sensors

MDPI

Article

A Challenge-Response Assisted Authorisation Scheme for Data Access in Permissioned Blockchains

Xiaoshuai Zhang *, Chao Liu, Kok Keong Chai and Stefan Poslad

School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, UK; c.liu@qmul.ac.uk (C.L.); michael.chai@qmul.ac.uk (K.K.C.); stefan.poslad@qmul.ac.uk (S.P.)

* Correspondence: xiaoshuai.zhang@qmul.ac.uk

Received: 15 June 2020; Accepted: 12 August 2020; Published: 19 August 2020

Abstract: Permissioned blockchains can be applied for sharing data among permitted users to authorise the data access requests in a permissioned blockchain. A consensus network constructed using pre-selected nodes should verify a data requester's credentials to determine if he or she have the correct permissions to access the queried data. However, current studies do not consider how to protect users' privacy for data authorisation if the pre-selected nodes become untrusted, e.g., the pre-selected nodes are manipulated by attackers. When a user's credentials are exposed to pre-selected nodes in the consensus network during authorisation, the untrusted (or even malicious) pre-selected nodes may collect a user's credentials and other private information without the user's right to know. Therefore, the private data exposed to the consensus network should be tightly restricted. In this paper, we propose a challenge-response based authorisation scheme for permissioned blockchain networks named Challenge-Response Assisted Access Authorisation (CRA^3) to protect users' credentials during authorisation. In CRA^3, the pre-selected nodes in the consensus network do not require users' credentials to authorise data access requests to prevent privacy leakage when these nodes are compromised or manipulated by attackers. Furthermore, the computational burden on the consensus network for authorisation is reduced because the major computing work of the authorisation is executed by the data requester and provider in CRA^3.

Keywords: privacy enhancement; permissioned blockchain; access control; decentralised network

1. Introduction

Permissioned blockchain networks stem from blockchain [1,2] as a decentralised network structure used for trading or sharing data among permitted users (nodes). Permissioned blockchain networks have been applied in numerous fields, including security services [3,4], Internet of Things (IoT) [5,6], and reputation systems [7,8]. In a permissioned blockchain network, there are numerous pre-selected nodes that constitute the consensus network to realise authorisation that tolerate Byzantine faults, avoiding a single point of failure and providing system scalability [9,10].

In current studies, pre-selected consensus nodes are normally assumed to be trusted in order to authorise the data access and to transport users' private data in a consensus network [11]. However, one important issue that has not been considered is that users' private data can be utilised by the pre-selected nodes in the consensus network during authorisation, since these nodes may be manipulated by the attacker to be untrusted (malicious). To be specific, in many current designs of authorisation for permissioned blockchains [10,12], the nodes in a consensus network require the use of many users' private information (e.g., credentials and personal information) to authorise their data access requests. When some nodes are compromised by an attacker, the users' private information can be disclosed to an attacker without any barriers.

Furthermore, if pre-selected nodes in the consensus network are malicious, such nodes can exploit a user's private data to analyse his or her behaviour without such a user's right to know that this violates their privacy. For example, when a permissioned blockchain network is applied to smart power grids or smart charging, the behaviour of the registered user can be analysed based upon the uploading time and length of his or her electric bills if the nodes are malicious or compromised in the consensus network [13,14]. Similarly, in an eHealth scenario, a patient's private information could be leaked with respect to the above situation.

However, research regarding privacy protection for authorisation in a permissioned blockchain is currently in its infancy. Many studies does not consider how to protect users' private information during authorising data access in permissioned blockchains, as they regard the consensus network (pre-selected nodes) as a fully trusted part. Even though some recent studies (e.g., [15]) start to raise concerns protecting users' private information in the authorisation of a permissioned blockchain, such methods may leak users' private information to particular nodes in the consensus network of the permissioned blockchain. Therefore, in a permissioned blockchain, the protection of users' private information is still an open problem to be addressed for the authorisation of data access by the untrusted consensus network.

In this paper, our major novelty lies in considering privacy protection for the requisite authorisation so as to avoid privacy leakage when some nodes of the consensus network are untrusted (malicious or compromised) in a permissioned blockchain. A Challenge-Response Assisted Access Authorisation (CRA^3) scheme based on the challenge-response mechanism is proposed to protect users' private information during the authorisation for data access. The contributions of this paper are summarised as follows.

- Unlike many mainstream designs [10,12,16] that cannot protect users' private information in the authorisation, CRA^3 can realise authorisation without revealing the access permissions and other private information of users to nodes in the consensus network and keep users anonymous in the permissioned blockchain because we consider the consensus network as untrusted.
- A theoretical security model and proof are illustrated formally to show that CRA^3 can achieve the confidentiality of indistinguishability under chosen-ciphertext attacks (IND-CCA) [17] to avoid privacy leakage during the authorised data access in a permissioned blockchain.
- Compared with [12,16], the communication overhead decreases because the size of the information needed for the authorisation is much smaller in our designed algorithms.
- Most of the computing work for authorising the data access request is executed by the data requester and provider to decrease the transaction cost and the burden on the consensus network.

The rest of this paper is organised as follows. The related work is presented in Section 2 and the preliminaries are introduced in Section 3 to help understand our concrete algorithms for our CRA^3 scheme. Then, the problem we try to solve and the algorithm definitions of our CRA^3 scheme are described in Section 4. After that, our proposed CRA^3 scheme is demonstrated in Section 5, followed by a theoretical security analysis in Section 6. Furthermore, the experiments and the comparison of the results are demonstrated in Section 7 to analyse the performance of CRA^3 in terms of the computational time consumption, the communication overhead, and the transaction fee for authorisation. Finally, we conclude our work in Section 8.

2. Related Work

The authors in [12] presented a personal data transmission scheme via blockchain consensus networks. In their scheme, the stored personal data are transmitted by a blockchain with constructed access policies to authorise data access via a consensus network. As they regard the consensus network as a trusted network, all of the users' private data (including access permissions) are exposed to the consensus network in their scheme. If some nodes in the consensus network are compromised, the transmitted personal data can be revealed to attackers. The Healthcare Data Gateways scheme

(HDG) [16] has a similar issue in the gateway (consensus part) design; users' access tokens can be accessed via the gateway without any secure precaution.

Reference [18] discusses several applications of blockchain in IoT scenarios have been discussed including wireless software updates, smart charging, and car sharing services. However, there is no consideration for privacy protection in the consensus network when blockchains are applied to the above scenarios. A transaction framework for permissioned blockchains with a group policy was proposed, but the group policy was determined only by computing power without any data access control for users [19].

In the smart grid area, the authors in [20] proposed how to build permissioned blockchain networks for bill collection and power load adjustment. There are two reasons for utilising permissioned blockchain networks: first, the decentralisation feature of blockchain can achieve a fail-safe state to avoid a single point of failure; second, the structure of permissioned networks can block unauthenticated and unauthorised access. However, when some nodes are manipulated by the attackers in a consensus network, the solution to avoid privacy leakage is not discussed. More recently, an energy trading system supported by a permissioned blockchain was proposed by Gai et al. [15], where message validation was considered for a scenario of smart grids. The authors utilised group signature [21] to construct an identity validation algorithm to validate the edge nodes (and their messages) during the activities defined by the smart contracts. However, for the process of identity validation, the leader of the permissioned nodes knows the real identity of each node in the system setup phase. Meanwhile, this trading system can only ensure data integrity (signature) of messages but cannot provide confidentiality protection because messages between two edge nodes are plain to all permissioned nodes without any encryption. Furthermore, the issue of key revocation [22,23] is quite complicated and is not considered in such a large-scale decentralised system [24].

3. Preliminaries

3.1. Permissioned Blockchain Networks

The network structure of a permissioned blockchain shown in Figure 1 is the same as that of a public blockchain. All the nodes are anonymous in the network. However, the data (ledger) in each node are private, which means the data access between the two nodes should be authorised to ensure one node has the permissions to access the data in another node.

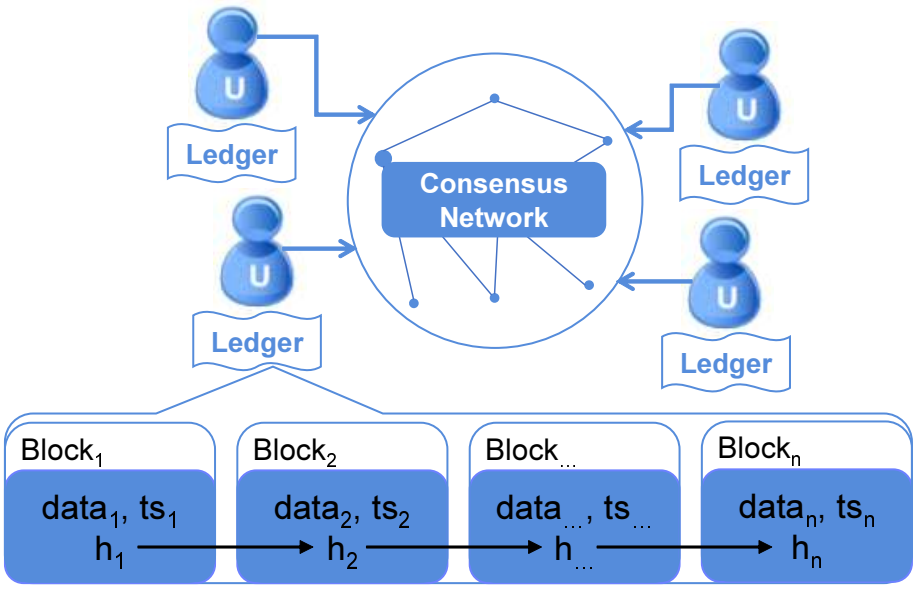

Figure 1. The structure of the permissioned blockchain network.

Meanwhile, the private ledger of each node is a blockchain structure. Each block contains data, a cryptographic hash value (h) and a timestamp (ts) in the blockchain [1]. The hash value for establishing the link between two blocks is generated by the following rules:

$$h_i = \begin{cases} Hash(data_1||ts_1) & , i = 1 \\ Hash(h_{i-1}||data_i||ts_i), & i = 2, ..., n \end{cases}$$

3.2. Lagrange Interpolating Polynomial

Let q be a prime and

$$f(x) = a_0 + a_1 x + ... + a_{t-1} x^{t-1} \ (mod\ q)$$

with a polynomial of degree t, where $a_0, a_1, ...a_t \in \mathbb{Z}_q$ are coefficients. Given any t points $\{(x_1, y_1), ..., (x_t, y_t)\}$ on $f(x)$, the coefficient $a_0 = f(0)$ can be computed with Lagrange interpolating polynomial

$$f(0) = \sum_{i \in A} \Delta_i y_i \ (mod\ q),$$

where $\Delta_i = \prod_{j \in A/\{i\}} \frac{x_j}{x_j - x_i} \ (mod\ q)$ are the Lagrange interpolation coefficients and $A = \{1, 2, ..., t\}$. Note that $f(x)$ can be reconstructed with any t points based on polynomial theory; however, $f(x)$ cannot be reconstructed ($f(0)$ cannot be computed either) with any points fewer than t [25].

4. Problem and Definitions

In this section, we first describe our scheme model and the targeted security problem we try to address. Then, seven algorithms are defined to construct our proposed CRA^3 scheme in the second part, which is followed by the security definitions including the correctness and confidentiality of our CRA^3 as the last part.

4.1. Problem Statement

The proposed scheme model is demonstrated in Figure 2 with three entities: two users (nodes) and the consensus network (constructed by pre-selected nodes in the permissioned blockchain network), denoted by U_A (data requester), U_B (data provider) and CN_{pm}, respectively. Since all the nodes in the network are anonymous (unknown identities), one node cannot trust another node in the permissioned blockchain network. Therefore, the consensus network CN_{pm} is needed as a mediator to finish the validations in the challenge-response phase. If U_A has the correct permissions to access his/her requested data, U_B establishes a secret channel with U_A to transport the encrypted data after the authorisation.

Since the nodes in CN_{pm} are assumed to be untrusted, one untrusted node can output the incorrect result of the authorisation and collect the user's private data from the communication in the challenge-response phase. If a node outputs the incorrect authorisation result, the consensus network can punish this dishonest node according to the applied consensus mechanism (e.g., Byzantine fault tolerance). On the other hand, if a malicious node collects the user's data from the challenge-response communication and then attempts to reveal the user's credentials (or other private information), our proposed scheme should prevent the private data leakage. Hence, the purpose of our proposed scheme is to authorise data access without involving the users' credentials ($U_{id} = \{U_1, U_2, ..., U_n\}$) whilst ensuring the transported data is confidential, i.e., $\forall M \in \{0,1\}^*, C = f(M)$, and any probabilistic polynomial-time algorithm $\mathcal{A}$ computes M or U_{id} with its advantage $Adv_{\mathcal{A}}^{CN_{pri}} = Pr[c = M \vee c \subseteq U_{id} | c = \mathcal{A}(C, D_{Ch}, D_{Re})] < \varepsilon$, where M is plaintext data, C is encrypted data that transmitted between U_A and U_B, D_{Ch} and D_{Re} denote the data used in the processes *Challenge* and *Response*, respectively, and ε represents a negligible probability.

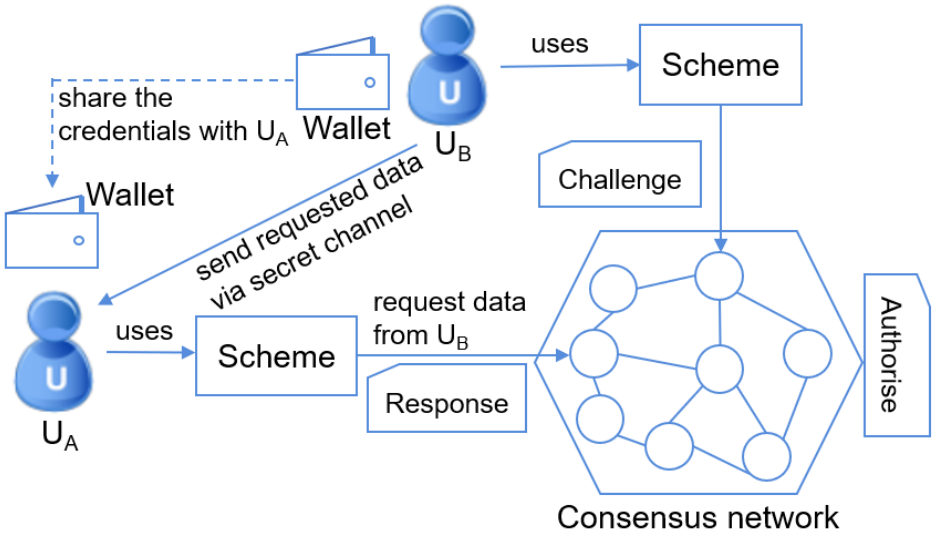

Figure 2. The system model of our proposed Challenge-Response Assisted Access Authorisation (CRA^3) scheme.

4.2. Scheme Definitions

In this section, we introduce the credentials used in the authorisation and define all of the seven algorithms that comprise our proposed CRA^3 scheme.

Credentials for authorisation: The credentials used for authorisation are the identity attributes. For instance, in the hospital scenario, if a doctor wants to request a patient's medical records, the identity attributes can be the patient's name, age, medical record number, social number, and so on. We assume that in reality, the patient has shared the identity attributes and a unique reference number to identify the needed identity attributes for the doctor's data request before inquiring about the data. We denote unique reference number (key) and the identity attributes (values) by Rn and the sequence $AT^v = \{AT_1^v, AT_2^v, ..., AT_n^v\}$, respectively.

- Setup (λ): This algorithm takes the security parameter λ and generates the public parameter pp.
- Request (pp): U_A uses the algorithm to send a data access request Q to U_B via CN_{pm}.
- Challenge (pp, Q): U_B constructs and sends the challenge Ch to U_A with the identity attributes sequence AT^v.
- Response (pp, Ch): U_A uses its own sequence AT'^v to calculate and send the response Re to the CN_{pm}.
- Authorise (Ch, Re): CN_{pm} validates the correctness of the response Re based upon the challenge Ch.
- Encrypt (pp, M, AT^v): U_B uses this algorithm to encrypt the requested data M then return the ciphertext C and a point P (for decryption use) to U_A.
- Decrypt (pp, C, P, AT'^v): U_A decrypts the encrypted data C to retrieve the requested data M with the given point P.

4.3. Security Definition

The definitions of correctness and the IND-CCA (indistinguishability under chosen-ciphertext attacks) security (confidentiality) for our CRA^3 scheme are illustrated as follows.

4.3.1. Correctness

For any $pp \leftarrow Setup(\lambda)$ and any plaintext $M \in \{0,1\}^*$, the CRA^3 scheme is correct if $Decrypt(pp, C, AT'^v) = M$ always holds, where $C = Encrypt(pp, M, AT^v)$.

4.3.2. Confidentiality (IND-CCA Security)

Formally, the adversary defined to prove the theoretical security of our proposed CRA^3 scheme is: *Type-IND adversary*.

Type-IND adversary: In the *Authorise* phase, the adversary cannot determine the message that the given challenge ciphertext is encrypted from, even though the sequence of the identity attributes AT^v is revealed to the adversary.

Game 1. *Let $\mathcal{A}_1$ be the given Type-IND adversary, and the index of the target data provider be t $(1 \leqslant t \leqslant n)$. The game played by the challenger $\mathcal{C}$ and the adversary $\mathcal{A}_1$ is described with the following five phases:*

- *Initialise:*

  $\mathcal{C}$ first generates the public parameter pp via running the algorithm $Setup$. Then, $\mathcal{C}$ generates n data providers (key-value pairs) $\{Rn^i, AT^{v_i}\}$ $(1 \leqslant i \leqslant n)$ and the target data provider is $\{Rn^t, AT^{v_t}\}$. The generated pp and all Rn^i $(1 \leqslant i \leqslant n)$ are given to the adversary $\mathcal{A}_1$.
- *Queries:*

  The following queries can be requested by $\mathcal{A}_1$ for polynomial times in the game:

1. *Identity attributes query* (i): $\mathcal{C}$ responds with the sequence of the random identity attributes rAT^{v_i};
2. *Encrypt query*(M, i): $\mathcal{C}$ outputs the ciphertext $C = Encrypt(pp, M, AT^{v_i})$ and the point P on the constructed polynomial $f(x)$ in the *Encrypt* phase;
3. *Decrypt query*(C, P, i): $\mathcal{C}$ decrypts C via running the algorithm *Decrypt*, then responds with the plain message.

- *Challenge:*

  $\mathcal{A}_1$ submits two equal-length messages M_0^* and M_1^*. $\mathcal{C}$ picks $\rho \in_R \{0, 1\}$, and then computes and returns the challenge ciphertext $C^* = Encrypt(pp, M_\rho^*, AT^{v_i})$.
- *Constraints:*

1. (M_0^*, t) and (M_1^*, t) are not allowed to appear in the above *Encrypt query*;
2. The target data provider's index t and the challenge ciphertext C^* are not allowed to appear in the above *Decrypt query*.

- *Guess:*

  $\mathcal{A}_1$ can win the game if its output $\rho' \in_R \{0, 1\}$ satisfies the condition $\rho = \rho'$.

  Now, the advantage of $\mathcal{A}_1$ could be defined as:

$$Adv_{\mathcal{A}_1}^{IND-CCA}(\lambda) = |Pr[\rho = \rho'] - \tfrac{1}{2}|.$$

Note that the probability analysis is presented after the *Guess* phase in the formal confidentiality proof of our CRA^3 scheme.

Definition 1 ((IND-CCA Security)**.** *The CRA^3 scheme is IND-CCA secure if the advantage $Adv_{\mathcal{A}_1}^{IND-CCA}(\lambda)$ of any probabilistic polynomial-time adversary $\mathcal{A}_1$ is negligible.*

5. Proposed Scheme CRA^3

In this section, we illustrate our proposed CRA^3 scheme (Challenge-Response Assisted Access Authorisation) with the seven algorithms defined in Section 4.2, including Setup, Request, Challenge, Response, Authorise, Encrypt, and Decrypt. In CRA^3, *AES* (Advanced Encryption Standard [26]) is used to encrypt the requested data and the Lagrange interpolating polynomial is utilised to construct challenge-response authorisation and protect the encrypting/decrypting key of *AES*.

- **Setup (λ):**

  This procedure outputs public parameters pp with the security parameter λ using the following steps.

  1. Generate a big prime q ($q > 2^{\lambda}$);
  2. Select one secure cryptographic hash function $H : \{0,1\}^{*} \rightarrow \{0,1\}^{\lambda}$;
  3. Select a symmetric encryption algorithm, e.g., AES (Advanced Encryption Standard);
  4. Output the public parameters $pp = (q, H, AES)$.

- **Request (pp):**

  The user U_A (data inquirer) prepares the query Q via the following steps.

  1. Decide on the data to be requested. Note that U_A should have the corresponding identity attributes (a sequence, AT'^{v}) and the unique reference number (R_n) that is shared by U_B. For illustrating the remaining parts of the proposed scheme, we assume the requested data is in one block B_{id};
  2. Prepare the unique reference number Rn then send the request $Q = (B_{id}, Rn)$ to U_B through CN_{pm}.

- **Challenge (pp, Q):** U_B generates the challenge Ch based upon the request Q from U_A via the following steps.

  1. Prepare the sequence of the identity attributes (values): $AT^{v} = \{AT_1^{v}, AT_2^{v}, ..., AT_n^{v}\}$ based upon the unique reference number $Rn \in Q$;
  2. Calculate the hash value of each element in the sequence AT^{v} to get the sequence $AH^{v} = \{H(AT_1^{v}), H(AT_2^{v}), ..., H(AT_n^{v})\}$;
  3. Construct a polynomial $f(x) = H(AT_1^{v}) + H(AT_2^{v})x + ... + H(AT_n^{v})x^{n-1}$ $(mod\ q)$, then pick n random points on the polynomial $f(x)$ as a set: $P = \{(x_i, y_i) | (x_i, y_i) \in f(x) \wedge i = 1...n\}$;
  4. Construct two sequences P_x and P_y of all the x_i and all the y_i in P: $P_x = \{x_i | x_i \in f(x) \wedge (x_i, f(x_i)) \in P \wedge i = 1...n\}$ and $P_y = \{y_i | y_i = f(x_i) \wedge x_i \in P_x \wedge i = 1...n\}$;
  5. Calculate the hash value of the sequence P_y: $PH_y = H(y_1, y_2, ..., y_n)$, $y_1, y_2, ..., y_n \in P_y$;
  6. Send the challenge P_x to U_A through CN_{pm}. Note that CN_{pm} should keep the $Ch = (PH_y)$ to execute the following *Authorise* phase.

- **Response (pp, Ch):**

  U_A generates the response Re to the challenge P_x from U_B via the following steps.

  1. Prepare the sequence of the identity attributes $AT'^{v} = \{AT_1'^{v}, AT_2'^{v}, ..., AT_n'^{v}\}$ (shared by U_B) based upon $R_n \in Q$;
  2. Construct a new polynomial $g(x) = H(AT_1'^{v}) + H(AT_2'^{v})x + ... + H(AT_n'^{v})x^{n-1}$ $(mod\ q)$;
  3. Take $P_x \in Ch$ to calculate the sequence $P_y' = \{y_i' | y_i' = g(x_i) \wedge x_i \in P_x \wedge i = 1...n\}$ and then hash the sequence P_y': $PH_y' = H(y_1', y_2', ..., y_n')$, $y_1', y_2', ..., y_n' \in P_y'$;
  4. Send the response $Re = (PH_y')$ to the consensus network CN_{pm}.

- **Authorise (Ch, Re):**

  The consensus network CN_{pm} validates the two hash values in Ch and Re. If $PH_y (\in Ch) = PH_y' (\in Re)$ holds (consensus check point), it means that the user U_A can be authorised to access the requested data B_{id} and the next phases are conducted; otherwise, the agent layer should deny the access request from U_A.

- **Encrypt (pp, Q, AT^v):**

  U_B encrypts the requested data via the following steps.

  1. Acquire the requested data M based upon $B_{id} \in Q$ from U_A and then calculate the hash value H_M of the data M: $H_M = H(M)$;
  2. Generate a secure key $k \in \mathbb{Z}_q$ for the symmetric encryption;
  3. Use AES to encrypt M with key k to get the ciphertext $C = AES_k(M, H_M)$. For decrypting $AES_k(M, H_M)$ to recover the plain data M, AES'_k is defined as the decryption process: $M = AES'_k(C = AES_k(M, H_M))$;
  4. Follow Section 3.2 to construct a polynomial $f^*(x)$ of degree n with k and AT^v: $f^*(x) = k + H(AT_1^v)x + H(AT_2^v)x^2 + ... + H(AT_n^v)x^n \ (mod\ q)$;
  5. Generate a random integer $x_p \in \mathbb{Z}_q$ and calculate a point $P(x_p, y_p = f^*(x_p))$;
  6. Return (C, P) to U_A through a secret channel.

- **Decrypt (pp, C, P, AT'^v):**

  U_A can decrypt the ciphertext C after passing the *Authorise* phase via the following steps.

  1. Use the sequence AT'^v organised in the former *Response* phase to construct a polynomial $g^*(x)$: $g^*(x) = a_0 + H(AT_1'^v)x + H(AT_2'^v)x^2 + ... + H(AT_n'^v)x^n \ (mod\ q)$. Note that a_0 is an unknown coefficient;
  2. Follow the Lagrange interpolation polynomial in the Section 3.2 to reconstruct the polynomial $g^*(x)$ fully, and then recover the key $k = g(0) = a_0 \in \mathbb{Z}_q$ for AES decryption with the point $P(x_p, y_p)$: $k = y_p - H(AT_1'^v)x_p - H(AT_2'^v)x_p^2 - ... - H(AT_n'^v)x_p^n \ (mod\ q)$;
  3. Decrypt C to retrieve the plaintext $(M, H_M) = AES'_k(C) = AES'_k(AES_k(M, H_M))$;
  4. If $H(M) = H_M$ holds, this algorithm outputs M; otherwise, it outputs $\perp$.

6. Theoretical Analysis of CRA^3

In this section, we first show the correctness of our proposed authorisation CRA^3 scheme and then prove the confidentiality of CRA^3. After that, the (data) integrity of CRA^3 is illustrated in the third subsection, which is followed by a comparison of the security features in different blockchain-related authorisation schemes, as given in the last subsection.

6.1. Correctness

In the *Authorise* phase, if the data requester has the correct sequence of the identity attributes AT'^v, the condition $AT^v = AT'^v$ holds,

$AT^v = AT'^v$
$\Leftrightarrow \{H(AT_1^v), H(AT_2^v), ..., H(AT_n^v)\} = \{H(AT_1'^v), H(AT_2'^v), ..., H(AT_n'^v)\}$
$\Leftrightarrow f(x) = g(x)$
$\Leftrightarrow P_y = P'_y \ (for\ the\ given\ P_x)$
$\Leftrightarrow PH_y = PH'_y$.

This means that the data requester can pass the *Authorise* phase if and only if this requester has the correct corresponding sequence of the identity attributes for the requested blocks.

To satisfy the condition in the correctness definition, the authorised data requester should retrieve the key $k \in \mathbb{Z}_q$ for AES decryption with the given point $P(x_p, y_p)$ on the polynomial $f(x)$ in the *Decrypt* phase. Meanwhile, P should present on the correct reconstructed polynomial as well. Since the condition $\{H(AT_1^v), H(AT_2^v), ..., H(AT_n^v)\} = \{H(AT_1'^v), H(AT_2'^v), ..., H(AT_n'^v)\}$ holds after the *Authorise* phase, the reconstructed polynomial $g(x)$ is the same as the original polynomial $f(x)$ except for the unknown first coefficient $a_0 = k$. Therefore, determining the secret key $g(0) = a_0 = k \in \mathbb{Z}_q$ for AES decryption requires only one point (shareholder) $P(x_p, y_p)$:

$k = a_0$
$= g(x_p) - AT_1'^v x_p - AT_2'^v x_p^2 - ... - AT_n'^v x_p^n$
$= f(x_p) - AT_1'^v x_p - AT_2'^v x_p^2 - ... - AT_n'^v x_p^n$
$= f(x_p) - AT_1^v x_p - AT_2^v x_p^2 - ... - AT_n^v x_p^n$
$= k \ (mod \ q)$.

Hence, the authorised data requester can reconstruct the polynomial $g(x)$ and restore the correct key k in the *Decrypt* phase to ensure $Decrypt(pp, C, AT'^v) = M$ holds, where $C = Encrypt(pp, M, AT^v)$.

6.2. Confidentiality (IND-CCA Security)

Theorem 1. *According to Definition 1 above, the proposed CRA^3 scheme is IND-CCA secure based on the Lagrange interpolating polynomial against the type-IND adversary in the random oracle model.*

To be specific, let γ be a random oracle and $\mathcal{A}_1$ be a Type-IND adversary with the advantage $Adv_{\mathcal{A}_1}^{IND-CCA}$ against our proposed scheme. Hypothetically, $\mathcal{A}_1$ requests a total of $Q_\gamma > 0$ queries to the oracle γ; then there is an algorithm $\mathcal{C}$ that can determine all the correct coefficients for the given Lagrange interpolating polynomial with the advantage of at least $\frac{2}{Q_\gamma} Adv_{\mathcal{A}_1}^{IND-CCA}$.

Proof. A polynomial $f(x)$ with the sequence of the identity attributes $AT^{v_i} = \{AT_1^{v_i}, AT_2^{v_i}, ..., AT_n^{v_i}\}$ $(1 \leqslant i \leqslant n)$ and a secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ consist of an instance of the Lagrange interpolating polynomial, where $f(x) = a_0^i + H(AT_1^{v_i})x + H(AT_2^{v_i})x^2 + ... + H(AT_n^{v_i})x^n$. The target data provider's index is defined as t $(1 \leqslant t \leqslant n)$. The challenger $\mathcal{C}$ aims to determine AT^{v_t} via executing $\mathcal{A}_1$ as the subroutine. Next, $\mathcal{C}$ and $\mathcal{A}_1$ play the game defined in Section 4.3.2.

- *Initialise*

  $\mathcal{C}$ first generates the public parameter $pp = (q, H, AES)$ and then sends pp to $\mathcal{A}_1$. After that, $\mathcal{C}$ generates n data providers (key-value pairs) $\{Rn^i, AT^{v_i} | 1 \leqslant i \leqslant n\}$ and the target data provider is denoted by $\{Rn^t, AT^{v_t}\}$. Note that all the generated Rn^i $(1 \leqslant i \leqslant n)$ are given to the adversary $\mathcal{A}_1$. Finally, $\mathcal{C}$ initialises one empty lists $List_\gamma$ and updates it continuously in the random oracle query *Identity attributes query*. If the same input is asked multiple times, the same answer will be returned.

- *Queries*

  $\mathcal{C}$ can respond to the queries requested by $\mathcal{A}_1$ polynomial times in the following ways.

  1. *Identity attributes query* (i): $\mathcal{C}$ generates the sequence of the identity attributes $rAT^{v_i} = \{rAT_1^{v_i}, rAT_1^{v_i}, ..., rAT_n^{v_i}\}$ randomly and saves (i, rAT^{v_i}) in $List_\gamma$ if it is the first time that i is queried. Then $\mathcal{C}$ respond with the sequence rAT^{v_i}. Otherwise, $\mathcal{C}$ should retrieve the sequence rAT^{v_i} from $List_\gamma$ directly then return it to $\mathcal{A}_1$.
  2. *Encrypt query*(M, i): $\mathcal{C}$ uses the algorithm *Encrypt* to output the ciphertext $C = Encrypt(pp, M, AT^{v_i})$ and the point P (P should be on the polynomial constructed with AT^{v_i} in the algorithm *Encrypt*).
  3. *Decrypt query*(C, P, i): $\mathcal{C}$ tries to decrypt C via running $Decrypt(pp, C, P, AT^{v_i})$ then responds with the plain message. Note that there is a conditional branch caused by i to be discussed.

  If $i = t$, for each item (i, rAT^{v_i}) in $List_\gamma$, $\mathcal{C}$ executes the operations.

  – Reconstruct the Lagrange interpolating polynomial $g^*(x)$ with rAT^{v_i} and P to determine the secret key $k = a_0$ for AES decryption.
  – Recover (M, H_M) by computing $AES'_k(C) = AES'_k(AES_k(M, H_M))$.

– If $H(M) = H_M$ holds, $\mathcal{C}$ returns M to $\mathcal{A}_1$. If there is no item in the $List_\gamma$ that satisfies the condition, $\mathcal{C}$ returns $\perp$ to $\mathcal{A}_1$.

If $i \neq t$, $\mathcal{C}$ runs the $Decrypt(pp, C, P, AT^{v_i})$ algorithm directly and then sends the output to $\mathcal{A}_1$ as the answer.

- *Challenge*

$\mathcal{A}_1$ submits two messages $M_1^*, M_2^* \in \{0,1\}^\lambda$ with the same length to $\mathcal{C}$, then $\mathcal{C}$ picks one random bit ρ from the set $\{0,1\}$. Finally, $\mathcal{C}$ computes the ciphertext C^* of M_ρ^* via the following steps:

1. Choose a secret key $k \in \mathbb{Z}_q$ for AES encryption and decryption;
2. Determine $f^*(x) = k + H(AT_1^{v_t})x + H(AT_2^{v_t})x^2 + ... + H(AT_n^{v_t})x^n$;
3. Pick a random point $P^*(x^*, f^*(x^*))$ on $f^*(x)$;
4. Compute $C^* = AES_k(M_\rho^*, H(M_\rho^*))$.

Finally, $\mathcal{C}$ sends the ciphertext C^* and the point P^* to the adversary $\mathcal{A}_1$.

- *Constraints*

1. (M_0^*, t) and (M_1^*, t) are not allowed to appear in the above *Encrypt query*;
2. The target data provider's index t and the challenge ciphertext C^* are not allowed to appear in the above *Decrypt query*.

- *Guess*

$\mathcal{A}_1$ outputs one bit ρ' from the set $\{0,1\}$. At the same time, $\mathcal{C}$ picks a random element (i, rAT^{v_i}) from $List_\gamma$ as the answer to the above given instance of the Lagrange interpolating polynomial.

- *Probability analysis*

An event $\mathcal{E}$ is defined as that the adversary $\mathcal{A}_1$ requests a query for the target sequence AT^{v_t} in the *Identity attributes query* during the described game above. If the event $\mathcal{E}$ has happened, AT^{v_t} occurs in at least one item of $List_\gamma$ at the end of this game.

However, if $\mathcal{E}$ does not happen, it means that $Pr[\rho^* = \rho^{*'} | \neg\mathcal{E}] = \frac{1}{2}$ holds. Meanwhile, the condition $Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant |Pr[\rho = \rho'] - \frac{1}{2}|$ holds because of the definition of the type-IND adversary ($\mathcal{A}_1$). Based upon the above analysis, the next two derivations can be illustrated.

$$\begin{aligned}
Pr[\varphi = \varphi'] &= Pr[\varphi = \varphi' | \mathcal{E}]Pr[\mathcal{E}] + Pr[\varphi = \varphi' | \neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&\leqslant Pr[\mathcal{E}] + Pr[\varphi = \varphi' | \neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&= Pr[\mathcal{E}] + \tfrac{1}{2}Pr[\neg\mathcal{E}] \\
&= Pr[\mathcal{E}] + \tfrac{1}{2}(1 - Pr[\mathcal{E}]) \\
&= \tfrac{1}{2} + \tfrac{1}{2}Pr[\mathcal{E}] \\
Pr[\varphi = \varphi'] &\geqslant Pr[\varphi = \varphi' | \neg\mathcal{E}]Pr[\neg\mathcal{E}] \\
&= \tfrac{1}{2}Pr[\neg\mathcal{E}] \\
&= \tfrac{1}{2} - \tfrac{1}{2}Pr[\mathcal{E}]
\end{aligned}$$

Hence, we can deduce that the following derivation holds:

$$Adv_{\mathcal{A}_1}^{IND-CCA} \leqslant |Pr[\rho = \rho'] - \tfrac{1}{2}| \leqslant \tfrac{1}{2}Pr[\mathcal{E}].$$

We can simplify this derivation such that $Pr[\mathcal{E}] \geqslant 2Adv_{\mathcal{A}_1}$.

In conclusion, at the end of the game between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}_1$, the probability of the target sequence AT^{v_t} being in the element(s) of $List_\gamma$ is at least $2Adv_{\mathcal{A}_1}^{IND-CCA}$.

Hence, the probability of generating the correct answer $\rho = \rho'$ is at least $\frac{2}{Q_\gamma} Adv_{\mathcal{A}_1}^{IND-CCA}$, where Q_γ represents the total number of the elements in the list $List_\gamma$. □

6.3. Data Integrity

In our CRA^3 scheme, the hash value $H_M = H(M)$ generated in the *Encrypt* algorithm can provide the data integrity of M. In the *Decrypt* algorithm, if the received C or P is incorrect or manipulated by the attacker in the communication between U_A and U_B, the wrong C (or P) leads to the abnormal result of AES decryption result so that $(M, H_M) = AES'_k(C)$ are incorrect (where $C = AES_k(M, H_M)$ and k is computed from P. Therefore, the condition $H(M) = H_M$ (step 4) cannot hold, which means our data integrity check can detect an abnormal C or P to protect the data integrity of M.

6.4. Comparison of Security Features

In Table 1, we compare the implemented security features of different blockchain-related authorisation schemes from the state-of-the-art of related work with that of our CRA^3 scheme. It is clear that most of the compared schemes can support permissioned blockchains but CRA^3 is the only one that can support an untrusted consensus network. Meanwhile, our CRA^3 can also provide authorisation, confidentiality, and integrity for data access. However, in other compared schemes, the integrity feature is only implemented by [15] and no scheme considers confidentiality. The *Decentralizing Privacy* (DP) [12] scheme requires a database as a storage media; however, the DP scheme itself cannot support confidentiality or integrity.

Table 1. Comparison of the security features in different blockchain-related authorisation schemes.

Scheme	Blockchain Type	Consensus Network Type	Security Features		
			Authorisation	Confidentiality	Integrity
[12]	Public/Permissioned	Trusted	√	× [1]	× [1]
[16]	Public	Trusted	√	×	×
[19]	Permissioned	Trusted	×	×	×
[15]	Permissioned	Trusted	√	×	√
CRA^3 *	Permissioned	Trusted/Untrusted	√	√	√

[1] The scheme depends on the deployed database to support the mentioned security feature. * CRA^3: our proposed scheme, Challenge-Response Assisted Access Authorisation.

7. Performance Evaluation and Results

The performance simulation and results are illustrated and discussed in this section. Two Raspberry Pi 2s [27] with Wi-Fi (as the mobile devices of the users U_A and U_B) and one conventional computer with an Intel i5 processor running at 3.30 GHz (as a node of the consensus network in the permissioned blockchain) were used to perform the simulation. The local computational time efficiency for executing CRA^3 was evaluated with respect to the time cost for transmitting encrypted data over Wi-Fi and the transaction fee (gas) of the consensus node in the simulation. Since there is yet no clear best practice to be used as a baseline for comparison, we selected an authorisation scheme for blockchain-based storage named *Decentralizing Privacy* (DP) [12] as our baseline. The authorisation supported by a trusted third party (TTP) in DP is policy-based but not anonymous, since the TTP knows the users' identities. However, the designed authorisation in CRA^3 is attribute-based and anonymous. Note that all the implemented experiments used the equivalent cryptographic security level (128-bit) [28], and that the transaction fee (gas) was calculated based upon the bytecodes generated by Ethereum Virtual Machine (EVM) [29] with PoA (Proof of Authority) [30] as the consensus mechanism.

First, the number of the attributes used for authorisation was varied from two to 10 in CRA^3 (respective of policies in DP) to compare the time taken for local computation including authorisation, encryption, and decryption algorithms in the two schemes implemented on a conventional computer.

The averaged results over 10 runs are shown in Figure 3. In the authorisation phase (Figure 3a), the time cost in both schemes increased with a similar trend when the number of attributes used was small. If the number of attributes used rose up to 10, our CRA^3 scheme needed 25% more time to authorise the access when compared with the DP scheme. For the encryption and decryption phases, the time cost for the DP scheme kept stable whilst the time cost of the CRA^3 scheme increased slowly, increasing with the number of attributes. On average, the time cost of the CRA^3 scheme was 55% lower than that of the DP scheme; see Figure 3b,c.

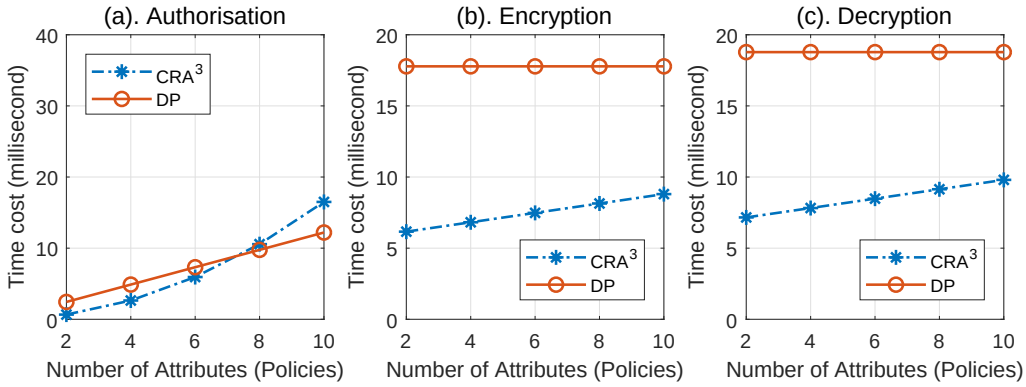

Figure 3. The time cost comparison of local computation on a Raspberry Pi 2 between CRA^3 and DP (*Decentralise Privacy* scheme [12]).

Meanwhile, we measured the time cost for transporting data between users and CN_{pm} (see Section 4.1) over Wi-Fi (Figure 4). The data included the attributes (i.e., policies) used for authorisation, the encrypted data (128 bytes) and the keys used for decryption in the two schemes. Since CRA^3 only transmits two points in the *Authorise* phase whereas the DP scheme requires two policy lists for authorisation, the time consumption for transmitting data via Wi-Fi in CRA^3 was about 24% lower than that in the DP scheme. Furthermore, the time cost in CRA^3 had a lower growth rate when compared with the DP scheme.

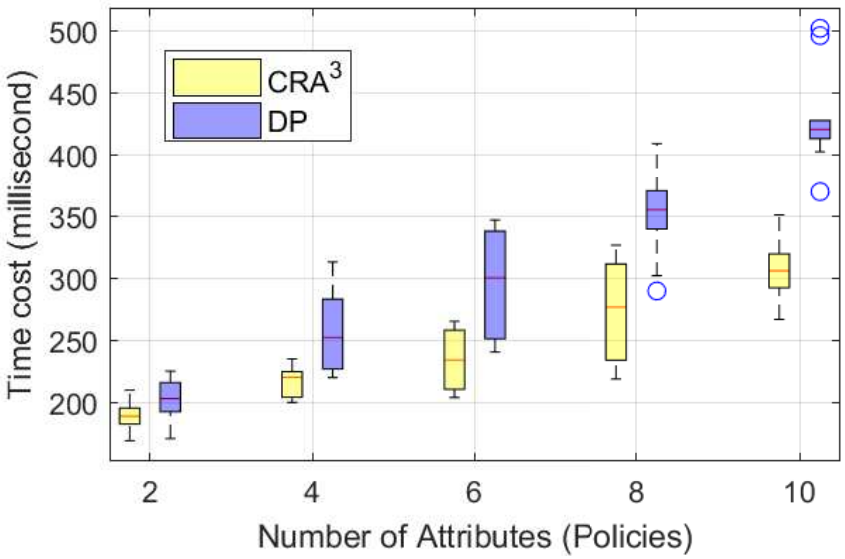

Figure 4. Comparison of the time costs of CRA^3 and DP [12] for transmitting data over Wi-Fi.

Thus, we summarise the total time cost of both local computation and data transmission via Wi-Fi in Figure 5. The total time cost in CRA^3 was around 30% lower than that in the DP scheme. While the

number of used attributes (i.e., policies) increased, the DP scheme consumed far more time than CRA^3, in total.

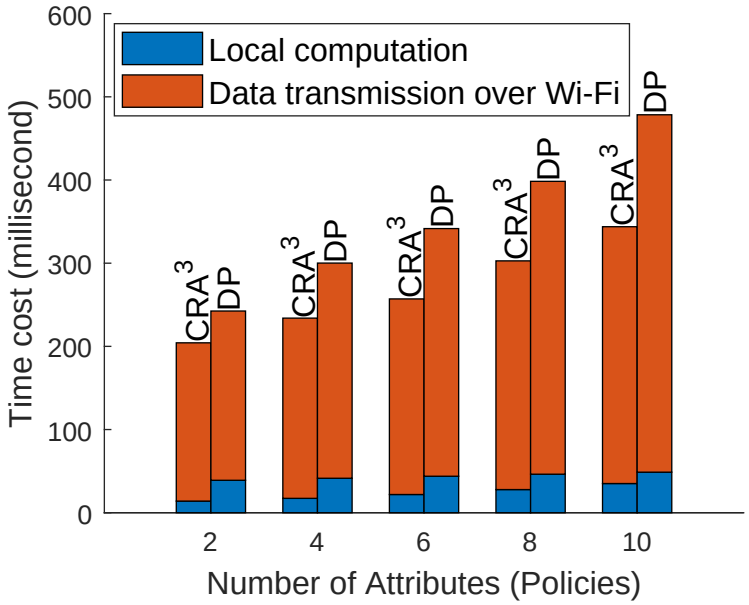

Figure 5. Comparison of total time cost of CRA^3 and DP [12].

Finally, the transaction fee (gas) for the *Authorise* phase performed in the consensus network was evaluated in a conventional computer (Figure 6). While the transaction fee of CRA^3 kept stable (and was non-sensitive to the variation of used attributes), the transaction fee increased by the number of used policies in the DP scheme. This is because the DP scheme compares two policy lists in the transaction for authorisation but CRA^3 only compares two points regardless of the number of used attributes.

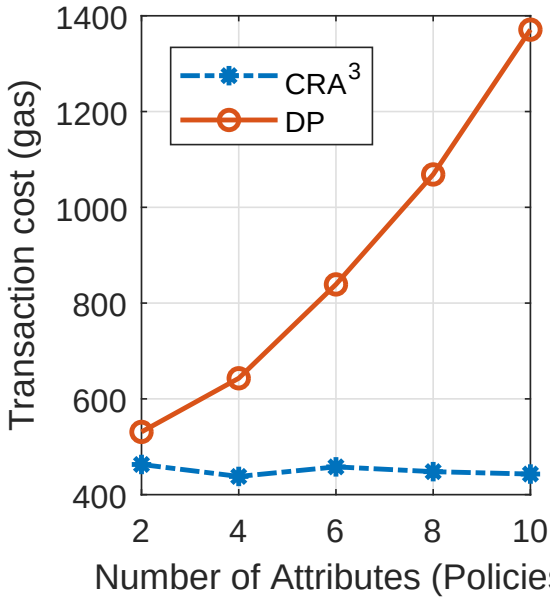

Figure 6. Transaction fee of CRA^3 and DP [12] for authorisation.

8. Conclusions

In this paper, we proposed a privacy-enhanced authorisation CRA^3 scheme under a consideration of untrusted nodes occurring in a consensus network of permissioned blockchain. Unlike existing work [10,12,16], CRA^3 does not expose users' credentials to the untrusted nodes in the consensus network for authorising data access. By applying CRA^3 in a permissioned blockchain, users (data providers) can share private data with valid data requesters without leaking their private information. Therefore, CRA^3 can help people to safeguard their privacy and prevent potential privacy leakage (e.g., caused by attackers) in permissioned blockchains. In terms of the communication overhead, CRA^3 reduces the time cost for the communication during the authorisation since the size of the required data for authorising data access request is much smaller when compared with other methods. Furthermore, our consensus verification only relies on one equation and other computational work is executed by the data requester and receiver; hence, the consensus cost (transaction fee) is visibly cut down to save the user's cost and the computational resource of the consensus network (i.e., lower workload) simultaneously.

Author Contributions: Conceptualization, X.Z., C.L., K.K.C. and S.P.; methodology, X.Z.; software, C.L.; validation, X.Z.; formal analysis, X.Z.; investigation, C.L.; writing—original draft preparation, X.Z. and C.L.; writing—review and editing, K.K.C. and S.P.; visualization, X.Z.; supervision, K.K.C. and S.P.; funding acquisition, K.K.C. and S.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by UK Research and Innovation (UKRI) with grant number 104317.

Acknowledgments: This research was supported in part by a PhD scholarship funded jointly by the China Scholarship Council and Queen Mary University of London.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nakamoto, S. Bitcoin: A Peer-to-peer Electronic Cash System. Available online: https://nakamotoinstitute.org/bitcoin/ (accessed on 31 October 2008).
2. Sukhwani, H.; Martínez, J.M.; Chang, X.; Trivedi, K.S.; Rindos, A. Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In Proceedings of the 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS), Hong Kong, China, 26–29 September 2017; pp. 253–255.
3. Noyes, C. Bitav: Fast anti-malware by distributed blockchain consensus and feedforward scanning. *arXiv* **2016**, arXiv:1601.01405.
4. Kopp, H.; Mödinger, D.; Hauck, F.; Kargl, F.; Bösch, C. Design of a privacy-preserving decentralized file storage with financial incentives. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Paris, France, 26–28 April 2017; pp. 14–22.
5. Zhang, Y.; Wen, J. An IoT electric business model based on the protocol of bitcoin. In Proceedings of the 2015 18th International Conference on Intelligence in Next Generation Networks (ICIN), Paris, France, 14 June 2015; pp. 184–191.
6. Zhang, X.; Poslad, S. Blockchain support for flexible queries with granular access control to electronic medical records (EMR). In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, KC, USA, 20–24 May 2018; pp. 1–6.
7. Sharples, M.; Domingue, J. The blockchain and kudos: A distributed system for educational record, reputation and reward. In *European Conference on Technology Enhanced Learning*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 490–496.
8. Liu, C.; Chai, K.K.; Zhang, X.; Chen, Y. Peer-to-peer electricity trading system: Smart contracts based proof-of-benefit consensus protocol. *Wirel. Netw.* **2019**, 25, 1–12. [CrossRef]
9. Buterin, V. On Public and Private Blockchains. Available online: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/ (accessed on 7 August 2015).
10. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23–26 April 2018; p. 30.

11. Smetanin, S.; Ometov, A.; Komarov, M.; Masek, P.; Koucheryavy, Y. Blockchain Evaluation Approaches: State-of-the-Art and Future Perspective. *Sensors* **2020**, *20*, 3358. [CrossRef] [PubMed]
12. Zyskind, G.; Nathan, O. Decentralizing privacy: Using blockchain to protect personal data. In Proceedings of the Security and Privacy Workshops (SPW), San Jose, CA, USA, 21–22 May 2015; pp. 180–184.
13. Quirós-Tortós, J.; Ochoa, L.F.; Lees, B. A statistical analysis of EV charging behavior in the UK. In Proceedings of the 2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM), Montevideo, Uruguay, 5 October 2015; pp. 445–449.
14. Hafez, O.; Bhattacharya, K. Queuing analysis based PEV load modeling considering battery charging behavior and their impact on distribution system operation. *IEEE Trans. Smart Grid* **2016**, *9*, 261–273. [CrossRef]
15. Gai, K.; Wu, Y.; Zhu, L.; Xu, L.; Zhang, Y. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet Things J.* **2019**, *6*, 7992–8004. [CrossRef]
16. Yue, X.; Wang, H.; Jin, D.; Li, M.; Jiang, W. Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control. *J. Med. Syst.* **2016**, *40*, 218. [CrossRef] [PubMed]
17. Wenbo, M. *Modern Cryptography: Theory and Practice*; Prentice Hall PTR: Upper Saddle River, NJ, USA, 2003.
18. Dorri, A.; Steger, M.; Kanhere, S.S.; Jurdak, R. Blockchain: A distributed solution to automotive security and privacy. *IEEE Commun. Mag.* **2017**, *55*, 119–125. [CrossRef]
19. Min, X.; Li, Q.; Liu, L.; Cui, L. A permissioned blockchain framework for supporting instant transaction and dynamic block size. In Proceedings of the 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, 23 August 2016; pp. 90–96.
20. Pop, C.; Cioara, T.; Antal, M.; Anghel, I.; Salomie, I.; Bertoncini, M. Blockchain based decentralized management of demand response programs in smart energy grids. *Sensors* **2018**, *18*, 162. [CrossRef] [PubMed]
21. Ateniese, G.; Camenisch, J.; Joye, M.; Tsudik, G. A practical and provably secure coalition-resistant group signature scheme. In *Annual International Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 255–270.
22. Boneh, D.; Shacham, H. Group signatures with verifier-local revocation. In Proceedings of the 11th ACM Conference on Computer and Communications Security, Washington, DC, USA, 25–29 October 2004; pp. 168–177.
23. Ling, S.; Nguyen, K.; Roux-Langlois, A.; Wang, H. A lattice-based group signature scheme with verifier-local revocation. *Theor. Comput. Sci.* **2018**, *730*, 1–20. [CrossRef]
24. Perera, M.N.S.; Nakamura, T.; Hashimoto, M.; Yokoyama, H. Traceable and Fully Anonymous Attribute Based Group Signature Scheme with Verifier Local Revocation from Lattices. In *International Conference on Network and System Security*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 675–684.
25. Dawson, E.; Donovan, D. The breadth of Shamir's secret-sharing scheme. *Comput. Secur.* **1994**, *13*, 69–78. [CrossRef]
26. Daemen, J.; Rijmen, V. Reijndael: The Advanced Encryption Standard. *Dobb J. Softw. Tools Prof. Program.* **2001**, *26*, 137–139.
27. Monk, S. *Programming the Raspberry Pi: Getting Started with Python*; Mcgraw-Hill: New York, NY, USA, 2013.
28. Barker, E.; Barker, W.; Burr, W.; Polk, W.; Smid, M. Recommendation for key management part 1: General (revision 3). *Nist Spec. Publ.* **2012**, *800*, 1–147.
29. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *EIP-151*, 1–32.
30. Ekparinya, P.; Gramoli, V.; Jourjon, G. The attack of the clones against proof-of-authority. *arXiv* **2019**, arXiv:1902.10244.