Blockchain-Based Secure Data Provenance for Cloud Storage

Yuan Zhang[1(✉)], Xiaodong Lin[2], and Chunxiang Xu[1]

[1] University of Electronic Science and Technology of China, Chengdu, China
zy_loye@126.com
[2] Wilfrid Laurier University, Waterloo, Canada

Abstract. Data provenance, which records the history of the ownership and process of a document during its lifecycle, is essential for the success of cloud storage systems. However, it also inevitably incurs some challenging security and privacy issues. In this paper, to address these challenging issues, we present an efficient and secure data provenance scheme and realize it in a system called ESP. ESP is characterized by employing a blockchain-based provenance record chain and can provide a secure and efficient system for data outsourcing, where the correctness, integrity, and timeliness of provenance records can be ensured. Furthermore, we introduce a concept of window of latching (WoL) to assess the practicality of secure provenance schemes. We analyze the security of ESP and evaluate the performance of ESP via implementation, which shows WoL of ESP is short and demonstrates ESP is secure and practical.

Keywords: Secure provenance · Blockchain · Cloud storage
Digital investigation

1 Introduction

In today's big data era, digital data are explosively generated and people are increasingly outsourcing their data to cloud servers so as to enjoy efficient data management services without bearing heavy local storage costs [1]. While cloud storage is very helpful in many aspects, public cloud service could put user information in danger [2]. Meanwhile, it is also very important to keep track of what happens to these data throughout its lifecycle (from creation to ownership transfer to destruction or deletion), such as its ownership and custodial history as well as how it has been accessed by its users, which is also known as data provenance [3,4]. For example, in a digital investigation, digital evidences must be strictly secured and clearly documented about its ownership transfer as well as how it was handled during its lifecycle. It is usual that the defendant challenges the authenticity of a digital evidence during the trial. The most common types of

Yuan Zhang—This work is supported by National Key R&D Program of China (No. 2017YFB0802000) and China Scholarship Council.

digital evidence are hard disk images, and the defendant may question the hard disk image that investigators are working on and presented in the courtroom is not the same one acquired from the hard disk found at the crime scene.

In the past decades, many security mechanisms have been developed to ensure security and privacy of sensitive information, as well as achieve accountability and auditability through data access logging or audit trails [5], such as logging activities on data creation, modification, and access. Much of the focus has been on protecting digitally stored information from unauthorized access or modification. However, despite extensive research on information security and privacy, little attention has been paid to securing provenance information and providing assurance that a data document is trustworthy. It is worth mentioning that as the current best of practice, log files are also protected from tampering. In the banking industry, any activities, such as bank transfers, can only be recorded by creating a new log, and past logs cannot be modified or deleted for security reasons. Nevertheless, another important question still needs to be answered about whether the provenance information can be trusted to make sure that the corresponding data document is a trusted one after a series of user activities on the document which have been detailed in the provenance information.

Unlike the traditional file access auditing where file access activities are logged, provenance information contains the ownership history of data documents as well as activities occurred on them by their owners or users. Furthermore, such information is organized in chronological order during the lifecycles of the data documents, and allows accesses and activities of data documents to be tracked. As a result, it not only improves accountability and reliability [6], but also meets the requirements of emerging applications, such as maintaining the digital chain of custody in a digital forensics investigation [7–9], as well as regulatory compliance requirements and industry standards, such as HIPAA [10].

Actually, provenance information is not useful if it cannot be trusted, it is inadvisable to trust provenance information without proper protection. For example, whenever a Microsoft office document is created, Microsoft office automatically embeds an author name into the document. It has been proved very useful to solve crime. A good example of it is the BTK killer case [11]. Nevertheless, the assigned name for the document can be easily changed. This problem is further exacerbated by the fact that data documents and the corresponding provenance records are outsourced to the cloud storage which cannot be fully trusted, since the data documents and the provenance information would not be physically owned by data owners and they are transmitted over an insecure network [12]. Hence, it is crucial to ensure provenance information security.

Provenance information can be modeled as a sequence of records (each known as provenance record) which present details about how a data document was processed at every stage of its lifecycle. To guarantee the security of provenance information, similar to protection of data document itself, we can protect individual records of provenance information from unauthorized use or modification. However, the integration of security mechanisms in current clouds to ensure the security of the individual records would incur additional costs on both the cloud provider and users. As such, it is vital to point out what affects the practicality of secure provenance schemes and define how to evaluate the practicality.

In existing schemes [3,13], an identity manager is introduced to efficiently secure the outsourced data provence information, where any operation on the data performed by a user is required to be authorized by the identity manager. By doing so, the provenance records actually reflect the state information on the data during its lifecycle. Nevertheless, such mechanism suffers from a strong assumption that the identity manager is honest and reliable. Once the identity manager is compromised, the security of these schemes are broken: if the cloud server colludes with the identity manager, the outsourced provenance records can be modified without detection. In reality, compromising the identity manager is feasible for adversaries, since an adversary (e.g., a malicious cloud server) can perpetually incentivize the identity manager to deviate from the prescribed scheme over a long period of time. Furthermore, existing schemes do not consider the timeliness of provenance records.

In this paper, we propose an efficient and secure data provenance scheme for cloud storage systems called ESP. ESP is secure against provenance record forgery, removal, modification attacks. The security of ESP is guaranteed in the case that the identity manager is compromised, even if the malicious cloud server colludes with it. The key technique behind ESP is the blockchain-based currencies which provide a secure way to conduct transactions without a central authority. In ESP, each provenance record is integrated into a transaction on the blockchain, and all provenance records corresponding to one data form a record chain such that any one of them is corrupted, the chain is broken. With the integration of a provenance record in a transaction on the blockchain, the provenance record is time-stamped, and the time when the provenance record was generated can be extracted. As such, the provenance records in ESP not only keep track of what happened to the data, but also reflect when the data was processed. Detailed security analysis proves that ESP is secure against various attacks, even if the malicious cloud server/user colludes with the identity manager. Moreover, we introduce a concept of window of latching (WoL) which is one of the most important factors that affects the practicality of secure provenance schemes. We implement ESP and evaluate its performance. Experiment results show that WoL of ESP is short and is acceptable in reality, which demonstrates ESP is efficient and practical. Specifically, the contributions of this work are as follows:

- We formalize a model of data provenance, where the lifecycle of data documents is formally formulated. We also introduce the concept of WoL to measure the practicality of secure provenance schemes.
- We propose an efficient and secure data provenance scheme (ESP) for cloud storage systems. ESP employs a provenance record chain which is built on blockchain-based currencies, e.g., Ethereum, this ensures the secure auditability of provenance records in terms of correctness, integrity, and timeliness, even if the identity manager is compromised.

- We present security analysis to demonstrate that ESP can be secure and robust from various attacks. We implement ESP and conduct a comprehensive performance evaluation, which shows that ESP is highly efficient and practical.

2 Related Work

Data provenance provides sufficient information about target data that what happened to the data from creation to destruction. As we are moving into the age of big data where digital data are explosively generated nowadays and most of data are managed via the Internet with the aid of cloud systems, data provenance is pretty important to digital investigations [14,15]. Once a dispute arises in outsourced data, provenances serve as the most vital evidences for post investigation.

Lynch [16] first pointed out the need for trust and provenance in information retrieval. Hasan et al. [2] first defined the problem of secure provenance and argued that it is of vital importance in practice. Prior work on secure data provenance in cloud storage systems was proposed by Lu et al. [3], where the basic security requirements were first enumerated, i.e., unforgeability and conditional privacy preservation. The unforgeability ensures that a provenance record reflects the corresponding state of data, even if the data and the provenance record are outsourced to an untrusted environment; The conditional privacy preservation guarantees that only an authenticated entity can reveal the real identity recorded in the provenance, while anyone else cannot.

Following the Lu et al.'s work, several secure data provenance schemes have been proposed [13,17]. These schemes mainly focus on enhancing the functionality of secure provenance for cloud storage systems. However, in existing schemes, a trusted identity manager is introduced to secure the provenance records. If the identity manager is compromised, the security would be broken. Moreover, in existing schemes, lifecycles of data documents in cloud storage are not considered, the timeliness of provenance records has not been explored, and how to measure the practicality of secure provenance schemes is also not well investigated. In this paper, we propose ESP, an efficient and secure data provenance scheme that ensures the correctness, integrity, and timeliness of provenance records against the malicious identity manager.

3 Preliminaries

3.1 Basic Cryptographic Primitives

Secure Hash Function. A secure hash function h has the following three properties: h can take a message of arbitrary length as input, and output a short fixed-size message digest; Given x, it is easy to compute $h(x) = y$. However, given y, it is hard to calculate $h^{-1}(y) = x$; Given x, it is computationally infeasible to find $x' \neq x$ such that $h(x') = h(x)$.

Bilinear Maps. Let G be an additive group and G_T be a multiplicative group, they have the same prime order p. A bilinear map e: $G \times G \rightarrow G_T$ has the following properties. Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G$, $a, b \in Z_p^*$; Non-degeneracy: for $P, Q \in G$ and $P \neq Q$, $e(P, Q) \neq 1$; e can be computed efficiently.

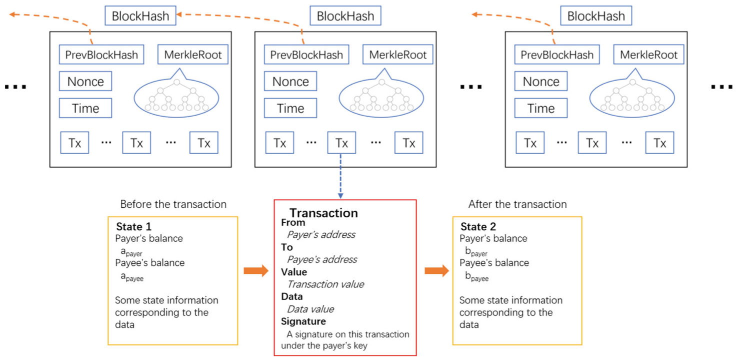


Fig. 1. A simplified Ethereum blockchain

3.2 Blockchain

We defer a brief introduction to the blockchain technique to **Appendix A. Blockchain**. We construct ESP on the Ethereum blockchain, since Ethereum is more expressive than other blockchain-based currencies. We show a simplified Ethereum blockchain in Fig. 1, where Tx denotes the transaction, BlockHash denotes the hash value of current block, PrevBlockHash denotes the hash value of the last block, Time denotes the time when the block is chained to the blockchain, and MerkleRoot denotes the root value of a Merkle hash tree formed by all transactions recorded in the block. The value token of the Ethereum blockchain is called Ethers.

In Ethereum, the state is made up of objects called "account". Generally, there are two types of accounts: externally owned accounts and contract accounts. Externally owned accounts are controlled by private keys and can conduct a transaction. Contract accounts are controlled by their contract code. For a transaction between two external owned accounts, if it is recorded into the blockchain, the balances of these two accounts are updated, where the user who conducts the transaction can set the "data" field to be any binary data she/he chooses. Therefore, Ethereum blockchain ensures the timeliness of the data state: when a payer transfers Ethers to a payee, a string Δ can be set to be the Data value of the transaction; After the block containing this transaction is added into the blockchain, Δ is recorded, which means that Δ is generated no later than the time when the block is chained to the block.

4 Models and Design Goals

4.1 A Model of Data Provenance

Lifecycle of a Data Document and Its Users. In this work, a data document lifecycle is viewed as a sequence of stages from creation to modification, destruction, and ownership transfer, which is shown in Fig. 2. After a data document has been created, it may go through many stages due to the document modification or ownership transfer. Finally, a document may be destructed or deleted, becoming unavailable to the users. Hence, an individual state of a data document can be uniquely identified by its content and owner, and can be represented as $St_i = \mathcal{H}(F_i, O_i)$, where St_i stands for a state where a document has been at, F_i means the content of the document at state of St_i, O_i is the owner of the document at state of St_i, and $\mathcal{H}$ stands for a secure hash function.

During the lifetime of a data document, users can play different roles in it, and can be classified into four types in general: *creator*, *owner*, *editor*, and *viewer*.

Creator: showing a user is the creator of a data document.

Owner: showing a user is the owner of the data document. By taking ownership of a data document, the user can assign other users permissions to the data document, including editing and viewing a data document, and transferring ownership of a data document. By default, the creator of a data document is also the owner, but document ownership can be transferred to another user by its current owner.

Editor: identifying a user is able to edit the data document.

Viewer: identifying a user can only view the data document.

Documents can have many editors and viewers, but only one creator during its lifetime and one owner at a time. In addition to the four aforementioned types of users, we assume there exists an *auditor* who can verify the validity and trustworthiness of any provenance information but without any knowledge of the user's identity who generates each individual provenance record [2,3].

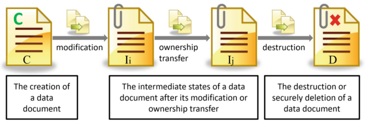

C
modification
Ii
ownership transfer
Ij
destruction
D
The creation of a data document
The intermediate states of a data document after its modification or ownership transfer
The destruction or securely deletion of a data document

Fig. 2. Document lifecycle

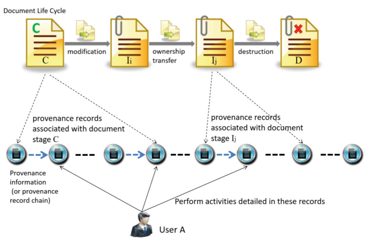

Document Life Cycle
C
modification
Ii
ownership transfer
Ij
destruction
D
provenance records associated with document stage C
provenance records associated with document stage Ij
Provenance information (or provenance record chain)
Perform activities detailed in these records
User A

Fig. 3. Provenance model

Provenance Model. As shown in Fig. 3, in data provenance, provenance information is organized into a chain in chronological order, where each chain item represents a provenance record which details how a data document was processed at every stage of its lifecycle. Each provenance record is also associated with a specific document stage, and a legitimate user (e.g., editors) may perform many actions on data documents. A typical provenance record consists of a specially formatted data block that contains information related to how a data document is processed at a time as well as its ownership information, which usually can be classified into two types: *Essential provenance data (EPD)*: information related to activities performed on the data document; *Nonessential provenance data (NPD)*: security overhead which has been generated by security mechanisms that are used to protect provenance information.

Measure the Practicality of Secure Provenance. With the provenance model, we introduce a concept of window of latching (WoL) to evaluate the practicality of secure provenance schemes.

Definition 1. Window of latching (WoL) means the time-interval between two successive provenance records that are accepted and published. The shorter WoL, the more practical the secure provenance scheme is.

4.2 Threat Model

In our threat model, we mainly consider the following security and privacy threats against data provenance.

Provenance Record Forgery Attack. A malicious user may collude with others to forge a valid provenance record in terms of the record's content and its timeliness.

Provenance Record Removal Attack. A malicious user colludes with others to remove one or several existing provenance records that have been generated due to the operations performed on data documents.

Modification Attack. Similar to the two above threats, a malicious user who may collude with others may attempt to tamper with provenance information by modifying the provenance records.

Repudiation Attack. A malicious user may deny that he performed an action on a data document.

Privacy Violation. Privacy violation refers to the attack that the identity of user who generates a provenance record is leaked out. Recall that in secure provenance schemes [3,13], only conditional privacy preservation can be ensured, where the identity manager has the ability to reveal the real identity recorded in the provenance record, while anyone else cannot.

4.3 System Model and Design Goals

As shown in Fig. 4, there are four different entities in ESP: users, an authenticated server, a cloud server, and an independent auditor. The authenticated server is used to authorize the users and control who can access the data. It also assists users in preserving their identity against adversaries. Data and the corresponding provenance information are generated by the users, and are stored in the cloud server. The auditor can check the provenance records' validity including their correctness, integrity, and timeliness.

Different from existing schemes [3,13], the authenticated server is not fully trusted by others, and thereby should be responsible for all its authorizations. As long as the authenticated server remains inaccessible to adversaries, we ensure both the security and privacy preservation. If both the authenticated server and cloud storage server are compromised, we retain the security guarantees on the provenance records in existing schemes.

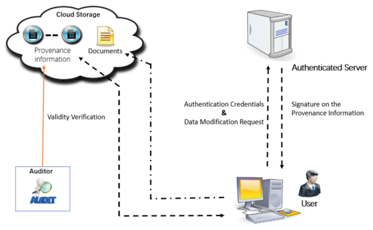


Fig. 4. The system model of ESP

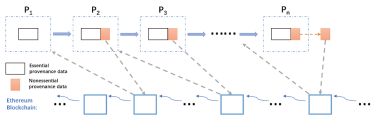


Fig. 5. Blockchain-based provenance record chain

Aiming at the above security challenges, our design goal is to develop an efficient and secure data provenance scheme in the cloud storage system. Specifically, the following goals should be achieved.

Security and Privacy Preservation. The validity of provenance records can be audited by authorized auditors with resistance against various attacks. The conditional privacy can be ensured.

Efficiency. It should efficiently work without introducing too much extra storage space caused by introducing security mechanisms, for example, digital signatures and cryptographic hashes. Its WoL should be as short as possible such that it can be applied in reality.

5 The Proposed Scheme

5.1 Overview

ESP consists of three parts: *system setup*, *secure provenance generation*, and *secure provenance verification.*

In the first part, the authenticated server assigns a human-memorisable password to each user, and maintains a list that records the assigned passwords and the corresponding identities. With the list, the authenticated server can authenticate each user securely and efficiently.

When a user wants to process a document, she/he needs to be authorized by the authenticated server. Then the authenticated server assists the user in generating a provenance record, this enables the user to prove herself/himself to the cloud server that she/he is qualified to process the document. Each provenance record is integrated into a transaction on the blockchain, where the user transfers a service charge to the authenticated server. This also time-stamps the provenance record.

To achieve the security, all provenance records are chained together with the aid of the Ethereum blockchain. This is shown in Fig. 5, where the provenance record chain is indicated by dashed gray lines. Assume that there currently are n provenance records, $P_1, P_2, \cdots, P_n$, each of them stands for a state of the underlying document at the corresponding stage during its liftcycle as modeled in Sect. 4.1. They are chained together as follows: from the second provenance record P_2, each record contains a data field that points to a block on the Ethereum blockchain, this block relates to the last provenance record. Each record is appended to the last one until it reaches the last one of the current provenance information, P_n. Finally, the last record will be signed by the authenticated server and the signature becomes the tail of the provenance record chain. In this case, if any existing record is modified or removed, the provenance record chain is broken. The computational costs to verify provenance records mainly depend on the hashing operation along with one signature verification for the last element or the tail of the provenance record chain. As a result, the verification is very fast.

5.2 Description of ESP

A set of user $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, ...\}$, an authenticated server $\mathcal{AS}$, a cloud storage server $\mathcal{C}$, and a third-party auditor $\mathcal{A}$ are involved in ESP.

System Setup:

- With the security parameter ℓ, the system parameters $\{p, G, G_T, P, e, E(\cdot), h, H\}$ are determined, where G is an additive group whose generator is P, $e : G \times G \rightarrow G_T$, G and G_T have the same prime order p, $E(\cdot)$ is a secure symmetric encryption algorithm, $h : \{0,1\}^* \rightarrow Z_p^*$, and $H : \{0,1\}^* \rightarrow G$.
- $\mathcal{AS}$ randomly chooses $s \in Z_p^*$, and computes $P_{pub} = sP$, and $k = h(s)$.
- $\mathcal{AS}$'s secret keys are (s, k), the corresponding public key is P_{pub}.
- For each $\mathcal{U}_i \in \mathcal{U}$ with identifier ID_i, $\mathcal{AS}$ assigns a human-memorisable password pwd_i to her/him, and stores (ID_i, pwd_i) locally.

Secure Provenance Generation:
Once a user $\mathcal{U}_i$ processes a document at $\mathcal{C}$ and generates a provenance record P_j, she/he will request $\mathcal{AS}$ to generate secure provenance on the document process.

Phase 1: With the identifier ID_i and password pwd_i, $\mathcal{U}_i$ makes mutual authentication with $\mathcal{AS}$ to establish a secure channel as follows.

- $\mathcal{U}_i$ randomly chooses $r_1, a \in Z_p^*$, and fetches the current timestamp ct.
- $\mathcal{U}_i$ computes C_1, C_2, where $C_1 = r_1 P$, $C_2 = E_k(ID_i||pwd_i||aP||ct)$, and $k = r_1 P_{pub}$.
- $\mathcal{U}_i$ sends (C_1, C_2) to $\mathcal{AS}$.
- After receiving (C_1, C_2), $\mathcal{AS}$ computes $k = sC_1 = sr_1 P = r_1 P_{pub}$, and extracts $ID_i||pwd_i||aP||ct$ from C_2 with k.
- $\mathcal{AS}$ checks the validity of the timestamp ct to resist the replay attack.
- $\mathcal{AS}$ authenticates $\mathcal{U}_i$ by checking whether (ID_i, pwd_i) is stored locally.
- $\mathcal{AS}$ randomly chooses $b \in Z_p^*$ and computes $sk = b(aP)$ as the session key.
- $\mathcal{AS}$ calculates $\mathcal{U}_i$'s pseudonym $PID_j = E_k(ID_i||ct||b)$, C_3, and C_4, where $C_3 = bP$, $C_4 = E_{sk}(ID_i||aP||bP||ct||PID_j)$.
- $\mathcal{AS}$ sends (C_3, C_4) to $\mathcal{U}_i$.
- With (C_3, C_4), $\mathcal{U}_i$ computes the session key $sk = aC_3 = abP$.
- $\mathcal{U}_i$ extracts $ID_i||aP||bP||ct||PID_j$ from C_4 with sk.
- $\mathcal{U}_i$ authenticates $\mathcal{AS}$ and confirms the correctness of sk by verifying the correctness of $ID_i||ct||aP||bP$.
- Since the session key sk is shared between $\mathcal{U}_i$ and $\mathcal{AS}$, a secure channel between them is established for secure provenance.

Different roles of $\mathcal{U}_i$ require different execution between $\mathcal{U}_i$ and $\mathcal{AS}$.

Creator: $\mathcal{U}_i$ creates a new document
If $\mathcal{U}_i$ creates a new document F, i.e., the provenance record P_j is P_1, where $P_1 = h(F_1||ID_i)$ and F_1 denotes the content of the document at the first stage, she/he requests a secure provenance from $\mathcal{AS}$ as follows.

- $\mathcal{U}_i$ sends P_1 to $\mathcal{AS}$ via the secure channel.
- $\mathcal{AS}$ extracts PID_1 from local storage (i.e., $j = 1$).
- $\mathcal{AS}$ generates a signature on P_1 and PID_1 as $\sigma_{T_1} = sH(P_1||PID_1)$, and sends σ_{T_1} to $\mathcal{U}_i$.
- $\mathcal{U}_i$ verifies $e(\sigma_{T_1}, P) \overset{?}{=} e(H(P_1||PID_1), P_{pub})$. If the verification fails, reject.
- $\mathcal{U}_i$ creates a transaction Tx_1 shown in Fig. 6, where $\mathcal{U}_i$ transfers service charge to the $\mathcal{AS}$'s account, and the data field of the transaction is set to $h(h(P_1||PID_1)||\sigma_{T_1})$.
- After Tx_1 is recorded into the Ethereum blockchain, $(P_1||PID_1||Bl_1, \sigma_{T_1})$ is sent to $\mathcal{C}$, and is published as the provenance record.

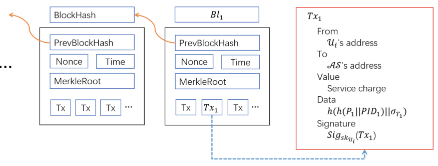


Fig. 6. The transaction conducted by the creator

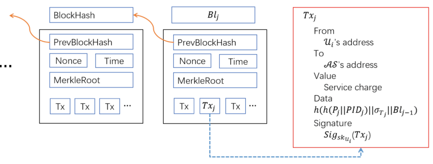


Fig. 7. The transaction conducted by the editor/viewer

Editor/Viewer: $\mathcal{U}_i$ edits/views an existing document
If $\mathcal{U}_i$ edits/views an existing document, without loss of generality, we assume the underlying document is F and the provenance record $P_j = h(F_j||ID_i)$ with $j \geq 2$, where F_j denotes the content of the document at the j-th stage. $\mathcal{U}_i$ interacts with $\mathcal{AS}$ as follows.

- $\mathcal{U}_i$ sends $(P_j, Bl_{j-1}, \sigma_{T_{j-1}})$ to $\mathcal{AS}$ via the secure channel, where Bl_{j-1} denotes the hash value of the block that contains the transaction whose data field is $h(h(P_{j-1}||PID_{j-1})||\sigma_{T_{j-1}})$.
- $\mathcal{AS}$ checks the validity of Bl_{j-1}, if the checking fails, reject.
- $\mathcal{AS}$ extracts PID_j from local storage.
- $\mathcal{AS}$ computes $\Theta(P_j) = H(P_j||PID_j||Bl_{j-1})$, generates a signature $\sigma_{T_j} = s \cdot \Theta(P_j)$, and $\mathcal{AS}$ sends σ_{T_j} to $\mathcal{U}_i$.
- $\mathcal{U}_i$ verifies σ_{T_j} by checking whether $e(\sigma_{T_j}, P) \overset{?}{=} e(\Theta(P_j), P_{pub})$.
- $\mathcal{U}_i$ creates a transaction Tx_j shown in Fig. 7, where $\mathcal{U}_i$ transfers service charge to the $\mathcal{AS}$'s account, and the data field of the transaction is set to $h(h(P_j||PID_j)||\sigma_{T_j}||Bl_{j-1})$.
- After Tx_j is recorded into the blockchain, $(P_j||PID_j||Bl_j, \sigma_{T_j}, Bl_{j-1})$ is sent to $\mathcal{C}$, and is published as the provenance record.

Finally, the secure provenance becomes

$$(P_1||PID_1||Bl_1, P_2||PID_2||Bl_2, \cdots, P_j||PID_j||Bl_j, \sigma_{T_j}).$$

Secure Provenance Verification: Given the provenance

$$(P_1||PID_1||Bl_1, P_2||PID_2||Bl_2, \cdots, P_j||PID_j||Bl_j, \sigma_{T_j}),$$

the auditor $\mathcal{A}$ checks its correctness as follows:

- Locate the last block Bl_j on the Ethereum blockchain, and verifies the validity of the last recorded provenance record $P_j||PID_j||Bl_j$.
- Compute $\Theta(P_j) = H(P_j||Bl_{j-1})$.
- Check whether $e(\sigma_{T_j}, P) \overset{?}{=} e(\Theta(P_j), P_{pub})$.
- Extract the data information from blockchain according to $Bl_1, ..., Bl_j$.
- Verify the integrity and timeliness of provenance by checking whether the hash value of provenance matches the extracted data. Here, the physical time when the provenance record was generated is derived from the height of the corresponding block. Specifically, assuming the time when $P_j||PID_j||Bl_j$ was generated is denoted by τ_j and the height of the corresponding block is denoted by ρ_j. $\tau_j = \tau_0 + \gamma \cdot \rho_j$ (seconds), where τ_0 is the physical time when the genesis block of Ethereum was generated (i.e., 2015-07-30, 03:26:13 PM +UTC) and γ is the average time that a block is mined in Ethereum.

If all the provenance records pass all the above checking, it can be accepted.

5.3 Remark

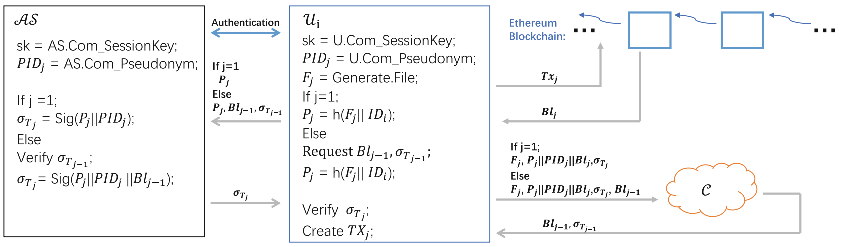
In ESP, $\mathcal{A}$ can check the time when a provenance record was generated by extracting the timestamp of the corresponding block from the blockchain. However, in Ethereum, the timestamp of a block cannot accurately reflect when transactions included in the block were generated, since the timestamp of the block might be confronted with up to 900 seconds errors. To overcome the time errors, in ESP, the auditor derives the transaction time from the height of the block including the transaction. The key observation is that the average time that a block is mined is deterministic and can be counted, and the blockchain height can be trusted to increase with respect of either short or long term, which is formalized as the chain-growth of blockchain [18]. By doing so, the time when a provenance record was generated suffers from around 15 seconds errors in ESP, which has improved the accuracy of timestamp significantly.

6 Security Analysis

ESP is secure against provenance record forgery, removal, modification, and repudiation attacks, even if the authenticated server is compromised. ESP also guarantees conditional privacy preserving. We defer the detailed security analysis to **Appendix B. Security Analysis of ESP**.

7 Implementation and Evaluation

We implement ESP by using JAVA, and the experiments are conducted on a laptop with Window 7 system, an Intel Core 2 i5 CPU and 8GB DDR3 of RAM. The security level is chosen to 80 bits and the hash function h is selected to SHA3-256. The implementation of ESP is illustrated in Fig. 8 and is described below. For clarity, we prefix calls with $\mathcal{AS}$ when they are made by $\mathcal{AS}$ and with $\mathcal{U}$ when they are made by $\mathcal{U}$.



$\mathcal{AS}$
sk = AS.Com_SessionKey;
PID_j = AS.Com_Pseudonym;
If j =1;
σ_{T_j} = Sig($P_j || PID_j$);
Else
Verify $\sigma_{T_{j-1}}$;
σ_{T_j} = Sig($P_j || PID_j || Bl_{j-1}$);
Authentication
If j=1
P_j
Else
$P_j, Bl_{j-1}, \sigma_{T_{j-1}}$
σ_{T_j}
$\mathcal{U}_i$
sk = U.Com_SessionKey;
PID_j = U.Com_Pseudonym;
F_j = Generate.File;
If j=1;
P_j = h($F_j || ID_i$);
Else
Request $Bl_{j-1}, \sigma_{T_{j-1}}$;
P_j = h($F_j || ID_i$);
Verify σ_{T_j};
Create TX_j;
Ethereum Blockchain:
Tx_j
Bl_j
If j=1;
$F_j, P_j || PID_j || Bl_j, \sigma_{T_j}$
Else
$F_j, P_j || PID_j || Bl_j, \sigma_{T_j}, Bl_{j-1}$
$\mathcal{C}$
$Bl_{j-1}, \sigma_{T_{j-1}}$

Fig. 8. Implementation of ESP

Com_SessionKey is an interactive algorithm to compute the session key between $\mathcal{U}$ and Com_Pseudonym is an algorithm to compute a pseudonym for $\mathcal{U}$.

The signature on the provenance record P_j and the pseudonym PID_j is implemented by $\mathsf{Sig(P_j||PID_j)}$. The verification of provenance record T_{j-1} is implemented by $\mathsf{Verify}\ \sigma_{\mathsf{T_{j-1}}}$. Generating/editing the target file on $\mathcal{U}$ is implemented by $\mathsf{Generate.File}$.

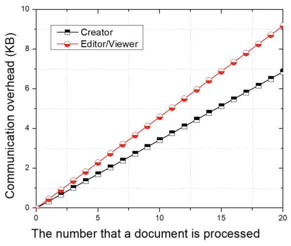


Fig. 9. Communication overhead of creator and editor/viewer

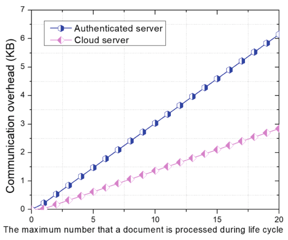


Fig. 10. Communication overhead of authenticated server and cloud server

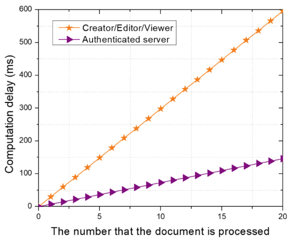


Fig. 11. Computational overhead

We show the communication overhead of the creator and editor/viewer in Fig. 9, where the size of human-memorisable password is set to 120 bits. We also show the communication overhead of the authenticated server and the cloud server in Fig. 10.

In ESP, generating the system parameters is a one-time computation, here we would not show the computation costs in initializing ESP. Instead, we show the computation delay on the users and authenticated server in Fig. 11. In ESP, generating a provenance record takes within 50 ms. ESP is constructed on the Ethereum blockchain. In Ethereum, a block as well as its transactions is considered *confirmed* if at least 12 consecutive blocks are mined following it. The average time that a block is mined is 15 seconds and hence a transaction takes averagely 15 seconds to be chained to the Ethereum blockchain. As such, publishing a new provenance record takes average 3.25 min in ESP, and the time interval between two successive provenance records only requires around 3.25 min. Another user may have to wait at least 3.25 min to work on the same document. It is the most important factor that affects the practicality of a secure provenance scheme, which is called window of latching (WoL) and is defined in *Definition* 1.

Another factor that affects the practicality of ESP is the costs to publish provenance record. The transaction fee in Ethereum can be set to be values from 0.000021 Ether to 0.000756 Ether, and the averagely fee is 0.000378 Ether. As of May, 2018, publishing a provenance record requires a user to pay average 25 US cents, which is acceptable to users with respect to the value of the data that ESP protects.

The above experiment results demonstrate that ESP is efficient in terms of communication and computation overhead. We have evaluated WoL of ESP, the evaluation results show that WoL is short and is acceptable in reality. The above analysis also indicates that WoL of ESP is mainly subject to the transaction confirmation time in the blockchain system, and the costs to publish provenance records are at the mercy of transaction fees in the blockchain system.

8 Conclusion

In this paper, we have proposed an efficient and secure data provenance scheme (ESP) for cloud storage systems. ESP employs the blockchain-based provenance record chain to ensure the correctness, integrity, and timeliness of provenance records. ESP protects users' real identities against the cloud storage server, which preserves users' privacy. Detailed security analyses have shown that ESP is secure and robust from various attacks with privacy preservation. Compared with existing schemes, ESP can resist the malicious identity manager. We have introduced the concept of window of latching (WoL) to evaluate the practicality of secure provenance schemes. We also have implemented ESP and show that WoL of ESP is short and can be acceptable in reality, which has demonstrated ESP is practical and efficient.

Appendix A. Blockchain

A blockchain is a shared immutable ledger for recording the history of transactions, it provides a tamper-proofing and distributed way to conduct transactions without a central authority. Technically, the blockchain is a linear collection of data elements, where each data element is called a *block*. All blocks are linked in chronological order to form a chain and secured using cryptography. Typically, each block contains a hash pointer, a timestamp, and transaction data, where the hash pointer points to the previous block as a link, and the timestamp indicates when the current block is chained to the blockchain [19]. Only valid transactions would be recorded into the blockchain.

The most prominent manifestation of blockchain is blockchain-based currencies, such as Bitcoin [19] and Ethereum [20], wherein the blockchain is used to serve as an open and distributed ledger that records transactions between two entities. The ledger here is verifiable and inherently resistant to modification of chained blocks. Participants who perform the transaction verification and maintain the blockchain are called *miner*.

The ledger of Ethereum blockchain can be considered as a state transition system. When a payer conducts a new transaction, she/he broadcasts the transaction to all miners. Each miner first verifies the validity of received transaction, and collects multiple new transactions into a block. Then each miner computes a valid nonce such that the hash value of the block is less than or equal to a value provided by the Ethereum system. The first miner who finds the nonce broadcasts the block including the nonce and a timestamp. Other miners accept the block only if the nonce and all transactions in it are valid. More details can be found in [20].

Appendix B. Security Analysis of ESP

ESP is secure against provenance record forgery attacks. In ESP, when a user $\mathcal{U}_i$ processes a document and requests $\mathcal{AS}$ to generate secure provenance on the document process, $\mathcal{AS}$ needs to authenticate $\mathcal{U}_i$. Without knowing the password pwd_i, an attacker cannot generate $C_1 = r_1 P$, $C_2 = E_k(ID_i||pwd_i||aP||ct)$ with $k = r_1 P_{pub}$. Note that this authorization is integrated into the corresponding provenance record and would be recorded into the blockchain, $\mathcal{AS}$ cannot deny the authorization and should responsible for it. In addition, since the timestamp ct has also been included, it can resist the replay attack. Due to this time-sensitive credential, an attacker cannot forge a secure provenance.

ESP is secure against provenance record removal/modification attacks. In the provenance record chain (shown in Fig. 5), if P_{j-1} is removed or modified, all successive hash values along the chain will be affected, and thereby the hash value of the last record $P_n||PID_n||Bl_n$ will be changed. In ESP, the provenance record is built on the BLS signature [21] which is existentially unforgeable. Even if the attacker colludes with $\mathcal{AS}$ to forge a signature, the transaction recorded into the blockchain cannot be removed/modified, collusion between any two entities in ESP can not remove/modify the published provenance records.

ESP also provides the non-repudiation. In ESP, the generation of a provenance record P_j for a user $\mathcal{U}_i$ requires the $\mathcal{AS}$'s assistance, where the pseudonym of $\mathcal{U}_i$, i.e., PID_i, is integrated into the provenance record. PID_i is derived from $\mathcal{U}_i$'s identity ID_i, and $\mathcal{AS}$ can "open" PID_i to prove the relationship between ID_i and PID_i. Furthermore, before publishing the secure provenance $(P_j||PID_j||Bl_j, \sigma_{T_j}, Bl_{j-1}), \mathcal{U}_i$ conducts a transaction Tx_j that transfers service charge to $\mathcal{AS}$, where the hash value of $(P_j||PID_j||\sigma_{T_j}||Bl_{j-1})$ is integrated into the transaction. By the security of Ethereum, anyone cannot impersonate others to conduct a transaction. Therefore, the auditor is able to confirm that P_j is generated by PID_i by checking the creator of the transaction related to P_j with the aid of $\mathcal{AS}$.

ESP also provides the conditional privacy preservation. To resist the privacy violation, the pseudonym $PID_j = E_k(ID_i||ct||b)$ in place of the real identity ID_i is included in the signature. Due to the security of $E(\cdot)$, the real identity ID_i cannot be disclosed from PID_j, the user privacy is preserved. The privacy preservation is also conditional, since $PID_j = E_k(ID_i||ct||b)$ is derived from ID_i with the master key k, once a provenance record P_j is in dispute, $\mathcal{AS}$ can use k to recover the real identity.

ESP enables auditors to securely check the timeliness of provenance records. In ESP, before a provenance record P_j is published by a user $\mathcal{U}_i$, a transaction Tx_j should be created and recorded into the Ethereum blockchain. The data value of Tx_j is set to the EPD and NPD related to P_j, if the block containing Tx_j is added to the blockchain, the EPD and NPD related to P_j is stored in the transaction. As such, integrating P_j into Tx_j essentially time-stamps P_j, and the time when the block containing Tx_j is chained to the blockchain represents the time when P_j is generated. This enables the auditor to check the timeliness of provenance records without introducing any trusted entity. Due to the security (chain-growth [18]) of Ethereum, anyone cannot modify the height-derived timestamp of P_j.

References

1. Zhang, Y., Xu, C., Liang, X., Li, H., Mu, Y., Zhang, X.: Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation. IEEE Trans. Inf. Forensics Secur. **12**(3), 676–688 (2017)
2. Hasan, R., Sion, R., Winslett, M.: Introducing secure provenance: problems and challenges. In: Proceedings of StorageSS, pp. 13–18 (2007)
3. Lu, R., Lin, X., Liang, X., Shen, X.: Secure provenance: the essential of bread and butter of data forensics in cloud computing. In: Proceedings of ASIACCS, pp. 282–292 (2010)
4. Xu, S., Ni, Q., Bertino, E., Sandhu, R.S.: A characterization of the problem of secure provenance management. In: Proceedings of ISI, pp. 310–314 (2009)
5. Audit trails. http://csrc.nist.gov/publications/nistbul/itl97-03.txt
6. Madden, B.A., Adams, I.F., Storer, M.W., Miller, E.L., Long, D.D., Kroeger, T.M.: Provenance based rebuild: using data provenance to improve reliability. Technical report, UCSC (2011)
7. Ashcroft, J., Daniels, D.J., Hart, S.V.: Forensic examination of digital evidence: a guide for law enforcement. https://www.ncjrs.gov/txtfiles1/nij/199408.txt (2011)
8. Lin, X., Chen, T., Zhu, T., Yang, K., Wei, F.: Automated forensic analysis of mobile applications on android devices. In: Proceedings of DFRWS USA (2018)
9. Zhang, Y., Xu, C., Li, H., Yang, K., Zhou, J., Lin, X.: HealthDep: an efficient and secure deduplication scheme for cloud-assisted eHealth systems. IEEE Trans. Ind. Inform., to appear. https://doi.org/10.1109/TII.2018.2832251
10. Health insurance portability and accountability act. https://en.wikipedia.org/wiki/Health_Insurance_Portability_and_Account-ability_Act
11. BTK killer. https://en.wikipedia.org/wiki/Dennis_Rader
12. Zhang, Y., Xu, C., Yu, S., Li, H., Zhang, X.: SCLPV: secure certificateless public verification for cloud-based cyber-physical-social systems against malicious auditors. IEEE Trans. Comput. Soc. Syst. **2**(4), 159–170 (2015)
13. Chow, S.S.M., Chu, C.-K., Huang, X., Zhou, J., Deng, R.H.: Dynamic secure cloud storage with provenance. In: Naccache, D. (ed.) Cryptography and Security: From Theory to Applications. LNCS, vol. 6805, pp. 442–464. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28368-0_28
14. Hasan, R., Sion, R., Winslett, M.: Preventing history forgery with secure provenance. ACM Trans. Storage **5**(4), 12 (2009)
15. Wang, Q., Hassan, W.U., Bates, A., Gunter, C.: Fear and logging in the Internet of Things. In: Proceedings of NDSS (2018)
16. Lynch, C.A.: When documents deceive: trust and provenance as new factors for information retrieval in a tangled web. J. Assoc. Inf. Sci. Technol. **52**(1), 12 (2001)
17. Martin, A.P., Lyle, J., Namiluko, C.: Provenance as a security control. In: Proceedings of TaPP (2012)
18. Badertscher, C., Maurer, U., Tschudi, D., Zikas, V.: Bitcoin as a transaction ledger: a composable treatment. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 324–356. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_11

19. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf
20. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, vol. 151 (2014)
21. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Proceedings of ASIACRYPT, pp. 514–532 (2001)