UEPF：A blockchain based Uniform Encoding and Parsing Framework in multi-cloud environments

Dehao Tao[1,4], Zhen Yang[2,*], Xuanmei Qin[3,4], Qi Li[1,4] Yongfeng Huang[3,4], and Yubo Luo[5]
[1]Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, 100084, China
[2] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, 100876, China
[3] Department of Electronic Engineering, Tsinghua University, Beijing, 100084, China
[4] Beijing National Research Center for Information Science and Technology (BNRist), Beijing 100084, China
[5]Department of Computer Science, University of North Carolina at Chapel Hill, North Carolina, 27599, USA
[Email: taodh19@mails.tsinghua.edu.cn, yangzhenyz@bupt.edu.cn, qxm17@mails.tsinghua.edu.cn, li-q20@mails.tsinghua.edu.cn, yfhuang@tsinghua.edu.cn, yubo@cs.unc.edu]
[*]Corresponding Author: Zhen Yang

Received March 24, 2021; revised July 3, 2021; accepted July 13, 2021; published August 31, 2021

Abstract

The emerging of cloud data sharing can create great values, especially in multi-cloud environments. However, “data island” between different cloud service providers (CSPs) has drawn trust problem in data sharing, causing contradictions with the increasing sharing need of cloud data users. And how to ensure the data value for both data owner and data user before sharing, is another challenge limiting massive data sharing in the multi-cloud environments. To solve the problems above, we propose a Uniform Encoding and Parsing Framework (UEPF) with blockchain to support trustworthy and valuable data sharing. We design namespace-based unique identifier pair to support data description corresponding with data in multi-cloud, and build a blockchain-based data encoding protocol to manage the metadata with identifier pair in the blockchain ledger. To share data in multi-cloud, we build a data parsing protocol with smart contract to query and get the sharing cloud data efficiently. We also build identifier updating protocol to satisfy the dynamicity of data, and data check protocol to ensure the validity of data. Theoretical analysis and experiment results show that UEPF is pretty efficient.

Keywords: cloud data sharing; data identifier; blockchain; smart contract; multi-cloud

http://doi.org/10.3837/tiis.2021.08.008 **ISSN : 1976-7277**

1. Introduction

In recent years, Cloud storage is widely used because it provides users with the convenience of large-scale data storage and sharing. As a successful profit model, more and more cloud service providers are emerging. However, cloud service providers generally do not support users to obtain data resources from other cloud service providers within a single cloud platform, which leads to the emergence of "data island" in the multi-cloud environments, and also brings inconvenience to users who want to make use of data storing in different clouds. On the other hand, with the advent of the era of big data, the value of data has continued to increase, and a paid sharing model has emerged for data sharing, in which, data owners hope that their shared data resources in the cloud can be seen by more data users. In a word, data owners and data users want to widely publish / obtain shared data resources, but cloud service providers cannot meet their needs because of "data island". This brings a big problem to data sharing in multi-cloud environments.

In order to meet the needs of users for data sharing in multi-cloud environments, a cloud data sharing framework based on search engine is proposed[12,13,14]. The URI of the data owner's shared data in the cloud is obtained by the open search engine[15,16] through the web crawler[17,18,19], and provided to the data users for retrieval. Data users locate and share data in the cloud through DNS[20,21] resolution to realize data sharing. This framework relies on the support of cloud service providers. But in the real environments, too many crawlers have led to a decline of cloud data service quality, and there is "data island" in the multi cloud environments, so many cloud service providers use anti-crawler technology, which leads to the failure of multi-cloud data sharing based on search engine.

Another cloud data sharing framework based on centralized server is proposed[23,24]. The shared data stored by data owners on different cloud platforms are collected and sorted into a unified format[22], stored in a centralized server and provided to data users. When data users want to share data, they can retrieve the unified format data to locate and obtain data. This framework is very efficient, but it faces the problem of single point failure. The failure of centralized server will cause data and economic losses to users.

In recent years, researchers have introduced blockchain to solve the single point of failure problem[1,2,9,36] and proposed a cloud data sharing framework based on blockchain[1,3,4,25]. When the data owner needs to share his own data, he submits a registration transaction in the blockchain network, and then the blockchain encodes a unique identifier for the data to refer to the data, and records the URI of the data. Data users obtain the URI of the data by retrieving and parsing the identifier of the data in the blockchain, and finally use the URI to address the shared data in the cloud. However, the existing methods still have shortcomings in the uniqueness of the identifier and the efficiency of identifier parsing.

This paper makes the following contributions:

1) We make definitions of the namespace and design the identifier pairs, which give enough consideration to both uniqueness of identifier and dynamicity of data.

2) We design efficient protocols based on the identifier pairs which reduce response time of UEPF and help data users to check the validity of data.

3) We analyze the efficiency of our proposed system and deploy an instance to evaluate its performance.

In the remainder of this paper, Section II introduces related work, including the security and privacy protection when sharing data, and different frameworks of sharing data. The design of UEPF system is elaborated in Section III. In Section IV, we evaluate the performance of our proposed system theoretically and experimentally. Finally, conclusions are drawn in Section V.

2. Related works

There are two types of related technologies for data sharing. One is concerned with the security and privacy protection when sharing data, and the other is concerned with the frameworks of sharing data. In the following, we will introduce them separately.

2.1 Security and privacy protection when sharing data

In [26], a dynamic attribute-based access control scheme is proposed, which sets a fine-grained valid time period for each attribute and reduces the waiting time of CSP caused by manual operations. In [27], a lightweight proxy re-encryption scheme is proposed, which builds the pre-encryption algorithm and designs a certificateless protocol that remove bilinear pair and have very high performance. Qin et al.[28] introduced Shamir secret sharing scheme and permissioned blockchain to eliminate the single point failure, and computed tokens cross domains to reduce communication and computation overhead on the data user side. Qin also made other work to realize access control based on different scenarios, such as [29,30].

There are some other related studies paying attention to internet of things and putting emphasis on privacy and security [31~35].

2.2 Frameworks of sharing data

Different frameworks are proposed for different scenarios, such as frameworks based on SE and DNS, based on Centralized Server, and based on Blockchain.

2.2.1 Frameworks based on SE and DNS

The most famous and widely used framework based on SE and DNS is [15], in which google is proposed. In [15], Brin et al. provided the first detailed public description of large-scale Web search engine. Google crawls and indexes webs efficiently and then make heavy use of use of the structure present in hypertext to produce much more satisfying search results. People can search webs by simply typing some words or a sentence, and then google returns indexes of webs as results to the user. People can get access to the webs by clicking the indexes.

However, due to the fact that so many cloud service providers use anti-crawler technology, frameworks based on search engine may not work on cloud data sharing.

2.2.2 Frameworks based on Centralized Server

Focusing on data sharing of a certain field, frameworks based on centralized server are proposed. In [23], QuerioCity, a platform to catalog, index and query highly heterogenous information coming from complex systems, is proposed. The data currency of QuerioCity is a dataset consisting of metadata, and thus, data from different sources can be used. Besides, an approach based on Semantic Web technologies is proposed to deal with the incremental and continuous integration of static and streaming data. Data from complex systems are collected and presented as indexes in the centralized QuerioCity platform. Users can get data through

these indexes.

However, there are always safety risks such as single point failure accompanying with centralized servers.

2.2.3 Frameworks based on Blockchain

In order to break through the limitation of shared data format, based on Namecoin and Bitcoin, Muneeb and Jude of Princeton University proposed a new system named Blockstack. It designs a system consists of several separate layers, and constructs a decentralized Internet by implementing a new public key infrastructure (PKI, non-blockchain based PKI systems can be learned through Keybase[5] and CONIKS[6]), which makes the identifier encoding and parsing system no longer limited by the data format. In order to improve the readability of the identifier, Blockstack adopts "name + namespace" as form of identifier. Ethereum[7] and others also use similar methods to encode readable identifiers. However, identifiers encoded in this way can't keep unique, so when users want to get a unique identifier, they have to try different names again and again, and finally end up hurting the readability of identifiers.

To improve the uniqueness and autonomy of the identifiers, PPK, an open organization of Beijing University of Posts and telecommunications, has open source a new system named open data index name (ODIN) [4] based on Bitcoin network. ODIN introduces the height of block where registration transactions recorded and serial number of registration transaction in block to form the prefix to ensure the uniqueness of the identifier. The user's naming of the data is used as the identifier suffix to improve the readability of the identifier. The logic positions of transactions used in identifiers help ODIN to keep identifiers unique, but a big problem occurs when users update their data as a logic position cannot represents multiple transactions. In ODIN system, it is difficult to manage the update operation of ODIN records, so a web server is introduced to facilitate users to obtain ODIN. Though the data stored in the web server can be verified using information from Bitcoin, the web server still has some other risks like single point failure.

Besides the problems discussed above, there are two more problems with these existing methods. The first, due to the limitation of Bitcoin, the write performance is very poor and the throughput is very low. The second, they pay little attention to validity of data besides a single signature verification, and that can never meet the need of users in big data era.

3. Design of UEPF System

To overcome the shortcomings of existing methods mentioned above, we propose a new system named UEPF. UEPF is based on consortium blockchain, which is much more efficient than public chains like Bitcoin. UEPF is designed with short-long identifier pairs, and it pays a lot of attention to validity of data. Identifiers are stored and shared with blockchain, and data can be stored in any storage entity. Here in this paper, we use cloud platforms as storage entities.

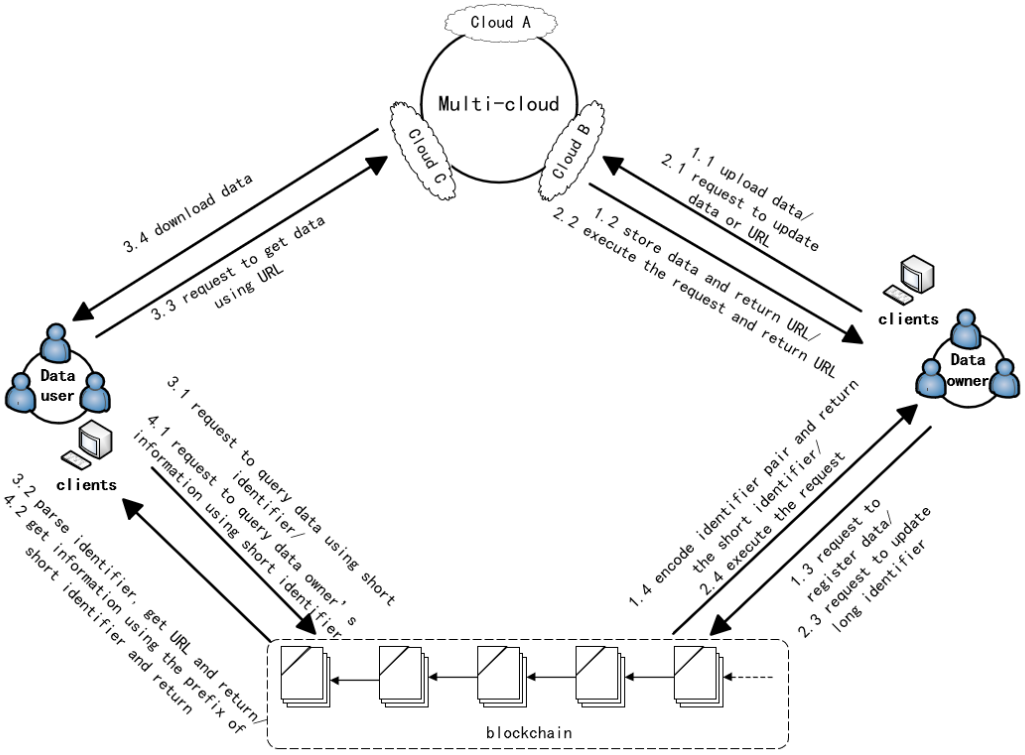


Fig. 1. Overview of UEPF system

3.1 Overview

The system architecture of UEPF is shown in **Fig. 1**, which includes data owners, data users, multi-cloud and blockchain network. The data owner uploads the data to the cloud and registers it in the blockchain network, who has the rights of modifying and deleting the data; the data user refers to the user who uses the data, and can query and obtain the data shared by others through the short identifier; the multi-cloud stores data, which can be shared after being registered on the blockchain network; the blockchain stores metadata and runs the encoding and parsing process of data identifier. The design of identifier pairs, the encoding protocol, the parsing protocol, identifier updating protocol and data check protocol will be detailed below.

Many existing methods such as ODIN and Blockstack chose Bitcoin as their underlying blockchain (although Blockstack is said that it can be applied above any blockchain, it has to make a huge change migrating to blockchains with smart contract in fact), but an obvious problem is that Bitcoin is too inefficient. The efficiency of Bitcoin cannot meet the need of a large data sharing system, and there are studies trying to solve it like Permacoin[10]. Besides, it is said there are selfish mining attacks[11] occur to Bitcoin. Here in our study, we introduce consortium blockchains as underlying blockchain. Compared to Bitcoin, consortium chains have greater throughput and shorter response time, and meanwhile its security is guaranteed. Here in this paper, we choose Hyperledger Fabric as our underlying blockchain.

One of the big differences between our proposed system and the existing methods is the location of metadata. Our system stores metadata in blockchain, and we make use of the functions of the blockchain and smart contract to achieve our system's functions. Meanwhile in the existing methods such as ODIN and Blockstack, the blockchains are used as a communication channel for announcing state changes, and the metadata are stored in external

entities. Regardless of the safety of external entities, they also bring another problem that the new nodes have to get a complete copy of metadata before they can join the systems, and it may lead to safety and efficiency issues.

3.2 Definitions of namespace

To ensure the uniqueness of identifiers, we make definitions of the namespace. From the perspective of users, we design "Global Domain" and "User Domain". Global Domain is the complete scope of multi-cloud and blockchain, and its basic unit is the single user. Each user corresponds to a User Domain. User Domain is the data storage scope of a single user on multi-cloud, and its basic unit is data. The prefix and suffix of the short identifier mentioned below will be unique in the "Global Domain" and "User Domain" respectively, so as to ensure the uniqueness of the complete short identifier.

3.3 Identifier Pairs

To make UEPF efficient, we designed short-long identifier pairs. The short identifiers are used to represent data, and the long identifiers are used to describe data. The long identifiers change when data is updated, and meanwhile the short identifiers keep unchanged.

A short identifier is the only identification of data and it means the short identifier shall be unique. In consortium chains, we have unique certifications for each user. We name the hash of certifications cert_hash, and choose the cert_hash as the prefix of short identifier, which is unique in Global Domain, and data_name named by data owner as the suffix. The data owner shall name his data with different names to make them unique in User Domain (no need to keep different with other users' data), and then the short identifier can keep unique in the whole namespace. Beside uniqueness, the prefix actually represents a user, and it means that we can store user information, such as certification and public key, using the prefix as key, which is helpful in the following protocols. And the suffix helps data owners to have their data marked as they want.

We organize all metadata into a long identifier, thus long identifier is also a tool of metadata management. Data owners can easily manage their data by managing metadata in the long identifier, and users can easily obtain information by getting metadata in long identifier. What's more, UEPF is a fundamental system which have the potential to exploit different functions to adapt to different environments, the long identifier makes that possible.

The advantages of identifier pairs are:

1) It is both unique and human-readable.
2) It satisfies the dynamicity of data.
3) The prefix of short identifier is related to owner's identity, and it avoids a lot of security issues such as counterfeiting.

Table 1. Identifier pair

	Purpose	**Metadata**
Short identifier	To identify data	cert_hash, data_name
Long identifier	To describe data	URL, transaction_hash, user_sig

Table 2. Introduction of basic metadata in UEPF

	Data flow	Purpose
cert_hash	Blockchain→Short identifier	To make short identifier unique in Global Domain, and to verify data owner's identity
data_name	Data owner →Short identifier	To make short identifier unique in User Domain, and to help data owners to mark their data as they want
URL	Cloud→Data owner→Long identifier	To locate data
transaction_hash	Blockchain→Long identifier	To locate registration transaction
user_sig	Data owner→Long identifier	To check data
user_pubkey	Data owner→User information	To check data
User_certificate	Blockchain→User information	To represent a user

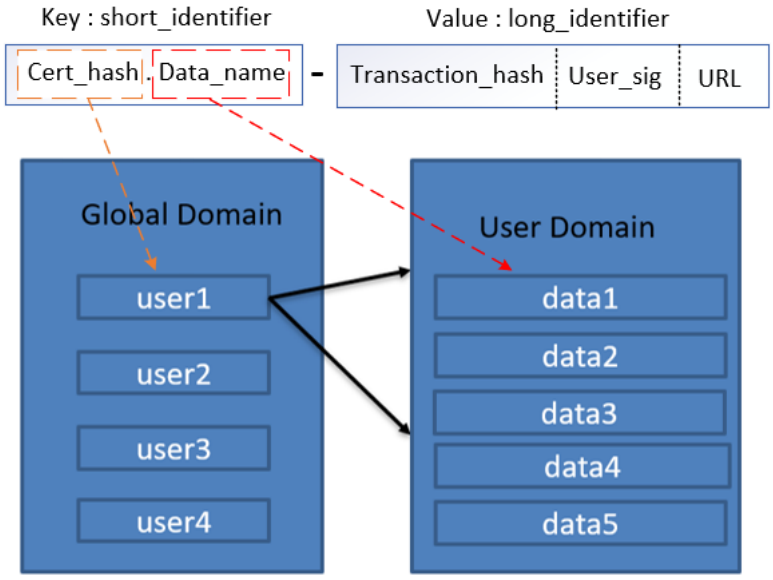


Fig. 2. Identifier pair and namespace

3.4 Identifier encoding protocol

To share data, data owner has to register his data in UEPF system. We design the identifier encoding protocol, which is also the process of data registration. There are four steps in this

protocol as the **Fig. 3** shows:

Step one, data owners upload their data to clouds by sending an upload request *Req_upload(data)*;

Step two, the clouds receive and store these data, and return the URL of data to the data owners by sending a response *Res_upload(URL)*;

Step three, when data owners decide to share their data with others, they send a registration request *Req_register(URL, user_sig, data_name)* to blockchain;

Step four, the blockchain generates a short identifier and a long identifier, and then returns the short identifier to the data owners by sending a response *Res_register(short_identifier)*.

The user_sig in the step three is data owner's digital signature of data, and the data_name is a name of data given by the data owner. The data_name should not be same with the data owner's other data_names (sequence numbers will be added at the end of data_name by UEPF if and only if the data_names are same).

In the step four, blockchain abstracts data owners' certification and then get hash of it as cert_hash. The cert_hash and the data_name make up a short identifier. After short identifier is generated, URL, transaction_hash, user_sig will make up a long identifier, in which the transaction_hash is generated by blockchain to represent the registration transaction. What's more, the long identifier is a tool to manage metadata, so if any metadata is needed in specific scenarios, it can be added into long identifier. A short identifier and a long identifier make up a key-value pair, and we can get metadata from long identifier using short identifier as key.

As a comparison, to register an identifier in Blockstack, users need to try again and again to see if the identifier they submit is unique over all the identifiers, and ODIN denies users the right to name their identifiers. The identifier encoding protocol we design focus on the advantages and abandons the disadvantages of Blockstack and ODIN.

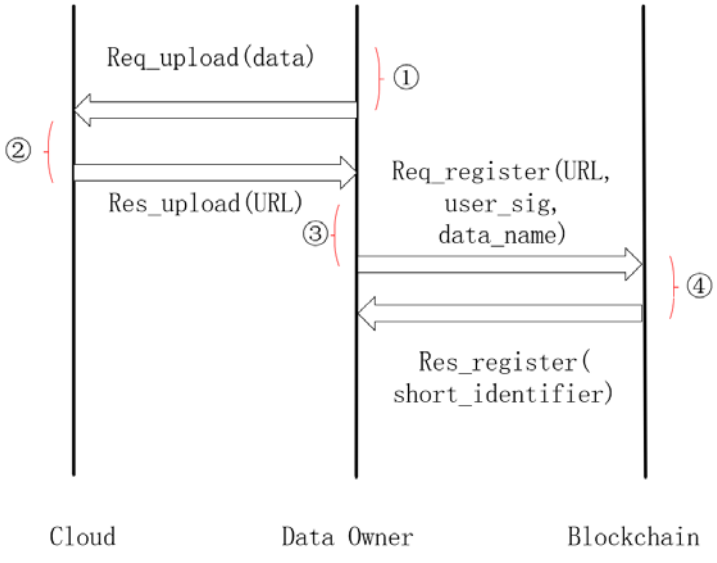


Fig. 3. Identifier encoding protocol

3.5 Identifier parsing protocol

The short identifiers are delivered to users and the long identifiers are invisible to users. We design the identifier parsing protocol for data users to query data with short identifiers. There are four steps in this protocol as the **Fig. 4** shows:

Step one, data users query data by sending a query request *Req_query(short_identifier)* to blockchain;

Step two, the blockchain executes the parsing process after receiving the request, and returns the metadata URL to the data users by sending a response *Res_query(URL)*;

Step three, data users send a query request *Req_query(URL)* to clouds to get data;

Step four, clouds send the data to the data users with a response *Res_query(data)*.

In the step two, the parsing process works like this: blockchain gets long identifiers from key-value pairs with the short identifiers as keys, and then gets the needed metadata from long identifiers. Here in this protocol the needed metadata is URL. Thus, data users can easily obtain data just by a single short identifier, and meanwhile, the parsing process provides a convenient way to implement expanded functions which need to get a certain metadata.

As mentioned above, Blockstack and ODIN store metadata in external entities, and thus their identifier parsing process runs outside the blockchain which may be easily attacked.

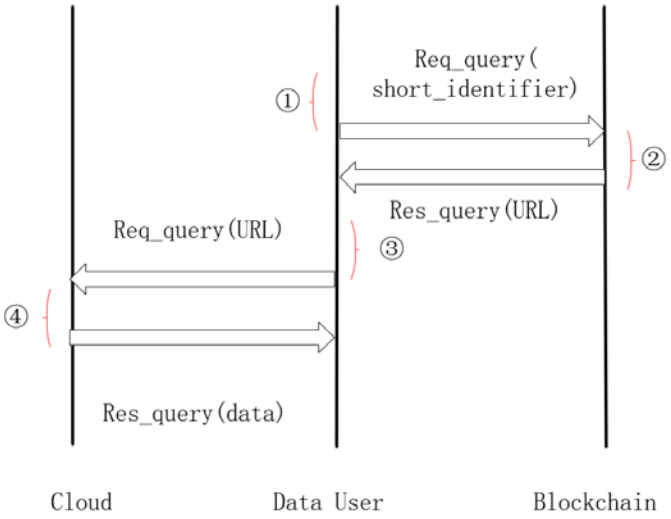


Fig. 4. Identifier parsing protocol

3.6 Identifier updating protocol

Data is not always static, and the content and description of data may change. We design the identifier updating protocol for data owners to update their data in both clouds and blockchain. There are four steps in this protocol as the **Fig. 5** shows:

Step one, data owners update their data in clouds by sending an updating request *Req_update(data)*, in which the parameter *data* is the new version of data.

Step two, the clouds receive and update these data, and return the URL of data to the data owners by sending a response *Res_update(URL)*, in which the parameter *URL* may be a new one;

Step three, the data owners also need to update the description of their data in blockchain, so they send an updating request *Req_update(URL, user_sig)* to blockchain, in which the *user_sig* is the digital signature of the updated data;

Step four, the blockchain regenerates the long identifier after an authentication, and then sends a response *Res_update(OK)* to the data owners.

In the step four, regenerating the long identifier means: blockchain gets long identifiers from key-value pairs with the short identifiers as keys, updates the metadata URL and user_sig in long identifiers, and then stores the key-value pairs with new values.

As mentioned above, a data owner's information, such as certification and public key, is stored in blockchain and the prefix of the short identifier is the key. Thus, we can easily get the data owner's certification and do authentication.

For the same reason with identifier parsing protocol, the identifier updating protocol we design is safer than existing methods. What's more, a node in Blockstack or ODIN need to read all blocks of Bitcoin to track the newest state, and meanwhile in our proposed system, the newest state is stored in blockchain which means even a new node can track the newest state without reading all the blocks or getting a copy of metadata from other nodes.

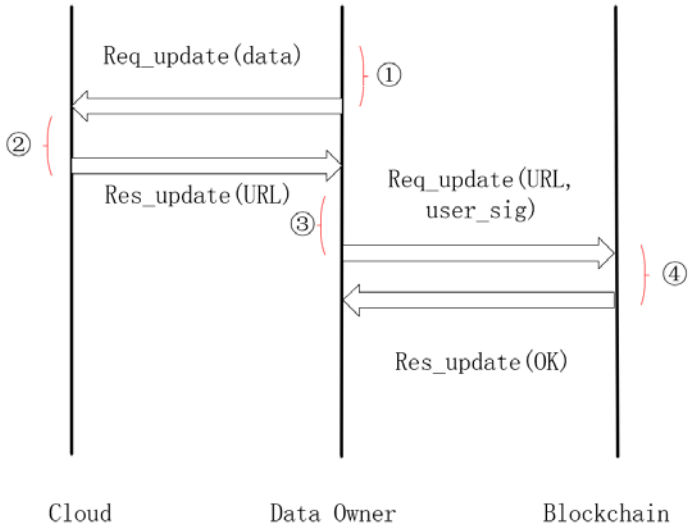


Fig. 5. Identifier updating protocol

3.7 Data check protocol

After downloading the data completely, checking data is a natural idea. We design the data check protocol for data users to see a) whether the data matches the short identifier; b) whether the data is actually provided by the data owner we thought to be. There are three steps in this protocol as the process (4) of **Fig. 6** shows:

Step one, data users send a data check request *Req_check(short_identifier)* to blockchain;

Step two, the blockchain gets the metadata *user_sig* and *user_pubkey* after receiving the request, and returns the metadata to the data users by sending a response *Res_check(user_sig, user_pubkey)*, in which the parameter *user_pubkey* is the data owner's public key;

Step three, data users use the *user_sig* and *user_pubkey* to verify the hash of data.

In step two, *user_sig* can be obtained from the long identifier, and *user_pubkey* can also be obtained from key-value pair using the prefix of short identifier as key.

If the verification in step three succeeds, we know that the data actually matches the short identifier as the *user_sig* is obtained with the short identifier, and the data is actually provided by the data owner we thought to be as the data owner's *user_pubkey* is used in the verification.

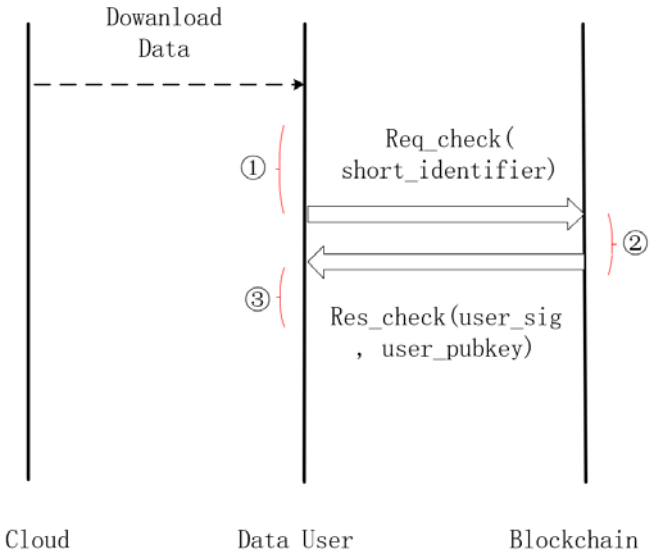


Fig. 6. Data check protocol

4. Performance Evaluation

In this section, we evaluate the performance of our proposed system theoretically and experimentally.

4.1 Theoretical comparison

Table 3. Comparisons among our UEPF system, Blockstack and ODIN

	UEPF	**Blockstack**	**ODIN**
Response time of registration	< 1s	About 10min	About 10min
Response time of querying	< 1s	< 1s	< 1s
Efficient protocols	√	√	×
Resource trace	√	×	×
Scalability	√	√	×

We compared our UEPF system with Blockstack in [3] and ODIN in [4], as these two systems already have successful practice and are accepted by many people, and the comparison is shown in **Table 3**. From **Table 3** we can see, for registering and querying an identifier, our UEPF system has best theoretical performance, and UEPF payed more attention to resource tracing than the other two systems as resource tracing raises more concern with users. They all perform well on response time of querying, but comparing to UEPF and Blockstack, protocols of ODIN are relatively inefficient, which also leads to ODIN's poor scalability.

4.2 Performance Analysis

To evaluate the performance of our UEPF system, we implemented it and the experimental environment is:

Blockchain: Fabric version 1.4.
Testing tool: caliper v0.2.0.
Virtual machine with 3.8GB memory and 1.60GHz CPU.

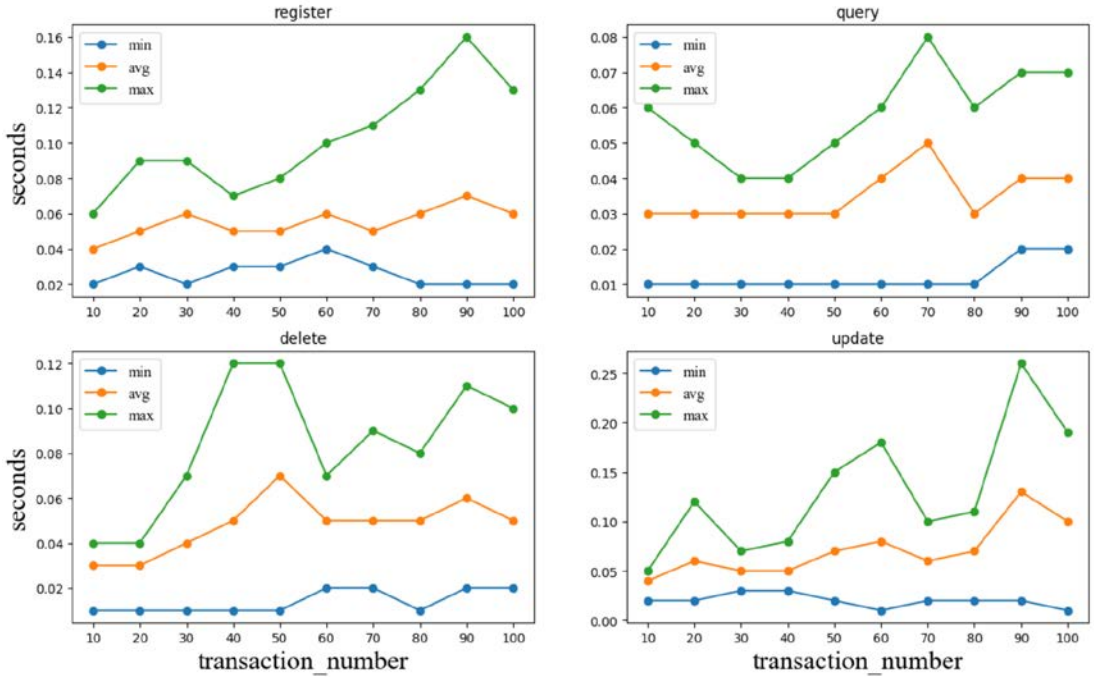


Fig. 7. Response time of operations: register, query, delete and update

We tested the performance of main functions of UEPF system and the results are shown in **Fig. 7**, in which the x-axis represents the transaction numbers of each function we sent to blockchain, and the y-axis represents the response time of each function. We tested maximal throughput (transaction number per second) of each function (102 for register, 119 for query, 121 for delete, 125 for update), and then let these functions run with different throughput. **Fig. 7** shows response time of different functions with different transaction number per second (i.e. throughput) where min represents minimum response time, avg represents average response time and max represents maximal response time. From **Fig. 7** we can see that the response time of functions increases when the throughput increases, and the query function performed best. The simulation results matched our expectations and theoretical analysis.
Besides, we also tested these functions with their maximal throughput for minutes, and the result shows UEPF system can perform well under lasting pressure.

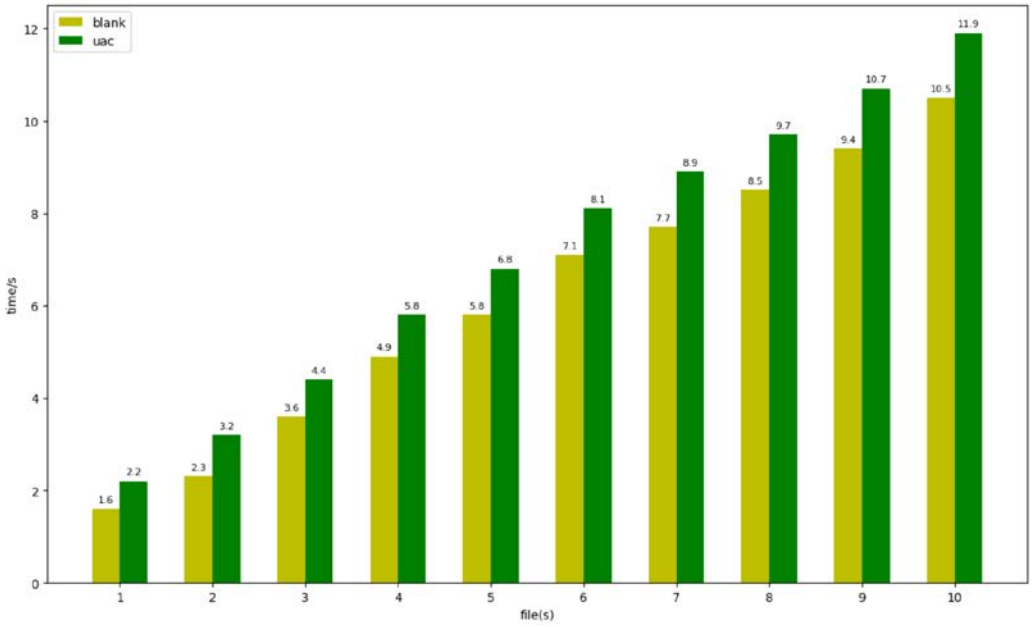


Fig. 8. Influence of UEPF on download latency

Some people believe that blockchains will greatly reduce efficiency, and thus we made an experiment to see if UEPF will reduce the efficiency of obtaining sharing data which are stored in four Ali cloud servers. In order to conveniently know about UEPF's performance, we set a blank control group which downloads data using a Linux command SCP with URL directly, and the other group gets URL from UEPF before downloading data with SCP command. To reduce the errors caused by network, we repeated this experiment 10 times and every file is 1 MB. The results are shown as **Fig. 8**, where the x-axis represents file numbers and the y-axis represents average download latency, we can see there are small difference between the two groups, and the difference grows slowly when more files are required to be downloaded. When we download one file of 1 MB, the latency that UEPF cause is 0.6s which is bigger than 1/4 of whole latency (2.2s), and when we download 10 files of 1 MB, the latency that UEPF cause is 1.4s which is smaller than 1/8 whole latency (11.9s). It means that when we obtain many files using UEPF, the latency of downloading files is the main part, and UEPF actually makes very small influence on obtaining sharing data.

What's more, since the data are stored in four Ali cloud servers which can be regarded as four clouds, we found that the multi-cloud environments bring no additional loss to the efficiency of our system comparing to single cloud environments. It can also be supported by our identifier parsing protocol which is always the same in both multi-cloud environments and single cloud environments.

5. Conclusion

Cloud storage has been widely used nowadays, but the emergence of "data island" disturbs data sharing between clouds. Protocols of previous methods cannot perform perfectly in multi-cloud environments. In this paper, we propose our own identifier encoding and parsing system UEPF. We make definitions of namespace and design identifier pairs which achieve the goals of being readable and easy-use, and we design protocols based on the identifier pairs. In section

IV, we analyze the advantages of UEPF system by comparing it with Blockstack in [3] and ODIN in [4], and then we deploy an instance to evaluate UEPF. The results show that protocols of UEPF are actually efficient and cause very small influence on latency.

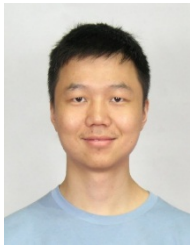
Acknowledgement

We thank the support of the National Natural Science Foundation of China (No. U1836204, No.U1936208, No.U1936216, No. 62002197).

References

[1] Namecoin. Website (CrossRef Link).
[2] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, A. Narayanan, “An empirical study of Namecoin and lessons for decentralized namespace design,” in *Proc. of the 14th Workshop on the Economics of Information Security*, pp. 1-21, June 22-23, 2015.
[3] Muneeb Ali, Jude Nelson, Ryan Shea, Michael J. Freedman, “Blockstack: A Global Naming and Storage System Secured by Blockchains,” in *Proc. of 2016 USENIX Annual Technical Conference*, pp. 181-194, June 22-24, 2016.
[4] Wang Jiye, Gao Lingchao, Dong Aiqiang, Guo Shaoyong, Chen Hui, Wei Xin, “Block Chain Based Data Security Sharing Network Architecture Research,” *Journal of Computer Research and Development*, 54(4), 742-749, 2017. Article (CrossRef Link)
[5] Keybase. Website (CrossRef Link)
[6] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, “CONIKS: bringing key transparency to end users,” in *Proc. of 24th USENIX Security Symposium (USENIX Security 15)*, pp. 383–398, 2015
[7] V. Buterin, “A next-generation smart contract and decentralized application platform,” *white paper*, 1-36, 2014.
[8] Deng, Zhiliang, et al., "Blockchain-based trusted electronic records preservation in cloud storage," *Comput. Mater. Continua*, 58.1, 135-151, 2019. Article (CrossRef Link)
[9] Du Zhen-Yuan, "Personal Data Security and Supervision in the Age of Large Data," *INTELLIGENT AUTOMATION AND SOFT COMPUTING*, 25.4, 847-853, 2019.
[10] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz, “Permacoin: Repurposing bitcoin work for data preservation,” in *Proc. of Security and Privacy (SP), 2014 IEEE Symposium on*, pp. 475–490, 2014. Article (CrossRef Link)
[11] I. Eyal and E. G. Sirer, “Majority is not enough: Bitcoin mining is vulnerable,” *CoRR*, abs/1311.0243, 2013.
[12] Pundt, Hardy, and Yaser Bishr, “Domain ontologies for data sharing–an example from environmental monitoring using field GIS,” *Computers & Geosciences*, 28.1, 95-102, 2002. Article (CrossRef Link)
[13] King, Gary, “An introduction to the dataverse network as an infrastructure for data sharing,” *Sociological Methods & Research*, 36.2, 173-199, 2007. Article (CrossRef Link)
[14] Grundy, Quinn, et al., “Data sharing practices of medicines related apps and the mobile ecosystem: traffic, content, and network analysis,” *bmj*, 364, 2019. Article (CrossRef Link)
[15] Brin, Sergey, and Lawrence Page, “The anatomy of a large-scale hypertextual web search engine,” *Computer networks and ISDN systems*, 30.1-7, 107-117, 1998. Article (CrossRef Link)
[16] Chen, Zhen-Lin, et al., “A high-speed search engine pLink 2 with systematic evaluation for proteome-scale identification of cross-linked peptides,” *Nature communications*, 10.1, 1-12, 2019. Article (CrossRef Link)
[17] Heydon, Allan, and Marc Najork, “Mercator: A scalable, extensible web crawler,” *World Wide Web*, 2.4, 219-229, 1999. Article (CrossRef Link)

[18] Thelwall, Mike, "A web crawler design for data mining," *Journal of Information Science*, 27.5, 319-325, 2001. Article (CrossRef Link)
[19] Farag, Mohamed MG, Sunshin Lee, and Edward A. Fox, "Focused crawler for events," *International Journal on Digital Libraries*, 19.1, 3-19, 2018. Article (CrossRef Link)
[20] Mockapetris, Paul, and Kevin J. Dunlap, "Development of the domain name system," *ACM SIGCOMM Computer Communication Review*, 18.4, 123-133, 1988. Article (CrossRef Link)
[21] Kim, Tae Hyun, and Douglas Reeves, "A survey of domain name system vulnerabilities and attacks," *Journal of Surveillance, Security and Safety*, 1.1, 34-60, 2020. Article (CrossRef Link)
[22] Chang, Yue Shan, et al., "Big data platform for air quality analysis and prediction," in *Proc. of 2018 27th Wireless and Optical Communication Conference (WOCC)*, IEEE, 2018. Article (CrossRef Link)
[23] Lopez, Vanessa, et al., "Queriocity: A linked data platform for urban information management," in *Proc. of International Semantic Web Conference*, Springer, Berlin, Heidelberg, pp. 148-163, 2012. Article (CrossRef Link)
[24] Park, Kyounghyun, Minh Chau Nguyen, and Heesun Won, "Web-based collaborative big data analytics on big data as a service platform," in *Proc. of 2015 17th international conference on advanced communication technology (icact)*, IEEE, 2015. Article (CrossRef Link)
[25] Xu, Xiwei, et al., "A taxonomy of blockchain-based systems for architecture design," in *Proc. of 2017 IEEE international conference on software architecture (ICSA)*, IEEE, 2017. Article (CrossRef Link)
[26] X. Qin, Y. Huang, Z. Yang and X. Li, "An access control scheme with fine-grained time constrained attributes based on smart contract and trapdoor," in *Proc. of 2019 26th International Conference on Telecommunications (ICT)*, Hanoi, Vietnam, pp. 249-253, 2019. Article (CrossRef Link)
[27] Qian, Xin, et al., "A No-Pairing Proxy Re-Encryption Scheme for Data Sharing in Untrusted Cloud," in *Proc. of International Conference on Artificial Intelligence and Security*, Springer, Cham, pp. 85-96, 2019. Article (CrossRef Link)
[28] Qin, Xuanmei, et al., "A Blockchain-based access control scheme with multiple attribute authorities for secure cloud data sharing," *Journal of Systems Architecture*, 112, 101854, 2020. Article (CrossRef Link)
[29] Qin, X., Huang, Y. & Li, X, "An ECC-based access control scheme with lightweight decryption and conditional authentication for data sharing in vehicular networks," *Soft Computing*, 24, 18881–18891, 2020. Article (CrossRef Link)
[30] Qin, Xuanmei, et al., "LBAC: A Lightweight Blockchain-based Access Control Scheme for the Internet of Things," *Information Sciences*, 554, 222-235, 2021. Article (CrossRef Link)
[31] Le Nguyen, Bao, et al., "Privacy preserving blockchain technique to achieve secure and reliable sharing of IoT data," *CMC-COMPUTERS MATERIALS & CONTINUA*, 65.1, 87-107, 2020. Article (CrossRef Link)
[32] Yang, Zhen, et al., "Protecting personal sensitive data security in the cloud with blockchain," *Advances in Computers*, 120, 195-231, 2021. Article (CrossRef Link)
[33] Yang, Zhen, et al., "Efficient secure data provenance scheme in multimedia outsourcing and sharing," *Computers, Materials & Continua*, 56.1, 1-17, 2018.
[34] Li, Jun, et al., "A distributed privacy preservation approach for big data in public health emergencies using smart contract and SGX," *CMC-COMPUTERS MATERIALS & CONTINUA*, 65.1, 723-741, 2020. Article (CrossRef Link)
[35] Bordel, Borja, et al., "Trust provision in the internet of things using transversal blockchain networks," *INTELLIGENT AUTOMATION AND SOFT COMPUTING*, 25.1, 155-170, 2019.



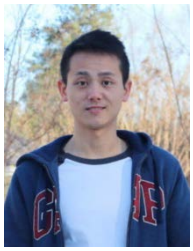
Dehao Tao. He is currently a master student in Institute for Network Sciences and Cyberspace, Tsinghua University. His research interests include blockchain technology and data security.

Zhen Yang, an assistant professor at the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China. He obtained his B.E. degree and Ph.D. degree in Electronic Information Science and Technology from Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2012 and 2018 respectively. His research interests are data security, multimedia security and privacy protection in the cloud and IoT.

Xuanmei Qin. She is currently pursuing a Ph.D. degree in the Department of Electronic Engineering, Tsinghua University. Her research interests include blockchain technology, data security, and access control.

Qi Li. She is currently a master student in Institute for Network Sciences and Cyberspace, Tsinghua University. Her research interests include blockchain technology and data security.

Yongfeng Huang, is a professor at the Department of Electronic Engineering, Tsinghua University, Beijing, China. His research interests include Cloud Computing, multimedia network and next generation Internet. He has published 5 books and over 50 research papers on computer network and multimedia communication.



Yubo Luo, born in 1991. PhD student. His main research interests include energy constrained system, machine learning, and Internet-of-Things (IoT).