Towards an Efficient Access Control and Computation Environment for IoT using Blockchain

Kwame O.-B Obour Agyekum[1], Jianbin Gao[2], Emmanuel Boateng Sifah[1], Hanlin Yang[1] and Qi Xia[3]
[1]School of Computer Science, University of Electronic Science and Technology of China, Chengdu, Sichuan, P.R.China
[2]Center for Digital Health, School of Resource and Environment, University of Electronic Science and Technology of China, Chengdu, Sichuan, P.R.China
[3]Center for Cyber Security, University of Electronic Science and Technology of China, Chengdu, Sichuan, P.R.China

Abstract—*Access to data has gained enough root due to the advent of technology. The Internet of Things (IoT) is believed to be a massive hit in some years to come. However, there are lots of malicious attempts by users when owners' data fall into the wrong hands. Gaining access to data isn't a problem now, but how the data is used in such a way as preserving the privacy of owners is a question yet unanswered. In this paper, we present a blockchain-based access control and a computation environment for data processing. Authentication to the blockchain network is achieved through the Elliptic Curve Digital Signature Authentication (ECDSA) protocol, which is robust against most attacks. Data owners are privileged to state to the blockchain network which information of theirs is sensitive and which ones are not. Processing of sensitive data is made in the computation environment while smart contracts are bound to insensitive data. We evaluated the latency of the computation environment and the results were satisfactory.*

Keywords: Access Control, blockchain, Internet of Things, smart contract, virtual computation environment

1. Introduction

The Internet of Things (IoT) has turned out to be a technology that has huge influence across many markets. IoT services are perceived to provide global reach across billions of simple, tiny devices. The rapid development of communication and networking technologies, a growing number of objects are connected to the internet day-in-day-out. The ubiquitous interconnection of physical objects essentially quickens information gathering, aggregation and sharing in the IoT, making the IoT a standout amongst the most basic structures for different applications, for example, smart healthcare, intelligent transportation, home automation, etc. [1], [2]. Nonetheless, such interconnection may likewise cause critical security concerns into IoT frameworks, since malicious entities can interfere with the frameworks to increase illicit access to the provided assets (e.g. information, services, computing units, etc.) by just deploying their own, or trading off existing IoT devices [3], [4]. Besides that, the constrained abilities of numerous IoT gadgets, and also the current access control frameworks in view of centralized and various leveled structures, create new difficulties in the IoT domain. Access control has been regarded as an increasingly important research issue in IoT [5] - [7].

Customary access control models are usually built on top of well-known access control models such as role-based access control model (RBAC) [8], attribute-based access control model (ABAC) [9] and the capability-based access control model (CapBAC) [10]. Notably, validating the access rights of subjects in the above schemes is done through a centralized entity, which turns out to be a single point of failure. Centralized access control frameworks were intended to address the issues of conventional human-machine oriented internet situations where devices are within the same trusted domain, which typically requires centralized access administration. In any case, some IoT scenarios are considerably more powerful than the customary situations in which IoT devices might be versatile [11] and belong to various management groups amid their lifetime. On the other hand, IoT devices can be overseen by a few managers at the same time. In addition, numerous IoT devices are constrained by power capabilities and can also be easily compromised by adversaries, so they cannot be fully trusted as access right validation entities. As a result, in an untrustworthy environment, there is the tendency for access control models to fail.

The question then is: how can we ensure distributed and trustworthy access control in IoT environments? The answer may reside in the implementation of blockchain technology, a key component behind modern cryptosystems like Ethereum [12] and Bitcoin [13]. Due to several advantages this technology has such as scalability, transparency and concurrency, centralized access control is eliminated. A single centralized access control server might become a bottleneck when access control queries and updates are frequent. One key component of this blockchain technology is the smart contract, an executable code that resides on the blockchain. On the basis of this, the blockchain has evolved into a promising distributed development platform that ensures trustworthiness is applications, and has hence attracted considerable attentions from several researches in IoT environments.

In this paper, however, we implement the blockchain technology in controlling access to data by users. Access is granted to a user by authorizing a user through the Elliptic Curve Digital Signature Authentication (ECDSA) module. Data owners also specify to the blockchain system which data of theirs are sensitive and not sensitive. For sensitive data, we make use of a computation environment where the data queried is fetched by the processing nodes, computations are made, and final results of the computation are presented to the user. For non-sensitive data, we employ smart contracts in the control of how a user makes use of data accessible to him. The paper therefore aims to achieve an efficient access control and data computation using the blockchain technology.

The remainder of this paper is organized as follows. Section 2 introduces the works related to access control while section 3 explains the key technologies used in this paper. We outline the architecture and design formulation of our blockchain system in Section 4. Section 5 addresses the security model of our work, and system evaluation and performance analysis is given in section 6. Section 7 concludes the paper.

2. Related Works

Authors in [14] conducted a systematic literature review on blockchain technology for the IoT. The survey described several research papers that manage the data collected by IoT devices. As an example, Wilson et al [15] described a system that verifies the identity of data, and Zyskind et al [16] described a method for preserving the data ownership of IoT devices. However, none of the papers in the survey propose an architecture that addresses access policies of the IoT devices in their entire lifecycle.

Authors in [17] considered the access control issue in an IoT network with service providers' cloud storage, client devices and smart homes, each containing a miner and numerous IoT gadgets. Each home miner keeps up a local private blockchain with a policy header putting away access control policies to control all the access requests related to the home. However, the authors disposed off the basic proof-of-work concept [18] in the blockchain innovation, bringing about an untrustworthy access control scheme. The fundamental motivation behind the blockchain is to serve as an immutable storage and distributed system for access control policies, though the processing capability of the blockchain was to a great extent squandered.

Utilizing the blockchain in storing access control policies has additionally been adopted in [16], [19]. As of late, the computing ability of the blockchain has been exploited in [20] for access control, where the blockchain assumes the role of a decentralized access control director. The authors utilized access tokens to represent access rights and the tokens can be transferred from one associate then onto the next through exchanges. When delivering a token, the sender installs access control policies into the locking scripts of the transaction output. The recipient of the token must open the locking script to demonstrate ownership of the token (i.e., the access rights to a certain resource). Utilizing this plan, an associate can be allowed access rights by getting a token, grant access rights to another subject by transferring the token, and access an object by spending a token. In spite of the fact that utilizing locking scripts for access control is a great thought, the computing capability of the locking scripts is essentially restricted.

3. Background

Data owners grant access to users who want to access their data, usually from IoT devices such as sensors. However, when data is moved to the cloud, data owners do not have direct control of their data and therefore need to rely on external systems to provide the same guarantees such as data availability, selective data access, protection against attacks, etc. Cloud providers are most of the time assumed to be honest-but-curious and hence trusted to effectively manage data. Encryption techniques are predominantly used to prevent malicious intents, however there are some disadvantages. In this section, definitions are given to the various techniques employed in this paper.

3.1 Blockchain

The blockchain is ideally a distributed database that does not require a central authority and dispenses with the requirement of a third party. A blockchain contains a set of blocks, and each block contains a hash of the previous block, making a chain of blocks from the beginning block (genesis) to the present/latest block. A genesis block is the principal block in the blockchain. Bitcoin [18] was the first widely used implementation of peer-to-peer trustless electronic cash. However, the potential uses of blockchain technology go beyond Bitcoin. In blockchain, there is decentralized control where no central authority dictates the rules of transactions. Full copies of every transaction ever executed in the system are stored on the blockchain, and made public to everyone on the network. This enhances data transparency and auditability. Information is also distributed on the network as every node keeps a copy of the transactional history. Transactions are validated by all the nodes on the network, and this breaks the paradigm of centralized consensus. For security concerns, the blockchain network is tamper-proof and cannot be manipulated by malicious users. These are the major advantages of the blockchain technology, and therefore an ideal component of solutions to IoT systems.

3.2 Internet of Things (IoT)

The Internet of Things (IoT) is a set of next generation devices that can sense aspects of the real world, like temperature, lighting, the presence or absence of people or objects, etc. and report that real-world data, or act on it. The Internet of Things is a system of systems, meaning all electronic devices will be connected to each other in a local area, forming a system, and further, these systems will be connected to each other, thereby forming a bigger, networked system.

3.3 Access Control

Access control is a method of controlling passage into or out of an area. In this work, we employ an authentication and access control module where only participants registered on the blockchain network can access information from data owners. Authentication is achieved by the verification of individuals who want to access information by the nodes on the blockchain network. The Identity Based Cryptography (IBC) traditionally suffers from the key-escrow problem as IBC requires the existence of a totally trusted entity, and this entity can sometimes impersonate a user in the system. Due to the utilization of the blockchain technology, every malicious intent is curbed. However, attribute-based access control is preferred in this work, as access is granted based on specification by the data owner on who can and cannot access data. Detailed description of how things work are given in the preceding sections.

4. System Model

4.1 ECC Preliminaries

In the mid 1980's, Victor Miller and Neal Koblitz first introduced the concept of Elliptic Curve Cryptography (ECC). The curve in ECC is defined by an equation $y^2 = x^3+ax+b$, where $4a^3 + 27b^2 \neq 0$. The security of ECC is based on the elliptic curve discrete logarithm problem (ECDLP), which has its definition as follows:

Given a base point, it is extremely difficult and almost improbable to compute the discrete logarithm of a random elliptic curve element. Let E be an elliptic curve defined over a finite field F_q, G be a random generator with an order p. Given a $Q = kG$, $k \in Z_p$, it is almost infeasible to find the integer k in polynomial time.

Compared to the RSA algorithm, ECC can ensure the same security level but with a smaller key size, as there is difficulty in solving the ECDLP than in integer factorization. In this paper, ECC is used as the basis for ensuring an efficient authentication between the verification unit and a prospective user who wants to access data owner's information on the cloud, via the blockchain network. The underlying technique used is the ECDSA, which shall be explained in details.

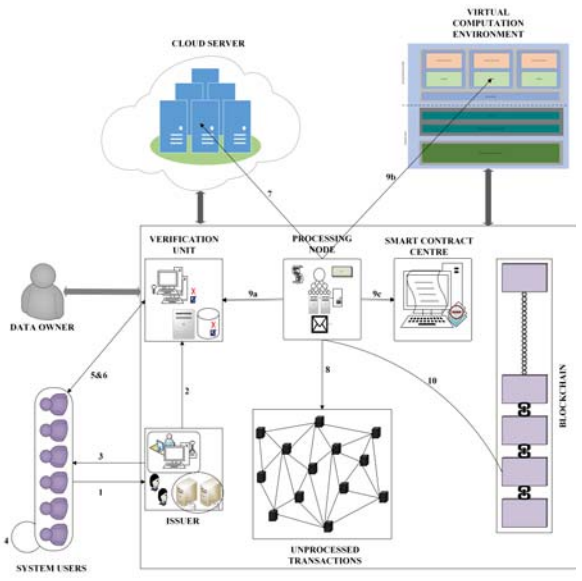

Fig. 1: System Architecture

4.2 System Design Formulation

The overall organogram of the system model is presented as follows.

1) A user who wants to access data contacts the issuer for network membership registration.
2) The issuer, upon receiving the request from the user, registers the user and issues a membership key (acts as an ID). The issuer fetches parameters from the Key Storage Unit, and based on the attributes of the user generates the membership key. The details of the user are then sent to the verifier.
3) Upon successful registration, the issuer gives the keying parameter to the user. This keying parameter serves as a proof of identity to be used on the system.
4) The user generates a private key to be used for any form of communication on the network. The key generation is shown in the authentication process in the next sections.
5) The verifier and the user establish a mutual authentication to check for the validity of the user's membership and also its genuineness. The verifier checks the user details from its database. Authentication is achieved by the ECDSA authentication method.
6) After a successful verification process, the user can now access the data s/he's requesting for.
7) The processing node then fetches the data requested from the user.
8) Due to the implementation of the blockchain network, each communication or transaction is stored and kept on the block. For scalability concerns, the hash of each transaction is stored, rather than the actual transaction. However, the transactions need to be processed in the

first place. The processing node fetches the unprocessed transactions (data access requests) and prepares to process them.

9) Prior to the processing of transactions, the processing node must check that the public key used for the transaction has been approved to participate as such. It looks up the database of the verification unit and verifies whether it is indeed there. If the public key of the user exists in the database, the processing node can proceed with processing the transaction. Otherwise, it ignores the transaction. For sensitive data, the processing node sends the data to the virtual computation environment for the necessary computations to be performed. For insensitive data, smart contracts are prepared and bound to the data before given to the user.
10) After all transactions are completed, they are then recorded into blocks.

4.3 Authentication Process

For a user to start any transaction on the network, he has to be verified by the verification unit. The various processes that lead to the user being authenticated are explained below.

1) **Setup Phase**: The domain parameters of the ECDSA algorithm are comprised of a field size q, where either $q = p$, and odd prime, or $q = 2^m$. There is also an indication FR, which is a field representation used for the elements of F_q. Two field elements a and b, which are in F_q are chosen and they define the equation of the elliptic curve E over F_q (i.e., $y^2 = x^3 + ax + b$ in the case $p > 3$, and $y^2 + xy = x^3 + ax^2 + b$ in the case $p > 2$). Furthermore, two field elements x_G and y_G define a finite point $G = (x_G, y_G)$ of prime order in $E(F_q)$. A point G of order n is also chosen, with $n > 2^{160}$ and $n > 4\sqrt{q}$; also a cofactor $h = \#E(F_q)/n$
2) **Domain Parameter Generation**
   a) Coefficients a and b are selected from F_q. Let E be the curve $y^2 = x^3 + ax + b$ in the case $q = p$, and $y^2 + xy = x^3 + ax^2 + b$ in the case $q = 2^m$.
   b) We then compute $N = \#E(F_q)$.
   c) Verification is then made whether N is divisible by a large prime $n(n > 2^{160}$ and $n > 4\sqrt{q})$. If not, then we resort to Step 1.
   d) We then verify that n does not divide $q^k - 1$ for each k, $1 \le k \le 20$. If not, we again resort to Step 1.
   e) We again verify that $n \ = q$. If not, we go back to Step 1.
   f) Finally, an arbitrary point G belonging to $E(F_q)$ is selected and we set $G = (N/n)G$. We repeat until $G \neq \emptyset$
3) **Domain Parameter Validation**: The domain parameter validation ensures that the domain parameters have the requisite arithmetical properties. The main aim for performing this validation in practice is to prevent malicious insertion of invalid domain parameters which may enable some attacks. The domain parameters D is a set that contains $D = (q, FR, a, b, G, n, h)$, and for the validation, a user performs explicit domain parameter validation as shown in *alg. 1*.

Algorithm 1: Validation of EC Domain Parameters

```
Require: INPUT a set of EC domain parameters
    $D = (q, FR, a, b, G, n, h)$
Ensure: OUTPUT either Acceptance or Rejection of the
    validity of $D$
 1: Verify whether $q$ is an odd prime $(q = p)$ or a power
    of 2 $(q = 2^m)$
 2: Verify that $FR$ is a "valid" representation of $F_q$
 3: Verify that $G \neq \emptyset$
 4: Verify that $a, b, x_G$, and $y_G$ are properly represented
    elements of $F_q$ (i.e, integers in the interval $[0, p-1]$ in
    the case $q = p$, and bit strings of length $m$ bits in the
    case $q = 2^m$).
 5: Verify whether $a$ and $b$ define the elliptic curve over $F_q$
    (i.e, $4a^3 + 27b^2 \neq 0 \mod p$ if $q = p$; $b \neq 0$ if $q = 2^m$)
 6: Verify that $G$ lies on the elliptic curve defined by $a$
    and $b$ (i.e, ${y_G}^2 = {x_G}^3 + ax_G + b$ in the case $q = p$, and
    ${y_G}^2 + x_G y_G = {x_G}^3 + a{x_G}^2 + b$ in the case $q = 2^m$)
 7: Verify that $n$ is prime
 8: Verify that $n > 2^{160}$ and also $n > 4\sqrt{q}$
 9: Verify that $nG = \emptyset$
10: Compute $h' = [(\sqrt{q} + 1)^2 / n]$, and verify that $h = h'$
11: Verify that $n$ does not divide $q^k - 1$ for each $k$,
    $1 \le k \le 20$
12: Verify that $n \neq q$
13: If any verification fails, then $D$ is invalid; otherwise $D$
    is valid
```

4) **ECDSA Key Pairs**: An ECDSA key pair is associated with a particular set of EC domain parameters. The public key is a random multiple of the base point, while the private key is the integer used to generate the multiple. Because the public key is generated by the issuer and later sent to the verification unit, the user has to prove its identity to the verifier.
   - **ECDSA Key Pair Generation** The user, who after registering to the network and wants to access data, does the following:
     1. Select a random integer d in the interval $[1, n-1]$.
     2. Compute $Q = dG$.
     3. Q is the public key of the user and d is the private key.
   - **Public Key Validation** As mentioned before, the validation is made to check for the same properties and also prevent malicious intents. The assurance

that a public key Q is valid can be provided to a user using the method illustrated in *alg. 2*.

```
Algorithm 2: Validation of ECDSA Public Key
Require: A public key $Q = (x_Q, y_Q)$ associated with valid domain parameters, $D$.
Ensure: Acceptance or Rejection of the validity of $Q$
1: Verify whether $q$ is an odd prime $(q = p)$ or a power of 2 $(q = 2^m)$
2: Check that $Q = \emptyset$
3: Check that $x_Q$ and $y_Q$ are properly represented elements of $F_q$ (i.e, integers in the interval $[0, p-1]$ in the case $q = p$, and bit strings of length $m$ bits in the case $q = 2^m$)
4: Check that $Q$ lies on the elliptic curve defined by $a$ and $b$.
5: Check that $nQ = \emptyset$
6: If any of the checks fail, then $Q$ is invalid; otherwise $Q$ is valid
```

5) **ECDSA Signature Generation and Verification**

- **Signature Generation** In order for a user to prove (sign) his identity, say m, the user with the domain parameters $D = (q, FR, a, b, G, n, h)$ and associated key pair (d, Q) does the following.
  1. Select a random integer k, $1 \leq k \leq n-1$
  2. Compute $kG = (x_1, y_1)$ and $r - x_1 \mod n$. If $r = \emptyset$, then go to Step 1
  3. Compute $k^{-1} \mod n$
  4. Compute $e = H(m)$
  5. Compute $s = k^{-1}(e + dr) \mod n$. If $s = 0$, then go to Step 1
  6. The user's signature for his/her identity m is (r, s)
- **Signature Verification** For a successful verification of the signature of the user, the verifier obtains an authentic copy of the user's domain parameters D, and associated public key Q. It is recommended that the verifier also validates D and Q. The verifier does the following
  1. Verify that r and s are integers in the interval $[1, n-1]$
  2. Compute $e = H(m)$
  3. Computes $w = s^{-1} \mod n$
  4. Computes $t_1 = ew \mod n$ and $t_2 = rw \mod n$
  5. The verifier then computes $X = t_1G + t_2Q$. If $X = \emptyset$, the signature is rejected. Otherwise, compute $v = x_1 \mod n$ where $X = (x_1, y_1)$
  6. Accept the signature if and only if $v = r$

If $v = r$, authentication becomes successful and the user can now be part of the network to access data. Otherwise, the user is revoked access.

5. Security Model

5.1 Smart Contract Design

A smart contract is a contract that is automatically enforced by computer protocols. They are programmable scripts that act as finite state machines that execute laid down instructions when an action has been activated. The continuous usage of blockchain technology has made it easier to register, verify and execute smart contracts. This eliminates the need for an intermediary.

Additionally, smart contracts report logs of every transaction that has taken place on the blockchain network. This ensures transparency on the network as every participant monitors every event on the network. For our smart contract for data access, sets or actions are initialized and are applicable to any form of access request made by a user. The basic action sets are read, write, delete, duplicate, move and copy. These actions when performed on a data will trigger the smart contract to report to the network, based on the rules established for that particular data.

In this work, the sensitivity of data is categorized into two, namely, sensitive (high) and insensitive (low). For sensitive data, the processing nodes fetch the data and make duplicates of it before it is sent to the virtual computation environment. When the processing is complete, results are returned to the user and the duplicate data is destroyed to prevent unnecessary overheads on the network. For insensitive data, and yet still to preserve the privacy of the data, rules pertaining to the usage of the data are stated and attached to the data package. A user is held culpable when any of the said rules are defaulted.

Monitoring of data is performed by the ***getAction*** function.

5.2 Virtual Computation Environment

Upon receipt of data access to sensitive data, the processing node, as mentioned earlier, retrieves the data from the cloud server and forwards it to the virtual computation environment for the necessary computations to be completed. Computation is made on the duplicate of the original data and after giving out the result to the processing node which is then forwarded to the user, the duplicate data is destroyed to prevent overheads.

The computation workspace [21] herein referred to as container, is a data analysis framework empowered through virtualization for big data processing and analytics in view of requests made by the user on specific (sensitive) data. Reports that are generated from the computations are set up in proper organizations and distributed to the user. The computations can be expressed as an information compartment that takes as data copies of the BlockData that have been preprocessed with respect to the preservation of privacy on the contents of the BlockData. BlockData inputs that are fed into the container are bolted and can never be opened, on

Algorithm 3: Contract on Data Access

```
Require: Initialization of Parameters:
    getAction, getSensitivity, getRequesterID, getDataID,
    retrieve, comment, report, accessControl.
Ensure: Setting up functions:
    func(getSensitivity)
    func(getAction)
    func(comment)
    func(accessControl)
    DATA MONITORING
    for func(getAction) == retrieve do
    func(comment)⟵ Reveal, Data with DataID has been retrieved
    report(comment||getRequesterID||getDataID)
    end for
    if func(getSensitivity)==Low then
    func(getAction)⟵retrieve data from cloud server
    func(comment)⟵Reveal, Data with DataID has been retrieved for requester
    func(accessControl)⟵Grant access to data with DataID, with rules on data usage
    report(comment||getRequesterID||getDataID)
    else (funcgetSensitivity)==High then
    func(getAction)⟵retrieve data from cloud server
    func(comment)⟵Reveal, Data with DataID has been retrieved and duplicate sent to virtual container for processing
    func(accessControl)⟵Processing results of data with DataID given to requester
    report(comment||getRequesterID||getDataID)
    end if
```

passage for information processing. This component is to guarantee that input data are pulverized when a collapsed-state action is called upon. The result of the computation would be results and reports, which will be removed from the framework by the applicable elements.

6. Evaluation and Discussion

6.1 Privacy and Security

For an adversary to alter access permissions in the blockchain, it requires the forgery of the digital signature of the user, or gaining control over the processes overseen by the processing node in the blockchain network. Due to the signature stemming from the discrete logarithm problem (hard problem), the security of the signature is assured. Also, because the blockchain is decentralized in its nature, consensus protocols (for example, proof of work) cater for the latter problem. Before the data is appended unto the block and ultimately connected to the chain, the data is encrypted by the processing nodes, integrity protected and authenticated. An adversary who even gains access to encryption keys cannot alter the stored transaction on the blockchain, because of the immutable nature of the blockchain system.

6.2 Access Control

Access control on this blockchain network is permissioned by the processing nodes on the network, based on the list of public keys in the database of the verifier. Prior to the processing of an unconfirmed transaction, the processing node that enforces access control methods must first verify that the public key associated with the transaction is found in the database of the verifier. In other words, it must ensure that the blockchain consists of transactions only from users permitted to transact, due to their public keys been found in the database of the verifier. The processing nodes can revoke access to malicious users, or users whose transacting keys cannot be found in the verifier's database. However, malicious processing nodes could grant access without verification but this action is limited since the transaction is encrypted, and moreover, incentives can be issued to honest nodes (those that follow protocol(s) correctly). Other list-sharing methods can also be deployed such as the processing node maintaining a local copy of the verifier's database, but we want to ensure our blockchain system maintains its scalability trait.

6.3 Latency

For efficient security on any transaction on the blockchain network, a lot of time has to be spent on the processing of a block, due to the double checks that need to be made by the system. Latency for an initial inclusion of a transaction on the blockchain network is relatively higher than in traditional systems. There are other factors that influence the delay in processing requests. Some of these factors include propagation time due to verification of users and acceptance of blocks; latency of the network itself which doesn't take into consideration the size of the block. Bandwidth limitations also affect the delay in processing of the blocks.

On a whole, due to the large number of requests that are made to data access from users, there are queuing times, delay in processing user requests and the time it takes for the processing nodes to finally deliver the results to the users. The smart contracts on the other hand play a part in the latency that is observed in the system, as there are strategic decisions that need to be made before a final output is made available. In this work, however, we record the latency involved the processing of requests by the computation environment, and it is shown in *fig. 2*.

From the figure, it is evident that it takes far less time for the virtual computation environment to compute requests on sensitive data made by a user. Our system therefore performs satisfactorily.

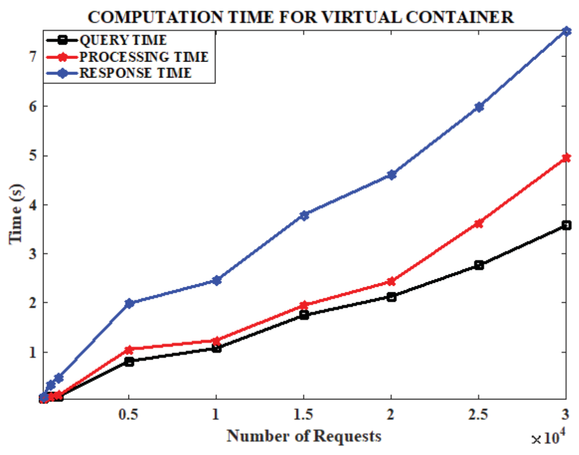

Fig. 2: Latency result of virtual computation

7. Conclusion

In this work, an efficient access control for IoT environment using blockchain network is presented. Before any user can access a data owner's information, the user has to be registered on the network and verified through the ECDSA algorithm, which is robust against attacks. Data owners enjoy a degree of freedom as they specify to the blockchain network which information of theirs users can gain full access. For sensitive data, we employ a virtual processing environment in which duplicates of the original data are made available and after processing are destroyed, though the results are given to the requesters. Smart contracts are also employed and bound to insensitive data, as this is still done to preserve the privacy of data owners. The latency of our computation environment is also checked, and the results indicate our system performs satisfactorily. In future works, we hope to explore more performance metrics and also delve more into the security aspect of our system.

Acknowledgment

This work was supported in part by the programs of International Science and Technology Cooperation and Exchange of Sichuan Province under Grant 2017HH0028 and Grant 2018HH0102, in part by the Fundamental Research Funds for the Central Universities under Grant ZYGX2016J152, Grant ZYGX2016J170 and Grant ZYGX2015J154, in part by the Key Research and Development Projects of High and New Technology Development and Industrialization of Sichuan Province under Grant 2017GZ0007, in part by the National Key Research and Development Program of China under Grant 2016QY04WW0802, Grant 2016QY04W0800, and Grant 2016QY04WW0803, and in part by the National Engineering Laboratory for Big data application on improving government governance capabilities.

References

[1] L. D. Xu, W. He, and S. Li, "Internet of things in industries: A survey," IEEE Trans. Ind. Informat., vol. 10, no. 4, pp. 2233-2243, 2014.
[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," IEEE Commun. Surveys Tuts., vol. 17, no. 4, pp. 2347-2376, 2015.
[3] C. J. DOrazio, K. K. R. Choo, and L. T. Yang, "Data exfiltration from internet of things devices: ios devices as case studies," IEEE Internet Things J., vol. 4, no. 2, pp. 524-535, 2017.
[4] E. Bertino and N. Islam, "Botnets and internet of things security," Computer, vol. 50, no. 2, pp. 76-79, Feb. 2017. [Online]. Available: doi.ieeecomputersociety.org/10.1109/MC.2017.62
[5] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," Computer Networks, vol. 76, pp. 146-164, 2015.
[6] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "Twenty security considerations for cloud-supported internet of things," IEEE Internet Things J., vol. 3, no. 3, pp. 269-284, June 2016.
[7] A. Ouaddah, H. Mousannif, A. A. Elkalam, and A. A. Ouahman, "Access control in the internet of things: Big challenges and new opportunities," Computer Networks, vol. 112, pp. 237-262, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128616303735
[8] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Rolebased access control models," Computer, vol. 29, no. 2, pp. 38-47, 1996.
[9] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," Computer, vol. 48, no. 2, pp. 85-88, 2015.
[10] R. S. Sandhu and P. Samarati, "Access control: principle and practice," IEEE Commun. Mag., vol. 32, no. 9, pp. 40-48, 1994.
[11] X. Sun and N. Ansari, "EdgeIoT: Mobile edge computing for the Internet of Things," IEEE Commun. Mag., vol. 54, no. 12, pp. 22-29, Dec. 2016.
[12] Ethereum smart contract platform. [Online]. Available:https://www.ethereum.org/
[13] Bitcoin - open source p2p money. [Online]. Available:https://bitcoin.org/en/
[14] M. Conoscenti, A. Vetra, and J. C. D. Martin, "Blockchain for the Internet of Things: A systematic literature review," in Proc. 13th Int. Symp. Internet Things Syst. Manag. Security (IOTSMS), 2016, pp. 1-6.
[15] D. Wilson and G. Ateniese, "From pretty good to great: Enhancing PGP using bitcoin and the blockchain," CoRR, vol. abs/1508.04868, pp. 368-375, Aug. 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1508.htmlWilsonA15
[16] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in Proc. IEEE Security Privacy Workshops (SPW), San Jose, CA, USA, 2015, pp. 180-184.
[17] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), March 2017, pp. 618-623.
[18] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[19] D. D. F. Maesa, P. Mori, and L. Ricci, "Blockchain based access control," in IFIP International Conference on Distributed Applications and Interoperable Systems. Springer, 2017, pp. 206-220
[20] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "Fairaccess: a new blockchain-based access control framework for the internet of things," Security and Communication Networks, vol. 9, no. 18, pp. 5943-5964, 2016.
[21] E.B. Sifah, Q. Xia, K.O.-B.O. Agyekum, S. Amofa, J. Gao, R. Chen, H. Xia, J.C. Gee, X. Du, and M. Guizani. 2018. Chain-based big data access control infrastructure. J. Supercomput. 80, (2018). DOI:https://doi.org/10.1007/s11227-018-2308-7