symmetry

MDPI

Article

Provably Secure Data Access Control Protocol for Cloud Computing

Ji Zhang [1,2,3,*], Anmin Chen [3] and Ping Zhang [2,3]

[1] Main Engine Department, China Airborne Missile Academy, Luoyang 471009, China
[2] Intelligent System Science and Technology Innovation Center, Longmen Laboratory, Luoyang 471000, China; zping@haust.edu.cn
[3] School of Mathematics and Statistics, Henan University of Science and Technology, Luoyang 471000, China; 220320080764@stu.haust.edu.cn
* Correspondence: zhangji@haust.edu.cn

Abstract: Currently, cloud storage servers are controlled by a third-party administrator. This semi-trusted approach gives rise to security concerns. Therefore, in cloud computing, some protocols use a key manager to encrypt the user's private data before uploading the data to the cloud. However, the security concerns that arise from the use of a key manager are not yet solved. In this respect, in this paper, a provably secure user cloud data access control protocol (DAC) is proposed based on existing cloud storage. Empirical tests confirm that the proposed approach is highly secure against adaptive selective ciphertext attacks and has excellent resistance to message attacks. A comprehensive performance evaluation, including time measurements, is conducted and the protocol is compared to other protocols, revealing the efficient file upload and download processes of the proposed approach. The results demonstrate the protocol's strong security, practicality, and operational efficiency.

Keywords: cloud computing; access control; DAC

Citation: Zhang, J.; Chen, A.; Zhang, P. Provably Secure Data Access Control Protocol for Cloud Computing. *Symmetry* **2023**, *15*, 2111. https://doi.org/10.3390/sym15122111

Academic Editors: Jia Hou, Jun Li and Xueqin Jiang

Received: 22 October 2023
Revised: 14 November 2023
Accepted: 21 November 2023
Published: 24 November 2023

1. Introduction

Cloud computing is a new service model that has arisen in recent years and become the focus of industry, academia, and government. Cloud computing brings together various computing infrastructures to form super-large-scale shared virtual resources and provides services to users via a network. This computing infrastructure can provide software services, hardware services, data storage services [1,2], etc. Users can store their massive data in the cloud, and cloud computing can provide effective services according to the needs of users [3]. Cloud storage has garnered widespread attention and support due to its cost-effectiveness and scalability; however, it is not without its challenges, notably in the domains of security, performance, and overall quality [4–9]. There is often a need to outsource some confidential data [10], for which privacy is the key concern [11,12]. For financial reasons, some cloud disk service providers may view or even disclose users' cloud disk data. The question of how to solve this security problem has become the research subject of many scholars.

For users, a cloud storage server is semi-trusted and, recently, a semi-trusted third party (key manager) and a new secure cloud storage protocol, File Assured Deletion (FADE), were proposed to deal with this problem [13]. One strategic approach to mitigate the security vulnerabilities inherent in cloud storage involves encrypting data prior to their upload, thereby increasing the protection against potential threats posed by the storage environment.The energy consumption, CPU utilization, and resource utilization of such encryption schemes have greatly decreased since Khan proposed the incremental version of proxy re-encryption scheme. To ensure data transmission, a robust multifactor authenticated key agreement scheme was proposed [14]. Formal and informal security analysis of this

approach confirms its ability to withstand known adversarial attacks, providing security and privacy for legitimate users. Seo et al. introduced a mediated certificateless public key encryption (mCL-PKE) scheme that circumvents the use of pairing operations [15]. This innovative scheme was subsequently implemented to devise a pragmatic resolution for the secure sharing of sensitive information within public cloud environments. Notably, this approach leverages the cloud infrastructure both as a secure storage repository and as a central hub for key generation.

Given the susceptibility of the FADE protocol to potential man-in-the-middle attacks, a Cloud Environment Data Security System (DaSCE) has been posited. This innovative system incorporates enhancements such as key exchange and digital signatures [16]. A protocol for a data security system situated within a cloud environment and reliant on a semi-trusted third party has also been introduced [17]. The protocol includes key management and access control, as well as file confirmation and deletion functions, employing the Shamir threshold secret sharing algorithm for key management.The latest works related to cloud data access protocols are listed and compared in Table 1.

Table 1. Cloud Data Access Protocols.

Protocol	Pros and Cons	Year	Ref.
Secret Sharing Group Key Management Protocol	Pros: The protocol reduces the potential security and privacy hazards associated with data. Cons: The group key management protocol may need enhancements to address forward and backward security issues.	2019	[18]
Certificateless Multi-Copy-Multi-Cloud Protocol	Pros: The protocol avoids the vulnerabilities of the certificateless approach. Cons: Insufficient attention has been given to the vulnerabilities inherent in this technology.	2020	[19]
Secure Access Control Protocol	Pros: The proposed protocol can protect fog nodes from outside attacks and inside attacks. Cons: The design process is quite complex.	2021	[20]
Blockchain-Assisted Security Protocol	Pros: This protocol shows efficient and enhanced security against various attacks. Cons: This protocol does not involve other environments.	2023	[21]

The key generation center is acknowledged as semi-trusted in two studies [16,17], but the security risks that it poses have not been fully mitigated. The key generation center has the ability to decrypt user data. Additionally, the authors of these studies have outlined the upload and download processes that data owners perform, revealing a deficiency in proper access control measures to control data access by consumers during file downloads. In recent years, many studies have focused on the challenges of cloud computing security. This article discusses in detail the challenges and issues of cloud computing security. Table 2 lists previous research articles that have emphasized the difficulties of cloud computing security.

Table 2. Objectives and Pros and Cons.

Objectives	Pros and Cons	Year	Ref.
Cloud Security	Pros: This article focuses on ensuring the security of cloud data. Cons: Insufficient attention has been given to the vulnerabilities inherent in this technology.	2023	[22]
Cloud Computing Security	Pros: This article focuses on improving cloud security by encrypting cloud data in cloud workers. Cons: Larger datasets are not discussed.	2023	[23]
Cloud Computing	Pros: This article provides a general overview of cloud computing. Cons: Weakness of resolving data breach issues.	2023	[24]
Cloud Data and Cloud Security	Pros: This article provides efficient secure communication. Cons: Weaknesses of this technology are not discussed.	2023	[25]

The introduction of a comprehensive secure cloud data sharing (SeDaSC) methodology was discussed in the academic work outlined in [26]. The research provides an algorithm for data owners to upload files and an access control algorithm for data consumers to download files securely.

However, regarding the encryption server as a trusted first party is too idealistic and not practical for users' private data, based on the existing cloud storage applications [27].

Our major contributions, as reported in this paper, are as follows.

1. A provably secure user cloud data access control (DAC) protocol has been proposed. Unlike other protocols, a third-party semi-trusted key generation center (which can be provided by network operators such as China Telecom, China Unicom, and China Mobile) is introduced to encrypt and upload data, eliminating the security threat of the key generation center. It also implements a series of complete access control functions, such as user file encryption uploads, friend download requests, and user authorization friend downloads.
2. An extensive analysis is conducted on the DAC protocol, proving that under the Larger Integer Factorization (IF) assumption, the file encryption algorithm of the protocol is indistinguishable under an adaptive Chosen Ciphertext Attack (IND-CCA). Under the condition of being a Gap Diffie–Hellman (GDH) group, we prove that the signature (i.e., authorization) in the protocol is Existential Unforgeability Against Adaptive Chosen Message Attacks (EUF-CMA) secure.
3. We conduct simulations to assess the performance of the DAC protocol, meticulously measuring the time allocated for diverse operations, and, subsequently, we juxtapose these results with those of alternative protocols proposed in the field. The simulation outcomes and comparative analysis affirm the effectiveness and practicality of the DAC protocol.

The remainder of this paper is organized as follows. Section 2 introduces some knowledge that needs to be used and provides a security model and proof of security. In Section 3, the details of the proposed DAC method are introduced. Section 4 demonstrates the security of the DAC protocol. The simulation and comparison results are presented in Section 5, and Section 6 summarizes this paper.

2. Preknowledge

2.1. Bilinear Pairing

Consider G_1 as a cyclic additive group generated by P, where the order of P is a prime number q. Similarly, define a cyclic multiplicative group G_2 with the same order q. Define the map e: $G_1 \times G_2 \rightarrow G_2$ with the following properties:

Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1, a, b \in Z_q$.

Non-degeneracy: There exists $P, Q \in G_1$, such that $e(P, Q) \neq 1$. In other words, the map does not send all pairs in $G_1 \times G_2$ to the identity in G_2.

Computability: An algorithm exists that efficiently computes $e(P, Q)$ for all combinations of $P, Q \in G_1$.

1. The Decisional Diffie–Hellman Problem (DDH): Given (P, aP, bq), compute abP.
2. The Decisional Diffie–Hellman Problem (DDH): Given (P, aP, bq, cP), compute c. If $c = ab(modq)$, (P, aP, bq, cP) is called a valid Diffie–Hellman tuple.

Definition 1. *The advantage of an algorithm A in solving the CDH in G is the probability*

$$Pr\{A(P, aP, bq)\} = abP : a, b \leftarrow Z_q.$$

We define a group G as a GDH group [28] if the DDH problem is polynomial-time solvable, yet there exists no probabilistic algorithm capable of solving the CDH problem with a non-negligible advantage within polynomial time.

2.2. System Model

As shown in Figure 1, the DAC model uses one tuple that consists of four components $<$ Us, KGC, Cloud, Fr $>$ to represent the following.

1. Us represents the user (data owner), which encrypts its data files through the key generation center and stores them in the cloud.
2. KGC represents the key generation center, which generates partial keys and provides users with partial encryption and decryption services. The function is similar to the key manager or encryption server in the aforementioned literature, but the method used is different.
3. The cloud stores encrypted files and related data.
4. Fr represents the friend (data consumer), who needs the user's cloud data to apply for file download authorization from the user.

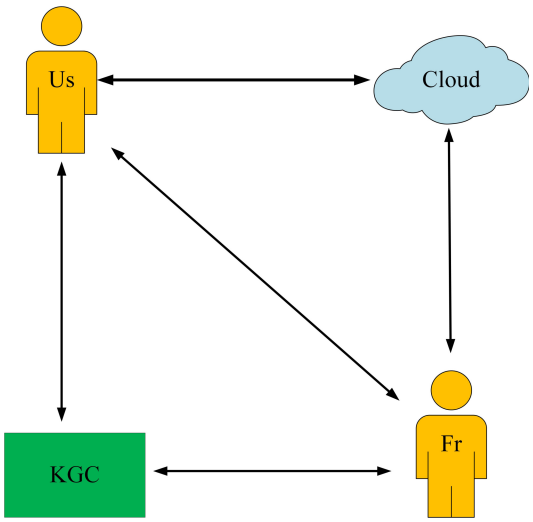

Figure 1. DAC model.

2.3. Security Model

The DAC system in this article will face three types of intruder attacks, which are abbreviated as A_1, A_2, and A_3.

A_1: Such adversaries have some keys to assist users in encrypting and decrypting data files, but they may still attack the communication between other entities in the system and intercept and try to decrypt user data files; A_1 is an intruder with semi-credible KGC capabilities.

A_2: This type of intruder stores the user's encrypted data files, but may still attack the communication between other entities in the system and try to decrypt the stored user data files; A_2 is an intruder who masters semi-trusted cloud resources.

A_3: This type of intruder sends a file download authorization request to the user, but it may still forge the user's file authorization and download and decrypt the user data file without the user's consent; A_3 intruder is a semi-credible Fr.

The following defines the IND-CCA security of the encryption algorithm in the protocol through the interactive game between the intruder A_1, A_2 and the challenger.

1. Initialization. The challenger constructs the DAC system, after which the intruder A_1 acquires the public key of DAC.
2. Enquiry. Intruder A_1 submits a decryption query to the challenger, who, upon decryption, furnishes the resulting plaintext to intrude A_1.
3. Challenge. Intruder A_1 generates two messages of equal length, denoted as F_0, F_1, and subsequently receives the ciphertext F_β from the challenger, where β is a random value between 0 and 1.
4. Guess. The intruder outputs β_1 and decides whether $\beta_1 = \beta$; if so, intruder A_1's attack is successful.

Definition 2. *Should intruder A, operating within polynomial time constraints, successfully breach the aforementioned security model with a negligible advantage denoted as* $Adv = |Pr\{\beta_1 = \beta\} - \frac{1}{2}|$*, it follows that the protocol described in this paper achieves IND-CCA.*

Through the interactive game between intruder A_3 and the challenger, the digital signature (authorization) in the protocol is defined to have existential unforgeability against chosen message attacks.

1. The challenger runs in the system to obtain the public and private keys and selects a random function $H(\cdot)$. Intruder A_3 obtains the public key.
2. Intruder A_3 can ask the challenger for $H(\cdot)$ and authorization for a message.
3. A_3 outputs a message and its signature, where A_3 has not requested the signature of the message from the challenger. If the signature (authorization) is verified, the intruder's attack is successful.

Definition 3. *In the event that an intruder* A_3*, operating within polynomial time constraints, successfully breaches the security model outlined above with a negligible advantage denoted as* $Adv = |Pr\{Exp = 1\}|$*, it is asserted that the protocol described in this paper is EUF-CMA.*

3. Data Access Control Protocol

DAC has introduced the key generation center (KGC), which allows users to encrypt files and upload them to the cloud. This prevents the cloud leakage of user data and is more secure than regular cloud disk applications. At the same time, the KGC helps users to encrypt and decrypt their data, sharing users' computational pressure [29].

In practical applications, a KGC can be provided by network operators (such as China Telecom, China Unicom, China Mobile, etc.), which are regarded as trusted in the literature [16], but network operators will also decrypt and disclose user data for commercial purposes, so it is more practical to treat the KGC as semi-trusted. The relevant symbols used in this section are shown in Table 3.

Table 3. Symbols and their meanings.

Symbol	Meaning
F_i	Data file
P_i	Policy file
(e_i, n_i)	KGC generated public key
(d_i, n_i)	KGC generated private key
$\{\cdot\}_{KEY}$	Encryption with symmetric key
(pk, sk)	Public private key pair of Us
sig_{KGC}	Authorization generated by Us for Fr using KGC decryption
sig_{cl}	Authorization of download of cloud data generated by Us for Fr
(PK, SK)	Public private key pair of Fr
U-key	Universal serial bus key

3.1. User File Upload Stage

Upon the user's submission of a policy file P_i to the KGC, a request is made for the generation of a public–private key pair. Subsequently, the KGC generates a key pair $(e_i, d_i \equiv e_i^{-1} mod\varphi(n_i))$ specifically linked to the provided policy P_i and dispatches the public key (e_i, n_i) to the user. In the following, for the sake of simplicity, "$modn$" by formula $\lambda = g^{\beta} modn$ is omitted. The user encrypts file F_i with s_i to generate $\{F_i\}_{s_i}$ and generates a random blinding factor r_i, calculates $r_i^{e_i}$, and multiplies it by $s_i^{e_i}$ to obtain $(s_i r_i)^{e_i}$. After this, the user uploads P_i,$(s_i r_i)^{e_i}$,$(F_i)_{s_i}$ to the cloud. Ultimately, the user purges all locally stored keys and files, retaining solely the pertinent policy file P_i and blinding factor R_i within their personal U-key [30].

3.2. Friend Download File Stage

G is a GDH group, and $H : (0,1)^* \to G$ is universal hashing. Us randomly selects sk as its private key in Z_q. Us computes $pk = g^{sk}$ as its public key. Below is the algorithm for friends to download user files.

1. As shown in Figure 2, PK, SK $PK = g^{SK}$, Fr randomly selects R_{KGC}, R_{cl} and then sends a file download request to Us. Fr $\to$ Us : $request(PK, R_{\mathrm{KGC}}, R_{\mathrm{cl}}, P_i)$.
2. After receiving Fr's request, Us will refuse if he does not agree with it; if he agrees to download the file, Us will randomly select x_{KGC}, x_{cl} and calculate

$$y_{\mathrm{KGC}} = g^{x_{\mathrm{KGC}}}, y_{\mathrm{cl}} = g^{x_{cl}}, \tag{1}$$

$$sig_{\mathrm{KGC}} = H(R_{\mathrm{KGC}})^{sk+x_{\mathrm{KGC}}}, \tag{2}$$

$$sig_{cl} = H(R_{cl})^{sk+x_{cl}}. \tag{3}$$

Then,

$$\mathrm{Us} \to \mathrm{KGC} : y_{\mathrm{KGC}}, R_{\mathrm{KGC}}, pk, H(\cdot), \tag{4}$$

$$\mathrm{Us} \to \mathrm{Fr} : sig_{\mathrm{KGC}}, sig_{cl}, E_{PK}(r_i), \tag{5}$$

$$\mathrm{Fr} : D_{SK}(E_{PK}(r_i)) = r_i, \tag{6}$$

where $E_{PK}(\cdot)$ and $D_{PK}(\cdot)$ are IND-CCA secure public key cryptography algorithms.

3. Fr sends a download request to the cloud. Fr $\to$ Cloud : $request(sig_{cl}, P_i)$.
4. The cloud determines whether $(g, H(R_{cl}), y_{cl} pk, sig_{cl})$ is a DH valid Diffie–Hellman tuple. If not, the request is rejected. If the authorization verification is passed, then Cloud $\to$ Fr : $P_i, (s_i r_i)^{e_i}, (F_i)_{s_i}$.
5. Fr sends a decryption request to the KGC. Fr $\to$ KGC : $request(sig_{\mathrm{KGC}}, P_i, (s_i r_i)^{e_i})$.
6. KGC determines whether $(g, H(R_{\mathrm{KGC}}), y_{\mathrm{KGC}} pk, sig_{\mathrm{KGC}})$ is a DH valid Diffie–Hellman tuple.

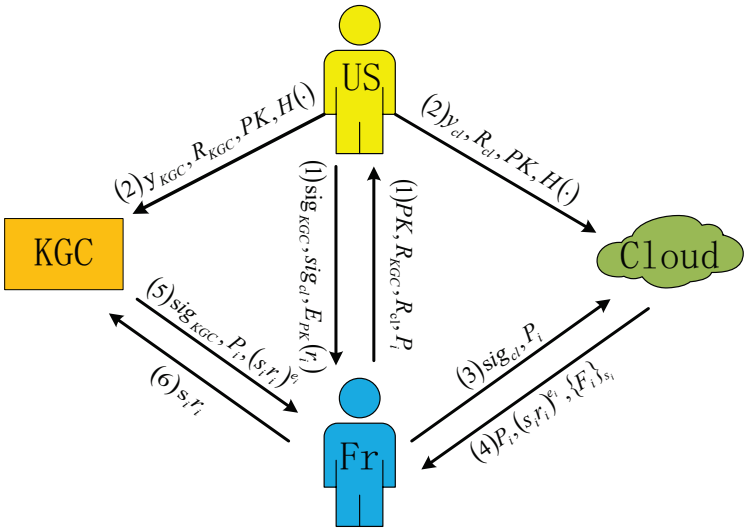

Figure 2. DAC protocol architecture.

If not, the request is rejected. If the authorization verification is passed, the KGC will decrypt $(s_i r_i)^{e_i}$ with private key d_i, and the KGC returns $s_i r_i$ to Fr. Fr uses r_i to decompose s_i, and finally $\left\{\{F_i\}_{s_i}\right\}_{s_i} = F_i$.

4. Protocol Analysis

4.1. Correctness Analysis

The cloud based on $y_{cl}, R_{cl}, pk, H(\cdot)$ from Us and sig_{KGC} from Fr is calculated as

$$\begin{aligned} &(g, H(R_{cl}), y_{cl} pk, sig_{cl}) \\ &= (g, H(R_{cl}), g^{x_{cl}} g^{sk}, H(R_{cl})^{sk+x_{cl}}) \\ &= (g, H(R_{cl}), g^{sk+x_{cl}}, H(R_{cl})^{sk+x_{cl}}). \end{aligned} \tag{7}$$

Thus, $(g, H(R_{cl}), y_{cl} pk, sig_{cl})$ is a DH valid Diffie–Hellman tuple. Authorization sig_{cl} is true.

The KGC based on $y_{cl}, R_{cl}, pk, H(\cdot)$ from Us and sig_{KGC} from Fr is calculated as

$$\begin{aligned} &(g, H(R_{\mathrm{KGC}}), y_{\mathrm{KGC}} pk, sig_{\mathrm{KGC}}) \\ &= (g, H(R_{\mathrm{KGC}}), g^{x_{\mathrm{KGC}}} g^{sk}, H(R_{\mathrm{KGC}})^{sk+x_{\mathrm{KGC}}}) \\ &= (g, H(R_{\mathrm{KGC}}), g^{sk+x_{\mathrm{KGC}}}, H(R_{\mathrm{KGC}})^{sk+x_{\mathrm{KGC}}}), \end{aligned} \tag{8}$$

Thus, $(g, H(R_{\mathrm{KGC}}), y_{\mathrm{KGC}} pk, sig_{\mathrm{KGC}})$ is a DH valid Diffie–Hellman tuple. Authorization sig_{KGC} is true.

4.2. Safety Certification

Theorem 1. *Under the condition of the IF assumption, the file encryption algorithm of this protocol is IND-CCA secure against A_1 intruder attacks.*

Proof. The intruder, the semi-trusted KGC, may intercept P_i, $(s_i r_i)^{e_i}$, $\{F_i\}_{s_i}$ by compromising the communication channel between Fr and the cloud through various attack methods and try to decrypt the user file F_i. It may also intercept $E_{\mathrm{PK}}(r_i)$ by attacking the communication between Fr and Us and try to decrypt it to obtain the user's blinding factor r_i. The following illustrates the two possible cases, respectively. □

Case 1: A_1 attacks the communication between Fr and the cloud.

Using the counter-evidence method, suppose that an IND-CCA intruder A_1 (KGC) breaks the encryption algorithm in the DAC protocol with non-negligible advantage ε. Then, there must be a simulator B_1 to solve the IF problem with non-negligible advantage 2ϵ.

Let $C = (C_1, C_2) = \left((s_i r_i)^{e_i}, \{F_i\}_{s_i}\right)$, and the IND-CCA game of DAC is as follows.

$$\begin{aligned} &Exp_{UCDAC,A_1}^{IND-CCA} \\ &GenUCDAC \rightarrow n_i, e_i, d_i, s_i, r_i, \\ &\quad A_1 \rightarrow (F_{i0}, F_{i1}), \\ &\quad \beta \leftarrow \{0,1\}, C^* = ((s_i r_i)^{e_i}, \{F_{i\beta}\}_{s_i}), \\ &\quad \beta_1 \leftarrow A_1(n_i, e_i, d_i, C^*), \end{aligned}$$

where (n_i, e_i, d_i) is a known variable, while (s_i, r_i) remains unknown. If $\beta_1 = \beta$, 1 is returned; otherwise, 0 is returned. The intruder cannot decrypt the target ciphertext C^*. The advantage of intruder A_1 is defined as

$$Adv_{UCDAC,A_1}^{IND-CCA} = \Pr\{[Exp_{UCDAC,A_1}^{IND-CCA}] = 1)\} = \varepsilon + 1/2.$$

Simulator B_1, equipped with the knowledge of $(n_i, e_i, d_i, \hat{C}_1)$, employs A_1 (attack DAC) as a subroutine to execute the following process with the objective of determining the decomposition factor $\hat{r}_i$ (or $\hat{s}_i$) of $(\hat{C}_1)^{d_i} \bmod n_i$ [31].

1. Randomly select a number $\hat{s}_i$ as an initial estimate for $\frac{(\hat{C}_1)^{d_i} \bmod n_i}{\hat{r}_i}$ (but B_1 does not actually know $\hat{r}_i$); meanwhile, assign (n_i, e_i, d_i) to A_1.
2. s_i inquiry: B_1 generates a list L, where the elements are triples of the form (r_i, C_1, s_i), for which the initial value is $(*, \hat{C}_1, \hat{s}_i)$, with $*$ representing an unknown component, and A_1 is permitted to query L at any time. Upon A_1 querying r_i, B_1 computes $s_i r_i = (C_1)^{d_i} mod n_i$ and provides the subsequent response:
   (a) If there exist items (r_i, C_1, s_i) in L, answer with s_i.
   (b) If there exist items $(*, C_1, s_i)$ in L, answer with s_i and replace $(*, C_1, s_i)$ with (r_i, C_1, s_i) in L.
   (c) Otherwise, randomly select a number s_i, answer with s_i, and store (r_i, C_1, s_i) in the list.
3. Decryption inquiry: When A_1 asks B_1 for $(\overline{C}_1, \overline{C}_2)$, B_1 responds as follows:
   (a) If there is a first term in L, and the second element is $\overline{C_1}$ (the term $(\overline{r_i}, \overline{C}_1, \overline{s_i})$ or $(*, \overline{C}_1, \overline{s}_i)$), then $\{\overline{C}_2\}_{\overline{s}_i}$ is used to answer.
   (b) Otherwise, randomly select a number $\overline{s}_i$, answer with $\{\overline{C}_2\}_{\overline{s}_i}$, and store $(*, \overline{C}_1, \overline{s}_i)$ in L.
4. Challenge: A_1 outputs message F_{i0}, F_{i1}, B_1 randomly selects $\beta \leftarrow \{0,1\}$, calculates $\hat{C}_2 = \{F_{i\beta}\}_{\hat{s}_i}$, and answers A_1 with $(\hat{C}_1, \hat{C}_2)$. Continue to answer A_1.
5. Guess: A_1 outputs guess β_1, and B_1 checks L; if there exist items $(\hat{r}_i, \hat{C}_1, \hat{s}_i)$, then output $\hat{r}_i$.

Let G be the event: When A_1 asks for $\hat{s}_i$ (i.e., $\frac{(\hat{C}_1)^{d_i} \bmod n_i}{\hat{r}_i}$) in the simulation, $\hat{s}_i$ appears in L.

In the event of the aforementioned attack, if $\hat{s}_i$ is not present in L, the entity A_1 is unable to acquire $\hat{s}_i$. Upholding the security considerations surrounding $\hat{C}_2 = \{F_{i\beta}\}_{\hat{s}_i}$, we derive the following:

$$\Pr\{\beta_1 = \beta | \neg G\} = \Pr\{Exp_{UCDAC,A_1}^{IND-CCA} = 1 | \neg G\} = 1/2,$$

where $\neg G$ denotes the complement event of G. In elucidating the definition of A_1 in an actual attack, we can discern that

$$Adv^{DND-CCA}_{UCDAC,A_i} = \Pr([Exp^{ND-CCA}_{UCDAC,A_i}] = 1) = \varepsilon + 1/2$$

$$\begin{aligned} Pr\{Exp^{IND-CCA}_{UCDAC,A_1} = 1\} &= Pr\{Exp^{IND-CCA}_{UCDAC,A_1} = 1|\neg G\}Pr\{\neg G\} \\ &\quad + Pr\{Exp^{IND-CCA}_{UCDAC,A_1} = 1|G\}P\{G\} \\ &\leq \{Exp^{IND-CCA}_{UCDAC,A_1} = 1|\neg G]Pr\{\neg G\} + Pr\{G\} \\ &= 1/2Pr\{\neg G\} + Pr\{G\} \\ &= 1/2(1 - Pr\{G\}) + Pr\{G\} \\ &= 1/2 + 1/2Pr\{G\}. \end{aligned} \tag{9}$$

That is,

$$\varepsilon = \left|\Pr\{Exp^{IND-CCA}_{UCDAC,A_1} = 1\} - \frac{1}{2}\right| \leq \frac{1}{2}\Pr\{G\}$$
$$\Pr\{G\} \geq 2\mathcal{E}.$$

In the aforementioned simulation scenario, the appearance of $\hat{r}_i$ within L is guaranteed with a minimum probability of 2ε. As B_1 meticulously scrutinizes the elements of L in a stepwise fashion during step 5, the success probability of B_1 aligns with $Pr\{G\}$. Consequently, it follows that B_1 effectively solves the IF problem, attaining a non-negligible advantage of 2ε. This circumstance starkly contradicts the inherent difficulty associated with IF. Therefore, the advantage ε attributed to an IND-CCA intruder A_1 in compromising the encryption algorithm is conclusively deemed negligible.

Case 2: A_1 attacks the communication between Fr and Us.

Because $E_{PK}(\cdot)$ is an IND-CCA secure public key cryptography algorithm, the advantage of A_1 breaking through the encryption algorithm can also be ignored.

In conclusion, the encryption algorithm in the DAC protocol is IND-CCA secure, and the theorem is proven.

Theorem 2. *In the case of the IF problem, for an A_2 attack, the file encryption algorithm of this protocol is IND-CCA secure.*

Proof. From Theorem 1, it can be seen that the encryption algorithm in this protocol is IND-CCA secure against semi-trusted KGC (A_1) attacks under the IF assumption.

Moreover, A_2 (semi-trusted cloud) does not know the KGC's private key d_i, and A_2 is more difficult to crack F_1 than A_1. □

Theorem 3. *Let H be a random oracle; if G is a GDH group, then the signature (i.e., authorization) in this protocol has existential unforgeability against chosen message attacks (EUF-CMA).*

Proof. There are two signatures in this protocol, but their construction method is the same. We only need to prove that their signature algorithm has existential unforgeability against chosen message attacks. For convenience and universality, we remove the subscripts KGC and Cl from $x_{cl}, x_{KGC}, y_{cl}, y_{KGC}, R_{cl}, R_{KGC}, sig_{cl}, sig_{KGC}$ and express the signature algorithm as follows:

$$pk = g^s k, y = g^x \tag{10}$$
$$h = H(R), sig = h^s k + x \tag{11}$$

□

Using the method of disproportion: If there is a polynomial time intruder A_3 (semi-trusted Fr) who can attack the authorization algorithm with a non-negligible advantage ε_3, and A_3 can make q_H times H queries at most, then there is a simulator B_3 that can solve the CDH problem with a non-negligible advantage $Adv_{B_3}^{CDH} \approx \frac{\varepsilon_3}{eq_H}$.

Simulator B_3 knows $(g, u = g^{a+b}, h)$ and takes A_3 (attack signature algorithm) as a subroutine, and its goal is to calculate h^{a+b}.

To simplify, and without loss of generality, we assume that

1. A_3 will not ask the random oracle twice;
2. If A_3 requests a signature of message R, it has asked $H(R)$ before;
3. If A_3 outputs (R,sig), it has asked $H(R)$ before.

B_3 regards $u = g^{a+b}$ as its public key, and $a + b$ is the secret key (B_3 does not know $a + b$); then, h^{a+b} is B_3's signature for a message, which is $sig = H(R) = h^{a+b}$, where (R, sig) is forged by A_3. B_3 can take h as the hash value of a message R_J, but B_3 does not know which message A_3 has forged, so it has to guess.

B_3, in order to hide the problem $(g, u = g^{a+b}, h)$, chooses a random number $v \in Z_q$, with $u \cdot g^v$ as the public key to A_3. The reduction process is as follows.

1. B_3 sends the generator g and the public key $u \cdot g^v \in G$ of group G to A_3, where the secret key corresponding to $u \cdot g^v = g^{a+b+v}$ is $a + b + v$. In addition, $J \in \{1, \cdots, q_H\}$ is randomly selected as a guess value; the H inquiry of A_3 corresponds to the final forged result.
2. H inquiry (at most q_H times). B_3 creates a list L_3, the initial value is null, and the element type is quadruple (R_I, w_I, c_I, t_I). When A_3 initiates the I-th inquiry (set the inquiry value as R_I), B_3 answers as follows:
   (a) If there is an item corresponding to R_I in L_3, it will respond with w_I.
   (b) Otherwise, B_3 randomly selects $c_I, t_I \in Z_q$: if $I = J$, then $w_I = hg^{c_I + t_I} \in G$ is calculated; otherwise, calculate $w_I = g^{c_I + t_I} \in G$..

   w_I is used as the response to the query, and (R_I, w_I, c_I, t_I) is stored in the list.
3. Signature inquiry (at most q_H times). When A_3 requests authorization from message R, let I satisfy $R = R_I$. R_I is the query value of the I-th H inquiry. B_3 answers the question as follows:
   (a) If $I \neq J$, then there is a quadruple (R_I, w_I, c_I, t_I) in L_3, and it calculates $sig_I = (ug^v)^{c_I + t_I}$ to answer A_3 for

$$sig_I = (ug^v)^{c_I + t_I} = \left(g^{a+b} g^v\right)^{c_I + t_I} = g^{(c_I + t_I)(a+b+v)} = y_I^{a+b+v} \tag{12}$$

   Thus, sig_I is the signature of secret key $a + b + v$ to R_I.
   (b) If $I = J$, the simulation is interrupted.
4. Output. A_3 outputs (R, sig). If $R \neq R_J$, then B_3 is interrupted; otherwise, B_3 outputs $\frac{sig}{h^v u^{c_J + t_J} g^{c_J v + t_J v}}$ as h^{a+b} for

$$\begin{aligned} sig &= w_J^{a+b+v} = (hg^{c_J + t_J})^{a+b+v} = h^{a+b+v} g^{(c_J + t_J)(a+b+v)} \\ &= h^{a+b} h^v (g^{a+b})^{(c_J + t_J)} g^{(c_J + t_J)v} = h^{a+b} h^v u^{c_J + t_J} g^{c_J v + t_J v} \end{aligned} \tag{13}$$

If B_3's guess is correct and A_3 outputs a forgery, then B_3 solves the CDH problem in step 4).

The success of B_3 is determined by the following three events:

(a) E_1:B_3 will not be interrupted in the signature inquiry of A_3;
(b) E_2:A_3 generates a valid message signature pair (R, sig);

(c) E_1:E_2 occurs and the subscript of the corresponding quadruple (R_I, w_I, c_I, t_I) of R is $I = J$.

$$\Pr\{E_1\} = (1 - \frac{1}{q_H})^{q_H}, \Pr\{[E_2 \mid E_1]\} = \mathcal{E}_3, \tag{14}$$

$$\Pr\{[E_3 \mid E_1E_2]\} = \Pr\{[I = J \mid E_1E_2]\} = \frac{1}{q_H}, \tag{15}$$

$$\begin{aligned} Adv_{B_3}^{CDH} &= Pr\{[E_1E_2E_3]\} \\ &= Pr\{[E_1]Pr\{[E_2|E_1]\}\}Pr\{[E_3|E_1E_2]\} \\ &= (1 - \frac{1}{q_H})^{q_H} \cdot \varepsilon_3 \cdot \frac{1}{q_H} \\ &\approx \frac{\varepsilon_3}{eq_H}. \end{aligned} \tag{16}$$

B_3 can solve the CDH problem with a non-negligible advantage $Adv_{B_3}^{CDH} \approx \frac{\varepsilon_3}{eq_H}$, which contradicts the difficulty of the CDH problem. Therefore, the advantage ε_3 of the polynomial time intruder A_3 to break the signature (authorization) algorithm is negligible, and the theorem is proven.

4.3. Scyther-Based Validation

We use the formal method, namely the Scyther platform, to simulate the proposed solution. The access control (AC) of the proposed DAC is executed through the Security Protocol Description Language (SPDL), similar to [32]. Three core roles, namely SR and CR for users and cloud servers, are described in the SPDL script. Scyther verifies all the claims as shown in Table 4. In addition, Scyther validates manual claims such as the claim of the user role (SR, Secret, SE) and the claim of the cloud server role (CR, Secret, SE). Furthermore, Scyther verifies the automatically generated user role claims, namely claim (SR, Alive), claim (SR, Niagree), and claim (SR, Nisynch). In the same way, Scyther verifies the claims made for the cloud server role, namely claim (CR, Secret, SE), claim (CR, Alive), claim (CR, Niagree), and claim (CR, Nisynch).

Table 4. Results generated using Scyther.

Claims	**For User/Smart U_j/SR**	**Attack Status**
Claim-a	claim (SR, Secret, SKe)	No attack found
Claim-b	claim (SR, Alive)	No attack found
Claim-c	claim (SR, Niagree)	No attack found
Claim-d	claim (SR, Nisynch)	No attack found
Claims	**For Cloud Server (CR)**	**Attack Status**
Claim-a	claim (CR, Secret, SKe)	No attack found
Claim-b	claim (CR, Alive)	No attack found
Claim-c	claim (CR, Niagree)	No attack found
Claim-d	claim (CR, Nisynch)	No attack found

5. Analytical Results

5.1. Security Comparison

In this section, we conduct an assessment of the proposed DAC mechanism with a focus on its security features. Our evaluation encompasses the security frameworks introduced by Haleh, Li, Tiwari, Zahid, and Tanveer. The accompanying Table 5 delineates the security attributes offered by DAC in comparison to other relevant security frameworks. Notably, Haleh's security framework exhibits a limitation in ensuring user anonymity during the authentication phase. Tiwari's framework, on the other hand, falls short in preventing server/user impersonation attacks. In contrast, DAC not only furnishes a robust

mechanism for the authentication of users and cognitive radios (CRs) but also ensures the establishment of a secure session key. Furthermore, DAC delivers a secure protocol for both the sharing and uploading of data.

Table 5. Comparison of security features.

Framework	Sec1 [1]	Sec2 [2]	Sec3 [3]	Sec4 [4]	Sec5 [5]
Haleh [33]	×	✓	✓	✓	✓
Li [34]	✓	✓	✓	✓	✓
Tiwari [35]	✓	×	×	✓	✓
Zahid [36]	✓	✓	✓	✓	✓
Tanveer [32]	✓	✓	✓	✓	✓
The proposed	✓	✓	✓	✓	✓

[1] Anonymity/untraceability. [2] User impersonation. [3] Server impersonation attack. [4] Data uploading/storage phase. [5] Data sharing phase. ✓ Feature is available and supported. × Feature is not supported.

5.2. Performance Evaluation

We simulated the protocol using two servers as the cloud storage and key generation center, deployed in a university, and using two laptops as users and friends, deployed in two different families. Among them, the configuration of the cloud server is as follows: 16-core CPU, 64 GB memory, 8 TB storage. The performance parameters for the key manager server KM are as follows: 32-core CPU, 128 GB memory, 1 TB storage, IP is 59.69.208.202. The hardware configurations of these two laptops are as follows: Intel Core i3-4030U processor, 4 GB memory, mechanical hard disk, US and Fr's home network bandwidth is 15 M. This program selects files of 0.1 MB, 0.5 MB, 1 MB, 10 MB, 50 MB, 100 MB, 500 MB for uploading and downloading. Figures 3 and 4 are the simulation diagrams created by MATLAB.

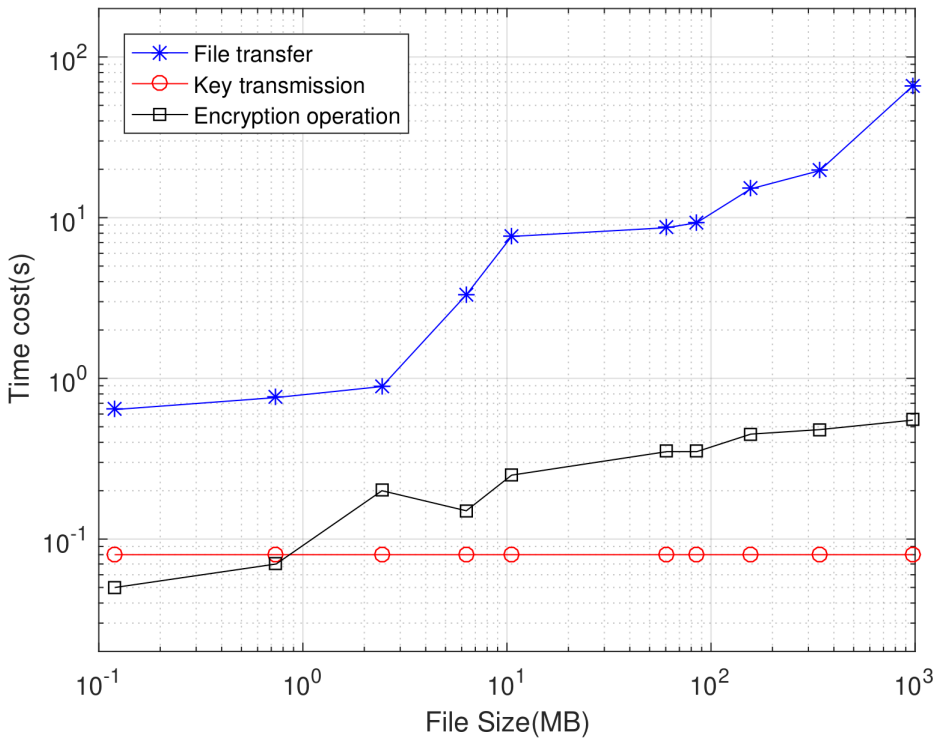

Figure 3. Operation time cost of user file upload phase.

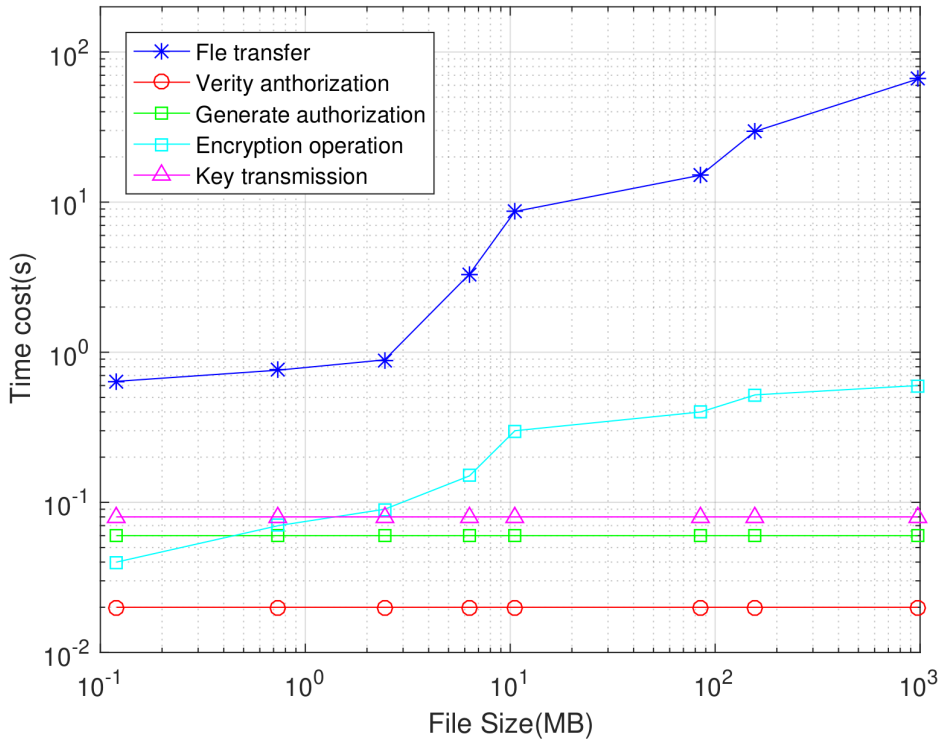

Figure 4. Operation time cost of friend file download phase.

As can be seen from Figure 3, in the user file upload stage, with the increase in the file size, the file transmission time is increasing, but the key transmission and encryption operation time is almost always below 0.5 s, which remains unchanged or changes little. As can be seen from Figure 4, in the friend file download stage, with the increase in the file size, the file transfer time is increasing, so that the last time is more than 30 s, but the generation authorization time, verification authorization time, key transfer time, and decryption operation time are almost always less than 0.5 s, which remains unchanged or changes little.From Figures 3 and 4, it can be seen that there are slight differences in the file transfer time, key transfer time, and encryption operation time between the user file upload stage and the user download stage at the same file size.

The simulation results show that the time cost of the protocol is very small (such as generation authorization, verification authorization, key transmission, encryption/decryption operation), and it changes little with the increase in the files. The main time cost of this protocol is the file transmission time (that is, the time cost of the existing cloud storage applications), which is compared with the existing cloud storage applications, and the user experience is similar.

We have conducted a comparative analysis of the DAC methodology with the schemes delineated in [7,9,11]. This comparison hinges on the time intervals required for both the file upload and file download phases. The results presented in Table 6 underscore the superiority of the DAC protocol over other schemes in both the file upload and file download phases. In Table 6, "UL" signifies the upload phase, "DL" designates the download phase, and the temporal unit is seconds.

DAC is an improved protocol based on the introduction of a semi-trusted KGC to existing cloud storage applications (such as cloud disks). Compared with the existing cloud storage applications, it adds some cryptographic operations, which can more safely realize the access control of users to their files, thus improving the security of the protocol. Therefore, this protocol has good theoretical and application value and can provide a meaningful reference for the upgrade of the current and most popular cloud storage services.

Table 6. Comparison of turnaround times.

File Size (MB)	[14]		[15]		[18]		DAC	
	UL	DL	UL	DL	UL	DL	UL	DL
0.1	1.48	1.15	1.4	0.99	0.80	0.80	0.53	0.55
0.5	1.89	1.31	1.48	1.03	0.94	0.96	0.74	0.82
1	2.90	1.85	2.06	1.48	1.24	1.18	1.10	1.17
10	14.59	10.45	14.95	9.90	6.43	6.48	5.37	5.61
50	60.37	35.90	58.56	35.57	9.01	10.24	7.69	8.53
100	155.15	61.59	112.41	59.14	17.37	20.68	14.03	17.53
500	872.09	400.21	492.03	229.81	33.24	39.25	30.35	33.11

6. Conclusions

We introduce the DAC protocol as an innovative cloud storage security solution tailored to group data. This protocol not only ensures data confidentiality but also facilitates secure data sharing and incorporates an array of comprehensive access control features. Moreover, the protocol innovatively incorporates a semi-trusted third party to effectively mitigate security threats stemming from the key generation center. Notably, we conducted a rigorous security analysis of the protocol and assessed its performance by evaluating the time consumption for file upload and download operations. The outcomes of these evaluations affirm the applicability of the DAC protocol in the realm of cloud computing, particularly in the context of safeguarding users' private data storage, demonstrating its strong practicality and operational efficiency. In subsequent developments, our goals center around enhancing the protocol's efficiency with regard to communication, computing, and storage costs, all while upholding an unaltered standard of security.

Author Contributions: Conceptualization, methodology, formal analysis, and investigation: J.Z. and P.Z.; writing—original draft preparation: A.C.; writing—review and editing: J.Z. and A.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Key Science and Technology Research Project of Henan Province of China (Grant No. 222102210053); the Key Scientific Research Project in Colleges and Universities of Henan Province of China (Grant No. 21A510003); and the Major Science and Technology Projects of Longmen Laboratory (Grant Nos. 231100220400, 231100220300).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nivedhaa, R.; Justus, J. A Secure Erasure Cloud Storage System Using Advanced Encryption Standard Algorithm and Proxy Re-Encryption. In Proceedings of the 2018 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 3–5 April 2018; pp. 755–759. [CrossRef]
2. Singh, P.; Saroj, S.K. A secure data dynamics and public auditing scheme for cloud storage. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 695–700.
3. Sengupta, B.; Nikam, N.; Ruj, S.; Narayanamurthy, S.; Nandi, S. An Efficient Secure Distributed Cloud Storage for Append-Only Data. In Proceedings of the 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2–7 July 2018; pp. 146–153. [CrossRef]
4. Ali, M.; Bilal, K.; Khan, S.U.; Veeravalli, B.; Li, K.; Zomaya, A.Y. DROPS: Division and Replication of Data in Cloud for Optimal Performance and Security. *IEEE Trans. Cloud Comput.* **2018**, *6*, 303–315. [CrossRef]
5. Fatemi Moghaddam, F.; Ahmadi, M.; Sarvari, S.; Eslami, M.; Golkar, A. Cloud computing challenges and opportunities: A survey. In Proceedings of the 2015 1st International Conference on Telematics and Future Generation Networks (TAFGEN), Kuala Lumpur, Malaysia, 26–28 May 2015; pp. 34–38. [CrossRef]
6. Yuefei, Z.; Bin, L. Research and development of data storage security audit in cloud. *Comput. Sci.* **2020**, *47*, 290–300.

7. Li, L.; An, X. Research on Storage Mechanism of Cloud Security Policy. In Proceedings of the 2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS), Hunan, China, 10–11 August 2018; pp. 130–133. [CrossRef]
8. Markandey, A.; Dhamdhere, P.; Gajmal, Y. Data Access Security in Cloud Computing: A Review. In Proceedings of the 2018 International Conference on Computing, Power and Communication Technologies (GUCON), Greater Noida, India, 28–29 September 2018; pp. 633–636. [CrossRef]
9. Mogarala, A.G.; Mohan, K.G. Security and Privacy Designs Based Data Encryption in Cloud Storage and Challenges: A Review. In Proceedings of the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 10–12 July 2018; pp. 1–7. [CrossRef]
10. Shaik, N.S.; Ketepalli, G.; Reddy, V.N.; Reddy, T.M.K. Cryptograhy and Pk-Anonymization Methods for Secure Data Storage in Cloud. In Proceedings of the 2019 Third International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 12–14 December 2019; pp. 472–477. [CrossRef]
11. Vora, A.V.; Hegde, S. Keyword-based private searching on cloud data along with keyword association and dissociation using cuckoo filter. *Int. J. Inf. Secur.* **2019**, *18*, 305–319. [CrossRef]
12. Kodumru, N.L.; Supriya, M. Secure Data Storage in Cloud Using Cryptographic Algorithms. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 16–18 August 2018; pp. 1–6. [CrossRef]
13. Wang, H.; He, D.; Han, J. VOD-ADAC: Anonymous Distributed Fine-Grained Access Control Protocol with Verifiable Outsourced Decryption in Public Cloud. *IEEE Trans. Serv. Comput.* **2020**, *13*, 572–583. [CrossRef]
14. Rafique, F.; Obaidat, M.S.; Mahmood, K.; Ayub, M.F.; Ferzund, J.; Chaudhry, S.A. An Efficient and Provably Secure Certificateless Protocol for Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8039–8046. [CrossRef]
15. Seo, S.H.; Nabeel, M.; Ding, X.; Bertino, E. An Efficient Certificateless Encryption for Secure Data Sharing in Public Clouds. *IEEE Trans. Knowl. Data Eng.* **2014**, *26*, 2107–2119. [CrossRef]
16. Ali, M.; Malik, S.U.R.; Khan, S.U. DaSCE: Data Security for Cloud Environment with Semi-Trusted Third Party. *IEEE Trans. Cloud Comput.* **2017**, *5*, 642–655. [CrossRef]
17. Akhila, M.; Hemalatha, E.; Parvathi, S.; Karthikeyan, L. Data security in cloud using semi trusted third party key manager. *Int. J. Sci. Res. Sci. Technol.* **2016**, *2*, 111–113.
18. Han, S.; Han, K.; Zhang, S. A Data Sharing Protocol to Minimize Security and Privacy Risks of Cloud Storage in Big Data Era. *IEEE Access* **2019**, *7*, 60290–60298. [CrossRef]
19. Bian, G.; Chang, J. Certificateless Provable Data Possession Protocol for the Multiple Copies and Clouds Case. *IEEE Access* **2020**, *8*, 102958–102970. [CrossRef]
20. Ben Daoud, W.; Rekik, M.; Meddeb-Makhlouf, A.; Zarai, F.; Mahfoudhi, S. SACP: Secure Access Control Protocol. In Proceedings of the 2021 International Wireless Communications and Mobile Computing (IWCMC), Harbin City, China, 28 June–2 July 2021; pp. 935–941. [CrossRef]
21. Thakur, G.; Kumar, P.; Deepika.; Jangirala, S.; Das, A.K.; Park, Y. An Effective Privacy-Preserving Blockchain-Assisted Security Protocol for Cloud-Based Digital Twin Environment. *IEEE Access* **2023**, *11*, 26877–26892. [CrossRef]
22. Singh, D.; Chitkara, M. Advanced Privacy-Aware Protocol Placement in Cloud Security. In Proceedings of the 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), Ballar, India, 29–30 April 2023; pp. 1–5. [CrossRef]
23. Gundale, M.; Mishra, A. Security Models of cloud computing using Machine Learning Network Security Application. In Proceedings of the 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), Greater Noida, India, 28–30 April 2023; pp. 340–346. [CrossRef]
24. Kaur, M.; Kaimal, A.B. Analysis of Cloud Computing Security Challenges and Threats for Resolving Data Breach Issues. In Proceedings of the 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 23–25 January 2023; pp. 1–6. [CrossRef]
25. Mishra, S.; Chitkara, M. Service Level Trust Key Encryption based Cloud Security using Starvation End-Point Encryption. In Proceedings of the 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS), Raichur, India, 24–25 February 2023; pp. 1–5. [CrossRef]
26. Ali, M.; Dhamotharan, R.; Khan, E.; Khan, S.U.; Vasilakos, A.V.; Li, K.; Zomaya, A.Y. SeDaSC: Secure Data Sharing in Clouds. *IEEE Syst. J.* **2017**, *11*, 395–404. [CrossRef]
27. Kumar, V.; Mohammed Ali Al-Tameemi, A.; Kumari, A.; Ahmad, M.; Falah, M.W.; Abd El-Latif, A.A. PSEBVC: Provably Secure ECC and Biometric Based Authentication Framework Using Smartphone for Vehicular Cloud Environment. *IEEE Access* **2022**, *10*, 84776–84789. [CrossRef]
28. Boneh, D.; Lynn, B.; Shacham, H. Short signatures from the Weil pairing. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, 9–13 December 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 514–532.
29. Dhakad, N.; Kar, J. EPPDP: An Efficient Privacy-Preserving Data Possession With Provable Security in Cloud Storage. *IEEE Syst. J.* **2022**, *16*, 6658–6668. [CrossRef]
30. Gupta, M.; Kumar, B.S. Lightweight Secure Session Key Protection, Mutual Authentication, and Access Control (LSSMAC) for WBAN-Assisted IoT Network. *IEEE Sens. J.* **2023**, *23*, 20283–20293. [CrossRef]

31. Xu, S.; Han, X.; Xu, G.; Ning, J.; Huang, X.; Deng, R.H. An Adaptive Secure and Practical Data Sharing System with Verifiable Outsourced Decryption. *IEEE Trans. Serv. Comput.* **2023**, 1–13. [CrossRef]
32. Tanveer, M.; Bashir, A.K.; Alzahrani, B.A.; Albeshri, A.; Alsubhi, K.; Chaudhry, S.A. CADF-CSE: Chaotic map-based authenticated data access/sharing framework for IoT-enabled cloud storage environment. *Phys. Commun.* **2023**, *59*, 102087. [CrossRef]
33. Amintoosi, H.; Nikooghadam, M.; Kumari, S.; Jun, F.; Xiong, H.; Kumar, S.; Rodrigues, J.J.P.C. Secure and Authenticated Data Access and Sharing Model for Smart Wearable Systems. *IEEE Internet Things J.* **2022**, *9*, 5368–5379. [CrossRef]
34. Li, Q.; Ma, J.; Li, R.; Liu, X.; Xiong, J.; Chen, D. Secure, efficient and revocable multi-authority access control system in cloud storage. *Comput. Secur.* **2016**, *59*, 45–59. [CrossRef]
35. Tiwari, D.; Chaturvedi, G.K.; Gangadharan, G. ACDAS: Authenticated controlled data access and sharing scheme for cloud storage. *Int. J. Commun. Syst.* **2019**, *32*, e4072. [CrossRef]
36. Ghaffar, Z.; Ahmed, S.; Mahmood, K.; Islam, S.H.; Hassan, M.M.; Fortino, G. An Improved Authentication Scheme for Remote Data Access and Sharing Over Cloud Storage in Cyber-Physical-Social-Systems. *IEEE Access* **2020**, *8*, 47144–47160. [CrossRef]