ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

On-Chain Verifiable Credential with Applications in Education

Nopporn Chotikakamthorn[1], Aye Mi San[2] and Chanboon Sathitwiriyawong[3]

ABSTRACT

A verifiable credential (VC) has been standardized and applied in various domains, including education. Due to its immutability, blockchain has been considered and used for credential issuance and verification. Most existing methods, however, are not compatible with the W3C VC standard. In this paper, an on-chain VC issuance and verification method has been described. The method is based on the standard VC data model and applicable to any credential type. It decomposes a VC document into a VC template and the corresponding value array(s). This allows a VC to be issued on-chain in the Bitcoin BTC network, which has a limited data-embedding capacity. The proposed method reduces blockchain resource consumption due to the reusability of a VC template. In addition, it allows the use of a concise VC fingerprint format instead of a full VC for credential exchange. Two issuance modes, namely the full on-chain and partial on-chain, are proposed targeting different use cases. The proposed method has been applied for issuing and verifying two learning credential types. The method was evaluated on the Bitcoin Testnet to measure time and space complexities. With the reduced-size VC fingerprint, the proposed method can embed a VC on a traditional paper-based credential as a compact-sized QR code. The proposed method offered faster VC issuance and verification than an existing standard-based verifiable credential method.

Article information:

Keywords: Verifiable Credential, Blockchain, Learning Credential, Blockchain Overlayed Network

Article history:
Received: March 18, 2024
Revised: May 16, 2024
Accepted: June 27, 2024
Published: July 13, 2024
(Online)

DOI: 10.37936/ecti-cit.2024183.256091

1. INTRODUCTION

One of the drawbacks of a paper-based learning credential is the difficulty in coping with a counterfeit certificate [1-2]. To verify the credential authenticity, traditional methods often require workforce and time to go through credential verification procedures. In addition, emerging trends in education, such as competency-based learning and modular learning/credit-bank programs, call for a more flexible and efficient means of certifying and verifying a learning achievement [3-5]. A digital credential provides a viable alternative to a paper-based credential to solve these problems [6-9]. A digital credential is machine-readable and possibly verifiable through cryptographic digital signature. Existing digital learning credential systems and standards include those of [6, 8-10]. In addition to being a more effective solution to counter fake credentials, digital credentials (or badges) can include more detailed evidence of achievement, not just a grading result as in a conventional academic transcript.

Various forms of digital learning credentials have been proposed to meet the above requirements. One of the most studied approaches is a blockchain-based solution [11-18]. These blockchain-based methods benefit from features of distributed ledger systems such as immunity and unforgery. However, most of these methods are either based on proprietary systems or non-standard credential data models. These problems hinder those systems from global and wide-scale adoption. In the era of globalization, this limits the possibility of cross-credit and credit transfer across organizationally or geographically dispersed institutes. One of the promising solutions to these problems is the introduction of a Verifiable Credential (VC) [10]. A verifiable credential is a standardized

[1,2,3]The authors are with the School of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Bangkok Thailand 10520, E-mail: nopporn@it.kmitl.ac.th, ayemisan.it@gmail.com and chanboon@it.kmitl.ac.th
[1]Corresponding author: nopporn@it.kmitl.ac.th

machine-readable and cryptographically secured digital credential. The W3C VC standard can be applied to issue various types of digital credentials including business, medical, and educational credentials [19-20]. A proof method can be used to verify the authenticity of the claims made by that VC. Current VC cryptographic proof methods can be categorized into digital signature-based and blockchain-based methods. Examples of standard digital signature methods include [21] and [22]. Among blockchain-based methods, [13] is one of the few which adhere to the VC standard.

Blockchain-based credential methods offer some benefits that are desirable in the education domain. As pointed out in [17], due to the persistence of blockchain technology, its usage could make it possible to verify a credential even when the credential's issuer no longer exists or operates. In addition, blockchain networks provide a witness to recorded information. This feature is useful or even necessary in certain cases. For example, a learning credential issued at the time when the signing key was valid should remain a valid credential despite the key being later revoked. Reissuing those previous credentials every time the key is revoked is not practical or not possible (e.g., an issuer has already ceased operations). This issue calls for time-stamp evidence for credential proof. A non-blockchain proof method typically requires a trusted third party to provide a witness on a credential issuing time.

In this paper, a blockchain-based verifiable credential system is described. Unlike most other blockchain-based methods, the proposed method is standard-based, using a credential document format described in [10]. Unlike Blockcert, where a blockchain is only used as a signing mechanism (i.e., as a digital signature), the proposed method uses the blockchain as an integrated part of the system for storing, resolving, and verifying a credential. This allows the proposed method to issue a VC on-chain. Usage scenarios for this on-chain VC method include:

- The case where VCs are required to be publicly available. A VC of a revocation list is one such example. Also, making credentials of prestigious awards or achievements publicly known could be desirable from a holder's point of view.
- An issued VC needs to be verifiable without any supportive service from its issuer. For example, this could be the case of verifying a learning credential, previously issued by an academic institute which since then ceased operations.
- A VC is used to complement a physical credential to form a hybrid credential solution. Although a physical credential might be obsolete and disappear from its use some days in the future, it currently remains a standard credential form in most parts of the world. Also, a credential in its physical or tangible form can withstand technological changes. By embedding a link (e.g., a QR code) to its digital counterpart, a physical credential can benefit from the integrity and authenticity proof provided by the linked VC. The proposed on-chain VC can be represented by a concise VC fingerprint form, to be used as a linkage between a pair of physical and digital credentials.

To use in a wide range of applications, a credential method should be adaptable to various blockchain systems. At this stage of development, however, the proposed method was designed based on the Bitcoin network. Bitcoin (BTC) was selected in this study for two reasons. First, it is the oldest and one of the most stable and secured public blockchain networks. The BTC network was also chosen in [13] for this reason. Second, it is the least common denominator among current major blockchain systems regarding the level of support and flexibility in building a decentralized application. Any method that works under such constraint is expected to be adaptable to other blockchain networks (e.g., BSV, BCH, Ethereum). While one of the main criticisms of BTC is its comparatively high transaction cost, the proposed method allows a batch of credentials to be issued in a single transaction. This can significantly decrease the cost per credential to justify its use in practice economically.

The paper is organized as follows. Section 2 describes the related works on digital credentials and blockchain overlayed protocols. In Section 3, a blockchain-based on-chain verifiable credential method is introduced. Its implementation and application in education are next described in Section 4. The evaluation result is then reported. Finally, Section 5 provides concluding remarks.

2. RELATED WORK

In this section, a review of existing works on blockchain-based learning credentials is first discussed. It is then followed by a discussion on the use of Bitcoin to build an overlayed or application-layer protocol.

2.1 Blockchain-based Digital Credential Systems

In this paper, a digital credential based on the VC standard as defined by W3C is considered [10]. Under the VC standard framework, three parties are involved in issuing and verifying a VC. They are the issuer, the holder, and the verifier. The issuer issues a credential containing information about a subject as a set of claims to the holder. The holder is responsible for storing and using the issued credential. The verifier is the one who needs to validate the claims in the credential received from the holder. JSON-LD is the main VC document format. In addition to its core standard [10], a verifiable credential is enabled by digital signature technology, which makes it possible for a digital credential to be verifiable. The VC standard has been developed in such a way that various digital signature methods can be applied and used as a proof method. Examples of digital signature standards that have been proposed as a VC proof method include [21-25].

While there have been many blockchain-based credential methods appearing in the literature, only those in [13, 24] and the future version of [26-27] comply with the VC standard. Blockcert [13] is a decentralized application running on top of the Bitcoin network to provide services differing from the core Bitcoin services. The rules that define how the blockchain is used within each application's context are commonly known as an overlayed or application-layer protocol. For all such protocols, data is stored in an OP_RETURN output and is used by an external system outside the Bitcoin network. In [13], a Merkle root constructed from an aggregation of credential hashes was stored in an OP_RETURN output. It is used as proof that the credentials corresponding to the leaf-node hashes were issued by the one who owned the UTXO spent in the transaction.

In [14], a two-layer approach was proposed for issuing and verifying academic credentials. A public blockchain was only used to store a summary of state changes in a private blockchain where actual learning records were stored. Disclosure of private information was allowed for both issuers and holders. A public blockchain was used to verify the existence and authenticity of information stored in an educator's private blockchain. In [15], a blockchain was used to store the hash of a signed credential through the Credential Registry smart contract. Optionally, the blockchain could be used to store a credential header containing a list of attributes that must always be disclosed. A redactable signature was applied in [15] to make selective attribute disclosure possible.

Another blockchain-based system was described in [16] for recording and verifying students' course completions in higher education. Blockchain tokens were used to represent a course credit value. A registered higher-education institute sends a certain number of tokens to students' addresses for their academic achievements. Along with the number of tokens, the system also stored the sender's official name and a course identification. In [17], a blockchain was used to record a series of events. Each event was a claim made about a document, an issuer, or a receiver (holder). An event for document verification contained information about the corresponding issuer, receiver, and linked documents. To save a transaction cost, one or more events could be bundled, and the bundle hash was registered in a single blockchain transaction. The raw data of an event and the linked document(s) were stored in a DHT decentralized storage such as IPFS. The method allowed an issuer to form a group of documents to enforce full disclosure of all documents in the group. Instead of directly signing and publishing a document, as in the case of Blockcert, the method added an event as a layer of abstract, to which one or more documents were linked.

In [26], a complete system was developed for the issuance and verification of academic credentials. It was based on the OpenAttestation framework [27], with some parts tailored to suit the specific needs of the education domain. OpenAttestation is a framework for deploying verifiable credentials. The framework supports two methods for issuing a credential. The first uses a digital signature method (such as ECDSA), and the underlying Ethereum-based DID system infrastructure, for document authentication. The other one employs the Ethereum blockchain and a smart contract for keeping records of credential documents and their statuses. With a blockchain-based issuing method, a hash of a learning credential (or a Merkle root of the credential hash tree) is stored in the blockchain when a certain smart contract's function is called. If the blockchain-based method is used, when a key is compromised, only documents signed after the leakage need to be revoked. On the other hand, if the digital signature method is used, all documents that have been issued by the revoked key must be invalidated. This scenario demonstrates one of the benefits of using a blockchain-based method instead of an off-chain signature-based method. For many VC applications including its use in education, it is not practical to revoke all past credentials, then reissuing the replacement ones and sending them to the respective holders. In this case, for a signature-based verifiable credential to be of practical use, a trusted third-party authority is needed to provide a timestamping service [28-29]. Use of a timestamping service is not uncommon in digital document signing and is well supported by many document authoring and processing tools. However, this will add extra time and cost to the VC issuing and verifying tasks. Like other recent methods based on Ethereum or smart-contract blockchains [30-32], the current version (v2) of OpenAttestation, and thus OpenCert, uses its own verifiable credential data model, which is incompatible with the W3C VC standard. However, the framework has been under revision and its version 3 data model was promised to be compatible with that of the VC standard. This method and Blockcert are just the two blockchain-based verifiable credential methods that adhere to the VC standard data model [10].

The use of a QR code to embed a digital credential on its physical counterpart has been considered by some previous works [18, 27, 33]. The main concern is on credential data size against the feasible QR code's physical and embedding sizes. While the system in [27] supports encoding a credential as a QR code, a credential size in many cases could either exceed the code embedding capacity or result in an excessive physical code size. In [33], the problem was solved by only embedding the credential values in a QR code. However, there is no detail described for the document schema used in [33], which may not be compatible with the VC data model standard. Also, the method in [33] was based on a fixed set of document attributes. It is not clear how it can be applied to a VC with different sets of attributes, without making changes in some ways at an implementation level. The method proposed in this paper offers a concise and generic VC fingerprint, allowing its QR code to be used as a linkage between any pair of physical and digital credentials.

2.2 Bitcoin Overlayed Protocols

The Bitcoin network was traditionally used mainly to transfer a digital currency, a BTC coin. However, over time, new applications of the network have since appeared [34-37]. These applications are defined by their associated protocols. Those overlayed (or application-layer) protocols rely on the immutability and openness of Bitcoin while delivering new services by adding some metadata to Bitcoin transactions. Most, if not all of them, make use of a special Bitcoin instruction called OP RETURN, to add arbitrary data as part of an unspendable transaction output. For the main Bitcoin network (BTC), the maximum data size that can be added to a blockchain transaction by this method is 80 bytes. Alternative Bitcoin networks, such as Bitcoin Cash (BCH) and Bitcoin SV (BSV), offer a larger OP_RETURN data size (over 200 bytes).

3. ON-CHAIN VERIFIABLE CREDENTIAL PROTOCOL

On-chain VC protocol is a blockchain overlayed protocol used to create and store a verifiable credential either entirely or partially on-chain. The protocol was designed such that it works under the limited OP_RETURN payload capacity of the BTC network, and minimizes the use of expensive blockchain resources. It is based on the concept of a VC template, which can be regarded as a digital equivalent of a document form. The proposed method separates a VC document into a VC template and an ordered set of key values (later called the value array and denoted by $\boldsymbol{v}$). As a result, a VC can be presented or distributed using a combination of the value array and the reference to the corresponding VC template. Such a combination, named a VC fingerprint here, is smaller in size than its original VC document. The proposed method makes it possible to

- Publish the VC itself on-chain so that it can be accessed by anyone and at any time in the future (see Section 2 for some of the use cases).
- Make it possible to embed a compact VC fingerprint in a paper or physical credential using a convenient and widely-used QR code.
- Publish standard VC templates and make them available for anyone to issue their own VCs.

The lifecycle of the on-chain VC protocol is illustrated in Fig. 1. To define a new on-chain VC specification, a Protocol Specification (PS) transaction must first be issued and included in the blockchain network. The purpose of this transaction type is to

- obtain a unique protocol identifier, which is the TxID of a successfully recorded transaction, and to
- declare information describing a protocol specification, which includes a usage-specific VC template.

Anyone can use a protocol specification that has successfully been included through the PS transaction, by initializing the protocol instance through the Protocol Instance Initialization (PI) transaction. This transaction provides a starting point for the chain of subsequent Protocol Data (PD) transactions. It refers to the protocol used by the TxID of that protocol's PS transaction. Also, chain-specific data is defined here.

A PD transaction is where a single or multiple VCs are issued. There can be many PD transactions, issued one after another, until the instance chain is closed or terminated. When the instance chain is no longer needed or must be terminated for any reason, a PT transaction is issued. The only purpose of this transaction is to stop and avoid future use of the instance.

A structure of the on-chain VC protocol data for any of these transaction types is described in Table 1. A detail of each transaction type is given next.

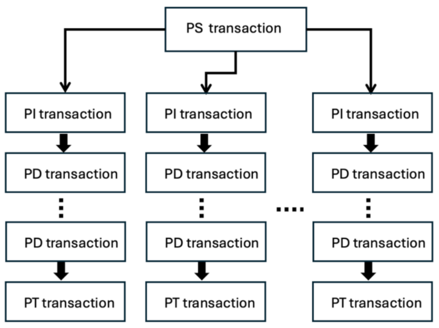


Fig.1: *The On-chain VC Protocol Life Cycle.*

3.1 Protocol Specification (PS) Transaction

To register a new on-chain VC protocol specification, the PS transaction is created and issued to the blockchain network. What is contained in the PS transaction is the protocol specification, which includes the protocol description and a usage-specific VC template. The OP_RETURN payload data structure is shown in Table 2. From the table, the payload starts with the prefix 'OCVC' to ease indexing work by a third-party service provider. Note that, to create an actual transaction payload, each field in Table 2 must be prepended with a block-length byte as specified by the OP_PUSHDATA function. The fields associated with the PS transaction will be subsequently denoted by the subscript S (e.g., T_S stands for the TYPE field of the PS transaction).

Table 1: *Structure of an on-chain VC transaction output data (OP_RETURN output).*

Field Name	Size	Description
PROTOCOL IDENTIFIER / PREFIX	4 bytes	"OCVC"
PAYLOAD TYPE (*T*)	1 byte	Transaction type is defined by the high-order 3 bits: 000: PS transaction 001: PI transaction 010: PD transaction 011: PT transaction 100 - 111: reserved for future use.
DATA (*D*)	Variable length	
EXTRA (*X*)	Variable length	Additional CBOR-encoded data (optional)

Table 2: *Payload detail for the PS transaction.*

Field Name	Size	Description
TYPE	1 byte	The high-order 3 bits: 000 The low-order 5 bits: Define how the protocol specification is declared. 00000: Text (UTF-8) 00001: Hashlink URL 00010: IPFS CID 00011: File hash 00100 – 11111: reserved
DATA	Variable length	Protocol specification, declared using the method defined by the low-order 5 bits of the TYPE field.

Table 3: *Payload detail for the PI transaction.*

Field Name	Size	Description
TYPE	1 byte	The high-order 3 bits: 001 The low-order 5 bits: Define how the protocol and the corresponding PS transaction are identified. 00000:TxID of the PS transaction (binary) 00001 – 11111: reserved
DATA	Variable length	TxID of the PS transaction corresponding to the used protocol specification

3.2 Protocol Instance Initialization (PI) Transaction

After the PS transaction is included in a Bitcoin block, the transaction is referred to by its TxID. To use the protocol specification, the protocol instance initialization (PI) transaction is next submitted. This transaction contains at least one input and two outputs. The first input must be the UTXO belonging to the issuer's Bitcoin address. The first output must generate the UTXO for the same Bitcoin address as that of the first input. The second output is used to store protocol data using the OP_RETURN opcode and is, therefore, unspendable. The issuer's Bitcoin address will be used to associate the subsequent protocol data transactions with this transaction.

The payload structure of the PI transaction is shown in Table 3. From the table, the DATA subfield inside the payload contains the identifier of the specification to use. The optional initialization data could be added to the EXTRA field (see Table 1). The fields associated with the PI transaction will be subsequently denoted by the subscript I (e.g., T_I stands for the TYPE field of the PI transaction).

Table 4: *Payload detail for the PD-type transaction.*

Field Name	Size	Description
TYPE	1 byte	The high-order 3 bits: 010 The low-order 5 bits: Payload data type 00000: Text (UTF-8) 00001: CBOR 00010: Hash value (SHA-256)
DATA	Variable length	Data recorded by the transaction, with the type defined by the low-order 5 bits of the TYPE field.

Table 5: *Payload detail for the PT transaction.*

Field Name	Size	Description
TYPE	1 byte	The high-order 3 bits: 011 The low-order 5 bits: Payload data type 00000: Text (UTF-8) 00001 – 11111: reserved
DATA	Variable length	Data accompanying the termination transaction. The format is defined by the low-order 5 bits of the TYPE field.

3.3 Protocol Instance Data (PD) Transaction

After the PI transaction is included in a block, the UTXO corresponding to the first output of that transaction is used to create the first PD transaction. As for the PI transaction, the PD transaction contains at least one input and two outputs. The essential requirement regarding the input and outputs for any subsequent PD transaction is the same as that of the PI transaction. As a result, the PI, PD, and PT transactions form a chain of protocol instance transactions.

The payload structure of the PD transaction is shown in Table 4. From the table, the DATA subfield inside the payload contains information on each credential's key values, where its data type is specified by the low-order 5 bits of the TYPE subfield. The fields associated with the PD transaction will be subsequently denoted by the subscript D (e.g., T_D stands for the TYPE field of the PD transaction).

3.4 Protocol Instance Termination (PT) Transaction

When the issuer of the protocol-instance transaction chain no longer wants to use the created chain, he/she can stop future use of the chain by issuing the PT transaction. The essential requirements regarding the transaction's input and outputs are the same as those of the PI and PD transactions. The payload structure of the PT transaction is shown in Table 5. From the table, the DATA subfield inside the payload contains any text description (e.g., reasons for termination). The fields associated with the PT transaction will be referred to by the subscript T (e.g., T_T stands for the TYPE field of the PT transaction).

Given the foundation of the on-chain VC protocol described above, what is contained in the payload and extra fields for the four transaction types is shown in Table. 6.

Table 6: *Payload detail for the four transaction types.*

Field Name	Data Type	CONTENT
D_S	Hashlink/CID/ Text/File hash	A VC protocol specification written in the following multi-graph JSON-LD format: [{ ••• }, { ••• }, { ••• }] where the three objects in the above array are a protocol description, a VC template, and an external linked data object.
D_I	Binary	TxID of the protocol's PS transaction
X_I	CBOR	Common key values
D_D	CBOR	VC-specific key values
D_T	Text	Reason to terminate

3.5 The VC Template and Value Array Formats

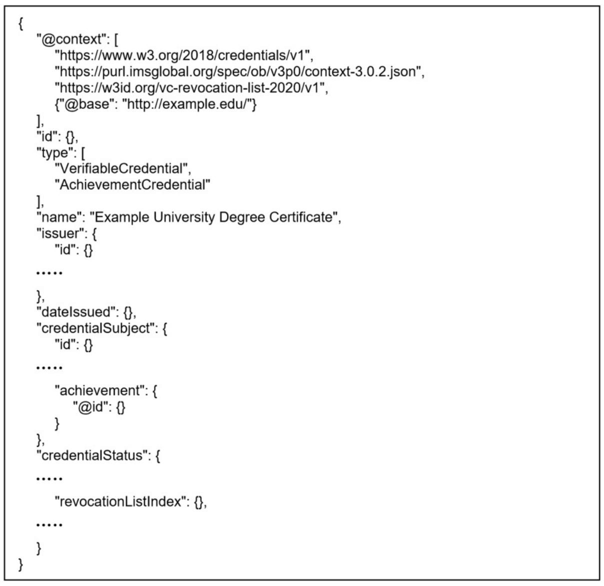
The VC credential template (also denoted by C_T) is a normalized JSON-LD document with some key values being substituted by empty objects or arrays (defined by a pair of curly braces or square brackets respectively, in JSON). The template format was defined such that it meets the following requirements.

- The template should be written so that it remains a valid JSON document. This allows the template to be processed by existing JSON processors, software libraries, or programming languages that have built-in support for a JSON object.
- The template should remain neutral to any templating language or engine. This offers openness and flexibility in how the proposed method is implemented.

Fig. 2 shows an example of a VC template. It is a template of a typical university degree credential.

Value arrays can be stored in both PI and PD transactions. The value array stored in the PI transaction is used to fill the template's empty arrays, while that of the PD transaction is used to fill the template's empty objects. The value array format was defined in such a way that it can be efficiently encoded using CBOR. Also, it can be processed by existing programming languages, templating engines, and object serialization methods. The array construction rules are summarized below.

1. Key values of a document (root) object are stored as elements of an array.
2. If a key value in 1) is an object, an empty array is first created as a placeholder. Then, the object's key values are stored as another array, called a child array. The child array is finally placed inside the placeholder array.
3. If a key value in 1) is an array of objects, additional child array(s) is created just like the first child array, as explained in 2). The resulting array is then appended to the placeholder array.

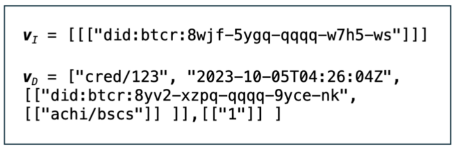


```
{
   "@context": [
      "https://www.w3.org/2018/credentials/v1",
      "https://purl.imsglobal.org/spec/ob/v3p0/context-3.0.2.json",
      "https://w3id.org/vc-revocation-list-2020/v1",
      {"@base": "http://example.edu/"}
   ],
   "id": {},
   "type": [
      "VerifiableCredential",
      "AchievementCredential"
   ],
   "name": "Example University Degree Certificate",
   "issuer": {
      "id": {}
   .....
   },
   "dateIssued": {},
   "credentialSubject": {
      "id": {}
   .....
      "achievement": {
         "@id": {}
      }
   },
   "credentialStatus": {
   .....
      "revocationListIndex": {},
   .....
   }
}
```

Fig.2: *A VC template of a degree credential.*

Examples of the value arrays are shown in Fig. 3. Note that blank spaces are included in these arrays for readability purposes only. Fig. 4 shows the complete VC document corresponding to the template and value arrays described in Figs. 2-3.

By decomposing a VC document into a VC template and the corresponding value array(s), a VC can be distributed using a compact VC fingerprint format. In addition, to create a proof value, a simpler JSON canonicalization algorithm (JCS) can be applied instead of a more complicated JSON-LD RDF canonicalization algorithm as required by other proof methods such as [13, 24]. A VC is typically written as a JSON-LD document and therefore can contain linked data. To ensure that the claims made by a VC cannot be altered through modification of externally linked data, RDF canonicalization is normally required to create a proof value. In our method, a VC template can include a copy of externally linked data. As a result, the template document is self-contained and can be made immutable by including the PS transaction in a blockchain. An example of the VC template, preceded by a protocol description to form a complete protocol specification, is shown in Fig. 5. From the figure, the template includes 3 JSON-LD objects, placed inside an array. The first object represents a protocol description. It is followed by the main VC template. The last one is for externally linked data.


```
vI = [[["did:btcr:8wjf-5ygq-qqqq-w7h5-ws"]]]

vD = ["cred/123", "2023-10-05T04:26:04Z",
[["did:btcr:8yv2-xzpq-qqqq-9yce-nk",
[["achi/bscs"]] ]],[["1"]] ]
```

Fig.3: *An example of the value arrays.*

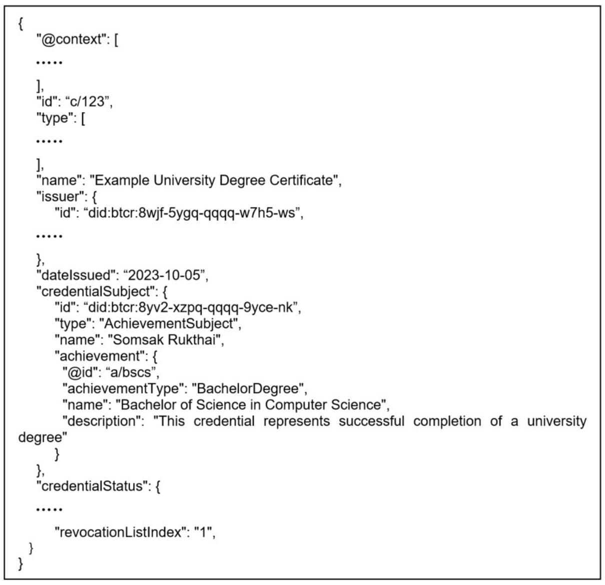

```
{
  "@context": [
  .....
  ],
  "id": "c/123",
  "type": [
  .....
  ],
  "name": "Example University Degree Certificate",
  "issuer": {
     "id": "did:btcr:8wjf-5ygq-qqqq-w7h5-ws",
  .....
  },
  "dateIssued": "2023-10-05",
  "credentialSubject": {
     "id": "did:btcr:8yv2-xzpq-qqqq-9yce-nk",
     "type": "AchievementSubject",
     "name": "Somsak Rukthai",
     "achievement": {
       "@id": "a/bscs",
       "achievementType": "BachelorDegree",
       "name": "Bachelor of Science in Computer Science",
       "description": "This credential represents successful completion of a university
degree"
     }
  },
  "credentialStatus": {
  .....
     "revocationListIndex": "1",
  }
}
```

Fig.4: *A VC document rendered from the value arrays and the VC template in Fig. 2-3.*

Steps to render a VC document ($\boldsymbol{C}_R$) from its template ($\boldsymbol{C}_T$) and one or more value array $\boldsymbol{v}$ (such as $\boldsymbol{v}_I$ and $\boldsymbol{v}_D$) are described by the algorithm in Fig. 6.

3.6 On-chain VC Issuance

There are two issuing modes for the proposed method. The first one is to store the two value arrays ($\boldsymbol{v}_I$ and $\boldsymbol{v}_D$) directly in blockchain transactions. This method will be referred to as the 'Full On-chain' (FO) mode. Let $H(.)$ be a hash function. The second issuing method stores the value array $\boldsymbol{v}_I$ in the PI

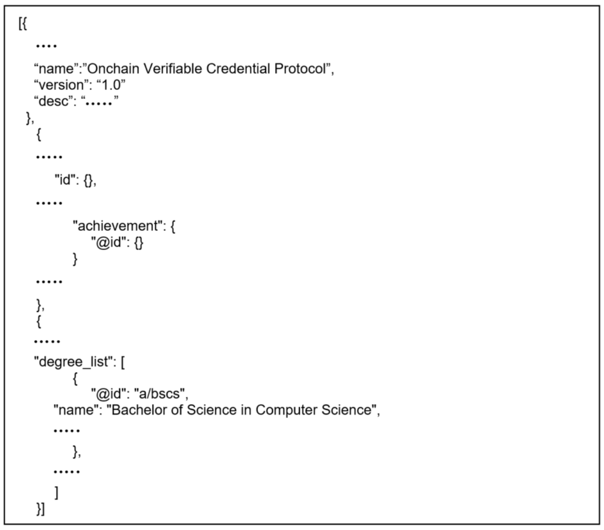

```
[{
  ....
  "name":"Onchain Verifiable Credential Protocol",
  "version": "1.0"
  "desc": "....."
 },
  {
  .....
     "id": {},
  .....
       "achievement": {
          "@id": {}
       }
  .....
  },
  {
  .....
  "degree_list": [
       {
          "@id": "a/bscs",
     "name": "Bachelor of Science in Computer Science",
     .....
       },
     .....
    ]
  }]
```

Fig.5: *A protocol specification containing a protocol description, a VC template, and external linked data.*

transaction, and only $H(\boldsymbol{v}_D)$ or its Merkle tree root hash is published on-chain. The latter method will be referred to as the 'Partial On-chain' (PO) mode. The two modes serve different usage scenarios. The FO mode is aimed at the scenarios where it is required or desirable for a VC to be publicly available. Examples of such usage include publishing a revocation list or BTCR DID document as a verifiable credential, as well as issuing a VC containing claims on prestigious achievements or awards. The PO mode is suitable when a large number of credentials are to be issued, or privacy protection is prioritized. With the PO mode, issuing bulk credentials on a single data transaction is made possible by creating a Merkle tree of the credential document hashes. In this case, the resulting Merkle root hash is stored in D_D instead of $H(\boldsymbol{v}_D)$.

The algorithm for issuing an on-chain VC for any of the two modes is described in Fig. 7 (see [38] for the underlying algorithm of **mkTree**()).

3.7 VC Fingerprint Format for Compact Credential Distribution

To embed a verifiable credential in a physical credential, the VC data size is constrained by many factors including the capacity of the QR code itself and the available space on the embedding media. The on-chain VC offers a much-reduced VC size by distributing only key-value data of the VC. This concise VC format is referred to as a VC fingerprint and is denoted by C_f. In addition to $\boldsymbol{v}_D$, C_f contains additional information, such as TxID of the transaction where the credential was issued and a verification method. These components are separated into three parts, from where the complete VC fingerprint is obtained. The structure of C_f is shown in Fig. 8. By borrowing the JWT format, the fingerprint consists of the header, the payload, and the proof, with two

ALGORITHM 1: VC Document Rendering Algorithm

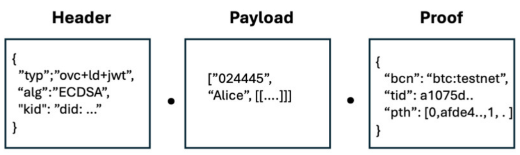
```
Input: $C_T$, v, T. //T ∈ {'object', 'array'}
Output: $C_R$  //The rendered verifiable credential
Initialization: Let AA = 'array', OB = 'object'
[$C_R$, i] ← renderVC($C_T$, v, T)
function renderVC(o, v, T) //a function for recursive call
    i ← -1
    keys ← getKeys(o) //get a list of (JSON) keys in o
    for (k in keys)
        val_k ← o[k] //each key's value
        t_vk ← typeof val_k //data type of each value
        l_vk ← len(val_k) //length of each value
        is_emptyA ← t_vk == AA & l_vk == 0)
        if (t_vk != OB & !is_emptyA) then
            //no empty object or array
            continue
        end
        if (t_vk == T & l_vk == 0)
            i ← i + 1
            o[k] ← v[i]. //fill with value in v
        else //non-empty object-type value
            if (typeof v[i + 1] == typeof v[i + 1][0] == AA)
                i ← i + 1
                o_tmp ← []
                for (v_i in v[i])
                    o_k ← val_k //make a copy of val_k
                    [C_i, j] ← renderVC(o_k, v_i, T)
                    if (j == -1)
                        i ← i - 1
                        break
                    end
                    o_tmp.push(C_i) //append to o_tmp
                end
                if (len(o_tmp) == 1) //no. of elements in o_tmp
                    o[k] ← o_tmp[0]
                elseif (len(o_tmp) > 1)
                    o[k] ← o_tmp
                end
            end
        end
    end
    return (o, i)
end
```

Fig.6: *VC document rendering algorithm.*

ALGORITHM 2: On-Chain VC Issuing Algorithm

```
Input: utxo_in, k_pr, Vd, M
// the utxo(s) to spend, issuer's private key, an array of vd
//issuing mode M ∈ {'FO', 'PO'}
Output: txid, utxo_out, h_tr //transaction id, output utxo(s), and a hash tree
Initialization: h_tr ← null
if (M == 'FO') then // 'FO' mode
    TYPE ← 0x40 | 0x01. //data trans + CBOR payload
    v ← cbor(Vd [0]) //cbor encoding the value array
    Dd = genPayLoad(v, TYPE) //get payload (see table 1-6)
else
    TYPE ← 0x40 | 0x02 //data trans + binary payload
    h_tr ← mkTree(Vd) //construct a Merkle hash tree
    Dd ← genPayLoad(h_tr.root, TYPE) //generate payload
    //according to table 1-6, hr_tr.root is the Merkle root hash
end
[txid, utxo_out] ← bcSend(Dd, k_pr, utxo_in) //issue to a
//Bitcoin blockchain with data output (OP_RETURN). utxo_out
contains information about the transaction outputs
```

Fig.7: *On-chain VC issuing algorithm.*



```
{
 "typ";"ovc+ld+jwt",
 "alg":"ECDSA",
 "kid": "did: ..."
}
```
•
```
["024445",
"Alice", [[....]]]
```
•
```
{
 "bcn": "btc:testnet",
 "tid": a1075d..
 "pth": [0,afde4..,1, . ]
}
```

Fig.8: *VC written in a JWT-style format.*

'.' signs separating them. The payload part is used to store $\boldsymbol{v}_D$. The first and the last parts are JSON objects as described in Fig. 8. Due to the use of Bitcoin as the main integrity proof mechanism, the 'alg' is assigned the value 'ECDSA', the digital signature method used by Bitcoin. The proof part contains information about the blockchain network (e.g., "bcn":"btc:testnet"), the data transaction ID (e.g., "tid": a1075d..), and the proof path data (e.g., "pth": [0,afde4..]). The last key-value pair contains an array of hash values used along with $H(\boldsymbol{v}_D)$ to calculate the Merkle root hash. Each hash element is preceded by an integer 0 or 1 to denote the next hash's position (left or right) on its branch in a binary Merkle tree.

Each of the three parts is separately encoded using a combination of CBOR and Base64URL encoding methods. The results are concatenated to form a complete VC fingerprint.

3.8 On-chain VC Verification

Verification of an on-chain VC follows the algorithm in Fig. 9. From the figure, the first step is slightly different for the two issurance modes (FO and PO). The information obtained from C_f, namely $\boldsymbol{v}_D$ and TxID of a PD transaction, is first used to verify that $\boldsymbol{v}_D$ is stored in the specified transaction. Verification is straightforward for the FO mode. For the PO mode, $\boldsymbol{v}_D$ is first hashed and then the resulting hash value is used, along with the Merkle tree proof path, to calculate the Merkle root hash by the function **calRootHash**() (see [38] for the underlying algorithm). The root hash is used in place of $\boldsymbol{v}_D$ to verify that $\boldsymbol{v}_D$ exists in the specified transaction. Next, the transaction input is used to follow the VC transaction chain in a backward direction until the PI transaction is found (**bcGetInit**(), see Fig. 10 for the underlying algorithm). From there, $\boldsymbol{v}_I$ and the PS transaction ID can be retrieved. In some applications where the VC specification and template of the verifying document are known in advance, there may be no need to construct a complete VC document. In this case, a verification is performed by checking whether the PS transaction contains a valid protocol specification. To render a complete VC document, if required, a VC template is retrieved from the protocol specification as indicated in the PS transaction. It is then used by **renderDoc**() to render a VC document using the algorithm described in Fig. 6. If the document contains a link to external data, JSON-LD framing is then performed to create a self-contained VC.

4. IMPLEMENTATION AND EXPERIMENTS

The method was implemented and tested using a full Bitcoin node, connected to a Bitcoin Testnet with the transaction indexing turned on. The issuing and verifying system was written in Python, making an RPC connection with the Bitcoin daemon (bitcoind). All experiments were conducted on a computer with a 2.20 GHz CPU and 16GB RAM, running Windows 10 64-bit operating system. To control irrelevant network condition factors, the same machine was used to run all systems shown in Fig. 11.

From the figure, the learning record system contained a database of learning achievements used to issue credentials during the experiments. The BTCR DID system was a minimum implementation of the BTCR standard [39] for a Bitcoin-based DID system. It was used to issue and verify DID identifiers and the corresponding keys for credential issuers and recipients. The decentralized IPFS system was used as a VC template repository. The on-chain VC system was a decentralized application that provided on-chain VC issuing and verification services. In addition, to remove a network variation factor from consideration, the linked schemas used in the credentials were cached and stored locally.

ALGORITHM 3: On-Chain VC Verification Algorithm

```
Input: Cf //a credential fingerprint
Output: verified, Cr //result, and a rendered credential
Initialization:verified ← FALSE, Cr ← {}
[head, payload, proof] ← decode(Cf) //decomposed and decode
if (head is not a correct on-chain vc header) return
tr_type = 'PD' // to get a protocol data transaction
[isValid, Dd] ← bcRead(proof.tid, tr_type) //get trans. data
if (!isValid) return
if (isEmpty(payload)) then // 'FO' mode
    v_d ← Dd
else // 'PO' mode
    v_d ← payload
    h_vd ← H(payload) // calculate a payload hash
    h_mr ← calRootHash (h_vd, proof.pth) //get a root hash
    if (h_mr != Dd) return
end
//next, get the init trans. data and the associated public key
[isValid, Di, Xi, k_pub] ← bcGetInit(proof.tid)
if (!isValid) return
tid_s ← Di  //get the TxID of the protocol spec. trans.
v_i ← Xi  //get v_i from X_I
tr_type = 'PS' // to get a protocol spec transaction
[isValid, Ds] ← bcRead(tid_s, tr_type)
if ((!isValid) return
iss_did ← get issuer's did from Ds, v_i/v_d
validIss ← chkIssuer(iss_did, k_pub) //check issuer validity
    with the DID system
v_spec ← get a vc spec using information in Ds
if ((!validIss) return
Cr ← renderDoc(vc_spec, v_i, v_d)  //render a complete VC
```

Fig.9: *On-chain VC verification algorithm.*

Two sets of learning credentials were considered. The first credential set used in this study consisted of certificates issued for winners and participants of an annual contest on computational thinking or CT (see Fig. 12 for its VC template). The other credential set contained typical university degree certificates (see Fig. 2 for the corresponding VC template). The first set contained credentials issued for the final round of the competition, which was participated by 86 secondary and high school students in Thailand. The credential set was further divided into three subsets. The first subset contained credentials issued to the winners and runners-up of the competition. The second set contained the 'Honorable Mention' credentials. The third subset contained credentials issued for the rest as a participation certificate. For credentials in the first subset, the FO mode was used to issue an on-chain VC to the Bitcoin Testnet. The other two subsets adopted the PO mode for VC issuance. Note that all issuances were made for experimental purposes. No actual or real student private information was disclosed in any of the two modes. For the FO mode, due to a limited payload size of the OP_RETURN output in the Bitcoin network, six credentials were individually issued in each PD transaction (2 winners and 4 runners-up). For the honorable mentioned credentials in the second subset, by using the PO mode, a single PD transaction was used to publish an aggregated set of 10 credentials. For the participation credentials in the third subset, all 70 credentials were aggregated and published in a single PD transaction with the PO mode. In the second data set, degree credentials were published using the PO mode. A total of 500 credentials corresponding to 5 degree programs were published across 10 transactions to represent a user case with a long transaction chain (see Table 7 for more detail). For all credentials issued by the proposed methods (FO and PO), the issuer ID (BTCR did) was the only element of $\boldsymbol{v}_I$., due to the limited capacity of the BTC transaction data.

The proposed method was compared with two

ALGORITHM 4: Backward Traversal (bcGetInit())

```
Input: tid //txid of the credential's PD transaction
Output: isValid, Di, Xi, k_pub
Initialization: isValid ← FALSE, Di ← NULL, Xi ← NULL
do
    tid ← getPrevTxId(tid) //get the previous txid
    [isValid, Dd] ← bcRead(tid, 'PD')
while isValid
[isValid, Di, Xi, k_pub] ← bcRead(tid,'PI') // get PI trans.
```

Fig.10: *Backward transaction traversal algorithm.*

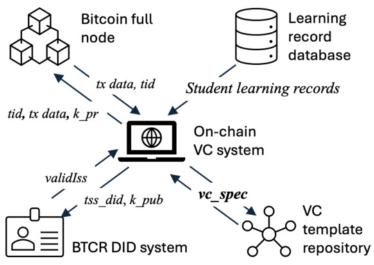


Fig.11: *System setup for the experiments.*

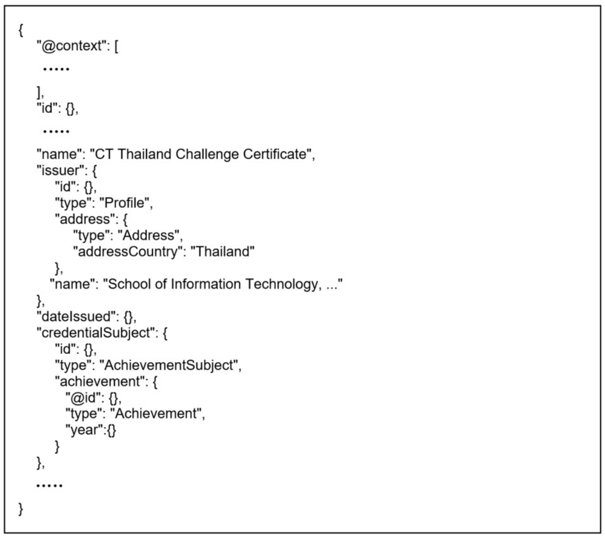
other blockchain-based VC methods, both employing the Merkle proof. The first of the two methods, referred to as 'BT-1', was equivalent to [13]. The only difference from [13] was in the proof approach, where a format similar to that of the proposed method was applied instead of the integrity proof. This compact format allowed a fair comparison with the proposed method. As for the 'BT-1' method, the second method, 'BT-2', stored the credential proof using the same JWT-style format. Unlike the 'BT-1' method, where the payload part contained a complete VC document, the value array of credential attributes was stored in the payload part instead. The proof part contained the same data as for the proposed method, with the addition of the VC template location (the IPFS CID in this case). Thus, to issue or verify a credential using the 'BT-2' method, a full VC document was first rendered from the value array (in the payload) and the linked VC template. Subsequent issuance/verification steps were the same as those of the 'BT-1' method. For all methods, each of the three parts of the JWT-style credential was first encoded using CBOR, followed by Base64URL encoding. The resulting encoded data was embedded in a QR code using the M-level ECC error correction. The required physical QR code size was obtained by assuming a mobile phone camera was used as a scanner. Based on the recommendation in [40], the suggested QR code size was calculated as in [33] for two scanning distances, 5" and 10", respectively.

Performance evaluation was conducted in the experiments to measure space and time complexities. Issuance and verification of all data sets were repeated 10 times to obtain the averaged results reported here. For each method, the size of the credential fingerprint $\boldsymbol{C_f}$ and the QR code size are shown in Tables 8-10. The VC issuing and verification times are shown in Table 11 and Table 12, respectively (see Table 7 for the number of credentials in each data set). Results for some credential data sets were omitted because they were similar to those of the representative data sets shown in the tables. From Table 8, the BT-1 credential size was mainly contributed by the payload size. A QR code version higher than 30 was required to embed the BT-1 credential data. The required QR code size for a 10" scanning distance (Size ♯1) was more than 5" in all cases, while the required size for a 5" scanning distance (Size ♯2) was around 3".

From Table 9, a substantial reduction in a credential size was obtained for the BT-2 method, where the proposed VC decomposition approach was applied. The required QR code version ranged from 17 to 21, and the QR code size of the 5" scanning distance was reduced to around 2". The result in Table 10 shows that the proposed method offered a further reduction in the fingerprint size. In particular, the FO-mode method reduced the required QR code size to less than 1". The PO-mode method delivered around a 10% reduction in credential fingerprint sizes as compared with those achieved by the BT-2 method. The size reduction was because, for the PO-mode method, part of the credential's key values and the VC template storage location were already stored on-chain in the PI and PS transactions. A small-sized QR code is desirable for good visual design and is required when limited space is available on a credential paper.

Execution times for the payload generation (1), BTC transaction issuance (2), and QR code generation (4) are shown in Table 11. Apart from the transaction issuance times, the other results are the average times per credential. Note that each transaction issuing time was measured up to the point where the transaction was successfully signed and available for blockchain inclusion (i.e., time due to a consensus mechanism not included). From the table, the transaction issuing time was comparable across different data sets and methods, as expected. The payload generation times for the BT-1 and BT-2 are higher than the corresponding ones in the proposed method. This is due to the need to perform the JSON-LD RDF canonicalization in BT-1 and BT-2. In addition, a JSON-LD framing step was also required by BT-2 to resolve any linkage to external data. This is the reason for the longer payload generation time in the BT-2 method. In each data set, the QR code generation time for the BT-1 method is the highest among all methods, while that of the FO-mode method is the lowest. This is because the QR code generation time depends on the size of the data to be encoded. Regarding the combined issuing and QR code generation time, the proposed PO-mode method yielded more than 60% and 30% performance improvements over the BT-1 and BT-2 methods, respectively.

Table 12 shows the computational times at various credential verification steps. From the table, 'tx order' refers to, for each data set, the order of each PD transaction issuance. It contained times to (1) decode C_f, (2) get data from all transactions required for verification, (3) render and verify VC, and (4) decode a QR code. From the table, verification time by the BT-1 method was the fastest when measured using the total (time) ♯1 (excluding (4)). While the decoding time (1) took longer than other methods due to a larger size of C_f, the 'render & verify' time (3) was faster for the BT-1 method because the VC rendering step was not required. Most of the rendering and verifying time (3) in the proposed method was spent retrieving a VC template from an IPFS node. The time spent to render a VC was only ∼1 ms. By using this performance indicator, the verification time of the BT-2 method was also faster than that of the proposed method. The slower time to get transaction data (2) in the proposed method was due to the required backward transaction-traversal step. Fig. 13 shows the transaction-traversal times at various PD-transaction chain lengths for both credential data types. From the figure, the traversal time was linearly increased with the length of the OCVC PD-transaction chain. At the chain length of 6, the traversal time accounted for 40% of the total time to get transaction data (2). When taking into consideration the QR code decoding time, the overall verification time is shown in Table 12 (Total ♯2). With this indicator, the proposed method achieved the best result, followed respectively by the BT-2 and BT-1 methods. The BT-1 method required a much longer time to decode a QR code because of its larger embedded data.



```
{
   "@context": [
    .....
   ],
   "id": {},
    .....
   "name": "CT Thailand Challenge Certificate",
   "issuer": {
      "id": {},
      "type": "Profile",
      "address": {
         "type": "Address",
         "addressCountry": "Thailand"
      },
     "name": "School of Information Technology, ..."
   },
   "dateIssued": {},
   "credentialSubject": {
      "id": {},
      "type": "AchievementSubject",
      "achievement": {
        "@id": {},
        "type": "Achievement",
        "year":{}
      }
   },
   .....
}
```

Fig.12: *VC template of the CT certificate.*

Table 7: *Detail of credential data sets used in the experiment.*

Credential type	Dataset name	Num. creds.	Num. cred. per trans.	Num. PD trans.	Num. proof hashes
Degree	DE75	75	75	1	7
	DE300	300	60	5	6
	DE100	100	50	2	6
	DE20	20	20	1	5
	DE5	5	5	1	3
CT	CT6	6	1	6	-
	CT10	10	10	1	4
	CT70	70	70	1	7

5. CONCLUDING REMARKS

The on-chain verifiable credential issuance and verification method has been described in this paper. The method was based on decomposing a VC into a template-value pair. Two credential issuance modes have been proposed. The FO-mode method allows a credential to be issued entirely on-chain. The PO-mode method offers a more economical solution for

Table 8: *The size of Cf and the QR code's physical size using the BT-1 method.*

Data set	Total size (byte)	Payload size (byte)	Proof size (byte)	QR code size #1 (in)	QR code size #2 (in)	QR code Ver.
DE5	1446	1202	208	5.8	2.9	32
CT10	1718	1426	256	6.28	3.14	35
DE300	1605	1201	368	5.96	2.98	33
DE75	1643	1205	402	6.12	3.06	34

Table 9: *The size of Cf and the QR code's physical size using the BT-2 method.*

Data set	Total size (byte)	Payload size (byte)	Proof size (byte)	QR code size #1 (in)	QR code size #2 (in)	QR code Ver.
DE5	460	147	277	3.4	1.7	17
CT10	485	124	325	3.56	1.78	18
DE300	619	146	437	3.72	1.86	19
DE75	654	147	471	4.04	2.02	21

Table 10: *The size of Cf and the QR code's physical size using the proposed method.*

Data set	Total size (byte)	Payload size (byte)	Proof size (byte)	QR code size #1 (in)	QR code size #2 (in)	QR code Ver.
CT6	112	0	76	1.8	0.9	7
DE5	350	106	208	3.08	1.54	15
CT10	375	83	256	3.08	1.54	15
DE300	508	104	368	3.56	1.78	18
DE75	544	106	402	3.56	1.78	18

Table 11: *Times to issue a credential (in ms).*

Method	Data set	Payload (1)	Issue Tx (2)	(1) + (2)/N_c (3)	QR (4)	Total (3+4)
FO	CT6	0.07	30.58	30.65	13.18	43.83
PO	DE5	0.02	31.80	6.38	35.56	41.94
	CT10	0.01	31.80	3.19	36.43	39.62
	DE300	0.01	30.47	0.52	51.48	52.00
	DE75	0.01	30.48	0.42	53.27	53.69
BT-1	DE5	5.09	30.18	11.13	142.56	153.69
	CT10	5.15	32.97	8.45	167.32	175.77
	DE300	4.93	32.40	5.47	154.40	159.87
	DE75	4.97	31.15	5.39	161.47	166.86
BT-2	DE5	13.85	33.43	20.54	47.29	67.82
	CT10	13.72	33.77	17.10	49.89	66.99
	DE300	13.65	31.25	14.17	61.24	75.41
	DE75	13.59	34.14	14.05	67.06	81.11

Table 12: Times to verify a credential (in ms).

Method	Data set (name – tx. order)	Decode C_f (1)	Get trans. data (2)	Render & verify VC (3)	Total #1 (1+2+3)	Decode QR code (4)	Total #2 (1+2+3 + 4)
FO	CT6-1	0.29	18.25	51.87	70.41	8.46	78.87
PO	CTI70-8	0.99	34.70	45.84	81.52	38.48	120.00
	DE300-1	0.98	19.68	48.96	69.62	40.37	109.99
	DE20-6	0.87	29.59	50.45	80.91	30.84	111.74
	DE5-10	0.67	33.31	50.00	83.99	27.07	111.06
BT-1	CT6	2.87	16.83	9.57	29.27	97.31	126.58
	CT70	3.49	19.83	9.82	33.14	207.95	241.09
	DE300	3.03	19.61	9.59	32.23	181.24	213.47
	DE20	2.82	19.27	9.41	31.49	193.26	224.75
	DE5	2.62	15.03	8.77	26.41	206.19	232.60
BT-2	CT6	0.65	14.35	49.64	64.64	20.04	84.67
	CT70	1.21	15.58	48.40	65.19	52.26	117.45
	DE300	1.22	15.02	48.35	64.59	50.36	114.95
	DE20	1.12	15.53	55.74	72.39	43.15	115.54
	DE5	0.93	17.38	59.86	78.17	33.34	111.50

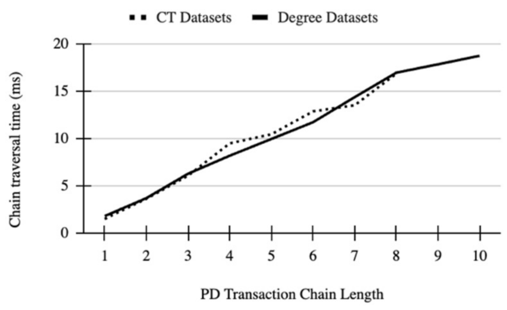


Fig.13: *Times required for the PD-transaction chain traversal at different chain lengths.*

issuing a batch of credentials. The proposed method allows the use of a concise VC fingerprint in place of a full VC document. It is thus possible to embed a VC in a paper-based credential using a small-sized QR code. The proposed method offered faster overall VC issuance and verification performance. In addition to a BTC wallet, issuing a credential using the proposed method requires a function to transform student learning records into a value array and a VC fingerprint format. Credential verification is more complicated and requires access to a Bitcoin full node (through a dedicated Bitcoin full node or an API call to a blockchain explorer service). Future works include the uses of the proposed method to store DID documents and VC-based revocation lists on-chain, as well as extending the current work to other blockchain systems. For account-based systems such as Ethereum, however, a combination of EOA (Externally Owned Accounts) and SCA (Smart Contract Accounts) is required in place of the transaction chain used in the UTXO-based BTC system.

ACKNOWLEDGEMENT

The work of the second author was financially supported by King Mongkut's Institute of Technology Ladkrabang through grant number KREF126007.

AUTHOR CONTRIBUTIONS

Conceptualization, N.C. and A.M.; methodology, N.C.; software, A.M.; validation, N.C., A.M., and C.S.; formal analysis, N.C.; writing—original draft preparation, N.C.; writing—review and editing, N.C., and C.S.; funding acquisition, N.C. All authors have read and agreed to the published version of the manuscript.

References

[1] J.J. Carmichael and S.E. Eaton, "Fake Degrees and Fraudulent Credentials in Higher Education: Conclusions and Future Directions," *In: Eaton, S.E., Carmichael, J.J., Pethrick, H. (eds) Fake Degrees and Fraudulent Credentials in Higher Education. Ethics and Integrity in Educational Contexts*, vol. 5, Springer, Cham, 2023.

[2] G. Grolleau, T. Lakhal and N. Mzoughi, "An Introduction to the Economics of Fake Degrees," *Journal of Economic Issues*, vol. 42, no. 3, pp. 673–693, 2008.

[3] R. McGreal and D. Olcott, "A strategic reset: micro-credentials for higher education leaders," *Smart Learning Environments*, vol. 9, no. 9, 2022.

[4] K.K. Miller, T.J. de St Jorre, J.M. West and E.D. Johnson, "The potential of digital credentials to engage students with capabilities of importance to scholars and citizens," *Active Learning in Higher Education*, vol. 21, no. 1, pp. 11-22, 2020.

[5] S. Sadiq and S. Zamir, "Effectiveness of modular approach in teaching at university level," *Journal of Education and Practice*, vol. 5, no. 17, pp. 103-109, 2014.

[6] S. N. Braxton, S. Carbonaro and N. Jankowski, "Comprehensive Learner Record as a Vehicle for Assessment and Learning Transparency in a Skills Economy," in *Y. Huang (Ed.), Handbook of Research on Credential Innovations for Inclusive Pathways to Professions*, IGI Global, pp. 214-233, 2022.

[7] B. Chakroun and J. Keevy, "Digital credentialing: implications for the recognition of learning across borders," *Meeting Report*, UNESCO, 2018, [Online]. Available: `https://doi.org/10.54675/SABO8911` (accessed 15 Feb. 2024).

[8] A. Perisic, I. Perisic, M. Lazic and B. Perisic, "The foundation for future education, teaching, training, learning, and performing infrastructure - The open interoperability conceptual framework approach," *Heliyon*, vol. 9, no. 6, 2023.

[9] 1EDTECH, "Open Badges Specification Candidate Final Public Spec Version 3.0," 2023, [Online]. Available: `https://1edtech.github.io/openbadges-specification/ob\_v3p0.html` (accessed May. 2023).

[10] M. Sporny, D. Longley and D. Chadwick, "Verifiable Credentials Data Model v2.0 W3C Working Draft," 2023, [Online]. Available: `https://www.w3.org/TR/2023/WD-vc-data-model-2.0-20230507/` (accessed 15 Feb. 2024).

[11] B. Rodrigues, M. Franco, E. Scheid, S. Kanhere and B. Stiller, "A technology-driven overview on blockchain-based academic certificate handling," *Blockchain Technology Applications in Education*, IGI Global, pp. 197–223, 2020.

[12] A. Rustemi, F. Dalipi, V. Atanasovski and A. Risteski, "A Systematic Literature Review on Blockchain-Based Systems for Academic Certificate Verification," *IEEE Access*, vol. 11, pp. 64679-64696, 2023.

[13] "Blockcerts blockchain credentials," [Online]. Available: `https://www.blockcerts.org/` (accessed 15 Feb. 2024).

[14] K. Kuvshinov, I. Nikiforov, J. Mostovoy, D. Mukhutdinov, K. Andreev and V. Podtelkin, "Disciplina: Blockchain for Education," [Online]. Available: `https://www.disciplina.io/yellowpaper.pdf` (accessed 15 Feb. 2024).

[15] R. Mukta, J. Martens, H. -y. Paik, Q. Lu and S. S. Kanhere, "Blockchain-Based Verifiable Credential Sharing with Selective Disclosure," *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Guangzhou, China, pp. 959-966, 2020.

[16] M. Turkanović, M. Hölbl, K. Košič, M. Heričko and A. Kamišalić, "EduCTX: A Blockchain-Based Higher Education Credit Platform," *IEEE Access*, vol. 6, pp. 5112-5127, 2018.

[17] C. Brunner, F. Knirsch and D. Engel, "SPROOF: A Platform for Issuing and Verifying Documents in a Public Blockchain," in *Proc. 5th Int. Conf. Inf. Syst. Secur. Privacy*, pp. 15–25, 2019.

[18] C.S. Hsu, S.F. Tu and P.C. Chiu, "Design of an e-diploma system based on consortium blockchain and facial recognition," *Education and Information Technologies*, vol. 27, no. 4, pp. 5495–5519, Jan. 2022.

[19] Y. Mezquita, B. Podgorelec, A. Gil-González and J. Corchado, "Blockchain-Based Supply Chain Systems, Interoperability Model in a Pharmaceutical Case Study," *Sensors*, vol. 23, no. 4, pp. 1-19, 2023.

[20] M. Gottlieb and G. Bacharach, "Cleaning Up: Interplay Between European Standards and Verifiable Credentials for Higher Education Institutions," *Lecture Notes in Networks and Systems*, vol. 634, pp. 777–788, 2023.

[21] D. Longley, M. Sporny, "Data Integrity EdDSA Cryptosuites v1.0," *W3.org*, 2023, [Online]. Available: `https://www.w3.org/TR/vc-di-eddsa/` (accessed Feb. 15, 2024).

[22] D. Longley, M. Sporny, "Data Integrity ECDSA Cryptosuites v1.0," *W3.org*, 2024, [Online]. Available: `https://www.w3.org/TR/vc-di-ecdsa/` (accessed Feb. 15, 2024).

[23] "BBS Cryptosuite v2023," *W3.org*, 2024, [Online]. Available: `https://www.w3.org/TR/vc-di-bbs/` (accessed Feb. 15, 2024).

[24] "Merkle Proof Signature Suite 2019," *W3C Community Group*, 2021, [Online]. Available: `https://w3c-ccg.github.io/lds-merkle-proof-2019/` (accessed Feb. 15, 2024).

[25] "Merkle Disclosure Proof 2021," [Online]. Available: `https://w3c-ccg.github.io/Merkle-Disclosure-2021/` (accessed Feb. 15, 2024).

[26] "Getting started - OpenCerts," *Government Techno-logy Agency (Singapore)*, 2022, [Online]. Available: `https://docs.opencerts.io/docs` (accessed Feb. 15, 2024).

[27] "OpenAttestation: Document Endorsement and Verification Framework," *Government Technology Agency (Singapore)*, 2023, [Online]. Available: `https://www.openattestation.com/` (accessed Feb. 15, 2024).

[28] C. Adams, P. Cain, D. Pinkas and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)," *Network Working Group Ietf.org*, 2001, [Online]. Available: https://www.ietf.org/ rfc/rfc3161.txt (accessed 15 Feb. 2024).

[29] S. Santesson, N. Pope, "ESSCertIDv2 Update for RFC 3161," *Internet Engineering Task Force (IETF)*, 2010, [Online]. Available: `https://www.rfc-editor.org/rfc/rfc5816.html` (accessed 15 Feb. 2024).

[30] G. Michoulis, S. Petridou, K. Vergidis, G. Michoulis, George, "Verification of Academic Qualifications through Ethereum Blockchain: An Introduction to VerDe," *XIV Balkan Conference on Operational Research*, pp. 429-433, 2020.

[31] J. Tellew and T. Kuo, "CertificateChain: decentralized healthcare training certificate management system using blockchain and smart contracts," *JAMIA Open*, vol. 5, no. 1, April 2022.

[32] R.J. Maestre, J. Bermejo Higuera, N. Gámez Gómez *et al.*, "The application of blockchain algorithms to the management of education certificates," *Evol. Intel.*, vol. 16, pp. 1967–1984, 2023.

[33] A. Tariq, H. Binte Haq and S. T. Ali, "Cerberus: A Blockchain-Based Accreditation and

Degree Verification System," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 4, pp. 1503-1514, 2023.

[34] M. Bartoletti and L. Pompianu, "An Analysis of Bitcoin OP_RETURN Metadata," *In: Brenner, M., et al. Financial Cryptography and Data Security: Lecture Notes in Computer Science*, vol 10323, Springer, Cham, 2017.

[35] J. Mols and E. Vasilomanolakis, "Visualizing the Bitcoin's OP_RETURN operator," *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (Mobihoc '20)*, pp. 305–306, 2020.

[36] M. Bartoletti, B. Bellomy and L. Pompianu, "A Journey into Bitcoin Metadata," *Journal of Grid Computing*, vol. 17, pp. 3–22, 2019.

[37] P. Rodwald, "An Analysis of Data Hidden in Bitcoin Addresses," Theory and Engineering of Dependable Computer Systems and Networks: Advances in Intelligent Systems and Computing, vol 1389. Springer, Cham, 2021.

[38] J. Bukowski *et. al.*, "merkle-tools," [Online]. Available: `https://github.com/tierion/merkle-tools` (accessed May. 2024).

[39] W. Fdhila, N. Stifter, K. Kostal, C. Saglam and M. Sabadello, "Methods for Decentralized Identities: Evaluation and Insights," *Lecture Notes in Business Information Processing*, vol 428, Springer, 2021.

[40] "QR code, 2D barcode invented by Denso Wave," Denso Wave Incorporated, [Online]. Available: `http://www.qrcode.com` (accessed 15 Mar. 2024).

Nopporn Chotikakamthorn received a B.Eng. degree in electronic engineering from King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand in 1990, and a Ph.D. degree in Electrical and Electronic Engineering from Imperial College, London, in 1996. Since 2001, he has been an Associate Professor at the School of Information Technology, King Mongkut's Institute of Technology Ladkrabang. His research interest includes signal processing and its application, human-computer interaction, information technology acceptance, multimedia technology in education, and learning analytics.

Aye Mi San is currently a PhD candidate at the School of Information Technology, King Mongkut's Institute of Technology Ladkrabang, , Thailand. She received her Bachelor of Technology degree in Information Technology from Technological University Thanlyin, Myanmar, and her Master of Science degree in Information Technology from King Mongkut's Institute of Technology Ladkrabang, Thailand. Her research interests include cryptography, electronic payment systems (such as micropayment protocols and mobile payment systems), blockchain technology, and credential verification systems.

Chanboon Sathitwiriyawong received his BEng degree in Electrical Engineering from Prince of Songkla University, Thailand in 1986. He received his MSc in Data Telecommunications and Networks in 1993 and his PhD in Electronic and Electrical Engineering in 1996 from the University of Salford, United Kingdom. He is currently an Associate Professor at the School of Information Technology, King Mongkut's Institute of Technology Ladkrabang. His research areas of interest include network security, cryptography, software defined networking, and ad hoc and sensor networks.