sensors

MDPI

Article

SM2-Based Offline/Online Efficient Data Integrity Verification Scheme for Multiple Application Scenarios

Xiuguang Li [1,2], Zhengge Yi [3], Ruifeng Li [2], Xu-An Wang [2], Hui Li [1,*] and Xiaoyuan Yang [2,3]

[1] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710126, China; lixiuguang00@126.com
[2] Cryptographic Engineering College, Chinese People's Armed Police Force Engineering University, Xi'an 710086, China; rfli46@163.com (R.L.); wangxazjd@163.com (X.-A.W.); yxyangyxyang@163.com (X.Y.)
[3] Key Lab of the Armed Police Force for Network and Information Security, Xi'an 710086, China; yi257172@163.com
* Correspondence: lihui@mail.xidian.edu.cn

Abstract: With the rapid development of cloud storage and cloud computing technology, users tend to store data in the cloud for more convenient services. In order to ensure the integrity of cloud data, scholars have proposed cloud data integrity verification schemes to protect users' data security. The storage environment of the Internet of Things, in terms of big data and medical big data, demonstrates a stronger demand for data integrity verification schemes, but at the same time, the comprehensive function of data integrity verification schemes is required to be higher. Existing data integrity verification schemes are mostly applied in the cloud storage environment but cannot successfully be applied to the environment of the Internet of Things in the context of big data storage and medical big data storage. To solve this problem when combined with the characteristics and requirements of Internet of Things data storage and medical data storage, we designed an SM2-based offline/online efficient data integrity verification scheme. The resulting scheme uses the SM4 block cryptography algorithm to protect the privacy of the data content and uses a dynamic hash table to realize the dynamic updating of data. Based on the SM2 signature algorithm, the scheme can also realize offline tag generation and batch audits, reducing the computational burden of users. In security proof and efficiency analysis, the scheme has proven to be safe and efficient and can be used in a variety of application scenarios.

Keywords: cloud storage; data integrity; public auditing; SM2

Citation: Li, X.; Yi, Z.; Li, R.; Wang, X.-A.; Li, H.; Yang, X. SM2-Based Offline/Online Efficient Data Integrity Verification Scheme for Multiple Application Scenarios. *Sensors* **2023**, *23*, 4307. https://doi.org/10.3390/s23094307

Academic Editor: Xiaojie Wang

Received: 14 March 2023
Revised: 17 April 2023
Accepted: 23 April 2023
Published: 26 April 2023

1. Introduction

Cloud storage technology is convenient and flexible, its use growing rapidly at home and abroad [1]. Big data from Internet of Things (IoT) devices and medical big data also use cloud storage technology to provide services. However, after users have stored data in the cloud, although they can thereby access convenient storage and management services, they also lose the power to control the data directly. Therefore, ensuring data integrity in the cloud has become a hot research topic for scholars [2]. Data integrity verification technology uses cryptography-related technology to design appropriate schemes that convince users that their data, when stored in the cloud server, is secure and complete, by means of a series of interactions between the auditor and the cloud server. Using this technique can effectively deter cloud service providers (CSP) from deliberately concealing the issues of data loss or corruption from users due to their fear of damaging their reputations. It also effectively stops users from unreasonably making accusations or claims against CSPs simply because of suspicion, thus effectively protecting the legitimate rights of both users and CSPs [3].

IoT devices have been widely used and have become a convenient and universal access terminal for Internet services. However, IoT devices have limited storage space and weak

computing power to support complex data computing and big data storage [4]. Therefore, the cloud, with its powerful computing potential and storage capacity, is generally used to expand the functions of IoT devices so that IoT devices can obtain massive data storage and strong data analysis capabilities. As one of the main service areas of cloud computing, the cloud storage mode of IoT device data allows IoT users to store their data in the cloud to compensate for the lack of storage space on IoT devices [5]. However, with the loss of the physical ownership and control of outsourced data, IoT device users are concerned about the integrity of their data. Therefore, it is necessary to conduct an integrity audit on the data of IoT devices using the cloud storage mode. Compared with general cloud audit schemes, the design of data audit schemes for the IoT has higher requirements [6]. First, public verification is required. IoT devices are often resource-constrained and need to support complex calculations; therefore, audit schemes require a third-party auditor (TPA) to be able to verify data integrity on behalf of the users. Second, privacy protection is needed. Data privacy protection is the most important problem in the IoT's cloud auditing scheme. Over the duration of the scheme's implementation, the contents of the challenge file should remain confidential to the TPA. Many IoT-embedded devices will generate a large quantity of personal and private data information. If this sensitive information is exposed in the integrity verification process, the privacy of IoT users may be disclosed to the integrity verifier or to the public. Third, the scheme is required to be lightweight. Computing capacity, storage capacity, bandwidth, and other resources of IoT devices are often greatly limited. Thus, audit schemes with lower computing costs are more suitable for the IoT. Fourth, a batch audit is required. There are many types and numbers of IoT devices. The audit scheme must support batch audits for multiple users to quickly verify the integrity of the massive amount of IoT data.

The sources of healthcare-derived big data mainly include clinical big data generated during patients' medical treatment, health-related big data generated by wearable human health-monitoring devices, and biological big data generated by life sciences research and medical institutions. However, despite the large amount of data stored in medical databases, it is still not easy to comprehensively record information on all diseases. Since electronic medical records are not fully available, a large amount of data comes from manual records. Biases and incomplete content arising from the recording process, uncertainty in textual expressions, and incomplete data storage are the root causes of incomplete medical and health big data, so it is crucial to audit the integrity of medical data [7]. In addition, the integrity audit scheme of medical data needs to achieve privacy protection. Detailed personal information and the health status of patients are often directly recorded in healthcare big data, and these sensitive forms of data require greater privacy protection. Finally, dynamic update capabilities are also necessary. Patients' consultation and onset times involve process changes, while the waveform and image data of the medical examination are time series. The patient's health status is not static but is always in a state of dynamic change.

Motivation. We believe that it is essential and urgent to design a data integrity verification scheme that can be better applied to the cloud storage environment of IoT data and medical data, and the most appropriate scheme must meet the following functions:

(1) Public auditing: anyone can perform the audit. Generally, experienced and skilled TPAs are entrusted by the users to perform the audit task.
(2) Dynamic updating of cloud data: users can insert, delete, and modify the data stored in the cloud at any time.
(3) Privacy protection: the TPA cannot know the contents of the user data. It is also preferable that CSP should not know the contents of the user data.
(4) Lightweight computation: the users' computational overhead should be as small as possible.
(5) Batch audits for multiple users: the most appropriate scheme is able to implement batch audits for multi-user data.

However, we found that most existing cloud storage schemes do not meet the above five conditions well. Therefore, we designed an efficient offline/online data integrity

verification scheme. The proposed scheme is not only applicable to the integrity audit of cloud data but is also applicable to the integrity verification of IoT data and medical data.

2. Related Works

In early remote data integrity verification schemes, the auditor needs to download all data from the cloud and use the locally stored metadata to confirm the integrity, which requires high communication and calculation costs and takes a long time to achieve, resulting in a great waste of computing power. In 2007, Ateniese et al. [8] proposed the first provable data possession (PDP) scheme. Their scheme divided the data files into blocks. The auditor only needed to download partial data blocks from the CSP to verify the integrity of all data, with a high probability. For 1,000,000 4 KB blocks, assuming that 1% of the blocks have been deleted or tampered with by the CSP, the auditor only needs to verify the integrity of 460 blocks to judge the integrity of all data with a greater than 99% confidence probability. In 2007, Juels et al. [9] first proposed the proofs of a retrievability scheme to audit data. Their scheme used error correction codes and sampling detection technology to recover the damaged data after detecting that the integrity of the cloud data was damaged. However, their scheme does not support public auditing, and the number of audits is limited.

With the increasing demands of users, scholars have expanded various functions based on the scheme proposed by Ateniese et al. In their study [8], a dynamic data updating function is added to the cloud audit scheme to enable users to modify the data stored in the cloud more flexibly. If the cloud data are directly modified, the tag and index will not match, and subsequent verification work cannot be completed. Therefore, various appropriate data structures are proposed to achieve dynamic data updates. In order to prevent malicious auditors from colluding with CSPs or stealing users' data privacy, the random mask technology and blockchain technology are combined in cloud audit schemes to achieve security goals. In order to enable auditors to audit the data integrity of more than one user at a time, the batch audit function is added to the cloud audit scheme, which improves the efficiency of large-scale audits. In meeting the needs of one user after another, cloud data audit schemes gradually become more mature. However, with cloud storage technology, the existing cloud audit schemes are no longer fully applicable to the cloud storage environment for IoT and medical data.

The cloud audit scheme proposed in [10] constructs a multi-leaf authentication method based on the Merkle tree. The scheme can simultaneously authenticate multiple leaf nodes and realize batch data updates. The proposed scheme also supports log auditing. Users can verify whether the auditors perform their audit work honestly by checking the log files generated by auditors. However, the scheme does not mention comprehensive privacy protection, and there is a security problem wherein attackers can forge data tags to pass the audit. Hou et al. [11] designed a public audit protocol supporting blockless verification and batch verification practices; the protocol uses a chameleon certification tree to implement the efficient dynamic operations of outsourcing data, reduces the computational cost caused by data updates, and further improves audit efficiency. Nevertheless, the scheme does not describe how to achieve privacy protection for users and requires the computation of many bilinear pairs during the upload block verification and bulk audit phases. Based on the BLS signature, Mishra et al. [12] used a binomial binary tree and an indexed hash table data structure to construct an efficient and dynamically updated cloud audit scheme. However, the scheme cannot achieve batch audits.

Fan et al. [13] built a flexible auditing scheme that supports efficient dynamic updating based on the alliance blockchain. However, the scheme does not consider the batch audits of large-scale users. The ID-based offline/online PDP protocol that was constructed in [14] is based on an offline/online signature. The scheme supports batch verification and entire dynamic data operation but cannot realize data content privacy protection for cloud servers. The audit scheme introduced in [15] is based on an ID with compressed cloud storage, and it only uses encrypted data blocks in a self-verified way to audit the cloud data. Xu et al. [16]

introduced the concept of transparent integrity auditing. They proposed a concrete scheme, based on the blockchain, which does not rely on third-party auditors while freeing users from high communication costs in data integrity auditing.

Ji et al. [17] proposed an ID-based data integrity verification scheme with the designated auditor. In their scheme, only the auditor designated by the user could join the audit task, which improved the scheme's security compared with the previous ID-based audit schemes. However, the scheme needed to be more comprehensive. Li et al. [18] proposed an audit scheme based on a redactable signature. CSP can transform the signature directly, without the additional sanitizer, while sharing sensitive data. The signature can also be used to authenticate the source of sharing data. Lin et al. [19] proposed a consortium blockchain-based audit protocol. This protocol can check the abnormal behavior of auditors, but the scheme needs to be more comprehensive to achieve batch audits. In addition, during the audit process, the above schemes used numerous high-cost operations, such as the power index, point hash function, and bilinear mapping, thereby incurring high computing costs; thus, it cannot be applied to the environment of IoT data and medical data cloud storage completely.

Our Contributions. In this paper, we propose an efficient offline/online data integrity verification scheme for multiple application scenarios. Our contributions can be summarized as follows:

(1) Based on the SM2 signature algorithm and the SM4 block encryption algorithm, we have constructed an offline/online remote data audit scheme. The scheme supports dynamic data updates, comprehensive privacy protection, and batch audit capability. Based on the advantages of offline tags and scheme design, our scheme has low computational overheads and is suitable for lightweight environments.
(2) We have carried out a security analysis and proof of the scheme. The scheme is resistant to forgery attacks from the storage side and achieves comprehensive privacy protection; even the storage side cannot obtain the real content of the data.
(3) We analyzed the scheme's efficiency and compared the functions and computing costs with the existing schemes, proving the comprehensiveness of the scheme's functions and its high efficiency.

Organization. We have organized the rest of this paper as follows. Section 3 introduces the system model and the security model. The background knowledge used in the scheme's construction process and defines the proposed scheme's system and security model are introduced in Section 4. In Section 5, the concrete scheme is described. We analyze the scheme's performance and compared it with other schemes in Section 6. In Section 7, we conclude our work. We analyze the security of the scheme in Appendix A.

3. The System Model and Security Model

3.1. System Model

The system model of the scheme is shown in Figure 1. Three interacting entities are included: the CSP provides data storage services to users for payment, but it is not trusted and may delete data from the cloud or pry into the data privacy of its users for profit. The data owner (DO) is the owner of the data, uploading the data to the cloud to save their own storage overhead, but does not want the data privacy to be compromised. The TPA is a semi-honest auditor commissioned by users. They will faithfully perform the task of auditing the integrity of the data in the cloud, on the one hand, but on the other hand, they are curious about the content of the data.

The operation process of the proposed audit scheme includes the following algorithms:

(1) Setup: the CSP runs the algorithm, which inputs the security parameter, λ, and generates the public parameters $\{E, G, q, g\}$.
(2) KeyGen: the DO runs the algorithm, which outputs the private key, k_s, and the public key, k_p.

(3) OffTagGen: the DO runs the algorithm, which inputs k_s and the random numbers d_i, l, outputting the offline tags, r'_i, s'_i.
(4) OnTagGen: the DO runs the algorithm, which inputs r'_i, s'_i and data blocks m_i, then outputs the online tags r_i, s_i.
(5) ChalGen: the TPA runs the algorithm, which inputs the random number π and outputs the indexes, $\{i_j\}_{(1\le j\le c)}$.
(6) ProofGen: the CSP runs the algorithm, which inputs the $\left\{m_{i_j}, r_{i_j}, s_{i_j}, i_j\right\}_{(1\le j\le c)}$ and outputs the proof $\{\rho, s, r\}$.
(7) VerifyProof: the TPA runs the algorithm, which inputs the proof $\{\rho, s, r\}$ and outputs "true" or "false" to indicate the integrity of the data.

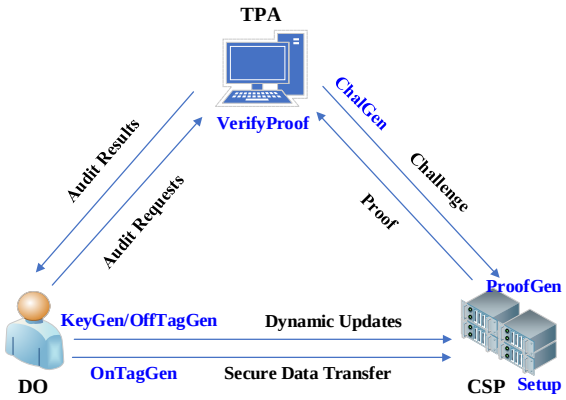

Figure 1. System model.

3.2. Security Model

In the existing data integrity audit schemes, security analysis often considers the CSP to be unreliable; it will forge tags in an attempt to pass the audit. Therefore, we mainly prove the unforgeability of the current scheme in the security analysis; this means that if the DO's data are corrupted, this must be detected by the interaction between the CSP and TPA when executing the scheme. That is, the CSP cannot forge integrity evidence and pass the data integrity audit under the condition that the data security is damaged; thus, it must carefully maintain the cloud data. We can define the unforgeability of the scheme with the following game:

Game: Assuming that C is the challenger, C runs the Setup algorithm to generate the system parameters and sends the system parameters to an adversary, A. In this security model, we assume that the adversary A has great privileges, although these privileges are unlikely to be possessed in a real situation. In Appendix A, we will show that even if the adversary, A, has all the privileges assumed herein, he/she is unable to break the auditing scheme proposed in this paper, thus demonstrating that the scheme has high security strength. Except for the target user that adversary A wants to attack, he/she can inquire about any other user's information. Specifically, A can ask the following predictor:

(1) Public key query: When A queries the public key of ID_w, C runs the KeyGen algorithm to generate k_{wp} and returns k_{wp} to A.
(2) Private key query: When A queries the public key of ID_w, C runs the KeyGen algorithm to generate k_{ws} and returns k_{ws} to A.
(3) Tags query: A can obtain the tag of m_{wi} under the public key k_{wp} of ID_w.

Based on the above query, after A is challenged, if A outputs the aggregate tag $\{\rho^*_w, s^*_w, r^*_w\}$ with the ID^*_w, ${k_{wp}}^*$, and the following conditions are met, then A wins the game. That is, our scheme is forgery-resistant.

Condition 1: The forged aggregation tags $\{\rho^*_w, s^*_w, r^*_w\}$ meet the verification equations.
Condition 2: There is no interruption of the public key query.
Condition 3: All the blocks m^*_{wi} of ID^*_w have been queried tags.

4. Preliminaries

4.1. Chinese Commercial Cryptography Algorithm

In 2010, the State Cryptography Administration of China released the elliptic curve-based SM2 cryptographic algorithm. The SM2 algorithm has high cryptographic complexity, fast processing speed, lower machine performance consumption, better performance, and more security. Its security has been proven by the authors of [20], and SM2 is more secure against generalized key substitution attacks. In 2012, the Security Commercial Code Administration Office of China released the SM4 block cipher standard. This is similar to AES-128, with simplified round key generation, and it is mainly used for data encryption. The encryption algorithms and decryption algorithms both use 32 rounds of a nonlinear iterative structure, the S box is a fixed 8-bit input and 8-bit output, the number of calculation rounds is large, and nonlinear changes are added, which make them more effective in defending against key-leaking Trojans [21]. The SM2/4 algorithm has been incorporated into the ISO/IEC international standard. Given its excellent security and performance, it is believed that it will be recognized or adopted by more and more organizations and individuals in China or outside of China.

Our scheme uses the SM2 digital signature algorithm to construct the audit scheme and the specific steps of the SM2 digital signature algorithm are as follows [22]. To facilitate understanding, we define and explain the various notations that appear in this paper in Table 1.

Table 1. Notations used in this paper.

Notations	Descriptions
λ	The system initialization parameter.
E	The elliptic curve.
G	The additive cyclic group.
q	A large prime number.
g	G.
k_s	The user's secret key.
k_p	The user's public key.
Z_q^*	The prime field.
d_i, m_i', l	Random numbers.
D_i, t	Intermediate parameters.
r_i', s_i'	Offline tags.
M	The user's data file.
$(m_1 \dots m_n)$	n data blocks.
ID	The identity of the file.
r_i, s_i	Online tags.
t_i	The timestamps of m_i.
v_i	The version numbers of m_i.
n	The number of total data blocks.
c	The number of challenged blocks.
per	The pseudo-random function.
π	The input parameter of per.
x_i, y_i	The coordinates of D_i.
ρ, s, r	The proof of data possession.

(1) Key generation: the selected elliptic curve equation is $y^2 = x^3 + ax + b$. Let g be the base point on the elliptic curve; the integer $k_s \in Z_q^*$ is randomly selected as the private key, then the public key $k_p = k_s \cdot g$ is calculated.

(2) Signature: Let the data to be signed be m. The signer first selects a random integer $d \in Z_q^*$, sets $d \cdot g = (x_1, y_1)$, and computes $r = m + x_1$, $s = (1 + k_s)^{-1}(d - rk_s)$; the signature of the message m is $\{r, s\}$.

(3) Verification: After receiving m and $\{r, s\}$, the verifier calculates $t = r + s$, $(x_1, y_1) = sg + tk_p$, and $r' = x_1 + m$. If the values of r' and r are equal, the signature is correct.

4.2. Dynamic Hash Table

Our scheme uses the dynamic hash table data structure proposed in Reference [23] to achieve a dynamic update of the data in the cloud. The dynamic hash table is a two-dimensional data structure, as shown in Figure 2.

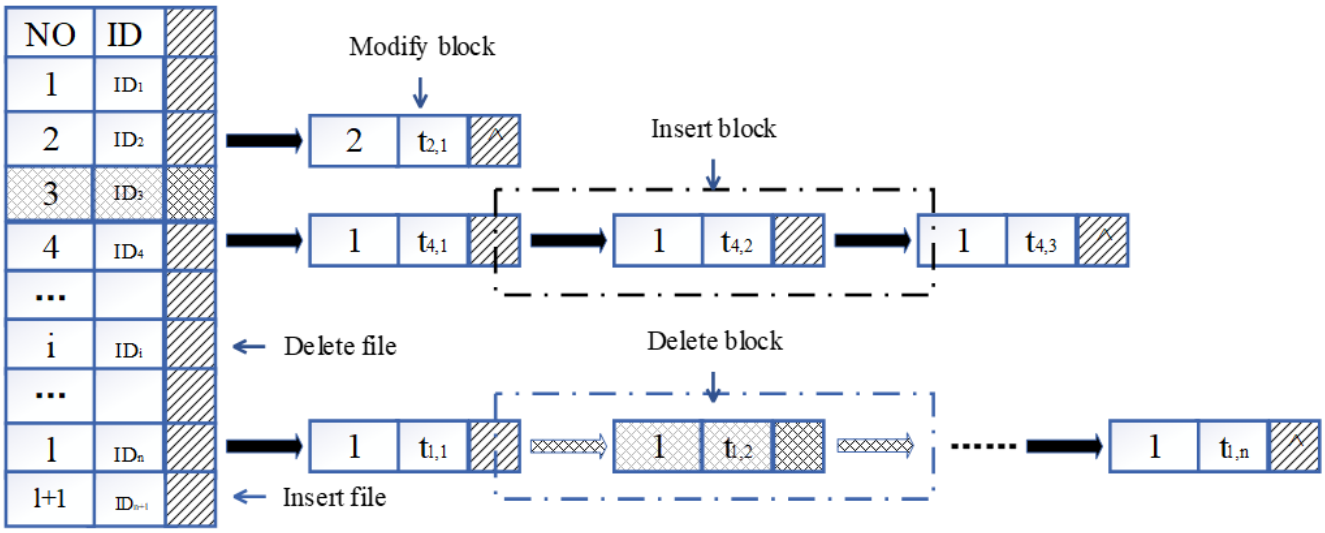

Figure 2. Dynamic hash table.

The table includes both file and data block elements. In the file element, NO. indicates the index value of the corresponding file, while ID indicates the identification of the corresponding file and a pointer of the first data block of this file. In the data block element, t_i indicates the timestamp of the data block, and v_i indicates the version number of the data block. The version number is initially set to 1 and its value is incremented by 1 for each change of the data block. The data block elements in the dynamic hash table are connected by a chain table, and each data block element is a node in the chain table, while each node includes the version information of the data block, the timestamp, and a pointer to the next node. Once the dynamic hash table is established, operations such as search, insert, deletion, and modification can be performed at either the file level or the data block level.

4.3. Elliptic Curve Discrete Logarithm Problem

The elliptic curve discrete logarithm problem (ECDLP): Let G be an additive cyclic group of elliptic curves of the order of the large prime q and set $g \in G$ as a generator. ECDLP means that, given $g, a \cdot g \in G$, an attacker A calculates $a \in Z_q^*$. The probability that the attacker A can solve the ECDLP in polynomial time is negligible:

$$\Pr\left[A(ag, g) = a : a \overset{R}{\leftarrow} Z_q^*\right] \leq \varepsilon \tag{1}$$

where ε represents the negligible probability; that is, it is computationally infeasible to solve the ECDLP.

5. SM2-Based Offline/Online Efficient Data Integrity Verification Scheme

In this section, we give a detailed description of the proposed scheme.

(1) Setup$(\lambda) \rightarrow (E, G, q, g)$: the CSP inputs the security parameter λ and generates the public parameters $\{E, G, q, g\}$. $E : y^2 = x^3 + ax + b \bmod p$ is the elliptic curve, p and q are large prime numbers, G is an additive cyclic group of order q defined on E, and g is the generator of the group, G.

(2) KeyGen $\rightarrow (k_s, k_p)$: the DO randomly selects $k_s \in Z_q^*$ as the private key and calculates $k_p = k_s \cdot g \in G$ as the public key.

(3) $\text{OffTagGen}(k_s, d_i, l) \rightarrow (r'_i, s'_i)$: we set the number of blocks for the file to n, and the block processing can improve the calculation efficiency and realize sampling verification. The DO randomly selects $\left\{d_i, l \in Z_q^*\right\}_{1 \le i \le n}$, calculates $D_i = d_i \cdot g \in G$, and sets the coordinates of D_i to $\{x_i, y_i\}$. For $i \in [1, n]$, the DO calculates:

$$r'_i = x_i + l \tag{2}$$

$$s'_i = (1 + k_s)^{-1}(d_i - r'_i k_s) \tag{3}$$

and obtains the offline tag $\left\{r'_i, s'_i\right\}_{1 \le i \le n}$.

(4) $\text{OnTagGen}(\left\{r'_i, s'_i\right\}, m_i) \rightarrow \{r_i, s_i\}$: the DO uses the SM4 block cipher algorithm to encrypt the data file M with identity ID, and then divides M into n blocks as $\left\{m_i \in Z_q^*\right\}_{1 \le i \le n}$, for each data block m_i. The DO generates the corresponding timestamp t_i and version number v_i, and calculates:

$$r_i = m_i + r'_i \tag{4}$$

$$s_i = s'_i - k_s(1 + k_s)^{-1} m_i. \tag{5}$$

The DO receives the online tag $\{r_i, s_i\}_{1 \le i \le n}$, then sends $\{\text{ID}, i, m_i, r_i, s_i, t_i, v_i\}_{1 \le i \le n}$ to CSP, sends $\{\text{ID}, i, t_i, v_i, D_i, l\}_{1 \le i \le n}$ to TPA, and finally delete the local data.

(5) $\text{ChalGen}(\pi) \rightarrow \left\{i_j\right\}$: the TPA selects the random number $\pi \in Z_q^*$ and sends it to the cloud server. Both parties take π as input, run the same pseudo-random function, per, and obtain the random c numbers $\left\{i_j\right\}_{(1 \le j \le c)}$ in $[1, n]$ as the indexes of the challenged data blocks.

(6) $\text{ProofGen}(\left\{m_{i_j}, r_{i_j}, s_{i_j}, i_j\right\}_{(1 \le j \le c)}) \rightarrow$ proof: after the CSP receives the audit request and generates the indexes of the challenged data blocks, it calculates $\rho = \sum_{j=1}^{c} m_{i_j}$, $s = \sum_{j=1}^{c} s_{i_j}$, and $r = \sum_{j=1}^{c} r_{i_j}$, and sends the proof $\{\rho, s, r\}$ to the TPA as the proof of data possession.

(7) $\text{VerifyProof}(\rho, s, r, k_p, D_{i_j}, x_{i_j}) \rightarrow$ true/false: the TPA receives the proof $\{\rho, s, r\}$, calculates $t = r + s, D = \sum_{j=1}^{c} D_{i_j}$, $x = \sum_{j=1}^{c} x_{i_j}$, and verifies whether the following equations hold:

$$s \cdot g + t \cdot k_p = D \tag{6}$$

$$x + \rho + cl = r. \tag{7}$$

If Equations (6) and (7) hold, the DO is informed that the data integrity is not compromised. The correctness of them is derived as follows:

$$\begin{aligned} & s \cdot g + t \cdot k_p \\ &= s \cdot g + (r + s) k_s \cdot g \\ &= \textstyle\sum_{j=1}^{c} s_{i_j}(1 + k_s) \cdot g + \sum_{j=1}^{c} r_{i_j} k_s \cdot g \\ &= \textstyle\sum_{j=1}^{c} (d_{i_j} \cdot g - r'_{i_j} k_s \cdot g - k_s m_{i_j} \cdot g + (m_{i_j} + r'_{i_j}) k_s \cdot g) \\ &= D \end{aligned} \tag{8}$$

$$\begin{aligned} & x + \rho + cl \\ &= \textstyle\sum_{j=1}^{c} x_{i_j} + \sum_{j=1}^{c} m_{i_j} + cl \\ &= \textstyle\sum_{j=1}^{c} (x_{i_j} + m_{i_j} + l) \\ &= \textstyle\sum_{j=1}^{c} (r'_{i_j} + m_{i_j}) \\ &= r \end{aligned} \tag{9}$$

(8) DynamicUpdate*:* our scheme enables dynamic update operations on the cloud data, including insertion, deletion, and modification. Since the number of data blocks involved in the dynamic update is small, offline tags are not required in the dynamic update process. When a data block, m_i, needs to be modified to m_j, the DO selects a random number, d_j, to calculate $D_j = d_j \cdot g \in G$, where the coordinate of D_j is set to $\{x_j, y_j\}$. Then, v_j and t_j are generated for the data block m_j, and the tags $r_j = m_j + x_j + l$ and $s_j = (1 + k_s)^{-1} \cdot (k_j - r_j \cdot k_s)$ are calculated. Finally, $\{\mathrm{ID}, i, m_j, r_j, s_j\}$ and $\{\mathrm{ID}, j, D_j, t_j, v_j\}$ are sent to the CSP and TPA, respectively. After receiving $\{\mathrm{ID}, i, D_j, t_j, v_j\}$, the TPA finds the $i - th$ node of the linked list corresponding to the file M in the dynamic hash table, and then replaces v_i and t_i with v_j and t_j. After receiving $\{\mathrm{ID}, i, m_j, r_j, s_j\}$, the CSP finds the location of m_i and replaces m_i, r_i, s_i with m_j, r_j, s_j.

When the DO needs to insert the data block m_j in front of the data block m_i, they first select a random number d_j to calculate $D_j = d_j \cdot g$ and set the coordinate of D_j as (x_j, y_j). Then, they generate v_j and t_j for data block m_j and calculate the tags $r_j = m_j + x_j + l$, $s_j = (1 + k_s)^{-1} \cdot (k_j - r_j \cdot k_s)$. Finally, the DO sends $\{\mathrm{ID}, i, m_j, r_j, s_j\}$ and $\{\mathrm{ID}, i, D_j, t_j, v_j\}$ to the CSP and TPA, respectively. After receiving $\{\mathrm{ID}, i, D_j, t_j, v_j\}$, the TPA finds the i-th node of the linked list corresponding to the file M in the dynamic hash table and inserts a new node after the $i - th$ node with the content v_j, t_j. After receiving $\{\mathrm{ID}, i, m_j, r_j, s_j\}$, the CSP finds the location of m_i, r_i, and s_i according to i, ID, and inserts m_j, r_j, s_j in front of them.

When the data block m_i needs to be deleted, $\{\mathrm{ID}, i\}$ is sent to the CSP and TPA. After receiving $\{\mathrm{ID}, i\}$, the TPA deletes the $i - th$ node of the linked list corresponding to the file M in the dynamic hash table. After receiving $\{\mathrm{ID}, i\}$, the CSP deletes m_i, r_i, and s_i according to i.

(9) BatchAudit: the scheme can implement a batch audit for multi-user cloud data. Each DO $\{u_w\}_{1 \le w \le x}$ randomly selects the private key, $k_{ws} \in Z_q^*$, and calculates the public key,$k_{wp} = k_{ws} \cdot g \in G$. The DO $\{u_w\}_{1 \le w \le x}$ randomly selects $\left\{d'_{wi}, l_w \in Z_q^*\right\}_{1 \le i \le n}$, calculates $D_{wi} = d_{wi} \cdot g \in G$, and sets the coordinates of D_i to $\{x_{wi}, y_{wi}\}$ for $i \in [1, n]$, calculates: $r'_{wi} = x_{wi} + l_w$, $s'_{wi} = (1 + k_{ws})^{-1}(d_{wi} - r'_{wi} k_{ws})$, and obtains the offline tag $\left\{r'_{wi}, s'_{wi}\right\}_{1 \le i \le n}$. The DO u_w uses the SM4 block cipher algorithm to encrypt the data file M_w with the identity, ID_w, and then divides M_w into n blocks, expressed as $\left\{m_{wi} \in Z_q^*\right\}_{1 \le i \le n}$; for each data block m_{wi}, the DO u_w generates the corresponding timestamp t_{wi} and version number v_{wi}, and calculates: $r_{wi} = m_{wi} + r'_{wi}$, $s_{wi} = s'_{wi} - k_{ws}(1 + k_{ws})^{-1} m_{wi}$, as the online tag $\{r_{wi}, s_{wi}\}_{1 \le i \le n}$, then sends $\{\mathrm{ID}_w, i_w, m_{wi}, r_{wi}, s_{wi}, v_{wi}, t_{wi}\}_{1 \le i \le n}$ to the CSP, sends $\{\mathrm{ID}_w, i_w, t_{wi}, v_{wi}, D_{wi}, l_w\}_{1 \le i \le n}$ to the TPA, and finally deletes the local data. The TPA selects a random number π as the parameter of per and sends it to the CSP. Both sides run the same pseudo-random function, per, and obtain the random number $i_{j\ (1 \le j \le c)}^{w}$ as the index of the challenged data block. After the CSP generates the indexes of the challenged data blocks, it calculates $\rho = \sum_{w=1}^{x} \sum_{j=1}^{c} m_{wi_j}$, $s = \sum_{w=1}^{x} \sum_{j=1}^{c} s_{wi_j}$, and $r = \sum_{w=1}^{x} \sum_{j=1}^{c} r_{wi_j}$, then $\{\rho, s, r\}$ will be sent to the TPA as the proof. The TPA receives the proof, computes $t = r + s$, $D = \sum_{w=1}^{x} \sum_{j=1}^{c} D_{wi_j}$, and $x = \sum_{w=1}^{x} \sum_{j=1}^{c} x_{wi_j}$, and verifies the following equations:

$$sg + \sum_{w=1}^{x} tk_{wp} = D \tag{10}$$

$$x + \rho + \sum_{w=1}^{x} l_w = r. \tag{11}$$

If Equations (10) and (11) hold, the TPA informs the total x DOs that data integrity has not been compromised. The correctness of them is derived as follows:

$$\begin{aligned} & sg + \sum_{w=1}^{x} tk_{wp} \\ & = \sum_{w=1}^{x} \sum_{j=1}^{c} s_{wi_j} g + \sum_{w=1}^{x} (k_{ws} \sum_{j=1}^{c} r_{wi_j} g + s_{wi_j} g) \\ & = \sum_{w=1}^{x} ((\sum_{j=1}^{c} s_{wi_j} g + k_{ws} s_{wi_j} g) + \sum_{j=1}^{c} r_{wi_j} k_{ws} g) \\ & = \sum_{w=1}^{x} (\sum_{j=1}^{c} (1 + k_{ws}) s_{wi_j} g + \sum_{j=1}^{c} r_{wi_j} k_{ws} g) \\ & = \sum_{w=1}^{x} (\sum_{j=1}^{c} (d_{wi_j} g - r'_{wi_j} \cdot k_{ws} g - k_{ws} m_{wi_j} g + r_{wi_j} k_{ws} g)) \\ & = \sum_{w=1}^{x} (\sum_{j=1}^{c} (d_{wi_j} g - r'_{wi_j} \cdot k_{ws} g - k_{ws} m_{wi_j} g + r_{wi_j} k_{ws} g)) \\ & = D \end{aligned} \tag{12}$$

$$\begin{aligned} & x + \rho + c \sum_{w=1}^{x} l_w \\ & = \sum_{w=1}^{x} \sum_{j=1}^{c} x_{wi_j} + \sum_{w=1}^{x} \sum_{j=1}^{c} m_{wi_j} + c \sum_{w=1}^{x} l_w \\ & = \sum_{w=1}^{x} (\sum_{j=1}^{c} x_{wi_j} + \sum_{j=1}^{c} m_{wi_j} + c l_w) \\ & = \sum_{w=1}^{x} (\sum_{j=1}^{c} (x_{wi_j} + m_{wi_j} + l_w)) \\ & = r \end{aligned} \tag{13}$$

6. Performance Analysis

In this section, the computational overhead of the scheme and the advantage of the offline/online tags are first analyzed, then we compare the functions of our scheme with existing schemes [10–14], which proves that our scheme is more suitable for the IoT data storage environment and medical data storage environment. The schemes in Refs. [10–14] are novel cloud data audit schemes proposed in recent years. They are not out of date and, at the same time, they have been tested by scholars in the past two years. Then, we compare the computational overhead of our scheme with the schemes in Refs. [10–14] numerically. Finally, we experimentally verify the results of the numerical analysis of computational overhead to visualize the performance of our scheme.

We set $G1$ and $G2$ to be the additive cyclic group of $E : y^2 = x^3 + ax + b \bmod p$ and the multiplicative cyclic group. p is a 512-bit prime number and q is a 160-bit prime number. The experiment was run on a 64-bit Windows 10 operating system with an i5 CPU, 2.5 GHz main frequency, and a 4 GB memory environment, using the JPBC library. After selecting a Type A elliptical curve and defining each operation, we ran each operation 10,000 times to obtain the average time overhead. The meaning of each operation and the corresponding time cost are shown in Table 2. To simplify the description, n is used here to denote the total number of data blocks, and c is used to denote the number of challenged data blocks. Because of the large values of n and c, we omit the operations' single occurrence in our analysis of the calculation overhead.

In the OffTagGen phase, the user needs to compute $D_i = d_i \cdot g$ and $r'_i = x_i + l$, so the computational overhead is about $n|M_{G1}| + n|A_Z|$. In the OnTagGen phase, the user needs to compute $r_i = m_i + r'_i$ and $s_i = s'_i - k_s(1 + k_s)^{-1} m_i$, so the computational overhead is about $n|M_Z| + 2n|A_Z|$. In the ProofGen phase, the CSP computes $\rho = \sum_{j=1}^{c} m_{i_j}$, $s = \sum_{j=1}^{c} s_{i_j}$, and $r = \sum_{j=1}^{c} r_{i_j}$, and the computational overhead is about $3c|A_Z|$. In the VerifyProof phase, after computing $t = r + s$, $D = \sum_{j=1}^{c} D_{i_j}$, and $x = \sum_{j=1}^{c} x_{i_j}$, the auditor also verifies the equations $sg + tk_p = D$ and $x + \rho + cl = r$, and the computational overhead is about $c|A_Z| + c|A_{G1}|$. After using the offline/online tags, the computational overhead of the user in the scheme is about $n|M_{G1}| + 3n|A_Z| + n|M_Z|$. If offline/online tags are not used, the user needs to calculate $D_i = d_i \cdot g$, $r_i = m_i + x_i + l$ and $s_i = (1 + k_s)^{-1} \cdot (d_i - r_i \cdot k_s)$; the computational overhead of the user is about $n|M_{G1}| + 3n|A_Z| + 2n|M_Z|$.

We compared our scheme with the existing certificateless schemes; the function comparison is shown in Table 3. As can be seen from Table 3, although other schemes are novel, their functions are not comprehensive. Our proposed scheme is the most comprehensive and the most suitable for the cloud storage environment of IoT data and medical data.

Table 2. Time cost of each operation.

Symbols	Description	Time Cost/ms
$\|A_Z\|$	computational cost of an addition on Z_q^*	0.0003
$\|M_Z\|$	computational cost of a multiplication on Z_q^*	0.0006
$\|E_Z\|$	computational cost of an exponentiation on Z_q^*	0.0226
$\|A_{G1}\|$	computational cost of an addition on $G1$	0.0055
$\|M_{G1}\|$	computational cost of a doubling on $G1$	0.7179
$\|M_{G2}\|$	computational cost of a multiplication on $G2$	0.0511
$\|H_Z\|$	computational cost of a hash operation to Z_q^*	0.0002
$\|H_{G2}\|$	computational cost of a hash operation to $G2$	1.1268
$\|E_{G2}\|$	computational cost of an exponentiation on $G2$	0.8107
$\|P\|$	Bilinear pair operations	5.8853

Table 3. Function comparison of each scheme.

	Dynamic Update	Batch Audit	Offline Tags	Privacy Protection Against the Cloud
Scheme [10]	Yes	Yes	No	No
Scheme [11]	Yes	Yes	No	No
Scheme [12]	Yes	Yes	No	Yes
Scheme [13]	Yes	No	No	Yes
Scheme [14]	Yes	Yes	Yes	No
Our scheme	Yes	Yes	Yes	Yes

The numerical computational overhead comparison of our scheme and other existing schemes is shown in Table 4. In the current cloud data audit schemes, the calculation overhead of the ProofGen and VerifyProof stages is borne by the CSP and TPA, respectively, while the calculation overhead of the TagGen stage is borne by the users themselves; the users only need to bear the calculation overhead in the TagGen stage. Because of the strong computing capability of the CSP and TPA, in the design of cloud data audit schemes, more emphasis should be placed on reducing the computing cost of the user side, that is, reducing the computing cost of the audit scheme in the TagGen stage. It can be seen from Table 4 that in the TagGen stage, the computational overhead of this scheme and the scheme in [14] is the smallest and is significantly smaller than other schemes. Therefore, this scheme and the scheme in [14] are more user-friendly and can be applied to equipment with lower computational power, which is more reasonable and efficient in its design. At the ProofGen stage, the computational overhead of our scheme is also significantly lower than that of other schemes. In the case where the number of challenged data blocks, c, increases gradually, the computational overhead of the other schemes increases at a faster and more dramatic rate than that of this scheme, and the advantages of our scheme are more significant.

Table 4. Comparison of the computational overhead.

	TagGen	GenProof	VerifyProof
Scheme [10]	$n(\|H_Z\| + \|M_{G2}\| + 3\|E_{G2}\| + s\|M_Z\| + s\|A_Z\|) \approx 2.4924n$	$cs\|M_Z\| + cs\|A_Z\| \approx 0.009c$	$c(\|M_Z\| + \|A_Z\|) + s(\|E_{G2}\| + \|M_{G2}\|) + 2\|P\| \approx 0.0009c + 2\|P\|$
Scheme [11]	$n(\|H_{G2}\| + 3\|E_{G2}\| + \|M_{G2}\|) \approx 3.61n$	$c(\|M_Z\| + \|A_Z\| + \|E_{G2}\| + \|M_{G2}\|) \approx 0.8627c$	$c(\|H\| + 2\|M_{G2}\| + 2\|E_{G2}\|) + 2\|P\| \approx 1.7238c + 2\|P\|$
Scheme [12]	$n(\|H_{G2}\| + \|E_{G2}\| + \|E_Z\|) \approx 1.9601n$	$c(\|M_Z\| + \|H_{G2}\| + 2\|E_{G2}\| + 2\|M_{G2}\| + \|A_Z\|) \approx 2.0406c$	$2\|P\|$
Scheme [13]	$n(s+1)(\|E_{G2}\| + \|M_{G2}\|) + n\|H_{G2}\| \approx 10.6066n$	$cs(\|M_Z\| + \|A_Z\|) + c\|M_{G2}\| + c\|E_{G2}\| \approx 0.8708c$	$(c+s)(\|M_{G2}\| + \|E_{G2}\|) + 2\|P\| + c\|H_{G2}\| \approx 1.9886c + 2\|P\|$
Scheme [14]	$n(2\|A_Z\| + \|M_Z\|) \approx 0.0012n$	$c(2\|M_Z\| + \|A_Z\| + \|E_Z\|) \approx 0.0241c$	$c\|A_Z\| + c\|M_Z\| + 3\|P\| \approx 0.0009c + 3\|P\|$
Our scheme	$n(2\|A_Z\| + \|M_Z\|) \approx 0.0012n$	$3c\|A_Z\| = 0.0009c$	$c\|A_Z\| + c\|A_{G1}\| \approx 0.0058c$

In order to test the performance of the scheme in terms of practical application and more intuitively compare the computational cost of each scheme, each scheme is run within the experimental environment, and the time costs in the stages of TagGen, ProofGen, and VerifyProof are recorded, as shown in Figures 3–5. The number of sectors s is set at 10 [23].

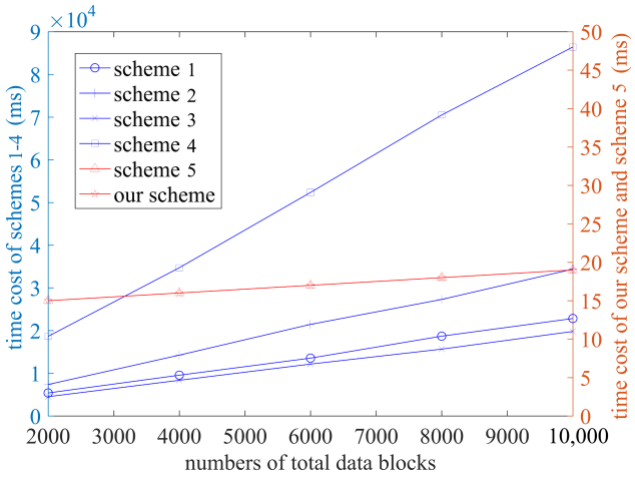

Figure 3. The time cost of the TagGen phase (Schemes 1–5 correspond to references [10–14], respectively).

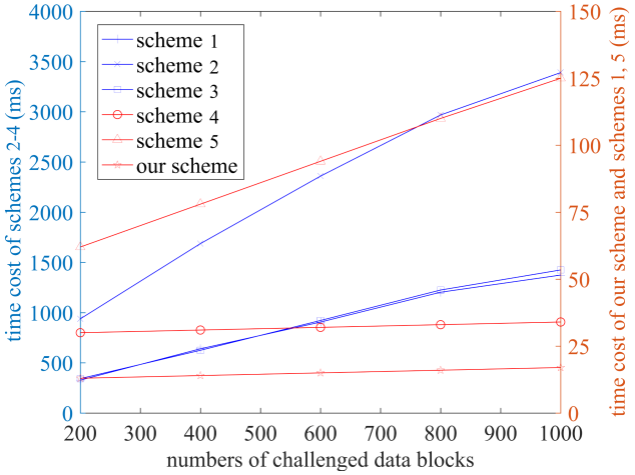

Figure 4. The time cost of the GenProof phase (Schemes 1–5 correspond to references [10–14], respectively).

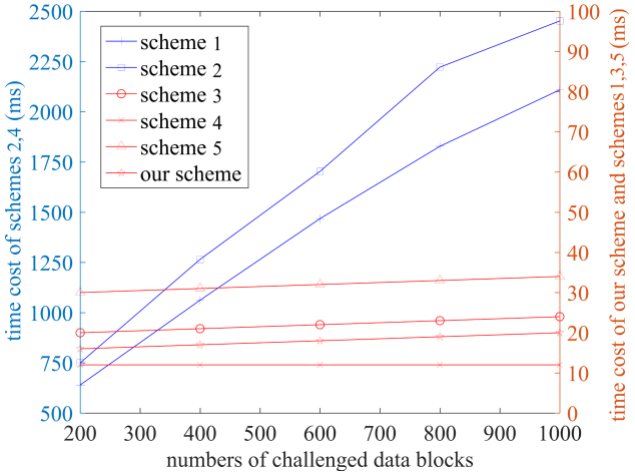

Figure 5. The time cost of the VerifyProof phase (Schemes 1–5 correspond to references [10–14], respectively).

Figure 3 shows the time cost of each scheme in the TagGen phase when the total number of data blocks is set to 2000, 4000, 6000, 8000, and 10,000, respectively. It can be concluded that the time cost of each scheme increases as the number of data blocks increases, but the time costs of the scheme in [14] and of our scheme do not increase significantly as the number of data blocks increases. This is due to the use of exponential operations in Refs. [10–13], which consume a significant amount of computational capacity. However, in our proposed scheme, the computation of tags is divided into two stages: OffTagGen and OnTagGen. For the users, their computation burden should mainly take into account the online tag computation. In our scheme, the online tag computation only requires simple addition and multiplication operations, resulting in a small computation overhead. Even with a large amount of data, it will not impose a significant computation burden on users. Under the conditions of the same number of data blocks, the time cost of the schemes in Refs. [10–13] is significantly higher than that of the scheme in Ref. [14] and in this scheme.

The time cost of the GenProof and VerifyProof phases is shown in Figures 4 and 5, when the number of challenged blocks is set to 200, 400, 600, 800, and 1000, respectively. It can be concluded that in the GenProof stage, the time cost of the schemes in Refs. [10,14] and our scheme is relatively low, and ours is the lowest. Scheme [12] has the highest time cost. In the VerifyProof stage, the time cost of our scheme and the schemes in Refs. [10,12,14] are significantly lower than that of the schemes in Refs. [11,13]. With the increase in the number of data blocks, the audit efficiency of our scheme becomes more prominent.

According to the above performance analysis, our scheme has more comprehensive functions and less time cost at each stage, especially in the TagGen stage, so it is more compatible with lightweight devices. Therefore, our scheme is more suitable for the IoT storage environment and medical data storage environment.

7. Conclusions

In this paper, we constructed an efficient SM2-based offline/online data integrity verification scheme for IoT and medical data. In the stage of preprocessing data of the scheme, users use the SM4 symmetric encryption algorithm to encrypt data. We used the encrypted data to generate tags and then uploaded them to the cloud, thus achieving full data privacy protection. In the scheme, users employ the SM2 signature algorithm to construct data tags in the uploading data stage. The scheme divided tags into offline parts and online parts. Users can calculate the offline tags in advance to reduce computing costs. The scheme uses a dynamic hash table to support the dynamic update of cloud data and realizes batch audits of multi-user data. It can adapt to the IoT and medical data storage environment. The theoretical safety analysis proves the scheme's safety. The high level of efficiency of the proposed scheme is demonstrated by comparing it with five existing schemes in terms of efficiency. In future work, we will focus on adding more functions to the existing audit schemes to meet the increasing needs of users in the cloud storage environment.

Author Contributions: X.L. and Z.Y. contributed equally to this work; X.L. was responsible for the writing of the article and the construction of the scheme. Z.Y. was responsible for the derivation of the formulas in the article and gave some significant ideas. R.L. was responsible for the validation and formal analysis. X.-A.W. was responsible for the collecting of resources related to this article. H.L. was responsible for the verification of the security of this article. X.Y. revised the finished manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China [62172436] and [62102452].

Data Availability Statement: All relevant data has been provided in the article. If someone have any other needs, he or she can contact the authors by email.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

Appendix A

In this section, we provide a provable security analysis of our scheme via the following theorems.

Theorem A1. ***(Unforgeability):*** *Under the random prediction model, it is assumed that adversary A breaks the proposed scheme with a nonnegligible advantage ε within time t. The execution times of A accessing the public key query, private key query, and tags query are* q_{pk}, q_{sk}, *and* q_t, *respectively. Then there is an algorithm C, which can solve the DL problem by calculating* $\varepsilon' \geqslant \varepsilon\left(\frac{1}{1-q_{\mathrm{pk}}}\right)^{q_{\mathrm{pk}}} \frac{n(n-1)\cdots(n-c+1)}{q_t(q_t-1)\cdots(q_t-c+1)}$ *in time* $t' < t + t_{\mathrm{inv}} + (3q_{\mathrm{t}}+1)t_a + 2q_{\mathrm{t}}t_m + q_{\mathrm{t}}t_M$, *where* t_{sm} *and* t_{inv} *represent the scalar multiplication time on* G_1 *and the inverse operation time on* Z_q^*, *respectively.*

Proof. A is the adversary and C is the DL problem challenger. Given that $(g, B)_{B=bg\in G}$, the goal of C is to use A to solve the DL problem to compute b.

C runs the Setup algorithm to generate the system parameters and sends the system parameters to A. A can ask the following predictor:

Public key query: C holds the list $F = \{\mathrm{ID}_w, k_{ws}, k_{wp}, c_w\}$; the initial list is empty. When A queries the public key of ID_w, if F has the public key of ID_w, then k_{wp} is returned. Otherwise, C randomly selects $c_w \in \{0, 1\}$, the probability of $c_w = 0$ is $\zeta = 1/q_{\mathrm{pk}}$; if $c_w = 0$, the challenge is terminated. If $c_w \neq 0$, C selects $B_w \in Z_q^*$, returns B_w to A as the public key k_{wp}, then adds $\{\mathrm{ID}_w, k_{ws}, k_{wp}, c_w\}$ to F.

Private key query: C holds the list $E = \{\mathrm{ID}_w, k_{ws}\}$ and the initial list is empty. When A queries the partial private key of ID_w, if E has ID_w, then k_{ws} is returned. Otherwise, C selects $k_{ws} \in Z_q^*$, returns k_{ws} to A, then adds $E = \{\mathrm{ID}_w, k_{ws}\}$ to E.

Tags query: C holds the list $\{\mathrm{ID}_w, m_{wi}, r_{wi}, s_{wi}, d_{wi}, l_w\}$ and the initial list is empty. When A queries the tag of $(\mathrm{ID}_w, m_{wi}, k_{wp})$, if L has $\{\mathrm{ID}_w, m_{wi}, r_{wi}, s_{wi}, d_{wi}, l_w\}$, then r_{wi} and s_{wi} are returned. Otherwise, C randomly selects $d_{wi}, l_w \in Z_q^*$, calculates $D_{wi} = d_{wi}g$, and sets the coordinates of D_{wi} to (x_{wi}, y_{wi}), then calculates $r_{wi} = m_{wi} + x_{wi} + l_w$, $s_{wi} = (1 + k_{ws})^{-1} \cdot (k_i - r_i \cdot k_{ws})$, returns r_{wi}, s_{wi} to A, and adds $\{\mathrm{ID}_w, m_{wi}, r_{wi}, s_{wi}, d_{wi}, l_w\}$ to L. □

Challenge: Let ID_w^* be a user's identity, where ID_w^* has never been queried as the private key. Let all blocks m_{wi}^* of ID_w^* have been queried tags and where C has queried the public key of ID_w^*. C runs the ChalGen algorithm to select the random number $\pi \in Z_q^*$ and send it to A. C and A take π as the input, run the same pseudo-random function, per, and obtain the random c numbers $i_{j(1\leq j\leq c)}$ in $[1, n]$ as the indexes of the challenged data blocks.

Forge: A calculates $\rho_w^* = \sum_{j=1}^{c} m_{wi_j}^*$, $s_w^* = \sum_{j=1}^{c} s_{wi_j}^*$, and $r_w^* = \sum_{j=1}^{c} r_{wi_j}^*$, and sends ρ_w^*, s_w^*, r_w^* to C; C calculates $D_w = \sum_{j=1}^{c} D_{wi_j}$ and $x_w = \sum_{j=1}^{c} x_{wi_j}$. A wins the game if $\{\rho_w^*, s_w^*, r_w^*\}$ pass the Equations (A1) and (A2):

$$s_w^* g + (s_w^* + r_w^*) B_w = D_w \tag{A1}$$

$$x_w + \rho_w^* + cl_w = r_w^*. \tag{A2}$$

Therefore:

$$\begin{aligned} &s_w^* g + (s_w^* + x_w + \rho_w^* + cl_w) B_w = D_w \\ &(s_w^* + x_w + \rho_w^* + cl_w) B_w = \textstyle\sum_{j=1}^{c} d_{wi_j}^* g - s_w^* g \\ &B_w = \textstyle\sum_{j=1}^{c} (d_{wi_j} - s_{wi_j}^*)(s_w^* + x_w + \rho_w^* + cl_w)^{-1} g \end{aligned} \tag{A3}$$

So C can calculate:

$$b = \textstyle\sum_{j=1}^{c} (d_{wi_j} - s_{wi_j}^*)(s_w^* + x_w + \rho_w^* + cl_w)^{-1} \tag{A4}$$

and solve the DL problem.

We define the terms as follows. Event E_1 indicates that there is no interruption in the public key query. Event E_2 indicates that the forged aggregation tags $\{\rho_w^*, s_w^*, r_w^*\}$ are valid. Event E_3 indicates that all the blocks m_{wi}^* of ID_w^* have been queried tags. Therefore:

$$\mathrm{Adv}_C^{\mathrm{DL}} = \Pr[\mathrm{E}_1\mathrm{E}_2\mathrm{E}_3] \geq \varepsilon \left(\frac{1}{1-q_{\mathrm{pk}}}\right)^{q_{\mathrm{pk}}} \frac{n(n-1)\cdots(n-c+1)}{q_t(q_t-1)\cdots(q_t-c+1)} \tag{A5}$$

and C uses the time t':

$$t' < t + t_{\mathrm{inv}} + (3q_{\mathrm{t}}+1)t_a + 2q_{\mathrm{t}}t_m + q_{\mathrm{t}}t_M \tag{A6}$$

We can reach the following conclusion: under the random prediction model, if A can break our scheme with a non-negligible ε within t, then there is an algorithm C that can solve the DL problem by the advantage $\varepsilon' \geqslant \varepsilon\left(\frac{1}{1-q_{\mathrm{pk}}}\right)^{q_{\mathrm{pk}}} \frac{n(n-1)\cdots(n-c+1)}{q_t(q_t-1)\cdots(q_t-c+1)}$ in time $t' < t + t_{\mathrm{inv}} + (3q_{\mathrm{t}}+1)t_a + 2q_{\mathrm{t}}t_m + q_{\mathrm{t}}t_M$.

Theorem A2. ***(Privacy protection):*** *The scheme supports privacy protection for the user's data and a private key against both the CSP and TPA.*

Proof. In the OnTagGen stage of the scheme, the user first employs the SM4 block encryption algorithm to encrypt the original data file and obtains the encrypted data blocks, m_i. The online tags are calculated using the encrypted data block, m_i, and the uploaded data are also the encrypted data. Therefore, even if the cloud stores a large quantity of data and tags, it is impossible to know the original data content. In the VerifyProof stage, TPA is unable to calculate the original data value from the aggregate data obtained and the aggregate tag. As a result, entities in the scenario other than the users cannot know the contents of the users' data. □

The user's private key, k_s, is only related to $\{s_i\}_{(1\leq i\leq n)}$ in $\{\mathrm{ID}, i, t_i, v_i, m_i, r_i, s_i\}_{(1\leq i\leq n)}$, stored at the cloud server. Therefore, the following system of equations will be listed when the cloud server tries to obtain the private key:

$$\begin{cases} s_1 = (1+k_s)^{-1} \cdot (d_1 - r_1 \cdot k_s) \\ s_2 = (1+k_s)^{-1} \cdot (d_2 - r_2 \cdot k_s) \\ \vdots \\ s_n = (1+k_s)^{-1} \cdot (d_n - r_n \cdot k_s) \end{cases} \tag{A7}$$

k_s and d_i are unknown to CSP. Since there are $n+2$ unknowns in n equations, the number of unknowns is always more than the number of equations so the private key k_s cannot be calculated.

References

1. Ji, Y.; Shao, B.; Chang, J.; Bian, G. Flexible identity-based remote data integrity checking for cloud storage with privacy preserving property. *Clust. Comput.* **2021**, *25*, 337–349. [CrossRef]
2. Gudeme, J.R.; Pasupuleti, S.; Kandukuri, R. Certificateless Privacy Preserving Public Auditing for Dynamic Shared Data with Group User Revocation in Cloud Storage. *J. Parallel Distrib. Comput.* **2021**, *156*, 163–175. [CrossRef]
3. Li, J.; Yan, H.; Zhang, Y. Certificateless Public Integrity Checking of Group Shared Data on Cloud Storage. *IEEE Trans. Serv. Comput.* **2021**, *14*, 71–81. [CrossRef]
4. Tian, Y.; Zhang, Z.; Xiong, J.; Chen, L.; Ma, J.; Peng, C. Achieving graph clustering privacy preservation based on structure entropy in social IoT. *IEEE Internet Things J.* **2022**, *9*, 2761–2777. [CrossRef]
5. Li, Q.; Xia, B.; Huang, H.; Zhang, Y.; Zhang, T. TRAC: Traceable and Revocable Access Control Scheme for mHealth in 5G-enabled IIoT. *IEEE Trans. Ind. Inform.* **2021**, *18*, 3437–3448. [CrossRef]
6. Xiong, J.; Ma, R.; Chen, L.; Tian, Y.; Li, Q.; Liu, X.; Yao, Z. A personalized privacy protection framework for mobile crowdsensing in IIoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4231–4241. [CrossRef]

7. Zhang, X.; Huang, C.; Zhang, Y.; Zhang, J.; Gong, J. LDVAS: Lattice-Based Designated Verifier Auditing Scheme for Electronic Medical Data in Cloud-Assisted WBANs. *IEEE Access* **2020**, *8*, 54402–54414. [CrossRef]
8. Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Kissner, L.; Peterson, Z.; Song, D. Provable Data Possession at Untrusted Stores. In Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07), Alexandria, VA, USA, 29 October–2 November 2007; pp. 598–609.
9. Juels, A.; Kaliski, B.S. Pors: Proofs of retrievability for large files. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 29 October–2 November 2007; pp. 584–597.
10. Guo, W.; Zhang, H.; Qin, S.; Gao, F.; Jin, Z.; Li, W.; Wen, Q. Outsourced Dynamic Provable Data Possession with Batch Update for Secure Cloud Storage. *Future Gener. Comput. Syst.* **2019**, *95*, 309–322. [CrossRef]
11. Hou, G.; Ma, J.; Liang, C.; Li, J. Efficient Audit Protocol Supporting Virtual Nodes in Cloud Storage. *Trans. Emerg. Telecommun. Technol.* **2020**, *32*, e3911. [CrossRef]
12. Mishra, R.; Ramesh, D.; Edla, D.R. BB-tree based secure and dynamic public auditing convergence for cloud storage. *J. Supercomput.* **2020**, *77*, 4917–4956. [CrossRef]
13. Fan, K.; Li, F.; Yu, H.; Yang, Z. A Blockchain-Based Flexible Data Auditing Scheme for the Cloud Service. *Chin. J. Electron.* **2021**, *30*, 1159–1166.
14. Rabaninejad, R.; Asaar, M.R.; Attari, M.A.; Aref, M. An identity-based online/offline secure cloud storage auditing scheme. *Clust. Comput.* **2020**, *23*, 1455–1468. [CrossRef]
15. Yang, Y.; Chen, Y.; Chen, F.; Chen, J. An Efficient Identity-Based Provable Data Possession Protocol with Compressed Cloud Storage. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1359–1371. [CrossRef]
16. Li, S.; Xu, C.; Zhang, Y.; Du, Y.; Chen, K. Blockchain-Based Transparent Integrity Auditing and Encrypted Deduplication for Cloud Storage. *IEEE Trans. Serv. Comput.* **2023**, *16*, 134–146. [CrossRef]
17. Ji, Y.; Shao, B.; Chang, J.; Xu, M.; Xue, R. Identity-based remote data checking with a designated verifier. *J. Cloud Comput.* **2022**, *11*, 7. [CrossRef]
18. Li, S.; Han, J.; Tong, D.; Cui, J. Redactable Signature-Based Public Auditing Scheme with Sensitive Data Sharing for Cloud Storage. *IEEE Syst. J.* **2022**, *16*, 3613–3624. [CrossRef]
19. Lin, Y.; Li, J.; Kimura, S.; Yang, Y.; Ji, Y.; Cao, Y. Consortium Blockchain-Based Public Integrity Verification in Cloud Storage for IoT. *IEEE Internet Things J.* **2022**, *9*, 3978–3987. [CrossRef]
20. Yang, A.; Nam, J.; Kim, M.; Choo, K.K.R. Provably-Secure (Chinese Government) SM2 and Simplified SM2 Key Exchange Protocols. *Sci. World J.* **2014**, *2014*, 825984. [CrossRef] [PubMed]
21. Wang, D.; Wu, L.; Zhang, X. Key-leakage hardware Trojan with super concealment based on the fault injection for block cipher of SM4. *Electron. Lett.* **2018**, *54*, 810–812. [CrossRef]
22. Yan, J.; Lu, Y.; Chen, L.; Nie, W. A SM2 Elliptic Curve Threshold Signature Scheme without a Trusted Center. *KSII Trans. Internet Inf. Syst. (TIIS)* **2016**, *10*, 897–913.
23. Tian, H.; Chen, Y.; Chang, C.; Jiang, H.; Huang, Y.; Chen, Y.; Liu, J. Dynamic-Hash-Table Based Public Auditing for Secure Cloud Storage. *IEEE Trans. Serv. Comput.* **2017**, *10*, 701–714. [CrossRef]