Journal of Computers Vol. 30 No. 6, 2019, pp. 252-262
doi:10.3966/199115992019123006021

An Auditing Scheme for Cloud-based Checkout Systems

Tao-Ku Chang*, Cheng-Yen Lu

Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien 974, Taiwan
{tkchang, 610121110}@gms.ndhu.edu.tw

Received 10 May 2019; Revised 10 June 2019; Accepted 2 July 2019

Abstract. The goal of this paper is to design and implement a security mechanism for cloud-based checkout systems based on a chain-hashing scheme. Many cloud-based checkout systems are developed for merchants, however, storing transaction data in cloud storage is commonly associated with serious security risks. The repudiation problem exists between merchants and service providers. We need a scheme that enables the service provider to prove its innocence and the merchant to prove its guilt. This proof of innocence is also called auditing. This paper designs a secure cloud-based checkout system, and we use chain hashing to design and auditing scheme for checkout systems.

Keywords: auditing, cloud security, e-commerce

1 Introduction

Cloud technologies are applied to many commercial application systems. Cloud computing has given birth to the checkout system as Software as a Service (SaaS). Cloud-based checkout systems have many advantages over traditional checkout systems. Users do not need to construct a computer facility, or maintain and purchase specific hardware, rather it can be accessed from the internet using an internet browser wherever there is an internet connection. The cloud-based checkout systems are independent of plat-form and operating system limitations. They are made compatible with a wide range of hardware. The user data, including transactions and inventory information, are stored in cloud storage and allow the system to not run locally, so there is no requirement of installation.

However, there are a substantial amount of existing security issues concerning storing business data or even commercial secrets in cloud storage: (1) Cloud service providers could lose data due to possible system setting errors, computing mistakes, or server crashes; (2) The service provider is likely to then restore the files using a backup of an earlier version of the files and their associated digital signatures, and can then deny the user's access to the latest version of files lost. This is called a roll-back attack [1] or replay attack [2]; (3) Outside attackers could access or even tamper with the data from the server illegally. Employees of cloud service providers could inadvertently allow the external attacks. Even though cloud service providers have constructed a perfect security system, these types of security flaws are still inevitable. Therefore, the concern diminishes many people's willingness to store important data or even vital commercial secrets in the cloud. Also, none of today's cloud-based checkout services provide security guarantees in their service-level agreements (SLAs).

This research proposed a mutual auditing mechanism for cloud-based checkout systems. Client and service providers exchange attestations for every transaction. These attestations are chain hashed and will be used in auditing to ensure the accuracy of the transaction information in cloud storage. Once the transaction information uploaded to the cloud appears to be damaged or tampered with, merchants and cloud service providers could clarify and prove the origin of the mistakes. It can guarantee mutual nonrepudiation between the clients and service providers.

* Corresponding Author

The remainder of this paper is organized as follows: Section 2 gives an overview of related works and technologies, Section 3 presents the proposed mechanism, Section 4 presents a security analysis and implementation, and finally, conclusions about the work described in the paper are drawn in Section 5.

2 Related Works

We need a scheme that enables the service provider to prove its innocence and the users to prove its guilt. This proof of innocence is also called auditing [3]. To solve the mutual auditing issue between cloud service providers and users, some systems consider cloud storage untrusted and provide a way to detect violations of integrity, write-serializability, and freshness. SUNDR is a network file system designed to store data securely on untrusted servers [4]. SUNDR lets clients detect any attempt at unauthorized file modification by malicious server operators or users. SUNDR's protocol achieves a property called fork consistency, which guarantees that clients can detect any integrity or consistency failures as long as they see each other's file modifications. Clients maintain a version structure list which stores a list of the version's modifications in the storage server. When user X performs a file system operation, X's client acquires the global lock and downloads the latest version structure for each user and group. SUNDR deals with the violation detection problem using the so-called "forking" semantics [5] which is also employed in [6-8]. These solutions guarantee integrity, and by adding some extra out-of-band communication among the clients can also be used to achieve a related notion of consistency.

Popa et al. proposed a system called CloudProof that contains a protocol to detect and prove violations of four desirable security properties of cloud storage: confidentiality, integrity, write serializability, and read freshness (denoted as CIWF) [9]. Users can detect the violations of CIWF of the cloud storage system and the service provider can disprove false accusations made by users; that is, the users of CloudProof cannot frame the cloud. Its framework supports the mutual nonrepudiation guarantee. These proofs are based on attestations, which are signed messages that bind the users to the requests they make and the service provider to maintaining the data in a certain state. The protocol proposed in CloudProof is able to maintain the mutual nonrepudiation property of cloud storage. Hwang et al. also proposed C&L scheme [10] which does not require multiple client devices to exchange any messages and can still guarantee nonrepudiation and maintain the CIWF requirements. It also proposed how to apply the hash tree [11] to remove accumulated attestations.

3 The Proposed Auditing Scheme

The goal of this paper is to make it possible to detect violations of security properties and guarantee mutual nonrepudiation between the merchants and service providers of the cloud-based checkout system. The client device and service provider exchange attestations for every transaction. These attestations are chain hashed so that the client device of the merchant only has to store the last attestation that contains the last chained hash. The service provider keeps all the attestations for use when proofs are required.

Table 1. The symbol descriptions

Symbol	Description
M	The devices of a merchant
P	Cloud Service provider
CompanyID	Merchant ID (account of a system manager)
CompanyName	Merchant name
CompanyPWD	Password of a merchant
pub(.)	Public key
StaffID	Staff's ID of a merchant
StaffPWD	Staff's Password of a merchant
AA	Administrative authority
OP	Operation (add, delete, modify, query)
Description	Other information
TID	Transaction ID
TD	Transaction Data (Transaction Detail)
LSN	Local Sequence Number

Table 1. The symbol descriptions (continue)

Symbol	Description
CH_i	Hash chain. $CH_{i-1}=h(Q_{i-1}, CH_{i-2})_{pri(P)}$, $CH_0=h(h(TID)_{pri(M)})_{pri(P)}$
TDHC	Hash chain of the transaction detail. $TDHC_i=h(TDHC_{i-1}, h(TD_i))$, $i=2, \ldots, n$. $TDHC_1=h(h(TD_1))$
$h(\ )_{pri(.)}$	Hash function
L_i	Inform message

The proposed scheme involves the following entities: a merchant (M), and a service provider (P), which have the following public and private key pairs: (pri(M), pub(M)) and (pri(P), pub(P)), respectively. $[O]_{pri(x)}$ and $[O]_{pub(x)}$ are used to denote a digital signature and the encryption of data object O that is generated by the private and public keys of a subject x. Data objects within square brackets that are separated by commas are first connected and then have cryptographic operations performed on them. The scheme comprises of four phases: a registration phase, a login phase, a transaction phase, and an auditing/clearing phase. The symbol descriptions of the following figures are shown in Table.

3.1 Registration Phase

The merchant has to register an account and then registers the employees' accounts. Fig. 1 shows the following steps and message exchanges involved in the registration phase:

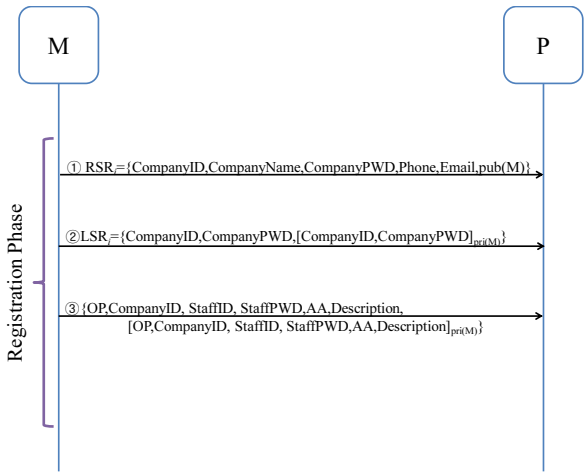

Fig. 1. The message exchange of the registration phase

(1) M sends a register service request, RSR_i={CompanyID, CompanyName, CompanyPWD, Phone, Email, pub(M)}, to P for registration.

(2) After registration, M sends a login service request, LSR_i={CompanyID, CompanyPWD, $[CompanyID, CompanyPWD]_{pri(M)}$}, to P for login.

(3) After login, M adds the data {OP, CompanyID, StaffID, StaffPWD, AA, Description, $[OP, CompanyID, StaffID, StaffPWD, AA, Description]_{pri(M)}$} for employee's registration.

3.2 Login Phase

The merchant has to login to the checkout system and then employees login with his/her account and password before starting the checkout service. Fig. 2 shows the steps and message exchanges in the login phase:

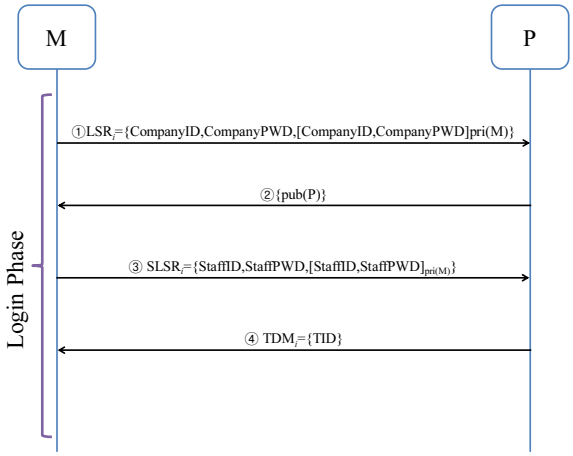

Fig. 2. The message exchange of the login phase

(1) M sends LSR_i, where LSR_i={CompanyID, CompanyPWD, [CompanyID, CompanyPWD]$_{pri(M)}$}, to P for login.

(2) P receives LSR_i. It verifies the signature and checks whether the account and password are correct. When login process is successful, P sends {pub(P)} to M.

(3) M stores {pub(P)} for verification. Employee who has an account logins for checkout services. M sends $SLSR_i$ (Staff Login Service Request), where $SLSR_i$={StaffID, StaffPWD, [StaffID, StaffPWD]$_{pri(M)}$}, to P.

(4) P receives $SLSR_i$. It verifies the signature and checks whether employee's account and password are correct. When login process is successful, P sends TDM_i, where TDM_i (Tansaction ID Delivery Message)={TID}, to M. TID is a non-duplicate transaction ID. Its format is {StaffID + TIME}. For example, TID will be {A610201810120708} when the checkout system assigns a TID to an employee whose ID is "A610" at 07:08 p.m. on October 12, 2018.

3.3 Transaction Phase

The following steps (depicted in Fig. 3) are involved when a customer intends to check out.

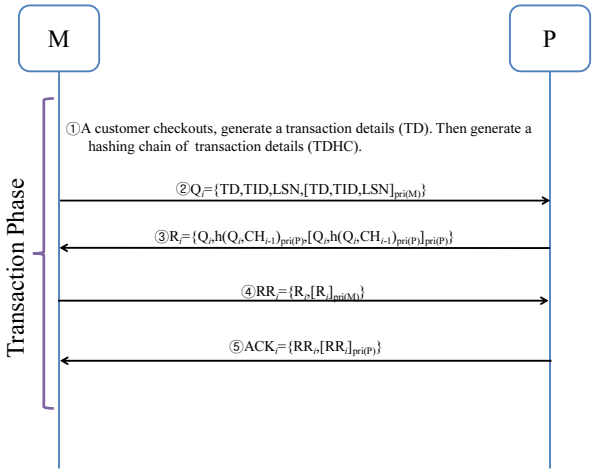

Fig. 3. The message exchange of the transaction phase

(1) When a customer checks out, the client device of M produces transaction data, TD and makes a TDHC (Transaction Data Hash Chain), where $TDHC_i$=h($TDHC_{i-1}$, h(TD_i)), i=2, …, n, $TDHC_1$=h(h(TD_1)). TDHC contains each hash value of TD_i and is stored on the client device for future auditing while other data is stored on the service provider.

(2) The device of M sends a request message Q_i, where Q_i={TD, TID, LSN, [TD, TID, LSN]$_{pri(M)}$}, to P. Local sequence number (LSN), which is an increasing number, is bound with TID and maintained by

the merchant. TID and LSN enable to make sure that the legality of a sequence of messages between the merchant and the service provider in the chain-hashing.

(3) When P receives Q_i, it verifies if this is a valid request message by checking the digital signature and then checks if LSN is continuous. P sends a response message R_i, where $R_i=\{Q_i, h(Q_i, CH_{i-1})_{pri(P)}, [Q_i, h(Q_i, CH_{i-1})_{pri(P)}]_{pri(P)}\}$, to the device of M. Note that CH_{i-1} is $(i\text{-}1)^{th}$ chain hash and $CH_{i-1}=h(Q_{i-1}, CH_{i-2})_{pri(P)}$, $CH_0=h(h(TID)_{pri(M)})_{pri(P)}$.

(4) When the device of M receives R_i, it verifies if its signature is valid. Then it sends reply-response message RR_i to P, where $RR_i=\{R_i, [R_i]_{pri(M)}\}$. Note that the service provider keeps all RR_i for when proofs are required. These reply-response messages are not deleted until auditing and clearing phase is completed.

(5) When P receives message from M, it stores TD and RR_i to the database. Finally, it sends an acknowledgement message ACK_i, where $ACK_i=\{RR_i,[RR_i]_{pri(P)}\}$, to M. M receives ACK_i and increments the value of LSN by 1. It indicates that this transaction process is completed. M just need to keep the newest ACK_i as the attestation and ACK_{i-1} will be deleted.

3.4 Auditing and Clearing Phase

Over a period of time, it is possible for a client device of the merchant to transfer a huge number of transaction data, which could result in an excessive overhead for service providers to store the server-side attestations or when an employee intends to sign out. We need a way to audit these attestations and discard them. The details of the message exchanges in the auditing and clearing phase illustrated in Fig. 4 are as follows:

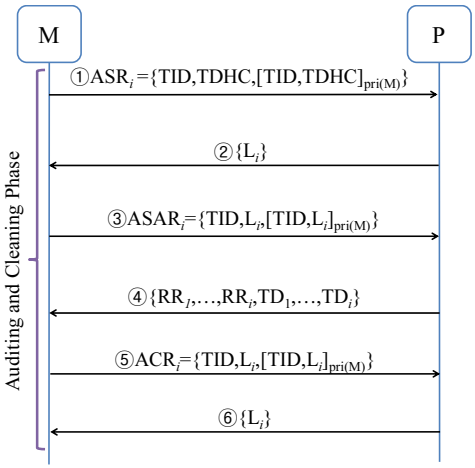

Fig. 4. The message exchange of the auditing and clearing phase

(1) When an employee of M signs out or intends to clear attestations, the device of M sends ASR_i, where $ASR_i=\{TID, TDHC, [TID, TDHC]_{pri(M)}\}$ to P for the request of auditing and clearing.

(2) P receives ASR_i from M. It verifies if the signature is valid. It calculates $TDHC_i$' according to all transaction data that belong to the same TID in ASR_i. If $TDHC_i$ in ASR_i and $TDHC_i$' are the same, P sends an acknowledgement message, $\{L_i\}$, to the client device of M.

(3) When the device of M receives the acknowledgement message, it replies a request of auditing server-side attestation and transaction ($ASAR_i$), where $ASAR_i=\{TID, L_i, [TID, L_i]_{pri(M)}\}$, to P.

(4) When P receives $ASAR_i$ from M, it sends all of transaction data TD_s with the same TID and RR_s to M for auditing.

(5) M verifies all RR_s from P if the signature of each transaction is valid. If all RR_s are correct, M sends a request of clearing attestation (ACR_i), where $ACR_i=\{TID, L_i, [TID, L_i]_{pri(M)}\}$, to P.

(6) When P receives ACR_i, it verifies if the signature is valid. P deletes $\{RR_1, RR_2, \ldots, RR_i\}$ and the authority of TID. Finally, it sends $\{L_i\}$ to M for replying that the process of auditing and clearing is complete. The client device of M will log out automatically. It must apply a new TID, then the checkout service could be continued.

When the auditing is unsuccessful, this mechanism enables the service provider to prove its innocence and the users to prove its guilt. The details of the message exchanges when the auditing and clearing is unsuccessful are illustrated in Fig. 5.

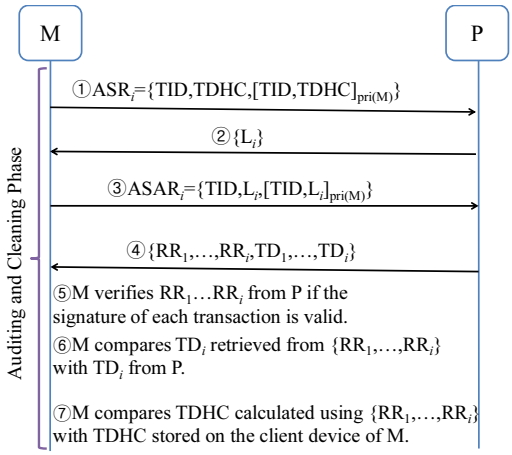

Fig. 5. The message exchange of the auditing failure

(1) When an employee of M signs out or intends to clear attestations, the device of M sends ASR_i, where ASR_i={TID, TDHC, [TID, TDHC]$_{pri(M)}$} to P for the request of auditing and clearing.

(2) P receives ASR_i from M. It verifies if the signature is valid. It calculates $TDHC_i$' according to all transaction data that belong to the same TID in ASR_i. If $TDHC_i$ in ASR_i and $TDHC_i$' are not the same, P sends an error message, {L_i}, to the client device of M. {L_i} contains auditing failure announcement and which transaction data is incorrect.

(3) The device of M receives the error message of auditing, but it still replies a request of auditing server-side attestation and transaction ($ASAR_i$), where $ASAR_i$={TID, L_i, [TID, L_i]$_{pri(M)}$}, to P.

(4) When P receives $ASAR_i$ from M, it sends all of transaction data TD_s with the same TID and RR_s to M for auditing.

(5) M verifies all RRs from P if the signature of each transaction is valid. If one of signatures is incorrect, that can prove service provider's guilt.

(6) If all signatures in RR_s are valid, the client of M compares TD_i retrieved from {RR_1, …, RR_i} with TD_i from P. If one of comparisons is inconsistent, that can prove service provider's guilt.

(7) If all attestations from P are correct (step5 and step 6), M compares TDHC calculated using {RR_1, …, RR_i} with TDHC stored on the client device of M. If one of comparisons is inconsistent, that can prove the merchant's guilt.

4 Security Analysis and Implementation

In this study, we made no assumption about the honesty of the merchant or the service provider. In this mechanism, TID is used to identify each employee when logging in the checkout system even though the same client device is used by the same employee. LSN is to ensure data serializability and freshness. Furthermore, LSN is maintained by the merchant, based on TID and LSN, cloud service providers could not launch a roll-back attack to avoid the accountability. The required security and privacy features were implemented as follows:

(1) Merchant counterfeits TDHC and ACK: The merchant can modify TDHC and ask compensation for inconsistent TD modified by the service provider. RR stored in the service provider is signed by the merchant; the merchant cannot repudiate its signature. The service provider can compare TD from RR with counterfeit TD from the merchant to prove its innocence. Moreover, RR includes the newest LSN, the merchant cannot add or delete one TD for compensation. Even though it modifies ACK and LSN, the signature from service provider will be invalid and result in the inability of hash chain CH to work correctly.

(2) Cloud service provider counterfeits RR and TD: The service provider can counterfeit RR and TD to avoid its accountability due to bugs, crashes, operator errors, or misconfigurations. In this scheme, the

merchant keeps the newest ACK, which can prove the number of transactions, where ACK_i={RR_i, $[RR_i]_{pri(P)}$}, as the attestation. If the service provider counterfeits RR or recovers TD, the signature from the merchant will be invalid.

According to the proposed scheme shown in Section 3, we implement a system to prove that this scheme can work smoothly. We first define the formats of the transaction data, transaction data hash chain, request, response, reply-response, and ACK messages. These messages were represented as XML documents in our implementation. Fig. 6 to Fig. 11 show examples of transaction data, transaction data hash chain, request, response, and reply-response, and ACK messages in the chain-hashing on transaction phase, respectively.

```
<Transaction>
  <ReceiptNum>AA12345678</ReceiptNum>
  <Date>2018/10/10 16:30:06</Date>
  <CompanyID>test</CompanyID>
  <StaffID>cchaha</StaffID>
  <TotalPrice>25</TotalPrice>
  <Detail>
    <Good>
      <Barcode>4710782171879</Barcode>
      <Price>25</Price>
      <Amount>1</Amount>
    </Good>
  </Detail>
</Transaction>
```

Fig. 6. The example of TD (Transaction Data)

```
<TDHashChain>
  <RootHash>XHfGEwNAeONXQlOgdLULLNQiVbaiLs17JRDZ2sVxsls=</RootHash>
  <TDFile>
    <TransactionData>
      <Name>TDcchaha20181010162956_1</Name>
      <Hash>XHfGEwNAeONXQlOgdLULLNQiVbaiLs17JRDZ2sVxsls=</Hash>
    </TransactionData>
  </TDFile>
</TDHashChain>
```

Fig. 7. The example of TDHC (Transaction Data Hash Chain)

```
<Request>
  <Transaction_data>
    <Transaction>
      <ReceiptNum>AA12345678</ReceiptNum>
      <Date>2018/1010 16:30:06</Date>
      <CompanyID>test</CompanyID>
      <StaffID>cchaha</StaffID>
      <TotalPrice>25</TotalPrice>
      <Detail>
        <Good>
          <Barcode>4710782171879</Barcode>
          <Price>25</Price>
          <Amount>1</Amount>
        </Good>
      </Detail>
    </Transaction>
  </Transaction_data>
  <TID>cchaha20181010162956</TID>
  <LSN>1</LSN>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod
       Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <SignatureMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="">
        <Transforms>
          <Transform Algorithm=
           "http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
        </Transforms>
        <DigestMethod
          Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <DigestValue>Yt5LW6FRBrth4pMiOLxnaBndMUI=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>
      b2s21f7DihrMq7gc9J8K50rpvzjArtXY7uPJZVieZcGraO38wnLwj8TK5irLWfwdmro0YVent0k+OHM42ZNlFu7rAvVrzdtWIART9m
      lYrjW0xZTG5SBljb9Fxg6CT4+Adg4sEeKu1Gab7stoD6mn+wsSf/CePq2x1+Siz/d9/5Y=</SignatureValue>
  </Signature>
</Request>
```

Fig. 8. The example of Q_i={TD, TID, LSN, $[TD, TID, LSN]_{pri(M)}$}

```
<Response>
  <Request>
    <Transaction_data>
      <Transaction>
        <ReceiptNum>AA12345678</ReceiptNum>
        <Date>2018/10/10 16:30:06</Date>
        <CompanyID>test</CompanyID>
        <StaffID>cchaha</StaffID>
        <TotalPrice>25</TotalPrice>
        <Detail>
          <Good>
            <Barcode>4710782171879</Barcode>
            <Price>25</Price>
            <Amount>1</Amount>
          </Good>
        </Detail>
      </Transaction>
    </Transaction_data>
    <TID>cchaha20181010162956</TID>
    <LSN>1</LSN>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
        <CanonicalizationMethod
         Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
        <SignatureMethod
         Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
        <Reference URI="">
          <Transforms>
            <Transform Algorithm=
              "http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
          </Transforms>
          <DigestMethod
           Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <DigestValue>Yt5LW6FRBrth4pMiOLxnaBndMUI=</DigestValue>
        </Reference>
      </SignedInfo>
      <SignatureValue>
        b2s21f7DihrMq7gc9J8K50rpvzjArtXY7uPJZVieZcGraO38wnLwj8TK5irLWfwdmro0YVent0k+OHM42ZNlFu7rAvVrzdtWIAR
        T9mlYrjW0xZTG5SBljb9Fxg6CT4+Adg4sEeKulGab7stoD6mn+wsSf/CePq2x1+Siz/d9/5Y=</SignatureValue>
    </Signature>
  </Request>
  <ChaingHash>EcZHnXVswpssB2IbZ4gERQfsI3yZtTbTHVLBJpPd1rc=</ChaingHash>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod
       Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <SignatureMethod
       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="">
        <Transforms>
          <Transform Algorithm=
           "http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
        </Transforms>
        <DigestMethod
         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <DigestValue>5NIb2kCLEcyRNS64RV+7z2Y2d1Y=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>
      tBKn6Yd22NjaJfGeZRj0M8rWj1Ws6HvzJiz14wUNtvCFbvN3ZixVS/2jbDtZvWHWL2NxGSjAlJNIUrLRgSivRjra+SZdmnCT9TTMlb
      kkXkh4/z7I++z8u3wZATOiivQvKRYeEypGNidctQDyz38mKIdwqgDqcb8yHKGZABRd0Ww=</SignatureValue>
  </Signature>
</Response>
```

Fig. 9. The example of $R_i=\{Q_i, h(Q_i, CH_{i-1})_{pri(P)}, [Q_i, h(Q_i, CH_{i-1})_{pri(P)}]_{pri(P)}\}$

```
<ResponseOfResponse>
  <Response>
    <Request>
      <Transaction_data>
        <Transaction>
          <ReceiptNum>AA12345678</ReceiptNum>
          <Date>2018/10/10 16:30:06</Date>
          <CompanyID>test</CompanyID>
          <StaffID>cchaha</StaffID>
          <TotalPrice>25</TotalPrice>
          <Detail>
            <Good>
              <Barcode>4710782171879</Barcode>
              <Price>25</Price>
              <Amount>1</Amount>
            </Good>
          </Detail>
        </Transaction>
      </Transaction_data>
      <TID>cchaha20181010162956</TID>
      <LSN>1</LSN>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
          <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
          <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
          <Reference URI="">
            <Transforms>
              <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <DigestValue>Yt5LW6FRBrth4pMiOLxnaBndMUI=</DigestValue>
          </Reference>
        </SignedInfo>
        <SignatureValue>
        b2s21f7DihrMq7gc9J8K50rpvzjArtXY7uPJZVieZcGraO38wnLwj8TK5irLWfwdmro0YVent0k+OHM42ZNlFu7rAvVrzdtWIAR
        T9mlYrjW0xZTG5SBljb9Fxg6CT4+Adg4sEeKulGab7stoD6mn+wsSf/CePq2x1+Siz/d9/5Y=
        </SignatureValue>
      </Signature>
    </Request>
    <ChaingHash>EcZHnXVswpssB2IbZ4gERQfsI3yZtTbTHVLBJpPd1rc=</ChaingHash>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
        <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
        <Reference URI="">
          <Transforms>
            <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <DigestValue>5NIb2kCLEcyRNS64RV+7z2Y2d1Y=</DigestValue>
        </Reference>
      </SignedInfo>

<SignatureValue>tBKn6Yd22NjaJfGeZRj0M8rWj1Ws6HvzJiz14wUNtvCFbvN3ZixVS/2jbDtZvWHWL2NxGSjAlJNIUrLRgSivRjra+SZd
mnCT9TTMlbkkXkh4/z7I++z8u3wZATOiivQvKRYeEypGNidctQDyz38mKIdwqgDqcb8yHKGZABRd0Ww=</SignatureValue>
    </Signature>
  </Response>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
   </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <DigestValue>0Cqo4wo5gvYEFLEVvyWWe71Qq8A=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>
     hBaILSvt6BSEha85o+vvuPrMlvBrciBI3rGzCtFwwsP44yIXq3WVkIsJz+/fD3QvGXGjAjTbk6nEADqyY31tbQxErqw19V6xuMeILCh
     ZRBCRXvgS42w+VywquylYvGb+a7bTfsCe5bq4gOGu31JWv62/huRV6iUJ0lpHob8GWA0=</SignatureValue>
  </Signature>
</ResponseOfResponse>
```

Fig. 10. The example of $RR_i=\{R_i, [R_i]_{pri(M)}\}$

```
<Acknowledgement>
  <ResponseOfResponse>
    <Response>
      <Request>
        <Transaction_data>
          <Transaction>
            <ReceiptNum>AA12345678</ReceiptNum>
            <Date>2018/10/10 16:30:06</Date>
            <CompanyID>test</CompanyID>
            <StaffID>cchaha</StaffID>
            <TotalPrice>25</TotalPrice>
            <Detail>
              <Good>
                <Barcode>4710782171879</Barcode>
                <Price>25</Price>
                <Amount>1</Amount>
              </Good>
            </Detail>
          </Transaction>
        </Transaction_data>
        <TID>cchaha20181010162956</TID>
        <LSN>1</LSN>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <CanonicalizationMethod
             Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
            <Reference URI="">
              <Transforms>
                <Transform
                 Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
              </Transforms>
              <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
              <DigestValue>Yt5LW6FRBrth4pMiOLxnaBndMUI=</DigestValue>
            </Reference>
          </SignedInfo>
          <SignatureValue>
          b2s21f7DihrMq7gc9J8K50rpvzjArtXY7uPJZVieZcGraO38wnLwj8TK5irLWfwdmro0YVent0k+OHM42ZNlFu7rAvVrzdtWIA
          RT9mlYrjW0xZTG5SBljb9Fxg6CT4+Adg4sEeKu1Gab7stoD6mn+wsSf/CePq2x1+Siz/d9/5Y=</SignatureValue>
        </Signature>
      </Request>
      <ChaingHash>EcZHnXVswpssB2IbZ4gERQfsI3yZtTbTHVLBJpPd1rc=</ChaingHash>
      <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
        <SignedInfo>
          <CanonicalizationMethod
           Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
          <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
          <Reference URI="">
            <Transforms>
              <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <DigestValue>5NIb2kCLEcyRNS64RV+7z2Y2d1Y=</DigestValue>
          </Reference>
        </SignedInfo>
        <SignatureValue>
        tBKn6Yd22NjaJfGeZRj0M8rWj1Ws6HvzJiz14wUNtvCFbvN3ZixVS/2jbDtZvWHWL2NxGSjAlJNIUrLRgSivRjra+SZdmnCT9TT
        MlbkkXkh4/z7I++z8u3wZATOiivQvKRYeEypGNidctQDyz38mKIdwqgDqcb8yHKGZABRd0Ww=</SignatureValue>
      </Signature>
    </Response>
    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
        <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
        <Reference URI="">
          <Transforms>
            <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
          </Transforms>
          <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <DigestValue>0Cqo4wo5gvYEFLEVvyWWe71Qq8A=</DigestValue>
        </Reference>
      </SignedInfo>
      <SignatureValue>
      hBaILSvt6BSEha85o+vvuPrMlvBrciBI3rGzCtFwwsP44yIXq3WVkIsJz+/fD3QvGXGjAjTbk6nEADqyY31tbQxErqw19V6xuMeILC
      hZRBCRXvgS42w+VywquylYvGb+a7bTfsCe5bq4gOGu31JWv62/huRV6iUJ0lpHob8GWA0=</SignatureValue>
    </Signature>
  </ResponseOfResponse>
  <AttestationChainHash>g38vAxDwIRZ5CjZ4lNyKWd6UBDdCilwIcFYE5b7p5eY=</AttestationChainHash>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
        <DigestValue>sM5iNIWEvhpwOc3eBGEWOd2YT8U=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>
     HRQqYh3bzO8k+xtSywnB9fsDisfu59Ssk+LQvGTz5Dz2dZMQnmGEAJWav72xjmKVuSjP23YvO7vDtdtGfFGE/ODPw+hSxNp93AUiTG0
     +a3A4RacCjngwZpKoEg0GELkNkhc4/IV9ntBBpZj4cWBqin21V3N7a7jghqVPWlGF/+w=</SignatureValue>
  </Signature>
</Acknowledgement>
```

Fig. 11. The example of $ACK_i=\{RR_i, [RR_i]_{pri(P)}\}$

5 Conclusion

This research was designed to propose a mutual auditing mechanism for a cloud-based checkout system. It provides and ensures the evidence of nonrepudiation and continuity between merchants and cloud service providers through exchanging attestations from every transaction and the hashing of these attestations as a chain. Merchants only need to retain the last attestation and TDHC instead of all the transaction details for auditing. Furthermore, cloud service providers do not need to worry about merchants forging transaction data in order to get reimbursement which results in the loss of reputation and business. The proposed scheme can be applied to a mobile checkout system and combine with mobile payment in the future.

References

[1] J. Feng, Y. Chen, D. Summerville, W.-S. Ku, Z. Su, Enhancing cloud storage security against roll-back attacks with a new fair multi-party non-repudiation protocol, in: Proc. the 8th IEEE Consumer Communications and Networking Conference, 2011.

[2] A. Shraer, C. Cachin, A. Cidon, I. Keidar, Y. Michalevsky, D. Shaket, Venus: verification for untrusted cloud storage, in: Proc. The 17th ACM Workshop on Cloud Computing Security Workshop, 2010.

[3] M.A. Shah, M. Baker, J. Mogul, R. Swaminathan, Auditing to keep online storage services honest, in: Proc. the 11th USENIX Workshop on Hot Topics in Operating Systems, 2007.

[4] J. Li, M. Krohn, D. Mazières, D. Shasha, Secure Untrusted Data Repository (SUNDR), in: Proc. the 6th conference on Symposium on Operating Systems Design & Implementation, 2004.

[5] D. Mazières, D. Shasha, Building secure file systems out of Byzantine storage, in: Proc. the 21st Annual ACM Symposium on Principles of Distributed Computing, 2002.

[6] C. Cachin, A. Shela, A. Shraer, Efficient Fork-Linearizable access to untrusted shared memory, in: Proc. the 26th Annual ACM Symposium on Principles of Distributed Computing, 2007.

[7] M. Majuntje, D. Dobre, M. Serafini, N. Suri, Abortable Fork-Linearizable storage, in: Proc. the 13th International Conference on Principles of Distributed Systems, 2009.

[8] C. Cachin, M. Geisler, Integrity protection for revision control, in: Proc. The 7th International Conference on Applied Cryptography and Network Security, 2009.

[9] R.A. Popa, J.R. Lorch, D. Molnar, H. Wang, L. Zhuang, Enabling security in cloud storage SLAs with CloudProof, in: Proc. 2011 USENIX Annual Technical Conference, 2011.

[10] G.-H. Hwang, J.-Z. Peng, W.-S. Huang, A mutual nonrepudiation protocol for cloud storage with interchangeable accesses of a single account from multiple devices, in: Proc. the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 2013.

[11] R.C. Merkle, A digital signature based on a conventional encryption function, in: Proc. Conference on the Theory and Application of Cryptographic Techniques, 1987.