南　　華　　大　　學

資訊管理學系

碩士論文

在雲端儲存體做一個新型有效公開稽核方案

A new efficient public auditing scheme for cloud storage

研 究 生：林哲宇

指導教授：周志賢

中華民國106年　01　月　15　日

南　華　大　學

資訊管理學系

碩 士 學 位 論 文

A new efficient public auditing scheme for cloud storage

研究生：林哲宇

經考試合格特此證明

口試委員：劉建人

邱宏彬

周志賢

指導教授：周志賢

系主任(所長)：

口試日期：中華民國　106　年　01　月　05　日

南 華 大 學 碩 士 班 研 究 生

論 文 指 導 教 授 推 薦 函

資訊管理 系碩士班 林哲宇 君所提之論文

A new efficient public auditing scheme for cloud storage

係由本人指導撰述，同意提付審查。

指導教授 周志賢

106 年 1 月 5 日

南華大學資訊管理學系碩士論文著作財產權同意書

立書人：林哲宇 之碩士畢業論文

中文題目：在雲端儲存體做一個新型有效公開稽核方案

英文題目：A new efficient public auditing scheme for cloud storage

指導教授：周志賢 博士

學生與指導老師就本篇論文內容及資料其著作財產權歸屬如下：

☑ 共同享有著作權

☐ 共同享有著作權，學生願「拋棄」著作財產權

☐ 學生獨自享有著作財產權

學　　生：林哲宇 （請親自簽名）

指導老師：周志賢 （請親自簽名）

中 華 民 國 105 年 01 月 05 日

致　謝

承蒙指導教授 周志賢博士，在學生就讀研究所期間的悉心教導與鼓勵，並且教導專業知識與論文的撰寫，不斷給予指導與啟發，並於學生論文撰寫期間，百忙中抽空逐字斧正，使論文得以順利完成，心中萬分感激。在這兩年的研究期間，也要特別感謝南華大學資管系系上全體教師與同一間實驗室大學部與碩專班的同學。在學期間因有你們的陪伴而增添了不少色彩，也讓我有不斷前進的動力。由於您們的幫助，本論文才得以順利完成，在此一同致上由衷的感謝。

同時也要感謝家人給予的支持，由於您們給我的鼓勵與關懷，讓我能無後顧之憂的完成學業。最後，謹將此論文獻給愛我的家人及所有幫助過我的人，謝謝你們！

林哲宇 謹致於

南華大學資訊管理學系

資訊安全 實驗室

2017年 1月

雲端儲存體做一個新型有效公開稽核方案

學生：林哲宇　　指導教授：周志賢

南　華　大　學　資訊管理學系碩士班

摘　　要

隨著雲端儲存體服務日益進步，且經由雲端資料的稽核方案可以讓使用者有效地稽核他們儲存在雲端伺服器裡面的資料有無完整性，以確保雲端伺服器是誠實的，因此客戶才可以完全確定放在雲端伺服器裡的資料沒有被任意竄改或更動，另外資料擁有者也能夠確保資料上傳到雲端伺服器時有進行資料的加密，因此公開稽核的服務方案是非常重要的。在本文中，我們提出一個有效率的公開稽核方案，主要是為了證明出惡意雲端伺服器不能偽造出不存在的消息塊，而且能讓客戶可以完全地相信雲端伺服器裡面的資料是安全的，另外我們還有做出證明計算大量資料的正確完整性，最後我們有實現證明我們的協議是非常有效率與安全的。

關鍵詞：雲端儲存體、公開稽核、橢圓曲線、資料完整性

A new efficient public auditing scheme for cloud storage

Student：Zhe-Yu Lin　　Advisors：Dr. Jue-Sam Chou .

Department of Information Management
The Graduated Program
Nan-Hua University

ABSTRACT

The cloud storage service grows rapidly due to that cloud data auditing scheme can enable cloud users to effectively verify the integrity of their outsourced data, for ensuring that the data are faithfully stored by the cloud server. The user can hence completely assure that the stored data is untampered or inadvertently changed. therefore, the function correctness of the auditing scheme is extremely important. In this paper, we propose such an efficient scheme, where a malicious server can't forge a nonexistent message block to be successfully verified by a challenger. The proposed scheme allows not only for several blocks, which is the case for many research in the literature but also for batch files stored in the cloud. The proof shows that our protocol is correct and can be fulfilled very efficiently.

Keywords: cloud storage, public auditing, elliptic curve, data integrity

目　錄

表　目　錄

圖　目　錄

1. Introduction

Cloud computing can provide several benefits such as, broad network access rapid elasticity, measured service, on-demand self-service, and resource pooling as mentioned by NIST [37-39]. It offers the possibility of improving system management efficiency and changes the current hardware and software design type for computer utilization. In the cloud, after the client had stored data, he typically does not retain the original data mainly due to the data consistency maintenance. This is because the client may cooperate on the stored data with his partners at different places. Therefore he must assure that any device can get the latest version of the data. In other words, for the data consistency client always trust the cloud services provider once he had stored data. It's unnecessary for the client to maintain the data file.

Therefore, a good cloud contractor needs to focus on the following three issues:
1. Better storage management efficiency.
2. To reduce a lot of hardware and software cost.
3. The client is allowed to access the data anywhere anytime.

For the benefits provided, cloud computing service is widely accepted in the real life. Recently, the public auditing technology for ensuring data integrity is obtaining more and more attention. People who may be equipped with different terminals such as phones, tablets, laptops, desktop computers, and some other devices can take advantage of low-cost cloud storage to store their data anytime anywhere. That is, they can access personal information which may be stored in large multinational corporations, independent on their real locations. With cloud computing, you can also build a private cloud, and work with multiple partners to fulfill collaborative design, product development, or order processing. The processing result can be instantly shared. Even, it can be applied to the public sector for public information publishing and many more. Although cloud computing makes our life more convenient, it brings us new security and privacy challenges due to that it is on the air. For this reason, many persons do not want to use cloud storage due to the serious security problems. They concern about the integrity of the outsourced when facing several factors that may result in data corruption. First, the cloud service providers are usually not fully trusted. They may update data without notifying the data owners. Second, the stored data may be broken because of the cloud server's fails, management errors, or the adversary attack. However, in order to preserve the good service reputation, cloud service provider may hide the data loss event. Therefore, the issues of data leakage and integrity on the cloud storage have

become the main concerns of cloud client. How to determine that the data stored in the cloud is complete and safe is a very important issue.

Without maintaining the data file at the client side, the cloud clients will lose the control of the data. This leads to the untrustness of the client to the cloud storage provider. Therefore, the cloud data integrity checking is necessary. To this end, there has been many researchers [1-3, 5-8, 9, 12,13, 15, 20, 21,31-33] working in this field, attempting to resolve the problem. In this article, we refer to these related technologies as Provable Data Possessing (PDP).

However, this study found that most of the PDP technologies in the literature are implemented with higher computation cost, because they used expensive cryptographic operations such as, RSA, bilinear pairing cryptosystems to fulfill their scheme (Pairing-Based Cryptosystem is referred to as PBC). According to National Institute of Standards and Technology (NIST) recommendations for the same security level, a RSA cryptosystem with key length 1024/2048 bits is equivalent to an elliptic curve cryptosystem with only 163/233 bits. This means that the output length of elliptic curve cryptographysystem which is directly related to the cost of communications is about 1/6 to 1/8 to the RSA cryptosystem. According to the literature [35-36], the computation cost of RSA cryptosystem is about 3.2 to 6 multiple to the ECC with the same security level. In Tble 1, we compare three cryptography systems in the dimensions of communication cost and computation costs. From the table shown, we can easily see that ECC obviously has advantage.

Table 1：Comparisons of 3 type's systems' communication and computation cost

cryptography system	Communication costs	Computing costs
Elliptic-Curve Cryptosystem	1	1
RSA	6~8	3.2
Pairing-Based Cryptosystem	1	7.5

Up to now, we only see lower cost scheme using ECC cryptosystem to design their protocol, but their scheme doesn't possess the whole nine properties mentioned in [48].

2. Literature review

Cloud computing has significantly changed the way of computing resources usage. It can provide dynamic service model for resource usage via the internet which mainly resorts to the broad network access, and can be configured to share resources in a timely manner thought the network. Enterprises and internetc lients take advantage of the essential characteristics of cloud computing such as, rapid network access and on-demand self-service, to commission their outsourced data to the cloud.

In this computing environment, storage service providers(CSP) to get rid of their own storage maintenance burden. Cloud server must have a trustable, manageable, and effective accessible storage infrastructure for clients to create, store, and update data. In the recent research [22,29,32,35,41-47], all assume that the CSP is partially trusted. Under this assumption, it's possible that the data's integrity is defective, because the client cannot control the outsourced data. Therefore, it's necessary for the client to perform the integrity checking for his outsourced data. However, often, the client computation capability is limited especially when the mobile devices become popular. To resolve this problem, a third party with better expertise and capabilities is delegated to measure the cloud storage reliability and validity, on behalf of the clients when needed. This model is called public auditing scheme.

In this model, there are three entities: Client, Third party auditor (TPA), Cloud server. Among them involves the information transfer process. The client delegates the right of data integrity verification to the TPA. TPA will challenge the cloud server for obtaining the integrity proof. Subsequently, often receiving the proof from the server, TPA will return this proof to the client, as illustrated in Fig.1. There are many areas of encryption work known as privacy protection public auditing (PPPA), for cloud storage system design [1-27].

Wang et al.[34] proposed a Knox: Privacy-Preserving Auditing for Shared Data with Large Groups in the cloud. He can give third-party auditors (TPA) users the ability to verify the integrity of their data without retrieving the entire data. In addition, Zhu et al.[8]propose an efficient approach based on probabilistic query and periodic verification for improving the performance of audit services. They claimed that their scheme is can support provable updates to

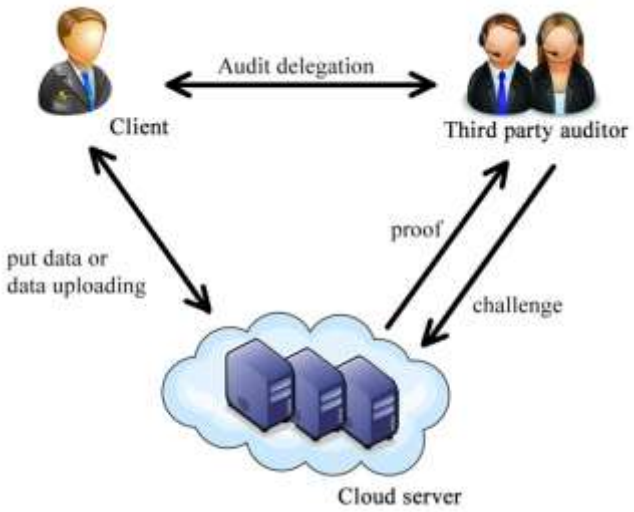


Figure 1：A traditional public auditing system model

outsourced data, and timely abnormal detection. But also leakage of the user's secret and some of the documents of the dynamic and efficient security audit services, which also allows the attacker has the advantage of easy pass.

As mentioned earlier, for blinding the server chooses a random element $r \in_R Z_p$ by using the same pseudorandom function and let $r = f_{k_3}(chal)$, where k_3 is a pseudorandom function key generated by the server for each auditing. It then calculates $R = u^r \in G$ and computes $\mu^* = \sum_{i=s_1}^{s_c} v_i m_i$, $\mu = \mu^* + \mathrm{rh(R)} \in Z_p$, and $\sigma = \prod_{i=s_1}^{s_c} \sigma_i^{v_i}$.

From the received (μ, σ, R), we can see that since $\sigma = \prod_{i=s_1}^{s_c} \sigma_i^{v_i}$ $= \prod_{i=s_1}^{s_c} (H(i)^{v_i} \cdot u^{\sum_{i=s_1}^{s_c} v_i m_i})^x$, a malicious server can regard $v_i s$ as constants and $m_i s$ as variables. He then computes $\mu^* = \sum_{i=s_1}^{s_c} v_i m_i$ using the constants $v_i s$ and the message blocks stored. That is, he can obtain an equation containing multiple variables, the $m_i s$, which in mathematics has more than one solution. This means that other than the original $m_i s$, the malicious server can find out some message blocks satisfying the

equation without alerting σ. We take S_c=3 as an example. Suppose the values of v_is are (6, 8, 9), and the values of m_is are (1, 4, 2) respectively, then the plane can be defined by $6x+8y+9z=56(=6m_1^*+8m_2^*+9m_3^*)$, where $m_i^*, i=1 \text{ to } 3$, are the forged message blocks. We know that this plane also passes through the point (5, 1, 2). This implies that the malicious server can forge the message blocks from (1, 4, 2) to (5, 1, 2) without alerting the value σ. Moreover due to the independence between $\mu^*(=\sum_{i=s_1}^{s_c} v_i m_i)$ and R, the malicious server can even set $R'=u^{r'}$ and $\mu'=\mu^*+r'h(R')\in Z_p$ and sends (μ', σ, R') to *TPA*. *TPA* will accept the verification without detection. That is, the proof of the selected blocks is not unique. This might lead the scheme incur more vulnerabilities.

Recently, several articles proposed also have the same problem. For the interested readers, please refer to [34,40-45].

3. Security requirements in the system model

In this section, we describe the cloud storage system model and its security requirements. Then, we illustrate the goal of our design.

3.1 System model

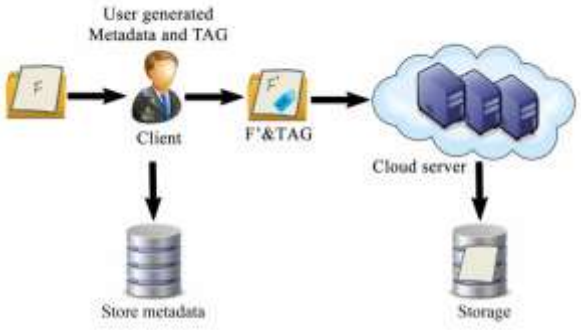
The following will describe the cloud storage integrity checking technology. The scenario is shown in Fig.2. We divide it into two layers: (a) data file preprocessing (b) data file verification.

(a) Data file preprocessing

When the client uploads the file F into the cloud, he first needs to generate the metadata, for instance cutting the file into blocks of the same size, then computing the corresponding metadata called tags. After this, the client will store F and all its tags to the cloud server and then delete them from its own storage.

(b) Data file Verification

When auditing, according to the metadata stored in the server's storage, the cloud server read out the corresponding file blocks and their tags to compute the proof. If the proof is correct, the cloud server is considered honesty.



(a) Data file preprocessing

(b) Verification the data file

Figure 2：Data has to prove technology architecture

3.2 Security requirement

In the model, three important security threats are defined [41, 49]. We show and explain them as follows.

(1)Impersonation attack: Although, the attacker hasn't the blocks and the corresponding tags, the attacker hasn't the blocks and the corresponding tags, he impersonates the cloud server trying to respond the correct proof.

(2)Replay attack: The malicious cloud server or attacker replays the previously record proof to fool the client, but indeed has changed the corresponding blocks secretly.

(3)Forgery attack: The malicious cloud server may forge a block m_i with its tag to satisfy the verification challenged from the auditor.

3.3 Our design goal

In this section, we present a new ECC-based auditing and scheme prove its security. The main reason why our scheme takes attempt to meet the security requirements is the reduction computing cost. In addition, me list list the following as our design goal:

1. Our preliminary construction is to design only for a single file block to satisfy proof verification. Then, it can be applied to the whole file's integrity checking.
2. It has minimum communication and computation cost.
3. The proof can be certified effectively. That is, once a client had retrieved damaged data, he can instantly identify it.
4. The role of auditor is moved from TPA to the user.

4. The proposed scheme

To ensure the integrity of stored data, we propose a new efficient public auditing scheme for cloud storage. We first define and show a list of used notations in Table2.

Table 2：The definitions of used notations

Notations table	
G_1	a additive cycle group on an Elliptic Curve with order q
$H_1(\cdot)$	a hash function mapping from $G_1 \rightarrow \{0,1\}^*$
$H_2(\cdot)$	a hash function mapping from $\{0,1\}^* \rightarrow Z_q$
s	client secret key
Y	system public key
m_i	the ith block of the shared cloud data file, that is $F=\{m_1,\ldots,m_n\}$
r_i	a random integer in Z_q^*
r_{i1}	a nonce chosen by the client
r_{i2}	a nonce chosen by the server
T_2	a timestamp
$Proof_i$	the proof a message sent to the client by the cloud server
P	a generator of group G_1
i,k_i	the challenge message sent to the cloud server by the client

Then, we illustrate our scheme as follows, It consists of four phases:

1. Setup phase: This phase includes the system private key and system public key generations.

2. TagBlock phase: In this phase, the client performs the data file pre-processing to generate block-tags. The client will then store the file with all its tags to the cloud server. It then deletes the stored file.
3. Challenge phase: In this phase, the client sends a challenge request to the server, The server generates an integrity proof corresponding to the set of challenged blocks, and returns it to the client.
4. ChallengeVerify phase: In this phase, when the client receives the proof from the server, he will perform calculations to see whether it is correct.

We now describe the four phases in details in the following and also show them in Figure3.

4.1 Setup phase

The system initialization generates the master secret and public keys, and system parameters. let G1 be an additive cyclic group of prime order q and P be a generator of G1 where q is a large prime which is at least 160 bits or more. Define two cryptographic hash functions with the mapping: H_1: $G_1 \rightarrow \{0,1\}^*$ and H_2: $\{0,1\}^* \rightarrow Z_q$, The client secret key is $s \in Z_q^*$, and public key is $Y = sP$. Afterwards, the system publishes the system parameters $\{G_1, P, q, Y, H_1, H_2\}$.

4.2 Tagblock generation phase

The main purpose of this phase is to verify whether the stored file block or blocks had been stored honestly. The preliminary step is to generate the corresponding tag for a single blocks m_i in a file F. The client first divides file F into n blocks of equal size.

We denote it as $F=\{ m_i \| 1 \le i \le n \}$. After that, for each block m_i the client generates its tag by choosing a random number $r_i \in Z_q$ and computes its $Tag_i (d_i, A_i, B_i, c_i, D_i)$, where as $d_i = H_1(s\ r_i\ m_i), A_i = r_i\ m_i P$, $B_i = r_i Y$, $c_i = H_2(s \| H_1(A_i)) m_i$ and $D_i = c_i\ B_i$. The client then calculates

$$v_i = r_i\, m_i + sH_1(B_i) + r_i\, d_i\, s \qquad \text{....(1)}$$

He then sends v_i, m_i, Tag_i to the server for its verifications.After receiving, the server checks to see whether $v_i P =? A_i + H_2(B_i) Y + d_i B_i$ holds. If it does, the cloud server stores m_i together with its tag Tag_i into the storage. We demonstrate the correctness of equation **(1)** as follows.

$$v_i P = (r_i\, m_i + sH_1(B_i) + r_i\, d_i\, s) P = A_i + H_1(B_i) + d_i\, B_i$$

Client (File)	**P: generator**	**Server**
Step phase Privkey = s Pubkey = $Y = sP$	$H_1: G_1 \rightarrow \{0,1\}^*$ $H_2: \{0,1\}^* \rightarrow Z_q$	
TagBlock generation phase $d_i = H_2(r_i \| m_i \| s)$, $A_i = r_i m_i P$, $B_i = r_i Y$, $c_i = H_2(s \| H_1(A_i)) m_i$ $D_i = c_i B_i$ $v_i = r_i m_i + sH_1(B_i) + r_i d_i s$ $m_i, d_i, A_i, B_i, v_i, c_i, D_i$		
	$m_i, d_i, A_i, B_i, v_i, c_i, D_i$ →	verify $v_i P = ? A_i + H_2(B_i) Y + d_i B_i$ store $m_i, Tag_i = \{d_i, A_i, B_i, c_i, D_i\}$
Challenge phase For each i, $1 \le i \le n$ choose r_{i1} and k_i	i, k_i, r_{i1} →	choose r_{i2} compute $l_i = r_{i1} r_{i2} d_i$ $t_i = H_2(T_2)$ compute t_i^{-1} $R_i = k_i d_i m_i B_i$ $a_i p = r_{i1} r_{i2} (d_i + t_i H_2(k_i d_i A_i)) P$ $Q = t_i r_{i1} r_{i2} Y$ $w_i = H_1(r_{i1} r_{i2} P)$ $Proof_i = \{ l_i P, R_i, a_i P, Q_i, W_i, T_2, t_i, t_i^{-1} \}$
	← $Proof_i = \{ l_i P, R_i, a_i P, Q, w_i, T_2, t_i, t_i^{-1} \}$	
Obtain $Proof_i$ Compute $t_i ? = H_2(T_2)$ $H_1(s^{-1} t_i^{-1} Q) ? = w_i$		
ProofVerify phase $l_i P = ? a_i P - t_i r_{i1} r_{i2} H_2(s^{-1} R_i) P$		

Figure 3：Data block m_i integrity checking.

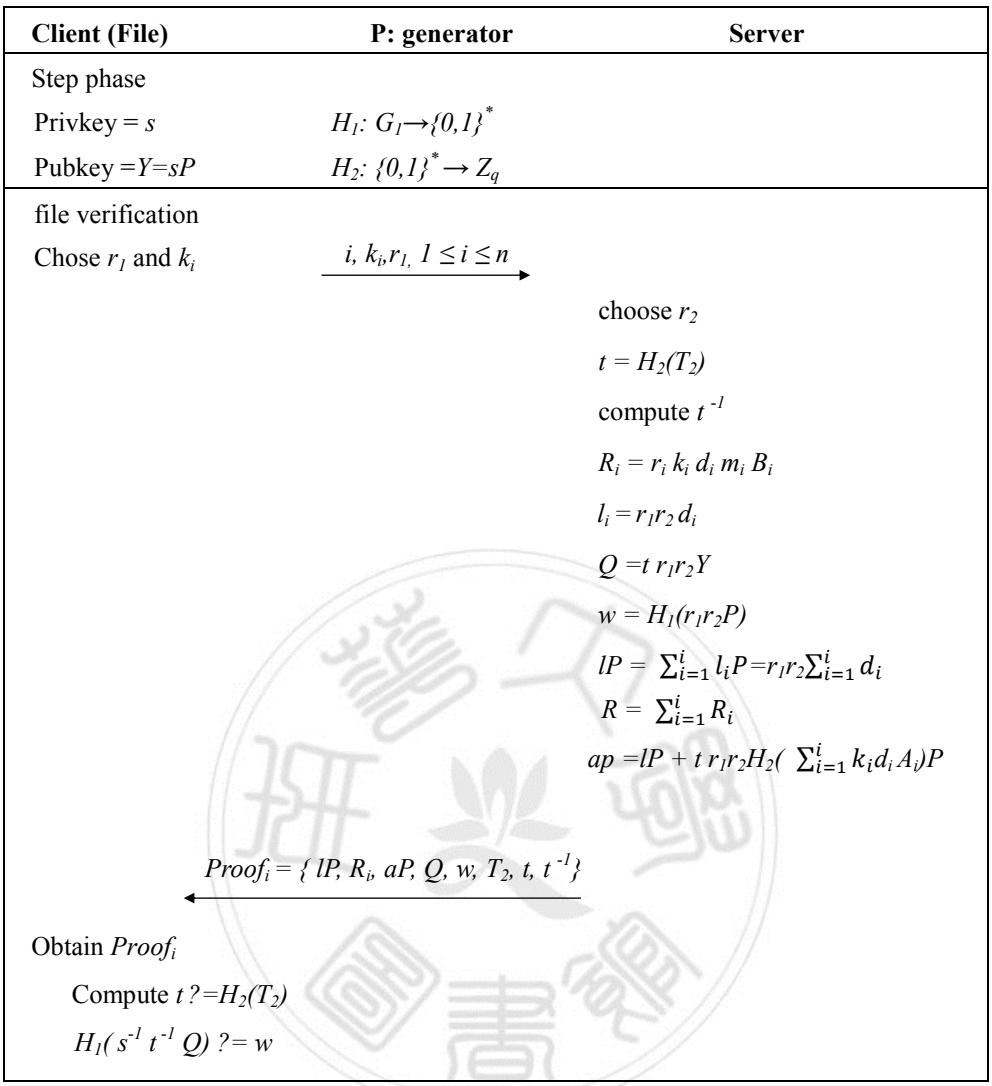


Figure 4：Batch verification for a file F

4.3 Challenge phase

After the storing of m_i, and Tag_i for $1 \le i \le n$, the client can examine whether the cloud server honestly store his delegated file blocks by randomly selects $i \in 1 \le i \le n$ and k_i, r_{i1} to challenge the server. Subsequently, the cloud server randomly selects r_{i2} and computes $l_i = r_{i1} r_{i2} d_i$.and $R_i = r_{i1} k_i d_i m_i B_i$. It then uses system timestamp T_2 to calculate $t_i = H_2(T_2)$, $Q = t_i r_{i1} r_{i2} Y$, $w_i = H_1(t_i r_{i1} r_{i2} P)$, and $a_i p = r_{i1} r_{i2} (d_i p + t_i H_2(k_i d_i A_i))P$

The server then generates $Proof_i = \{ l_i P, R_i, a_i P, Q, w_i, T_2, t_i, t_i^{-1} \}$ and send to the client for the client's checking.

4.4 Proof verification phase

In this phase, we demonstrate two kinds of verifications: (1)single block verification and (2)batch verifications for a file.

1.Single block verification

After receiving $Proof_i = \{ l_iP, R_i, a_iP, Q, w_i, T_2, t_i, t_i^{-1} \}$, the client computes and checks to see whether the $t_i=H_2(T_2)$. If so, he computes to see whether $H_1(s^{-1},t_i^{-1},Q) = w_i$ holds. If it holds, this implies that the client's challenge is realtime. The client them checks weather the following equation holds

$$
\begin{aligned}
l_iP \stackrel{?}{=} & \; a_iP - t_i r_{i1} r_{i2} H_2(s^{-1}R_i)P \\
& = a_iP - t_i r_{i1} r_{i2} H_2(s^{-1}k_i d_i m_i B_i)P \\
& = a_iP - t_i r_{i1} r_{i2} H_2(s^{-1}k_i d_i m_i r_i sP)\mathrm{P} \\
& = a_iP - t_i r_{i1} r_{i2} H_2(k_i d_i m_i r_i P)P \\
& = r_{i1} r_{i2} d_i \mathrm{P} + t_i r_{i1} r_{i2} H_2(k_i d_i A_i)\mathrm{P} - t_i r_{i1} r_{i2} H_2(k_i d_i m_i r_i P)P \\
& = l_i \mathrm{P} + t_i r_{i1} r_{i2} H_2(k_i d_i r_i m_i \mathrm{P})\mathrm{P} - t_i r_{i1} r_{i2} H_2(k_i d_i m_i r_i P)P \\
& = l_i \mathrm{P} \qquad \textbf{(2)}
\end{aligned}
$$

2.Bath verification for a file

The client can also perform batch auditing for the whole file F by launching the challenge for each block. The server computes $t = H_2(T_2)$, t^{-1}, $Q = t\, r_1 r_2 Y$,
$w = H_1(r_1 r_2 P)$, $li = r_1\, r_2\, d_i$,
$l_i = r_1\, r_2\, d_i$
$lP = \sum_{i=1}^{i} l_i P = r_1 r_2 \sum_{i=1}^{i} d_i P$,
$R = \sum_{i=1}^{i} R_i = \sum_{i=1}^{i} l_i\, d_i\, m_i\, B_i$,
$aP = r_1 r_2 \sum_{i=1}^{i} d_i\, P + t\, r_1 r_2\, H_2(\sum_{i=1}^{i} k_i\, d_i\, A_i)P$.

The server then transfers these parameters $lP, R_i, aP, Q, w, T_2, t, t^{-1}$ to the client for verification. The client verifies whether $t=H_2(T_2)$. If so, he computes to see whether $H_1\left(s^{-1}t^{-1}Q\right)=w$ holds. If it holds, this implies that the client's challenge is realtime. The client then checks whether the following equation holds.

$lP ?= aP - tr_1 r_2 H_2(s^{-1}R)P$

$?= aP - H_2(s^{-1}R)Q$

If the equation holds, the client believe that the data is well maintained. The protocol is illustrated in Figure 4.

5. Security analyses

In this section, we will show that our public auditing protocol is secure by using the following security features.

(1) Impersonate attack

Assume that an attacker E can successfully forge $proof_i$ without having the block m_i, That is E can pretend the server to respond with correct $proof_i$ without m_i and its tag once he had received message from the client. However, without m_i and its tag, E cannot compute $l_i p, R_i, a_i P$ to pass the clients verification $l_i P = ? a_i P - r_{i1} r_{i2} H_2(s^{-1} R_i) P$.

(2)Replay attack

Assume that attacker E had record the two messages transferred between the client and server and launches a replay attack. That is when a client challenged the server with the same message 1, he can replay message 2 to fool the client that he is the cloud server, However, in our protocol, the timestamp T_2 and its hash $H_2(T_2)$ can thwart such an attack. Because the client will examine the freshness of T_2 which is binded into B to be checked By the client the using the equation $H_1(t^{-1}Q) = ? H_1(r_{i2}Y)$.

(3) Forgery attack

Forgery attack means that an attacker wants to forge block m_i's tag without the block. That is, he can choose a random forged block m_i, and compute its tag $d_i = H_1(r_i' m_i', s)$, $A_i = r_i' m_i' P$, $B_i = r_i' Y$ successfully. However, without the knowledge of s, E cannot compute d_i for generating the proof to pass the client's verification.

6. Comparisons and Discussion

In this section, we compare our proposed scheme with some of the others in the literature and discuss several issues about the advantage of employing ECC cryptosystem in the secure public auditing design of cloud storage.

6.1 Comparisons

We describe the reason why our scheme can satisfy the following properties and which represent the ideal goal of a data auditing protocol as described in [48]. We use Table 5 to show the comparison result with the other related works in these properties. From Table 5, we can see that our scheme outperforms the other schemes.

1. Batch auditing: Our scheme provides batch auditing, as shown in equation 3.
2. Public auditing: As long as we substitute the role of user to third party, our scheme also provide public auditing. That is, the secret s originally possessed by the user is now possessed by the third party.
3. Blockless verification: In own scheme, no actual data blocks are sent by the server to the client for verification in the consideration of communication overhead.
4. Privacy Preservation: Because the auditor does not know the client secret s.
5. Data Dynamics: When the data owner stores data into the cloud, the data owner can access the data to perform certain operations such as, insertion or deletions anytime anywhere. Therefore, the audit mechanism must support the data owner to operate the data dynamically.
6. Unbounded number of challenges: The client can present unlimited number of challenges to the server. In our proposal, the client can launch unlimited number of challenges to the server. Therefore, our scheme meet this requirement.
7. Non reproducibility: In own scheme, the server cannot pass the verification without the owner's actual data m_i and its tag.
8. Recoverability: There should exists mechanism which can spring back the original data by using available data. In our method, as long as server gives the block m_i's $Tag_i = \{d_i, A_i, B_i, c_i\}$, the client can recover m_i.
9. Adaptability: The auditing protocol should comply with virtual machines dynamic mutability.

From Table 5, we can see that our scheme outperform the others in the above six properties which an auditing scheme should possess.

Besides it can also easily seen that, other than the above six properties, our scheme

also meet the other three ones which are met show in figure of [48].

Table 3：A performance comparison

Protocols	Batch auditing	Public auditability	Blockless verification	Privacy preservation	Data dynamics	Unbounded number of challenges	ECC
Ateniese et al. (2007) [2]	No	Yes	Yes	No	No	No	No
Ateniese et al. (2008) [3]	No	Yes	Yes	No	No	No	No
Wang et al. (2013) [6]	No	No	No	No	No	No	No
Erway et al. (2009) [12]	No	No	No	No	Yes	Yes	No
Zheng and Xu (2012) [18]	No	Yes	No	No	No	No	No
Wang et al. (2009) [20]	No	No	No	No	No	No	No
Wang et al., (2011) [23]	Yes	Yes	Yes	No	Yes	Yes	No
Zhu et al. (2011) [27]	No	Yes	No	No	No	No	No
Yang and Jia (2013) [28]	Yes	Yes	No	Yes	Yes	No	No
Hao et al. (2011) [30]	No	Yes	No	Yes	Yes	No	No
Zhang et al. (2015) [41]	No	Yes	No	No	No	No	Yes
Ours	Yes	Yes	Yes	Yes	Yes	Yes	Yes

6.2 Discussion

In this paper, we use ECC to design a flexible, high-secure, more practical data storage integrity auditing protocol. This is mainly due to that ECC cryptography key length is far more less than the other public key cryptosystems (such as RSA, Elgamal, Bilinear pairing), and thus is far more faster to process at same security level, when compared with the others. Thus, our for applications such as, smart cards, mobile phones, wireless memory devices, such as NFC limited environment. Moreover, we do not use the traditional third-party cloud data storage auditing system architecture with the three roles:(Client, TPA, Cloud Server), It can operate efficiently with Minimum computication and communication overhead. Therefore, our scheme can save the client's cost in delegating the auditing to the third party. This allows the client to do any audit conveniently.

6.3 Future work

In future work, we will use ECC to design e-cash transacting using the card transaction mode of Near Field Communication (NFC), is a short-range high-frequency wireless communication technology that allows non-contact, peer-to-peer data transmission between electronic devices, and can operate fast and smoothly. With attractive properties, smart phones embedded with NFC become very popular. It brings mobile device client dynamic usage experience which lets client operate in a new interactive wireless way. At present, many researchers of NFC technology have gradually developed it with smart card (SC) to form the mobile payment scheme on the

market. In the mobile payment life cycle, the data is from the mobile device through the wireless network to reach the payment platform. Then, the payment instructions on the card are implemented to complete the payment action. We also can use the characteristics of ECC to design the efficient wireless payment transaction with same security level which needs more complex computation when using the other cryptosystems such as, RSA, Bilinear pairing and so on.

7. Conclusions

In this paper, we first propose the effective auditing scheme by ECC cryptosystem in cloud computing, which allows the client to audit the efficiently outsourced data. Our scheme not only has computation and communication advantage, but also the satisfies night desirable properties for data integrity auditing protocols, which are not yet complete of fully achievable at present, as insisted in Gary et al. Besides we also show the our scheme meet the security requirement auditing protocol needs in section 3.

In addition, we discuss that our technology can be adopted to NFC smart response to enforce wireless payment transaction, on the market.

Totally speaking, our proposal is competitive in modern cloud data auditing scheme.

8. References

[1] Ateniese, G., Burns, R., Curtmola, R., Herring, J., Khan, O., Kissner, & Song, D. (2011). Remote data checking using provable data possession. ACM Transactions on Information and System Security (TISSEC), 14(1), 12.

[2] Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, et al., 2007. Provable data possession at untrusted stores. In: Proceedings of the 14th ACM.

[3] Ateniese, G., Di Pietro, R., Mancini, L. V., & Tsudik, G. (2008, September). Scalable and efficient provable data possession. ACM. conference on computer and communications security, CCS 2007. p. 598–609.

[4] Husain, M. I., Ko, S. Y., Uurtamo, S., Rudra, A., & Sridhar, R. (2014). Bi directional data verification for cloud storage. Journal of Network and Computer Applications, 45, 96-107.

[5] Wang, C., Wang, Q., Ren, K., & Lou, W. (2010, March). Privacy-preserving public

auditing for data storage security in cloud computing. In INFOCOM, 2010 proceedings IEEE (pp. 1-9). .

[6] Wang, C., Chow, S. S., Wang, Q., Ren, K., & Lou, W. (2013). Privacy-preserving public auditing for secure cloud storage. Computers, IEEE Transactions on, 62(2),362-375.

[7] Worku, S. G., Xu, C., Zhao, J., & He, X. (2013). Secure and efficient privacy-preserving public auditing scheme for cloud storage. Computers & Electrical Engineering40, (2014), 1703-1713. .

[8] Zhu, Y., Hu, H., Ahn, G. J., & Yau, S. S. (2012). Efficient audit service outsourcing for data integrity in clouds. Journal of Systems and Software, 85(5), 1083-1095

[9] Zhu, Y., Hu, H., Ahn, G. J., & Yu, M. (2012). Cooperative provable data possession for integrity verification in multi-cloud storage. Parallel and Distributed Systems, IEEETransactions on, 23(12), 2231-2244.

[10] Zhu, Y., Ahn, G. J., Hu, H., Yau, S. S., An, H. G., & Hu, C. J. (2013). Dynamic audit services for outsourced storages in clouds. Services Computing, IEEE Transactions on, 6(2), 227-238.

[11] Xu, C., He, X., & Abraha-Weldemariam, D. (2012). Cryptanalysis of Wang's auditing protocol for data storage security in cloud computing. In Information Computing and Applications (pp. 422-428). Springer Berlin Heidelberg.

[12] Erway, C., Küpçü, A., Papamanthou, C., & Tamassia, R. (2009, November). Dynamic provable data possession. In Proceedings of the 16th ACM conference on Computer and communications security (pp. 213-222). ACM.

[13] Di Pietro, R., & Sorniotti, A. (2012, May). Boosting efficiency and security in proof of ownership for deduplication. In Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security (pp. 81-82). ACM.

[14] Bowers KD, Juels A, Oprea A. HAIL: a high-availability and integrity layer for cloud storage. In: Proceedings of the 6th CM conference on computer and communications security, CCS 2009. p. 187–98.

[15] Deswarte Y, Quisquater J-J, Saïdane A. In: Jajodia S, Strous L, editors. Remote integrity checking: integrity and internal control in information systems VI, vol. 140. Boston: Springer; 2004. p. 1–11.

[16] Dodis Y, Vadhan S, Wichs D. Proofs of retrievability via hardness amplification. In: Reingold O, editor. Theory of cryptography, vol. 5444. Berlin/Heidelberg: Springer; 2009. p. 109–27.

[17] Shacham H, Waters B. Compact proofs of retrievability. In: Pieprzyk J, editor. Advances in cryptology – ASIACRYPT 2008, vol. 5350. Berlin/Heidelberg: Springer; 2008. p. 90–107.

[18] Zheng Q, Xu S. Secure and efficient proof of storage with deduplication. In: Proceedings of the second ACM conference on data and application security and privacy, CODASPY 2012. p. 1–12.

[19] Zheng Q, Xu S. Fair and dynamic proofs of retrievability. In: Proceedings of the first ACM conference on data and application security and privacy, CODASPY 2011. p. 237–48.

[20] Wang Q, Wang, Li J, Ren K, Lou W. Enabling public verifiability and data dynamics for storage security in cloud computing. In: Backes M, Ning P, editors. Computer security – ESORICS 2009, vol. 5789. Berlin/Heidelberg: Springer; 2009. p. 355–70.

[21] Zhuo H, Sheng Z, Nenghai Y. A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability. IEEE Trans Knowl Data Eng 2011;23:1432–7.

[22] Chunxiang X, Xiaohu H, Daniel-Abraha W. Cryptanalysis of Wang's auditing protocol for data storage security in cloud computing. In: Liu C et al., editors. Information computing and applications, vol. 308. Berlin Heidelberg: Springer; 2012. p. 422–8.

[23] Wang Q, Wang C, Ren K, LouW, Li J. Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans Parallel Distrib yst 2011;22:847–59

[24] Juels, A., & Kaliski Jr, B. S. (2007, October). PORs: Proofs of retrievability for large files. In Proceedings of the 14th ACM conference on Computer and communications security (pp. 584-597). ACM.

[25] Chen, L. (2013). Using algebraic signatures to check data possession in cloud storage. Future Generation Computer Systems, 29(7), 1709-1715.

[26] Yu, Y., Ni, J., Ren, J., Wu, W., Chen, L., & Xia, Q. (2014). Improvement of a Remote Data Possession Checking Protocol from Algebraic Signatures. In Information Security Practice and Experience (pp. 359-372). Springer International Publishing.

[27] Zhu, Y., Wang, H., Hu, Z., Ahn, G. J., Hu, H., & Yau, S. S. (2011, March). Dynamic audit services for integrity verification of outsourced storages in clouds. In Proceedings of the 2011 ACM Symposium on Applied Computing (pp. 1550-1557). ACM

[28] Yang, K., & Jia, X. (2013). An efficient and secure dynamic auditing protocol for data storage in cloud computing. Parallel and Distributed Systems, IEEE Transactions on, 24(9), 1717-1726., K., & Jia, X. (2012). Data storage auditing service in cloud computing: challenges, methods and opportunities. World Wide

Web, 15(4), 409-428.
[29] Yang, K., & Jia, X. (2013). An efficient and secure dynamic auditing protocol for data storage in cloud computing. Parallel and Distributed Systems, IEEE Transactions on, 24(9), 1717-1726.
[30] Hao, Z., Zhong, S., & Yu, N. (2011). A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability. Knowledge and Data Engineering, IEEE transactions on, 23(9), 1432-1437.
[31] Itani, W., Kayssi, A., & Chehab, A. (2009, December). Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures. In Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on (pp. 711-716). IEEE.
[32] Kamara, S., & Lauter, K. (2010). Cryptographic cloud storage. In Financial Cryptography and Data Security (pp. 136-149). Springer Berlin Heidelberg.
[33] Yang, J., Wang, H., Wang, J., Tan, C., & Yu, D. (2011). Provable data possession of resource-constrained mobile devices in cloud computing. Journal of networks, 6(7), 1033-1040.
[34] Wang, B., Li, B., & Li, H. (2012, January). Knox: privacy-preserving auditing for shared data with large groups in the cloud. In Applied Cryptography and Network Security (pp. 507-525). Springer Berlin Heidelberg.
[35] D. Johnson and A. Menezes, "The Elliptic Curve Digital Signature
[36] Algorithm (ECDSA)," Centre for Applied Cryptographic Research, University of Waterloo, August 1999.
[37] NIST, Digital Signature Standard (DSS), FIPS 186-3, June 2009,
[38] http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf
[39] NIST, Recommendation for Key Management - Part 1: General(Revised), Special Publication 800-57, March 2007, http://csrc.nist.gov/groups/ST/toolkit/documents/SP800-57Part1_3-8-07.pdf
[40] Liu C, Yang C, Zhang X, Chen J. External integrity verification for outsourced big data in cloud and IoT: A big picture. Future Generation Computer Systems 49, (2015), 58-67.
[41] Zhang J, Dong Q. Efficient ID-based public auditing for the outsourced data in cloud storage. Information Sciences 343-344, (2016), 1-14.
[42] Yang G, Yu J, Shen W, Su Q, Fu Z, Hao R. Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability. The Journal of Systems and Software 113, (2016), 130-139.
[43] Yu Y, Niu L, Yang G, Mu Y, Susilo W. On the security of auditing mechanisms for

secure cloud storage. Future Generation Computer Systems 30, (2014), 127–132.
[44] Yu Y, Au M-H, Susilo W, Ni J, Zhang Y, Vasilakos A-V, Shen J. Cloud Data Integrity Checking with an Identity-based Auditing Mechanism from RSA. Future Generation Computer Systems, (2016), 1-16.
[45] Yu Y, Ni J, Au M-H, Liu H, Wang H, Xu C. Improved security of a dynamic remote data possession checking protocol for cloud storage. Expert Systems with Applications 41, (2014), 7789–7796.
[46] Malina L, Hajny J, Dzurenda P, Zeman V. Privacy-preserving security solution for cloud services. Journal of Applied Research and Technology 13. (2015), 20-31.
[47] Tan, S., Jia, Y., 2015. NaEPASC: a novel and efficient public auditing scheme for cloud data. J. Zhejiang Univ. Sci. C 15 (9), 794–804.
[48] Garg, N., & Bawa, S. 2016. Comparative analysis of cloud data integrity auditing protocols. Journal of Network and Computer Applications, 66, 17-32.
[49] Alizadeh, M., Abolfazli, S., Zamani, M., Baharun, S., & Sakurai, K. (2016). Authentication in mobile cloud computing: A survey. Journal of Network and Computer Applications, 61, 59-80.