we attain the integrity of a stateful blockchain-based PKI, but with the size of each block being reduced to the size of a transaction, which, by an apt choice of algorithms, has been limited to a constant size. Similarly, we have also reduced the time required for key verification.

4.2 *Mapping to odd primes*

The main overhead of this stateless blockchain is the mapping of $[id_i \| pk_i]$ to an odd prime by the subroutine $odd\_prime([id_i \| pk_i])$. But this mapping is necessary, since otherwise, an adversary can convince the verifier that a signature is included in the accumulator when it is actually not. For illustration, let the signatures of three identities evaluate to $23, 56$ and 79. Therefore, the accumulator $A_t = g^{23*56*79} = g^{101752}$. Now an adversary, to prove the inclusion of 28, gives as proof the signature $\sigma = 28,$ and the witness $w = g^{\frac{101752}{28}} = g^{3634}$. The verifier then checks $(g^{3634})^{28} = g^{101752} = A_t$ and gets convinced that 28 is included in the accumulator.

To prevent an adversary from falsely proving an inclusion in this manner, no element of the accumulator should be a factor of any other element. Thus, the accumulated elements should be primes, necessitating a mapping of the signatures to distinct primes. An easy way to map a given $[id_i \| pk_i]$ to a prime number less than 2^x is to find the least integer i such that the first x bits of $H(id_i \| pk_i)$ is a prime number [11]. Hashing to prime numbers has been explored independently, and efficient methods of generating random primes exist in literature, of which [24] and [25] are notable examples.

An improvement is to let the identities themselves provide a nonce such that $([id_i \| pk_i] \| nonce)$ is accumulated instead of just $[id_i \| pk_i]$. However, this enables an adversary to accumulate the same $[id_i \| pk_i]$ twice. But in stateless blockchains it is guaranteed that no $[id_i \| pk_i]$ will be

accumulated twice, where committing to the current state of the accumulator serves as a means to ensure uniqueness [11].

4.3 *Complexity*

Here we analyse the computational complexity for n users in each step of our proposal:

4.3.1 Key generation

Key generation is a part of the conventional blockchain based PKI models too, and it happens offline. The key generation method that we have adapted from [7, 26] is of $O(n)$ and is therefore viable.

4.3.2 Storing a binding commitment

This step comprises of two parts:

- Mapping of $[id_i \| pk_i]$ to an odd prime by the subroutine $odd\_prime([id_i \| pk_i])$, and
- Updating the accumulator after having computed all the odd primes for the identities in the new block.

In the first part, the subroutine $odd\_prime([id_i \| pk_i])$ runs in constant time [11], and so for n users, mapping to odd primes is done in $O(n)$ time.

In the second part, updating the Accumulator involves modular exponentiation, which can be optimised with multi-exponentiation techniques [27] to $O(n / log\ n)$ multiplications in $\mathbb{G}$.

4.3.3 Key sharing

Here, an identity needs to generate its witness and include this in the message which it will send to the receiver. This witness generation involves modular exponentiation, which can be optimised with multi-exponentiation techniques [27] to $O(n / log\ n)$ multiplications in $\mathbb{G}$.

We note here that in the conventional PKI model [4], an identity id_i sends the tuple (id_i, pk_i, σ_i) to id_j, whereas in our proposal, id_i sends the tuple $(id_i, pk_i, \sigma_i, w_i)$ with the additional w_i which would enable the receiver id_j to verify the inclusion of id_i in the next step. This w_i which is not a part of the the message sent by id_i in the conventional PKI model is the only additional offline communication necessitated by our proposal. However, as w_i is just an element of $\mathbb{G}$ and is therefore a constant number

of bits long, it does not constitute a significant overhead to the message sent by id_i.

4.3.4 Witness verification, online

In this step, the receiving identity id_j verifies the witness against the accumulator commitment A_t in the blockchain for the inclusion of id_i using the *ver_mem* function. This witness verification involves modular exponentiation, which can be optimised with multi-exponentiation techniques [27] to $O(n / log\ n)$ multiplications in $\mathbb{G}$.

4.4 *Experimental evaluation*

We successfully implemented our new stateless blockchain based PKI, on the Ethereum blockchain. The setup we used for our experiments is as follows:

- IDE: Remix 0.23.3
- Compiler: soljson-v0.8.7+commit.e28d00a7.js
- Langauge: Solidity
- EVM: JavaScript VM
- Initial account balance: 100 Ether
- Gas limit: 3000000
- Gas Price: 20 Gwei

We studied our proposal's performance in each of the five steps, for various ranges of the number of identities in the network, and list the noteworthy observations here:

1. Even for thousands of identities, the two stages of storing a binding commitment together take only a few seconds for the computations, which is well within the standard block time of 10 minutes [5], thereby showing that the computational overhead is negligible.
2. As to the smart contract and the inclusion of the block in the blockchain, we inspected the gas costs recorded in the logs corresponding to the status "Transaction mined and execution succeed", and averaged over many deployments to a gas cost of 265748, from which we calculate the transaction fee at a gas price of 20 Gwei as $20 * 265748 * 10^{-9} = 0.005314960$ ETH.

3. The time taken for witness generation is negligibly small (($\leq 10^{-4}$ seconds).
4. Witness verification for several identities takes only a few seconds, well within the standard block time of 10 minutes [5], thereby showing that the computational overhead is negligible.

The above observations corroborate the feasibility of our proposed model.

5. Conclusion and future work

We expect this new approach of building a PKI on a stateless blockchain to have far reaching consequences, enabling even very low-powered devices to participate in the storage and the verification operations of the blockchain. It also opens new lines of thought on the aggregating of multiple proofs in a single proof and witness, and the batch verifying of multiple proofs in one instance instead of an iterated verification of all the proofs individually. While the two are certainly possible, the question is about their optimization. The methodology currently used for the witness generation itself can be further optimized with Bezout coefficients [11], and individual verification processes can likewise be made decentralized and transparent with smart contracts, which we hope to do in our future work.

Disclosure statement

The Authors declare that there is no conflict of interest.

Declarations

Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

Acknowledgements

This research is undertaken as part of the project "Research and Development of Secure and Privacy Preserving Blockchain based Smart Contract and its Applications" funded by Science and Engineering Research Board (SERB) [EEQ/2021/000305].

References

[1] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. M. Adams, "X.509 internet public key infrastructure online certificate status protocol—OCSP," RFC 2560, pp. 1–23 (1999). [Online]. Available: https://doi.org/https://api.semanticscholar.org/CorpusID:27218601.

[2] C. Fromknecht, D. Velicanu, and S. Yakoubov, "A decentralized public key infrastructure with identity retention," IACR Cryptol. ePrint Arch., vol. 2014, p. 803 (2014).

[3] A. Yakubov, W. M. Shbair, A. Wallbom, D. Sanda, and R. State, "A blockchain-based PKI management framework," in Proc. NOMS 2018 – 2018 IEEE/IFIP Network Operations and Management Symposium, pp. 1–6 (2018). [Online]. Available: https://doi.org/10.1109/NOMS.2018.8406325.

[4] L. Axon and M. Goldsmith, "PB-PKI: A privacy-aware blockchain-based PKI," in Proc. International Conf. Security and Cryptography, pp. 311–318 (2017). [Online]. Available: https://doi.org/10.5220/0006419203110318. [Accessed: Jan. 14, 2025].

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," [Online]. Available: http://bitcoin.org/bitcoin.pdf. [Accessed: Jan. 14, 2025].

[6] A. J. Joseph Antony and K. Singh, "A blockchain-based public key infrastructure for IoT-based healthcare systems," *The Computer Journal*, vol. 79 (2023). [Online]. Available: https://arxiv.org/abs/https://academic.oup.com/comjnl/advance-article-pdf/doi/10.1093/comjnl/bxad079/51023752/bxad079.pdf. [Accessed: Jan. 14, 2025].

[7] A. J. Antony and K. Singh, "Enhancing privacy in a blockchain-based public key infrastructure," in Proc. 2020 Third ISEA Conf. Security and Privacy (ISEA-ISAP), pp. 87–93 (2020). [Online]. Available: https://doi.org/10.1109/ISEA-ISAP49340.2020.235005. [Accessed: Jan. 14, 2025].

[8] Y. C. Elloh Adja, B. Hammi, A. Serhrouchni, and S. Zeadally, "A blockchain-based certificate revocation management and status verification system," *Computers & Security,* vol. 104, p. 102209 (2021). [Online]. Available: https://doi.org/10.1016/j.cose.2021.102209. [Accessed: Jan. 14, 2025].

[9] Z. Zhai, S. Shen, and Y. Mao, "BPKI: A secure and scalable blockchain-based public key infrastructure system for web services," *Journal of Information Security and Applications,* vol. 68, p. 103226 (2022).

[Online]. Available: https://doi.org/10.1016/j.jisa.2022.103226. [Accessed: Jan. 14, 2025].

[10] A. Panigrahi, A. K. Nayak, and R. Paul, "Smart contract assisted blockchain-based public key infrastructure system," *Trans. Emerging Telecommunication Technologies*, vol. 34, no. 1, p. 4655 (2023). [Online]. Available: https://doi.org/10.1002/ett.4655. [Accessed: Jan. 14, 2025].

[11] D. Boneh, B. Bünz, and B. Fisch, "Batching techniques for accumulators with applications to IOPS and stateless blockchains," *LNCS*, vol. 11692, pp. 561–586 (2019). [Online]. Available: https://doi.org/10.1007/978-3-030-26948-7_20.

[12] A. Panwar and V. Bhatnagar, "Analyzing the performance of data processing in private blockchain-based distributed ledger," *Journal of Information and Optimization Sciences*, vol. 41, pp. 1–12 (2020). [Online]. Available: https://doi.org/10.1080/02522667.2020.1809095.

[13] S. J. M. Muhammed Miah and S. Venkatraman, "Blockchain: At a glance idea for information science researchers," *Journal of Information and Optimization Sciences*, vol. 42, no. 7, pp. 1589–1624 (2021). [Online]. Available: https://doi.org/10.1080/02522667.2021.1930644.

[14] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Comput.*, vol. 17, no. 2, pp. 281–308 (1988). [Online]. Available: https://doi.org/10.1137/0217017.

[15] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," Cryptology ePrint Archive, Paper 2000/025 (2000). [Online]. Available: https://eprint.iacr.org/2000/025.

[16] J. C. Benaloh and M. de Mare, "One-way accumulators: A decentralized alternative to digital signatures (extended abstract)," in Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of Cryptographic Techniques, Lofthus, Norway, May 23–27, pp. 274–285 (1993). [Online]. Available: https://doi.org/10.1007/3-540-48285-7_24.

[17] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in Advances in Cryptology—CRYPTO 2002, M. Yung, Ed., Berlin, Germany: Springer, pp. 61–76 (2002).

[18] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography, Boca Raton, FL, USA: CRC Press (1996). [Online]. Available: http://cacr.math.uwaterloo.ca/hac.

[19] F. Baldimtsi, J. Camenisch, M. Dubovitskaya, A. Lysyanskaya, L. Reyzin, K. Samelin, and S. Yakoubov, "Accumulators with applications to anonymity-preserving revocation," Cryptology ePrint Archive, Paper 2017/043 (2017). [Online]. Available: https://eprint.iacr.org/2017/043.

[20] H. Lipmaa, "Secure accumulators from Euclidean rings without trusted setup," in Applied Cryptography and Network Security, F. Bao, P. Samarati, and J. Zhou, Eds., Berlin, Germany: Springer, pp. 224–240 (2012).

[21] P. Todd, "Making UTXO set growth irrelevant with low-latency delayed TXO commitments," [Online]. Available: https://petertodd.org/2016/delayed-txo-commitments. [Accessed: Jan. 14, 2025].

[22] I. Damgård and M. Koprowski, "Generic lower bounds for root extraction and signature schemes in general groups," in Advances in Cryptology — EUROCRYPT 2002, L. R. Knudsen, Ed., Berlin, Germany: Springer, pp. 256–271 (2002).

[23] B. Wesolowski, "Efficient verifiable delay functions," Cryptology ePrint Archive, Paper 2018/623 (2018). [Online]. Available: https://eprint.iacr.org/2018/623. [Accessed: Jan. 14, 2025].

[24] R. Cramer and V. Shoup, "Signature schemes based on the strong RSA assumption," Cryptology ePrint Archive, Report 1999/001 (1999). [Online]. Available: https://ia.cr/1999/001. [Accessed: Jan. 14, 2025].

[25] P.-A. Fouque and M. Tibouchi, "Close to uniform prime number generation with fewer random bits," in Automata, Languages, and Programming, J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, Eds., Berlin, Germany: Springer, pp. 991–1002 (2014).

[26] N. Koblitz, "Elliptic curve cryptosystems," Math. Comp., vol. 48, pp. 243–264 (1987). [Online]. Available: https://doi.org/10.1090/S0025-5718-1987-0866109-5.

[27] J. Groth, "Linear algebra with sub-linear zero-knowledge arguments," in Advances in Cryptology - C, S. Halevi, Ed., Berlin, Germany: Springer, pp. 192–208 (2009).

Received December, 2023