BDEC: Enhancing Learning Credibility via Post-Quantum Digital Credentials

Zoey Ziyi Li$^{1[0000-0002-6419-0734]}$, Xinyu Zhang $^{*1,2[0009-0008-7764-4265]}$, Hui Cui$^{1[0000-0002-5820-2233]}$, Jun Zhao$^{1[0009-0003-8423-177X]}$, and Xuan Chen$^{1[0009-0002-1538-8792]}$

^{1} Monash University, Melbourne, Australia
`Ziyi.Li,Xinyu.Zhang1,Hui.Cui,Jun.Zhao1,Xuan.Chen1@monash.edu`
^{2} Data61, CSIRO, Sydney, Australia

Abstract. Digital credentials can streamline the verification process, reduce fraud, and enhance the portability and accessibility of academic records, and thus digital credentials become increasingly essential for verifying academic achievements and qualifications in a rapidly digitizing world. However, current research on digital credentials in the education scenario falls into two categories: 1) blockchain-verifiable credentials, which either rely on blockchain security or classical cryptographic problems and are not quantum-safe; and 2) post-quantum anonymous credentials, primarily lattice-based, facing challenges in achieving conditional linkability and efficient revocation.
In this paper, we propose a generic framework for constructing an anonymous credential system named Blockchain-based Digital Education Credential (BDEC). Our framework requires only a digital signature scheme and a zkSNARK. We formally prove the unforgeability, anonymity, and unlinkability of the proposed framework and show that it additionally supports conditional linkability and revocability. By instantiating the framework with the post-quantum signature Loquat [Crypto'24] and zkSNARK Aurora [Eurocrypt'19], we construct a post-quantum anonymous credential system whose security relies on collision-resistant hash functions and the Legendre PRF. Furthermore, our resulting system offers significant improvements in its concrete performance over the lattice-based system [Crypto'23].

Keywords: Digital Credential · Anonymous · Post-Quantum · Blockchain

1 Introduction

Traditionally, education certificates have served as the primary means to prove an individual's educational achievements, skills, and employability in the job market [23]. However, recent incidents have seriously challenged the credibility of educational certificates. For example, a private university in India sold 36,000

*First and second authors contributed equally to this work, and their sequence is according to alphabetical order.

fake degrees for $1,362 each, causing graduates in Singapore, Malaysia, the US, and Canada to face pressure from employers worldwide to verify their qualifications [22]. Additionally, 150 Indian students in Canada faced deportation after being victims of an immigration scam that provided them with forged college admission letters [15]. On the other hand, honest students holding authentic certificates are at risk of credential misuse and theft, as demonstrated by LinkedIn's 2021 breach, which resulted in 700 million users' profiles being sold on the Dark Web [27]. In addition, the rise of micro-credentials, defined by the European Commission [9], as proofs of learning outcomes acquired through short learning experiences, often results in fragmented learning records. This fragmentation not only makes it difficult to track continuous progress but also leads to repetitive proofs of the same skills [10]. These issues underscore the urgent need for a more credible digital credential scheme with the following desired properties, which is the focus of our study:

- Unforgeability: Ensures that credentials cannot be counterfeited.
- Anonymity: Protects user privacy by preventing the linkage of credentials to their identity.
- Unlinkability: Guarantees user privacy one step further as different shown credentials cannot be linked to same identity.
- Conditional Linkability: Supports selective linkability to address fragmented learning records.
- Revocation: Enables the ability to revoke credentials if they are compromised or invalidated, ensuring that only legitimate credentials remain active.

1.1 Related Work

To address the above-mentioned issues, the most state-of-art approaches fall into two categories: blockchain-verifiable credentials and post-quantum anonymous credentials. These recent works build upon the foundational concept of pseudonymous credentials, first proposed by Chaum in 1984 [4], which enables users to sign documents without revealing their identity. The Camenisch and Lysyanskaya (CL) signature scheme [3] further advanced this field by providing a framework for creating secure and efficient anonymous credentials using group signatures and zero-knowledge proofs. Until 2014, Garman et al. introduced decentralized anonymous credentials (DAC), leveraging the benefits of distributed technology to enhance privacy and security [11].

Blockchain-verifiable credentials: In the recent blockchain domain, significant efforts have been made to combat credential forgery and theft. Notably, Damgard et al. introduced a credential identity layer that balances accountability and privacy in blockchain systems to mitigate "Anti Money Laundering" issues [7]. Other proposals have utilized Verifiable Credentials (VC) and blockchain Non-Fungible Tokens (NFTs) for various applications, such as electric vehicle authentication [20] and education credentials [16]. Traceability has also gained considerable research attention in blockchain proposals. Zhuang et al. proposed a privacy-preserving and traceability scheme for intellectual property [29], and similar efforts have been made for numerous supply chain management systems [17,18]. Regarding conditional linkability, Coconut introduced the first selective disclosure attribute-based credential scheme supporting threshold issuance, although it lacked support for traceability or revocation [26]. Shi et al. addressed this limitation by incorporating threshold tracing and revocation properties [24]. In terms of updatability, Garman et al.[11] discussed the creation of updatable (e.g., stateful) anonymous credentials, although their work did not fully implement this feature. Despite these advancements, current approaches largely rely on blockchain security or classical cryptographic problems (strong RSA or Diffie–Hellman assumption) and are not quantum-safe. None of them can satisfy all the desired security properties comprehensively.

Quantum-resistant Credential: Jeudy et al. [12] proposed the first lattice-based anonymous credentials system, leveraging the hardness of the Short Integer Solution (SIS) and Learning With Errors (LWE) problems. Their approach marks a significant step towards quantum-resistant credentials, but it currently supports only a single message, making its use for anonymous verifiable credentials (VCs) cumbersome in terms of data size and performance. Other notable quantum-based proposals include the work of Blazy et al. [2], who provided a generic framework for constructing anonymous credential schemes, and Kazmi et al. [13], who introduced a pseudonymous credential scheme specifically designed for use in payment systems. The only education-focused study is by Shrivas et al. [25], who proposed a quantum-resistant university credentials verification system using lattice-based cryptography and hash-based signatures within a consortium blockchain. However, this study lacks comprehensive security proofs and code implementation. Overall, the current quantum-resistant credential proposals are primarily lattice-based, facing challenges in achieving conditional linkability and efficient revocation, thus highlighting the need for further research, possibly exploring symmetric-key primitives based schemes [8].

1.2 Our Contributions

Our contributions are twofold:

- **Contribution I**. We propose a generic framework for constructing anonymous credential systems that also achieve conditional linkability and revocability. Our framework assumes only *black-box* access to a digital signature scheme and a zero-knowledge succinct non-interactive argument of knowledge (zkSNARK) protocol, allowing for efficient instantiation with post-quantum building blocks. We provide a formal security analysis of the proposed framework, proving its unforgeability, anonymity, and unlinkability. Notably, if the underlying signature and zkSNARK schemes achieve perfect zero-knowledge, which is commonly the case, our system achieves perfect anonymity and unlinkability.

- **Contribution II**. We provide a post-quantum instantiation of our framework using Loquat [28], a post-quantum signature scheme based solely on collision-resistant hash functions and the Legendre PRF. We instantiate the underlying zkSNARK with Aurora [1] and Fractal [5], respectively. To demonstrate the practicality of our system, we analyze and compare the sizes of key pairs, credentials, and credential proofs with those of existing post-quantum anonymous credential systems. Assuming 128-bit security, our instantiation with Loquat and Aurora results in public key, private key, credential size, and credential proof size reductions of 2,447, 86,752, 5.56, and 4.82 times, respectively, compared to the lattice-based anonymous credential scheme proposed by Jeudy et al. [12].

Studies	Garman et al. [11]	Damgard et al. [7]	Coconut [26]	Jeudy et al. [12]	Shrivas et al. [25]	This study
Unforgeability	✓	✓	✓	✓	✓	✓
Anonymity	✓	✓	✓	✓	✓	✓
Unlinkability	–	–	✓	✓	✓	✓
Conditional Linkability	–	–	–	–	–	✓
Quantum-resistance	–	–	–	✓	✓	✓

Table 1. Properties Comparison with Existing Anonymous Credential Systems

1.3 Organizations

The rest of the paper is organized as follows. Section 2 provides a brief review of definitions used in this paper. Section 3 presents the framework of a blockchain-based digital education credential system (BDEC). In Section 4, we detail BDEC based on decentralized anonymous credential schemes, and analyze its security. We evaluate the performance of our proposed BDEC system in Section 5 and conclude the paper in Section 6.

2 Preliminaries

In this section, we briefly describe relevant definitions to be used in this paper.

2.1 zkSNARKs

Let $\mathcal{R}$ be an efficiently decidable binary relation for an NP language $L_{\mathcal{R}}$.

Definition 1 (zkSNARK). *A zero-knowledge succinct non-interactive argument of knowledge (zkSNARK) consists of a tuple of probabilistic polynomial algorithms, denoted as* $(\mathcal{S}, \mathcal{P}, \mathcal{V})$ *such that*

– $\texttt{crs} \leftarrow \mathcal{S}(1^\lambda, \mathcal{R})$. *On input a security parameter λ in unary, the algorithm outputs a common reference string* $\texttt{crs}$.
– $\pi \leftarrow \mathcal{P}(\texttt{crs}, w, x)$. *On input a common reference string* $\texttt{crs}$ *and a witness-statement pair (w, x), the algorithm outputs a succinct argument π.*
– $0/1 \leftarrow \mathcal{V}(\texttt{crs}, x, \pi)$. *On input a common reference string* $\texttt{crs}$*, a claimed statement x and the argument π, the algorithm outputs* 1 *(accept) if π is convincing (i.e., the prover knows a witness w such that $(w, x) \in R$ for some NP relation R).*

2.2 Digital Signature

Definition 2 (Digital Signature). *A signature scheme consists of the following algorithms* $(\texttt{Sig.Setup}, \texttt{Sig.KeyGen}, \texttt{Sig.Sign}, \texttt{Sig.Verify})$*:*

1. $\texttt{Sig.pp} \leftarrow \texttt{Sig.Setup}(1^\lambda)$. *On input a security parameter λ in unary, the algorithm generates a set of public parameters denoted as* $\texttt{Sig.pp}$.
2. $(sk, pk) \leftarrow \texttt{Sig.KeyGen}(\texttt{Sig.pp})$*: On input public parameters* $\texttt{Sig.pp}$ *generated in the setup phase, the algorithm outputs a key pair (sk, pk).*
3. $\sigma \leftarrow \texttt{Sig.Sign}(\texttt{Sig.pp}, sk, m)$*: On input public parameters* $\texttt{Sig.pp}$*, a secret key sk, and a message m, the (randomized) algorithm outputs a signature σ.*
4. $(0/1) \leftarrow \texttt{Sig.Verify}(\texttt{Sig.pp}, pk, m, \sigma)$*: On input public parameters pp, a public key pk, a message m, and a signature σ, the algorithm outputs 1 (accept) if the signature is valid. Otherwise, outputs 0 (reject).*

Due to page limits, we refer readers to the standard security notions of zk-SNARKs and digital signatures as detailed in [19] and [28], respectively.

2.3 Blockchain

Since its adoption in the Bitcoin cryptocurrency [21], the blockchain platform, which is essentially a chain of blocks storing relevant digital data, has been proposed to a wide range of applications. As a distributed ledger, the blockchain enables the data to be stored on a number of nodes (or computers). Due to the fact that the data can be accessed by anybody, it is challenging for any single entity to fully control of the whole blockchain network. Such a distributed feature of the blockchain, integrating with smart contracts [6], allows users to deploy applications on the blockchain without the involvement of any trusted authority.

3 System Framework

In this section, we describe the framework of a blockchain-based digital education credential system, as well as its security requirements.

3.1 System Architecture

Let $N \in \mathbb{Z}^+$ be the maximum attributes $\{a_1, ..., a_N\}$ (here $\{a_i\}_{i\in[1,N]}$ could be Student Name: ABC, University: XYZ University, Major: Law, Year: 2023, etc.) that each student U in the blockchain-based digital education credential BDEC scheme can claim. At the beginning, each student U randomly selects a secret sk_U as the private user-key, and then derives a series of pseudonym public and private keys $(ppk_{U,TA}, psk_{U,TA})$ to communicate with different educational authority TA (e.g., colleges, universities, tertiary training organizations). The public key $ppk_{U,TA}$ plays the role of the pseudonym identity between the student U and the authority TA. When the student U needs to obtain an education credential on certain eligible attributes $\mathbf{A} = \{a_1, ..., a_m\}$ (where $m \in \mathbb{Z}^+ \leq N$), the student U creates a credential $c_{U,TA}$ on attributes $\mathbf{A}$ to the relevant educational institute TA from its keys $(ppk_{U,TA}, psk_{U,TA})$ established with the authority TA, as well as a corresponding proof $\pi_{c_{U,TA}}$ on the credential $c_{U,TA}$. Once the authority TA verifies the authenticity of the credential $c_{U,TA}$ over attributes $\mathbf{A}$ and the proof $\pi_{c_{U,TA}}$ provided by the student U, the information including $(ppk_{U,TA}, c_{U,TA}, \pi_{c_{U,TA}})$ will be published to the blockchain by the authority TA. In Figure 1, we briefly describe the architecture of a blockchain-based digital education credential (BDEC) scheme.

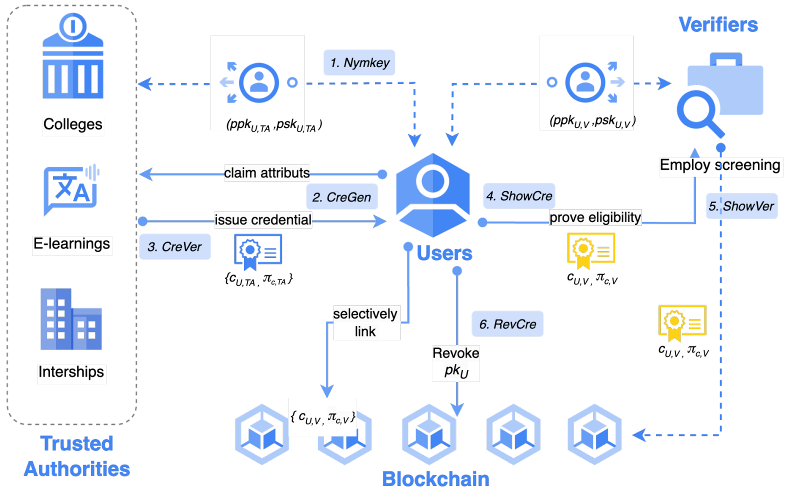

Fig. 1. The Architecture of A Blockchain-based Digital Education Credential Scheme.

Assume that an IT graduate (say Bob) owns two credentials: one on attributes (Name: ABC, University: XYZ University, Degree: Bachelor of Software Engineering, Year: 2023, GPA: 4.8/5) approved by the XYZ University and a credential, and the other one on attributes (Name: ABC, Company: Q Tech, Job: Junior Software Engineer, Level: Internship, Duration: 3 months, Year: 2022).

Considering that a technology company V posts a software engineer job and Bob is willing to apply this job, in this scenario, Bob U needs to prove its capacity to the company V to become short listed. Bob U firstly generates a pseudonym public and private key pair $(ppk_{U,V}, psk_{U,V})$ to communicate with the company V, and then it extracts a shown credential $c_{U,V}$ on a statement that "($ppk_{U,V}$ has a bachelor degree and work experiences in Software Engineer from the XYZ University" and a proof $\pi_{c_{U,V}}$ for the ownership of the shown credential $c_{U,V}$. The company V checks the validity of the credential $c_{U,V}$, as well as the associated proof $\pi_{c_{U,V}}$, and accepts them to the blockchain if it would like to.

Once the shown credential $c_{U,V}$, the proof $\pi_{c_{U,V}}$, and other related information (in one transaction between the student U and the verifier V) is included in the blockchain, it is helpful for the student U to avoid repeatedly proving the ownership of the same attributes to various verifiers. Take the previous scenario as an example, Bob may need to apply several software engineer jobs posted by different companies. Bob may not want to repeatedly generating the ownership proof $\pi_{c_{U,V}}$ about the shown credentials $c_{U,V}$ proving the same statement about his eligibility. In this case, Bob U can simply generate a proof linking the previously approved shown credential on the blockchain rather than a "duplicate" proof on the shown credential.

3.2 Blockchain-Based Digital Education Credentials

A blockchain-based digital education credential BDEC scheme is composed of the following algorithms.

- $(par, L_R) \leftarrow Setup(1^\lambda)$. Taking the security parameter λ as the input, this algorithm, run by the system, outputs the public parameter par and an initially empty revocation list L_R.
- $(sk_U, pk_U) \leftarrow PriGen(par)$. Taking the public parameter par as the input, this algorithm, run by each user U, outputs a long term key pair (sk_U, pk_U). Note that both the secret and public keys will remain hidden unless the user decides to publish pk_U through the $RevCre$ algorithm.
- $(ppk_{U,TA}, psk_{U,TA}) \leftarrow NymKey(par, sk_U)$. Taking the public parameter par and the private user key sk_U as the input, this algorithm, run by the user (and authenticated by the trusted entity TA), outputs a pair of pseudonym public and private keys $(ppk_{U,TA}, psk_{U,TA})$.
- $(c_{U,TA}, \pi_{c_{U,TA}}) \leftarrow CreGen(par, sk_U, pk_U, ppk_{U,TA}, psk_{U,TA}, \mathbf{A}, aux)$. Taking the public parameter par, the user's long term key pair (sk_U, pk_U), the pseudonym public and private key pair $(ppk_{U,TA}, psk_{U,TA})$ and an attribute set $\mathbf{A}$ (and the description aux of attributes $\mathbf{A}$) as the input, this algorithm, run by the user U, outputs a credential $c_{U,TA}$ for attributes $\mathbf{A}$ and a proof $\pi_{c_{U,TA}}$ proving the credential ownership.
- $\{0,1\} \leftarrow CreVer(par, c_{U,TA}, \pi_{c_{U,TA}}, ppk_{U,TA}, \mathbf{A}, aux)$. Taking the public parameter par, the credential $c_{U,TA}$, the proof $\pi_{c_{U,TA}}$ for the credential $c_{U,TA}$, the pseudonym public key $ppk_{U,TA}$ and the attributes $\mathbf{A}$ (as well as the description aux of attributes) as the input, this algorithm, run by the issuing authority, outputs 1 for a valid proof $\pi_{c_{U,TA}}$ or 0 otherwise.

– $(c_{U,V}, \pi_{c_{U,V}}) \leftarrow ShowCre(par, sk_U, pk_U, \{ppk_{U,TA}, psk_{U,TA}, c_{U,TA}, csk_{U,TA}\}$, $\mathbf{A}$, $aux)$. Taking the public parameter par, the private user key sk_U, a set of the pseudonym public key $ppk_{U,TA}$ and private key $psk_{U,TA}$, the credential $c_{U,TA}$ over attributes $\mathbf{A}$ and the corresponding credential private key $csk_{U,TA}$, this algorithm, run by the user U, outputs a shown credential $c_{U,V}$ on attributes $\mathbf{A}$ and the corresponding proof $\pi_{c_{U,V}}$ for the shown credential $c_{U,V}$.
– $\{0,1\} \leftarrow ShowVer(par,\ c_{U,V},\ \pi_{c_{U,V}},\ ppk_{U,TA},\ \mathbf{A}',\ aux)$. Taking the public parameter par, the shown credential $c_{U,V}$ over attribute $\mathbf{A}'$ and its proof $\pi_{c_{U,V}}$ as the input, this algorithm, run by the verifier V, outputs 1 for a valid poof $\pi_{c_{U,TA}}$ and 0 otherwise.
– $pk_U \cup L_R \leftarrow RevCre(par,\ pk_U,\ c_{U,TA})$. Taking the public parameter par, the user's long term public key pk_U, the credential $c_{U,TA}$ to be revoked as the input, this algorithm, run by the user U, adds pk_U to the revocation list L_R.

A BDEC scheme is correct if for any security parameter λ, all participants run algorithms as above, it always hold $CreVer(par,\ c_{U,TA},\ \pi_{c_{U,TA}},\ ppk_{U,TA},\ \mathbf{A},$ $aux) \rightarrow 1$ and $ShowVer(par,\ c_{U,V},\ \mathbf{A},\ \pi_{c_{U,V}}) \rightarrow 1$.

3.3 Security Definitions

A BDEC scheme must meet three basic security requirements, including unforgeability, anonymity and unlinkability.

– **Unforgeability**. This means that anybody, without knowing the private user key of the student U, should not be able to forge a credential $c_{U,TA}$ on behalf of the pseudonym public key established between the user U and the trusted entity TA.
We define the security game of Unforgeability for the BDEC scheme between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$ as follows.
  • Setup. The algorithm $\mathcal{B}$ generates the public parameter par as in the Setup algorithm and sends par to $\mathcal{A}$. In addition, $\mathcal{B}$ initializes an empty list L_U storing each user U's long term key pair (sk_U, pk_U), pseudonym key pair $(ppk_{U,TA},\ psk_{U,TA})$, the credential $c_{U,TA}$ on attributes $\mathbf{A}$, and the credential proof $\pi_{c_{U,TA}}$ authenticated by each trusted entity TA.
  • Query. $\mathcal{A}$ issues the following queries to the algorithm $\mathcal{B}$.
    ∗ $\mathcal{A}$ sends a user-key queries to the `PriGen` oracle on a user U. If there exists a long term user key pair (sk_U, pk_U) for U in the list L_U, $\mathcal{B}$ responds with the corresponding key pair. Otherwise, $\mathcal{B}$ generates (sk_U, pk_U) by running the $PriGen$ algorithm and adds the key pair (sk_U, pk_U) to the list L_U. $\mathcal{B}$ sends (sk_U, pk_U) to $\mathcal{A}$.
    ∗ $\mathcal{A}$ sends a pseudonym key query to the `NymKey` oracle on a user U and a trusted entity TA. If there is already a pseudonym public and private key pair in the list L_U between the user U and the trusted entity TA, $\mathcal{B}$ responds with the pseudonym public key $ppk_{U,TA}$. Otherwise, $\mathcal{A}$ generates a pseudonym public and private key pair $(ppk_{U,TA},$

$psk_{U,TA}$) via running the $NymKey$ algorithm, adds them to the list L_U, and sends the pseudonym public and private key ($ppk_{U,TA}$, $psk_{U,TA}$) to $\mathcal{A}$.

  ∗ $\mathcal{A}$ sends a credential generation query to the `CreGen` oracle on a user U, attributes ($\mathbf{A}$, aux) and a trusted entity TA. $\mathcal{B}$ returns a user credential $c_{U,TA}$ to $\mathcal{A}$ if the credential $c_{U,TA}$ is already in the list L_U. Otherwise, $\mathcal{B}$ runs the $CreGen$ algorithm to create a credential $c_{U,TA}$ over attributes $\mathbf{A}$, along with the credential proof $\pi_{c_{U,TA}}$. $\mathcal{B}$ adds them to the list L_U, and sends the credential and the proof ($c_{U,TA}$, $\pi_{c_{U,TA}}$) to $\mathcal{A}$.

- Output. $\mathcal{A}$ outputs a pseudonym public key $ppk^*_{U,TA}$ for the user U^*, an attribute set $\mathbf{A}^*$, a user credential $c^*_{U,TA}$ and a proof $\pi^*_{c_{U,TA}}$ between the user U^* and the trusted entity TA^*. $\mathcal{A}$ wins this game if $CreVer(par$, $c^*_{U,TA}$, $\pi^*_{c_{U,TA}}$, $ppk^*_{U,TA}$, $\mathbf{A}^*$, $aux^*) = 1$, and the following conditions hold.
  ∗ The user U^* has never been queried to the `PriGen` oracle, and
  ∗ $ppk^*_{U,TA}$ is not the output of `NymKey` oracle, and
  ∗ $c^*_{U,TA}$ is not the output of `CreGen` oracle.

We say that the BDEC system is unforgeable if the advantage function referring to the above security game

$$\mathbf{Adv}^{\mathsf{UNF}}_{\mathsf{BDEC}} = \mathbf{Pr}[\mathsf{BDEC}^{\mathcal{A}} \Longrightarrow 1]$$

is negligible in the security parameter λ for any probabilistic polynomial-time (PPT) adversary algorithm $\mathcal{A}$.

– **Anonymity**. This means that given any pseudonym public key $ppk_{U,TA}$ and the associated credential $c_{U,TA}$, anybody, should not be able to distinguish the real identities of the user U.

We define the anonymity security game for the BDEC system between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$ as follows.

- Setup. The same as that in the Unforgeability game.
- Query 1. The same as that in the Unforgeability game except that $\mathcal{A}$ is not allowed to issue queries to the `PriGen` oracle on users in the Challenge Phase.
- KeyGen. $\mathcal{A}$ sends to $\mathcal{B}$ a list of attributes $\mathbf{A}^*$ (along with the aux that describes the attributes). On input the message from $\mathcal{A}$, $\mathcal{B}$ first generates two pseudonym key pairs $(psk^{(0)}_{U,TA}, ppk^{(0)}_{U,TA})$ and $(psk^{(1)}_{U,TA}, ppk^{(1)}_{U,TA})$. $\mathcal{B}$ forwards ($psk^{(0)}_{U,TA}$, $psk^{(1)}_{U,TA}$) to $\mathcal{A}$.
- Query 2. The same as that in Query 1.
- Challenge. $\mathcal{B}$ randomly selects $b \in \{0,1\}$ and generates credential $c^{(b)}_{U,TA}$ and credential proof $\pi^{(b)}_{U,TA}$ using $psk^{(b)}_{U,TA}$. $\mathcal{B}$ forwards ($ppk^{(b)}_{U,TA}$, $c^{(b)}_{U,TA}$, $\pi^{(b)}_{U,TA}$) to $\mathcal{A}$.
- Query 3. The same as that in Query 1.
- Output. The algorithm $\mathcal{A}$ outputs a guess b' and wins this game if $b' = b$.

We say that the BDEC system is anonymous if the advantage function referring to the above security game

$$\mathbf{Adv}_{\mathsf{BDEC}}^{\mathsf{ANON}} = |\mathbf{Pr}[b' = b] - 1/2|$$

is negligible in the security parameter λ for any probabilistic polynomial-time (PPT) adversary algorithm $\mathcal{A}$.

– **Unlinkability.** This considers the anonymity one step further, which means that given any two pseudonym public keys $ppk_{U,V}$ and their corresponding credentials $c_{U,V}$, anybody should not be able to tell whether they belong to the same user U or not.
  We define the unlinkability security game for BDEC between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$ as follows.
  - Setup. The same as that in the Unforgeability game.
  - Query 1. The same as that in the Unforgeability game except that $\mathcal{A}$ is not allowed to issue queries to the `PriGen` oracle on users in the Challenge Phase.
  - Challenge. The algorithm $\mathcal{B}$ creates two pseudonym key pairs $(psk_{U,V}^{(0)}, ppk_{U,V}^{(0)}$ and $(psk_{U,V}^{(1)}, ppk_{U,V}^{(1)})$ and an attribute set $\mathbf{A}^*$. Then, $\mathcal{B}$ randomly selects $b \in \{0, 1\}$, and creates a credential $c_{U,V}^{(b)}$ with respect to $ppk_{U,V}^{(b)}$ on attributes $\mathbf{A}^*$. $\mathcal{B}$ forwards the credential $c_{U,V}^{(b)}$ to the algorithm $\mathcal{A}$.
  - Query 2. The same as that in Query 1.
  - Output. The algorithm $\mathcal{A}$ outputs a guess b' and wins this game if $b' = b$.

  We say that the BDEC scheme is unlinkable if the advantage function referring to the above security game

$$\mathbf{Adv}_{\mathsf{BDEC}}^{\mathsf{UNL}} = |\mathbf{Pr}[b' = b] - 1/2|$$

  is negligible in the security parameter λ for any probabilistic polynomial-time (PPT) adversary algorithm $\mathcal{A}$.

In addition to the aforementioned three properties, we require the BDEC system to satisfy the conditional linkability and revocability.

– Conditional linkability. This serves two primary purposes: solving the issue of discontinuous learning records (fragmented learning) and avoiding repetitive verification of the same attributes. First, Students often accumulate learning outcomes through various courses and platforms, leading to fragmented records. Conditional linkability allows students to link these fragmented records together, providing a continuous and coherent learning history. Second, students may need to present the same attributes to different verifiers multiple times, requiring repeated generation of shown credentials and proofs. To reduce this workload, the BDEC scheme should be armed with a mechanism enabling students to reuse previous shown credentials and proofs. Specifically, conditional linkability in the BDEC scheme allows students to link previous credentials and proofs to current ones, claiming ownership of the same attributes without regenerating them, as long as these

credentials and proofs are included in the blockchain. If privacy is a concern, students can opt not to link, as their credentials and proofs under various pseudonyms on the blockchain do not reveal their real identities.
- Revocability. In the BDEC scheme, students need to store their private user-keys and pseudonym private keys in private. If any of those secrets is compromised, all associated (shown) credentials need to become inaccessible immediately. To prevent any compromised credential from being continuously utilised by malicious entities, it is of necessity for the BDEC scheme to achieve Revocability such that students can disqualify their approved credentials if they found that their private keys might be compromised.

4 Construction

In this section, we detail a blockchain-based digital education credential system based on decentralized anonymous credential schemes, and analyze its security.

4.1 A Generic Construction

Below, we describe a generic construction of our BDEC system based on digital signature schemes and zkSNARKs. For definitions and further details of the building blocks, we refer readers to Section 2.

- $par \leftarrow Setup(1^\lambda)$. On input a security parameter λ in unary, the algorithm runs $\texttt{Sig.pp} \leftarrow \texttt{Sig.Setup}(1^\lambda)$ and $\texttt{crs} \leftarrow \mathcal{S}(1^\lambda)$, where $\mathcal{S}$ denotes the setup algorithm of the underlying zkSNARK. The algorithm outputs $par = (\texttt{Sig.pp}, \texttt{crs})$. Note that we assume the security parameter λ is an implicit input to all later algorithms.
- $(sk_U, pk_U) \leftarrow PriGen(par)$. On input the public parameters $par = (\texttt{Sig.pp}, \texttt{crs})$, the algorithm runs $(sk_U, pk_U) \leftarrow \texttt{Sig.KeyGen}(\texttt{Sig.pp})$ and outputs (sk_U, pk_U). Note that both the secret and public keys will remain hidden unless the user decides to publish pk_U through the $RevCre$ algorithm.
- $(ppk_{U,TA}, psk_{U,TA}) \leftarrow NymKey(par, sk_U)$. On input the public parameters $par = (\texttt{Sig.pp}, \texttt{crs})$ and the user's long term secret key sk_U, the algorithm randomly selects the pseudonym public key $ppk_{U,TA} \overset{\$}{\leftarrow} \{0,1\}^\lambda$. The algorithm computes the corresponding pseudonym secret key as $psk_{U,TA} \leftarrow \texttt{Sig.Sign}(\texttt{Sig.pp}, sk_U, ppk)$ (i.e., the psk is the signature generated by treating ppk as a message and using sk_U as the signing key). Note that a user U can have as many pseudonym public and secret key pairs as it expects. For privacy concerns, the user U can use different pseudonym public and private key pairs to communicate with different TAs.
- $(c_{U,TA}, \pi_{c_{U,TA}}) \leftarrow CreGen(par, sk_U, pk_U, ppk_{U,TA}, psk_{U,TA}, \mathbf{A}, aux)$. On input the public parameter $par = (\texttt{Sig.pp}, \texttt{crs})$, user's long term key pair (sk_U, pk_U), a pseudonym public and secret key pair $(ppk_{U,TA}, psk_{U,TA})$, a list of attributes $\mathbf{A} = (a_1, \ldots, a_m)$ (together with attributes descriptions aux), the algorithm computes the credential $c_{U,TA} \leftarrow \texttt{Sig.Sign}(\texttt{Sig.pp}, sk_U, h_{U,TA})$.

Note that $h_{U,TA} = \mathcal{H}(\mathbf{A})$ is the hash digest of all attributes, where the definition of the collision-resistant $\mathcal{H}$ is defined in $\texttt{Sig.pp}$. Finally, the algorithm generates a proof $\pi_{c_{U,TA}}$ with respect to the zkSNARK statement $x = (c_{U,TA}, h_{U,TA}, ppk_{U,TA})$ and the witness $w = (pk_U, psk_{U,TA})$:

$$\begin{aligned}\pi_{c_{U,TA}} = \mathcal{P}(\texttt{crs}, x, w) :& \\ &\texttt{Sig.Verify}(\texttt{Sig.pp}, pk_U, h_{U,TA}, c_{U,TA}) = 1 \wedge \\ &\texttt{Sig.Verify}(\texttt{Sig.pp}, pk_U, ppk_{U,TA}, psk_{U,TA}) = 1.\end{aligned}$$

In other words, the circuit to be proven by zkSNARK is defined as the AND concatenation of two signature verification algorithms. The algorithm outputs $(c_{U,TA},\ \pi_{c_{U,TA}})$. The user may submit $(c_{U,TA},\ \pi_{c_{U,TA}}, \mathbf{A}, aux, ppk_{U,TA})$ to the blockchain.

– $\{0,1\} \leftarrow CreVer(par,\ c_{U,TA},\ \pi_{c_{U,TA}},\ ppk_{U,TA},\ \mathbf{A},\ aux)$. On input the public parameter $par = (\texttt{Sig.pp}, \texttt{crs})$, a credential $c_{U,TA}$ on attributes $\mathbf{A}$, a pseudonym public key $ppk_{U,TA}$, a proof $\pi_{c_{U,TA}}$, and the attributes description aux, the algorithm verifies if $\pi_{c_{U,TA}}$ is valid. In particular, the algorithm first computes $h_{U,TA} = \mathcal{H}(\mathbf{A})$. It then defines the NIZKPoK statement $x = (c_{U,TA}, h_{U,TA}, ppk_{U,TA})$ and invokes the zkSNARK verifier $\mathcal{V}(\texttt{crs}, x, \pi_{c_{U,TA}})$. The algorithm outputs 1 if and only if the zkSNARK verifier algorithm $\mathcal{V}$ outputs 1. Otherwise, outputs 0. The TA nodes should only accepts the credential if and only if $CreVer$ returns 1.

– $(c_{U,V}, \pi_{c_{U,V}}) \leftarrow ShowCre(par, sk_U, pk_U, \{ppk_{U,TA}, psk_{U,TA}, c_{U,TA}, csk_{U,TA}\}$, $\mathbf{A},\ aux)$. Assume the user U owns a list of tuples $(ppk_{U,TA}, psk_{U,TA}, c_{U,TA}, \mathbf{A})$, and it intends to prove the eligibility of $\mathbf{A}' = (a_1, \ldots, a_n) \subseteq A$ to a verifier V, where n attributes are embedded in k $(k \leq n)$ credentials $\{c^{(j)}_{U,TA}\}^k_{j=1}$. On input the public parameters $par = (\texttt{Sig.pp}, \texttt{crs})$, user's long term key pair (sk_U, pk_U), a list of the selected pseudonym public key $ppk_{U,TA}$ and corresponding private key $psk_{U,TA}$, the credential $c_{U,TA}$ over attributes $\mathbf{A}$, the algorithm first generates pseudonym keys for communicating with the verifier: $(ppk_{U,V}, ppk_{U,V}) \leftarrow NymKey(par,\ sk_U)$. The user generates the credential $c_{U,V} \leftarrow \texttt{Sig.Sign}(\texttt{Sig.pp}, sk_U, h_{U,V})$ where $h_{U,V} \leftarrow \mathcal{H}(\mathbf{A}')$ and a proof $\pi_{c_{U,V}}$ with respect to statement $x = ((h^j_{U,TA}, ppk^{(j)}_{U,TA})^k_{j=1}, h_{U,V}, c_{U,V}, ppk_{U,V})$ and witness $w = ((psk^{(j)}_{U,TA})^k_{j=1}, pk_U, psk_{U,V})$:

$$\begin{aligned}\pi_{c_{U,V}} = \mathcal{P}(\texttt{crs}, x, w) :& \\ &\wedge^k_{j=1} \texttt{Sig.Verify}(pk_U, ppk^{(j)}_{U,TA}, psk^{(j)}_{U,TA}) = 1 \\ &\wedge \texttt{Sig.Verify}(pk_U, ppk_{U,V}, psk_{U,V}) = 1 \\ &\wedge \texttt{Sig.Verify}(pk_U, h_{U,V}, c_{U,V}) = 1\end{aligned}$$

The circuit proven by NIZKPoK is $k+2$ AND concatenation of signature verification algorithm.

Considering the list of attributes $\mathbf{A}'$ may need to be repeatedly prove to different verifiers. We would recommend first uploading the tuple $(c_{U,V},\ \mathbf{A}',$

aux, $\pi_{c,V}$, $ppk_{U,V}$) which was shown to verifier V to the blockchain. Then, for any future proofs on attributes $\mathbf{A}'$ to a distinct verifier V', the user U can generate a proof $\pi'_{c_{U,V}}$ with respect to the statement $x = (ppk_{U,V}, ppk'_{U,V})$ and witness $w = (pk_U, psk_{U,V}, psk'_{U,V})$, where $(ppk'_{U,V}, ppk'_{U,V}) \leftarrow NymKey(par,$ $sk_U)$:

$$\begin{aligned}\pi'_{c_{U,V}} &= \mathcal{P}(\texttt{crs}, x, w) : \\ &\quad \wedge \texttt{Sig.Verify}(pk_U, psk_{U,V}, ppk_{U,V}) = 1 \\ &\quad \wedge \texttt{Sig.Verify}(pk_U, psk'_{U,V}, ppk'_{U,V}) = 1\end{aligned}$$

The circuit is an AND concatenation of two signature verification algorithms. The user can simply append $(\pi'_{c,V}, ppk'_{U,V})$ to the tuple which was already uploaded to blockchain.

– $\{0,1\} \leftarrow ShowVer(par,\ c_{U,V},\ \pi_{c_{U,V}},\ ppk_{U,TA},\ \mathbf{A}',\ aux)$. On input the public parameter $par = (\texttt{Sig.pp}, \texttt{crs})$, the shown credential $c_{U,V}$ over attributes $\mathbf{A}' = (a_1, \ldots, a_n)$ (together with aux, the attributes description), a proof of credential possession $\pi_{c_{U,V}}$, and the pseudonym public key attached with each credential $ppk_{U,TA}$, the algorithm verifies if $\pi_{c_{U,V}}$ is valid. In particular, the algorithm computes $h_{U,V} \leftarrow \mathcal{H}(\mathbf{A}')$ and $h^{(j)}_{U,TA} \leftarrow \mathcal{H}(\mathbf{A})$ for $j \in [k]$. It defines the zkSNARK statement $x = ((h^{(j)}_{U,TA}, ppk^{(j)}_{U,TA})^k_{j=1}, h_{U,V}, c_{U,V}, ppk_{U,V})$ and invokes the zkSNARK verifier algorithm $\mathcal{V}(\texttt{crs}, x, \pi_{c_{U,V}})$. The algorithm returns 1 if and only if the zkSNARK verifier returns 1. Otherwise, the algorithm returns 0.
– $pk_U \cup L_R \leftarrow RevCre(par,\ pk_U,\ c_{U,TA})$. On input the public parameter par, the user's long term public key pk_U, the credential $c_{U,TA}$ to be revoked, this algorithm publishes pk_U and adds pk_U to the revocation list L_R. For any future submission $(c_{U,V},\ \pi_{c_{U,V}}, \mathbf{A}, aux)$ to the blockchain, the verifier node V selects a public key pk_U from the list L_U and then run an initial check on the proof about the pseudonym public key: $\texttt{Sig.Verify}(pk_U, \pi_{c_{U,V}}, h_{U,V})$, where $h_{U,V} \leftarrow \mathcal{H}(\mathbf{A}')$. If there exists pk_U that makes the verification pass (Sig.Verify returns 1), the submission will be rejected.

4.2 Security Analysis

Theorem 1 (Unforgeability). *The* BDEC *system is unforgeable under the assumption that the underlying signature scheme is existentially unforgeable under chosen-message attacks (EUF-CMA) and that the underlying zkSNARK proof system is knowledge sound.*

Proof sketch: Assume the adversary is able to forge the credential and credential proof (without user's long term secret and public key), then it means the adversary must either forge a valid signature or forge a valid zkSNARK proof.

Theorem 2 (Anonymity). *The* BDEC *system is anonymous under the assumption that both the underlying signature scheme and the zkSNARK are zero-knowledge.*

Proof Sketch: Since pseudonym public keys are just random bit strings chosen by the user, no one should be able to identify the read signer based on these pseudonym public keys. Furthermore, since the signature and zkSNARK are both zero-knowledge, the credential and the credential proof should not leak any information about user's real identity.

Theorem 3 (Unlinkability). *The* BDEC *system is unlinkable under the assumption that both the underlying signature scheme and the zkSNARK are zero-knowledge.*

Proof Sketch: The pseudonym public keys should not be linked to the same user as they are random bit strings. Since the signature scheme is randomized and the zkSNARK is zero-knowledge, the credential and the credential proof should not be linked to the same user.

We show the formal security proofs of unforgeability, anonymity, and unlinkability in Appendix A.

Conditional Linkability. In the *ShowCre* algorithms, the user U is able to link any pseudonym public keys in the proof by showing that two pseudonym keys are generated with the same long term secret key. Thus, for any previously proven attributes (which have been accepted by the blockchain), the user U has the capability to include them in the current credential shown process without proving the ownership of those attributes for a second time.

Revocability. The *RevCre* algorithm supports the credential revocation. Once the user U finds out that its private user key sk_U is lost or leaked, the user U can publish the corresponding public user key pk_U to the revocation list L_R. For any submitted credential $c_{U,V}$ and the proof $\pi_{U,V}$, the verifier nodes will run the signature verification algorithm to check whether the submitted credential $c_{U,V}$ is a valid signature with respect to any public key pk_U in the list L_R.

We further discuss the approach to achieve hidden attributes from verifier in Appendix B.

4.3 Symmetric-Key Primitives Based Post-Quantum Instantiation

We describe the post-quantum instantiation of our BDEC system using the state-of-the-art SNARK-friendly signature scheme Loquat [28]. The security of Loquat relies solely on collision-resistant hash functions and the Legendre PRF, making it post-quantum secure. The primary advantage of Loquat is its balanced performance between signature size/speed and SNARK-friendliness. Since our framework requires the user to prove signature validity during *CreGen* and *ShowCre*, it is crucial that the underlying signature scheme has a relatively small verification circuit. Although one-time signatures such as Lamport+ [14] have better verification complexity, they are unsuitable for system instantiation because the user needs a long-term key pair for signature generation. It is impractical for the user to maintain a state that ensures one-time secret keys are never reused.

We instantiate the zkSNARK with Aurora [1] and Fractal [5], which rely solely on collision-resistant hash functions and features a transparent setup. Both zkSNARKs has proof size polylogarithmic in the circuit size it proves, ensuring small overhead for the credential proof generated in $ShowCre$, especially when the number of TAs is large. Thus, the system is highly scalable. Furthermore, by leveraging recursive proof generation of Fractal, the credential proof produced by $ShowCre$ can maintain a size independent of the number of involved TAs.

5 Performance Evaluation

To provide a general idea of the performance of our proposed **BDEC** system, we analyse two main metrics, namely credential size and running time for $CreGen$, $CreVer$, $ShowCre$, and $ShowVer$ algorithms.

For $CreGen$, the running time is mainly depending on the signing time for the signature and the time for generating the zkSNARK proof for the verification of two signatures. In this analysis, we take Loquat [28] and Aurora [1] as concrete signature and zkSNARK schemes for our generic construction. The signing time for Loquat signature is $O(\lambda)$, where λ is the system security parameter. The proof generation time for Aurora is $O(N \log N)$, where N is the number of gates in the circuit. Since N is a polynomial of λ, this can be written as $O(poly(\lambda) \log \lambda)$. Correspondingly, the running time for $CreVer$ is mainly depending on the verification time for the zkSNARK proof, and for Aurora it is $O(N)$, or $O(poly(\lambda))$.

Similarly, for $showCre$ the running time is mainly depending on the signing time for the signature and the time for generating the zkSNARK proof for the verification of $k+2$ signatures. In this case, the signing time is $O(\lambda)$ and the proof generation time is $O(k \cdot poly(\lambda) \log \lambda)$, where λ is the system security parameter and k is the number of TAs involved and in the worst case $k = |\mathbf{A}'|$. Correspondingly, the running time for $ShowVer$ is mainly depending on the verification time for the zkSNARK proof, hence $O(k \cdot poly(\lambda))$.

For credential size, it is depending on the sizes of the signature and the zkSNARK proof. Table 2 shows empirical data for the running times and credential sizes with 100 bit security, for different k values. We note that the performances get less practical when the parameter k increases to a relatively large value. However, this is due to the performances of the state-of-the-art zkSNARK schemes. Given that our proposed construction is generic and modular, this can be improved by switching the zkSNARK component to a future scheme with better prover time, verifier time and proof size.

Table 3 compares the concrete performance of our scheme, instantiated with Loquat and Aurora, to the previously proposed post-quantum anonymous credential system in [12]. To ensure a fair comparison, we use Loquat with 128-bit security. For the credential proof size comparison, we assume both the number of

	Aurora					Fractal			Fractal-R			(est.)
	$k = 2$	6	14	30	62	2	6	14	2	6	14	30
t_I (s)	-	-	-	-	-	33	88	218	5K	13K	32K	32K
t_P (s)	14	33	67	160	553	119	251	556	3K	6K	14K	14K
t_V (s)	2.4	4.3	10	20	62	0.2	0.47	0.78	2.4	5.64	9.6	9.6
$\|\sigma\|$ (KB)	126	137	182	197	207	131	140	145	131	140	145	145

Table 2. Performance of Credential Proof in *ShowCre* using Aurora, Fractal, and Recursive Fractal (Fractal-R) (adapted from [28, Table 5]). t_I = indexer time, t_P = proving time, t_V = verification time, k = the number of TAs involved.

attributes and TAs to be 10, aligning with the assumptions in [12]. It is important to note that in practice, the number of TAs is often greately smaller than the number of attributes, and the credential proof size in our scheme depends only on the number of TAs, not the number of attributes.

Scheme	$\|pk_U\|$	$\|sk_U\|$	$\|ppk_U\|$	$\|psk_U\|$	$\|c\|$	$\|\pi\|$	Assump.
[12]	9,789	10,844	-	-	317	724	Lattice
Ours	4	0.125	0.125	57	57	150	SK

Table 3. Concrete Performance Comparison of Post-Quantum Anonymous Credential Systems. $|pk_U|$ & $|sk_U|$: User's long term public-secret key pairs; $|ppk_U|$ & $|psk_U|$: User's pseudonym public-secret key pairs; $|c|$: credential size; $|\pi|$: credential proof size. All sizes are in KB. SK denotes Symmetric-Key Based Assumptions (Legendre PRF and Collision-Resistant Hash).

6 Conclusions

In this work, we propose a generic framework for an anonymous credential system. The framework requires only two building blocks: a digital signature scheme and a zkSNARK. We demonstrate that our framework satisfies unforgeability, anonymity, and unlinkability with formal security proofs. Additionally, our framework achieves conditional linkability and revocability, two features absent in existing works.

By instantiating the framework with the post-quantum signature scheme LOQUAT [28] and the post-quantum zkSNARK named Aurora [1], we obtain a post-quantum anonymous credential system whose security relies solely on collision-resistant hash functions and the Legendre PRF. The concrete performance of the resulting system is significantly more practical than existing anonymous credential systems based on lattices [12].

References

1. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for r1cs. In: Eurocrypt. pp. 103–128. Springer (2019)
2. Blazy, O., Chevalier, C., Renaut, G., Ricosset, T., Sageloli, E., Senet, H.: Efficient implementation of a post-quantum anonymous credential protocol. In: Proceedings of the 18th International Conference on Availability, Reliability and Security. Association for Computing Machinery, New York, NY, USA (2023). `https://doi.org/10.1145/3600160.3600188`, `https://doi.org/10.1145/3600160.3600188`
3. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Security in Communication Networks: Third International Conference, SCN 2002 Amalfi, Italy, September 11–13, 2002 Revised Papers 3. pp. 268–289. Springer (2003)
4. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM **28**(10), 1030–1044 (1985)
5. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Eurocrypt. pp. 769–793. Springer (2020)
6. Cuccuru, P.: Beyond bitcoin: an early overview on smart contracts. Int. J. Law Inf. Technol. **25**(3), 179–195 (2017). `https://doi.org/10.1093/IJLIT/EAX003`, `https://doi.org/10.1093/ijlit/eax003`
7. Damgård, I., Ganesh, C., Khoshakhlagh, H., Orlandi, C., Siniscalchi, L.: Balancing privacy and accountability in blockchain identity management. In: Paterson, K.G. (ed.) Topics in Cryptology - CT-RSA 2021 - Cryptographers' Track at the RSA Conference 2021, Virtual Event, May 17-20, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12704, pp. 552–576. Springer (2021). `https://doi.org/10.1007/978-3-030-75539-3_23`, `https://doi.org/10.1007/978-3-030-75539-3_23`
8. Dutto, S., Margaria, D., Sanna, C., Vesco, A.: Toward a post-quantum zero-knowledge verifiable credential system for self-sovereign identity. Cryptology ePrint Archive (2022)
9. European Commission: A european approach to micro-credentials: Final report (2020), `https://education.ec.europa.eu/education-levels/higher-education/micro-credentials`
10. Fitzgerald, R., Huijser, H.: The opportunities and challenges in the portability and authentication of micro-credentials and short courses in a post-covid landscape. Technology-Enhanced Learning and the Virtual University pp. 465–477 (2023)
11. Garman, C., Green, M., Miers, I.: Decentralized anonymous credentials. IACR Cryptol. ePrint Arch. **2013**, 622 (2014), `https://api.semanticscholar.org/CorpusID:6055016`
12. Jeudy, C., Roux-Langlois, A., Sanders, O.: Lattice signature with efficient protocols, application to anonymous credentials. Cryptology ePrint Archive, Paper 2022/509 (2022), `https://eprint.iacr.org/2022/509`, `https://eprint.iacr.org/2022/509`
13. Kazmi, R.A., Le, D.P., Minwalla, C.: Privacy-preserving post-quantum credentials for digital payments. In: International Conference on Financial Cryptography and Data Security. pp. 118–137. Springer (2022)
14. Khaburzaniya, I., Chalkias, K., Lewi, K., Malvai, H.: Aggregating and thresholdizing hash-based signatures using starks. In: AsiaCCS. ACM (2022)

15. Khare, V.: Indian students face exit from canada over fake papers (March 18 2023), https://www.bbc.com/news/world-us-canada-65898527, bBC Hindi
16. Li, Z.Z., Wang, H., Gasevic, D., Yu, J., Liu, J.K.: Enhancing blockchain adoption through tailored software engineering: An industrial-grounded study in education credentialing. Distributed Ledger Technologies: Research and Practice **2**(4), 1–24 (2023)
17. Manoj, T., Makkithaya, K., Narendra, V.: A blockchain-based credentials for food traceability in agricultural supply chain. In: 2023 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER). pp. 19–24. IEEE (2023)
18. Musamih, A., Yaqoob, I., Salah, K., Jayaraman, R., Omar, M., Ellahham, S.: Using nfts for product management, digital certification, trading, and delivery in the healthcare supply chain. IEEE Transactions on Engineering Management (2022)
19. Nitulescu, A.: zk-snarks: A gentle introduction. Ecole Normale Superieure (2020)
20. Parameswarath, R.P., Gope, P., Sikdar, B.: User-empowered privacy-preserving authentication protocol for electric vehicle charging based on decentralized identity and verifiable credential. ACM Transactions on Management Information Systems (TMIS) **13**(4), 1–21 (2022)
21. Ruffing, T.: Cryptography for Bitcoin and friends. Ph.D. thesis, Saarland University, Germany (2020), https://d-nb.info/1209947285
22. Sarkar, S.: India's fake degrees: Hundreds in singapore, malaysia, us, canada left questioning qualifications after manav bharti university scandal (March 3 2021), https://www.scmp.com/week-asia/people/article/3123929/indias-fake-degrees-hundreds-singapore-malaysia-us-canada-left, south China Morning Post
23. Selvaratnam, R.M., Sankey, M.D.: An integrative literature review of the implementation of micro-credentials in higher education: Implications for practice in australasia. Journal of Teaching and Learning for Graduate Employability **12**(1), 1–17 (2021)
24. Shi, R., Feng, H., Yang, Y., Yuan, F., Li, Y., Pang, H.H., Deng, R.H.: Threshold attribute-based credentials with redactable signature. IEEE Transactions on Services Computing (2023)
25. Shrivas, M.K., Kachhwaha, S., Bhansali, A., Singh, S.V.: Quantum-resistant university credentials verification system on blockchain. In: 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON). pp. 1–6. IEEE (2022)
26. Sonnino, A., Al-Bassam, M., Bano, S., Meiklejohn, S., Danezis, G.: Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. arXiv preprint arXiv:1802.07344 (2018)
27. UpGuard: The 72 biggest data breaches of all time (2023), https://www.upguard.com/blog/biggest-data-breaches, accessed: 2023-10-17
28. Zhang, X., Steinfeld, R., Esgin, M.F., Liu, J.K., Liu, D., Ruj, S.: Loquat: A SNARK-friendly post-quantum signature based on the legendre PRF with applications in ring and aggregate signatures. Cryptology ePrint Archive, Paper 2024/868 (2024), https://eprint.iacr.org/2024/868
29. Zhuang, C., Dai, Q., Zhang, Y.: Bcppt: A blockchain-based privacy-preserving and traceability identity management scheme for intellectual property. Peer-to-Peer Networking and Applications **15**(1), 724–738 (2022)

Appendix

A Security Proofs

A.1 Proof of Theorem 1: Unforgeability

Proof. Assume there exists an adversary $\mathcal{A}$ that wins the unforgeability game with non-negligible probability ϵ, we show that there exists an algorithm $\mathcal{B}$ that can break the EUF-CMA of the underlying signature scheme with probability at least $\epsilon - \epsilon_\pi$, where ϵ_π is the knowledge error of the zkSNARK.

Let $\mathcal{A}$ output a tuple $(ppk^*_{U,TA}, \mathbf{A}^*, c^*U, TA, \pi^*_{c_{U,TA}})$ such that $CreVer(par, c^*_{U,TA}, \pi^*_{c_{U,TA}}, ppk^*_{U,TA}, \mathbf{A}^*, aux^*) = 1$. Recall that the $CreVer(\cdot)$ algorithm invokes the zkSNARK verification algorithm $\mathcal{V}(\texttt{crs}, x^*, \pi^*_{c_{U,TA}})$, where $x^* = (c^*_{U,TA}, h^*_{U,TA}, ppk^*_{U,TA})$ for $h^*_{U,TA} = \mathcal{H}(\mathbf{A}^*)$. Since the proof $\pi^*_{c_{U,TA}}$ is valid and the zkSNARK is knowledge sound, there exists a polynomial time extractor of the zkSNARK that can extract a valid witness $w^* = (pk^*_U, psk^*_{U,TA})$ with probability at least $\epsilon - \epsilon_\pi$, where ϵ_π denotes the knowledge error of the zkSNARK. In other words, the extractor outputs a valid signature $psk^*_{U,TA}$ with respect to the message $ppk^*_{U,TA}$ and the signing key sk_U, breaking the existential unforgeability of the underlying signature scheme. □

A.2 Proof of Theorem 2: Anonymity

Proof. We construct a simulator $\mathcal{S}$ that outputs an indistinguishable transcript (including the pseudonym key, the credential, and the credential proof) from the real transcript generated by *CreGen*. Assume that our simulator $\mathcal{S}$ has black box access to the signature's simulator algorithm, denoted as $\mathcal{S}_\sigma$.

On input the adversary messages $\mathbf{A}^*$ and aux, the simulator $\mathcal{S}$ selects a random long term public key pk^*_U and a random pseudonym public key $ppk^*_{U,TA}$. $\mathcal{S}$ invokes the signature simulator $\mathcal{S}_\sigma$ with input $(ppk^*_{U,TA}, pk^*_U)$ to generate a simulated signatures $psk^*_{U,TA}$. Note that pseudonym key pair $(psk^*_{U,TA}, ppk^*_{U,TA})$ is indistinguishable from the real pseudonym key pairs as the signature scheme is zero-knowledge. Similarly, to simulate the credential, $\mathcal{S}$ invokes $\mathcal{S}_\sigma$ with inputs $(h^*_{U,TA}, pk^*_U)$, where $h^*_{U,TA} = \mathcal{H}(\mathbf{A}^*)$. Under the same reason, the simulated credential is indistinguishable from the real credential owing to zero-knowledgeness of the signature scheme.

Finally, $\mathcal{S}$ runs the zkSNARK prover algorithm $\pi^*_{c_{U,TA}} \leftarrow \mathcal{P}(\texttt{crs}, x, w)$ with respect to the statement $x = (c^*_{U,TA}, h^*_{U,TA}, ppk^*_{U,TA})$ and witness $w = (pk^*_U, psk^*_{U,TA})$. As the proof is honestly generated, both $psk^*_{U,TA}$ and $c^*_{U,TA}$ are valid signatures, the credential proof $\pi^*_{c_{U,TA}}$ is valid. Furthermore, due to the zkSNARK achieves zero-knowledge, the proof hides pk^*_U.

Note that if the underlying signature scheme and zkSNARK achieve perfect zero-knowledge, then our system achieves perfect anonymous (i.e., the simulated transcript is perfectly indistinguishable from the real transcript). □

A.3 Proof of Theorem 3: Unlinkability

Proof. We construct a simulator $\mathcal{S}$ that outputs an indistinguishable transcript (the pseudonym key, the credential, and the credential proof) from the real transcript generated from $ShowCre$. Assume that our simulator $\mathcal{S}$ has black box access to the signature's simulator algorithm and zkSNARK's simulator algorithm, denoted as $\mathcal{S}_\sigma$ and $\mathcal{S}_\pi$, respectively.

The simulator $\mathcal{S}$ selects a random long term public key pk_U^* and a random pseudonym public key $ppk_{U,V}^*$. $\mathcal{S}$ invokes the signature simulator $\mathcal{S}_\sigma$ with input $(ppk_{U,V}^*, pk_U^*)$ to generate a simulated signatures $psk_{U,V}^*$. Note that pseudonym key pair $(psk_{U,V}^*, ppk_{U,V}^*)$ is indistinguishable from the real pseudonym key pairs as the signature scheme is zero-knowledge. Similarly, to simulate the credential, $\mathcal{S}$ invokes $\mathcal{S}_\sigma$ with inputs $(h_{U,V}^*, pk_U^*)$, where $h_{U,V}^* = \mathcal{H}(\mathbf{A}^*)$. Under the same reason, the simulated credential is indistinguishable from the real credential owing to zero-knowledgeness of the signature scheme.

Finally, $\mathcal{S}$ invokes the zkSNARK simulator $\mathcal{S}_\pi$ on input $(\texttt{crs}, x)$, where $x = ((h_{U,TA}^{(j)*}, ppk_{U,TA}^{(j)*}), h_{U,V}^*, c_{U,V}^*, ppk_{U,V}^*)$ to obtain a simulated proof $\pi_{c_{U,V}}^*$. The proof is indistinguishable from the real proof due to the zero-knowledgeness of the underlying zkSNARK.

Note that if the underlying signature scheme and zkSNARK achieve perfect zero-knowledge, then our system achieves perfect unlinkability (i.e., the simulated transcript is perfectly indistinguishable from the real transcript). □

B Hidden Attributes from Verifier

In our **BDEC** system, we assume users publishes the credential, credential proof, along with all attributes to the public ledger (e.g., blockchain). Given that many attributes cover general information without sensitive data, this is sufficient for many applications. Nevertheless, there might exist scenarios where a user would like to hide attributes shown to verifiers in $ShowVer$ phase. We recommend the system to incorporate the Merkle tree accumulator to enforce such requirement. To be specific, in the $CreGen$ algorithm, instead of signing the hash digest of all attributes, the user signs the Merkle root of the accumulated attributes (i.e., treating each attribute as a leaf of the tree). Then, the user only needs to publish the Merkle root, credential, and credential proof on chain. The proof can still be verified by TA if we assume TAs have full list of attributes of each user. If the user wants to show a subsets of attributes to a verifier, it can produces an authentication path (a.k.a. the membership proof) for each attribute. Hence, the verifier can effectively verify whether provided attributes were approved by TA, without knowing any additional attributes outside of the subset.